Verdi® and Siloti® Tcl Reference

Version O-2018.09-SP2, March 2019

SYNOPSYS®

Copyright Notice and Proprietary Information

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys company and certain product names are trademarks of Synopsys, as set forth at http://www.synopsys.com/company/legal/trademarks-brands.html.
All other product or company names may be trademarks of their respective owners.

Free and Open Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

www.synopsys.com

Contents

Introduction

Overview

The *Verdi* platform supports a command language interface for Tk and C based applications. An overview of the Verdi Command Language Architecture is shown in the following figure:

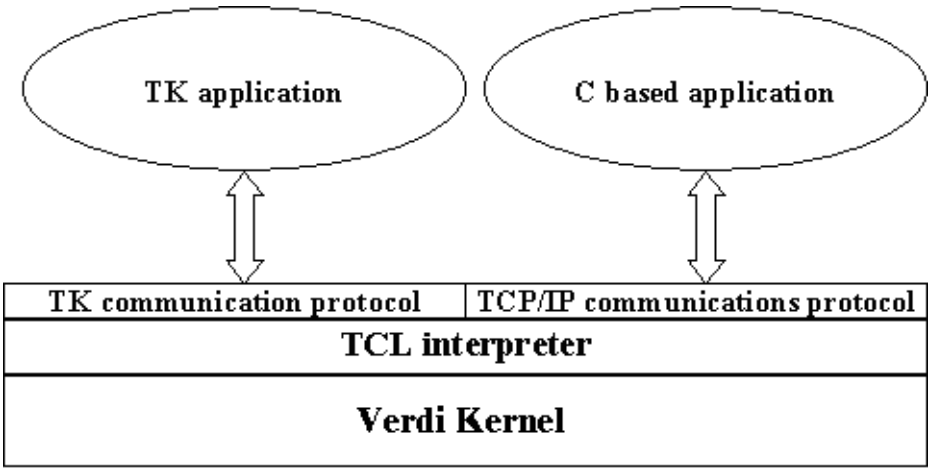


Figure: Verdi Command Language Architecture

With Verdi's embedded Tcl interpreter, the Verdi platform can communicate with Tk and C based applications using Tk and TCP/IP communication protocol respectively.

The communication methods for Tk command client and socket command client are described in detail in the following section. The tutorials and references are also provided in the following chapters.

Tk Command Client

With the Tk send commands, the Verdi platform can easily communicate with other Tcl/Tk applications. Topics include the following:

- Invoking the Verdi platform from another Tk application
- Invoking commands from another Tk application (Tk send command)
- Adding event Callbacks to your Tk application (Verdi event Callbacks - Tk send)
- Replaying commands (playback)
- Tk commands
- Debugging

The following figure depicts the Verdi Command Language Interface with Tk applications (Tk command clients):

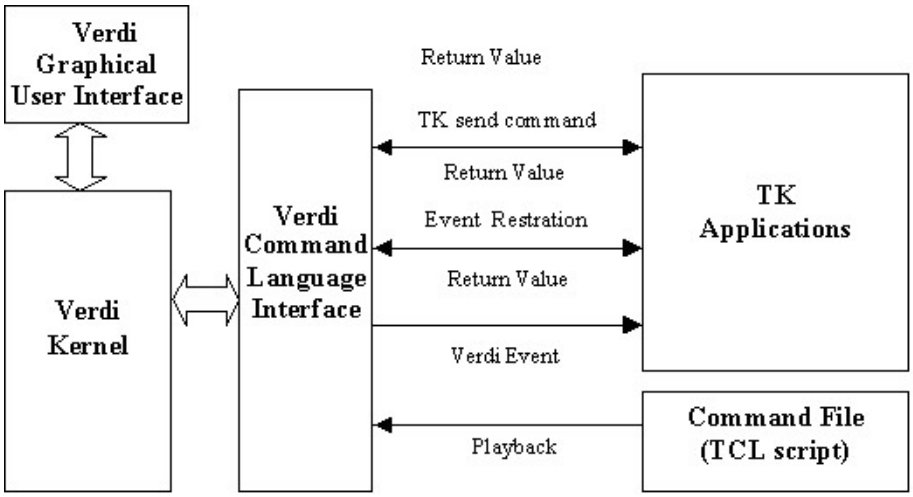


Figure: Verdi Command Language Interface with Tk applications

Invoking Verdi from Another Tk Application

Commands cannot be sent as soon as the Verdi platform is invoked. The registration of commands and Tk name must be completed before proceeding. Therefore, use the following polling mechanism to check on the status of the registration process. For example:

```
exec dubussy -tkName <tk name goes here> &
while {1} {
```

```
set s [winfo interps]
if {-1 != [string first <tk name goes here> $s]} {
break;
}
after 500
}
send <tk name goes here> {debRestoreSession mySession.ses}
...
```

Invoking Commands from Another Tk Application

To invoke Verdi commands from another Tk application, a Tk name for the Verdi platform must be registered with the `-tkName` command line option. The syntax for registering the Verdi platform's Tk name is as follows:

```
>Verdi -tkName <Tk name>
```

<Tk name>: Choose any name for the Tk name. However, if another Tk application already has the same name, then a sequential numeric value is appended to the Tk name so that it is unique. For example, "test#2" is assigned for the second Verdi Tk name of "test" when there is already an existing Verdi platform with Tk name "test".

To send a command to the Verdi platform, use the Tk `send` command. The syntax of the send command is shown below:

```
send <Tk name> <Verdi's Command Reference>
```

NOTE: If a command was successfully executed by the Verdi platform, a value of *TCL_OK* is returned to the calling Tcl/Tk application; otherwise, the return value is *TCL_ERROR*. Each command expects a certain number and type of arguments. Upon receiving a command, the Verdi platform checks the command arguments; if this check fails, the command is not executed.

Adding Event Callbacks to Tk Applications

The Verdi platform can call back the invoking Tk application when specific events occur. This is done by sending a command to the Verdi platform which registers a callback function along with a reason used to determine when to call the callback function. A special command `AddEventCallback` is used to set up the callback information.

NOTE: Tk applications do not accept commands that use `xhost` style security protection. For this reason, use `xauth` instead.

The exception to the `xhost` rule is when the Tk application and the Verdi platform are running on the same host and both are displayed to an xserver running on that host. In this situation, the `xhost -` command can be used to enable access control. This allows the Tk application to receive event notifications. This also applies when using VNC (Virtual Networking Computing) since the VNC server (which is a modified xserver) is running on the same host as both the Tk application and the Verdi platform. This scheme, however, does not work if the host is a UNIX server and the display is configured to an xserver running on another machine such as a PC xserver.

An easier alternative is to compile Tk with security turned *off*. This way it does not matter which security scheme is used. The Tk application always receives event notifications.

AddEventCallback

Syntax

```
AddEventCallback TkAppName CallbackFun Reason async
```

Arguments

`TkAppName`

This is the name of the Tk application (which is usually the name appearing in the title). The Verdi platform uses this name to identify the Tcl interpreter to send commands to. If the Tk application is running under the control of a script file, it is usually the name of the script file.

The following Tk commands can be used to get the application name:

```
winfo name
winfo interps
selection get APPLICATION
```

`CallbackFunc`

This is the Tk callback command/procedure.

`Reason`

This is the reason for the callback. The Verdi platform invokes the callback when the specified reason occurs. Refer to the Reason Table for details.

`Async`

In most cases, 1 should be specified for this parameter. Synchronous event callbacks should be used in special situations only where the server should wait for the client to process the event before continuation (for example, before exit). If not used carefully, the server (Verdi) may hang with synchronous events.

RemoveEventCallback

Description

Call this function to remove a previously installed event handler. To remove an event handler, *TkAppName*, *CallbackFunc*, and *Reason* fields must match the original value passes to *AddEventCallback* when registering for the event.

Syntax

```
RemoveEventCallback TkAppName CallbackFunc Reason
```

Arguments

`TkAppName`

This is the name of the Tk application. The Verdi platform uses this name to identify the Tcl interpreter to send the commands to. If the Tk application is running under the control of a script file, it is usually the name of the script file.

`CallbackFunc`

This is the original Tk callback command/procedure. It should match the name of the callback function passed to *AddEventCallback*.

`Reason`

This is the original reason for the callback. It should match the name of the reason passed to *AddEventCallback*.

Reason Table

System

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
AllEvents	This is a special event reason, which if registered causes all events to be sent to the client application. There is no *AllEvents* reason that is sent to the client application. It is simply a convenient way to register for all events at one time. See the reason table for a description of each individual event and its calldata.	None
debCursorTimeChange	Sent when the Verdi cursor time is changed.	new cursor time
debDndSrc	Sent when a drag operation starts.	The name of the source window (that is, *nTrace* and *nWave*).
debDndDst	Sent on a drop operation.	The name of the target window (that is, *nTrace* and *nWave*).
debExit	Sent when the Verdi platform exits.	None
debPreferenceAction	Sent for the action of the preference form in the Verdi platform.	OK | Cancel | Apply
debRestoreSession	Sent when the Verdi platform restores the debugging session from the file.	Session file name
debSaveSession	Sent when the Verdi platform saves the debug session.	Session file name
debSimResultClose	Sent when the Verdi platform closes the simulation result.	Simulation file name

debSimResultOpen	Sent when the Verdi platform loads the simulation result.	Simulation file name
debSwitchInterMode	Sent when the Verdi platform interactive mode is turned *on* or *off*.	On | Off
iscCommand	Sent when the Verdi platform sends commands to simulator. When the iscCommand reason is registered, the Verdi platform does not send the commands typed by you to simulator. It does callback instead and sends the command string as call data.	Command string
iscOutput	Sent when there is any simulator output. Simulator still prints the output to the message window no matter the reason is registered or not.	simulator output string.
guiInfoMessage	Sent when information message is displayed.	message body
guiWarnMessage	Sent when warning message is displayed.	message body
guiXMap	Sent when a top level window is mapped (made visible).	The X Window id of the topmost shell. guiXMap -xid shell_id
guiXUnmap	Sent when a top level window is unmapped (removed from screen).	The X Window id of the topmost shell. guiXUnmap -xid shell_id

Event

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
tsCreateWindow	Sent when a new *Temporal Flow View* window is created.	window id
tsCloseWindow	Sent when a new *Temporal Flow View* window is closed.	window id

nTrace

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
srcCreateWindow	Sent when a new *nTrace* window is created.	window id
srcCloseWindow	Sent when a *nTrace* window is closed.	window id
srcCloseFSDBFile	Sent when the Verdi platform closes an FSDB file.	FSDB file name
srcChangeScope	Sent when *nTrace* changes current scope.	new scope name
srcDblClkObj	Sent when you double-click on *nTrace*.	object type, object name object type = wire, reg, module, instance, task, function, parameter, entity, architecture, signal, variable, procedure, port, generic
srcDragOVASignal	Sent when the OVA signal is dragged.	DragOVASignal
srcDragSVASignal	Sent when the SVA signal is dragged.	DragSVASignal
srcImportDesign	Sent when the Verdi platform imports design.	top module (Verilog), entity/ configuration (VHDL)
srcOpenFSDBFile	Sent when the Verdi platform loads an FSDB file.	FSDB file name

srcPaldDClickCB	Sent when double-clicking the source line number area.	[-file *fileName*] [-line *lineNo*] [-scope *scopeName*] **Example:** srcPaldDClickCB -file "test.v" -line "10" -scope "system"
srcReloadDesign	Sent when the Verdi platform reloads design.	top module (Verilog), entity/ configuration (VHDL)
srcRMBClkObj	Sent when you click on an instance or signal with right mouse button, before the context menu is popped up.	win_ptr{obj{obj_path}} where *win_ptr*: source window pointer, in hex format. *obj*: INST|SIGNAL *obj_path*: full hierarchy path of the object.
srcSimContinueCB	Sent when the Verdi platform continues the simulation in interactive mode.	None
srcSimFinishCB	Sent when the Verdi platform finishes the simulation in interactive mode.	None
srcSimNextEventCB	Step to the next executed statement at the current simulation time.	None
srcSimNextTimeCB	Sent when the Verdi platform goes to the next simulation time step in interactive mode.	None
srcSimRunCB	Sent when the Verdi platform runs the simulation in interactive mode.	None
srcSimStopCB	Sent when the Verdi platform stops the simulation in interactive mode.	Non
srcSimTraceTimeCB	Sent when the Verdi platform trace time is changed in interactive mode.	srcSimNextTimeCB -time *timeValue* **Example** srcSimTraceTimeCB -time "300"

swnTraceWindowType	Sent when you switch between *nTrace* and *nSchema* window.	-win: *nTrace*/*nSchema*
swnTraceViewType	Sent when you switch between source code view and schematic view in *nTrace*.	-win: Source/Schematic
tbDebugRunSim	Run the simulator.	None
tbDebugSimQuit	Terminate the simulator.	None
tbDebugStepNext	Step to the first executed statement of the next simulation time.	None
wtchAddSignal	Notify that signals have been added to the watch window.	-tab tabNumber -index rowNumber -signal signalPath -value valueString where *-tab tabNumber*: The tab in watch window, starts from 1. *-index rowNumber*: The position (row number) of the added signal, starts from 1. *-signal signalPath*: The signal's full path. *-value valueString*: The value string of the signal.
wtchCloseWindow	Notify that the watch window has been closed.	None
wtchDelSignal	Notify that signals have been deleted from the register window.	-tab tabNumber (-index rowNumber -signal signalPath) | (-all) where *-tab tabNumber*: The tab in watch window, starts from 1. *-index rowNumber*: The position (row number) of the deleted signal, starts from 1. *-signal signalPath*: The signal's full path. *-all*: Indicate that all the signal in the specified tab are deleted.
wtchOpenWindow	Notify that the watch window has been opened.	None

wtchSignalValueChange	Notify that signal value has changed.	-tab tabNumber -index rowNumber -signal signalPath -value valueString where *-tab tabNumber*: The tab in watch window, starts from 1. *-index rowNumber*: The position (row number) of the added signal, starts from 1. *-signal signalPath*: The signal's full path. *-value valueString*: The value string of the signal.
wtchTimeChange	Notify that the watch window time has changed.	*time* where *time*: New time of the register window.

nSchema

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
schAddObj	Sent when adding the instance, port, or signal to *nShema*.	-win [winId] objType [addInst | -addPort | -addSignal]
schChangeScope	Sent when *nSchema* changes current scope.	window_id new_scope_name
schCloseWindow	Sent when an *nSchema* window is closed.	window id
schCreateWindow	Sent when a new *nSchema* window is created.	window id
schDblClkObj	Sent when you double-click on *nSchema*.	window_id {object type (net, port or instance), object name}
schGetClockSource	Sent when executing the Source menu commands for tracing clock.	-win [winId] objType {[SIGNAL | INST | INSTPORT | INSTPPIN | PORT]}
schPreSelectObj	Sent when moving the cursor to instance, port, or signal on *nSchema*.	-win [winId] objType [-inst | -port | -signal]
schRMBClkObj	Sent when you click on an instance or signal with right mouse button, before the context menu is displayed.	win_ptr{obj{obj_path}} where *win_ptr*: source window pointer, in hex format. *obj*: INST|SIGNAL *obj_path*: full hierarchy path of the object.
schSelectObj	Sent when selecting an object in *nSchema*.	window id
schUnselectObj	Sent when unselecting an object in *nSchema*.	window id

nWave

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
wvAddMarker	Sent when *nWave* adds a new marker.	marker name, time
wvAddSignal	Sent when any signal adds to signal window.	delimiter, -clear (optional), #signal, signal_name_list
wvAddSignalByIdCode	Sent when a signal is added to the waveform.	signal id list, -clear (optional), #signal
wvCloseFSDBFile	Sent when *nWave* closes an FSDB file.	FSDB file name
wvCloseWindow	Sent when *nWave* window is closed.	window id
wvCreateWindow	Sent when a new *nWave* window is created.	window id
wvCursorTimeChange	Sent when the cursor clicks on *nWave*.	time, signal name, FSDB file name
wvDblClkActiveTrace	Active trace driver information sent when a signal is double-clicked.	signal time value fileName lineNumber variableName activeDriverNumber {activeDriverInfo1} {activeDriverInfo2}
wvDblClkWaveForm	Sent when double-clicking the signal in the *nWave* window.	signal name, time, value **NOTE:** When double-clicking the property signal in the *nWave* window, the calldata will include the start time and end time. For example, system.i_cpu.aa.call_t2_qa_t2 {500,700,failure} {500,800,success})
wvDeleteMarker	Sent when *nWave* deletes a marker.	marker name
wvMarkerTimeChange	Sent when marker time changes.	time, signal name, FSDB file name

wvOpenFSDBFile	Sent when *nWave* loads an FSDB file.	FSDB file name
wvRMBClickSignalPane	Sent when you right-click on a signal name in *nWave*'s signal pane.	signal {signal_name}
wvSaveEvent	Sent when you save events/complex events to a file.	file name
wvSelectAllSignals	Sent when you select all signal names in *nWave*'s signal pane.	signal_count
wvSelectSignal	Sent when you select a signal name in *nWave*'s signal pane.	signal_count {signal_name1} {signal_name2}
wvWindowTimeUnitChanged	Sent when time unit changes.	time unit
wvZoom	Sent when *nWave* zooms its view range.	start time, end time

nState

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
fsmCloseWindow	Sent when an FSM window is closed.	window id
fsmCreateWindow	Sent when a new FSM is created.	window id
fsmDblClkObj	Sent when you double-click on an FSM window.	object type, object name object type = state, transition, action, port

nCompare

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
cmpExit	Sent when *nCompare* exits.	None

Transaction Table View

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
stsCloseTable	Sent when the Verdi platform closes a new table in the *Transaction Statistic* window.	Void* table_handle
stsCreateTable	Sent when the Verdi platform creates a new table in the *Transaction Statistic* window.	Void* table_handle
taAddStream	Sent when any stream was added.	*streamNameList*
taCloseFSDBFile	Sent when the *Analysis* window closes an FSDB file.	FSDB file name
taCloseWindow	Sent when an *Analysis* window is closed.	Window ID
taCreateWindow	Sent when a new *Analysis* window is created.	Window ID
taCursorTimeChange	Sent when cursor time changes.	-time *time*

taMarkerTimeChange	Sent when marker time changes.	-time *time*
taOpenFSDBFile	Sent when the *Analysis* window loads an FSDB file.	FSDB file name

nRegister

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
regAddSignal	Notify that signals have been added to the register window.	-win window -point "x,y" {signal1} {signal2} ... {signaln} where -win *window*: window id -point "x,y": The upper left coordinate of singal(s) box.
regCloseWindow	Notify that an existing register window has been closed.	-win *window*
regCreateWidget	Create a widget object to *nRegister* window.	regCreateWidget [-win window] -type type -bbox "x1,y1" "x2,y2" where -win *window*: window id -type *type*: Widget type of the new object. Possible values are Push Button, Text, and Option Menu. -bbox *"x1,y1" "x2,y2"*: Bounding box of the newly created widget.
regDeleteSignal	Notify that signals have been deleted from the register window.	-win window -siglist {signal1} {signal2} ... {signaln} where -win *window*: window id -siglist: List of signals been deleted.
regNeedRedraw	Sent when the object that needs to be redrawn exists in the *nRegister* window.	None
regOpenWindow	Notify that a new register window is opened.	-win *window*
regRedraw	Notify that register window needs to be redrawn. You need to invoke **regRedraw** command to redraw the *nRegister* window.	window id

regSelect	Notify that signals are selected.	-win window -selectList objID1 objID2 ... objIDn where -win *window*: window id -selectList: List of signals been selected.
regSignalValueChange	Notify that signal value has changed.	-win window -obj objID -signal signalName -value valueString -signalColor colorID -valueColor colorID -borderColor colorID -font fontID where -win *window*: window id -obj *objID*: Object ID of this signal. -signal *signalName*: Displayed signal name of this signal. -value *valueString*: Displayed value of this signal. -signalColor *colorID*: Color of signal name. -valueColor *colorID*: Color of value string. -borderColor *colorID*: Border color of this signal object. -font *fontID*: Font used to display this signal.
regTimeChange	Notify that the current register time has changed.	-win window -time time where -win *window*: window id -time *time*: New time of the register window.
regUnselect	Notify that all signals had been unselected.	window id
regWidgetEvent	The widget is pressed or activated.	regWidgetEvent [-win window] -type type -obj objID where -win *window*: window id -type *type*: Widget type of the new object. Possible values are Push Button, Text, and Option Menu. -obj *objID*: Object ID of the activated widget.

nMemory

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
memCloseWindow	Notify that an existing *nMemory* window is closed.	-win *window*
memOpenWindow	Notify that a new *nMemory* window is opened.	-win *window*
nMemoryValueChangeViaUI	Sent when the display value changes in the *nMemory* window.	None

Watch Signal

Reason	Descriptions	Calldata (For Tk Interface calldata is preceded with the reason name)
wtchSignalValueChange	Sent when the signal value changes in the watch window.	-tab *TableIndex* -index *rowIndex* -signal *signalName* -value *signalValue*
wtchSignalValueChangeViaUI	Sent when the signal value is changed manually in the watch window.	-signal *signalName* -oldSignalValue *signalValue* -newSigValue *signalValue*

Example

NOTE: This example should be placed in a file called *demo*; otherwise, it does not work.

```
#!wish -f

# Callback Implementation
#
proc myFunc args {
    puts "Received CursorTime Change Event"
    puts "Parameters:"
    foreach i $args {
        puts $i
    }
}

button .b -text "Command Callback Test!" -command exit pack .b

# Invoke Verdi
```

```
#
exec Verdi -tkName abc &
while {1} {
    set s [winfo interps]
    if {-1 != [string first abc $s]} {
        break;
}
after 500
}

# Register Callbacks
#
send abc {AddEventCallback demo myFunc wvCursorTimeChange 1}
```

User - Defined Event Callbacks

Sometimes it is useful for a third party tool to register for an event callback where the reason is defined by the registering application. The Verdi.menu file can be modified to add a user defined menu item. When activated, this menu item calls the third party tool with the user defined reason.

Usually, event callbacks are registered using either AddEventCallback (Tcl/Tk applications) or CmdCltAddEventCallback (C API interface). The reason field should normally reference a pre-defined reason otherwise an error occurs. However, if the reason begins with UDR_ (User Defined Reason) then that reason is defined dynamically. The menu file can then be modified to call back the invoking application with the new reason.

Example

Modify the Verdi.menu file to add a user defined menu.

```
Menu File
{
    "Test"    _s    f.tcl     "Test"    "EventTrigger
UDR_MY_REASON clientdata"
    ...
}

Tk Application: (myTkApp)

# Register Callbacks
#
send Debussy {AddEventCallback myTkApp myEventCallbackProc
UDR_MY_REASON 0}
```

Replaying Commands

The Verdi platform replays the commands by typing the following command when invoking the Verdi platform:

```
>Verdi -play <filename>
```

<*filename*>: The name of the command file that has been programmed manually.

Tk Commands

A full set of Tk commands is available for tool customization and integration. Tk commands allow the creation of sophisticated graphical user interfaces by using the Tcl scripting language. The Tk commands are registered by default when the Verdi platform starts and can be used in scripts with *-play* and the Tcl source commands.

If the Tk script wants to receive event notifications from the Verdi platform (see event reason table for full list of events), then the **AddEventCallback** command can be used.

Example

```
proc wvCreateWindowFunc args {
    puts "=========================================="
    puts "Tk App. Recieved Event"
    puts "Parameters:"
    foreach i $args {
        puts $i;
    }
    puts "=========================================="
}

AddEventCallback [tk appname] wvCreateWindowFunc wvCreateWindow 1
```

Examples can be found under <*NOVAS_INSTALL_DIR*>*/etc/access/tk_library/demos/*.*tcl*. You may use the Tcl source command in the command entry window of the Verdi platform or `-play` command line option to run the samples.

Debugging

The Verdi platform can display commands to stdout as they are received by setting the following environment variable.

To turn on the commands displaying:

```
>setenv CMD_TRACE 1
```

To turn off the commands displaying:

```
>setenv CMD_TRACE 0
```

The command and its arguments are both output.

Socket Command Client

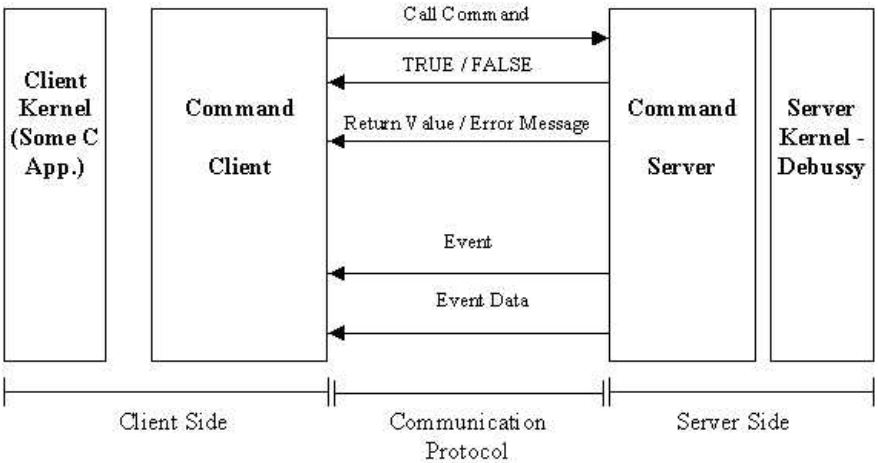
The Verdi platform's command language interface also supports a C interface for third party client-side integration. Third party tools can use this interface to send commands to and receive event notifications from a command server (that is, Verdi).

The library names to link with are *libnclt.a* and *libsys.a*. These libraries are included in the Verdi package. Shared library versions also exist. The header file *cmdclt.h* contains the needed function prototypes.

- Overview
- Setup Verdi as a server
- Implement a client process
- API references

Overview

The following figure depicts the Verdi Command Language Interface with C applications (Socket Command Clients):



Setting Up Verdi as a Server

To start the server for accepting commands, pass the command line arguments:

-serverPort <*port*> for socket interface when invoking the Verdi platform.

Syntax:

```
>Verdi -serverPort <port>
```

<*port*>: the port range is between 1024 and 65535.

Example:

```
>Verdi -serverPort 11060
```

Setting Up a Client Process

There are three steps to setup a client application so it can send commands to the Verdi platform.

1. Create (CmdCltCreate)
2. Connect (CmdCltConnect)
3. Call Command (CmdCltCallCommand)

For example:

```
CMDCLT cmdclt;
char*         result;
cmdclt = CmdCltCreate(0);
CmdCltConnect(cmdclt, 11060, "sps32", -1);
if (CmdCltCallCommand(cmdclt, "debImport -f run.f", 0, &result))
    printf("Command successful. Result = %s\n", result);
else
    printf("Command failed. Error = %s\n", result);
...
CmdCltDestroy(cmdclt);
```

Application Programming Interface (API) References

This document describes how to use the C interface. Use of this API hides the details of the communication protocol.

CMDCLT	CmdCltCreate(char silent)
void	CmdCltDestroy(CMDCLT cmdclient)
void	CmdCltConnect(CMDCLT cmdclient, int port, char* serverName, int timeout)
void	CmdCltSetAbortProc(CMDCLT cmdclient, procVoid func, void* clientData)
char	CmdCltCallCommand(CMDCLT cmdclient, char* command, char async, char** returnString);

char	CmdCltAddEventCallback(CMDCLT cmdclient, char* reason, procVoid func, void* clientData);
void	CmdCltCheckForEvents(CMDCLT cmdclient)
	Helper Functions:
char	CmdCltCheckConnection(CMDCLT cmdclient, char** errorMsg)
char	CmdCltGetDotFormat(char* machineName);
char	CmdCltGetLocalMachineName()
int	CmdCltGetFreePort()
char	CmdCltIsPortFree(int port)

CMDCLT CmdCltCreate(char silent)

`silent`

If TRUE, suppresses status information from being dumped to stdout.

Value Returned

Pointer to a Command Client object.

Remarks

Create an instance of a Command Client object. CmdCltDestroy MUST be called to free the returned object.

Example

```
CMDCLT cmdclt;
cmdclt = CmdCltCreate(0);
...
CmdCltDestroy(cmdclt);
```

void CmdCltDestroy(CMDCLT cmdclient)

`cmdclient`

Pointer to Command Client object.

Remarks

Destroys a Command Client Object. This function MUST be called to free the object returned by CmdCltCreate.

Example

```
CMDCLT cmdclt;
cmdclt = CmdCltCreate(0);
...
CmdCltDestroy(cmdclt);
```

void CmdCltConnect(CMDCLT cmdclient, int port, char* serverName, int timeout)

`cmdclient`

Pointer to Command Client object.

`port`

Port on which to communicate with server. 1024 > port < 65535

`serverName`

Server name. This can be the machine name or an IP address. A value of NULL means the local machine.

`timeout`

Timeout period. If a connection cannot be made within timeout, CmdCltConnect returns. A value of -1 means the function does not return until a connection has been established.

Remarks

Connects to the server. This function does not return until a connection has been established. The server side must be started with the command line option `-serverPort <port number>` where `<port number>` is the same as port. It is not necessary to start the server first.

For example,

```
>Verdi -serverPort 11060
```

Example

```
CMDCLT cmdclt;
cmdclt = CmdCltCreate(0);
CmdCltConnect(cmdclt, 11060, "sps32", -1);
...
CmdCltDestroy(cmdclt);
```

void CmdCltSetAbortProc(CMDCLT cmdclient, procVoid func, void* clientData)

`cmdclient`

Pointer to Command Client object.

`func`

Callback function.

`clientData`

Application defined data that is passed to your callback. This value can be NULL.

Remarks

Can be used to set a function that is called back when the connection with the server is broken.

Whenever CmdCltCallCommand or CmdCltAddEventCallback is called, a check of the connection with the server is made. If the connection is broken, func is called. CmdCltCheckConnection can be called to manually check the connection.

Example

```
void AbortProc(void* clientData)
{
    char* myString = (char*)clientData;
    printf("%s\n", myString);
}
int main()
{
    CMDCLT cmdclt;
    char myString[100];
    cmdclt = CmdCltCreate(0);
    CmdCltConnect(cmdclt, 11060, "sps32", -1);
    strcpy(myString, "Connection broken");
    CmdCltSetAbortProc(cmdclt, AbortProc, myString);
    ...
    CmdCltDestroy(cmdclt);
    return 1;
}
```

char CmdCltCallCommand(CMDCLT cmdclient, char* command, char async, char** returnString)

`cmdclient`

Pointer to Command Client object.

`command`

Command to call. This can be any valid Tcl command.

`async`

If 0, this command is executed synchronously, otherwise it is executed asynchronously.

`returnString`

Points to the result of the call. If return value is 0, returns an error message. Otherwise points to the return result.

Value Returned

0 on error.

Remarks

Used to send a command/Tcl expression to the server and retrieve the result. Sending the command asynchronously can greatly speed up the execution of the command, however, using asynchronous commands it is not possible to get the returnString and *CmdCltCallCommand* will always return TRUE. This function calls *CmdCltCheckForEvents* to check for any pending events before sending the command.

Example

```
CMDCLT cmdclt;
char*         result;
cmdclt = CmdCltCreate(0);
CmdCltConnect(cmdclt, 11060, "sps32", -1);
if (CmdCltCallCommand(cmdclt, "Test A B C", 0, &result))
    printf("Command successful. Result = %s\n", result);
else
    printf("Command failed. Error = %s\n", result);
...
CmdCltDestroy(cmdclt);
```

char CmdCltAddEventCallback(CMDCLT cmdclient, char* reason, procVoid func, void* clientData);

`cmdclient`

Pointer to Command Client object.

`reason`

Event notification callback reason. (See below)

`func`

Callback function. Must be of the form:
void myEventCB(char* eventname, void* clientData, char* callData)

Value Returned

0 on error.

Remarks

Registers a function to be called when a particular event (reason) in the server occurs. To check for events CmdCltCheckForEvents MUST be called.

func should be of the following form:

```
void myEventCB(char* eventname, void* clientData, char* callData)
```

The value pointed to by callData is always a string, the value of which depends on reason.

Please refer to the Reason Table in the *Adding Event Callbacks to Tk Applications* section of this chapter.

Note

If you register for an event with this function, you must call CmdCltCheckForEvents at some regular interval. Please see CmdCltCheckForEvents.

Example

```
void CursorTimeChangeCB(void* clientData, void* callData)
{
    char* cursorTime = (char*)callData;
    printf("Received CursorTimeChange Event. Value = %s\n",
cursorTime);
}
int main()
{
    CMDCLT cmdclt;
```

```
    cmdclt = CmdCltCreate(0);
    CmdCltConnect(cmdclt, 11060, "sps32", -1);
    CmdCltAddEventCallback(cmdclient, "CursorTimeChange",
     wvCursorTimeChangeCB, NULL);
    ...
    CmdCltDestroy(cmdclt);
    return 1;
}
```

void CmdCltCheckForEvents(CMDCLT cmdclient)

`cmdclient`

Pointer to Command Client object.

Remarks

Flushes all pending events. Call this function to check for any pending events and call registered event handlers. This function should be called at some regular interval. Every 100 ~ 500 milliseconds is a good interval. For GUI based applications, a timer can be used.

Note

If events have been registered for with CmdCltAddEventCallback, then this function SHOULD be called, if not, then this function does nothing and does not need to be called.

char CmdCltCheckConnection(CMDCLT cmdclient, char** errorMsg)

`cmdclient`

Pointer to Command Client object.

Remarks

Use this function to test that the listening socket is still alive. It returns TRUE on success and FALSE on error. errorString is only set if there is an error. This function calls the function registered with CmdCltSetAbortProc, if the connection is broken. CmdCltCallCommand and CmdCltAddEventCallback both call this function.

char* CmdCltGetDotFormat(char* machineName);

This function returns the dot notation (IP Address) given the machine name.

char* CmdCltGetLocalMachineName()

This function gets the local machine name.

int CmdCltGetFreePort()

This function gets a free port on the local machine. Returns -1 on error. After 1000 failed attempts, returns -1.

char CmdCltIsPortFree(int port)

This function tests if the given port is free or not (local machine). It returns TRUE if the port is free and FALSE if not free or an error occurs.

Tutorials

Creating and Using a Command Replay File

The command replay file contains commands that are executed when Verdi is started with the `-play` command-line option. For Verdi's commands, see the Verdi Command Language Reference chapter.

Let's create a command replay file that does the following tasks:

1. Import a design in Gate.
2. Create a new waveform window.
3. Load the FSDB file, namely `verilog.fsdb`.
4. Add signals in Group 1.
5. Zoom waveform's time to 300 ~ 600.
6. Create a new schematic window.
7. Push in scope `system.i_cpu`.
8. Push in scope `system.i_cpu.i_ALUB`.
9. Trace the `system.i_cpu.i_ALUB.ACC_tmpreg[1]`signal's drivers.

The command replay file is saved as `a.cmd` and is as follows:

```
debImport -f run.f
srcSetScope system.i_cpu
#wave
set wave1 [wvCreateWindow]
wvSetHierDelimiter -delim /
wvOpenFile -win $wave1 verilog.fsdb
wvAddSignal -win $wave1 /system/CYCLE /system/R_W /system/VMA {/
system/addr[7:0]} /system/clock {/system/data[7:0]}
wvZoom 300 600
#schema
set schema [schCreateWindow -delim . system]
schPointSelect -win $schema 7837 1434
schPushViewIn -win $schema -delim . system.i_cpu
schPointSelect -win $schema 16828 7975
schPushViewIn -win $schema -delim . system.i_cpu.i_ALUB
schZoom -win $schema 16788 46523 63701 92282
schZoom -win $schema 16951 60475 61054 85778
schPointSelect -win $schema 34229 78128
schTraceDriver -win $schema -mode oneLevel -delim .
{system.i_cpu.i_ALUB.\ACC_tmp_reg[1]}
```

To execute the command replay file, `a.cmd`, type the following on the command line when invoking Verdi:

```
>verdi -play a.cmd
```

Invoking Verdi Commands From a Tk Application

This tutorial shows how to send Verdi's command from a Tk application. The sample Tk application, *Signal Finder*, is designed to read signals from a file, to find and trace load/driver of the selected signals in the Tk application and to show the results in nSchema.

The subsequent sections show how to communicate the Tk application with Verdi and how to invoke the Tk application - Signal Finder.

NOTE: Use only the Tcl/Tk application and the command client library (`libnclt.a`) to communicate with Verdi.

Communication Between the Tk Application and Verdi

To send commands to Verdi from a Tk application, you must first register a Tk name for Verdi with the `-tkName` command-line option. The syntax for `-tkName` is as follows:

```
>Verdi -tkName <TK name>
```

For example:

```
>Verdi -tkName abc
```

`<TK name>`: You can choose any name for the Tk name. However, if another TK application already has the same name then a sequential numeric value is appended to the Tk name so that it is unique. For example, `test#2` is assigned for the second Verdi TK name, `test`, when there is already an existing Verdi with TK name of `test`.

Alternatively, you can assign the Tk name to Verdi in your Tk Application, such as the sample Tk application, *Signal Finder*, as follows:

```
exec Verdi -tkName <Tk name> &
```

The source code for the sample Tk application *Signal Finder* is shown below. In this example, abc is used as <Tk name>.

```
if {$argc != 1} {
    error "Usage: wish signal_finder.tcl signalfile"
}

frame .top -borderwidth 5
pack .top -side bottom -fill x
button .top.quit -text Quit -command exit
button .top.find -text Find -command {
    send abc "schSetScope -signal [selection get]"
}
button .top.driver -text "Trace Driver" -command {
    send abc "schTraceDriver [selection get]"
    send abc "schFit -selected"
}
button .top.load -text "Trace Load" -command {
    send abc "schTraceLoad [selection get]"
    send abc "schFit -selected"
}
pack .top.quit .top.find .top.driver .top.load -side right

listbox .signals -yscrollcommand ".scroll set"
scrollbar .scroll -command ".signals yview"

pack .scroll -side right -fill y
pack .signals -side right -expand 1 -fill both

set f [open [lindex $argv 0]]
while {[gets $f line] >= 0} {
    .signals insert end $line
}
close $f

bind .signals <Double-Button-1> {
    set sig [selection get]
    send abc "schSetScope -signal $sig"
    send abc "schFit -selected"
}

set tkAppList [winfo interp]
if {[lsearch $tkAppList abc] < 0} {
    exec Verdi -tkName abc &
}
while {[lsearch $tkAppList abc] < 0} {
    after 1000
    set tkAppList [winfo interp]
}
send abc {debImport -f run.f}
send abc {schCreateWindow}
```

Invoking the Tk application

Copy the Tk application file (`signal_finder.tcl`) and the data file (`signals.list`) to the following directory of your demo case:

```
../demo/Verilog/Gate
```

Invoke the Tk application, *Signal Finder*, with the Tk toolkit, `wish`, on the command line:

```
>wish signal_finder.tcl signals.list
```

Select the `system.i_cpu.i_ALUB.PC` signal in the Tk application and click **Trace Load**. It sends the trace load command of the selected signal to *nSchema* as shown below.

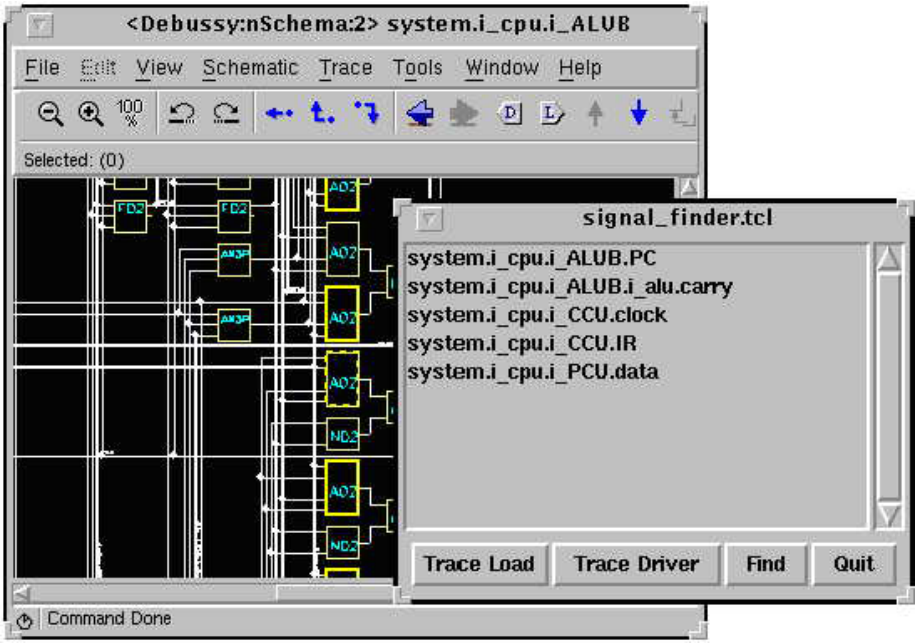


Figure: Trace Load

Support Verdi Windows in Tk Containers

From version 5.2v10, Verdi supports commands/event callbacks that facilitate embedding Verdi windows into Tk Container windows. Verdi adds new Tk event callback reasons. These events return the X window id along with internal window id when a window is open, closed, or iconized (that is, `wvCreateWindow ... -xid 0x123456`).

The Tk application can register for these events in the normal way.

A sample Tk application is provided below:

```
#!../src/bltwish

#############################################################
#
# This simple program illustrates interfacing with Verdi's
# Tk interface. It captures registered events for
# Verdi top level windows. It then uses the corresponding
# corresponding command to get the X window Id and embed
# in a blt container window. ONLY nWave and nTrace events
# are illustrated in this script.
#
# Event list:
#
# wvCreateWindow
# wvCloseWindow
# schCreateWindow
# schCloseWindow
# fsmCreateWindow
# fsmCloseWindow
# srcCreateWindow
# srcCloseWindow
#
# Corresponding X Window command list:
#
# wvGetXWindowId -id <winId>
# schGetXWindowId -id <winId>
# fsmGetXWindowId -id <winId>
# srcGetXWindowId -id <winId>
#
# Is not an example of how to use BTL package. Needs to be
# modified to reflect system settings for blt package and
# Verdi installation path.
#
# NOTE:
# The Tk package MUST be compiled with security turned off
# or use a key-based protection for display like xauth
# rather then xhost. (See Tk documentation for send
# command)
#
#############################################################
```

```
package require BLT
namespace import blt::*

set deb_win_list [list]

proc debExitFunc args {
    exit
}

proc wvCloseWindowFunc args {
    puts "========================================="
    puts "Tk App. Recieved wvCloseWindow Event"

    set wid [lindex $args 1]
    puts "Internal Window ID: $wid"
    set xwid [send test wvGetXWindowId -id $wid]
    puts "X Window ID: $xwid"

    # just to make sure we do not add the same window twice.
    global deb_win_list
    set idx [lsearch -exact $deb_win_list $xwid]
    if {$idx != -1} {
        lreplace $deb_win_list $idx $idx
    }
}

proc wvCreateWindowFunc args {
    puts "========================================="
    puts "Tk App. Recieved wvCreateWindow Event"

    set wid [lindex $args 1]
    puts "Internal Window ID: $wid"
    set xwid [send test wvGetXWindowId -id $wid]
    puts "X Window ID: $xwid"

    # just to make sure we do not embed twice.
    global deb_win_list
    if {[lsearch -exact $deb_win_list $xwid] == -1} {
        set c .c$xwid

        container $c
        $c configure -window $xwid
        pack $c -fill both -expand yes

        lappend deb_win_list $xwid
    }
}

proc srcCloseWindowFunc args {
    puts "========================================="
    puts "Tk App. Recieved srcCloseWindow Event"

    set wid [lindex $args 1]
    puts "Internal Window ID: $wid"
```

```
    set xwid [send test srcGetXWindowId -id $wid]
    puts "X Window ID: $xwid"

    # just to make sure we do not add the same window twice.
    global deb_win_list
    set idx [lsearch -exact $deb_win_list $xwid]
    if {$idx != -1} {
        lreplace $deb_win_list $idx $idx
    }
}

proc srcCreateWindowFunc args {
    puts "========================================="
    puts "Tk App. Recieved srcCreateWindow Event"

    set wid [lindex $args 1]
    puts "Internal Window ID: $wid"
    set xwid [send test srcGetXWindowId -id $wid]
    puts "X Window ID: $xwid"

    # just to make sure we do not embed twice.
    global deb_win_list
    if {[lsearch -exact $deb_win_list $xwid] == -1} {
        set c .c$xwid

        container $c
        $c configure -window $xwid
        pack $c -fill both -expand yes

        lappend deb_win_list $xwid
    }
}

exec /da100/integ/SOL2/deb5.2/debugProd/bin/Verdi -tkName test &
while {1} {
    set s [winfo interps]
    if {-1 != [string first test $s]} {
        break;
    }
    after 500
}

send test {AddEventCallback demo_new srcCreateWindowFunc
srcCreateWindow 1}
send test {AddEventCallback demo_new srcCloseWindowFunc
srcCloseWindow 1}
send test {AddEventCallback demo_new wvCreateWindowFunc
wvCreateWindow 1}
send test {AddEventCallback demo_new wvCloseWindowFunc
wvCloseWindow 1}
send test {AddEventCallback demo_new debExitFunc debExit 1}
```

C Application

This tutorial shows how a C client program can send commands to Verdi and receive event notifications from Verdi. It also demonstrates how to write a C client.

NOTE: Windows users only. Both client and server machines must be using Windows Socket 2 (Winsock2). If not, then communication fails. You can download Windows Socket 2 Update from the Microsoft web site.

NOTE: Use only the Tcl/Tk application and the command client library (`libnclt.a`) to communicate with Verdi.

Setup Verdi As a Server

To start the server so that it can accept commands, you should pass the command-line arguments:

`-serverPort <port>` for socket interface when invoking Verdi.

Syntax:

```
>Verdi -serverPort <port number>
```

Implement a Client Process

1. To compile and run the program, you must link with `libnclt.a`, `libnsys.a`, and the standard Motif libraries. Some other system libraries may also need to be linked. Here is our compile script:

   ```
   CC -o socketClient -I/usr/dt/share/include
   -I<your_API_install_path_root>/<platfotform> socketClientW.c
   -L/usr/openwin/lib -L/usr/dt/lib <your_API_install_path_root>/
   <platform>/libnclt.a  <your_API_install_path_root>/<platform>/
   libnsys.a  -lXm -lXt -lX11 -lsocket -lnsl -lelf
   ```

   (for example, if you try to integrate on the SOL2 machine and the API install path is `/install_root`,)

   ```
   CC -o socketClient -I/usr/dt/share/include -I/install_root/
   SOL2 socketClientW.c -L/usr/openwin/lib -L/usr/dt/lib /
   install_root/SOL2/libnclt.a /install_root/SOL2/libnsys.a  -lXm
   -lXt -lX11 -lsocket -lnsl -lelf
   ```

2. Change the include and library paths according to your system configuration as follows:

   ```
   >setenv LD_LIBRARY _PATH <library path>
   ```

3. Invoke the C application `-socketClientW` with the following command line:

   ```
   >socketClientW <machine name or IP of server> <port number>
   ```

   Your desktop should look like the following:

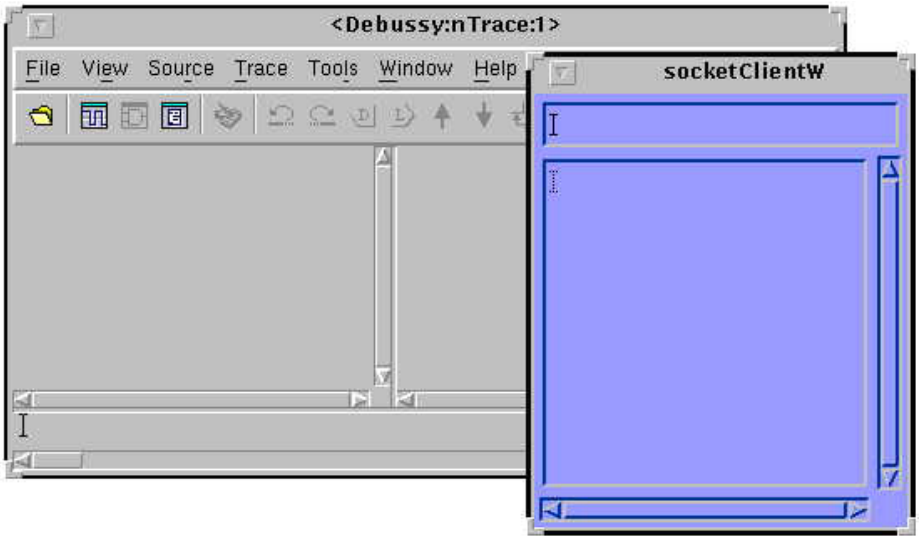


Figure: socketClientW

Now, you can enter Verdi commands in the **socketClientW** window. These commands are passed to Verdi's Tcl interpreter.

The `socketClientW.c` program

```
/*
    This simple program demonstrates how to write
    a C client that can send commands to and receive
    event notifications from Novas tools (e.g. Verdi).

    Usage:

    Verdi -serverPort <port number>
    socketClient <machine name or IP> <port number>

    e.g.

    sps32 > Verdi -serverPort 11060
    sps11 > socketClient sps32 11060
*/

#include <stdlib.h>
#include "Xm/RowColumn.h"
#include "Xm/Text.h"
```

```
#include "Xm/RowColumn.h"
#include "Xm/Form.h"
#include "Xm/PushB.h"
#include "Xm/Label.h"
#include "cmdclt.h"

CMDCLT _cmdclt;
XtAppContext _app;

void
AllEventsCB(char* eventName, void* clientData, void* callData)
{
    char *buf, *curtext;
    int  calllen = 0;

    if (callData)
        calllen = strlen((char*)callData);

    curtext = XmTextGetString((Widget)clientData);
    buf = (char*)malloc((calllen + strlen(curtext) +
    strlen((char*)eventName)) * sizeof(char) + 100);
    if (callData)
        sprintf(buf, "Event: %s callData: %s\n%s", eventName,
    (char*)callData, curtext);
    else
        sprintf(buf, "Event: %s callData: No calldata\n%s",
        eventName, curtext);
        XmTextSetString((Widget)clientData, buf);
        free(buf);
        XtFree(curtext);
}

void
ConnectionBroken(void* clientData)
{
    XmTextSetString((Widget)clientData, "Connection seems to have
    broken. You should restart server and client");
}

static void
CheckForEventsCB(XtPointer clientData, XtIntervalId *timerid)
{
    CmdCltCheckForEvents(_cmdclt);
    XtAppAddTimeOut(_app, 100, CheckForEventsCB, NULL);
}

static void
ExitCB(Widget widget, XtPointer clientData, XtPointer callData)
{
    exit(0);
}

static void
```

```
SendCommand(Widget widget, XtPointer clientData, XtPointer
callData)
{
    char *cmd, *buf, res, *result, *curtext;
    int  reslen = 0;
    Widget stext = (Widget)clientData;

    if (cmd = XmTextGetString(widget))
    {
        res = CmdCltCallCommand(_cmdclt, cmd, 0, &result);

        /*
        We don't know how big the result is so it is
        best to dynamically allocate the storage.
        */
        curtext = XmTextGetString(stext);
        if (result)
            reslen = strlen(result);
        buf = (char*)malloc((reslen + strlen(curtext)) *
        sizeof(char) + 100);
        if (res)
        {
            if (result)
                sprintf(buf, "Result: %s\n%s", result, curtext);
            else
                sprintf(buf, "Result: No result string\n%s",
                curtext);
        }
        else
        {
            if (result)
                sprintf(buf, "Error: %s\n%s", result, curtext);
            else
                sprintf(buf, "Error: No error message\n%s",
                curtext);
        }
        XmTextSetString(stext, buf);

        free(buf);
        XtFree(cmd);
        XtFree(curtext);
    }
}

int main(int argc, char* argv[])
{
    Widget toplevel, form, text, stext, exitBtn, label1, label2;
    Arg args[20];
    int n;
    XmString xmstring;

    if (argc != 3)
    {
        printf("Usage: socketClientC <server> <port>\n");
```

```
        printf("Server must be started with -serverPort
<port>\n");
        exit(0);
    }

    XtSetLanguageProc(NULL, NULL, NULL);

    toplevel = XtVaAppInitialize(&_app, "Demos", NULL, 0,
        &argc, argv, NULL, NULL);

    form = XtVaCreateWidget("form", xmFormWidgetClass, toplevel,
                                        NULL);
    XtManageChild(form);

    n = 0;
    xmstring = XmStringCreate("Enter Command:",
    XmFONTLIST_DEFAULT_TAG);
    XtSetArg(args[n], XmNlabelType, XmSTRING); n++;
    XtSetArg(args[n], XmNlabelString, xmstring); n++;
    XtSetArg(args[n], XmNalignment, XmALIGNMENT_BEGINNING); n++;
    XtSetArg(args[n], XmNtopAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNrightAttachment, XmATTACH_FORM); n++;
    label1 = XmCreateLabel(form, "Command", args, n);
    XtManageChild(label1);
    XmStringFree(xmstring);

    n = 0;
    XtSetArg(args[n], XmNtopAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNtopWidget, label1); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNrightAttachment, XmATTACH_FORM); n++;
    text = XmCreateText(form, "command", args, n);
    XtManageChild(text);

    n = 0;
    xmstring = XmStringCreate("Result:", XmFONTLIST_DEFAULT_TAG);
    XtSetArg(args[n], XmNlabelType, XmSTRING); n++;
    XtSetArg(args[n], XmNlabelString, xmstring); n++;
    XtSetArg(args[n], XmNalignment, XmALIGNMENT_BEGINNING); n++;
    XtSetArg(args[n], XmNtopAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNtopWidget, text); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNrightAttachment, XmATTACH_FORM); n++;
    label2 = XmCreateLabel(form, "Result", args, n);
    XtManageChild(label2);
    XmStringFree(xmstring);

    n = 0;
    xmstring = XmStringCreate("Exit", XmFONTLIST_DEFAULT_TAG);
    XtSetArg(args[n], XmNlabelType, XmSTRING); n++;
    XtSetArg(args[n], XmNlabelString, xmstring); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNrightAttachment, XmATTACH_FORM); n++;
```

```
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_FORM); n++;
    exitBtn = XmCreatePushButton(form, "Exit", args, n);
    XtManageChild(exitBtn);
    XmStringFree(xmstring);

    n = 0;
    XtSetArg(args[n], XmNwidth, 300); n++;
    XtSetArg(args[n], XmNheight, 200); n++;
    XtSetArg(args[n], XmNeditMode, XmMULTI_LINE_EDIT); n++;
    XtSetArg(args[n], XmNtopAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNtopWidget, label2); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNrightAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNbottomWidget, exitBtn); n++;
    stext = XmCreateScrolledText(form, "result", args, n);
    XtManageChild(stext);

    XtRealizeWidget(toplevel);

   XtAddCallback(text, XmNactivateCallback, SendCommand, stext);
    XtAddCallback(exitBtn, XmNactivateCallback, ExitCB, NULL);

    _cmdclt = CmdCltCreate(0);
    CmdCltConnect(_cmdclt, atoi(argv[2]), argv[1], -1);
    CmdCltSetAbortProc(_cmdclt, (procVoid)ConnectionBroken,
    stext);
    CmdCltAddEventCallback(_cmdclt, "AllEvents",
    (procVoid)AllEventsCB, stext);
    XtAppAddTimeOut(_app, 100, CheckForEventsCB, NULL);

    XtAppMainLoop(_app);
}
```

References

Verdi Common Command Rules

Current Window Concept

1. Every command whose action is dependent on window should provide an argument `-win windowId` to let users change the current window and the action is applied to the new current window. If there is no win argument in the command, it actions to the current window.

   For example,

   ```
   wvOpenFile -win $_nWave1 test1.fsdb
   ```

   The above command changes *nWave*'s current window to **$_nWave1** and open file in **$_nWave1**.
2. If a new window is created, it is set as the current window. For example, **schCreateWindow** creates a new window and set it as the current window.
3. If the current window is closed, a new window needs to be set as the current window. Otherwise, subsequent commands may fail.

Naming Style

All command names have a prefix. The prefix depends on which tool the command is for. For example, all *nSchema* commands have `sch` as a prefix.

Following are the prefixes:

Prefix	**Tools**
deb	debussy
src	nTrace
sch	nSchema
wv	nWave
fsm	nState
wtch	watch
reg	Register

Prefix	Tools
mem	Memory

- Command names are composed of a tool prefix, action, and object. For example, `wvLoadFile`. Action and Object's first letter is uppercase.
- Command parameters which are optional are prefixed with a dash. For example, `schPrint -color true`.
- Some common words are abbreviated to a shorthand notation. The conventions are as follows:

Short	Long
attr	attribute
hier	hierarchical
delim	delimiter
pos	position
dup	duplicate

Parameter Sequence

There are two types of parameters: optional and required parameters. Optional parameters (prefixed with a dash) do not need to be specified. They are not position dependant and can be placed in any position. Required parameter must be specified and their positions in the parameter list are sometimes fixed.

For example, the following two commands are equivalent:

```
schTraceDriver -mode module system.i_cpu.clk
schTraceDirver system.i_cpu.clk -mode module
```

Color Names

```
-color colorname
```

RGB	Color Name
255 255 255	white
230 230 250	lavender
238 238 209	lightyellow2
255 228 225	mistyrose
192 192 192	gray

RGB	Color Name
112 128 144	slategray
255 0 0	red
255 69 0	orangered
255 99 71	tomato
205 0 0	red3
255 165 0	orange
255 140 0	darkorange
255 255 0	yellow
192 255 62	olivedrab1
188 238 104	darkolivegreen2
255 215 0	gold
238 221 130	lightgoldenrod
154 205 50	yellowgreen
0 255 0	green
0 250 154	mediumspringgreen
0 238 118	springgreen2
67 205 128	seagreen3
50 205 50	limegreen
34 139 34	forestgreen
0 0 255	blue
100 149 237	cornflowerblue
123 104 238	mediumslateblue
0 191 255	deepskyblue
135 206 250	lightskyblue
142 229 238	cadetblue2
0 255 255	cyan
175 238 238	paleturquoise
0 206 209	darkturquoise
102 205 170	mediumaquamarine
127 255 212	aquamarine
160 32 240	purple
189 183 107	darkkhaki

RGB	Color Name
160 82 45	sienna
205 133 63	peru
222 184 135	burlywood
244 164 96	sandybrown
210 105 30	chocolate
165 42 42	brown
250 128 114	salmon
255 105 180	hotpink
255 20 147	deeppink
255 182 193	lightpink
238 216 174	wheat2
255 0 255	magenta
238 130 238	violet
127 255 212	aquamarine1
118 238 0	chartreuse2
238 169 184	pink2
204 204 204	gray80
0 0 0	black
0 255 127	springgreen
107 142 35	olivedrab
143 188 143	darkseagreen
180 238 180	darkseagreen 2
46 139 87	seagreen
60 179 113	mediumseagreen
32 178 170	lightseagreen
152 251 152	palegreen
255 192 203	pink
245 222 179	wheat

Line style

```
-lineStyle style_name
```

Pattern	Name
	line_solid
	short_dashed
	dot_dashed
	dash_dot
	dash_dot_dot
	short_dashed_1
	dot_dash_1
	dash_dot_1
	dash_dot_dot_1
	long_dashed

Font Name

"Helvetica 8", "Helvetica 10", "Helvetica 12", "Helvetica 14", "Helvetica 18", "Helvetica 24"

"Helvetica bold 8", "Helvetica bold 10", "Helvetica bold 12", "Helvetica bold 14", "Helvetica bold 18", "Helvetica bold 24"

"Times 8", "Times 10", "Times 12", "Times 14", "Times 18", "Times 24"

"Times bold 8", "Times bold 10", "Times bold 12", "Times bold 14", "Times bold 18", "Times bold 24"

"Clean 8", "Clean 10", "Clean 12", "Clean 14", "Clean 16"

"Clean bold 8", "Clean bold 10", "Clean bold 12", "Clean bold 14", "Clean bold 16"

"Courier 8", "Courier 10", "Courier 12", "Courier 14", "Courier 18", "Courier 24"

"Courier bold 8", "Courier bold 10", "Courier bold 12", "Courier bold 14", "Courier bold 18", "Courier bold 24"

"Fixed 8", "Fixed 10", "Fixed 12", "Fixed 14", "Fixed 20"

"Small 12"

Radix

Hex	Hexadecimal
Oct	Octal
Bin	Binary
UDec	Unsigned Decimal
2Com	Signed 2’s Complement
1Com	Signed 1’s Complement
ASCII	ASCII
SMag	Signed Magnitude

System Tcl Commands

Environment Commands

sysGetEnv

Description

Gets the value for an environment variable.

Syntax

```
sysGetEnv [-name] name
```

Arguments

`-name` *`name`*

the environment variable name

Value Returned

Returns the value of an environment variable.

Example

The following two statements have the same effect:

```
sysGetEnv -name ENV1
sysGetEnv ENV1
```

sysPutEnv

Description

Sets an environment variable for the current session. The variable name is set to the value of the environment variable.

Syntax

```
sysPutEnv [-name name -value value] | [name=value]
```

Arguments

`-name` *`name`*

the environment variable name

`-value` *`value`*

the new value for the environment variable

Value Returned

Returns the value of an environment variable.

Example

The following two statements have the same effect:

```
sysPutEnv -name ENV1 -value 1
sysPutEnv ENV1=1
```

General System Commands

sysConsoleEnable, sysInfoEnable, sysWarnEnable

Description

Uses the following three Tcl commands to turn on/off warning messages:

`sysWarnEnable`

`sysInfoEnable`

`sysConsoleEnable`

Syntax

```
sysWarnEnable [-enable | -disable]
sysInfoEnable [-enable | -disable]
sysConsoleEnable [-enable | -disable]

set tmp [send Verdi wvCreateWindow]
...
...
send Verdi sysRaiseWindow $tmp
```

sysEnableTclAutoEscape

Description

Uses this command to tell Verdi that incoming commands may contain []. The command protects the content against the nested evaluation rather than evaluating the content contained within the []. Pass 1 to enable and 0 to disable this command.

Syntax

```
sysEnableTclAutoEscape 1 | 0
```

The default value is *0*, protecting against [] is not done.

NOTE: **sysEnableTclAutoEscape** limits the flexibility of Verdi's commands as it is no longer possible to nest Tcl commands. For example, sending a command such as `set a [wvCreateWindow]` does not work. The command must be evaluated in two separate steps.

Value Returned

None.

sysEnableTclEventFilter

Description

Use this command to tell Verdi to filter all Tcl special symbols from event information before sending the event to the connected client application. Pass 1 to enable and 0 to disable this command. This command is only useful for the socket interface (`libnclt.a`) or shared library interfaces to the command language. Tk programs should not call this function by default, Tcl removes such symbols.

Syntax

```
sysEnableTclEventFilter 1 | 0
```

sysRaiseWindow

Description

Raises the top-level window with `windowId` to the top of the window stacking order. If the top-level window is in an iconized state, it is returned to its normal non-iconized view.

The calldata as passed to the create window event callbacks, can also be used as `windowId` to `sysRaiseWindow`. For details, see the AddEventCallback command.

Syntax

```
sysRaiseWindow windowId
```

Argument

`windowId`

It can be any top-level window ID as returned from the *create window* functions (for example, **wvCreateWindow**, **schCreateWindow**, **fsmCreateWindow**, and **srcCreateWindow**).

Example

```
# Assuming a Tk script is communicating with Verdi which
# is using the default Tk name "Verdi".
```

User Interface Commands

Graphical User Interface Commands

verdiDockWidgetSetCurTab

Description

Sets the dock widget as the active tab.

Syntax

```
verdiDockWidgetSetCurTab -dock dockWidgetName
```

Arguments

`-dock` *`dockWidgetName`*

Specifies the dock widget name.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Example

```
verdiDockWidgetSetCurTab -dock widgetDock_<Decl._Tree>
verdiDockWidgetSetCurTab -dock widgetDock_<Inst._Tree>
verdiDockWidgetSetCurTab -dock windowDock_nWave_2
```

verdiGetAttribute

Description

Obtains the attribute of windows or widgets.

Syntax

```
verdiGetTypeCount –win winName | -widget widgetName |
-dock dockName -type title | winId | parentId | width | height |
size | rect | showState | maxSize | minSize | dockState | all
```

Arguments

`all`

Obtains all attributes.

`-dock dockName`

Specifies the dock name.

`dockState`

Obtains the dock state. The options include: **undocked**, **top**, **bottom**, **left**, and **right**.

`height`

Obtains the height.

`maxSize`

Obtains the maximum size.

`minSize`

Obtains the minimum size.

`parentId`

Obtains the parent window id.

`rect`

Obtains the rectangle. The format is *left_top_x*, *left_top_y*, *width*, and *height*.

`showState`

Obtains the show state. The options include: `normal`, `minimized`, `maximized`, and `hidden`.

`size`

Obtains the size. The format is *width* and *height*.

`title`

Obtains the title.

`-type title | winId | parentId | width | height | size | rect | showState | maxSize | minSize | dockState | all`

Specifies the type. The options include: `title`, `winId`, `parentID`, `width`, `height`, `size`, `rect`, `showState`, `MaxSize`, `MinSize`, `dockState`, and `all`.

`-widget` *`widgetName`*

Specifies the widget name.

`width`

Obtains the width.

`winID`

Specifies the window id.

Value Returned

Returns attributes if successful; Otherwise, returns 0.

Examples

```
verdiGetAttribute –win $_nWave2 –all

nWave_2: type=nWave, title=*<nWave:2> No File Opened,
win_id=69216913, parent_id=69216857, rect=(405,517,900,255),
show_state=normal, max_size = (16777215,16777215),
min_size=(0,0), dock_state=bottom
```

verdiGetTypeCount

Description

Obtains the count of windows or widgets.

Syntax

```
verdiGetTypeCount –win winName | -widget widgetName |
-dock dockName | -type typeName [-all]
```

Arguments

`-all`

Counts for all windows or widgets of the same type. The default is to count for visible windows or widgets.

`-dock dockName`

Specifies the dock name.

`-type typeName`

Specifies the type name. The options include: `Verdi`, `nWave`, `nSchema`, `hdlHier`, `signalList`, `hdlSrc`, `messageWindow`, `pdmlSrc`, `pdmlHier`, `pdmlModeTable`, `WelcomePage`, `watchSignals`, `watchExpressions`, and `svtbSrc`.

`-widget widgetName`

Specifies the widget name.

`-win winName`

Specifies the window name.

Value Returned

Types count if successful; Otherwise, returns 0.

Examples

```
verdiGetTypeCount –win $_nWave2
```

verdiHideBanners

Description

Hides or shows the banner for the active frame/window.

Syntax

```
verdiHideBanners -win winName -on|off
```

Arguments

`-win` *winName*

Specifies the active frame/window name.

`-on|off`

Specifies *on* to hide the banner. Specifies *off* to show the banner.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Examples

```
verdiHideBanners -win Verdi_1 -off
verdiHideBanners -win Verdi_1 -on
```

verdiHighlightSignal

Description

Sets or resets the signal highlight color.

Syntax

```
verdiHighlightSignal [-sigColor {$signalName $color}[{$signalName
$color}]][-apply|-close]
```

Arguments

`-sigColor {$signalName $color}`

Specifies the signal's complete hierarchy name, for example, `top.A\[3:0\]` and the color to be assigned for the signal, for example, `ID_BLUE3`.

`-apply`

Applies highlight color settings.

`-close`

Closes the **Highlight** dialog box.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Examples

```
verdiHighlightSignal -sigColor {"top.A\[3:0\]" ID_BLUE3}
{"top.B\[3:0\]" ID_YELLOW1}
verdiHighlightSignal -sigColor {"top.A\[3:0\]" N/A }
verdiHighlightSignal -apply
verdiHighlightSignal -close
```

verdiLoadHighlight

Description

Loads the highlighted color settings to the file.

Syntax

```
verdiLoadHighlight $fileName
```

Arguments

`fileName`

Specifies the file name to load the highlighted color settings.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Examples

```
verdiLoadHighlight color.hls
```

verdiSaveHighlight

Description

Saves the highlighted color settings to the file.

Syntax

```
verdiSaveHighlight $fileName
```

Arguments

`fileName`

Specifies the file name to save the highlighted color settings.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Examples

```
verdiSaveHighlight color.hls
```

verdiSetFont

Description

Changes font settings in Verdi.

NOTE: The font setting applies only to QT widget and windows.

Syntax

```
verdiSetFont -font fontname -size size [-monoFont font]
[-monoFontSize size]
```

Arguments

`-font`

Specifies the font name.

`-size`

Specifies the size.

`-monoFont`

Specify the monospaced font name.

`-monoFontSize`

Specify the monospaced font size.

Value returned

Returns 1 if successful; Otherwise, returns 0.

Examples

```
verdiSetFont -font Helvetica -size 14
verdiSetFont -font "Bitstream Vera Sans" -size 11
verdiSetFont -monoFont "Courier" -monoFontSize 12
```

verdiSetPrefEnv

Description

Sets preference options for the GUI configuration in the Verdi platform.

Syntax

```
verdiSetPrefEnv [-bUndockNewCreatedWindow on|off]
[-bKeepUndockedWidgetsOnTop on|off]
```

Arguments

`-bKeepUndockedWidgetsOnTop on|off`

Specifies *on* to place undocked widgets in front of the main window. Specifies *off* to place undocked widgets behind the main window after a random click on the main window. The default is *on*.

`-bUndockNewCreatedWindow on|off`

Specifies *on* to display a newly-created frame as a stand-alone window. Specifies *off* to automatically dock newly-created frames to the main window in the default location. The default is *off*.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Examples

```
verdiSetPrefEnv -bUndockNewCreatedWindow on
verdiSetPrefEnv -bKeepUndockedWidgetsOnTop on
```

verdiSyncBrowserDir

Description

Turns *on* or *off* the **Synchronize File Browser Directory** option in the **General -> Configure GUI** page of the *Preferences* form.

Syntax

```
verdiSyncBrowserDir -on|-off
```

Arguments

`-on|-off`

Specifies *on* to enable the **Synchronize File Browser Directory** option; Specifies *off* to disable the option. When this option is turned *on*, the file browser directory is synchronized to the previously opened file browser directory. When this option is turned *off*, the file browser directory in each form operates independently. The default is *on*.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Examples

```
verdiSyncBrowserDir -off
```

verdiWindowBeWindow

Description

Undocks the specified window.

Syntax

`verdiWindowBeWindow -win` *`win_ID`*

Arguments

`-win` *`win_ID`*

Specifies the window ID.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Example

```
verdiWindowBeWindow -win $_nWave2
```

verdiWindowPrependTitle

Description

Sets the prefix for the specified window title.

Syntax

```
verdiWindowPrependTitle -win window -preTitle preTitle
```

Arguments

`-preTitle`

Specifies the prefix of the window title.

`-win` *`window`*

Specifies the active frame/window name.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Example

```
verdiWindowPrependTitle -win $_nWave2 -preTitle "wave2 pretitle"
```

wtHideWindow

Description

Hides top-level windows at startup. Top-level windows are suppressed at startup when Verdi is started with the `-nowinmap` command-line option.

Syntax

```
wtHideWindow -win $xwid
```

Arguments

`-win` *`$xwid`*

X window ID. The X window ID is obtained via the `xxxGetXWindowId` commands (that is, `srcGetXWindowId`).

Value Returned

No value returned on success. Otherwise the interpreter result is set to a message to indicate the error.

Example

```
proc myCreateWindowFunc args {
    set wid [lindex $args 1]
    set xwid [send test srcGetXWindowId -id $wid]
    send test wtHideWindow -win $xwid
}

exec Verdi -nowinmap -tkName test &
...

send test {AddEventCallback demo_new myCreateWindowFunc
srcCreateWindow 1}
```

wtShowWindow

Description

Shows top-level windows at startup. Top-level windows are suppressed at startup when Verdi is started with the `-nowinmap` command-line option.

Syntax

```
wtShowWindow -win $xwid
```

Arguments

`-win` *`$xwid`*

X window ID. The X window ID can be obtained via the **xxxGetXWindowId** commands (i.e. **srcGetXWindowId)**.

Value Returned

No value returned on success. Otherwise the interpreter result is set to a message to indicate the error.

Example

```
proc myCreateWindowFunc args
   {set wid [lindex $args 1]
    set xwid [send test srcGetXWindowId -id $wid]
    send test wtShowWindow -win $xwid}

exec Verdi -nowinmap -tkName test &
...

send test {AddEventCallback demo_new myCreateWindowFunc
srcCreateWindow 1}
```

Menu and Toolbar Commands

qwAction

Description

This command creates an action. An action corresponds to an item on a menu or a toolbar button. This created action should be used with other commands (for example, `qwAddMenuAction`).

The return value is the action pointer object.

Syntax

```
qwAction [-id mainwinId] -name name [-tcl tclCommand]
[-icon icon] [-sep] [-shortcut shortcut] [-override]
[-tooltip tip] [-checkable] [-checked | -unchecked]
[-disabled | -enabled] [-invisible | -visible] [-text text]
```

Arguments

`-checkable`

Specifies that the action is toggled on or off. The action becomes a checkable toggle command when it is added to the menu (for example, when using the `qwAddMenuAction` command).

`-checked`

Specifies the action as a default-as-*on* checkable toggle command.

`-disabled`

Disables the created action. The created command is disabled (grayed out) when it is added to the menu (for example, when using the `qwAddMenuAction` command).

`-enabled`

Enables the created action. The disabled (grayed out) command is enabled (for example, when using the `qwAddMenuAction` command).

`-icon` *`icon`*

Specifies to show an icon for the action. The `icon` value is the full path and the image file name. The file should be in the PNG format.

`-id` *`mainwinId`*

The `windowId` argument refers to the main window. This option is ignored if it is executed within the context of the `qwConfig` command, otherwise it is required.

`-invisible`

Hides the created action. The created command is invisible to users even if the command is added to the menu.

`-name` *`name`*

The `name` value refers to the object. If the `name` value is the name of an existing action, that action is used.

`-override`

This command overrides the duplicated shortcut for the existing actions. For example, when an action is created with the shortcut **CTRL+S**, and there are some existing actions with the same shortcut, then the `-override` option tries to remove the duplicated shortcuts for the existing actions.

`-sep`

Specifies whether the action is a separator.

`-shell` *`shellCommand`*

Runs the specified shell command when the action is activated.

`-shortcut` *`shortcut`*

Specifies the bind key to use with this action.

`-tcl` *`tclCommand`*

Runs the specified Tcl command when an action is activated.

`-text` *`text`*

Displays a string for the action.

`-tooltip` *`tip`*

The ToolTip (information) to display in the yellow tip window if the action is placed on a toolbar and the cursor moves over it.

`-unchecked`

Specifies the action as a default-as-*off* checkable toggle command.

`-visible`

Shows the hidden action. The invisible command becomes visible to users.

Example

```
qwAction -id $_nWave2 -name "list_registers" -text "List All
Registers" -tcl "source list_registers.tcl"
qwAction -id $_Verdi_1 -name "clean1" -text "Clean1" -tcl "puts
```

```
CLEAN!!CLEAN!!-1" -shortcut "CTRL+S" -override
```

qwActionGroup

Description

This command creates an action group. An action group corresponds to a pull-down menu or a toolbar. It consists of 0 or more actions. Returns value is the `actionGroup` pointer object.

Syntax

```
qwActionGroup [-id mainwinId] -name name [-text text]
[-invisible | -visible] [-disabled | -enabled]
[-exclusive | -nonexclusive] [-actionNames actionNameList | -
actions actionList]
```

Arguments

`-actionNames` *`actionNameList`*

A list of named action objects. The named actions should be created with `qwAction`.

`-actions` *`actionList`*

A list of action objects. This is the return value of `qwAction` in the Tcl list format.

`-disabled`

Not sensitive to the user input.

`-enabled`

Sensitive to the user input.

`-exclusive`

Specifies that only one action within this group is selected at one time.

`-id` *`mainwinId`*

Window id of the main window. This option is ignored if it is executed within the context of the `qwConfig` command, otherwise it is required.

`-invisible`

Specifies to hide an action.

`-name` *`name`*

The `name` value refers to the object. If the `name` value is the name of an existing action group, that group is used.

`-nonexclusive`

Specifies if multiple actions within this group are selected at one time.

`-text` *`text`*

The display string. The default is the action name. For details, see `-name`.

`-visible`

Specifies to show the action.

Example

```
qwActionGroup -id $_nWave2 -name "my_group_1" -actionNames
{"list_registers" "list_latches"}
```

qwAddMenuGroup

Description

Creates a drop-down menu by adding an action group to the menu.

Syntax

```
qwAddMenuGroup [-id mainwinId] [-before menuName] [-group
groupName]
```

Arguments

`-before` *`menuName`*

Inserts before the specified pull-down menu/item.

`-group` *`groupName`*

Specifies the name of the group to be inserted.

NOTE: Refer to the form opened with the **Tools->Customize Menu/Toolbar** command to identify the available group names.

`-id` *`mainwinId`*

Window id of the main window. This option is ignored if it is executed within the context of the `qwConfig` command, otherwise it is required.

Example

```
qwAddMenuGroup -id $_nWave2 -group "my_group_1"
```

qwAddMenuAction

Description

Adds an action to an existing menu. To add an existing group as a submenu, an action is created using `qwAction` with the same name as the group. For details, see Example 2 at the end of the document.

Syntax

```
qwAddMenuAction [-id mainwinId] [-action actionName]
[-before actionName] [-group groupName]
```

Arguments

`-action` *`actionName`*

Specifies the name of the action to insert.

`-before` *`actionName`*

Specifies the name of the action to insert this action before.

`-group` *`groupName`*

Specifies the name of the group (menu) to be inserted.

NOTE: Refer to the form opened with the **Tools->Customize Menu/Toolbar** command to identify the available group names.

`-id` *`mainwinId`*

Window id of the main window. This option is ignored if it is executed within the context of the `qwConfig` command, otherwise it is required.

Example

```
qwAddMenuAction -id $_nWave2 -action "my_group_1" -group "File"
-before "Open..."
```

qwAddToolBarAction

Description

Adds an action to an existing toolbar.

Syntax

```
qwAddToolBarAction [-id mainwinId] [-action actionName] [-before
actionName] [-group <group-name>]
```

Arguments

`-action actionName`

Specifies the name of the action to insert.

`-before actionName`

Specifies the name of action to insert this action before.

`-group groupName`

Specifies the name of the group (toolbar) to be inserted.

`-id mainwinId`

Window id of the main window. This option is ignored if it is executed within the context of the `qwConfig` command, otherwise it is required.

Example

```
qwAddToolBarAction -id $_nWave2 -action "my_group_1" -group
"WAVE_OPEN"
```

qwAddToolBarGroup

Description

Creates a new toolbar by adding an action group to the main window.

Syntax

```
qwAddToolBarGroup [-id mainwinId] [-group groupName]
```

Arguments

`-group` *`groupName`*

Specifies the name of the group to be inserted.

`-id` *`mainwinId`*

Window id of the main window. This option is ignored if it is executed within the context of the `qwConfig` command, otherwise it is required.

Example

```
qwAddToolBarGroup -id $_nWave2 -group "my_group_1"
```

qwConfig

Description

This command groups menu/toolbar commands and applies those commands to a window type. If the target window type does not exist, the commands apply to the created window. This command groups the commands that are applied to all windows of the same type. It is called multiple times for the same window type.

Syntax

```
qwConfig -type type -cmds [list {<cmd>} {<cmd>} ...]
```

Arguments

`-cmds [list {<cmd>} {<cmd>} ...]`

Lists Tcl commands to configure menus or toolbars. `<cmd>` corresponds to one Tcl command.

`-type` *`type`*

Specifies the window type to which the commands are applied. Possible values include *Verdi, nWave, nSchema, nECO, nRegister, nCompare,*

nMemory, sourceFileViewer, nPowerMap, SequenceDiagram, tFlowView, tTraceXRst, nClk, ClockTreeBrowser, ClockDomain, CheckCrossingPaths, ClockStatistics, EventSequence, nTrans, nTransComparison, nTransData, nTransStatistics, nTransRelationship, nState, nEditSchematic, nEditor, nText, CRAnalyzer, ScanChainExtract, tsSwitchAnalyzerReport, FFT, and *SmartLog*.

Examples

Example 1

This example adds a new menu command to the **File** pull-down menu of *nWave*. The command is added to the top of the menu.

```
qwConfig -type nWave -cmds [list {qwAction -name "mycmd1" -text
"My Command1" -shell "xclock&" -shortcut "Ctrl+Shift+X"}]

qwConfig -type nWave -cmds [list {qwAddMenuAction -action
"mycmd1" -group "File" -before "Open..."}]
```

Example 2

This example adds two new commands to a submenu in the **File** pull-down menu. The submenu itself is an action. The submenu group and action must have the same name.

```
# create a new group "mygrp1"
qwConfig -type nWave -cmds [list {qwAction -name "mycmd4" -text
"My Command3" -shell "xclock&" -shortcut "Ctrl+Shift+X"}]
qwConfig -type nWave -cmds [list {qwAction -name "mycmd5" -text
"My Command4" -tcl "sysInfo 123" -shortcut "Ctrl+Shift+Y"}]
qwConfig -type nWave -cmds [list {qwActionGroup -name "mygrp1" -
actionNames {"mycmd4" "mycmd5"}}]

# add as submenu
qwConfig -type nWave -cmds [list {qwAction -name "mygrp1" -text
"My Sub Menu"}]
qwConfig -type nWave -cmds [list {qwAddMenuAction -action
"mygrp1" -group "File" -before "Open..."}]
```

Example 3

This example adds the `mygrp1` group from Example 2 as a new toolbar.

```
qwConfig -type nWave -cmds [list {qwAddToolBarGroup -group
"mygrp1"}]
```

Example 4

This example adds the `mygrp1` group from Example 2 as a new pull-down menu to the menubar.

```
qwConfig -type nWave -cmds [list {qwAddMenuGroup -group
"mygrp1"}]
```

Example 5

Combine examples 1 ~ 4 into a single `qwConfig` command.

```
qwConfig -type nWave -cmds [list{
    qwAction -name "mycmd1" -text "My Command1" -shell "xclock&"
    } {
    qwAddMenuAction -action "mycmd1" -group "File" -before
    "Open..."
    } {
    qwAction -name "mycmd4" -text "My Command3" -shell "xclock&"
    } {
    qwAction -name "mycmd5" -text "My Command4" -tcl "sysInfo
    123"
    } {
    qwActionGroup -name "mygrp1" -actionNames {"mycmd4" "mycmd5"}
    } {
    qwAction -name "mygrp1" -text "My Sub Menu"} {qwAddMenuAction
    -action "mygrp1" -group "File" -before "Open..."
    } {
    qwAddToolBarGroup -group "mygrp1"
    } {
    qwAddMenuGroup -group "mygrp1"
    }]
```

Example 6

This example adds a group of commands to the `display waveform` command that is added to the right-click menu in the `<Inst._Tree>` frame.

```
qwConfig -type Verdi -cmds [list \
    {qwAction -name "All" -text "all" -tcl  {GetScope all}} \
    {qwAction -name "IN" -text "input" -tcl  {GetScope input}} \
    {qwAction -name "OUT" -text "output" -tcl  {GetScope output}}
    \
    {qwAction -name "INOUT" -text "inout" -tcl  {GetScope inout}}
    \
    {qwAction -name "REG" -text "register" -tcl  {GetScope
    register}} \
    {qwAction -name "SIGNAL" -text "signal" -tcl  {GetScope net}}
    \
    {qwActionGroup -name "Display Waveform" -actionNames {"All"
    "IN" "OUT" "INOUT" "REG" "SIGNAL"}} \
    {qwAction -name "Display Waveform" -text "display waveform"}
    \
    {qwAddMenuAction -action "Display Waveform" -group "Inst
    Tree"} \
```

Example 7

This example disables the submenu from Example 2 for all windows of the `nWave` type.

```
qwConfig -type nWave -cmds [list {qwAction -name "mygrp1"
-disabled}]
```

Example 8

This example adds a new action in the toolbar through a Tcl variable. It is recommended to use `[list qwAction ....]` instead of `{qw...}` to make sure tcl variable is interpreted correctly.

```
set act1Name "T1A3Name"
set act1Text "T1A3"
set act1Tcl "puts T1A3"
set toolBarName "CustToolBar1Name"
qwConfig -type nWave -cmds [list
    [list qwAction -name $act1Name -text $act1Text -tcl
    $act1Tcl]]
qwConfig -type nWave -cmds [list
    [list qwAddToolBarAction -action $act1Name -group
    $toolBarName]]
```

qwRemoveMenuGroup

Description

Removes a pull-down menu.

Syntax

`qwRemoveMenuGroup [-id` *`mainwinId`*`] [-group` *`groupName`*`]`

Arguments

`-group` *`groupName`*

Specifies the group (menu) to remove. If not specified, the entire menu bar is removed.

`-id` *`mainwinId`*

Window id of the main window. This option is ignored if it is executed within the context of the `qwConfig` command, otherwise it is required.

Example

```
qwRemoveMenuGroup -id $_nWave2 -group "my_group_1"
```

qwRemoveToolBarGroup

Description

Removes a toolbar.

Syntax

```
qwRemoveToolBarGroup [-id mainwinId] [-group groupName]
```

Arguments

`-group` *groupName*

Specifies the group (toolbar) to remove. If not specified, the entire toolbar is removed. The group/actions on the toolbar are not deleted and are added back again by using `qwAddToolBarGroup` or `qwAddToolBarAction`.

`-id` *mainwinId*

Window id of the main window. This option is ignored if it is executed within the context of the `qwConfig` command, otherwise it is required.

Example

```
qwRemoveToolBarGroup -id $_nWave2 -group "my_group_1"
```

Miscellaneous Commands

Message Command

nsMsgClearMsg

Description

Deletes the selected message in the **Compile**, **Trace**, **Search** or **Interconnection** tabs of the *Message* frame.

Syntax

```
nsMsgClearMsg -tab cmpl|trace|search|intercom -index {levelNum}
```

Arguments

`-index {levelNum}`

Specifies the level for the selected message.

`-tab cmpl|trace|search|intercom`

Specifies the tab in the *Message* frame.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Example

```
nsMsgClearMsg -tab trace -index {0 1 0}
```

nsMsgSave

Description

Saves the contents of a tab in the *Message* frame to the specified file.

Syntax

```
nsMsgSave -file filename [-tab cmpl|trace|search|intercon]
```

Arguments

`-file filename`

Specifies a file to save.

`-tab cmpl|trace|search|intercom`

Specifies the tab in the *Message* frame for the contents to save. When this option is not specified, the active tab is saved.

Value Returned

Returns 1 if successful; Otherwise, returns 0.

Example

```
nsMsgSave -tab trace -file trace.rst
```

Name-Mapping Command

debSetUserNameMapFunc

Description

This command maps the signal name in two Verdi platforms and signals are dragged and dropped between the two Verdi platforms.

Syntax

```
debSetUserNameMapFunc -func myMapFunc
```

Arguments

`-func` *`myMapFunc`*

The customized mapping rules.

Value Returned

String.

Example

Assume that there are two different modules, `top.v` and `testbench.v`:

top.v

```
// arithmetic left shift
module top(in1, imp_out1, imp_out2, imp_out3);

input [3:0] in1;
output [5:0] imp_out1;
output [7:0] imp_out2;
output signed [7:0] imp_out3;

    assign imp_out1 = in1 <<< 2;
    assign imp_out2 = in1 <<< 2;
    assign imp_out3 = in1 <<< 4;

endmodule
```

testbench.v

```
// arithmetic left shift
module testbench(in1, imp_out1, imp_out2, imp_out3);
```

```
input [3:0] in1;
output [5:0] imp_out1;
output [7:0] imp_out2;
output signed [7:0] imp_out3;

    assign imp_out1 = in1 <<< 2;
    assign imp_out2 = in1 <<< 2;
    assign imp_out3 = in1 <<< 4;

endmodule
```

Their Tcl command sets are as follows:

top.tcl

```
proc myMapFunc args {
    puts "myMapFunc"
## $args is the drop string including
##  type ( signal or instance ), name and delimiter
## ex: -signal {top.imp_out1[5:0]} -delim .
## you could puts $args to see the dropped string
## In this regexp, we extract the name part into variable signal
    regexp {(.*)\{(.*)\} (.*)} $args match stam signal

    if { [regexp {testbench(.*)} $signal match orig] == 1 }  {
## replace testbench to top and return
  return top$orig;
    }
    return $args;
}

debSetUserNameMapFunc -func myMapFunc
```

testbench.tcl

```
proc myMapFunc args {
    puts "myMapFunc"
    regexp {(.*)\{(.*)\} (.*)} $args match stam signal

    if { [regexp {top(.*)} $signal match orig] == 1 }  {
    return testbench$orig;
    }
    return $args;
}

debSetUserNameMapFunc -func myMapFunc
```

When you execute the following commands, the after-mapping signals are dragged and dropped between the two Verdi platforms:

```
Verdi top.v -play top.tcl
Verdi testbench.v -play testbench.tcl
```

nTrace

Window

debCloseAllWindows

Description

Close all the windows in Verdi.

Syntax

```
debCloseAllWindows
```

Value Returned

Number of windows closed by this command.

Example

```
debCloseAllWindows
```

debExit

Description

Quit Verdi.

Syntax

```
debExit
```

Value Returned

None.

Example

```
debExit
```

srcCloseWindow

Description

Close the *File Viewer* window.

Syntax

```
srcCloseWindow -win window
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcCloseWindow
```

srcCreateWindow

Description

Create a window and invoke the *File Viewer* window.

Syntax

```
srcCreateWindow
srcCreateWindow -scope scopeName
srcCreateWindow -file filename [startLine] [endLine]
srcCreateWindow -scope scopeName -file filename [startline]
[endline]
```

Arguments

`-scope scopeName`

Specify the scope name to view the window.

`-file filename`

Specify the file information, include file name, and optional start and end line.

Value Returned

File viewer window ID if successful; otherwise, returns 0.

Example

```
set current_window [srcCreateWindow]
srcCreateWindow -scope {system.top}
srcCreateWindow -scope {system.i_cpu.U38} -file cpu.v 100 200
srcCreateWindow -file cpu.v
```

srcGetXWindowId

Description

Obtain the X window ID.

Syntax

```
srcGetXWindowId [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID.

Value Returned

X window ID if successful; otherwise, returns 0.

Example

```
srcGetXWindowId -win $nTrace
```

srcHBDrag

Description

Drag the specified signal, module, or instance from the *Hierarchy Browser* pane.

Syntax

```
srcHBDrag [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
srcHBDrag -win $_nTrace1
```

srcHBDrop

Description

Drop the specified signal, module, or instance to the *Hierarchy Browser* pane.

Syntax

```
srcHBDrop [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
srcHBDrop -win $_nTrace2
```

srcHideHBWindow

Description

Hide the *Hierarchy Browser* pane in the *nTrace* window.

Syntax

```
srcHideHBWindow -on|-off
```

Arguments

`-on|-off`

Specify *on* to hide the *Hierarchy Browser* pane.

Specify *off* to display the *Hierarchy Browser* pane.

Value Returned

1 if successful; 0 otherwise.

Example

```
srcHideHBWindow -on
```

srcLowerWindow

Description

Move the *nTrace* window to the bottom of the window stacking order. If you do not specify the *nTrace* window, it applies to the Verdi main window.

Syntax

```
srcLowerWindow [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcLowerWindow -win $nTrace
```

srcRaiseWindow

Description

Raise the *nTrace* window to the top of the window stacking order. If you do not specify the *nTrace* window, it applies to the Verdi main window.

Syntax

```
srcRaiseWindow [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcRaiseWindow -win $nTrace
```

srcResizeWindow

Description

Resize the *nTrace* window.

Syntax

```
srcResizeWindow x y width height
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcResizeWindow 0 0 400 500
```

srcViewImportLogFile

Description

Open a *File Viewer* window to view the compile log.

Syntax

```
srcViewImportLogFile
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcViewImportLogFile
```

File

debImport

Description

Import the design.

Syntax

```
debImport [-aliasIgnoreHier] [-lib libName] [-noInc] [-recordhlc
cmdFile] [-top topModule(s)] [-vhdl [-93|-vhdl08] | -verilog]
[-vtop file] [-simflow]
```

Arguments

`-aliasIgnoreHier`

Ignore hierarchy differences when applying alias settings to the signals. This option must be used with the `-aliasFile` option.

`-lib libName`

Specify the library name.

`-noInc`

Suppresses the incremental loading.

`-recordhlc cmdFile`

Record the command language history to the specified *cmdFile*.

`-simflow`

Specify that the Verdi Knowledge Database is generated by VC/VCS Native Compile and the library mapping from the *synopsys_sim.setup* setup file is used.

`-top topModule(s)`

Specify the top module name.

`-vhdl [-93|-vhdl08] | -verilog`

Specify the language of the imported design. For VHDL, if **-93** is not specified, use **VHDL-87** as the default version of the VHDL syntax.

`-vtop file`

Specify the virtual top file to import the design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debImport -vhdl -top "system top1" -f run.f
debImport -vhdl -top "system" -f run.f
debImport -vhdl -93 -f run.f
debImport -f run.f
debImport -f run.f -aliasIgnoreHier -aliasFile "signal.alias"
debImport -vhdl -vhdl08 -f run.f -top system
```

debReload

Description

Reload the design and data with all the existing applications.

Syntax

```
debReload
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debReload
```

srcCompile

Description

Compile a Verilog or VHDL design.

Syntax

```
srcCompile -verilog|-vhdl compileOptions designFiles
srcCompileLibMap libName libMappingPath
```

Arguments

`-verilog|-vhdl`

Select the design language to be compiled.

`compileOptions`
The utilities to be selected for compilation.

`designFiles`
The design to be compiled.

`libName`
Specify the library name.

`libMappingPath`
Import the library file.

Value Returned

1 if successful, 0 otherwise

Example

```
srcCompile “-f” “/demo/source/verilog/run.f” -win $_nTrace1 -path
```

srcGetCaretFile

Description

Query the file name where the caret (cursor) is currently located.

Syntax

```
srcGetCaretFile [-win window]
```

Arguments

`-win` *`window`*
Specify the window ID of the invoking source code window.

Value Returned

If there is an empty design, it returns *no file loaded*. If the design is loaded, it returns the file name and path where the caret (cursor) is currently located.

Example

```
srcGetCaretFile -win $_nTrace1
>> no file loaded
>> /home/testcase/system.v
```

srcGetCaretLine

Description

Query the current line number where the caret (cursor) is located.

Syntax

```
srcGetCaretLine [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

If there is an empty design, it returns *-1*. If the design is loaded, it returns the line number where the caret (cursor) is currently located.

Example

```
srcGetCaretLine -win $_nTrace1
>> -1
>> 22
```

srcOpenSrcByImportLog

Description

Open the source code window to the location referenced by the specified line in the log file. The **View Import Log** command should be opened first.

Syntax

```
srcOpenSrcByImportLog [-win window] [-line lineNo]
```

Arguments

`-line lineNo`

The line number of the line in the import log file.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcOpenSrcByImportLog -win $_nTrace2 -line 9
```

srcTBOpenViewLog

Description

Open the source code window to display the simulation view log file.

Syntax

```
srcTBOpenViewLog
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBOpenViewLog
```

debRestoreSession

Description

Restore the status from the session file.

Syntax

```
debRestoreSession sessionFile
```

Arguments

`sessionFile`

Specify the *sessionFile* which is saved by **debSaveSession** command.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debRestoreSession debug.ses
```

debSaveSession

Description

Save current status into a session file.

Syntax

```
debSaveSession [-keynote keynote] sessionFile
```

Arguments

`-keynote`

Specify the keynote shown on the header in the *sessionFile*.

`sessionFile`

Specify the *sessionFile* which is saved by **debSaveSession** command.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debSaveSession  debug.ses
```

debLoadSimResult

Description

Load the specified FSDB file.

Syntax

```
debLoadSimResult fsdbFile [-time timeRange] [-extractSameNet]
```

Arguments

fsdbFile

Specify the FSDB file to be loaded.

-time *timeRange*

Specify the load time range.

-extractSameNet

Specify to extract equivalent signals in the FSDB file. See the **Extract Equivalent Nets** option in the *Load Simulation Results* form invoked by the **File -> Open Waveform File** command.

Value Returned

Loaded FSDB file if successful; otherwise, returns 0.

Examples

```
debLoadSimResult test.fsdb -extractSameNet
```
Load *test.fsdb* that extracts all the same net in the design.

debAppendSimResult

Description

Append one or more FSDB files in Verdi.

Syntax

```
debAppendSimResult fsdbFileName
```

Arguments

fsdbFileName

The complete path of the secondary FSDB file name.

Value Returned

Loads the FSDB file if successful; otherwise, returns 0.

Example

```
debAppendSimResult /qa/home/chsu/hdl_only_2.fsdb
```

debCloseSimResult

Description

Close the loaded FSDB file.

Syntax

```
debCloseSimResult
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debCloseSimResult
```

debGetImportError

Description

Get number of errors resulted from the import command.

Syntax

```
debGetImportError
```

Value Returned

Number of errors.

Example

```
set nErrors [debGetImportError]
```

debGetImportWarning

Description

Get number of warnings resulted from the import command.

Syntax

```
debGetImportWarning
```

Value Returned

Number of warnings.

Example

```
set nWarn [debGetImportWarning]
```

srcGetDefLine

Description

Get the definition line number of the input HDL object.

Syntax

```
srcGetDefLine ObjectName [-delim delimiter]
```

Arguments

`[-delim delimiter]`

Specify the delimiter for the input project name. The default is '.'.

`ObjectName`

Specify the input object with the complete hierarchy name.

Value Returned

The definition line number of the input HDL object.

NOTE: If the input is not a legal HDL object, the following string is the output: Enter a legal hierarchical name. If the input HDL object is not specified in the *srcGetDefLine -delim* format, the usage of the **srcGetDefLine** command is the output.

Example

```
srcGetDefLine "top.a" -delim "/"
```

srcGetFileName

Description

Get the source file name of the input HDL object.

Syntax

```
srcGetDefLine ObjectName [-delim delimiter]
```

Arguments

`[-delim delimiter]`

Specify the delimiter for the input project name. The default is '.'.

`ObjectName`

Specify the input object with the complete hierarchy name.

Value Returned

The string of the file name of the input HDL object.

NOTE: If the input is not a legal HDL object, the following string is the output: Enter a legal hierarchical name. If the input HDL object is not specified in the *srcGetFileName -delim* format, the usage of the **srcGetFileName** command is the output.

Example

```
srcGetFileName "top.a" -delim "/"
```

srcSaveAs

Description

Save current content in the *nEditor* window into a file.

Syntax

```
srcSaveAs -win window -file filename
```

Arguments

`-file filename`

Specify the file name which is saved by **srcSaveAs** command.

`-win window`

Specify the window ID of the invoking *nEditor* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSaveAs -win $_nTrace1 -file tmp.v
```

Scope

srcSetScope

Description

Set the active scope.

Syntax

```
srcSetScope [-win window] [-delim delim] [-file includeFile]
scopename
```

Arguments

`-delim delim`

Delimiter of the hierarchical name.

`-file`

If the file is specified after the switched scope and if the specified file is the included file within this scope, the source code window displays the source code of this specified included file.

`scopename`

Specify the active scope.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetScope -delim {/} {system/I_cpu/I_CCU}
srcSetScope -delim {/} {system/I_cpu/I_CCU} -file {textconfig.v}
```

srcGetScope

Description

Get the current active scope.

Syntax

```
srcGetScope [-win window]
```

Arguments

`-win` *window*

Specify the window ID of the invoking source code window.

Value Returned

Current active scope name.

Example

```
set active_scope [srcGetScope]
```

Variable

srcDumpVariableValue

Description

Dump the variable value.

Syntax

```
srcDumpVariableValue [-win window] lineNumber [-snapShot]
```

NOTE: The `lineNumber` must be specified before the `-snapShot` option.

Arguments

`lineNumber`

Dump value of the variables on the specified line.

`-snapShot`

Flag to only dump value for the current time.

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcDumpVariableValue 50 -snapShot
```

Trace

srcActiveTrace

Description

Find the real driver(s) of the specified signal. When the signal is not specified, the action is applied to the selected signal.

Syntax

```
srcActiveTrace [-delim delim] [-win window] [-r] [-range start end]
[signalName] [-TraceByDConWave] [-TraceTime] [-TraceValue]
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the name list.

`-r`

Relate the *signalName* to the current scope.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal.

`signalName`

Specify the signal name.

`-TraceByDConWave`

Execute active trace by double-clicking the *nWave* signal.

`-TraceTime`

Execute active trace by double-clicking the specified cursor time.

`-TraceValue`

Execute active trace with the specified signal value.

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

A list of scope and file/line information for the current traced result if successful; otherwise, returns 0.

Example

```
srcActiveTrace -range 1 2 {system.CH[4:0]}
set tracelist [srcTraceDriver -r {CH[4:0]}]
(tracelist will be {system} {system.v} {100} )
```

srcBackwardHistory

Description

Trace the history backward and change the view accordingly.

Syntax

```
srcBackwardHistory [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcBackwardHistory
```

srcForwardHistory

Description

Trace the history forward and change the view accordingly.

Syntax

```
srcForwardHistory [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcForwardHistory
```

srcGetConnectedSignalList

Description

Get all the trace connectivity signals.

Syntax

```
srcGetConnectedSignalList [signalName]
```

Arguments

`signalName`

Specify the signal to get the connected signal list with full hierarchy name.

Value Returned

The full hierarchy name of all the connectivity signals.

Example

```
srcGetConnectedSignalList "system.clock"
```

Generates the following output:

```
{ {system.i_cpu.clock} {system.CHILD3.Clock}
{system.CHILD2.Clock} {system.CHILD1.Clock}
{system.i_cpu.i_ALUB.clock}
{system.clock} {system.i_cpu.i_CCU.clock} {system.MASTER.Clock}
{system.i_pram.clock} }
```

srcNextTraced

Description

View next instance or replace the window with the next scope, which contains the trace result.

Syntax

```
srcNextTraced [-win window] [-scope]
```

Arguments

`-scope`

If specified, replace the window with the next scope with traced result.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

When `-scope` is specified, a list of scope and instance name if successful; otherwise, returns 0.

Example

```
srcNextTraced
srcNextTraced -scope
```

srcPrevTraced

Description

View previous instance or replace the window with previous scope in the traced result.

Syntax

```
srcPrevTraced [-win window] [-scope]
```

Arguments

`-scope`

If specified, replace the window with the previous scope with traced result.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

When `-scope` is specified, a list of scope and instance name if successful; otherwise, returns 0.

Example

```
srcPrevTraced
srcPrevTraced -scope
```

srcResetHistory

Description

Reset all the trace history.

Syntax

```
srcResetHistory [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcResetHistory
```

srcResetTracedMarkColor

Description

Reset the mark color of the traced result.

Syntax

```
srcResetTracedMarkColor [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcResetTracedMarkColor
```

srcSaveHistoryToFile

Description

Save the trace history to the specified file.

Syntax

```
srcSaveHistoryToFile [-win window] filename listHistoryid
```

Arguments

`filename`

Specify the output file name.

`listHistoryid`

Specify list of the history ID shown on the traced result on the message window.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSaveHistoryToFile trace.out 1 3 5
```

srcTraceConnectivity

Description

Find the connectivity(s) of the specified signal. If the signal is not specified then the action is applied to the selected signal.

Syntax

```
schTraceConnectivity [-delim delim] [-win window] [-r] [-range
start end] [signalName]
```

Arguments

`-delim delim`

Delimiter of the name list.

`-r`

If specified, the **signalName** is related to the current scope.

`-range start end`

Specify the index range for tracing the partial bit range for the specified bus signal.

`signalName`

Specify the signal name.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

A list of scope and instance name for the current traced result if successful; otherwise, returns 0.

Example

```
srcTraceConnectivity -range 1 2 {system.CH[4:0]}
srcTraceConnectivity -r {CH[4:0]}
```

srcTraceDriver

Description

Find the driver(s) of the specified signal. If the signal is not specified then the action is applied to the selected signal.

Syntax

```
schTraceDriver [-delim delim] [-win window] [-r] [-range start end]
[signalName]
```

Arguments

`-delim delim`

Delimiter of the name list.

`-r`

If specified, the **signalName** is related to the current scope.

`-range start end`

Specify the index range for tracing the partial bit range for the specified bus signal.

`signalName`

Specify the signal name.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

A list of scope and file/line information for the current traced result if successful; otherwise, returns 0.

Example

```
srcTraceDriver -range 1 2 {system.CH[4:0]}
set tracelist [srcTraceDriver -r {CH[4:0]}]
(tracelist will be {system} {system.v} {100} )
```

srcTraceFanIn

Description

Find the fan-in register(s) for the selected signal.

Syntax

`srcTraceFanIn -win window [fullHierarchicalSignalName]`

Arguments

`fullHierarchicalSignalName`

Specify the signal name with full hierarchical path.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTraceFanIn "system.data\[7:0\]" -win $nTrace1
```

srcTraceFanOut

Description

Find the fan-out register(s) for the selected signal.

Syntax

```
srcTraceFanOut -win window [fullHierarchicalSignalName]
```

Arguments

`fullHierarchicalSignalName`

Specify the signal name with full hierarchical path.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTraceFanOut "system.data\[7:0\]" -win $nTrace1
```

srcTraceLoad

Description

Find the load(s) of the specified signal. If the signal is not specified then the action is applied to the selected signal.

Syntax

```
srcTraceLoad [-delim delim] [-win window] [-r] [-range start end]
[signalName]
```

Arguments

`-delim delim`

Delimiter of the name list.

`-r`

If specified, the **signalName** is related to the current scope.

`-range start end`

Specify the index range for tracing the partial bit range for the specified bus signal.

`signalName`

Specify the signal name.

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

A list of scope and instance name for the current traced result if successful; otherwise, returns 0.

Example

```
srcTraceLoad -range 1 2 {system.CH[4:0]}
srcTraceLoad -r {CH[4:0]}
```

Radix

srcSetRadix

Description

Set radix for displaying the signal values.

Syntax

```
srcSetRadix [-win window] [notation] -format Bin|Oct|Hex|Dec|ASCII
```

Arguments

`-format Bin|Oct|Hex|Dec|ASCII`

Specify the signal value format from one of the following:
Bin | Oct | Hex | Dec | ASCII

`notation`

Specify the signal value notation style from one of the following:
-Signed | -Unsigned | -2Com | -1Com | -Mag

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetRadix -format Hex
```

srcSetParaRadix

Description

Set radix for displaying parameter values.

Syntax

```
srcSetParaRadix [-win window] [notation] -format
Bin|Oct|Hex|Dec|ASCII
```

Arguments

`notation`

Specify the signal value notation style from one of the following: -Signed | -Unsigned | -2Com | -1Com | -Mag

`-format Bin|Oct|Hex|Dec|ASCII`

Specify the signal value format from one of the following: Bin | Oct | Hex | Dec | ASCII

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetParaRadix -format Hex
```

srcAddAliasFile

Description

Add alias from file.

Syntax

```
srcAddAliasFile [-win window] filename
```

Arguments

`filename`

Alias file name.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAddAliasFile -win $current_window cpu.alias
```

srcAddAliasProgram

Description

Add alias from an executable program.

Syntax

```
srcAddAliasProgram [-win window] programName
```

Arguments

`programName`

Specify the program name.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAddAliasProgram aliasprogram
```

srcRemoveAlias

Description

Remove the alias.

Syntax

```
srcRemoveAlias [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcRemoveAlias
```

Select

srcAction

Description

Execute the double-click action.

Syntax

```
srcAction -pos x y -cmdMessage -win window
```

Arguments

`-cmdMessage`

Display the command message.

`-pos x y`

The selected position on the geometry of the window.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAction -pos 176 1 -cmdMessage -win $_nTrace1
```

srcAddSelectedToWave

Description

Add all the selected instances or signals to the synchronized *nWave* window.

Syntax

```
srcAddSelectedToWave [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAddSelectedToWave -win $window
```

srcDeselectAll

Description

Deselect all the selected instances or signals on the specified window.

Syntax

```
srcDeselectAll [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcDeselectAll -win $current_window
```

srcGetSelectSet

Description

Query back all objects in select set. The objects supported include:

- Property signal, include OVA/PSL assert, event, and boolean objects.
- Property scope, include OVA/PSL scope, and module objects.

Syntax

```
srcGetSelectSet [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

Selected instances, signals, scopes, and signals of property of assertion if successful; otherwise, returns 0.

The format of selected objects are as follows:

```
{instance instance_count {instnace_list}}
{signal signal_count {signal_list}}
{prop_scope prop_scope_count {Prop_scope_list}}
{prop_signal prop_signal_count {prop_signal_list}}
```

Example

```
srcGetSelectSet -win $_nTrace2
```

Value Returned Examples

1. Both instances and signals are selected:

   {instance 1 {{system.i_cpu.i_PCU}}} {signal 2 {{system.i_cpu.S1} {system.i_cpu.IR[1:0]}}}

2. Only instances are selected:

   {instance 2 {{system.i_cpu.i_PCU} {system.i_cpu.i_CCU}}}

3. Only signals are selected:

   {signal 2 {{system.i_cpu.clock} {system.i_cpu.reset}}}

4. Scopes and signals of assertion property are selected:

   ```
   {prop_scope 1 {{ALU_test_top.dut}}} {prop_signal 2
   {{ALU_test_top.dut.assert_finish_div_in_time}
   {ALU_test_top.dut.event_tc4}}}
   ```

srcHBGetSelSet

Description

Get current selected set in the *Hierarchy Browser* pane.

Syntax

`srcHBGetSelSet [-win` *`window`*`]`

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

Number of selected node, type information, and selected module if successful.

The format of the selected node is as follows:

```
{num of select sets{type_info{node Name}{hierarchy path
information}}}
```

The supported type information is as follows:

- forbid
- check
- cover
- vunit
- psl_property

Example

```
srcHBSelect "system.MASTER" -win $_nTrace1 -prop state_seq
srcHBGetSelSet -win $_nTrace1
```

Value Returned Example

```
{ 1 {check{state_seq}{system.MASTER}}}
```

srcHBSelect

Description

Select the tree node in the design hierarchy tree.

Syntax

`srcHBSelect -win` *`window scopeName`*

`srcHBSelect -win` *`window scopeName`* `-prop` *`propName`*

If you execute the select command on a property tree node.

Arguments

`-prop` *`propName`*

Specify the property (OVA or PSL) tree node type to be selected.

`-win` *`window scopeName`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
srcHBSelect -win $_nTrace1 "system.top.i_CPU"

srcHBSelect "system.MASTER" -win $_nTrace1 -prop state_seq

srcHBSelect -win $_nTrace1 "system.i_cpu" -prop
cpu_scope_i_cpu_check2
```

srcHBSetMultiSel

Description

Allow multiple selections in the hierarchical browser.

Syntax

```
srcHBSetMultiSel [0 | 1]
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcHBSetMultiSel 1
srcHBSetMultiSel 0
```

srcSelect

Description

Select the instances or signals in a window.

Syntax

```
srcSelect [-win window] [-toggle] -inst listInstname
srcSelect [-win window] [-toggle] [-line lineNumber]
[-pos nthSignal] -signal listSignalname
srcSelect [-win window] [-file filename] -range listLineRange
srcSelect [-win window] -word -line lineNumber
-pos wordPositionInLine
```

Arguments

`-file filename`

Change the source code to display the contents of the specified file.

`-inst listInstname`

Specify the local instance name list.

`-line lineNumber`

Specify the line to be selected.

`-pos nthSignal`

Specify the position of the selected signal.

`-pos wordPositionInLine`

Specify the position of the word in the line.

`-range listLineRange`

Specify list of the line range in the {startline endline [#of the startline] [#of the endline]}.

`-signal listSignalname`

Specify the local signal name list. When the `-line` option is specified, only one specified signal name is allowed.

`-toggle`

Toggle the selected instances or signals.

`-win window`

Specify the window ID of the invoking source code window.

`-word`

Specify the word to be selected or unselected.

Value Returned

Number of objects selected.

Example

```
srcSelect -win $current_window -toggle -signal {S1} {reset}
{clock} -inst {i_ALUB}
srcSelect -inst {regA} {regB}
srcSelect -line 100 {TOP.cpu.clock}
(for the external reference signal select scheme)
srcSelect -line 100 -pos 2 {a}
(assign o1 = (a & b) | (a & c) -> the second a is selected)
srcSelect -range { 100 200 2 200 4} {240 280}
srcSelect -word -line 20 -pos 3 -win $_nTrace2
srcSelect -file "master.v" -range {100 105}
```

nBench

srcOpenHVLDrvLoadTable

Description

Invoke a new *Trace Driver/Load Table*.

Syntax

```
srcOpenHVLDrvLoadTable [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nBench* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcOpenHVLDrvLoadTable -win $_nTrace1
```

hvlBrowserSignalInvoke

Description

Invoke a new *Browse Signals* window.

Syntax

```
hvlBrowserSignalInvoke [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nBench* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
hvlBrowserSignalInvoke -win $_nTrace2
```

hvlBrowserSignalSet

Description

Browse all signals in the testbench.

Syntax

```
hvlBrowserSignalSet -scope scopePath -signal signalPath
-time time -annot on|off
```

Arguments

`-scope scopePath`

Set new active scope to the scope tree.

`-signal signalPath`

Highlight the signal in the *Signal List* pane.

`-time time`

Set the current tag time.

`-annot on|off`

Set the annotation *on* or *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
hvlBrowserSignalSet -scope "sys" -annot FALSE
```

hvlBrowserSignalAdd

Description

Add the selected signal(s) in the *Signal List* pane to the *Hierarchical Signal Tree* pane.

Syntax

```
hvlBrowserSignalAdd -signal signalPath
```

Arguments

`-signal` *`signalPath`*

Add the signal to the *Signal List* pane.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
hvlBrowserSignalAdd -signal \
           "sys.pci_transaction.\$cover_initiate" -signal \
           "sys.pci_transaction.address" \
           -signal "sys.pci_transaction.bus_id" \
           -signal "sys.pci_transaction.command" -signal \
           "sys.pci_transaction.data" -signal \
           "sys.pci_transaction.dual_address" -signal \
           "sys.pci_transaction.initiate" \
           debLoadSimResult \
           /qa/ddts_case/SPSqa52024/testcase04/hvl.fsdb
```

hvlBrowserSignalDumpAu

Description

Dump the signal data in the *Browse Signals* window.

Syntax

```
hvlBrowserSignalDumpAu
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
hvlBrowserSignalDumpAu
```

HvlTraceHDLDriver

Description

Trace driver from HVL to HDL.

Syntax

```
HvlTraceHDLDriver -win window -signal signalPath
```

Arguments

`-signal signalPath`

The full path of HVL signal.

`-win window`

Specify the window ID of the invoking *nBench* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
HvlTraceHDLDriver -win $_nTrace2 -signal "sys.f1.sig"
```

HvlTraceHDLLoad

Description

Trace load from HVL to HDL.

Syntax

```
HvlTraceHDLLoad -win window -signal signalPath
```

Arguments

`-signal signalPath`

The full path of the HVL signal.

`-win window`

Specify the window ID of the invoking *nBench* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
HvlTraceHDLLoad -win $_nTrace2 -signal "sys.f1.sig"
```

srcHVLDrvLoadTableSet

Description

Specify the HVL DriverLoad table in the current scope.

Syntax

```
srcHVLDrvLoadTableSet -scope scopepath
```

Arguments

`-scopepath`

Specify the complete hierarchical path for a scope.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcHVLDrvLoadTableSet  -scope "sys.cpu_duv"
```

srcHVLDrvLoadTableDump

Description

Dump all data in the HVL DriverLoad table.

Syntax

```
srcHVLDrvLoadTableDump
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcHVLDrvLoadTableDump
```

OVA/PSL

debLoadPropFiles

Description

Load OVA/PSL file(s).

Syntax

```
debLoadPropFiles -f File1[File2 File3 ...]
```

Arguments

`File1`

The first OVA/PSL source file.

`File2 File3...`

(Optional arguments) the 2nd, 3rd, OVA/PSL source files, and so on.

Value Returned

Loaded OVA/PSL file name if successful; otherwise, returns 0.

Example

```
debLoadPropFiles -f /verify1/regression/demo/ova/verilog/alu.ova
\
            /verify1/regression/demo/ova/verilog/alub.ova
```

srcOpenPropViewer

Description

This command is used to log and execute the double-click action on an assertion in the hierarchy and the right-click **Show Definition** and **Show Definition in New Window** commands.

Syntax

`srcOpenPropViewer [-ova|-psl] [-primary] [-scope` *`scopeName`*`] [-prop` *`propValue`*`|-propfullname` *`propFullName`*`] [-setscope]`

Arguments

`-ova|-psl`

Specify if you are opening an OVA or PSL file.

`-primary`

This is an optional argument. If used, *nTrace* opens the property source file in the primary OVA/PSL file viewer.

`-scope` *`scopeName`*

Specify design scope. The option must be followed by design scope full hierarchical path.

`-prop` *`propValue`*`|-propfullname` *`propFullName`*

Specify the property value or the full property name for the design scope.

`-setscope`

Set the active scope.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcOpenPropViewer -ova -scope "system.MASTER" -prop "state_seq"
-setscope -primary
```

srcGotoPropObj

Description

It is used to record and play the command **Show Definition** of OVA/PSL assert, event, and boolean object.

Syntax

`srcGotoPropObj -win` *`window`* `-scope` *`scopePath`* `-prop` *`propObj`*

Arguments

`-win` *`window`*

Specify the window ID of the invoking OVA/PSL file viewer.

`-scope` *`scopePath`*

Specify the scope to which the OVA/PSL object belongs.

`-prop` *`propObj`*

Specify the name of the OVA/PSL object.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcGotoPropObj -win $_nTrace1 -scope system.i_alu -prop
test_logic.gt.a_cmp
```

SDF

debLoadSDFFile

Description

Load the SDF file.

Syntax

```
debLoadSDFFile -delim delim [-design topDesignName] sdfFile
[-writeDDBDir dirName] -checkUndef
```

Arguments

`-checkUndef`

Check the undefined SDF value when importing the SDF file.

`-delim` *`delim`*

Specify the delimiter of the imported design.

`-design` *`topDesignName`*

Relate the loaded SDF design to the specified design name.

`sdfFile`

Specify the SDF file name.

`-writeDDBDir` *`dirName`*

Specify the directory where the *.ddb* file is the output.

NOTE: The user must have write permission for the specified directory.

Value Returned

Loaded SDF file name if successful; otherwise, returns 0.

Example

```
debLoadSDFile -design {system.top} cpu.sdf
debLoadSDFFile cpu.sdf -writeDDBDir "/tmp"
```

debCloseSDFFile

Description

Close the loaded SDF file.

Syntax

```
debCloseSDFFile
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debCloseSDFFile
```

srcViewSDFLog

Description

Create a *File Viewer* window to view the log of SDF file.

Syntax

```
srcViewSDFLog -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcViewSDFLog -win $_nTrace1
```

View

debSignalAlias

Description

List the same signals across hierarchy of the specified signal.

Syntax

```
debSignalAlias [-delim delim] [-snapshot] -tofile filename
signalName
```

Arguments

`-delim delim`

Delimiter of the signal name.

`-tofile`

Specify the output file name.

`-snapShot`

If specified, only dump current time.

`signalName`

Specify signal name.

Value Returned

of the same signals if successful; otherwise, returns 0.

Example

```
set nalias [debSignalAlias -delim .  {system.clk2} -tofile {/dar/
integ/aliasnet.txt}
($nalias is 20)
```

srcAddBookMark

Description

Create the bookmark on the source code window.

Syntax

```
srcAddBookMark [-win window] filename lineNumber
```

Arguments

`filename`
Specify the file name.

`lineNumber`
Specify the line number.

`-win window`
Specify the window ID of the invoking source code window.

Value Returned

Index of bookmark if successful; otherwise, returns 0.

Example

```
set bm1 [srcAddBookMark {cpu.v} 10]
```

srcDelBookMark

Description

Delete the bookmark from the source code window.

Syntax

```
srcDelBookMark [-win window] [-all] filename lineNumber
```

Arguments

`-all`
If specified, all the bookmarks are deleted from the window.

`filename`
Specify the file name.

`lineNumber`
Specify the line number.

`-win window`
Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcDelBookMark "cpu.v" 100
```

srcDisplaySubprog

Description

Set the option whether to display sub-program on the design tree.

Syntax

```
srcDisplaySubprog [-win window] -subProgram on|off
```

Arguments

`-subProgram`

Option to display sub-program on the design tree.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcDisplaySubprog  -win $window  -subProgram on
```

srcDumpPara

Description

Dump all HDL parameter values under the specified instance (scope). This command is only for Verilog designs.

Syntax

```
srcDumpPara -inst instName -delim delimiter
```

Arguments

`-inst instName`

Specify the instance name.

`-delim` *`delimiter`*

Specify the signal name delimiter. "." is used if the delimiter is not specified.

Value Returned

Get parameters if successful; otherwise, returns 0.

Example

```
srcDumpPara -inst system*CHILD1 -delim “*” {{ST_A 0} {ST_B 1}
{ST_C 2} {ST_D 3}}
srcDumpPara -inst system*CHILD2 -delim {{Zero 0} {One 1} {Two 2}
{Three 3} {Four 4} {Five 5}}
```

srcDumpUserDefColorMap

Description

Dump user define color map.

Syntax

`srcDumpUserDefColorMap -win` *`window`*

Argument

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcDumpUserDefColorMap -win $_nTrace1
```

srcDumpVariableValue

Description

Dump the value of the process variable.

Syntax

```
srcDumpVariableValue -win window lineNumber
```

Arguments

lineNumber

The line number of the invoking line.

-win *window*

Specify the window ID of the invoking source code window.

Value Returned

of variables.

Example

```
srcDumpVariableValue 100
```

srcExportHier

Description

Dump design hierarchy tree to ASCII file.

Syntax

```
srcExportHier -win window [-fullHierarchy] -file filename
[-ignoreTable] [-xprop]
```

Arguments

-fullHierarchy

If not specified, only dump current expanded tree hierarchy.

-file *filename*

Specify the output file name.

-ignoreTable

If specified, the node mapping table of the exported ASCII text file is not dumped.

`-win` *`window`*

Specify the window ID of the invoking source code window.

`-xprop`

If specified, the x-propagation information is saved to the output file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcExportHier -win $_nTrace1 -file tree.log -ignoreTable
srcExportHier -win $_nTrace1 -xprop -file tree.log
```

srcFindScope

Description

Performs relevant actions in the **Source -> Find Scope** command in the *nTrace* window.

Syntax

```
srcFindScope -open|-close|-sortby Alphabet|Call Level|
-selectscope rowIndex|-selectinst rowIndex|
-filter filterString -case on|off|-saveas file|
-scope File|Task|Function|Module
```

Arguments

`-case on|off`

Specify whether filtering the scopes with the specified string is case-sensitive. The default is *on* (case-sensitive).

`-close`

Close the *Find Scope* form.

`-filter` *`filterString`*

Specify the filter string.

`-open`

Open the *Find Scope* form.

`-sortby Alphabet|Call Level`

Specify whether to sort by **Alphabet** or **Call Level** type.

`-saveas` *`file`*

Save as an instance list for the selected scope.

`-selectinst` *`rowIndex`*

Select one instance from the instance list.

`-selectscope` *`rowIndex`*

Select the scope from the scope list. Multiple selection for the scope is allowed.

`-scope File|Task|Function|Module`

Select the scope type from the type list. The types include: **File**, **Task**, **Function**, and **Module**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcFindScope -open
srcFindScope -close
srcFindScope -sortby "Alphabet"
srcFindScope -sortby "Call Level"
srcFindScope -selectscope 1
srcFindScope -selectinst 1
srcFindScope -filter "*p"
srcFindScope -saveas scope.log
srcFindScope -scope "File"
srcFindScope -scope "Task"
srcFindScope -scope "Function"
srcFindScope -scope "Module"
srcFindScope -filter "a*" -case on
```

srcFindScopeGoto

Description

Performs *nTrace*'s **Source -> Find Scope -> Go To** actions. Going to the file name of the hierarchy browser pane or the scope name in the *nTrace* source code pane works.

Syntax

```
srcFindScopeGoto [-win window] -scope scopeName|-file filename -
pos posIndex
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`-scope scopeName`

The selected scope name on the source code window.

`-file filename`

The selected file name on the hierarchy browser window.

`-pos`

The selected position on the source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcFindScopeGoto -win $_nTrace1 -file system.v -pos 1
srcFindScopeGoto -win $_nTrace2 -scope system.i_cpu.i_ALUB
```

srcGetBookMark

Description

Get the information of the specified bookmark.

Syntax

```
srcGetBookMark [-win window] index
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`index`

Bookmark index returned from the **srcAddBookMark** command.

Value Returned

Bookmark information (file line) if successful; otherwise, returns 0.

Example

```
set bm [srcGetBookMark 1]
```

srcGetBookMarkNum

Description

Get the total number of the bookmarks on the window.

Syntax

```
srcGetBookMarkNum [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

Number of bookmarks if successful; otherwise, returns 0.

Example

```
set nBM [srcGetBookMarkNum]
```

srcGotoBookMark

Description

Go to the specified bookmark.

Syntax

```
srcGotoBookMark [-win window] index | -name label
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

`index`

Bookmark index returned from the **srcAddBookMark** command.

`-name label`

Bookmark name named from the **srcLabelBookMark** command.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcGotoBookMark -win $_nTrace1 3
srcGotoBookMark -win $_nTrace1 -name "AAA"
```

srcGotoCurStatement

Description

Go to the statement the simulator is currently stopped at.

Syntax

`srcGotoCurStatement [-win` *`window`*`]`

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcGotoCurStatement
```

srcGotoInstPort

Description

Go to the statement where the signal upper level instantiated.

Syntax

```
srcGotoInstPort [-win window] [-delim delim] fullSignalName
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`-delim delim`

Delimiter of the scope name.

`fullSignalName`

Hierarchical signal name.

srcGotoLine

Description

Go to the specified line.

Syntax

```
srcGotoLine [-win window] [-select] lineNumber [-setActive]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`-select`

Whether to select the line after jump to the line.

`lineNumber`

Specify the line number.

`-setActive`

Set the scope where the line jumped to an active scope.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcGotoLine 100
srcGotoLine -win $_nTrace1 10 -setActive
```

srcGoto1stExec

Description

Go to the first executable line.

Syntax

```
srcGoto1stExec [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcGoto1stExec
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcGotoInstPort
```

srcHBShowInfo

Description

Show the design tree information.

Syntax

```
srcHBShowInfo -win window
```

Argument

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcHBShowInfo -win $_nTrace1
```

srcHBShowLibCell

Description

Display the library cell nodes for some library cell set.

NOTE: The Tcl command logs the path name which is the upper node of the library cell set.

Syntax

```
srcHBShowLibCell [full path of parent of library set node] | [-win window]
```

Argument

`full path of parent of library set node`

Specify the full path of the parent of the library set node.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcHBShowLibCell “system.i_cpu.i_CCU" -win $_nTrace1
```

srcLabelBookMark

Description

Label the bookmark with an alias string.

Syntax

```
srcLabelBookMark [-win window] index labelString
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`index`

Bookmark index returned from the **srcAddBookMark** command.

`labelString`

Specify the string for the bookmark.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcLabelBookMark 1 "CPU_test"
```

srcPopViewUp

Description

Pop view up from port. The upper connected instance port is selected.

Syntax

```
srcPopViewUp [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcPopViewUp
```

srcPushViewIn

Description

Push view in from the instance port. The lower connected port is selected.

Syntax

```
srcPushViewIn [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcPushViewIn
```

srcSchematicView

Description

Show schematic (as opposed to source code) in the *nTrace* window.

Syntax

```
srcSchematicView
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSchematicView
```

srcSearchString

Description

Search the string on the specified window.

Syntax

```
srcSearchString string [-win window] [-range {location}|-allfiles]
[-next|-prev] [-case] [-tofile file] [-fileType filterString]
```

Arguments

`-allFiles`

Specify to search through all files.
This option is only for the *nTrace* source code window, not for the *File Viewer* window.

`-fileType filterString`

Specify a string to filter files.

`-case`

Perform case-sensitive searching.

`-next`

Search forward through the file.

`-prev`

Search backward through the file.

`-range {location}`

Specify to search in the current file.

`string`

Specify the search range.

`-tofile file`

When specified, the found file line information is the output to this specified file.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

File line information if successful; otherwise, returns 0.

Example

```
srcSearchString "system" -win $_nTrace1 -next -case -allfiles -
fileType "*.sv;*.vs"
```

srcShowCalling

Description

Jump to the instantiation statement for the scope.

Syntax

```
srcShowCalling [-win window] [-delim delim] [scopeName]
[-listAllInst]
```

Arguments

`-delim delim`

Scope name delimiter.

`-listAllInst`

Jump to the statement where the module is instantiated and show all instances of the module in the *nTrace*'s message pane.

`scopeName`

When specified, jump to the scope's instance statement. If not, jump to the instance statement of the current active scope.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowCalling {system.i_ALUB.i_cpu}
srcShowCalling -win $_nTrace1 -listAllInst
srcShowCalling -win $_nTrace1 "system.i_cpu" -listAllInst
```

srcShowDeclaration

Description

Show a variable declaration.

Syntax

```
srcShowDeclaration [-win window] varPath
```

Arguments

`varPath`

The full hierarchy path of a variable path.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowDeclaration -win $_nTrace2 sys.p.a
```

srcShowDefine

Description

Jump to the module or architecture statement that defines the master for the instantiation statement.

Syntax

```
srcShowDefine [-win window] [-delim delim] [-r] [instanceName]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`instanceName`

If specified, jump to the master of this instance name. If not, use the selected instance on the source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowDefine {system.i_ALUB.i_cpu}
```

srcShowEntity

Description

Jump to the entity statement for the architecture.

Syntax

```
srcShowEntity [-win window] [architectureName]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`architectureName`

If specified, jump to the entity of the specified architecture. If not, jump to the entity of the selected architecture.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowEntity {alub(alub)}
```

srcShowFile

Description

Show the specified file information on the window.

Syntax

```
srcShowFile [-win window] [-scope scopeName] -file filename [-line
lineNumber]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`-scope scopeName`

If specified, you can use this as the scope of this window. If not, you can choose a random scope as the default scope.

`-file filename`

Specify the full path name of the file to be viewed.

`-line`

If specified, it jumps to the specified line. If not, it jumps to the first executable line of the file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowFile -scope {system.i_cpu} -file cpu.v -line 100
srcShowFile -file cpu.v
```

srcShowReference

Description

Show all references in the message window.

Syntax

```
srcShowReference [-win window] sys.tmp_node
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

`sys.tmp_node`

The full hierarchy path of an instance.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowReference -win $n_Trace2 "sys.tmp_node"
```

srcShowSignalDefine

Description

Jump to the signal declaration line of the selected signal.

Syntax

```
srcShowDefine [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowSignalDefine
```

srcShowSignalType

Description

Jump to the signal type declaration line of the selected signal.

Syntax

```
srcShowSignalType [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowSignalType -win $_nTrace1
```

srcShowSymbol

Description

Add text description or symbol description for annotation in the *nTrace* source code pane indicator area.

Syntax

```
srcShowSymbol -win windowId -collect "{filename lineNum scope}"
-text textString|-symbol symbolString
```

Arguments

`-win window`

Specify the window ID.

`-collect "{filename1 lineNum1 scope1}{filename2 lineNum2 scope2}... {filenameN lineNumN scopeN}"`

Specify one or more combinations containing the file name, the line number and the full scope path. Each combination needs to be surrounded by curly {} braces and the entire set needs to be in double quotes ("").

`-text textString`

Specify the text string to annotate in the indicator area for the location specified in the `-collect` option. The maximum text string is 4 letters. A string over 4 letters is truncated.

This option may also be simultaneously specified with `-symbol` option.

`-symbol symbolString`

Specify the symbol to annotate in the indicator area for the location specified in the `-collect` option. A string needs to be selected from the following selection:

```
cycle, halfArrow, trapezoid, flag, rect,
plusSquare, minusSquare, leftHalfCircle,
rightHalfCircle.
```

This option may also be simultaneously specified with `-text` option.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowSymbol -win $_nTrace1 -collect "{CPU.v 15 system.i_cpu}" -
text bkma

srcShowSymbol -win $_nTrace1 -collect "{CPU.v 15 system.i_cpu}" -
symbol cycle

srcShowSymbol -win $_nTrace1 -collect "{CPU.v 10 system.i_CPU}
{CPU.v 11 system.i_CPU}" -symbol trapezoid
```

srcSignalView

Description

Open the signal list pane in *nTrace* window to show signals in the selected module.

Syntax

```
srcSignalView -on|-off
```

Argument

`-on|-off`

Show or hide the *Signal List* pane in the *nTrace* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalView -on
```

srcSignalViewSelectAll

Description

Select all the signals in the current page in the *Signal List* pane.

Syntax

srcSignalViewSelectAll –curPage

Argument

`-curPage`

Specify the current page to select.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewSelectAll –curPage
```

srcSignalViewGetSelectSet

Description

Query the selected signals in the *Signal List* pane.

Syntax

```
srcSignalViewGetSelectSet
```

Value Returned

Returns the *Hierarchy Name* of the selected signal if successful, else returns 0.

Examples

```
srcSignalViewGetSelectSet
```

When no signal is selected to query, the return value is 0.

When one signal is selected to query, the return value is:

```
{signal 1 {{system.CHILD2.Clock}}}
```

When multiple signals are selected to query, the return value is:

```
{signal 2 {{system.CHILD2.Clock} {system.CHILD2.Reset}}}
```

When a signal array is selected to query, the return value is:

```
{signal 3 {{system.CHILD2.Mux1_Sel[1:0]}
{system.CHILD2.Mux1_Sel[1]} {system.CHILD2.Mux1_Sel[0]}}}
```

srcSignalViewAddSpaInstruToWave

Description

Add the instrumented SPA signals to the *nWave* window.

Syntax

```
srcSignalViewAddSpaInstruToWave -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewAddSpaInstruToWave -win $_nTrace1
```

srcSignalViewAddSrsnInstruToWave

Description

Add the instrumented SRSN signals to the *nWave* window.

Syntax

```
srcSignalViewAddSrsnInstruToWave -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewAddSrsnInstruToWave -win $_nTrace1
```

srcSignalViewCollapse

Description

Collapse the selected signal in the *Signal List* pane.

Syntax

```
srcSignalViewCollapse -row rowNumber
```

Argument

`-row` *rowNumber*

The position of the signal to collapse.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewCollapse -row 5
```

srcSignalViewExpand

Description

Expand the selected signal in the *Signal List* pane.

Syntax

```
srcSignalViewExpand -signalFullName
```

Argument

`-signalFullName`

The signal name with the full path.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
rcSignalViewExpand "testbench.top.child.mvsim_flag"
```

srcSignalViewFilterByType

Description

Show or hide the specified signal type(s) in the *Signal List* pane.

Syntax

```
srcSignalViewFilterByType [-input on|off] [-output on|off]
[-inout on|off] [-wire on|off] [-register on|off] [-parameter
on|off] [-constant on|off] [-event on|off][-buffer on|off]
[-linkage on|off] [-unit on|off][-assertion on|off][-[-isolation
on|off] [-retention on|off] [-levelShift on|off] [-boundaryport
on|off] [-simulationonly on|off] [-other on|off] [-all on|off]
[-srsn on|off] [-upfInserted on|off]
```

Argument

`-all on|off`

Turn the display of all signals *on* or *off*.

`-assertion on|off`

Turn the display of assertion data type signals on or off.

`-boundaryport on|off`

Turn the display of boundary port information on or off.

`-buffer on|off`

Turn the display of buffer data type signals on or off.

`-constant on|off`

Turn the display of constant data type signals on or off.

`-event on|off`

Turn the display of event data type signals on or off.

`-boundaryport on|off`

Turn the display of boundary port information *on* or *off*.

`-inout on|off`

Turn the display of inout ports *on* or *off*.

`-input on|off`

Turn the display of input ports *on* or *off*.

`-isolation on|off`

Turn the display of Isolation signals in power designs *on* or *off*.

`-levelShift on|off`
Turn the display of Level-shifter signals in power designs *on* or *off*.

`-linkage on|off`
Turn the display of linkage data type signals on or off.

`-other on|off`
Turn the display of other signals *on* or *off*.

`-output on|off`
Turn the display of output ports *on* or *off*.

`-register on|off`
Turn the display of register data type signals *on* or *off*.

`-parameter on|off`
Turn the display of parameter data type signals on or off.

`-retention on|off`
Turn the display of Retention signals in power designs *on* or *off*.

`-simulationonly on|off`
Turn the display of simulation information *on* or *off*.

`-srsn on|off`
Turn the display of SRSN signals in power designs *on* or *off*.

`-unit on|off`
Turn the display of unit data type signals on or off.

`-upfInserted on|off`
Turn the display of UPF inserted signals in the HDL design *on* or *off*.

`-wire on|off`
Turn the display of wire type signals *on* or *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewFilterByType -output on
srcSignalViewFilterByType -input off
srcSignalViewFilterByType -all on
```

srcSignalViewSelect

Description

Specify the signal name(s) to be selected in the *Signal List* pane.

Syntax

```
srcSignalViewSelect -signal signalName1 signalName2 SignalName...
```

Argument

`-signal` *signalName*

Specify the signal name(s) to select.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewSelect -signal "system.clock" "system.reset"
```

srcSignalViewSetFilter

Description

Show signals matching the specified string in the search text field of the *Signal List* pane.

Syntax

```
srcSignalViewSetFilter patString
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewSetFilter "clo*"
```

srcSignalViewSort

Description

Sort signals in the *Signal List* pane based on the specified sorting method.

Syntax

```
srcSignalViewSetFilter -name|-declaration
```

Argument

`-name`

Sort signals by the signal name.

`-declaration`

Sort signals by the definition order.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewSort -type
```

srcSourceCodeView

Description

Show source code (as opposed to schematic) in the *nTrace* window.

Syntax

```
srcSourceCodeView
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSourceCodeView
```

Search

srcSetSearchMode

Description

Set the search mode for subsequent **srcSearchPrev** or **srcSearchNext** commands.

Syntax

```
srcSetSearchMode -negedge | -posedge | -anychange | -success |
 -failure | -start_eval
```

Arguments

`-anyChange`

Specify to search by any value change.

`-failure`

Specify to search for the threads with status failure.

`-negedge`

Specify to search by falling edge.

`-posedge`

Specify to search by rising edge.

`-start_eval`

Specify to search for the start time of the threads.

`-success`

Specify to search for the threads with status success.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetSearchMode -posedge
srcSetSearchMode -win $window -negedge
```

srcSetCursorTime

Description

Set the cursor time.

Syntax

```
srcSetCursorTime time
```

Arguments

`time`

Specify the cursor time.

Value Returned

The current cursor time if successful; otherwise, returns 0.

Example

```
srcSetCursorTime 1000
```

srcGetCursorTime

Description

Get the cursor time.

Syntax

```
srcGetCursorTime [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

Cursor time if successful; otherwise, returns 0.

Example

```
srcGetCursorTime
```

srcSearchPrev

Description

Search the previous item found. If found, jump to the previous value change of selected signal that matches the current search mode.

Syntax

```
srcSearchPrev [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if the current cursor time is successful; otherwise, returns 0.

Example

```
srcSearchPrev
```

srcSearchNext

Description

Search the next item found. If found, jump to the next value change of the selected signal that matches the current search mode.

Syntax

```
srcSearchNext [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if the current cursor time is successful; otherwise, returns 0.

Example

```
srcSearchNext
```

srcFndInstportCreate

Description

Open the *Find Signal* form with signal, instance, or instport as the searching type when the **srcFndInstportCreate**, **srcFndInstCreate**, and **srcFndSigCreate** commands are invoked respectively.

Syntax

```
srcFndSigCreate
srcFndInstCreate
srcFndInstportCreate
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcFndSigCreate
srcFndInstCreate
srcFndInstportCreate
```

srcFndSignalSearch

Description

Search for signals.

NOTE: Invoke the Tcl **srcFndSigCreate** command before using this command.

Syntax

```
srcFndSignalSearch [-delim delimString] [-case on|off]
[-fullHierarchy on|off] [-libcell on|off] [-name searchPattern] [-
tofile]
```

Arguments

`-delim` *`delimString`*

Specify the signal delimiter.

`-case on|off`

Turn the case matching *on* or *off*. The default is *on*.

`-fullHierarchy on|off`

Turn whether to search in the full scope *on* or *off*. The default is *off*.

`-libcell on|off`

Turn whether to include library cell *on* or *off*. The default is *off*.

`-name` *`searchPattern`*

Enter search pattern in the text field for signals, instances, or instports.

`-tofile`

Dump the search results to file directly without limiting signal/instance numbers.

NOTE: The Tcl **srcFndSigCreate** command is not required when using `-tofile` option.

Value Returned

Value string if successful; otherwise, returns 0.

Example

```
srcFndSignalSearch -delim "." -case on - fullHierarchy off
-libcell on -name "system.i_cpu.i_ALUB.*"
```

srcFndInstSearch

Description

Search for instances.

NOTE: Invoke the Tcl **srcFndSigCreate** command before using this command.

Syntax

```
srcFndInstSearch [-delim delimString] [-case on|off]
[-fullHierarchy on|off] [-libcell on|off] [-name searchPattern]
[-tofile]
```

Arguments

`-delim` *`delimString`*

Specify the instance delimiter.

`-case on|off`

Turn the case matching *on* or *off*. The default is *on*.

`-fullHierarchy on|off`

Turn whether to search in the full scope *on* or *off*. The default is *off*.

`-libcell on|off`

Turn whether to include library cell *on* or *off*. The default is *off*.

`-name` *`searchPattern`*

Enter search pattern in the text field for signals, instances, or instports.

`-tofile`

Dump the search results to file directly without limiting signal/instance numbers.

NOTE: The Tcl **srcFndSigCreate** command is not required when using `-tofile` option.

Value Returned

Value string if successful; otherwise, returns 0.

Example

```
srcFndInstSearch -delim "." -case on - fullHierarchy off
-libcell on -name "system.*"
```

srcFndInstportSearch

Description

Search for instports.

Syntax

```
srcFndInstportSearch [-delim delimString] [-case on|off]
[-fullHierarchy on|off] [-libcell on|off] [-name searchPattern]
```

Arguments

`-delim` *`delimString`*

Specify the instport delimiter.

`-case on|off`

Turn the case matching *on* or *off*. The default is *on*.

`-fullHierarchy on|off`

Turn whether to search in the full scope *on* or *off*. The default is *off*.

`-libcell on|off`

Turn whether to include library cell *on* or *off*. The default is *off*.

`-name` *`searchPattern`*

Enter search pattern in the text field for signals, instances or instports.

Value Returned

Value string if successful; otherwise, returns 0.

Example

```
srcFndInstportSearch -delim "." -case on -fullHierarchy off
-libcell off -name "system.*"
```

srcFndInstportSel

Description

Search for a specified Instport.

Syntax

`srcFndInstportSel -name` *`objectName`*

Arguments

`-objectName`

Specify the full scope path of the instance port.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcFndInstportSel -name "system.i_cpu.i_ALUB.ALU.S1"
```

srcFndInstportDrag

Description

Drag a specified Instport.

Syntax

```
srcFndInstportDrag -name objectName
```

Arguments

-objectName

Specify the full scope path of the instance port.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcFndInstportDrag -name " system.i_pram.R_W"
```

srcFndInstportDumpAU

Description

Dump data in the *Find Instport* window.

Syntax

```
srcFndInstportDumpAU
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcFndInstportDumpAU
```

srcFndInstportSave

Description

Save resulting data to the specified file.

Syntax

```
srcFndInstportSave -file filename
```

Arguments

`-file` *`filename`*

Output the current result to the file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcFndInstportSave "output"
```

srcMatchParenthesis

Description

When the caret cursor is placed on a parenthesis of the *nTrace* source code pane or the *File Viewer* window (opened by the **Tools -> New File Viewer** command in the *nTrace* window), and the **Ctrl+m** shortcut keys are pressed on the keyboard, the corresponding parenthesis and the source code included within the parentheses are highlighted.

NOTE: The highlight color can be specified in the **Matched Parenthesis** type and **Code in Parentheses** type on the **Source Code** page -> **Color** page in the *Preferences* form.

Syntax

```
srcMatchParenthesis -win ID -line nLine -word nWord [-dump]
```

Arguments

`-win` *`ID`*

Specify the window ID.

`-line nLine`

Identify the line number of the parenthesis in the *nTrace* source code pane or the *File Viewer* window.

`-word nWord`

Identify the word number in the line of the parenthesis in the *nTrace* source code pane or the *File Viewer* window.

`-dump`

Dump the source code between the parentheses into a log file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcMatchParenthesis -win &_nTrace1 -line 10 -word 15
```

srcWatchExpMatchParenthesis

Description

When the caret cursor is placed on a parenthesis in the *Watch Expression* form (opened by the **Tools -> Watch Expressions** command in the *nTrace* window), and the **Ctrl+m** shortcut keys are pressed on the keyboard, the corresponding parenthesis and the source code included within the parentheses are highlighted.

NOTE: The highlight color can be specified in the **Matched Parenthesis** type and **Code in Parentheses** type on the **Source Code** page -> **Color** page in the *Preferences* form.

Syntax

```
srcWatchExpMatchParenthesis -pos lineNo wordNo
```

Arguments

`-pos lineNo wordNo`

Specify the position of the parenthesis where the caret cursor is placed in the *Watch Expression* form.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpMatchParenthesis -pos 0 1
```

Interactive

simEnable

Description

Turn the interactive mode *on* or *off*.

Syntax

```
simEnable on | off
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simEnable on
```

simRun

Description

Start the simulator.

Syntax

```
simRun [-no_msg_inout]
```

Argument

`-no_msg_inout`

Disable the simulator message display and the command input in the message window of *nTrace* when running the simulator.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
simRun
```

```
simRun -no_msg_inout
```

simContinue

Description

Continue the simulator.

Syntax

```
simContinue
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simContinue
```

simReset

Description

Reset the simulator.

Syntax

```
simReset
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simReset
```

simStop

Description

Stop the simulator.

Syntax

```
simStop
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simStop
```

simFinish

Description

Finish the simulation.

Syntax

```
simFinish
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simFinish
```

simKill

Description

Kill the simulation.

Syntax

```
simKill
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simKill
```

simGotoNextTime

Description

Run the simulator to go to the next time unit.

Syntax

```
simGotoNextTime
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simGotoNextTime
```

simGotoNextEvent

Description

Run the simulator to go to the next event statement.

Syntax

```
simGotoNextEvent
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simGotoNextEvent
```

simGotoTime

Description

Run the simulator to the specified time.

Syntax

```
simGotoTime time
```

Arguments

`time`

Specify the time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simGotoTime 1000
```

simGetTime

Description

Get the current simulation time.

Syntax

```
simGetTime
```

Value Returned

Current simulation time.

Example

```
set current_time [simGetTime]
```

simSetKeepBreakPoints

Description

All break points set in the previous simulation run are kept if **simSetKeepBreakPoints** command is *on*.

Syntax

```
simSetKeepBreakPoints [on|off]
```

(The default is *off*.)

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simSetKeepBreakPoints on
```

simSetSimulator

Description

Set the simulator preference.

Syntax

```
simSetSimulator -xl|-nc|-vcs|-mti -exec excutableName -arg args
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simSetSimulator -xl -exec verilog
```

simSetNoAppendOption

Description

Verdi does not append any additional option to invoke the simulator if **simSetNoAppendOption** command is *on*.

Syntax

```
simSetNoAppendOption [on|off]
(the default is off)
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simSetNoAppendOption on
```

simStopAtTimeZero

Description

The simulator stops after compilation if **simStopAtTimeZero** command is *on*.

Syntax

```
simStopAtTimeZero [on | off]
(the default is on)
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simStopAtTimeZero off
```

simGetStatus

Description

Get the simulator status.

Syntax

```
simGetStatus
```

Value Returned

Simulator status string; *terminated*, *compiling*, *stopped*, or *running*.

Example

```
simGetStatus
```

simSendCommand

Description

Send the simulator command.

Syntax

```
simSendCommand [-v] command
```

Arguments

`-v`

Verbose mode. If specified, the simulator output is displayed on the message window.

`command`

Simulator command.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
simSendCommand " #3000 $stop;"
```

simSetWorkDir

Description

Specify the simulation invoked directory.

Syntax

```
simSetiWorkDir -option [work|lib|usd] path
```

or

```
simSetWorkDir -option  [work|lib|usd] "path"
```

Arguments

`work`

It indicates **From Working Directory**.

`lib`

It indicates **From Library Directory**.

`usd`

It stands **From User Specify Dir**.

The use of double quotes for the **path** is optional and it only impacts Verdi simulation invoke directory when the **usd** option is used.

Value Returned

1 if successful; otherwise, returns 0.

tbDebugRunSim

Description

Run the simulator.

Syntax

```
tbDebugRunSim
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tbDebugRunSim
```

tbDebugSimQuit

Description

Terminate the simulator.

Syntax

```
tbDebugSimQuit
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tbDebugSimQuit
```

tbDebugStepNext

Description

Step to the first executed statement of the next simulation time.

Syntax

```
tbDebugStepNext
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tbDebugStepNext
```

Debug

srcAddDumpScope

Description

Specify the scope for the simulation focus that the step event and step time commands stops in.

Syntax

```
srcAddDumpScope strScope|-all
```

Arguments

`-all`

When specified, all scopes are the simulation focus.

`strScope`

Specify the scope as the simulation focus.

Value Returned

None.

Example

```
srcAddDumpScope system.i_cpu
```

srcCallStackDown

Description

Move the selected stack in the *Call Stack Window* one row down.

Syntax

```
srcCallStackDown
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcCallStackDown
```

srcCallStackReturn

Description

Run the simulation to the next line of the selected stack.

Syntax

```
srcCallStackReturn -callstackno [callstack index]
```

Arguments

`-callstackno` *`callstack index`*

Set the call stack index.

Example

```
srcCallStackReturn -callstackno 3
```

srcCallStackUp

Description

Move the selected stack in the *Call Stack Window* one row up.

Syntax

```
srcCallStackUp
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcCallStackUp
```

srcCloseCallStackWin

Description

Close the *Call Stack Window*.

Syntax

```
srcCloseCallStackWin
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcCloseCallStackWin
```

srcGetCurrentValue

Description

Get signal's current value at the current time.

Syntax

```
srcGetCurrentValue [-delim delim] signalName
```

Arguments

`-delim delim`

Delimiter of the name list.

`signalName`

Specify the signal name.

Value Returned

Value string if successful; otherwise, returns 0.

Example

```
srcGetCurrentValue -delim {/} {system/I_cpu/clock}
```

srcGotoCallStack

Description

Jump to a specific call stack.

Syntax

```
srcGotoCallStack -callstackno [callstack index]
```

Arguments

`-callstackno` *`callstack index`*

Set the call stack index.

Example

```
srcGotoCallStack -callstackno 3
```

srcOpenCallStackWin

Description

Open the *Call Stack Window*.

Syntax

```
srcOpenCallStackWin
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcOpenCallStackWin
```

Browse Cell Summary

srcBrowseCellCreate

Description

Create the *Browse Cell Summary* form.

Syntax

```
srcBrowseCellCreate
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcBrowseCellCreate
```

srcBrowseCellSetScope

Description

Switch the current view to the specified scope.

Syntax

```
srcBrowseCellSetScope -scope scopeName -delim delimiter
```

Arguments

`-scope` *`scopeName`*

The full hierarchical scope name the user switched to.

`-delim` *`delimiter`*

The delimiter for the scope name above.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcBrowseCellSetScope -scope
"picoJavaII.monitor.activity_monitor" -delim .
```

srcBrowseCellSelInst

Description

Highlights the selected instance.

Syntax

```
srcBrowseCellSelInst -inst instName
```

Arguments

`-inst` *`instName`*

The name of the selected instance.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcBrowseCellSelInst -inst
"activity_monitor:Always41#Always2:140:152:Reg"
```

srcBrowseCellSave

Description

Save the *Browse Cell Summary* form to a file in ASCII format.

Syntax

```
srcBrowseCellSave -file filename
```

Arguments

`-file` *`filename`*

Specify the output file name for the current results.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcBrowseCellSave -file “/rd40a/chsu/SPSqa44720.txt”
```

Watch Window

wtchCreateWindow

Description

Create the *Watch Signal* form.

Syntax

```
wtchCreateWindow [-wc App] -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

`-wc`

wc stands for *watch caller*, the calling application in *nTrace*.

`App`

Application. It is either *nTrace* or *nWave*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchCreateWindow -wc nWave -win $_nWave3
```

wtchCloseWindow

Description

Close the *Watch Signal* form.

Syntax

```
wtchCloseWindow
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchCloseWindow
```

wtchAddSignal

Description

Add signals to the *Watch Signal* form.

Syntax

```
wtchAddSignal watchNum [-delim delim] signalList
```

Arguments

`-delim delim`

Delimiter of the name list.

`signalList`

List of signal names.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchAddSignal 1 {system.i_cpu.clock} {system.i_mem.memory}
```

wtchDeleteSignal

Description

Delete signals from the *Watch Signal* form.

Syntax

```
wtchDeleteSignal watchNum [-delim delim] -index index signalList |
-all
```

Arguments

`-delim delim`

Delimiter of the name list.

`signalList`

List of signal names.

`-index`

nth entry on the *Watch Signal* form.

`-all`

Delete all the signals on the specified watch page.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchDeleteSignal 1 {system.i_cpu.clock} {system.i_mem.memory}
```

wtchAction

Description

Mimic the double-click action on a signal row in the *Watch* window.

Syntax

```
wtchAction -tab tabNumber -index rowNumber
```

Arguments

`-tab tabNumber`

Tab number, starting from 1.

`-index rowNumber`

Row number, starting from 1.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchAction -tab 3 -index 2
```

wtchExpandBus

Description

Expand or collapse the bus signal in the *Watch* window.

Syntax

```
wtchExpandBus -on|-off -tab tabNumber -index rowNumber -signal
signalPath
```

Arguments

`-on|-off`

On stands for expand bus and off stands for collapse bus.

`-tab` *`tabNumber`*

Tab number, starting from 1.

`-index` *`rowNumber`*

Row number, starting from 1.

`-signal` *`signalPath`*

When both `-index` and `-signal` options are specified and if the signal name at the `-index` position does not match the name `-signal` specified, then the command becomes invalid, that is, nothing happens.

When only `-signal` option is specified but `-index` option is not specified, and if there are multiple instances of the signal in the tab, then the command is applied to all instances of the signal.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchExpandBus -off -tab 3 -index 2
```

wtchSetSearchMode

Description

Set the search mode for subsequent of **wtchSearchPrev** or **wtchSearchNext** commands.

Syntax

```
wtchSetSearchMode -negedge | -posedge | -anyChange
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchSetSearchMode -posedge
```

wtchSetCursorTime

Description

Set the cursor time.

Syntax

```
wtchSetCursorTime time
```

Value Returned

The current cursor time if successful, -1 otherwise.

Example

```
wtchSetCursorTime time 100
```

wtchSearchPrev

Description

Search previous value change of the selected signals on the *Watch* window.

Syntax

```
wtchSearchPrev watchNum [-delim delim] signalList
```

Arguments

`-delim` *`delim`*

Delimiter of the name list.

`signalList`

List of signal names.

Value Returned

1 and the current cursor time if successful; otherwise, returns 0.

Example

```
wtchSearchPrev 1 -delim . {system.i_cpu.clock}
```

wtchSearchNext

Description

Search next value change of the selected signals on the *Watch* window.

Syntax

```
wtchSearchNext watchNum [-delim delim] signalList
```

Arguments

`-delim` *`delim`*

Delimiter of the name list.

`signalList`

List of signal names.

Value Returned

1 and the current cursor time if successful; otherwise, returns 0.

Example

```
wtchSearchNext 1 -delim . {system.i_cpu.clock}
```

wtchSetOptions

Description

Set display options for the *Watch* window.

Syntax

```
wtchSetOptions -showHierName on|off
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchSetOptions -showHierName on
```

wtchSetAliasName

Description

Set alias name of the *Watch* tab name.

Syntax

```
wtchSetAliasName watchNum aliasName
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchSetAliasName 1 {cpu}
```

wtchSetAvailableTab

Description

Set available tab number.

Syntax

```
wtchSetAvailableTab tabNumber
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchSetAvailableTab 5
```

wtchSaveToFile

Description

Save the signals in the current tab or all tabs on the *Watch* form to a specified file.

Syntax

```
wtchSaveToFile [tabNumber] filename
wtchSaveToFile [-AllTag] -file filename
```

Arguments

`-AllTag`

Save all the tab information, including the tab name and data, to a specified file.

`-file filename`

Specify the full file name to save all the tab information.

`tabNumber`

The tab number.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchSaveToFile 2 “/home/watch1”
wtchSaveToFile -AllTag -file /home/watch1
```

wtchRestoreFromFile

Description

Restore signals from a saved file.

Syntax

```
wtchRestoreFromFile [tabNumber] filename
wtchRestoreFromFile [-AllTag] -file filename
```

Arguments

`-AllTag`

Restore all tab information, including the tab name data.

`-file` *`filename`*

Specify the full file name that is saved by the **wtchSaveToFile** command.

`tabNumber`

The tab number.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wtchRestoreFromFile 2 “/home/watch1”
wtchRestoreFromFile -AllTag -file /home/watch1
```

srcWatchExpOpen

Description

Open the *Watch Expression* form.

Syntax

```
srcWatchExpOpen
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpOpen
```

srcWatchExpClose

Description

Close the *Watch Expression* form.

Syntax

```
srcWatchExpClose
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpClose
```

srcWatchExpAdd

Description

Add an expression to the *Watch Expression* form by specifying any operand in the expression. *nTrace* window traces the expression by the operand and adds the expression to the *Watch Expression* form.

Syntax

```
srcWatchExpAdd -win window -line lineNo -word wordNo
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

`-line` *`lineNo`*

The line number of the selected operand, on 0-based.

`-word` *`wordNo`*

The word position in the line of the selected operand, on 0-based.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpAdd -win $_nTrace1 -line 20 -word 12
```

srcWatchExpExpand

Description

Expand the sub-expression to cover the next associated operator. This command is equivalent to the **Expand** button in *Watch Expression* form.

Syntax

```
srcWatchExpExpand
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpExpand
```

srcWatchExpCollapse

Description

Collapse the sub-expression to de-associated current top level operator. This command is equivalent to the **Shrink** button in *Watch Expression* form.

Syntax

```
srcWatchExpCollapse
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpCollapse
```

srcWatchExpOneLevel

Description

Turn *on* or *off* the **Show One Level** option in the *Watch Expression* form.

Syntax

```
srcWatchExpOneLevel 0|1
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpOneLevel 1
```

srcWatchExpDump

Description

Dump current content of the source window in the *Watch Expression* form.

Syntax

```
srcWatchExpDump
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpDump
```

srcWatchExpSetTime

Description

Change the cursor time in the *Watch Expression* form.

Syntax

```
srcWatchExpSetTime time
```

Arguments

`time`

Cursor time value in nanosecond.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcWatchExpSetTime 123000
```

Show Variable

shvrCreateWindow

Description

Create the *Show Variable* form.

Syntax

```
shvrCreateWindow
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
shvrCreateWindow
```

shvrCloseWindow

Description

Close the *Show Variable* form.

Syntax

```
shvrCloseWindow
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
shvrCloseWindow
```

shvrAddVariable

Description

Add variable to the *Show Variable* form.

Syntax

```
shvrAddVariable [-delim delim] variableName
```

Arguments

`-delim delim`

Delimiter of the variable name.

`signalName`

Specify the variable name to be added.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
shvrAddVariable {system.i_cpu.var1}
```

shvrDelVariable

Description

Delete variable(s) from the *Show Variable* form.

Syntax

```
shvrDelVariable [-delim delim] variableName|-all
```

Arguments

`-delim delim`

Delimiter of the name list.

`variableName`

Specify variable name to be deleted.

`-all`

Delete all variables on the specified *Show Variable* form.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
shvrDelVariable {system.i_cpu.var1}
shvrDelVariable -all
```

Access

debFind

Description

Find the specified signal or instance through the design.

Syntax

```
debFind [-delim delim] -signal|-inst [-tofile filename] [-case
on|off] [-fullHierarchy on|off] [-libCell on|off] name
```

Arguments

`-delim delim`

Delimiter of the name list.

`-tofile`

Specify the output file name in which the found results are written to.

`-case on|off`

Turn the case matching *on* or *off*. The default is *on*.

`-fullHierarchy on|off`

Turn whether to search from full hierarchy *on* or *off*. The default is *off*.

`-libcell on|off`

Turn whether to search from the library cell *on* or *off*. The default is *off*.

`name`

Specify the pattern or name.

Value Returned

Handle of integrator if successful; otherwise, returns 0.

Example

```
set sigIter [debFind -inst {system.i_cpu.*}]
```

debGetVersion

Description

Get Verdi release version.

Syntax

```
debGetVersion
```

Value Returned

Version string.

Example

```
set version [debGetVersion]
```

debIterTopScope

Description

Get the top scopes. For a Verilog design, there may be more than one top scope.

Syntax

```
debIterTopScope
```

Value Returned

Handle of iterator if successful; otherwise, returns 0.

Example

```
set iterTopScope [debIterTopScope]
set top [debIterNext $iterTopScope]
```

debIterChildScope

Description

Get the child scopes under the specified scope.

Syntax

```
debIterChildScope [-delim delim] scopeFullName
```

Arguments

`-delim delim`

Delimiter of the specified scope full name.

`scopeFullName`

Specify the scope to be iterated.

Value Returned

Handle of iterator if successful; otherwise, returns 0.

Example

```
set iterChildScope [debIterChildScope {system.i_cpu}]
set childScope [debIterNext $iterChildScope]
```

debIterIOPort

Description

Get the IO ports of the named scope.

Syntax

```
debIterIOPort [-delim delim] scopeFullName
```

Arguments

`-delim delim`

Delimiter of the name list.

`scopeFullName`

Specify the scope to be iterated.

Value Returned

Handle of iterator if successful; otherwise, returns 0.

Example

```
set iterIO [debIterIOPort {system.i_cpu}]
set ioPort [debIterNext $iterIO]
```

debIterRecordField

Description

Get the record field signals of the named record signal.

Syntax

```
debIterRecordField [-delim delim] recordFullName
```

Arguments

`-delim delim`

Delimiter of the name list.

`recordFullName`

Specify the record to be iterated.

Value Returned

Handle of iterator if successful; otherwise, returns 0.

Example

```
set iterRecord [debIterRecordField {system.i_cpu.r}]
set recordSig [debIterNext $iterRecord]
($recordSig = r.field1.field11)
```

debIterNext

Description

Get next item from the iterator (for example, next signal, next scope).

Syntax

```
debIterNext iterateHandler
```

Arguments

`iterateHandler`

Handler.

Value Returned

Resultant string. NULL if no more.

Example

```
set iterIO [debIterIOPort {system.i_cpu}]
set ioPort [debIterNext $iterIO]
```

debIterCancel

Description

Cancel the iterator. After this command is issued, the named iterator can no longer be used for iterating.

Syntax

```
debIterCancel iterateHandler
```

Arguments

`iterateHandler`

Handler.

Value Returned

None.

Example

```
debIterCancel
```

debSaveAllFileNames

Description

Save all the file names of the design to output file. If `-file` option is specified, the modules which are defined in the specified file are appended to the file.

Syntax

```
debSaveAllFileNames [-file filename|-all] [-sort] outFileName
```

Arguments

`-file` *`filename`*

If specified, the module list of the specified file name is outputted also.

`-all`

If specified, all the corresponding module list of each file is the output. `-file` option is ignored.

`-sort`

If specified, the output is sorted by alphabetic order. The default is to use the original order.

`outFullName`

Specify the output file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debSaveAllFileName -file "cpu.v" "design_cpu.list"
```

debSaveAllModuleNames

Description

Save all the module names of the design to the output file. If `-module` option is specified, the instantiated list of the specified module is appended to the file.

Syntax

```
debSaveAllModuleNames [-module moduleName |-all] [-sort]
outFileName
```

Arguments

`-module`

If specified, the instantiated list of the specified module name is also the output.

`-all`

If specified, all the instantiated list of each module is the output. `-file` option is ignored.

`-sort`

If specified, the output is sorted by alphabetic order. The default is to use the original order.

`outFileName`

Specify the output file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debSaveAllModuleNames -module "CPU"  -sort "design_cpu.list"
```

debSaveAllTaskNames

Description

Save all the task names of the design to the output file.

Syntax

```
debSaveAllTaskNames [-task taskName | -all] [-sort] outFileName
```

Arguments

`-all`

When specified, all the instantiated list of each task is outputted. `-file` option is ignored.

`outFileName`

Specify the output file name.

`-sort`

When specified, the output is sorted by alphabetic order. The default is to use the original order.

`-task`

When specified, the instantiated list of the specified task name is outputted also.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debSaveAllTaskNames -task "FINISH_IT"  -sort "design_cpu.list"
```

debSaveAllFunctionNames

Description

Save all the function names of the design to the output file.

Syntax

```
debSaveAllFunctionNames [-function function name | -all] [-sort]
outfilename
```

Arguments

`-all`

When specified, all the instantiated list of each function is outputted. `-file` option is ignored.

`-function`

When specified, the instantiated list of the specified function name is outputted also.

`outFileName`

Specify the output file name.

`-sort`

When specified, the output is sorted by alphabetic order. The default is to use the original order.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debSaveAllFunctionNames -function "SIZE_EXPAND"  -sort
"design_cpu.list"
```

debIsScopeSwitchable

Description

Query whether the scope is switchable.

Syntax

```
debIsScopeSwitchable -delim delim scopeName
```

Arguments

`-delim` *`delim`*

Delimiter of the scope name.

`scopeName`

Specify scope name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debIsScopeSwitchable -delim . system.task1
```

debSetAnnotPartialBus

Description

If this command is *on*, all annotation values in Verdi for partial bus dumping are changed at once globally. For example, if a partial bus original value is 1...0.....1001.....01, then its value after enabling this option is [0]1[4]0[15]1001[27]01. The number in the square bracket interprets the following value position in this whole value string.

Syntax

```
debSetAnnotPartialBus -showIndex on|off
```

Arguments

`-showIndex`

Index for partial-valued bus annotation.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debSetAnnotPartialBus -showIndex on
```

srcIsRecordType

Description

Query whether the signal is record signal.

Syntax

```
srcIsRecordType -delim delim signalName
```

Arguments

`-delim delim`

Delimiter of the scope name.

`signalName`

Specify signal name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcIsRecordType -delim . system.record1
```

Configuration

debSetHBOptions

Description

Set options for the hierarchy tree on the *nTrace* window.

Syntax

```
debSetHBOptions {-moduleName on|off |-font font|-level level}
```

Arguments

`-moduleName on|off`

Turn the display of the module name *on* or *off*.

`-font`

Specify the font for the display;
"Courier 10","Courier 12","Courier 14",
"Courier 18","Courier 24", "Helvetica 10",
"Helvetica 12", "Helvetica 14",
"Helvetica 18","Helvetica 24",
"Times 10","Times 12","Times 14",
"Times 18","Times 24"

`-level`

Specify default level when you extend the hierarchy. The default is *1*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debSetHBOptions -modulename on -font "Helvetica 10" -level 2
```

srcAddFolder

Description

Add the user-defined folder to the current file and collapse the source code lines.

Syntax

```
srcAddFolder [-name Name] [-file fileName] -line beginLine endLine
[-color Color] [-comment Comment]
```

Arguments

`-color Color`

Specify the background color for the folder. The default is the source code background color.

`-comment Comment`

Specify a comment for the folder.

`-file fileName`

Specify the file to add the folder. The default value is the current top file.

`-line beginLine endLine`

Specify the begin and end lines to indicate the range to be included in the folder.

`-name Name`

Specify the folder name. The default value is the index of this folder.

Value Returned

User-defined folder index if successful; otherwise, returns 0.

Example

```
srcAddFolder -file ./test.v  -line 2 4
```

srcConvertSigNameCase

Description

Convert the case of a signal name to match the signals stored in the KDB. This command is useful for VHDL or mixed-language names in which the original case does not match the case stored in the KDB. If the input signal is from a Verilog or mixed-language design, the Verilog portions of the name must match the original case in the design.

Syntax

```
srcConvertSigNameCase -signal signalName
```

Arguments

`-signal signalName`

The signal name you want to convert.

Value Returned

Return the name stored in KDB.

Example

```
srcConvertSigNameCase -signal SYSTEM.I_CPU.i_ALUB.i_alu.ZERO
return
system.i_cpu.i_ALUB.i_alu.zero
```

srcDeleteFolder

Description

Delete the user-defined folder from the current file and expand the source code lines.

Syntax

```
srcDeleteFolder -index indexNumber [-all]
```

Arguments

`-all`

Delete all the user-defined folders.

`-index indexNumber`

Specify the user-defined index of the folder to be deleted.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcDeleteFolder -index 1
```

srcEditFolder

Description

Edit the user-defined folder.

Syntax

```
srcEditFolder -index indexNumber [-name Name] [-line beginLine
endLine] [-color Color] [-comment Comment]
```

Arguments

`-color Color`

Specify the new background color for the folder.

`-comment Comment`

Specify a comment for the folder.

`-index indexNumber`

Specify the user-defined index of the folder to be edited.

`-line beginLine endLine`

Specify the new begin and end lines of the folder.

`-name Name`

Specify the folder name. The default value is the index of this folder.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcEditFolder -index 1 -comment "a new comment"
```

srcGotoFolder

Description

Go to the corresponding source code of the specified folder.

Syntax

```
srcGotoFolder -index indexNumber
```

Arguments

`-index indexNumber`

Specify the folder to go to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcGotoFolder -index 1
```

srcHideFolderManager

Description

Hide the *Manage User-defined Folders* form.

Syntax

```
srcHideFolderManager
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcHideFolderManager
```

srcInstanceTreeExcludeNodes

Description

Remove node(s) matching the specified criteria from the *Instance Tree*.

Syntax

```
srcInstanceTreeExcludeNodes [-win window] string [-type type] [-
caseInsen on|off] [-incremental on|off]
```

Arguments

`-win window`

Specify the window ID of the invoked source code window.

`-type` *`type`*

Specify the type to be removed.

`-caseInsen on|off`

Specify a string without case-sensitivity check.

`-incremental on|off`

Adopt incremental method.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcInstanceTreeExcludeNodes -win $_nTrace1 * -type interface
class -caseInsen on -incremental on
srcInstanceTreeExcludeNodes -win $_nTrace1 system -type all -
caseInsen off -incremental off
```

srcInstanceTreeRestore

Description

Restore the *Instance Tree* to the original state.

Syntax

```
srcInstanceTreeRestore [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoked source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcInstanceTreeRestore -win $_nTrace1
```

srcJumpFolder

Description

Jump (automatically scroll) to the non-visible begin or end location of a folder.

Syntax

```
srcJumpFolder -win windowID -line lineNum -level folderLevel
-jumpBegin|-jumpEnd
```

Arguments

`-level folderLevel`

Specify which level of folders to expand or collect. The `-line` and `-level` options always need to be specified together.

`-line lineNum`

Line number. The `-line` and `-level` options always need to be specified together.

`-jumpBegin|-jumpEnd`

Jump to the begin or end location of the folders.

`-win windowID`

Specify the window ID of the *nTrace* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcJumpFolder -win $_nTrace1 -jumpEnd -line 112 -level 1
srcJumpFolder -win $_nTrace1 -jumpBegin -line 92 -level 2
```

srcSetBGHighLightColor

Description

Set the background highlight color for the specified signal(s) or instance(s) or reset all user-defined highlight colors.

Syntax

```
srcSetBGHighLightColor [-signal|-inst] "name1" "name2"..."nameN"
-color colorName
srcSetBGHighLightColor -reset
```

Arguments

`-signal|-inst "name1" "name2"..."nameN"`

Specify the signal/instance name(s) to highlight.

`-color colorName`

Specify the background color to highlight.

`-reset`

Reset all user-defined highlight colors.

Value Returned

1 if successful; otherwise, returns 0.

Examples

`srcSetBGHighLightColor -signal "system.clock" -color ID_BLUE2`
signal *system.clock* is highlighted as color ID_BLUE2.

`srcSetBGHighLightColor -inst "system.i_cpu""system.i_pcu" -color ID_GREEN5`
instance *system.i_cpu* and *system.i_pcu* is highlighted as color ID_GREEN5.

`srcSetBGHighLightColor -reset`
nTrace clears all user-defined background color.

NOTE: 1. You need to specify the signal name or the instance name with the full path and you must use the default delimiter ".".

2. There are 64 color IDs available as follows:
ID_BLACK, ID_GRAY1, ID_GRAY2, ID_GRAY3,
ID_GRAY4, ID_GRAY5, ID_GRAY6, ID_WHITE,

ID_RED1, ID_RED2, ID_RED3, ID_RED4,
ID_RED5, ID_RED6, ID_RED7, ID_RED8,

ID_ORANGE1, ID_ORANGE2, ID_ORANGE3, ID_ORANGE4,
ID_ORANGE5, ID_ORANGE6, ID_ORANGE7, ID_ORANGE8,

ID_YELLOW1, ID_YELLOW2, ID_YELLOW3, ID_YELLOW4,
ID_YELLOW5, ID_YELLOW6, ID_YELLOW7, ID_YELLOW8,

ID_GREEN1, ID_GREEN2, ID_GREEN3, ID_GREEN4,
ID_GREEN5, ID_GREEN6, ID_GREEN7, ID_GREEN8,

ID_CYAN1, ID_CYAN2, ID_CYAN3, ID_CYAN4,
ID_CYAN5, ID_CYAN6, ID_CYAN7, ID_CYAN8,

ID_BLUE1, ID_BLUE2, ID_BLUE3, ID_BLUE4,
ID_BLUE5, ID_BLUE6, ID_BLUE7, ID_BLUE8,

ID_PURPLE1, ID_PURPLE2, ID_PURPLE3, ID_PURPLE4,
ID_PURPLE5, ID_PURPLE6, ID_PURPLE7, ID_PURPLE8

srcSetDisplayAttr

Description

Set the display attributes for the type.

Syntax

```
srcSetDisplayAttr {typeName} -color {colorID}
srcSetDisplayAttr -font "font"
```

Arguments

`typeName`

3D Light | 3D Shadow | Alias | All | Annotation | Annotation Shadow | Architecture | Assertion Identifier | Background | Code In Parentheses | Comment | ComputedSignal | Constant | Entity | Folded Line Number | Function Annotation | Generic | HVL Drive | HVL Drive and Load | HVL Identifier | HVL Load | InOut Signal | Input Signal | Instance | Key Word | Line Number | Macro Value | Matching Parenthesis | Module | Operator | Others | Output Signal | Package | Parameter | Parameter Annotation | Port | Signal | Special Comment | Traced Mark

`-color`

Specify the color name.

`-font`

Specify the font for the source display as follows:
"Courier 10", "Courier 12", "Courier 14", "Courier 18", "Courier 24", "Helvetica 10", "Helvetica 12", "Helvetica 14", "Helvetica 18", "Helvetica 24", "Times 10", "Times 12", "Times 14", "Times 18", "Times 24"

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetDisplayAttr {Input Signal} -color {ID_RED5}
srcSetDisplayAttr {Parameter} -color {ID_ORANGE5}
srcSetDisplayAttr -font "Helvetica 12"
```

srcSetLineBackgroundColor

Description

Set the background color for source lines in the current file or reset all user-defined highlight colors.

Syntax

```
srcSetLineBackgroundColor -line beginLine endLine
-color colorID
srcSetLineBackgroundColor -reset
```

Arguments

`-color` *`colorID`*

Specify the background color.

`-line` *`beginLine endLine`*

Specify the begin and end lines to indicate the range to change the background color.

`-reset`

Reset all user-defined highlight colors.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetLineBackgroundColor -line 5 10 -color ID_BLUE2
```

srcSetOptions

Description

Set all the viewing options in the *nTrace* window.

Syntax

```
srcSetOptions [-win window] [-annotate on|off] [-grayFalseLogic
on|off] [-ExpandImplicitPort on|off] [-ExpandMacro on|off]
[-funcAnnotate on|off] [-leadingZeros on|off] [-lineNum on|off]
[-paraAnnotate on|off] [-traceHierarchy on|off] [-syncSelect
on|off] [-traceComplexPassThrough on|off] [-ExpandEncrypted
on|off] [-stopOnComplexInstPort on|off] [-tracePowerDesign on|off]
```

Arguments

`-annotate on|off`

Turn the display of the annotation *on* or *off*.

`-ExpandEncrypted on|off`

Turn the display of the encrypted code *on* or *off*. The default is *on*.

`-ExpandImplicitPort on|off`

Turn the display of the implicit port expansion *on* or *off*.

`-ExpandMacro on|off`

Turn the display of the macro expansion *on* or *off*.

`-funcAnnotate on|off`

Turn the display of the function annotation *on* or *off*.

`-grayFalseLogic on|off`

Enable or disable the graying of false logic in the source code pane.

`-leadingZeros on|off`

Turn the display of the leading zeros *on* or *off*.

`-lineNum on|off`

Turn the display of the line number *on* or *off*. If this is specified, `-win` is ignored.

`-paraAnnotate on|off`

Turn the display of the parameter annotation *on* or *off*.

`-stopOnComplexInstPort on|off`

When this option is turned *on*, complex expressions (that is, all expressions except slice, index, and concatenate) are regarded as a driver/load and

tracing stops on the instance port. For slice, index, and concatenate expressions, tracing passes through the instance port. The default is *off*.

`-traceComplexPassThrough on|off`

When this option is turned *on*, each variable is traced in complex pass-throughs automatically. When this option is turned *off*, tracing stops at complex pass-throughs. The default is *on*.

`-tracePowerDesign on|off`

When his option is turned *on*, all the power results of the selected signal in the traced scopes in *nTrace* are traced and the results are displayed in the *Message* frame. Double-clicking the power results in the *Message* frame jumps to the corresponding source code line. The default is *on*.

`-traceHierarchy on|off`

Turn whether to trace across hierarchy *on* or *off*. The default is *on*.

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetOptions -win $window -annotate on
srcSetOptions -win $window -traceHierarchy off
srcSetOptions -win $_nTrace1 -funcAnnotate on
srcSetOptions -win $_nTrace1 -ExpandMacro on
srcSetOptions -win $_nTrace1 -ExpandImplicitPort off
srcSetOptions -traceComplexPassThrough on
srcSetOptions -win $_nTrace1 -ExpandEncrpted off
srcSetOptions -win $_nTrace1 -stopOnComplexInstPort off
srcSetOptions -win $_nTrace1 -tracePowerDesign off
```

srcSetPreference

Description

Set all the viewing preference for all source windows.

Syntax

`srcSetPreference [-font` *`fontName`*`] [-3Dannot on|off] [-tabStop` *`tabStop`*`] [-treeFont` *`treeFontName`*`] [-moduleName on|off] [-libOption` *`libOption`*`] [-libDir` *`libDir`*`] [-libPath` *`libPath`*`] [cmdToolbar on|off]`

```
[-simToolbar on|off] [-baToolbar on|off] [-lineNumber on|off]
[-leadingZero on|off] [-traceCrossHier on|off] [-syncSignalSel
on|off] [-vhdlSimType ModelSim|NC] [-veriSimType XL|VCS]
[-hierSeparator delim] [-searchMatchCase on|off]
[-dbclickActiveTrace on|off] [-paraRadix Bin|Oct|Dec|Hex|Origin]
[-cmdEntryForm on|off] [-activeAnnot on|off] [-paraAnnot on|off]
[-scopeMatchCase on|off] [-showSpecialComment on|off]
[-traceThroughSubscope on|off] [-traceToHVL on|off]
[-showLibCells on|off][-showWndCtntWhileResizing on|off]
[-dndActionType ActionTypeString]
[-setDefaultEditor TurboEditor|Vi|Emacs|Texteditor|OtherEditor]
[-nLintOption [option string] -nLintGUIMode] [-FullHierTip on|off]
[-delimiterForFindSignal_Inst_Instport delimiter]
[-PrefetchViewAnnot on|off] [-ExpandMacro on|off] [-signalList
on|off] [-autoFolding on|off] [-cmtFolding on|off] [-stmtFolding
on|off] [-portListFolding on|off] [-treeBgColor colorID]
[-skipTopLevelName on|off] [-copyScopeNameDelimiter strDeli]
[-exitSave on|off] [-autoSaveInterval interval] [-saveDir dir]
[-savefileName filename] [-subProgFolding on|off]
[-showInstanceNameInTabLabel on|off] [-showMacro on|off]
[-showPDNameInTabLabel on|off] [-searchIntoSubtree on|off]
[-traceLimitedScope {fullHierarchyName}] [-encCmtFolding on|off]
[-passThrough {None}|{All}|{By Port Direction}]
[-instTreeNavigation on|off] [-declTreeNavigation on|off]
[-stackPaneNavigation on|off] [-classViewNavigation on|off]
[-objecViewNavigation on|off] [-localPaneNavigation on|off]
[-memPaneNavigation on|off] [-watchPaneNavigation on|off]
[-constraintDebugViewNavigation on|off] [-vhdlcase
original|lowercase] [-filterLPARootTree on|off]
[-traceShowTopLevelPort on|off]
[-traceActiveSnapTransition on|off]
[-traceSortScopeByDepthFirst on|off]
```

Arguments

`-3DAnnot on|off`

Turn the 3D annotation *on* or *off*.

`-activeAnnot on|off`

Turn the active annotation default setting *on* or *off*.

`-autoFolding on|off`

Turn the automatic statement and comment folding *on* or *off*.

`-autoSaveInterval interval`

Specify the time interval (in minutes) for automatically saving the session.

`-baToolbar on|off`

Turn the display of the annotation toolbar *on* or *off*.

`-classViewNavigation on|off`

Turn the display of navigation fields in the *Class* tab *on* or *off*. The default is *on*.

`-constraintDebugViewNavigation on|off`

Turn the display of navigation fields in the *Constraint Debug* tab *on* or *off*. The default is *on*.

`-cmdEntryForm on|off`

Turn the display of the command entry form *on* or *off*.

`-cmdToolbar on|off`

Turn the display of the command toolbar *on* or *off*.

`-cmtFolding on|off`

Turn the comment folding in the *nTrace* window *on* or *off*. When this option is turned *on*, the **Comment Folder** command and its sub-commands under the **View** menu in *nTrace* appear and the comment folders in *nTrace* source code pane are expanded.

`-copyScopeNameDelimiter` *`strDeli`*

Specify the delimiter for the signal full hierarchy name when copying a hierarchical path. The default is ".".

`-dbclickActiveTrace on|off`

Turn double-click to trace *on* or *off*.

`-declTreeNavigation on|off`

Turn the display of navigation fields in the *Declaration Tree* tab *on* or *off*. The default is *off*.

`-delimiterForFindSignal_Inst_Instport` *`delimiter`*

Specify the delimiter to use when searching for signals, instances, or instance ports in the *nTrace* and *nSchema* windows.

`-dndActionType` *`ActionTypeString`*

Specify the drag and drop action. Available drag and drop action types: **Trace Connectivity**, **Trace Driver**, **Trace Load**, and **Jump Declaration**. The default is **Trace Connectivity**.

`-encCmtFolding on|off`

When this option is turned *on*, the encrypted code is folded with the collapsed code symbol displayed in the *nTrace* line number pane. When this option is turned *off*, encrypted code is fully displayed. The default is *on*.

`-exitSave on|off`

Save the session automatically when exiting the session. The default is *on*.

`-ExpandMacro on|off`

When this option is turned *on*, macro statements in *nTrace* source code pane is expanded. When this option is turned *off*, macro statements are collapsed.

`-filterLPARootTree on|off`

When this option is turned *on*, hide the Low Power Assertion (LPA) root tree. When this option is turned *off*, show the Low Power Assertion (LPA) root tree. The default is *off*.

`-font`

Specify the font of the *nTrace* display text.

`-FullHierTip on|off`

When this option is turned *on*, the tip in non-flattened window shows the entire hierarchy.

`-hierSeparator` *`delim`*

Specify the hierarchical delimiter for pure spice designs.

`-instTreeNavigation on|off`

Turn the display of navigation fields in the *Instance Tree* tab *on* or *off*. The default is *off*.

`-leadingZero on|off`

Turn the display of the leading zero for annotation value *on* or *off*.

`-libDir` *`libDir`*

Import the library directory.

`-libOption` *`libOption`*

Import the library option(s).

`-libPath` *`libPath`*

Import the library file.

`-lineNumber on|off`

Turn the display of the line number *on* or *off*.

`-localPaneNavigation on|off`

Turn the display of navigation fields in the *Local* tab *on* or *off*. The default is *on*.

`-memPaneNavigation on|off`

Turn the display of navigation fields in the *Member* tab *on* or *off*. The default is *on*.

`-moduleName on|off`

Turn the display of the tree window module name *on* or *off*.

`-objecViewNavigation on|off`

Turn the display of navigation fields in the *Object* tab *on* or *off*. The default is *on*.

`-passThrough {All}|{By Port Direction}|{None}`

Set to show all passthroughs, passthroughs by port directions, or no passthroughs. The default is *By Port Direction*.

`-paraAnnot on|off`

Turn the parameter annotation default setting *on* or *off*.

`-paraRadix Bin|Oct|Dec|Hex|Origin`

Specify the parameter annotation radix format.

`-portListFolding on|off`

Turn the port list folding in the *nTrace* window *on* or *off*. When this option is turned *on*, the **Port List Folder** command and its sub-commands under the **View** menu in *nTrace* appear and port list folders in *nTrace* source code pane are expanded.

`-PrefetchViewAnnot on|off`

When this option is turned *on*, only values for signals displayed in the visible area of *nTrace*'s source code are pre-fetched. When this option is turned *off*, values for signals in the current scope are pre-fetched and loaded.

`-saveDir` *`dir`*

Specify the directory for the session to be saved.

`-savefileName` *`filename`*

Specify the file name for the session to be saved.

`-scopeMatchCase`

Turn the match case option for **Find Scope** field *on* or *off*.

`-searchIntoSubtree on|off`

When this option is turned *on*, searching in the *Design Browser* frame includes sub-trees and tree nodes can be found. The default is *off*.

`-searchMatchCase on|off`

Turn the case matching *on* or *off*.

`-setDefaultEditor TurboEditor|Vi|Emacs|Texteditor|OtherEditor`

Specify the default editor and save this information into the *rc* resource file.

`-showDuplicateRst on|off`

When this option is turned *on,* duplicate results are displayed.

`-showInstanceNameInTabLabel on|off`

When this option is turned *on*, the source code tabs are named with the active instance (module) name. The default is *off*.

`-showLibCells on|off`

When this option is turned *on*, the library cells are generated/displayed. When this option is turned *off*, the library cells are not generated/displayed.

`-showMacro on|off`

When this option is turned *on*, macro information is shown when specifying the `-paraAnnotate` option to show parameter annotation. When this option is turned *off*, macro information is not shown when specifying the `-paraAnnotate` option to show parameter annotation. The default is *on*.

`-showPDNameInTabLabel on|off`

When this option is turned *on*, the power source code tabs are named with the active power domain name. The default is *off*.

`-showSpecialComment on|off`

Shows the special comment. When it is turned *on*, the *nTrace* window displays special comments in ID_GREEN3.

`-showWndCtntWhileResizing on|off`

When this option is turned *on*, the *nWave* window is redrawn when the window is resized. When this option is turned *off*, the *nWave* window is only redrawn when the mouse is released.

`-signalList on|off`

When this option is turned *on*, the *Signal List* pane is shown automatically in *nTrace*. When this option is turned *off*, the *Signal List* pane is not shown automatically in *nTrace*. The default is *off*.

`-simToolbar on|off`

Turn the display of the simulation toolbar *on* or *off*.

`-skipTopLevelName on|off`

When this option is turned *on*, skip the top scope of the signal full hierarchy name during a hierarchical path copy. When this option is turned *off*, keep the top scope of the signal full hierarchy name during a hierarchical path copy. The default is *off*.

`-stackPaneNavigation on|off`

Turn the display of the navigation fields in the *Stack* tab *on* or *off*. The default is *on*.

`-stmtFolding on|off`

Turn the statement folding in the *nTrace* window *on* or *off*. When this option is *on*, the **Statement Folder** command and its sub-commands under the **View** menu in *nTrace* appear and statement folders in *nTrace* source code pane are expanded.

`-subProgFolding on|off`

Turn the sub-program folding in the *nTrace* window *on* or *off*. When this option is turned *on*, the **Source Code Folder** command and its sub-commands under the **View** menu in *nTrace* appear and sub-program folders in *nTrace* source code pane are expanded.

`-syncSignalSel on|off`

Turn the signal selection synchronization *on* or *off*.

`-tabStop` *`tabStop`*

Specify the number for the tab stop.

`-traceActiveDelect on|off`

When this option is turned *on,* the active driver detection is performed.

`-traceActiveShowInactive on|off`

When this option is turned *on,* both the active and the inactive drivers are displayed.

`-traceActiveSnapTransition on|off`

When this option is turned *on,* the **Snap to Last Transition** preference option is turned *on*.

`-traceAddToWave on|off`

When this option is turned *on*, the trace signals are added to the waveform.

`-traceCrossHier on|off`

Turn the crossing hierarchical trace action *on* or *off*.

`-traceHighlightActive on|off`

When this option is turned *on,* the active driver signals are highlighted.

`-traceHighlightColor gd_color_name`

Sets the color to highlight the active driver signals.

`-traceLatestOnTop on|off`

When this option is turned *on*, the latest trace is displayed in the front.

`-traceLimitedScope {`*`fullHierarchyName`*`}`

Specify the limited scope with full hierarchical name to trace in the desired scope only.

Specify another scope of interest to change the limited scope.

Leave the {*fullHierarchyName*} blank to remove the limited scope, such as:

```
srcSetPreference -traceLimitedScope
```

`-traceMaxClockCycle int_number`
Sets the Maximum Clock Cycles to Trace Value Change.

`-traceRaisingView on|off`
When this option is turned *on,* the raising trace view is traced.

`-traceShowIntermediate on|off`
When this option is turned *on,* the intermediate drivers are displayed.

`-traceShowPassThrough on|off`
When this option is turned *on,* passthrough results are displayed.

`-traceShowTopLevelPort on|off`
When this option is turned *on,* the top level port results are displayed.

`-traceSortByDepthFirst on|off`

When this option is turned *on,* the **Sort Scope by Depth-first** preference option is turned on.

`-traceToHVL on|off`

When this option is turned *on*, the commands **Trace Driver**, **Trace Load**, **Trace Connectivity**, and **Active Trace** in the *nTrace* source code pane locate all drivers, loads, and active drivers for the selected signal that exists in both the HDL and the HVL code. When this option is turned *off*, the commands **Trace Driver**, **Trace Load**, **Trace Connectivity**, and **Active Trace** in the *nTrace* source code pane only locate drivers, loads, and active drivers in the HDL code for the selected signal (tracing does not cross the boundary into the HVL). The default is *on*.

`-traceThroughSubscope on|off`

Tracing scope is only limited to the located scope of the selected signal and all its sub-scopes. The option applies to *nTrace*'s trace driver, trace load, and trace connectivity.

`-treeBgColor` *`colorID`*

Specify the tree window background color. The default is *ID_gray6*.

`-treeFont` *`treeFontName`*

Specify the tree window display font.

`-veriSimType XL|VCS`

Specify Verilog naming convention.

`-vhdlcase original|lowercase`

Specify the letter case of the VHDL source.

`-vhdlSimType ModelSim|NC`

Specify VHDL naming convention.

`-watchPaneNavigation on|off`

Turn the display of navigation fields in the *Watch* tab *on* or *off*. The default is *on*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetPreference -tabStop 4
srcSetPreference -nLintOption "-nologo" -nLintGUIMode on/off
srcSetPreference -delimiterForFindSignal_Inst_Instport "."
srcSetPreference -PrefetchViewAnnot on
srcSetPreference -ExpandMacro off
srcSetPreference -cmtFolding on -portListFolding on
srcSetPreference -copyScopeNameDelimiter "/"
srcSetPreference -skipTopLevelName on
srcSetPreference -traceLimitedScope {top.mem}
srcSetPreference -filterLPARootTree on
srcSetPreference -traceShowTopLevelPort on
srcSetPreference -traceSortScopeByDepthFirst on
srcSetPreference -traceActiveSnapTransition on
```

srcSetSignalPaneAttr

Description

Configure the preference settings for the signal list.

Syntax

```
srcSetSignalPaneAttr [-case on|off] [-sigNumberInPage number]
[-fgColor colorID] [-bgColor colorID] [-hiliteColor colorID]
[-font font]
```

Arguments

`-bgColor colorID`

Specify the background color of the signal list.

`-case on|off`

When this option is turned *on*, the search is case-sensitive. When this option is turned *off*, case is not considered during a search. The default is *on*.

`-fgColor` *`colorID`*

Specify the foreground color of the signal list.

`-font` *`font`*

Specify the font of the text in the signal list.

`-hiliteColor` *`colorID`*

Specify the highlight color of the signal list.

`-sigNumberInPage`

Specify the number of signal to display in one page.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetSignalPaneAttr -fgColor ID_BLACK -bgColor ID_GRAY6
-hiliteColor ID_BLUE5 -font "Courier 12"
srcSetSignalPaneAttr -sigNumberInPage 530
srcSetSignalPaneAttr -case on
```

srcSetUserDispAttr

Description

Change a list of signals to be colorized in the specified color in *nTrace*. When a signal color is changed, all the colors of the signal on different lines and different *nTrace* windows are also changed.

Executing this command updates all source windows immediately.

Syntax

```
srcSetUserDispAttr -signal signalName -color default
srcSetUserDispAttr -signal signalName -color colorName
[-line lineNumber] [-pos objectNumber]
srcSetUserDispAttr -reset
```

Arguments

`-signal` *`signalName`*

The full path of the signal. You must use default delimiter ".".

`-color` *`colorName`*

64 color IDs available: ID_BLACK, ID_GRAY1, ID_GRAY2, ID_GRAY3, ID_GRAY4, ID_GRAY5, ID_GRAY6, ID_WHITE, ID_RED1, ID_RED2, ID_RED3, ID_RED4, ID_RED5, ID_RED6, ID_RED7, ID_RED8, ID_ORANGE1, ID_ORANGE2, ID_ORANGE3, ID_ORANGE4, ID_ORANGE5, ID_ORANGE6, ID_ORANGE7, ID_ORANGE8, ID_YELLOW1, ID_YELLOW2, ID_YELLOW3, ID_YELLOW4, ID_YELLOW5, ID_YELLOW6, ID_YELLOW7, ID_YELLOW8, ID_GREEN1, ID_GREEN2, ID_GREEN3, ID_GREEN4, ID_GREEN5, ID_GREEN6, ID_GREEN7, ID_GREEN8, ID_CYAN1, ID_CYAN2, ID_CYAN3, ID_CYAN4, ID_CYAN5, ID_CYAN6, ID_CYAN7, ID_CYAN8, ID_BLUE1, ID_BLUE2, ID_BLUE3, ID_BLUE4, ID_BLUE5, ID_BLUE6, ID_BLUE7, ID_BLUE8, ID_PURPLE1, ID_PURPLE2, ID_PURPLE3, ID_PURPLE4, ID_PURPLE5, ID_PURPLE6, ID_PURPLE7, and ID_PURPLE8.

`-color default`

Reset the specified signal's color to the default color.

`-reset`

Delete all the user-defined signal colors and all signals are reset to their default colors.

`-line` *`lineNumber`*

Change signal color by line number.

`-pos` *`objectNumber`*

Change signal color by object position.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetUserDispAttr -signal "system.i_pram.a" -color ID_RED5
srcSetUserDispAttr -signal "stop" -color ID_BLUE4 -line "44" -pos
{2,3}
srcSetUserDispAttr -signal "stop" -color ID_YELLOW -line "44" -
pos {1}
srcSetUserDispAttr  -reset
srcSetUserDispAttr -signal "stop" -color ID_RED5 -line "44"
```

srcSetVHDLSimType

Description

Set VHDL naming style (ModelTech or NC).

Syntax

```
srcSetVHDLSimType ModelTech | NC
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetVHDLSimType ModelTech
```

srcShowFolderManager

Description

Show the *Manage User-defined Folders* form.

Syntax

```
srcShowFolderManager
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcShowFolderManager
```

srcToggleFolder

Description

Toggle to expand or collapse the folders.

Syntax

```
srcToggleFolder -win window [-line lineNum -level folderLevel |
-scope|-design] -expand|-collect [-macro] [-implicitPort] [-cmt]
[-stmt] [-portList] [-subProgram]
```

Arguments

`-cmt`

When specified, all comment folders are expanded or collapsed.

`-design`

Toggle all folders in the current design.

`-expand|-collect`

Toggle to expand or collapse the folders.

`-implicitPort`

Apply to implicit port folder when this option is specified.

`-level` *`folderLevel`*

Specify which level of folders to expand or collapse. The `-line` and `-level` options always need to be specified together. If `-macro` option is specified, this option is ignored.

`-line` *`lineNum`*

Line number. The `-line` and `-level` options always need to be specified together.

`-macro`

Apply to macro folder when this option is specified.

`-portList`

When specified, all port list folders are expanded or collapsed.

`-scope`

Toggle all folders in current scope. Toggle all folders in current file if `-macro` is specified.

`-stmt`

When specified, all statement folders are expanded or collapsed.

`-subProgram`

When specified, all sub-program folders are expanded or collapsed.

`-win` *`window`*

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcToggleFolder -win $_nTrace1 -collect -line 75 -level 1
srcToggleFolder -win $_nTrace1 -expand -line 101 -level 2
srcToggleFolder -win $_nTrace1 -design -expand -macro
srcToggleFolder -win $_nTrace1 -design -expand -implicitPort
srcToggleFolder -win $_nTrace1 -scope -collect -cmt
srcToggleFolder -win $_nTrace1 -scope -expand -portList
srcToggleFolder -win $_nTrace1 -scope -collect -subProgram
```

srcVHDLGetFullPath

Description

Get the original path in VHDL.

Syntax

```
srcVHDLGetFullPath -win windowId -scope fullPath
```

Arguments

`-scope fullPath`

Specify a full path for VHDL language.

`-win windowId`

Specify the window ID of the invoking *nTrace* window.

Value Returned

Return the original VHDL full path.

Example

```
srcVHDLGetFullPath -scope "system.foo"
srcVHDLGetFullPath -win $_nTrace1
```

Print

srcPrint

Description

Print the trace window content.

Syntax

```
srcPrint [-win window] [-header header] [-footer footer]
[-copy copyNum] [-orient landscape|portrait] [-paper
A4|A3|A2|A1|B|C|D|E] [-color on|off] {[-printer printerName]|[-
file printFile]} [-source curModule|actModule|files] [-tree
viewing|full] [-lineNum on|off] [-indicator on|off] [-annotate
on|off] [-column column] [-fontSize fontsize]
```

Arguments

`-annotate on|off`

Specify to turn printing the annotation value *on* or *off*.

`-color on|off`

Specify whether to use color printing. The default is *off*.

`-column column`

Specify the column in one page.

`-copy copyNum`

Specify the number of copies to print. The default is *1*.

`-file printFile`

Specify to print the *nTrace* window content to the specified print file.

`-fontSize fontsize`

Specify the font size.

`-footer footer`

Any printable message shown on the footer.

`-header header`

Any printable message shown on the header.

`-indicator on|off`

Specify to turn printing the indicator *on* or *off*.

`-lineNum on|off`

Specify to turn the printing line number *on* or *off*.

`-orient landscape|portrait`

Specify the orientation of the paper by selecting one of the options: **landscape** or **portrait**. The default is *landscape*.

`-paper A4|A3|A2|A1|B|C|D|E`

Specify the size of printer paper by selecting one of the paper size: A4, A3, | A2 | A1 | B | C | D | E. The default is *A4*.

`-printer printerName`

If specified, *nTrace* sends the window content to the specified printer. The default printer is *lp*.

`-source curModule|actModule|files`

Specify the source to display by selecting one from the options: curModule, actModule or files.

`-tree viewing|full`

If specified, the tree is printed out
viewing | full

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcPrint -win $current_window -printer lp
```

srcCapture

Description

Capture the current window to an image file in the PNG format.

Syntax

```
srcCapture [-win window] [-footer footer] [-region region] -file
filename
```

Arguments

`-file filename`

Specify the file name to save captured image.

`-footer footer`

Any printable message shown on the footer.

`-region region`

Specify the region to be printed or saved.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcCapture -footer "CPU" -file "/home/cpu.png"
```

srcAppendTextToMsgWin

Description

Append text strings to the *nTrace*'s message pane.

Syntax

```
srcAppendTextToMsgWin -text
```

Arguments

`-text`

Specify the text string.

Example

```
srcAppendTextToMsgWin -text "Append this string into message
pane"
```

Memory Definition Table

tfgDefineMem

Description

Define a memory.

NOTE: The **tfgDefineMem** command replaces the **tsDefineMem** command.

Syntax

```
tfgDefineMem -module mname -array arrayName [-banks value]
[-bit_range value_left value_right] -addr_range high_addr low_addr
[-g] [-ignoreSigNameCase]
```

Arguments

`-module mname`

Specify the memory module name.

`-array arrayName`

Specify memory array name.

`-banks value`

Specify the number of banks in the memory.

`-bit_range value_left value_right`

Specify the size of the memory.

`-addr_range high_addr low_addr`

Specify the address range.

`-g`

Declare as global memory.

`-ignoreSigNameCase`

Ignore signal name case.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDefineMem -module “sram” -array “mem”
tfgDefineMem -module mem -array_name ary -banks 16 -bit_range 7 0
-addr_range 1023 0 -g -ignoreSigNameCase
```

tfgUndefineMem

Description

Delete a previously defined memory.

NOTE: The **tfgUndefineMem** command replaces the **tsUndefineMem** command.

Syntax

```
tfgUndefineMem -module mname -array arrayName
```

Arguments

`-module` *`mname`*

Specify the memory module name.

`-array` *`arrayName`*

Specify memory array name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgUndefineMem -module mem -array_name ary
```

tfgDefineMemInit

Description

Define the memory initialization.

NOTE: The **tfgDefineMemInit** command replaces the **tsDefineMemInit** command.

Syntax

```
tfgDefineMemInit -module mname -array arrayName
{-time value | -clk clockExpression}
[-cond condExpression]
{-fromFile filename | -useValue 0|1|x | -config configFile}
[-addrLeft value_left -addrRight value_right]
[-format Hex|Bin] [-ps|-ns|-us] [-g] [-fromHdl]
```

Arguments

`-module mname`

Specify the memory module name.

`-array arrayName`

Specify memory array name.

`-time value | -clk clockExpression`

Specify the time the memory is initialized or specify the clock expression for memory initialization.

`-cond condExpression`

Specify the conditional expression for memory initialization.

`-fromFile filename | -useValue 0|1|x | -config configFile`

Specify the file name containing the memory initialization value or specify the value (0, 1, or x) to use for memory initialization or specify the file name containing the memory initialization configuration (configFile).

The format for configFile is:
[memorypathname] [filename]

For example:
Top.u0.u1 u0Mem.init
Top.u2.* u2Mem.init
* otherMem.init

Regular expressions can be used for path names in the configuration file; therefore, if multiple rules match for a path name, the first rule in the list that matches is used and subsequent matches are ignored.

`-format Hex|Bin`

Specify the format for the memory initialization file.

`-addrLeft value`

Specify the value for the left-most address to be initialized.

`-addrRight value`

Specify the value for the right-most address to be initialized.

`-ps|-ns|-us`

Specify the time unit as ps (picosecond), ns (nanosecond), or us (microsecond).

`-g`

Declare as global memory.

`-fromHdl`

Use the initialization value defined in HDL. This includes initialization by `$readmemb` or `$readmemh`, if it is specified in HDL.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDefineMemInit -module memory -array_name mem_array -clk clk
-time 10 -cond mem_init -addrLeft 0 -addrRight 1023 -format Hex
-useValue 10 -bank_sel bank_sel -ns -g
```

tfgDefineMemWrite

Description

Define the memory write conditions.

NOTE: The **tfgDefineMemWrite** command replaces the **tsDefineMemWrite** command.

Syntax

```
tfgDefineMemWrite -module mname -array arrayName
-clk clockExpression [-cond|-ncond condExpression]
{-addr | -naddr address}{-data | -ndata dataExpression}
[-bank_sel | -nbank_sel bsExpression] [-x_if_conflicts] [-g]
```

Arguments

`-module` *`mname`*

Specify the memory module name.

`-array` *`arrayName`*

Specify memory array name.

`-clk` *`clockExpression`*

Specify the clock expression for the memory write.

`-cond|-ncond` *`condExpression`*

Specify the conditional expression for the memory write. Use `-ncond` for a use new value. Optional.

`-addr|-naddr` *`address`*

Specify the write address signal. Use `-naddr` for a use new value.

`-data|-ndata` *`dataExpression`*

Specify the expression for the input data. Use `-ndata` for a new value.

`-bank_sel|-nbank_sel` *`bsExpression`*

Specify the expression for the bank selection condition. Use `-nbank_sel` for a new value.

`-x_if_conflicts`

Data is unknown if conflict.

`-g`

Declare as global memory.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDefineMemWrite -module memory -array_name mem_array -clk clk
-cond we -addr addr -data data -bank_sel bank_sel -x_if_conflicts
-g
```

tfgDefineMemRead

Description

Define the memory read conditions.

NOTE: The **tfgDefineMemRead** command replaces the **tsDefineMemRead** command.

Syntax

```
tfgDefineMemRead -module mname -array array_name
-clk clock_expression [-cond | -ncond cond_expression]
```

```
{-data | -ndata data_expression} [-bank_sel bs_expression]
[-oe | -noe output_enable] [-bank_sel bs_expression|-nbank_sel
bs_expression] [-latency_cycle num] [-delay_cycle num] [-g]
```

Arguments

`-module` *`mname`*

Specify the memory module name.

`-array_name` *`mem_array`*

Specify memory array name.

`-clk` *`clockExpression`*

Specify the clock expression for the memory bypass.

`-cond|-ncond` *`condExpression`*

Specify the conditional expression for the memory bypass. Use `-ncond` option if you want to use new value.

`-data|-ndata` *`data_expression`*

Specify the expression for the input data. Use `-ndata` option for a new value.

`-bank_sel` *`bsExpression`*

Specify the expression for the bank selection condition.

`-oe|-noe` *`output_enable`*

Specify the output enable signal. Use `-noe` option for a new value.

`-bank_sel|-nbank_sel` *`bsExpression`*

Specify the expression for the bank selection condition. Use `-nbank_sel` option for a new value.

`-g`

Declare as global memory.

`-latency_cycle` *`num`*

Specify the number of cycles until the output data is ready after the read is enabled. This provides support for pipeline reads.

`-delay_cycle` *`num`*

Specify the delay cycle.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDefineMemRead -module memory -array_name mem_array -clk clk
-cond re -addr addr -out out_data -oe oe -bank_sel bank_Sel -
latency_cycle 1 -delay_cycle 2-g
```

tfgDefineMemBypass

Description

Define the memory bypass conditions.

NOTE: The **tfgDefineMemBypass** command replaces the **tsDefineMemBypass** command.

Syntax

```
tfgDefineMemBypass -module mname -array_name array_name
-clk clock_expression [-cond | -ncond cond_expression]
{-data_in | -ndata_in data_expression} [-bank_sel bs_expression]
[-oe | -noe output_enable][-reg] -out output_signal
```

Arguments

`-array_name` *`arrayName`*

Specify the memory array name.

`-bank_sel` *`bsExpression`*

Specify the expression for the bank selection condition.

`-clk` *`clockExpression`*

Specify the clock expression for the memory bypass.

`-cond` *`condExpression`*

Specify the conditional expression for the memory bypass before the clock edge.

`-data_in|-ndata_in` *`dataExpression`*

Specify the expression for the input data.

`-module` *`mname`*

Specify the memory module name.

`-ncond` *`condExpression`*

Specify the conditional expression for the memory bypass after the clock edge.

`-noe output_enable`

Specify the output enable signal after the clock edge.

`-oe output_enable`

Specify the output enable signal before the clock edge.

`-out output_signal`

Specify the output data signal for the read.

`-reg`

Keep the value until the next array read.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDefineMemBypass -module sram -array_name mem
-clk @(posedge clk) -cond bypass -data_in temp_data -out dout
```

tfgDefineMemReadBypass

Description

Define bypass memory behavior.

NOTE: The **tfgDefineMemReadBypass** command replaces the **tsDefineMemReadBypass** command.

Syntax

```
tfgDefineMemReadBypass -module mname -array array_name
-clk clock_expression [-cond | -ncond cond_expression]
{-data | -ndata data_expression} [-bank_sel bs_expression]
[-oe | -noe output_enable] [-bank_sel bs_expression|-nbank_sel
bs_expression] [-wire | -reg] [-g]
```

Arguments

`-module mname`

Specify the memory module name.

`-array_name mem_array`

Specify the memory array name.

`-clk` *`clockExpression`*

Specify the clock expression for the memory bypass.

`-cond|-ncond` *`condExpression`*

Specify the conditional expression for the memory bypass. Use `-ncond` option for a new value.

`-data|-ndata` *`dataExpression`*

Specify the expression for the input data. Use `-ndata` option for a new value.

`-bank_sel` *`bsExpression`*

Specify the expression for the bank selection condition.

`-oe|-noe` *`outputEnable`*

Specify the output enable signal. Use `-noe` option for a new value.

`-bank_sel|-nbank_sel` *`bsExpression`*

Specify the expression for the bank selection condition. Use `-nbank_sel` option for a new value.

`-wire|-reg`

Specify wire or register like bypass memory read.

`-g`

Declare as global memory.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDefineMemReadBypass -module memory -array_name mem_array -clk
clk -cond re -data_in din -out dout -oe oe -bank_sel bank_sel
-wire -g
```

tfgSaveMemDef

Description

Save memory definition table to a file.

NOTE: The **tfgSaveMemDef** command replaces the **tsSaveMemDef** command.

Syntax

```
tfgSaveMemDef filename
```

Arguments

`filename`

Specify the file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSaveMemDef mdt.tbl
```

tfgLoadMemDef

Description

Load memory definition table from the saved file.

NOTE: The **tfgLoadMemDef** command replaces the **tsLoadMemDef** command.

Syntax

```
tfgLoadMemDef filename
```

Arguments

`filename`

Specify the file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgLoadMemDef mdt.tbl
```

tfgLoadMemDefFileList

Description

Load memory definition table from the list file.

NOTE: The **tfgLoadMemDefFileList** command replaces the **tsLoadMemDefFileList** command.

Syntax

```
tfgLoadMemDefFileList filename
```

Arguments

`filename`

Specify the file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgLoadMemDefFileList mdt.list
```

tfgListMemDef

Description

List the memory definition.

NOTE: The **tfgListMemDef** command replaces the **tsListMemDef** command.

Syntax

```
tfgListMemDef [0|1]
```

Arguments

`[0|1]`

0: all memory; 1: user defined only.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgListMemDef
```

tfgListMemModMap

Description

List the memory module mapping.

NOTE: The **tfgListMemModMap** command replaces the **tsListMemModMap** command.

Syntax

```
tfgListMemModMap [*/module] [*/memory array]
```

Arguments

`module`

Specify the module name.

`memory array`

Specify the memory array name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgListMemModMap * *
```

Message Frame

nsMsgAction

Description

This command helps to double-click on the specified node in the *tree* tab of the *Message* pane.

Syntax

```
nsMsgAction -tab trace|search|cmpl|intercon -index
nodeIndex
```

Arguments

`-tab trace|search|cmpl|intercon`

Specify the tree tab in the *Message* pane.

`-index nodeIndex`

Specify the index of the node to double-click the node.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nsMsgAction -tab trace -index {1 0}
nsMsgAction -tab cmpl -index {1 0}
```

NOTE: This is an existing TCL command, but it is not backward compatible for the trace results in the trace view. The index of the trace result changes because the tree level changes. That is, the leading trace does now have the scope node.

nsMsgDumpAU

Description

Dump the copied content of the selection.

Syntax

```
nsMsgDumpAU -copySel
```

Arguments

`-copySel`

Specify to dump the copied content of the selection.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nsMsgDumpAU -copySel
```

nsMsgGotoTime

Description

This command is used to jump to the time for the specified node in the *OneTrace* tab of the *Message* pane.

Syntax

```
nsMsgGotoTime -tab trace -index nodeIndex
```

Arguments

`-tab trace`

Specify the *OneTrace* tab in the *Message* pane.

`-index nodeIndex`

Specify the index of the node in the *OneTrace* tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nsMsgGotoTime -tab trace -index {0 0 0 1 }
```

nsMsgExpandItem

Description

This command expands the specified node in the tree tab of the *Message* pane. The command continues to trace when the traceable contributor node is expanded in the trace view for the first time.

Syntax

```
nsMsgExpandItem -tab trace|search|cmpl|intercon -index nodeIndex
```

Arguments

`-tab trace|search|cmpl|intercon`

Specify the tree tab in the *Message* pane.

`-index nodeIndex`

Specify the index of the node to expand it.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nsMsgExpandItem -tab trace -index {1 0}
```

NOTE: This is an existing TCL command, but provides continued tracing in this feature.

---

nsMsgSelect

Description

Logged when a *Message* frame is selected.

Syntax

```
nsMsgSelect -range {range1} {range2} { range3} …
```

Arguments

`-range {`*`rangeValues`*`}`

Specify the range(s) to be selected.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nsMsgSelect -range {1 0-1} {1 2 0-3}
```

nsMsgShowBitInfo

Description

This command shows the bit information for the specified node in the *OneTrace* tab of the *Message* pane.

Syntax

```
nsMsgShowBitInfo -tab trace -index nodeIndex
```

Arguments

`-tab trace`

Specify the *OneTrace* tab in the *Message* pane.

`-index nodeIndex`

Specify the index of the node to show bit information.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nsMsgShowBitInfo -tab trace -index {0 0}
```

nsMsgShowTreeColumn

Description

This command shows/hides the tree columns in the *OneTrace* tab of the *Message* pane.

Syntax

```
nsMsgShowTreeColumn -tab trace -columnId 1|2|3|4 -show 0|1
```

Arguments

`-tab trace`

Specify the *OneTrace* tab in the *Message* pane.

`-columnId 1|2|3|4`

Specify the column to show/hide.

- CoulmnId 1 - Specifies the value.
- CoulmnId 2 - Specifies the time.
- CoulmnId 3 - Specifies the fileline.
- ColumnId 4 - Specifies the scope

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nsMsgShowTreeColumn -tab trace -columnId 1 -show 0
nsMsgShowTreeColumn -tab trace -columnId 2 -show 1
```

nsMsgSwitchTab

Description

Logged when switching tabs in the *Message* frame.

Syntax

```
nsMsgSwitchTab -tab general|cmpl|trace||search|callstack|intercon
```

Arguments

`-tab general|cmpl|trace||search|callstack|intercon`

Specify a tab as the active tab from the *General*, *Compile*, *Trace*, *Search*, *Call Stack*, and *Interconnection* tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nsMsgSwitchTab -tab trace
```

X-propagation Commands

srcSetXpropLogFile

Description

Load the X-propagation log file in the Tcl flow.

Syntax

```
srcSetXpropLogFile logFile
```

Arguments

logFile

Specify the X-propagation log file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetXpropLogFile "/remote/XProp/visual_source_case/xprop.log
```

srcSetXpropOption

Description

Specify the X-propagation format in the Tcl flow.

Syntax

```
srcSetXpropOption tmerge|xmerge|xp_config
```

Arguments

`tmerge`

Use tmerge mode.

`xmerge`

Use xmerge mode.

`xp_config`

Use the **xp_config** setup file that contains the X-propagation configurations.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetXpropOption xmerge
```

Right-click Commands

srcAddBlockSignalToWaveform

Description

This command adds signals in the selected block to the synchronized *nWave* window.

Syntax

```
srcAddBlockSignalToWaveform -win window -line number
–nOffset number -range All|Sensitivity|Contents
```

Arguments

`-line number`

Specify the line number of the selected block.

`–nOffset number`

Specify the offset number for the macro.

`-range All|Sensitivity|Contents`

Specify signals to include by selecting one of the options from **All** (all signals), **Sensitivity** (signals in the sensitivity list), or **Contents** (all signals except signals in the sensitivity list).

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAddBlockSignalToWaveform -win $_nTrace1 -line 128 –nOffset 1
-range All
srcAddBlockSignalToWaveform -win $_nTrace1 -line 122
-range Sensitivity
srcAddBlockSignalToWaveform -win $_nTrace1 -line 116
-range Contents
```

srcAddSpaInstruToWave

Description

Add instrumented SPA signals to the *nWave* window.

Syntax

```
srcAddSpaInstruToWave -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal to add its instrumented signals.

`-win window`

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful, 0 otherwise.

Example

```
srcAddSpaInstruToWave -signal "/tb/top/i_sub1/i"
```

srcAddSrsnInstruToWave

Description

Add instrumented SRSN signals to the *nWave* window.

Syntax

```
srcAddSrsnInstruToWave -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal to add its instrumented signals.

`-win window`

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful, 0 otherwise.

Example

```
srcAddSrsnInstruToWave -signal "/tb/top/i_sub1/i"
```

srcCopyInstFullPath

Description

This command copies the full hierarchical name(s) of the selected instance(s) into the clipboard buffer.

Syntax

```
srcCopyInstFullPath -win window
```

Arguments

`-win` *`window`*

Specify the ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcCopyInstFullPath -win $_nTrace1
```

srcCopySignalFullPath

Description

This command copies the full hierarchical name(s) of the selected signal(s) into the clipboard buffer.

Syntax

```
srcCopySignalFullPath -win window
```

Arguments

`-win window`

Specify the ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcCopySignalFullPath -win $_nTrace1
```

srcFunctionStepin

Description

When this command is invoked, the calling function's stack information is listed in the **Function Stack** pane.

Syntax

```
srcFunctionStepin -win window
```

Value Returned

Value string if successful; otherwise, returns 0.

Example

```
srcFunctionStepin -win $_nTrace1
```

srcFunctionStepout

Description

Step out to the function calling position.

Syntax

```
srcFunctionStepout -win window
```

Value Returned

Value string if successful; otherwise, returns 0.

Example

```
srcFunctionStepout -win $_nTrace1
```

srcBackToLastScope

Description

Change the trace scope to the last scope.

Syntax

```
srcBackToLastScope
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcBackToLastScope
```

srcExpandTreebyLevel

Description

Expand tree by level for *nTrace* hierarchy tree. This Tcl command corresponds to the **View -> Hierarchy Tree by Level** command in the menu of the *nTrace* window and the **Expand Tree By Level** right-click command on the *nTrace Hierarchy Browser* pane.

Syntax

```
srcExpandTreebyLevel -win window -all|-level levelNumber
```

Arguments

`-all`

Expand all tree nodes.

`-level levelNumber`

Expand the selected instance in the *Hierarchy Browser* pane to the specified level. The valid level values are 1, 2, 3, 4 or 5. The default is *1*.

`-win window`

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcExpandTreebyLevel -win $_nTrace1 -all
srcExpandTreebyLevel -win $_nTrace1 -level "1"
```

srcRMBClkObj

Description

A call-back when right-clicking on an object so you can dynamically change the pop-up menu.

Syntax

`srcRMBClkObj [window ID in HEX] [SIGNAL` *`signalObject`*`|INST` *`instObject`*`]`

Arguments

`SIGNAL` *`signalObject`*

Specify the selected signal object.

`INST` *`instObject`*

Specify the selected instance object.

Value Returned

Window ID and the object if successful; otherwise, returns 0.

Example

```
srcRMBClkObj 36910178 {SIGNAL {system.i_cpu.clock}}
```

srcClearMessage

Description

Clear the messages in the *nTrace*'s message frame.

Syntax

```
srcClearMessage
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcClearMessage
```

srcAssertDebOpen

Description

Open the *Assertion Active Trace* form.

Syntax

```
srcAssertDebOpen -AssertName[assertion name] -beginTime [begin
time] -endTime [end time]
```

Arguments

`-AssertName`

Specify the assertion name.

`-beginTime`

Specify the assertion begin time.

`-endTime`

Specify the assertion end time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAssertDebOpen -AssertName tt.sb2.a_s2 -beginTime 5000 -endTime
25000
```

srcAssertDebClose

Description

Close the *Assertion Active Trace* form.

Syntax

```
srcAssertDebClose
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAssertDebClose
```

srcAssertDebExpExpand

Description

Show the expression label for lower level expressions.

Syntax

```
srcAssertDebExpExpand
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAssertDebExpExpand
```

srcAssertDebExpandAll

Description

Show the expression label for the lowest level expressions.

Syntax

```
srcAssertDebExpandAll
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAssertDebExpandAll
```

srcAssertDebCollapse

Description

Show the expression label for upper level expressions.

Syntax

```
srcAssertDebCollapse
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAssertDebCollapse
```

srcAssertDebSelect

Description

Select a signal/assertion/property/sequence in the *Assertion Active Trace* form to change the cursor time to the time that signal was evaluated.

Syntax

```
srcAssertDebSelect -signal [signal name] -line [line number] -word
[word number]
```

Arguments

`-signal`

Specify the signal name.

`-line`

Specify the specific line to apply the selection.

`-word`

Specify the specific word in the specified line.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAssertDebSelect -signal "tt.sb2.a_s2.p1.s2" -line 5 -word 39
```

srcAssertDebChangePath

Description

Double-click on the assertion/property/sequence in the *Assertion Active Trace* form to change the trace scope.

Syntax

```
srcAssertDebChangePath -signal [signal name] -line [line number] -
word [word number]
```

Arguments

`-signal`

Specify the signal name.

`-line`

Specify the specific line to apply the selection.

`-word`

Specify the specific word in the specified line.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcAssertDebChangePath -signal tt.sb2.a_s2.p1.s2 -line 5 -word 39
```

srcSignalViewAddSelectedToWave

Description

Add the selected signal(s) in the *nTrace* signal list pane to *nWave*.

Syntax

```
srcSignalViewAddSelectedToWave -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nTrace* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewAddSelectedToWave -win $_nTrace1
```

srcSignalViewSaveSelSignalsToFile

Description

Save the selected signal(s) in the *nTrace Signal List* pane to the specified file.

Syntax

```
srcSignalViewSaveSelSignalsToFile fileName
```

Arguments

`fileName`

Specify the filename (with full hierarchical path) to save the selected signals to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSignalViewSaveSelSignalsToFile "/xmpg1/home/LINUXAMD64/deb6.1/
sdProd/src/bin/novas/myfile.txt"
```

srcDisplayLibDefine

Description

Display the liberty file for the selected cell in the *nTrace* source code pane. Only single selection is supported.

Syntax

```
srcDisplayLibDefine
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcDisplayLibDefine
```

srcHBAddObjectToWave

Description

Add the selected IE(s) in the nTrace signal list pane to *nWave*.

Syntax

```
srcHBAddObjectToWave -IE
```

Arguments

`-IE`

Specify the IEs which need to be added to the *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcHBAddObjectToWave top.ia7.ib7.ic7.@{\top.ia7.ib7.ic7.in }
```

nSchema

Window

debSetLDCAction

Description

When the callback function is registered, the original double-click actions are invalid.

Syntax

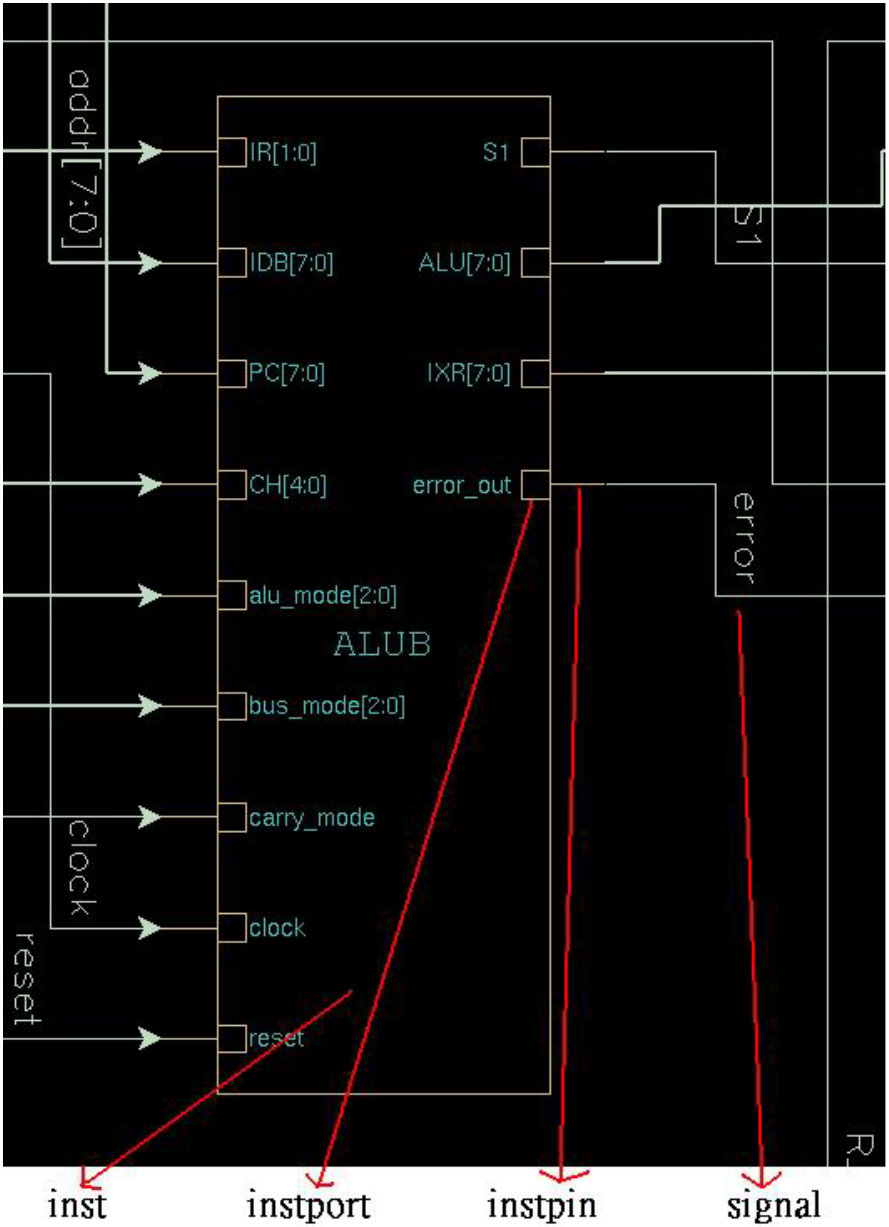
```
debSetLDCAction -func functionName
-winType [full|flattened|browser|all]
```

Arguments

`-func` *`functionName`*

Specify the user-defined function.

The syntax is ST_Defined_Func -inst|-instpin|instport|-signal|-port hierarchyName. Refer to the following example:



If you double-click on the instance, *nSchema* calls:

```
ST_Defined_LMB_Click2_Func -inst "system.i_CPU.i_ALUB"
```

If you double-click on the instpin, *nSchema* calls:

```
ST_Defined_LMB_Click2_Func -instpin
"system.i_CPU.i_ALUB.error_out"
```

If you double-click on the instport, *nSchema* calls:

```
ST_Defined_LMB_Click2_Func -instport
"system.i_CPU.i_ALUB.error_out"
```

If you double-click on the signal, *nSchema* calls:

```
ST_Defined_LMB_Click2_Func -signal
"system.i_CPU.i_ALUB.error"
```

`-winType [full|flattened|browser|all]`

Specify the window type of an *nSchema* window. The default window type is flattened window.

Value Returned

None.

Examples

```
proc ST_Defined_LMB_Click2_Func args {
    puts $args
    # customer can get the hierarchy_name here
}

debSetLDCAction -func ST_Defined_LMB_Click2_Func
debSetLDCAction -func "ST_Define_LMB_Click2_CB" -winType all
```

schCloseWindow

Description

Close the *nSchema* window.

Syntax

`schCloseWindow [-win` *`window`*`]`

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCloseWindow -win $_nSchema1
```

schCompile

Description

Compile the design.

Syntax

```
schCompile
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCompile
```

schConfigDelimiter

Description

Configure the delimiter at run-time.

Syntax

```
schConfigDelimiter -delim delim
```

Arguments

`-delim` *`delim`*

Specify the delimiter.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schConfigDelimiter -delim .
schConfigDelimiter -delim \
```

schCreateWindow

Description

Create a new *nSchema* window.

Syntax

```
schCreateWindow [-win window]
```

```
schCreateWindow [-delim delim] -scope scopeName
```

```
schCreateWindow [-win window] -browser [-rscope scopeName]
  [-signal listSignalName|-inst listInstname]
```

```
schCreateWindow [-win window] [-delim delim] [-r]
  -flattened [-inst {instName1} {instName2}...]
```

```
schCreateWindow [-win window] -trace
```

```
schCreateWindow [-win window] [-delim delim] [-r] -driver
  [-signal signalName|-inst instName]
```

```
schCreateWindow [-win window] [-delim delim] [-r] -load
  [-signal signalName|-inst instName]
```

```
schCreateWindow [-win window] [-delim delim] [-r] -conn
  [-signal signalName|-inst instName]
```

```
schCreateWindow [-win window] [-delim delim] [-r] -fanin
  [-signal signalName|-inst instName]
```

```
schCreateWindow [-win window] [-delim delim] [-r] -fanout [-signal
signalName|-inst instName]
```

```
schCreateWindow [-delim delim] -X [-time cursorTime] signalName
```

```
schCreateWindow [-delim delim] -activeFanIn [-time cursorTime]
-cycle cycleTime signalName [-nWave]
```

```
schCreateWindow [-delim delim] -busContention [-time cursorTime]
signalName
```

```
schCreateWindow [-win window] -hierFSM [-mode first|all]
```

```
schCreateWindow -win window -flattened -supplyNet "supply net
encoding name" -fileName filelName -lineNo number
```

```
schCreateWindow -clockTree -signal signalName -ff ffStopPin -latch
latchStopPin -macro macroStopPin -outputport stopOnOutputPort
-floating stopOnFloatingGates [-noSchema] [-color ignoreTreeColor]
[-instStop instName instPortName phase internalPhaseDelay]*
[-instIgnore instName instPortName]* [-instPassthrough instName
instPortName GeneratedClkPolarity]* [-instTrigger instName
```

```
instPortName Positive|Negative]* [-instExcepted instName
instPortName dont_size_cells|dont_touch_subtree|dont_buffer_net]*
[-cellName "cellName1" "cellName2"...] [-cellStop cellName
cellPortName phase internalPhaseDelay]* [-cellIgnore cellName
cellPortName]* [-cellPassthrough cellName cellPortName]*
[-cellTrigger cellName cellPortName Positive|Negative]*
[-cellExcepted cellName cellPortName dont_size_cells|
dont_touch_subtree|dont_buffer_net]* [-iccConstraint]
[-checkDesignConstant on|off] [-hierWin]
schCreateWindow [-win window] -ResetTree
schCreateWindow [-instpin fullInstPinName]
schCreateWindow [-win window] -flattened -path {instancePin1}
{instancePin2} ...
schCreateWindow [-win window] -driver | -load | -connectivity |
fanin | fanout -signal signalName -WithMatchBitWidth
```

Arguments

`-activeFanIn`

Create a schematic from the active fan-in cone with the value changed in the specified cycle time.

`-browser`

Create a browser window from the specified instance or signal. If not specified, use the selected instance or signal. If nothing is selected, an empty window is created.

`-busContention`

Create a schematic from the bus contention at the specified time.

`-cellExcepted [CellName] [PortName] [dont_size_cells| dont_touch_subtree|dont_buffer_net]`

Set the exception type of the cell as **dont_size_cells**, **dont_touch_subtrees** or **dont_buffer_nets**.

`-cellIgnore cellName cellInstPortName`

Specify the cell (*cellName*) or cell port (*cellInstPortName*) on the cell that should be ignored during clock tree extraction.

`-cellName "cellName1" "cellName2"...`

Specify the gate clock cell name(s). This option is only valid when using the `-clocktree` option.

`-cellPassthrough cellName cellInstPortName`

Specify the cell (*cellName*) or cell port (*cellInstPortName*) on the cell that should be passed through during clock tree extraction.

`-cellStop` *`cellName cellInstPortName`*

Specify the cell (*cellName*) or cell port (*cellInstPortName*) on the cell that should be stopped during clock tree extraction and specify its phase (*nonInvertRise, nonInvertFall, invertRise, or invertFall*) and a value for internal phase delay (*internalPhaseDelay*).

`-cellTrigger [`*`CellName`*`] [`*`PortName`*`] [Positive|Negative]`

Set the cell port as **Positive** or **Negative** in the **Trigger Type** selection field.

`-checkDesignConstant on|off`

If the option is *on*, clock tree extraction checks constant design signals. The default is *off*.

`-clockTree`

Generate a clock tree for the specified instance pin.

`-color` *`ignoreTreeColor`*

Specify the ignored tree color.

`-conn`

Create a schematic from the connectivity of the specified signal.

`-constant` *`constantName Radix Value`*

Specify the constant name, radix type (*Decimal*, *Hex*, *Octal*, or *Binary*), and value for a signal on the **Known Constant** section of the **SDC Settings** tab. If the constant value of the signal is not specified, the Tcl command is logged as ‘constantName dontcare’. Multiple constant names, radix, and value combinations may be specified.

`-constCellPort` *`constantName Radix Value`*

Specify the constant name, radix type (*Decimal*, *Hex*, *Octal*, or *Binary*), and value for a cell port on the **Known Constant** section of the **SDC Settings** tab. If the constant value of the cell port is not specified, the Tcl command is logged as ‘constantName dontcare’. Multiple constant names, radix, and value combinations may be specified.

`-cycle` *`cycleTime`*

Specify the cycle time.

`-delim` *`delimiter`*

Specify the delimiter of the specified hierarchical name.

`-driver`

Create a schematic from the driver of the specified signal.

`-fanin`

Create a schematic from the fan-in cone of the specified signal.

`-fanout`

Create a schematic from the fan-out cone of the specified signal.

`-ff` *`fStopPin`*

Specify the flip-flop pin types (**Clock pin**, **Non-clock pin**, or **All pins**) to stop during clock tree extraction.

`-fileName` *`filelName`*

Specify the file name where the supply net is located.

`-flattened [-inst {`*`instName1`*`} {`*`instName2`*`}...]`

Create a flattened window with one or more instances that are specified with the `-inst` option. If the `-inst` option is not specified, create a flattened window with the selected set in the window that is specified with the `-win` option.

`-floating ignore|stop`

Specify whether to stop on or ignore the floating gates.

`-hierFSM`

Extract the FSM instances.

`-hierWin`

Add an attribute on original Tcl commands to create a partial hierarchical view.

`-iccConstraint`

Switch to ICC mode in the CTS tab of the *Clock Tree Settings* form.

`-instExcepted [`*`InstanceName`*`] [`*`PortName`*`] [dont_size_cells|dont_touch_subtree|dont_buffer_net]`

Set the exception type of the instance as **dont_size_cells**, **dont_touch_subtrees**, or **dont_buffer_nets**.

`-instIgnore` *`instName instPortName`*

Specify the instance (*instName*) or instance port (*instPortName*) that should be ignored during clock tree extraction.

`-instPassthrough` *`instName instPortName GeneratedClkPolarity`*

Specify the instance (*instName*) or instance port (*instPortName*) that should be passed through during clock tree extraction and specify its generated clock polarity (*Invert* or *nonInvert*).

`-instpin` *`fullInstPinName`*

Specify the pin name as the clock root.

`-instStop` *`instName instPortName phase internalPhaseDelay`*

Specify the instance with *instName* or instance port with *instPortName* that should be stopped during clock tree extraction, and specify its phase (*nonInvertRise*, *nonInvertFall*, *invertRise*, or *invertFall*) and a value for the internal phase delay (*internalPhaseDelay*).

`-instTrigger [`*`InstanceName`*`] [`*`PortName`*`] [Positive|Negative]`

Set the instance port as **Positive** or **Negative** in the **Trigger Type** selection field.

`-latch latchStopPin`

Specify latch pin types (**Clock pin**, **Non-clock pin**, or **All pins**) to stop during clock tree extraction.

`-lineNo` *`number`*

Specify the line number where the supply net is located.

`-load`

Create a schematic from the load of the specified signal.

`-macro` *`macroStopPin`*

Specify the macro pins types (**Clock pin**, **Non-clock pin**, **All pins**, or **Passthrough**) to stop during clock tree extraction.

`-mode first|all`

Specify the mode for extracting the FSM instances. The options include: **first** and **all**. When **first** is specified, extract only the FSM instances of the current scope and the driver/load. When **all** is specified, extract all the FSM instances of the design.

`-noSchema`

Create a clock tree in the current window.

`-outputPort ignore|stop`

Specify whether to stop or ignore the primary output ports.

`-path` *`{instancePin1} {instancePin2}...`*

Specify all instance pins, including the hierarchy boundary ports, to create the path.

`-r`

The specified name is related to the current active scope. This is ignored when the current window is a partial flattened window.

`-rscope` *`scopeName`*

When specified, all the names are relative to this specified scope.

`-scope` *`scopeName`*

Create a schematic based on the specified scope. If the selected scope does not include any netlists, this Tcl command fails and the result is "0".

`-signal` *`signalName`*
Create a schematic from extracting a clock tree for the specified signal(s).

`-supplyNet "`*`supply net encoding name`*`"`
Specify the supply net encoding name.

`-time` *`cursorTime`*
Specify the cursor time. If not specified, the current cursor time is used.

`-trace`
Create a schematic from the trace result on the current or specified window.

`-win` *`window`*
Specify the window ID of the invoking *nSchema* window.

`-X`
Create a schematic from the trace X results at the specified time.

`-WithMatchBitWidth`
Display the warning message and stop creating the window for the signal if the specified signal does not exist in the design.

Value Returned

nSchema window ID if successful; otherwise, returns 0.

Examples

```
schCreateWindow -scope system.i_cpu
```
Create an *nSchema* window with `system.i_cpu` as the active scope.

```
schCreateWindow -trace
```
Create a partial flattened window with the trace results from the current window.

```
schCreateWindow -driver -signal {system.i_cpu.CH[4:0]}
```
Create a partial flattened window with the trace driver results of the specified signal `{system.i_cpu.CH[4:0]}`.

```
schCreateWindow -browser -rscope system IDB1 IDB2 IN2
```
Create a partial hierarchy window with the specified viewing objects.

```
schCreateWindow -browser ID1 ID2 ID3
```
Create a partial hierarchy window with the selected instances or signals under the scope system.

```
schCreateWindow -win $_nSchema2 -flattened
```

```
schCreateWindow -win $_nSchema2 -flattened -inst
"system.i_cpu.i_CCU.C20_reg"
```

Create an *nSchema* window with the selected instance(s) under the current window.

```
schCreateWindow -X -time 1000 system.i_cpu.data
```

Create a flattened window with trace X results.

```
schCreateWindow -activeFanIn -time 1000 -cycle 50 -nWave
system.i_cpu.data
```

Create a flattened window with the fan-in cone whose value changed within the time (950 - 1000), and also add the signals to *nWave*.

```
schCreateWindow -busContention -time 1000 system.i_cpu.data
```

Create a flattened window with the bus contention of the specified signal.

```
schCreateWindow -clockTree -signal "system.clock" -ff "Clock pin"
-latch "Control pin" -macro "Clock pin" -outputPort "stop" -
floating "ignore"
```

Create a clock tree from the signal system.clock, stop on the clock pin of flip-flops, the control pin of latches, the clock pin of marcos or primary output ports, and add the results to a new schematic window.

```
schCreateWindow -win $_nSchema2 -ResetTree
```

Create a reset tree from the selected signal in `$_nSchema2` and create a new schematic window with the reset tree result.

```
schCreateWindow -clockTree -signal "system.clock" -instIgnore
"system.i_cpu.i_ALUB.and\$2" "I0" -ff "Clock pin" -latch "Control
pin" -macro "Clock pin" -outputPort "stop" -floating "ignore" -
color ID_PURPLE1
```

Create a signal from system.clock with the **Show Ignored Tree** option on. The ignored tree shows in a schematic window with color `ID_PURPLE1.`

```
schCreateWindow -clockTree -signal "system.clock"
-instPassthrough \
"system.i_cpu.i_CCU.CCU:Always3:109:129:Reg""clock""NonInvert"\
-instPassthrough "system.i_cpu.i_CCU.CCU:Always3:109:129:Reg"
"C5" \-ff "Clock pin" -latch "Control pin" -macro "Clock pin" -
outputPort \"stop" -floating "ignore"

schCreateWindow -clockTree -signal "system.clock" "system.R_W"
"system.reset"  "system.VMA" "system.i_cpu.i_CCU.C6_reg.Q" -ff
"Clock pin" -latch "Control pin" -macro "Clock pin" -outputPort
"stop" -floating "ignore" -constant "system.reset" Binary 0 -
constant "system.clock" dontcare  -constCellPort  "AN2.B"
dontcare -constCellPort "AN2.A" dontcare

schCreateWindow -clockTree -signal "system.clock" -hierWin

schCreateWindow -instpin "top.i_alub.clock" -clockTree
```

Specify the pin name "top.i_alub.clock" as the clock root.

```
schCreateWindow -clockTree -instTrigger
"system.i_cpu.i_ALUB.i_alu.add_23.U7" "Z" "Positive"\
-instTrigger "system.i_cpu.i_ALUB.i_alu.add_23.U7" "B"
"Negative"\
-instExcepted "system.i_cpu.i_ALUB.i_alu.add_23.U18" "A"
"dont_size_cells"\
-signal "system.clock" -ff "Clock pin" -latch "Control pin"
-macro "Passthrough" -outputPort "stop" -floating "ignore"
-iccConstraint

schCreateWindow -clockTree -instStop
"system.i_cpu.i_ALUB.\\ACC_reg\[0\] " "CP" \
-instIgnore "system.i_cpu.i_ALUB.\\ACC_reg\[3\] " "CP" -signal \
"system.clock" -ff "Clock pin" -latch "Control pin" -macro \
"Clock pin" -outputPort "stop" -floating "ignore"

schCreateWindow -win $_windNo -flattened -supplyNet
"top.@{\$VDD(power_net)}"  -fileName /slowfs/sp_qa/ddts_case/
Siloti/SPS0149628/demo_1.0/supply_package_1.upf -lineNo 93

schCreateWindow -win $_nSchema1 -flattened -path
{top.x0.xi_toplevel4.\xiatd<7> .xi1.xmmn.D} \
{top.x0.xi_toplevel4.\xiatd<7> .xi1.y} \
{top.x0.xi_toplevel4.\xiatd<7> .xi3.a}

schCreateWindow -win $_nSchema1 -driver
-signal {system.i_cpu.CH[4]} -WithMatchBitWidth
```

schExport

Description

Export schematic diagram to file.

Syntax

`schExport [-win` *`window`*`] [-hdl` *`hdlFileName`*`] [-keynote` *`keynote`*`]`

Arguments

`-hdl` *`hdlFileName`*

Specify the output file.

`-keynote` *`keynote`*

Specify keynote string.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schExport -hdl "/tmp/test.v" -keynote "This is my third cmd file
test..."
```

schGetAllWindows

Description

Get all *nSchema* window IDs.

Syntax

```
schGetAllWindows
```

Value Returned

List the window IDs of all *nSchema* windows.
window1 (hide in *nTrace*), window2 (newest *nSchema* window),
...windowN (first *nSchema* window).

Example

```
set sch_window_list schGetAllWindows
```

schGetCurrentWindow

Description

Get the current active *nSchema* window.

Syntax

```
schGetCurrentWindow
```

Value Returned

Current *nSchema* window ID.

Example

```
set _nSchema1 [schGetCurrentWindow]
```

schGetCurrentWindowData

Description

Get the current window data.

Syntax

```
schGetCurrentWindowData -win window
```

Arguments

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

Current window data.

Example

```
schGetCurrentWindowData -win $_nSchema2
```

schGetCurrentWindowLib

Description

Get the current design name.

Syntax

```
schGetCurrentWindowLib -win window
```

Arguments

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

Current design name.

Example

```
schGetCurrentWindowLib -win $_nSchema2
```

schGetXWindowId

Description

Return the X window ID of the ID returned from *schCreate* window.

Syntax

```
schGetXWindowId [-id $xwid]
```

Arguments

`-id $xwid`

Tcl window variable ID.

Value Returned

X window ID if successful. Set the result of the command to *0x0* and return *CMDOBJ_ERROR* otherwise.

Example

```
schGetXWindowId -id $_nSchema1
```

schLoadSDCFile

Description

Load the specified SDC file.

Syntax

```
schLoadSDCFile [-win window] [-load fileName] [-design designTop]
[-delim delimiter]
```

Arguments

`-delim delimiter`

Specify the delimiter.

`-design designTop`

Specify the SDC top design name. If not specified, the default is the first top design.

`-load fileName`

Load the specified SDC file.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schLoadSDCFile -win $_nSchema4 -load "/slowfs/sp_dq2/qa/demo/61/
verilog/gate/sdc_system.sdc" -design "rds_testpd_topio" -delim "/
"
```

schLockRedraw

Description

Lock or unlock *nSchema* redrawing. When redraw is locked, *nSchema* does not update the schematic. The schematic is updated after a locked redraw is unlocked.

Syntax

`schLockRedraw -lock [on/off] -win` *`window`*

Arguments

`-lock`

If the value is *on*, *nSchema* redraw is locked; if the value is *off*, *nSchema* redraw is unlocked.

`-win`

Specify the window ID of the invoking nSchema window.

Example

```
schLockRedraw -lock on -win $_nSchema2
```

schLowerWindow

Description

Place the *nSchema* window at the bottom of the window stacking order.

Syntax

```
schLowerWindow [-win window]
```

Arguments

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schLowerWindow -win $_nSchema2
```

schRaiseWindow

Description

Raise the *nSchema* window to the top of the window stacking order.

Syntax

```
schRaiseWindow [-win window]
```

Arguments

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRaiseWindow -win $_nSchema2
```

schResizeWindow

Description

Resize the active window.

Syntax

```
schResizeWindow x y w h -win window
```

Arguments

`x y w h`

Specify the coordinates for the window screen by x and y; specify the width and height for the window by w and h.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schResizeWindow 300 300 300 300 -win $_nSchema2
```

schSave

Description

Save the design sheet.

Syntax

```
schSave [-win window] -bookDir directory –nodePath path
```

Arguments

`-bookDir directory`

Specify the directory of the design book.

`–nodePath path`

Specify the detail information of the design sheet.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSave -win $_nSchema4 -bookDir "/xmpg1/home/danghui_lin/
program/work.lib++/VerdiLib.lib++/ALUB.Book++" -nodePath
"ALUB.sch"
```

schSetCurrentWindow

Description

Set the specified *nSchema* window ID as the current window.

Syntax

```
schSetCurrentWindow windowID
```

Value Returned

Current *nSchema* window ID.

Example

```
schSetCurrentWindow $_nSchema1
```

Trace

schActiveFanIn

Description

Create a schematic from the active fan-in cone with the value changed in the specified cycle time.

Syntax

```
schActiveFanIn
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schActiveFanIn
```

schBackwardTraceHistory

Description

Step backward through the trace history and change the view accordingly.

Syntax

```
schBackwardTraceHistory [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schBackwardTraceHistory
```

schCollapse

Description

Collapse the forward/backward connections on the instance port. This command is only valid on flattened window.

Syntax

```
schCollapse [-win window] [-forward | -backward] [-instpin instancePin]
```

Argument

`-forward`

Collapse all forward connections.

`-backward`

Collapse all backward connections.

`-instpin instancePin`

Specify the instance pin.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCollapse
```

schDumpTracedLeafPin

Description

After driver, load, or connectivity is traced, dump the full path of the leaf pins in the latest trace result to the **.verdiLog/leafpin.rpt** file.

Syntax

```
schDumpTracedLeafPin [-delim delimiter]
```

Argument

`-delim delimiter`

The delimiter for the pin full name. The default is ".".

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSelect -win $_nSchema5 -instpin "system.i_cpu.\\h478/i_ALUB
.U276" "Z"
schTraceLoad -win $_nSchema5
schDumpTracedLeafPin
```

schDriverCount

Description

Show the number of drivers.

Syntax

```
schDriverCount -win window
```

Argument

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schDriverCount -win $_nSchema5
```

schExpandDriver

Description

Trace the driver(s) of the specified signal or instance.

NOTE: When encountering the module boundary, FSM Block, or specified instance ports in the stop list, the trace stops.

Syntax

```
schExpandDriver -win window [-stopList instPort1 instPort2...]
[-signal signalName|-inst instName]
```

Arguments

`-inst instName`

Specify the instance with the driver to be traced.

`-signal signalName`

Specify the signal with the driver to be traced.

NOTE: If no instance or signal is specified, the currently selected signal is set as the start point for tracing.

`-stopList instPort1 instPort2...`

Specify the instance ports that the trace stops at with full hierarchical names.

`-win window`

The window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schExpandDriver -stopList {system.i_cpu.i_ALUB.U235.A}
{system.i_cpu.i_ALUB.U239.B}
```

schExpandLoad

Description

Trace the load(s) of the specified signal or instance.

NOTE: When encountering the module boundary, FSM Block, or specified instance ports in the stop list, the trace stops.

Syntax

```
schExpandLoad -win window [-stopList instPort1 instPort2...]
[-signal signalName|-inst instName]
```

Arguments

`-inst` *instName*

Specify the instance with the load to be traced.

`-signal` *signalName*

Specify the signal with the load to be traced.

NOTE: If no instance or signal is specified, the currently selected signal is set as the start point for tracing.

`-stopList` *instPort1 instPort2...*

Specify the instance ports that the trace stops at with full hierarchical names.

`-win` *window*

The window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schExpandLoad -stopList {system.i_cpu.i_ALUB.U235.A}
{system.i_cpu.i_ALUB.U239.B}
```

schForwardTraceHistory

Description

Step forward through the trace history and change the view accordingly.

Syntax

```
schForwardTraceHistory [-win window]
```

Argument

`-win` *window*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schForwardTraceHistory
```

schGetUncompleteFanInPin

Description

Get names of the fan-in pins that are not complete.

Syntax

```
schGetUncompleteFanInPin -win window
```

Argument

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

The name list of the fan-in pins that are not complete.

Example

```
schGetUncompleteFanInPin -win $_nSchema3
```

schIgnoreScope

Description

Ignore the specified scopes when tracing.

Syntax

```
schIgnoreScope -ignore ignoreScope
```

Argument

`-ignore` *`ignoreScope`*

Specify the scope to be ignored.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schIgnoreScope -ignore "system.i_cpu"
```

schIterCellView

Description

Iterate current window cellview ID.

Syntax

```
schIterCellView [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

Current *nSchema* window cellview ID in hex format.

Example

```
schIterCellView -win $_nSchema2
```

schIterInst

Description

Iterate current window instance ID.

Syntax

```
schIterInst [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

An iterator that can traverse all instance IDs one by one.

Example

```
schIterInst -win $_nSchema3
```

schIterNet

Description

Iterate current window net ID.

Syntax

```
schIterNet [-win window]
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

An iterator which can traverse all net IDs one by one.

Example

```
schIterNet -win $_nSchema3
```

schIterNext

Description

Iterate next object for the iterator ID.

Syntax

```
schIterNext iteratorID [-name] [-visible] [-noRTLInst] [-bracket]
```

Arguments

`iteratorID`

The iterator ID.

`-name`

The name of the object.

`-visible`

Specify the types of instances to be displayed.

`-noRTLInst`

Filter out the RTL instances of the object.

`-bracket`

Keep the bracket of the object.

Value Returned

Object ID corresponding to the iterator.

Examples

`schIterInst -win $_nSchema2`
It returns *b88b710*.

`schIterNext b88b710`
It returns *ba3b8cc*.

`schIterNext b88b710`
It returns *ba3bb30*.

```
....
schIterNext 4105a28 -name -noRTLInst -bracket
```
example1: output1 ==>{\ACC_reg[0] }
example2: output2 ==>{zero_flag_reg}

schIterPort

Description

Iterate current window port ID.

Syntax

`schIterPort [-win window]`

Argument

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

An iterator which can traverse all port IDs one by one.

Example

```
schIterPort -win $_nSchema3
```

schIterQuery

Description

Query the object information received from *schIterNext*.

Syntax

```
schIterQuery objectID
```

Arguments

`objectID`
The object ID.

Value Returned

Information such as name, count, or type. Exact results depend on the iterate object type.

Example

```
schIterInst -win $_nSchema2
```
It returns *b88b710*.

```
schIterNext b88b710
```
It returns *ba3b8cc*.

```
schIterQuery ba3b8cc
```
It returns *i_cpu CPU* symbol hierarchy.

schIterStop

Description

Stop an iterator and free its memory.

Syntax

```
schIterStop iteratorID
```

Argument

`iteratorID`

The iterator ID.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schIterInst -win $_nSchema2
```

It returns *b88b710*.

```
schIterNext b88b710
```

It returns *ba3b8cc*.

```
schIterStop b88b710
```

It returns *1*.

schListPorts

Description

Get all instance ports of one instance.

Syntax

```
schListPorts [-win window] -inst shortInstanceName
```

Argument

`-win window`

Specify the window ID of the invoking *nSchema* window.

`-inst shortInstanceName`

The instance name in short format.

Value Returned

Port list.

Example

```
schListPorts -win $_nSchema2 -inst "i_cpu"
```

The value returned is *R_W VMA addr[7:0] clock data[7:0] reset*.

NOTE: The flattened window is not supported.

schLoadCount

Description

Show the number of loads.

Syntax

```
schLoadCount [-win window]
```

Argument

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schLoadCount -win $_nSchema5
```

schLoadFlatNetlistByFile

Description

Load the instance into a new flattened window.

Syntax

```
schLoadFlatNetlistByFile –file filename –delim delimiter
```

Argument

`–file filename`

Specify the input file.

`–delim delimiter`

Specify the delimiter.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schLoadFlatNetlistByFile -file "NetlistFile.txt"
```

schNextTraced

Description

View next instance or replace the window with the next scope that contains a trace result.

Syntax

```
schNextTraced [-win window] [-scope]
```

Argument

`-scope`

Replace the window with the next scope with a trace result.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

When `-scope` option is specified: a list of scope and instance names if successful; otherwise, returns 0.

Example

```
schNextTraced
schNextTraced -scope
```

schPowerActiveTrace

Description

Find the real driver(s) of the specified power signal.

Syntax

```
schPowerActiveTrace [-win window] [-signal signalName]
```

Argument

`-signal signalName`

Specify the signal with full hierarchy name.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPowerActiveTrace -win $_nSchema1 -signal
"system.i_cpu.i_CCU.clock"
```

schPowerTraceConnectivity

Description

Find the connectivity(s) of the specified power signal.

Syntax

```
schPowerTraceConnectivity [-win window] [-signal signalName]
```

Argument

`-signal signalName`

Specify the signal with full hierarchy name.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPowerTraceConnectivity -win $_nSchema1 -signal
"system.i_cpu.i_CCU.clock"
```

schPowerTraceDriver

Description

Find the driver(s) of the specified power signal.

Syntax

```
schPowerTraceDriver [-win window] [-signal signalName]
```

Argument

`-signal signalName`

Specify the signal with full hierarchy name.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPowerTraceDriver -win $_nSchema2 -signal
"system.i_cpu.i_CCU.clock"
```

schPowerTraceLoad

Description

Find the load(s) of the specified power signal.

Syntax

```
schPowerTraceLoad [-win window] [-signal signalName]
```

Argument

`-signal signalName`

Specify the signal with full hierarchy name.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPowerTraceLoad -win $_nSchema3 -signal
"system.i_cpu.i_CCU.clock"
```

schPrevTraced

Description

View previous instance or replace the window with previous scope in the traced result.

Syntax

```
schPrevTraced [-win window] [-scope]
```

Argument

`-scope`

Replace the window with the previous scope with the trace result.

`-win` *window*

Specify the window ID of the invoking *nSchema* window.

Value Returned

When `-scope` option is specified: a list of scope and instance names if successful, 0 otherwise.

Example

```
schPrevTraced
schPrevTraced -scope
```

schPropSetting

Description

Set propagation values for the intended signal/cell.

Syntax

```
schPropSetting -win window -name SignalCellInstPinName [-radix
Decimal|Hex|Octal|Binary] -signal|-cell|-instpin [-value value]
[-delete] [-extract] [-load SDCFileName] [-report] [-saveReport
FileName] [-stopAtReset] [-stopAtSet]
```

Arguments

`-delete`

Delete the selected signal, cell, and instance pin.

`-extract`

Perform the value propagation.

`-load` *`SDCFileName`*

Load the SDC file contents.

`-name` *`SignalCellInstPinName`*

Specify the name of the signal, cell, and instance pin.

`-radix Decimal|Hex|Octal|Binary`

Specify the radix value of the signal, cell, and instance pin.

`-report`

Show the differences between the FSDB value and the propagated value in the *Value Propagation Report* form.

`-saveReport` *`filename`*

Specify the file to save the propagated results.

`-signal|-cell|-instpin`

Identify whether the argument after `-name` is a signal, cell, or instance pin.

`-stopAtReset`

The value propagation stops at the reset pin.

`-stopAtSet`

The value propagation stops at the set pin.

`-value` *`value`*

Specify the value of the signal, cell, and instance pin.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

None.

Example

```
schPropSetting -win $_nSchema3 -name A.B.C -radix Decimal
-instpin -value 0

schPropSetting -win $_nSchema3 -saveReport report.txt

schPropSetting -win $_nSchema2 -extract -stopAtReset -stopAtSet
```

schQueryObj

Description

Query the information of instance pins and nets.

Syntax

```
schQueryObj -instport fullInstancePortName
schQueryObj -domainToId sourceName (r)|(f)
```

Argument

`-domainToId`

The ID of clock domain.

`-instport` *`fullInstancePortName`*

Specify the full hierarchical path of the instance and the port name.

`sourceName` `(r)|(f)`

Identify the source name with either rising or falling edge.

Value Returned

For port and instance port, it returns short name, direction, and number of bits.

For net, it returns clock domain ID(s).

Example

```
schQueryObj -instport "system.i_cpu.clock"
```

It returns *clock in 1*.

```
schQueryObj -domainToId {system.clock (r)} {system.clock (f)}
```

It returns *2 1*.

schQuickTrace2Point

Description

Specify the *Trace Two Points* where the criteria to trace between two points can be specified. The points can be either instance ports or signals.

Syntax

```
schQuickTrace2Point -action switch
```

Argument

`-action switch`

Used to switch the values in from/to text fields.

`-action trace`

Used to trace the 2 nodes.

Value Returned

1 as success, 0 as failure

Example

```
schQuickTrace2Point -action switch
schQuickTrace2Point -action trace
```

schRemoveTraceColor

Description

Remove the highlight of the newly expanded instance in *nSchema* window.

Syntax

`schRemoveTraceColor -win` *`window`*

Arguments

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRemoveTraceColor -win $_nSchema2
```

schResetHistory

Description

Reset all the trace history.

Syntax

```
schResetHistory [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schResetHistory
```

schSetTraceOptions

Description

Set trace stop options.

Syntax

```
schSetTraceOptions [-win window] [-stopOnModuleBoundary on|off]
[-stopOnFSM on|off] [-stopOnMOSCell on|off] [-2PointsStopOnFF
on|off]
```

Argument

`-stopOnModuleBoundary on|off`

Control whether to stop on module boundary when tracing fan-in/fan-out cones and driver/load/connectivity commands.

`-stopOnFSM on|off`

Control whether to stop on the FSM when tracing fan-in/fan-out cone.

`-stopOnMOSCell on|off`

Control whether to stop on the MOS cell when tracing fan-in/fan-out cone.

`-2PointsStopOnFF on|off`

Control whether to stop in the FF when trace from 2 points.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetTraceOptions -stopOnFSM off
schSetTraceOptions -stopOnMOSCell on
```

schTraceConnectivity

Description

Find the connectivity(s) of the specified signal or instance. If no signal or instance is specified, the action is applied to the selected signal or instance.

Syntax

```
schTraceConnectivity [-delim delim] [-win window] [-r] [-range
start end] [-signal signalName|-inst instName]
```

Arguments

`-delim` *`delim`*

Delimiter of the name list.

`-r`

Generate the full scope name when the instance/signal name does not have the scope name.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected object is an instance, or `-inst` option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal name.

`-inst` *`instName`*

Specify the lowest hierarchy instance name.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance names for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of windows.

Example

```
schTraceConnectivity -range 1 2 -signal {system.CH[4:0]}
schTraceConnectivity -r -signal {CH[4:0]}
```

schTraceDriver

Description

Find the driver(s) of the specified signal or instance. If a signal or instance is not specified, the action is applied to the selected signal or instance.

Syntax

`schTraceDriver [-delim` *`delim`*`] [-win` *`window`*`] [-r] [-range` *`start end`*`] [-signal` *`signalName`*`|-inst` *`instName`*`]`

Arguments

`-delim` *`delim`*

Delimiter of the name list.

`-r`

Generate the full scope name when the instance/signal name does not have the scope name.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected object is an instance, or `-inst` option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal name.

`-inst` *instName*

Specify the lowest hierarchy instance name.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance names for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of windows.

Example

```
schTraceDriver -range 1 2 -signal {system.CH[4:0]}
schTraceDriver -r -signal {CH[4:0]}
```

schTraceFanIn

Description

Find the fan-in cone of the specified signal or instance. If a signal or instance is not specified, the action is applied to the selected signal or instance.

Syntax

```
schTraceFanIn [-delim delim] [-win window] [-r] [-range start end]
[-signal signalName|-inst instName]
```

Arguments

`-delim` *delim*

Delimiter of the name list.

`-r`

Generate the full scope name when the instance/signal name does not have the scope name.

`-range` *start end*

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected object is an instance, or `-inst` option is specified, this range option is ignored.

`-signal` *signalName*

Specify the signal name.

`-inst` *instName*

Specify the lowest hierarchy instance name.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance names for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of windows.

Example

```
schTraceFanIn -range 1 2 -signal {system.CH[4:0]}
schTraceFanIn -r -signal {CH[4:0]}
```

schTraceFanOut

Description

Find the fan-out cone of the specified signal or instance. If a signal or instance is not specified, the action is applied to the selected signal or instance.

Syntax

```
schTraceFanOut [-win window] [-delim delim] [-r] [-range start end]
[-signal signalName|-inst instName]
```

Arguments

`-delim` *delim*

Delimiter of the name list.

`-r`

Generate the full scope name when the instance/signal name does not have the scope name.

`-range` *start end*

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected object is an instance, or `-inst` option is specified, this range option is ignored.

`-signal` *signalName*

Specify the signal name.

`-inst` *instName*

Specify the lowest hierarchy instance name.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance names for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of windows.

Example

```
schTraceFanOut -range 1 2 -signal {system.CH[4:0]}
schTraceFanOut -r -signal {CH[4:0]}
```

schTraceLoad

Description

Find the load(s) of the specified signal or instance. If a signal or instance is not specified, the action is applied to the selected signal or instance.

Syntax

```
schTraceLoad [-delim delim] [-win window] [-r] [-range start end]
[-signal signalName|-inst instName]
```

Arguments

`-delim delim`

Specify the delimiter of the name list.

`-r`

Generate the full scope name when the instance/signal name does not have the scope name.

`-range start end`

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected object is an instance, or `-inst` option is specified, this range option is ignored.

`-signal signalName`

Specify the signal name.

`-inst instName`

Specify the lowest hierarchy instance name.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance names for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of windows.

Example

```
schTraceLoad -range 1 2 -signal {system.CH[4:0]}
schTraceLoad -r -signal {CH[4:0]}
```

schTraceReport

Description

Generate a report for **2 Points**, **Fan In**, **Fan Out**, or for **Clock Tree** schematics. When SDF is loaded, the report can also show the delay of each path.

Syntax

```
schTraceReport [-2point] [-fanin] [-fanout] [-clockTree]
[-win window][-delim delim] [-stopOnFF on|off] [-stopOnTri on|off]
[-createWindow] -from "instanceName" "portName"
-to "instanceName" "portName"
```

Arguments

`-2point`

Generate the report in 2 points mode.

`-clockTree`

Generate the report in clock tree mode.

`-createWindow`

The new schematic window which is used for the traced report.

`-fanin`

Generate the report in fan-in mode.

`-fanout`

Generate the report in fan-out mode.

`-from`

Specify one instance loading port to start with.

`-stopOnFF on|off`

Indicates whether the trace stops at an FF boundary and/or a latch when tracing between two points.

`-stopOnTri on|off`

Indicates whether the trace stops at a TriState when tracing between two points.

`-to`

Specify one instance driving port to end with.

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schTraceReport -2point -win $_nSchema2 -delim "." -stopOnFF on
-stopOnTri on -from
"picoJavaII.monitor.trace_debug_monitor.trace_debug_monitor:SigOp
0:125:125:Combo" "OH_pj_clk" -to
"picoJavaII.cpu.icu.icu_dpath.ibuffer.ibuf_8.ibuf_len_flop"
"out\[3:0\]"

schTraceReport -clockTree -win $_nSchema2 -delim "." -stopOnFF on
-stopOnTri on -createWindow -from "test.test:Init3:108:119:Init"
"O0"
```

schTraceX

Description

Find the *X* value for the specified signal.

Syntax

`schTraceLoad [-delim` *`delim`*`] [-win` *`window`*`] [-signal` *`signalName`*`]`

Arguments

`-delim` *`delim`*

Specify the delimiter of the name list.

`-signal signalName`

Specify the signal.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schTraceX -delim "." -win $_nSchema4 -signal
"system.i_cpu.i_PCU.TDB\[7:0\]"
```

schTrace2Nodes

Description

Find the connected instances by tracing from a specified signal or instance port to the destination.

Syntax

```
schTrace2Nodes -win window [-delim delim] [-createWindow]
[-passCom signalNameList | instancePortList]
[-byPassCom signalNameList | instancePortList]
[-inst] -from fromSignalName -to toSignalName
[-lowestScope | -scope scopeName]
```

Arguments

`-byPassCom instancePortList`

Specify the instances not to be passed through.

`-byPassCom signalNameList`

Specify the signals not to be passed through.

`-createWindow`

Create a new *nSchema* window for the trace results.

`-delim delim`

Delimiter of the full hierarchical signal name.

`-from` *`fromName`*

Specify the from point for the full hierarchical signal name or the instance port.

`-inst`

Return all hierarchical instance names between these two signals as a result.

`-lowestScope`

Limit the trace to the lowest common scope between the from-point and to-point.

`-passCom instancePortList`

Specify the instances to be passed through.

`-passCom signalNameList`

Specify the signals to be passed through.

`-scope` *`scopeName`*

Limit the trace to paths within the specified scope.

`-to` *`toName`*

Specify the to point for the full hierarchical signal name or the instance port.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

- When tracing between the following nodes, the value returned is the same as the **schTrace2Signals** command:
  - from signal to signal
  - from signal to module instance port (scope port)
  - from module instance port to signal (scope port)
  - from module instance port (scope port) to module instance port (scope port)
- When tracing between the following node, the value returned is the same as the **schTrace2Points** command:
  - from primitive instance port to primitive instance port
- When tracing between the following nodes, the value returned is 1 if successful, otherwise, returns 0:
  - from primitive instance port to signal
  - from signal to primitive instance port
  - from primitive instance port to module instance port (scope port)

- from module instance port (scope port) to primitive instance port

Example

```
schTrace2Nodes -win $_nSchema2 –delim .
-from "system.i_cpu.i_ALUB.bus_mode\[2\]"
-to "system.i_cpu.i_ALUB.U251.A"
```

schTrace2Points

Description

Find the connected instances by tracing from a specified source to the destination.

Syntax

```
schTrace2Points [-win window] [-delim delim] [-r] [-stopOnFF
on|off] [-stopOnTri on|off] [-createWindow] [-passCom
instancePortList] [-byPassCom instancePortList] -from instancePort
-to instancePort [-lowestScope|-scope scopeName] [-passFFDataPort
on|off] [-passFFClockPort on|off] [-passFFControlPort on|off]
[-passFFSetResetPort on|off] [-passFFOtherPort on|off]
```

Arguments

`-byPassCom` *instancePortList*

Specify the instance port list that the path must not pass through.

`-createWindow`

Create a new *nSchema* window for trace result.

`-delim` *delim*

Delimiter of the instance name.

`-lowestScope`

Limit the trace to the lowest common scope between the from-point and to-point.

`-passCom` *instancePortList*

Specify the instance port list to be passed through.

`-passFFClockPort on|off`

Specify whether to pass through (*on*) or stop at (*off*) the clock port of flip-flops. This option works when *off* is specified for the `-stopOnFF` option.

`-passFFControlPort on|off`

Specify whether to pass through (*on*) or stop at (*off*) the control port of flip-flops. This option works when *off* is specified for the `-stopOnFF` option.

`-passFFDataPort on|off`

Specify whether to pass through (*on*) or stop at (*off*) the data port of flip-flops. This option works when *off* is specified for the `-stopOnFF` option.

`-passFFOtherPort on|off`

Specify whether to pass through (*on*) or stop at (*off*) other ports of flip-flops. This option works when *off* is specified for the `-stopOnFF` option.

`-passFFSetResetPort on|off`

Specify whether to pass through (*on*) or stop at (*off*) the set/reset port of flip-flops. This option works when *off* is specified for the `-stopOnFF` option.

`-r`

Generate the full scope name when the instance/signal name does not have the scope name.

`-scope` *`scopeName`*

Limit the trace to paths within the specified scope.

`-stopOnFF on|off`

Specify whether to traverse through FF.

`-stopOnTri on|off`

Specify whether to traverse through Tristate.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance names for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of windows.

Example

```
schTrace2Points -r -from regA D -to regD Q -createWindow -passCom
"system.i_cpu.i_ALUB.i_alu.U149.Z" -byPassCom
"system.i_cpu.i_ALUB.i_alu.U321.Z"

schTrace2Points -win $_nSchema2 -delim "." -stopOnFF off
-passFFDataPort on -passFFClockPort on -passFFControlPort off
-passFFSetResetPort off -passFFOtherPort off -stopOnTri on
-createWindow -from "CPU.C001" "A" -to "CPU.i_PCU.U190" "Z"
```

schTrace2Signals

Description

Find the connected instances by forward tracing from the specified signal to the destination.

Syntax

```
schTrace2Signals [-win window] [-delim delim] [-createWindow]
[-passCom signal1Name signal2Name...] [-byPassCom signal1Name
signal2Name...] [-inst] -from fromSignalName -to toSignalName
-createWindow [-lowestScope|-scope scopeName][-hideNeighbor
on|off]
```

Arguments

`-byPassCom signal1Name signal2Name...`

Specify the signals not to be passed through.

`-createWindow`

Create a new *nSchema* window for the trace results.

`-delim delim`

Delimiter of the full hierarchical signal name.

`-from fromSignalName`

Specify the from point for the full hierarchical signal name.

`-inst`

Return all hierarchical instance names between these two signals as result.

`-lowestScope`

Limit the trace to the lowest common scope between the from-point and to-point.

`-passCom signal1Name signal2Name...`

Specify the signals to be passed through.

`-scope scopeName`

Limit the trace to paths within the specified scope.

`-to toSignalName`

Specify the to point for the full hierarchical signal name.

`-hideNeighbor on|off`

Specify whether to show neighbor cells of the traced signal. If set to *on*, neighbor cells of the traced signal are not shown; if set to *off*, neighbor cells of the traced signal are shown.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

with [-inst]

If successful, returns all full hierarchical instance names between the two specified signals.

without [-inst]

If successful, returns all full hierarchical signal names between the two specified signals as result.

Example

```
schTrace2Signals -win $_nSchema2 -stopOnFF off -from
"system.i_cpu.i_ALUB.bus_mode\[2\]" -to
"system.i_cpu.i_ALUB.n774" -inst
```

Returns system.i_cpu.i_ALUB.U243, system.i_cpu.i_ALUB.U244, system.i_cpu.i_ALUB.U264, system.i_cpu.i_ALUB.U265, system.i_cpu.i_ALUB.U296, system.i_cpu.i_ALUB.U297, system.i_cpu.i_ALUB.U308, system.i_cpu.i_ALUB.U311 and {system.i_cpu.i_CCU.\bus_mode_reg[2] }

```
schTrace2Signals -win $_nSchema2 -delim "." -stopOnFF off
-passFFDataPort on -passFFClockPort on -passFFControlPort off
-passFFSetResetPort off -passFFOtherPort off -stopOnTri on
-createWindow -from "CPU.clock" -to "CPU.i_PCU.n543"
```

schTraceSignalStatus

Description

Provide the crossing hierarchy/local status for the signal.

Syntax

```
schTraceSignalStatus -signal signalName
```

Argument

`-signal signalName`

Specify the signal name.

Value Returned

1 if the signal passes through the hierarchy. 0 if the signal does not pass through the hierarchy.

Example

```
schTraceSignalStatus -signal system.reset
```

schTraceSLPath

Description

Find the shortest/longest path from specified source to destination.

Syntax

```
schTraceSLPath [-delim delim] [-win window] [-stopOnTri on|off]
[-longestPath | shortestPath] [-2point | fanIn | fanOut] [-pathNum
pathNum] -from instancePort -to instancePort [-passCom
instancePortList] [-byPassCom instancePortList] [-defFile
fileName]
```

Arguments

`-delim delim`

Delimiter of the instance name.

`-stopOnTri on|off`

Specify whether to traverse through Tristate.

`-pathNum pathNum`

Specify the number of paths to be reported.

`-passCom instancePortList`

Specify the instance port list that the path must pass through.

`-byPassCom instancePortList`

Specify the instance port list that the path must not pass through.

`-defFile fileName`

Specify the file name that contains the pass/bypass components or path definition.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schTraceSLPath -r -from regA D -to regD Q -shortesPath -pathNum
20 -2point
-passCom "system.i_cpu.i_ALUB.i_alu.U149.Z" -byPassCom
"system.i_cpu.i_ALUB.i_alu.U321.Z"
```

srcBrowseCellCreate

Description

Invoke the *Browse Cell Summary* form.

Syntax

```
srcBrowseCellCreate
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcBrowseCellCreate
```

Trace by Level

schTraceDriverByLevel

Description

Find the driver(s) of the specified signal or instance by specified level #. If there is no specified signal or instance, the action is applied to the selected signal or instance.

Syntax

```
schTraceDriverByLevel [-delim delim] [-win window] [-r] [-range
start end] [-passAllFF on|off] [-level level][-signal signalName|
-inst instName]
```

Arguments

`-delim` *`delim`*

Delimiter of the name list.

`-passAllFF on|off`

Whether pass through all FF.

`-level` *`level`*

Specify # of the level driver to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected is an instance, or `-inst` option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal name.

`-inst` *`instName`*

Specify the lowest hierarchy instance name.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance name for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of window.

Example

```
schTraceDriverByLevel -passAllFF on  -range 1 2 -signal
{system.CH[4:0]}
schTraceDriverByLevel -level 2 -r -signal {CH[4:0]}
```

schTraceLoadByLevel

Description

Find the load(s) of the specified signal or instance by specified level. If there is no specified signal or instance, the action is applied to the selected signal or instance.

Syntax

```
schTraceLoadByLevel [-delim delim] [-win window] [-r] [-range
start end] [-passAllFF on|off] [-level level][-signal signalName|
-inst instName]
```

Arguments

`-delim` *`delim`*

Delimiter of the name list.

`-passAllFF on|off`

Whether pass through all FF.

`-level` *`level`*

Specify # of level load to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected is an instance, or `-inst` option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal name.

`-inst` *`instName`*

Specify the lowest hierarchy instance name.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance name for the first traced result if successful; otherwise, returns 0.

If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0. This command is only applied to these two types of window.

Example

```
schTraceLoadByLevel -passAllFF on  -range 1 2 -signal
{system.CH[4:0]}
schTraceLoadByLevel -level 2 -r -signal {CH[4:0]}
```

schTraceConnectivityByLevel

Description

Find the connectivity(s) of the specified signal or instance by specified level. If there is no specified signal or instance, the action is applied to the selected signal or instance.

Syntax

```
schTraceConnectivityByLevel [-delim delim] [-win window] [-r]
[-range start end] [-passAllFF on|off] [-level level][-signal
signalName|-inst instName]
```

Arguments

`-delim` *`delim`*

Delimiter of the name list.

`-passAllFF on|off`

Whether pass through all FF.

`-level` *`level`*

Specify the number of level driver to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected is an instance, or `-inst` option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal name.

`-inst` *`instName`*

Specify the lowest hierarchy instance name.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance name for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of window.

Example

```
schTraceConnectivityByLevel -passAllFF on  -range 1 2 -signal
{system.CH[4:0]}
schTraceConnectivityByLevel -level 2 -r -signal {CH[4:0]}
```

schTraceFanInByLevel

Description

Find the fan-in cone of the specified signal or instance by specifying the level through the data path.

NOTE: When no signal or instance is specified, the action is applied to the selected signal or instance.

Syntax

```
schTraceFanInByLevel [-delim delim] [-win window] [-r]
[-range start end] [-passAllFF on|off] [-level level]
[-signal signalName |-inst instName]
```

Argument

`-delim delim`

Specify the delimiter of the name list.

`-inst instName`

Specify the lowest hierarchy instance name.

`-level level`

Specify the number of level driver to be traced.

`-passAllFF on|off`

When this option is turned *on*, pass through all flip-flops. When this option is turned *off*, stop at all flip-flops. The default is *on*.

`-r`

Generate a full scope name if the name of the instance or signal does not have the scope name.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal.

NOTE: When an instance is selected, or the `-inst` option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal name.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

- When the invoking window is a *full hierarchy* window: a list of scope and instance name for the first traced result if successful; otherwise, returns 0.
- When the invoking window is a *flattened window*: 1 if successful; otherwise, returns 0.

NOTE: This command is only applied to these two types of window.

Example

```
schTraceFanInByLevel -passAllFF on -range 1 2 -signal
{system.CH[4:0]}
schTraceFanInByLevel -level 3 -r -signal {CH[4:0]}
```

schTraceFanOutByLevel

Description

Find the fanout cone of the specified signal or instance. If there is no specified signal or instance, the action is applied to the selected signal or instance.

Syntax

```
schTraceFanOutByLevel [-delim delim] [-win window] [-r] [-range
start end] [-passAllFF on|off] [-level level][-signal signalName|
-inst instName]
```

Arguments

`-delim` *`delim`*

Delimiter of the name list.

`-passAllFF on|off`

Whether pass through all FF.

`-level` *`level`*

Specify the number of level driver to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If the selected is an instance, or `-inst` option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal name.

`-inst` *`instName`*

Specify the lowest hierarchy instance name.

Value Returned

If the invoking window is *full hierarchy* window: a list of scope and instance name for the first traced result if successful; otherwise, returns 0.
If the invoking window is *flattened window*: 1 if successful; otherwise, returns 0.
This command is only applied to these two types of window.

Example

```
schTraceFanOutByLevel -passAllFF on  -range 1 2 -signal
{system.CH[4:0]}
schTraceFanOutByLevel -level 2 -r -signal {CH[4:0]}
```

Analysis Result

schShowAnaReport

Description

Open the *Analysis Report* form.

Syntax

```
schShowAnaReport [-file filename] [-findfirst string]
[-findnext string] [-sort columnnum] [-order rownum]
[-srcorder rownum] [-suppress] [-showOn text| -showOn schema]
[-close]
```

Arguments

`-close`

Close the *Analysis Report* form.

`-file filename`

Specify the file to be loaded from the HAL (HDL Analysis and Linting) database.

`-findfirst string`

Find the first matching string for the Rule name of Message Grid and main detail message.

`-findnext string`

Find the next matching string for the Rule name of Message Grid and main detail message.

`-showOn schema`

Show the result in the *nSchema* window.

`-showOn text`

Show the result in the file viewer.

`-sort columnnum`

Specify the column number to sort the result. The column number starts from 0.

`-srcorder rownum`

Select the check point list.

`-suppress`

Suppress the selected message.

`-order` *`rownum`*

Select the main message.

Value Returned

1 if successful; 0 otherwise.

Examples

```
schShowAnaReport -file "/qa/home/chsu/hal041.xml"
schShowAnaReport -findfirst "str"
schShowAnaReport -findnext "str"
schShowAnaReport -sort "0"
schShowAnaReport -order 127
schShowAnaReport -srcorder 0
schShowAnaReport -suppress
schShowAnaReport -showOn schema
schShowAnaReport -close
```

schShowAnaRuleEdit

Description

Open the *Rule Edit* form.

Syntax

```
schShowAnaRuleEdit [-open] [-edit] [-rule rule] [-kind kind]
[-type type] [-file file] [-line line] [-module module]
[-signal signal] [-close]
```

Argument

`-close`

Close the *Rule Edit* form.

`-edit`

Edit the *Rule Edit* form.

`-file` *`file`*

Specify the source file.

`-kind` *`kind`*

Specify the category.

`-line line`
Specify the line number.

`-module module`
Specify the module name.

`-open`
Open the *Rule Edit* form.

`-rule rule`
Specify the name of the error rule.

`-signal signal`
Specify the signal name.

`-type type`
Specify the severity.

Value Returned

1 if successful; 0 otherwise.

Examples

```
schShowAnaRuleEdit -open
schShowAnaRuleEdit -edit -rule "CLKSNM" -kind "halcheck"
-type "Warning" -file "./test_rtl041.vhd" -line "297" -module
-signal "clk_1"
schShowAnaRuleEdit -close
```

schShowAnaSuppressList

Description

Open the *Suppress List* form.

Syntax

```
schShowAnaSuppressList [-open] [-del] [-delall]
[-saveas -file filename] [-load -file filename] [-apply]
[-close]
```

Arguments

`-apply`
Apply suppress list to all messages and do not close the *Suppress List* form.

`-close`

Close the *Suppress List* form.

`-del`

Delete the selected suppress rule.

`-delall`

Delete all suppress rules.

`-load -file` *`filename`*

Load another suppress list file.

`-open`

Open the *Suppress List* form.

`-saveas -file` *`filename`*

Specify the text file including the file path to save the *Suppress List* contents to.

Value Returned

1 if successful; 0 otherwise.

Examples

```
schShowAnaSuppressList -open
schShowAnaSuppressList -saveas -file "/qa/home/chsu/testcase/
demo/Verilog/Gate/testcases_052604/test041/test111"
schShowAnaSuppressList -load -file "/qa/home/chsu/testcase/demo/
Verilog/Gate/testcases_052604/test041/test111"
```

FSM

schExtendFSM

Description

Extend the hierarchical FSM one stage. This command is only active after invoking **schCreateWindow -hierFSM -mode first** or the window created from the **schExtHierFSM** command.

Syntax

```
schExtendFSM [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schExtendFSM -win $_nSchema1
```

schExtHierFSM

Description

Extract the hierarchical FSM from the current active scope on the current active window. The result is shown on a new created *nSchema* window.

Syntax

```
schExtHierFSM [-win window] -mode first|all
```

Arguments

`-mode first|all`

Specify the extraction mode from **first** or **all**.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

nSchema window ID if successful; otherwise, returns 0.

Example

```
schExtHierFSM -win $_nSchema1 -mode first
```

schGetFSMCount

Description

Count the number of FSM blocks in the specified scope.

Syntax

```
schGetFSMCount [-delim delimiter] [-scope scopeFullName]
```

Arguments

`-delim delimiter`

Specify the delimiter of the scope names.

`-scope scopeFullName`

Specify the scope with full hierarchical name.

Value Returned

The number of FSM blocks.

Example

```
schGetFSMCount -delim "." -scope "system"
```

schIsFSMBlock

Description

Check whether the selected instance is an FSM block.

Syntax

```
schIsFSMBlock [-win window] InstName
```

Arguments

`InstName`

Select the instance.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schIsFSMBlock -win $_nSchema1 MASTER
```

schViewFSM

Description

Push into an FSM block and invoke *nState*.

Syntax

```
schViewFSM [-win window] [-delim delim] FSMInstName
```

Arguments

`-delim delim`

Delimiter of the name list.

`FSMInstName`

FSM instance name.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schViewFSM
```

Scope

schColorInstByScope

Description

Specify colors for the instances in different scopes.

Syntax

```
schColorInstByScope -win window -scope scopeName -color colorId
```

Arguments

`-color colorId`

Specify the color for the instances.

`-scope scopeName`

Specify the scope.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schColorInstByScope -win $_nSchema3 -scope
"system.i_cpu.i_ALUB" \
          "system.i_cpu.i_ALUB.i_alu"
"system.i_cpu.i_ALUB.i_alu.add_23" \
          "system.i_cpu.i_ALUB.i_alu.add_23_1" \
          "system.i_cpu.i_ALUB.i_alu.add_24" \
          "system.i_cpu.i_ALUB.i_alu.add_25" \
          "system.i_cpu.i_ALUB.i_alu.sub_24" \
          "system.i_cpu.i_ALUB.i_alu.sub_24_1" \
          "system.i_cpu.i_ALUB.i_alu.sub_25" \
          "system.i_cpu.i_ALUB.i_alu.sub_25_1"
"system.i_cpu.i_CCU" \
          "system.i_cpu.i_PCU" -color "ID_GRAY3" "ID_RED3"
"ID_ORANGE3" \
          "ID_YELLOW3" "ID_GREEN3" "ID_CYAN3" "ID_BLUE2"
"ID_PURPLE2" \
          "ID_GRAY6" "ID_RED6" "ID_ORANGE6" "ID_YELLOW6"
```

schGetScope

Description

Query the current scope of the *nSchema* window.

Syntax

```
schGetScope [-win window]
```

Arguments

`-win window`

The window ID of the invoking source code window.

Value Returned

The current scope name of the specified *nSchema* window.

Example

```
set cscope [schGetScope -win $nSchema]
$cscope is "top.i_cpu"
```

schPopViewUp

Description

Open the schematic view to the upper hierarchical view.

Syntax

```
schPopViewUp [-win window] [-port portName]
```

Arguments

`-port portName`

If specified, after popping view up, the connected signal of the port is selected.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPopViewUp -port clock
```

schPushViewIn

Description

Push the schematic view into the specified instance.

Syntax

```
schPushViewIn [-win window] [-port portName] [instanceName]
```

Arguments

`-port portName`

Specify the port name of the master of an instance if specified. After pushing view in, the connected signal of the port is selected.

`instanceName`

If specify, push down to the specified instance hierarchy. Otherwise, push down the currently selected instance.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

2 *create_windowId*s if there is any new window created. 1 if successful; otherwise, returns 0.

Example

```
schPushViewIn -port {clock} {i_cpu}
```

schSaveCurrentScope

Description

Save the search results of the current scope to a file.

Syntax

```
schSaveCurrentScope -win window [-filter string]
-signal|-inst|-instport|-port|-module -file filename
```

Arguments

`-file filename`

Specify the file with the full path to save the search results.

`-filter string`

Specify the search string. The wildcard (*) is accepted. It is optional.

`-signal|-inst |-instport |-port|-module`

Specify the search type from one of the following:

- `-signal:` If specified, all signals are displayed in the *Find* form for searching.
- `-inst:` If specified, all instances are displayed in the *Find* form for searching.
- `-instport:` If specified, all instance ports are displayed in the *Find* form for searching.
- `-port:` If specified, all ports are displayed in the *Find* form for searching.
- `-module`: If specified, all modules are displayed in the *Find* form for searching.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSaveCurrentScope -win $_nSchema2 -filter "*input" -signal
-file ./save.txt
```

schSetScope

Description

Switch the current view by specified scope, instance, signal, or instance pin.

Syntax

```
schSetScope [-win window] [-delim delim] -scope scopeName
schSetScope [-win window] [-delim delim] -signal signalName
schSetScope [-win window] [-delim delim] -inst instName
schSetScope [-win window] [-delim delim] -instpin instanceName
pinName
```

Arguments

`-delim delim`

Delimiter of the hierarchical name.

`-scope scopeName`

The switched scope.

`-signal signalName`

Switch to the scope of the specified signal and select the signal.

`-inst instName`

Switch to the scope of the specified instance and select the instance.

`-instpin instanceName pinName`

Switch to the scope of the specified instance pin and select the instance pin.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetScope -delim {/} -scope {system/I_cpu/I_CCU}
schSetScope -delim {/} -instpin {system/I_cpu/I_CCU} {clock}
```

schSyncScopeWnd

Description

Synchronize the active scopes.

Syntax

```
schSyncScopeWnd –win window –bSyncWnd on|off
```

Arguments

`–bSyncWnd on|off`

When *on* is specified, the active scopes are synchronized. When *off* is specified, the active scopes are not synchronized. The default is *on*.

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSyncScopeWnd -win $_nSchema2 -bSyncWnd on
```

Radix

schAddAliasFile

Description

Add alias from file.

Syntax

```
schAddAliasFile [-win window] filename
```

Arguments

`filename`

Specify the alias file name.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schAddAliasFile -win $_nSchema1 cpu.alias
```

schAddAliasProgram

Description

Add alias from an executable program.

Syntax

```
schAddAliasProgram [-win window] programName
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

`programName`
Specify the program name.

`-win window`
Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schAddAliasProgram adder
```

schRemoveAlias

Description

Remove alias.

Syntax

```
schRemoveAlias [-win window]
```

Arguments

`-win window`
Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRemoveAlias
```

schSetSignalRadix

Description

Set radix for displaying signal values.

Syntax

```
schSetSignalRadix [-win window] [notation] [-format format]
```

Arguments

`notation`

Specify signal value notation. Notation options are **Signed**, **Unsigned**, **2Com**, **1Com**, and **Mag**.

`-format format`

Specify format of signal value. Format options are **Bin**, **Oct**, **Hex**, **Dec**, and **ASCII**.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetSignalRadix -format Hex
```

Select

schAddSelectedToWave

Description

Add all the selected signals to the synchronized *nWave* window.

Syntax

```
schAddSelectedToWave [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schAddSelectedToWave
```

schBusContentionByTimeRange

Description

Set the bus contention by the specified time range.

Syntax

```
schBusContentionByTimeRange BusName -delim delimiter -beginTime
time -endTime time
```

Arguments

`-beginTime` *`time`*

Specify the begin time.

`BusName`

Specify the bus.

`-delim` *`delimiter`*

Specify the delimiter of the full hierarchical signal name.

`-endTime` *`time`*

Specify the end time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schBusContentionByTimeRange "/system/i_cpu/VMA" -delim "/"
-beginTime "0"  -endTime "12500
```

schDeselectAll

Description

Deselect all the selected signals in the specified window.

Syntax

```
schDeselectAll [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schDeselectAll -win $_nSchema1
```

schGetComponentName

Description

Get the component name by the full-hierarchy instance name.

Syntax

```
schGetComponentName [-instance <full-hierarchy instance name>]
```

Value Returned

1 if successful, 0 if there are any errors in the component name.

Example

```
schGetComponentName -instance "system.i_CPU.i_ALUB.u255"
//will return AND2
```

schGetConnectedInst

Description

Get the name of the instance that is connected to the selected signal.

Syntax

```
schGetConnectedInst -win window [-signal signalName]
```

Arguments

`-signal` *`signalName`*

Select the signal.

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

1 if successful, 0 if there are any errors in the component name.

Example

```
schGetConnectedInst -win $_nSchema3 -signal
"system.CHILD2.ThreeOnly"
```

schGetInstName

Description

Returns the instance names that match the specified naming pattern in the specified scope.

Syntax

```
schGetInstName -instMatch matchName [-scope fullScopeName]
[-detailRTL on|off]
```

Arguments

`-instMatch matchName`

Return instance name from **instMatch**.

The *matchName* can be [Init| SigTap | SigOp | Always | FSM | Process| SigTapSigOp | AlwaysFSM | *]: [SeqNumber|*]:[BeginLineNumber|*]:[EndLineNumber|*]:[InstanceType|*]

`schGetInstName -instMatch SigTapSigOp:x:x:x:x` is the equivalent of `schGetInstName -instMatch SigTap:x:x:x:x` and `schGetInstName -instMatch SigOp:x:x:x:x.`

`schGetInstName -instMatch AlwaysFSM:x:x:x:x` is the equivalent of `schGetInstName -instMatch Always:x:x:x:x` add `schGetInstName -instMatch FSM:x:x:x:x`

NOTE: The wildcard character "*" can be used in any fields, but it cannot be used in all fields simultaneously.

`-scope fullScopeName`

The full hierarchical path to the scope name.

`-detailRTL on|off`

Turn the display of detailed RTL functional block *on* or *off*.

Value Returned

Instance names if successful; otherwise, returns 0.

Examples

Example 1

```
schGetInstName -instMatch "*:*:96:96:*" -scope
system.i_cpu.i_ALUB
```

It returns *1 {ALUB:Always4#SigOp3:96:96:Mux}*.

Where 1 is the value count. If you do not input `-scope` option**,** it uses the scope of the active window.

Example 2

```
schGetInstName -instMatch sigOp:0:120:*:*
schGetInstName -instMatch sigOp:0:*:*:*
schGetInstName -instMatch Always:*:*:1:*
schGetInstName -instMatch AlwaysFSM:*:*:*:*
schGetInstName -instMatch *:5:*:*:*

schGetInstName -win $_nSchema2 -instMatch FSM:0:*:*:*
//ps:SeqNumber=0
```

Returns *1 {mctrl(rtl):FSM0:254:1148:FSM}*

Example 3

```
schGetInstName -instMatch "*:*:*:96:*"  -scope
system.i_cpu.i_ALUB                       //end line 96
schGetInstName -instMatch "*:*:97:*:*" -scope system.i_cpu.i_ALUB
-detailRTL on
schGetInstName -instMatch "*:*:96:*:*" -scope system.i_cpu.i_ALUB
-detailRTL off //begin line 96
```

Example 4

```
schGetInstName -instMatch "*:*:*:*:FSM"
  Returns 3 {system.MASTER.fsm_master(@1):FSM0:31:105:FSM}
            {system.CHILD1.fsm_child1(@1):FSM0:30:69:FSM}
            {system.CHILD2.fsm_child2(@1):FSM0:34:101:FSM}
```

schGetInstPort

Description

It returns full instance port name which connect with a net.

Syntax

`schGetInstPort -signal` *`fullSignalName`* `[-inst` *`fullInstName`*`]`

Arguments

`-signal` *`fullSignalName`*

Signal name.

`-inst` *`fullInstName`*

The instance name.

Value Returned

Full instance port names if successful; otherwise, returns 0.

Example

```
schGetInstPort -signal "system.i_cpu.i_ALUB.CF" -inst
"system.i_cpu.i_ALUB.ALUB:Always14#Always6:123:127:Reg"
```

Returns *1 {system.i_cpu.i_ALUB.ALUB:Always14#Always6:123:127:Reg.ROH_CF}*.

Where 1 is the count value. If you do not input `-inst` option**,** it returns all connected instance ports.

schGetSelectCount

Description

Query back the number of objects in the selected set.

Syntax

```
schGetSelectCount [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

The number of selected objects if successful; otherwise, returns 0.

Example

```
set nselect [schGetSelectCount]
```

schGetSelectSet

Description

Query back all objects in select set.

Syntax

```
schGetSelectSet [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

The number of selected delim list of INST|SIGNAL/INSTPORT/PORT name if successful; otherwise, returns 0.

Example

```
set selset [schGetSelectSet]
$selset is 1 .  {SIGNAL {clk}}
```

schPreSelect

Description

Highlight the triggering event.

Syntax

```
schPreSelect
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPreSelect
```

schSelect

Description

Select the instances or signals in a window.

Syntax

```
schSelect [-win window] [-toggle] -inst listInstname [-delim
delim]
schSelect [-win window] [-toggle] -signal listSigname
schSelect [-win window] [-toggle] -instport listInstportname
schSelect [-win window] [-toggle] -instpin listInstpinname
schSelect [-win window] [-toggle] -port listPortname
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the specified instance.

`-inst` *`listInstname`*

Specify the hierarchical instance name list.

`-instport` *`listInstportname`*

Specify the local instance and port name list.

`-instpin` *`listInstpinname`*

Specify the local instance and pin name list.

`-port` *`listPortname`*

Specify the local port name list.

`-signal` *`listSigname`*

Specify the local signal name list.

`-toggle`

Toggle the selected. Deselect if it is in the selected set. Otherwise add into the select set.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

Number of objects selected.

Examples

```
schSelect -win $_nSchema1 -toggle -signal {S1} {reset} {clock}
schSelect -inst "system.i_cpu"
schSelect -win $_nSchema2 -inst "system/i_cpu" -delim /
schSelect -instport {Inst1} {In} {Inst2} {In2}
```

schSelectAll

Description

Select all objects in *nSchema* window.

Syntax

```
schSelectAll [-win window] [-signal|-inst|-pin]
```

Arguments

`-signal|-inst|-pin`

If specified, it selects all specified type object. Otherwise, select all objects in *nSchema* window.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

Number of objects selected if successful; otherwise, returns 0.

Example

```
schSelectAll -win $_nSchema1
```

schSlackSelect

Description

Select the signals overlapped with the specified signal.

Syntax

`schSlackSelect -win` *`window`* `-signal` *`signalName`*

Arguments

`-signal` *`signalName`*

Specify the signal.

`-win` *`window`*

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSlackSelect -win $_nSchema2 -signal "reset_fsm"
```

searchBusAuto

Description

Add an new attribute to select a bus name without range of bus.

Syntax

```
schSelect [-searchBusAuto Busname]
```

Arguments

`-searchBusAuto`

1.`-searchBusAuto` command can be used just with `-signal` option when **schSelect** Tcl command is used.

2. To select {IR[1:0]} or "IR\[1:0\]" without `-searchBusAuto` and `IR` with `-searchBusAuto` are the same.

3. Besides bus, select other signals with `-searchBusAuto` does not affect original behavior.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
original no "-searchBusAuto" attribute>
    schSelect -signal "IR\[1:0\]"       ----> select
successfully
    schSelect -signal "IR"                ----> fault

<use "-searchBusAuto" attribute>
    schSelect -searchBusAuto -signal "IR" ----> select
successfully
```

View

schAddAnnotation

Description

Add simple text and symbol annotations in *nSchema*. The symbol annotation can be shown when *ON* is specified for **schSymbolAnnotDisplay**.

Syntax

```
schAddAnnotation [-win window] [-color colorName]
[-shape shapeName] [-text textString]{[-pin fullPinName]|
[-port fullPortName]|[-inst fullInstanceName]|
[-signal fullNetName]} [-ver|-hor] [-sameNet] [-delim delim]
[-pos out] [–fulltext|-text textString]
```

For shape file:

```
schAddAnnotation -win window -text "UserText" -shapeFile
fullPathFileName [-inst|-port|-signal|-pin] fullObjectName
[-ver|-hor] [-sameNet]
```

Arguments

`-color colorName`

Specify the color for the symbol.

`-delim delim`

Specify the delimiter for the annotation.

`–fulltext textString`

Display the full text annotation ignoring the limit of the text length.

`-hor`

Display the annotation horizontally.

NOTE: This option works only for the signals specified with the `-signal` option.

`-pos out`

Display the symbol or text annotation at the instance pin for the port or at the upper left for the instance.

`-inst fullInstanceName`

Specify the instance to be annotated with full name.

`-pin` *`fullPinName`*

Specify the pin to be annotated with full name.

`-port` *`fullPortName`*

Specify the port to be annotated with full name.

`-sameNet`

Annotate the same net aliases.

`-shape` *`shapeName`*

Specify the shape to mark the symbol with. The shapes include: **square**, **rectangle**, **circle**, **diamond**, and **star**.

`-shapeFile` *`fullPathFileName`*

Specify the shape file with full path file name.

NOTE: Only the ***.xpm*** file is supported for the shape file.

`-signal` *`fullNetName`*

Specify the signal to be annotated with full net name.

`-text` *`textString`*

Display the text annotation.

`-ver`

Display the annotation vertically.

NOTE: This option works only for the signals specified with the `-signal` option.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
schAddAnnotation -inst system.i_pram -color ID_GREEN2 -text
"instant)" -shape circle

schAddAnnotation -text "port" -shape square -color ID_YELLOW7
-port "system.i_pram.R_W"

schAddAnnotation -text "net" -shape star -color ID_RED6 -signal
```

```
"system.data\[7:0\]"

schAddAnnotation -win $_nSchema2 -text "12345net" -shape star -
color ID_RED6 -signal "system.data\[7:0\]" -ver

schAddAnnotation -win $_nSchema2 -text "12345net" -shape
rectangle -color ID_ORANGE7 -signal "system.addr\[7:0\]" -hor

schAddAnnotation -win $_nSchema2 -text "\(i am inst\)" -shapeFile
"/debussy/home/jaw_lee/SOL2/deb5.3/regression/unittest/nSchema/
Misc/Tcl/clocktree.xpm" -inst "system.i_pram"

schAddAnnotation -win $_nSchema2 -text "12345net" -shape
rectangle -color ID_ORANGE7 -signal "system.addr\[7:0\]" -sameNet

schAddAnnotation -win $_nSchema2 -text "12345net" -shape star \
-color ID_RED6 -signal "system/data\[7:0\]" -delim /
```

NOTE: 1. `-ver` and `-hor` options work on `-signal` option only.
2. If you do not set `-ver` or `-hor` versions, *nSchema* shows annotation dynamically.

schAddViewMark

Description

Add a view mark.

Syntax

```
schAddViewMark [-win window] -keynote keynoteString
```

Arguments

`-keynote` *`keynoteString`*

Specify the keynote string.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

View mark index if successful; otherwise, returns 0.

Example

```
set view_index [schAddViewMark -keynote "system"]
```

schAdjustFontSize

Description

Adjust the font size automatically.

Syntax

```
schAdjustFontSize [-win window]
```

Arguments

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schAdjustFontSize -win $_nSchema3
```

schCellDelay

Description

Open a form to display the cell delay of the selected instance.

Syntax

```
schCellDelay [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCellDelay
```

schDeleteAnnotation

Description

Remove annotation data from a symbol (pin/port/instance/net).

Syntax

```
schDeleteAnnotation {[-pin fullPinName]|[-port fullPortName]|[-
inst fullInstanceName]|[-signal fullNetName]}
```

Arguments

`-pin fullPinName`

Remove annotation from pin *full pinName*.

`-inst fullInstanceName`

Remove annotation from instance *fullInstanceName*.

`-port fullPortName`

Remove annotation from port *fullPortName*.

`-signal fullNetName`

Remove annotation from net *fullNetName*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schDeleteAnnotation -inst system.i_pram
```

schDelViewMark

Description

Delete view mark.

Syntax

```
schDelViewMark [-win window] -all|-keynote keynoteString|index
```

Arguments

`-all`

If specified, delete all existing view marks.

`-keynote keynoteString`

Specify the keynote of the deleted view mark.

`index`

Specify the view mark index which is returned by **schAddViewMark**.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schDelViewMark $view_index
schDelViewMark -keynote "system"
```

schDisplaySource

Description

Open source code viewer to display the corresponding source code section of the selected instance.

Syntax

```
schDisplaySource [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schDisplaySource
```

schFit

Description

Fit the current design or the selected design in the window.

Syntax

```
schFit [-win window][-selected]
```

Arguments

`-selected`

Fit to selected.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schFit
schFit -selected
```

schFocusConnection

Description

Select the signal's connection.

Syntax

```
schFocusConnection [-win window] [signalName]
```

Arguments

`signalName`

If specified, use the specified *signalName*. Otherwise, use the selected signal.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

Number of the selected connections if successful; otherwise, returns 0.

Example

```
set nSelect [schFocusConnection clock]
```

schLastView

Description

Switch to the last view area.

Syntax

```
schLastView [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schLastView
```

schModifyPopDownTime

Description

Specify the number of seconds for the tip display time.

Syntax

```
schModifyPopDownTime [-win window][-time ms]
```

Arguments

`-time ms`

Specify the tip display time in millisecond.

`-win window`
The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schModifyPopDownTime -time 3000
```

schPageUp

Description

Switch to the last page of the current window.

Syntax

```
schPageUp [-win window]
```

Arguments

`-win window`
The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPageUp –win $_nSchema3
```

schPageDown

Description

Switch to the next page of the current window.

Syntax

```
schPageDown [-win window]
```

Arguments

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPageDown –win $_nSchema4
```

schPageSize

Description

Set the page size of the current window.

Syntax

```
schPageSize [-win window]
```

Arguments

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPageSize –win $_nSchema4
```

schPan

Description

Pan move the schematic (left, right, up, or down) based on the start and end point provided.

Syntax

```
schPan -win windowid -startPos startX startY -endPos endX endY
```

Arguments

`-startPos startX startY:`

The coordinate value that LMB drags starts in the *nSchematic* window.

`-endPos endX endY:`

The coordinate value that LMB drags ends in the *nSchematic* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPan -win $_nSchema3 -startPos -1000 2000 -endPos 3500 45000
```

schPanDown

Description

Pan 1/2 down of the current view area to the current window.

Syntax

`schPanDown [-win` *`window`*`]`

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPanDown
```

schPanLeft

Description

Pan 1/2 left of the current view area to the current window.

Syntax

```
schPanLeft [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPanLeft
```

schPanRight

Description

Pan 1/2 right of the current view area to the current window.

Syntax

```
schPanRight [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPanRight
```

schPanUp

Description

Pan 1/2 up of the current view area to the current window.

Syntax

```
schPanUp [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPanUp
```

schPruneLogic

Description

Prune logic from an instance with a control signal but without a clock signal (for example, mux).

Syntax

```
schPruneLogic -mode gray|reduce
```

Arguments

`-win`

This option must be specified. If this option is not specified, the action is executed in the current active window.

`-mode mode`

Specify the mode for prune logic. There are two modes: **gray** or **reduce**. **gray** grays out the associated logic but leaves it in the view. The **reduce** mode removes the logic from the view completely. The default is **reduce**.

Value returned

1 if successful, 0 otherwise

Example

```
schPruneLogic -mode gray
```

schRedraw

Description

Redraw the *nSchema* window.

Syntax

```
schRedraw -win window
```

Arguments

`-win window`

The ID of the *nSchema* window to redraw.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRedraw -win $_nSchema1
```

schSetMsgLine

Description

Show the string on the bottom bar of the schematic window.

Syntax

```
schSetMsgLine [-txt string] [-on|-off]
```

Arguments

`-txt string`

Add new string for left bottom label of the *nSchema*.

`-on`

Other functions cannot modify the contents of the left bottom label of *nSchema*, except the **schSetMsgLine** command.

`-off`

Other functions can modify the contents of the left bottom label of *nSchema*, including **schSetMsgLine** (default).

Value Returned

1 and the current cursor time if successful; otherwise, returns 0.

Example

```
schSetMsgLine -txt "test" -on
schSetMsgLine -off
```

schSetOptions

Description

Switch to view values of the specified object.

Syntax

```
schSetOptions -win window [-showPropagatedValue on|off]
[-showConstantNet on|off] [-modulePortName on|off] [-annotPower
on|off] [-annotSigPower on|off] [-SDCAnnot on|off] [-magnifier
on|off] [-pan on|off]
```

Arguments

`-annotPower on|off`

Specify whether to highlight or unhighlight the power domain color for instances and nets.

`-annotSigPower on|off`

Specify whether to highlight or unhighlight the power aware color for the Retention, Isolation, Level-shifted, or Boundary port signal types.

`-magnifier on|off`

Specify whether to invoke or close the magnifying glass.

`-modulePortName on|off`

Specify whether to view or hide module port names for hierarchical connectors in flattened schematic windows.

`-pan on|off]`

Specify whether to enable or disable the pan move icon.

`-SDCAnnot on|off`

Specify whether to display the SDC annotation.

`-showConstantNet on|off`

Specify whether to highlight or unhighlight the constant net.

`-showPropagatedValue on|off`

Specify whether to view or hide the propagated values.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetOptions -win $_nSchema3 -showPropagatedValue on
schSetOptions -win $_nSchema4 -showConstantNet off
schSetOptions -win $_nSchema4 -annotPower off
schSetOptions -win $_nSchema2 -magnifier on
schSetOptions -win $_nSchema3 -pan on
```

schSwitchToViewMark

Description

Switch to specified view mark.

Syntax

`schSwitchViewMark [-win` *`window`*`] -keynote` *`keynoteString`*`|index`

Arguments

`index`

View mark index returned by *schAddViewMark*.

`-keynote` *`keynoteString`*

Specify the keynote of the switched view mark.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSwitchToViewMark $view_index
schSwitchToViewMark -keynote "system"
```

schSymbolAnnotDisplay

Description

Turn the display of symbol annotation *on* or *off* for active and new *nSchema* window.

Syntax

```
schSymbolAnnotDisplay -win window [on|off]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

`-on`

Display symbol annotation.

`-off`

Hide symbol annotation.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSymbolAnnotDisplay -win $_nSchema2 -off
schSymbolAnnotDisplay -win $_nSchema2 -on
```

schZoom

Description

Zoom the view area of the invoking *nSchema* window.

Syntax

```
schZoom [-win window] point1_X point1_Y point2_X point2_Y
```

Arguments

`point1_X point1_Y point2_X point2_Y`

Specify the coordinates of the *nSchema* window.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schZoom -win $_nSchema2 4 30 804 500
```

schZoomIn

Description

Zoom the view area in by 1/2 of the invoking *nSchema* window.

Syntax

```
schZoomIn [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schZoomIn
```

schZoomOut

Description

Zoom the view area out by 1/2 of the invoking *nSchema* window.

Syntax

```
schZoomOut [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schZoomOut
```

schMinimap

Description

Invoke the minimap window for the active *nSchema* window.

Syntax

```
schSetOptions [-win window] [-minimap on|off] [-close] [-dock
on|off] [-pos] [-dragbounding left|right|top|bottom]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

`-minimap`

Set the minimap status, specify *on* to open the minimap and specify *off* to close the minimap.

`-close`

Close the minimap window.

`-dock: on(dock the window to $_nSchemaNum)/off(undock the window from $_nSchemaNum`

Specify *on* to dock the minimap window. Specify *off* to undock the minimap window and display it as a stand-alone window.

`-pos`

Reset the position of the bounding box in the minimap window.

`-dragbounding`

Resize the scale of the bounding box in the minimap window by dragging the specified edge to the specified position.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetOptions -win $_nSchema2 -minimap on
schSetOptions -win $_nSchema2 -minimap off
schMinimap -win $_nSchema2 -close
schMinimap -win $_nSchema2 -dock on
schMinimap -win $_nSchema2 -dock off
schMinimap -win $_nSchema2 - pos 900 800
schMinimap -win $_nSchema2 -dragbounding left -pos 900 800
```

Search

schFindHierFormCreate

Description

Open the *Find Signal/Instance/InstPort* form in the *nSchema* window.

Syntax

```
schFindHierFormCreate [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

Nothing if successful; otherwise, returns 0.

Example

```
schFindHierFormCreate -win $_nSchema2
```

schSetSearchMode

Description

Set the search mode for subsequent **schSearchPrev** or **schSearchNext** commands.

Syntax

```
schSetSearchMode -negedge | -posedge | -anyChange
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetSearchMode -posedge
```

schSetCursorTime

Description

Set the cursor time.

Syntax

```
schSetCursorTime time
```

Value Returned

The current cursor time if successful, -1 otherwise.

Example

```
schSetCursorTime 100
```

schSearchPrev

Description

Search, and if found, jump to selected signal's previous value change that satisfies the current search mode.

Syntax

```
schSearchPrev [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 and the current cursor time if successful; otherwise, returns 0.

Example

```
schSearchPrev -win $_nSchema1
```

schSearchNext

Description

Search, and if found, jump to selected signal's next value change that satisfies the current search mode.

Syntax

```
schSearchNext [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 and the current cursor time if successful; otherwise, returns 0.

Example

```
schSearchNext -win $_nSchema1
```

Edit

schAddSigWithConn

Description

Add a signal with a connector list.

Syntax

```
schAddSigWithConn -win window -signal signalName
-connector listConnname
```

Argument

`-connector listConnname`

Specify the connector list.

`-signal signalName`

Specify the signal to be added.

`-win window`

The window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schAddSigWithConn -win $_nSchema5  -signal "system.reset_fsm" -
conn "system.CHILD2.fsm_child2\(@1\):FSM0:34:101:FSM"
```

schAddViewObj

Description

Add objects to flattened schematic window.

Syntax

```
schAddViewObj [-win window] (-inst listInstname|-signal
listSignalname|-port listPortName|-portOnly leafOfTopBusPortName
|-instpin listInstPin) [-driver|-load]
```

```
schAddViewObj [-win window] -path {instancePin1}
{instancePin2} ...
```

Arguments

`-driver`

Expand signal drivers. Only valid when `-signal` option is specified.

`-load`

Expand signal loads. Only valid when `-signal` option is specified.

NOTE: The arguments `-driver` and `-load` do not expand load direction if the specified signal already added with driver direction. Vice versa.

`-inst listInstname`

Specify the local instance name list.

`-instpin listInstPin`

Specify the local instance and pin name list.

`-path {instancePin1} {instancePin2}...`

Specify all instance pins, including the hierarchy boundary ports, to create the path.

`-port listPortName`

Specify the local port name list.

`-portOnly leafOfTopBusPortName`

This is supported for adding only ports and nets which are connected to the ports in *nSchema* flattened window. The difference from `-port` option is that connected instances are not added.

nSchema adds this single bit leaf of top-level bus port into the specified flattened window with the net connects to it.

NOTE: Limitation:
1. *nSchema* does not merge the full range bus port if you add the full range bus port/signal by adding viewing object or trace result in this flattened window after the single bit port added. But they are connected to each other.
2. If the single bit port has not connected to any instance, it is deleted after removing viewing object functionality applied in this flattened window. But it is not deleted if it has other instance connected. That is, you need to do trace actions after adding this single bit port into the flattened window.
3. **Hide Extraneous Bus** does not have an effect when there is only one or more single bit ports in the flattened window without other instances.

`-signal listSignalName`

Specify the local signal name list.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
schAddViewObj -win $_nSchema1 -inst {alu} {cpu}
schAddViewObj -instpin {alu} {reset} {cpu} {clock}
schAddViewObj -signal {a} -driver
schAddViewObj -win $_nSchema3 -portOnly {CPU.addr[2]}
schAddViewObj -win $_nSchema9 -path
{top.x0.xi_toplevel4.\xiatd<7> .xi1.xmmn.D} \
{top.x0.xi_toplevel4.\xiatd<7> .xi1.y} \
{top.x0.xi_toplevel4.\xiatd<7> .xi3.a}
```

schCreatePort

Description

This command creates a port after you enter the module name, the port name, and the port direction.

Syntax

```
schCreatePort [-win window] -module "moduleName" -port "portName"
-direction input|output|inout
```

Argument

`-win window`

Specify the window ID of the invoking *nECO* window.

`-module "moduleName"`

Specify the module name.

`-port "portName"`

Specify the port name.

`-direction input|output|inout`

Specify the direction from either **input**, **output**, or **inout**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCreatePort -win $_nSchema3 -module “ALUB” -port “newp1” \
-direction input
```

schEditBundle

Description

Edit the bundle signal.

Syntax

```
schEditBundle [-create BundleName SignalList] |
[-rename OldBundleName NewBundleName] |
[-edit BundleName SignalList] |
[-delete BundleName] | [-autoBundle]
```

Argument

`-autoBundle`

Combine the signals with similar property into a bundle net automatically.

`-create` *`BundleName SignalList`*

Create the bundle signal.

`-delete` *`BundleName`*

Delete the bundle signal.

`-edit` *`BundleName SignalList`*

Edit the bundle signal.

`-rename` *`OldBundleName NewBundleName`*

Rename the bundle signal.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schEditBundle -win $_nSchema3 –create "/system/i_cpu/i_ALUB/
Bundle_net" "/system/i_cpu/i_ALUB/IR\[1:0\]"
```

schExpand

Description

Expand the forward/backward connection from the instance pin or signal on the partial hierarchy window.

Syntax

```
schExpand [-win window] -forward|-backward [-signal signalName| -instpin instanceName portName|-port portName]
```

Arguments

`-backward`

Expand the backward connection of the specified signal or connected signal of the instance pin.

`-forward`

Expand the forward connection of the specified signal or connected signal of the instance pin.

`-instpin instanceName portName`

Specify the instance port.

`-port portName`

Specify the port name. If none is specified, use the selected signal, instpin, or port. If not, do nothing.

`-signal signalName`

Specify the signal name.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schExpand -forward -port {clock}
```

schRedo

Description

Reverse execution to next state.

Syntax

```
schRedo [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRedo
```

schRemoveViewObj

Description

Remove viewing objects. If it is not specified, it is applied to the selected.

Syntax

```
schRemoveViewObj [-win window] [{-inst listInstname|-signal
listSignalname|-port listPortName|-instpin listInstPin}]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRemoveViewObj -win $_nSchema1 -inst {alu} {cpu} -signal
{clock}
```

schSetViewObjOption

Description

Set the insert mode of viewing objects.

Syntax

```
schSetViewObjOption [-win window] -mode mode
```

Arguments

`-mode mode`

Specify the mode from **append** or **replace**.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetViewObjOption -win $_nSchema1
```

schUndo

Description

Reverse execution to the previous state.

Syntax

```
schUndo [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schUndo
```

Editable Schematics

schAddComment

Description

Add text comment. The **Viewing Properties** command in the right-click option menu allows setting changes of text color, text font, text size and text alignment.

Syntax

```
schAddComment -win window -cmtType square -size X1 X2 X3 X4
-textColor Color -fillColor Color -lineColor Color
-lineStyle Style -alignment left|center|right -verticalAlignment
top|middle
```

Argument

`-alignment left|center|right`

Specify whether to align the text in the center, left, or right.

`-textColor Color`

Specify the text color.

`-verticalAlignment top|middle`

Specify whether to align the text on top or in the middle.

`-win window`

Specify the window ID of the invoking *nSchema* window.

NOTE: Some command options are logged in Tcl, but are invisible to the *Editable Schematic Window*. The following options have a default value and do not allow any changes:
`-cmtType` specifies the shape of the comment box. The default is set as square.
`-fillColor` specifies the background color of the comment box. The default is black.
`-lineColor` specifies the line color. The default is blue.
`-lineStyle` specifies the line style. The default is solid.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schAddComment -win $_nSchema9 -cmtType square -size 76 19670 2160
24756 -textColor ID_BLUE3 -fillColor ID_BLACK -lineColor ID_BLUE3
-lineStyle line_solid -alignment left -verticalAlignment top
```

schAddViewObj

Description

Drop specified objects to the current window.

Syntax

```
schAddViewObj -win window [-connectionMode exceptCommonInput|
currentNet] [-inst|-signal ObjectList]
```

Argument

`-connectionMode exceptCommonInput|currentNet`

If the dropped objects contain instance(s) which do not exist in the current schematic, a connection model must be specified between **Only Connect to Current Net(s)** (specified by **exceptCommonInput**) and **Fully Connect to Current Instance(s)** (specified by **currentNet**).

`-inst|-signal` *`ObjectList`*

Specify whether the added object is an instance or signal. *ObjectList* corresponds to the list of instances (specified by `-inst` option) or signals (specified by `-signal` option).

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schAddViewObj -win $_nSchema3 -connectionMode currentNet -inst
{i_cpu}
```

schCapture

Description

Capture the current schematic view and save it in the PNG file.

Syntax

```
schCapture -win window [-footer FooterString] -file FileName
-region N1 N2 N3 N4
```

Argument

`-footer FooterString`

Specify the footer string to add.

`-file FileName`

Save the view to the specified file name.

`-region N1 N2 N3 N4`

Specify the captured region as bounded by N1, N2, N3, and N4.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCapture -win $_nSchema3 -footer "123456" -file "z.png" -region
124 109 239 210
```

schFlip

Description

Flip the selected instances horizontally or vertically.

Syntax

```
schFlip -win window -horizontal|-vertical
```

Argument

`-horizontal|-vertical`

Flip the selected instances horizontally or vertically.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schFlip -win $_nSchema3 -horizontal
schFlip -win $_nSchema3 -vertical
```

schMoveObj

Description

Move the selected object to a new position.

Syntax

`schMoveObj -win` *`window`* `-offset` *`offsetX offsetY`*

Argument

`-offset offsetX offsetY`

The selected object is moved from the current location to an offset as specified by offsetX and offsetY.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schMoveObj -win $_nSchema3 -offset 1233 -796
```

schRearrangeSch

Description

Automatically rearrange the instances in the *nSchema* window to provide a better layout.

Syntax

```
schRearrangeSch -win window
```

Argument

`-win` *window*

Specify the window ID of the *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRearrangeSch -win $_nSchema3
```

schRemoveViewObj

Description

Delete the selected object(s) from the current window.

Syntax

```
schRemoveViewObj -win window
```

Argument

`-win` *window*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRemoveViewObj -win $_nSchema3
```

schRotate

Description

Rotate the selected instances to the left or right.

Syntax

```
schRotate -win window -left|-right
```

Argument

`-left|-right`

Rotate the selected instances to the left or right 90 degrees.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRotate -win $_nSchema3 -left
schRotate -win $_nSchema3 -right
```

schSetAVOption

Description

Set the abstract view options.

Syntax

```
schRotate -win window -left|-right
```

Argument

`-left|-right`

Rotate the selected instances to the left or right 90 degrees.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schRotate -win $_nSchema3 -left
schRotate -win $_nSchema3 -right
```

Clock

schCollectBlackBox

Description

Group the selected flip-flop(s) as a black box on the clock tree window.

Syntax

```
schCollectBlackBox [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCollectBlackBox -win $_nSchema1 -inst {alu} {cpu}
```

schSetAutoMergeOption

Description

Specify the auto merge settings.

Syntax

```
schSetAutoMergeOption [-AutoMerge][-win window -AutoMerge
on|off][-add|-delete -MergeCellName cellName]
[-add|-delete -ExcludeInstName instName -win window] [-save|load
filename -win window] [-reset]
```

Arguments

`-AutoMerge`

Enable the auto merge option and apply the merge rule to all windows.

`-win window -AutoMerge on|off`

Turn the auto merge option of the specified window *on* or *off*.

`-add|-delete -MergeCellName` *`cellName`*

Add or delete a symbol library cell in the merged cell list.

`-add|-delete -ExcludeInstName` *`instName`* `-win` *`window`*

Add or delete an instance in the excluded instance list in the specified window.

`-save|load` *`filename`* `-win` *`window`*

Save or load the rule of auto merge settings in the specified window into or from a file.

`-reset`

Reset all merged setting rule list (including the cell list and the instance list) and the excluded instances list.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetAutoMergeOption -add -ExcludeInstName top.cpu.and2 -win
$_nSchema2

schSetAutoMergeOption -add - MergeCellName INV

schSetAutoMergeOption -load ./MergeRule.txt -win $_nSchema2

schSetAutoMergeOption -save ./MergeRule.txt

schSetAutoMergeOption -delete -ExcludeInstName top.cpu.and2

schSetAutoMergeOption -delete -MergeCellName INV2

schSetAutoMergeOption -reset
```

schShowBBInfo

Description

Show the flip-flops list in the selected black box instance.

Syntax

```
schShowBBInfo [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schShowBBInfo -win $_nSchema1
```

Right-Click Commands

schCopyFullPathToClipborad

Description

This command copies the full hierarchical name(s) of the selected signal(s) into the clipboard buffer.

Syntax

```
schCopyFullPathToClipborad -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCopyFullPathToClipborad -win $_nSchema2
```

schShowFocusConnection

Description

Traces **Focus Connection** immediately of the selected net, port, or instance port and displays the result in a form.

Syntax

```
schShowFocusConnection [-win window] [-hide]
```

Arguments

`-hide`

Closes the *Show Focus Connection* form.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schShowFocusConnection -win $_nSchema2
```

schDisplayDetailRTL

Description

You can select a RTL instance in a non Detail RTL *nSchema* window. When invoking this command, Verdi creates a new *nSchema* window. It shows all detail RTL netlist to this new window.

After executing **schDisplayDetailRTL**, expand a RTL block into detail RTL extraction on a different *nSchema* window. You can call **schGetCurrentWindow** to get the window ID.

Syntax

```
schDisplayDetailRTL [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

No return value.

Example

```
schDisplayDetailRTL -win $_nSchema2
```

schDisplayLiberty

Description

Display the liberty file for the selected cell in *nSchema*. When more than one cell is selected, only one cell is displayed.

Syntax

```
schDisplayLiberty -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schDisplaLiberty -win $_nSchema2
```

Configuration

schCellDelayOptions

Description

Dump the cell delay information.

Syntax

```
schCellDelayOptions [-win window] -dump on|off
```

Arguments

`-dump on|off`

Turn dumping cell delay information *on* or *off*.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCellDelayOptions -dump on
```

schChangeDisplayAttr

Description

Change the display attributes for the specified signal(s) or instances(s). If there is nothing specified, apply to selected.

Syntax

```
schChangeDisplayAttr [-win window] {-delim delim | -color color|-
r |-signal listSignalname |-inst listInstname|-default|-defaultAll
|-style lineStyle|-instpin InstPinList|-local|-noSameNet} [-
applyToPin]
```

Arguments

`-applyToPin`

Set all pins connected to the signal to inherit the color of the net.

`-color` *`color`*

Specify the color.

`-default`

When this option is turned *on*, reset the user-defined display attribute to the default setting.

`-defaultAll`

When this option is turned *on*, reset all the signal and instance to the default display attribute.

`-delim` *`delim`*

Delimiter of the specified *signalInstName*.

`-inst` *`listInstname`*

Specify a list of instance names including the I/O ports and instance/gate ports.

`-instpin InstPinList`

Change the color of individual ports.

`-local`

Specify only the local color for the object. If both the global and local colors are set for the same object, the priority of local is higher than global.

`-noSameNet`

Annotate to the different net aliases.

`-r`

The *signalInstName* is related to the current active scope.

`-signal` *`listSignalName`*

Specify a list of signal names. If both `-inst` and `-signal` options are not specified, apply to the selected signal(s) and instance(s).

`-style` *`lineStyle`*

Specify the line style. The line style options are **line_solid**, **short_dashed**, **dot_dashed**, **dash_dot**, **dash_dot_dot**, **short_dashed_l**, **dot_dashed_l**, **dash_dot_l**, **dash_dot_dot_l**, and **long_dashed**.

NOTE: If `-color` option is not set, the new color of signal/instance selected are used.

`-win window`

Specify the window ID of the invoking *nSchema* window.

`x-local`

Specify only the local color for the object. If both global and local colors are set for the same object, the priority of local is higher than global.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schChangeDisplayAttr -win $_nSchema1 -delim . -color yellow4 \
-inst {system.i_cpu.i_ALUB} {system.i_cpu}
schChangeDisplayAttr -win $_nSchema1 -color red2 -r -signal \
clock {bus[0:3]}
schChangeDisplayAttr -win $_nSchema1 -default
schChangeDisplayAttr -default
schChangeDisplayAttr -win $_nSchema2 -color ID_YELLOW5 \-instpin
“system.i_cpu.i_ALUB.U297.A”
schChangeDisplayAttr -win $_nSchema2 -color ID_BLUE5 -signal \
"system/clock" -delim /
schChangeDisplayAttr -win $_nSchema3 -color ID_CYAN5 -inst \
"system.i_cpu" -local
schChangeDisplayAttr -default -win $_nSchema3 -inst \
"system.i_cpu" -local
schChangeDisplayAttr -defaultAll -win $_nSchema3 -local
schChangeDisplayAttr -color ID_CYAN5 -signal  \
"system.i_cpu.i_ALUB.clock" -noSameNet
```

schGetDisplayAttr

Description

Obtain the default display attributes of the current window.

Syntax

```
schGetDisplayAttr -win window -default
```

Arguments

`-default`

Return to the default display attributes.

`-win window`

The window ID of the invoking source code window.

Value Returned

The default display attributes.

Example

```
schGetDisplayAttr -win $_nSchema2 -default
```

schGroupInstMgr

Description

Groups all buffers and inverters together on the flattened schematic views.

Syntax

```
schGroupInstMgr -win window [-autoCreate on|off]
```

Arguments

`-autoCreate on|off`

Specify to group the buffers and inverters of the clock trees.

`-create $GroupInstNameList`

Group the instances.

`-delete $GroupInstName`

Delete the grouped instances.

`-info $GroupInstName`

Display the names of the grouped instances.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if success, 0 otherwise

Example

```
schGroupInstMgr -win $_nSchema3 -autoCreate on

schGroupInstMgr -win $_nSchema4 -autoCreate off

schGroupInstMgr -win $_nSchema3 -info GroupInst2
```

```
schGroupInstMgr -win $_nSchema3 -create GroupInst3 -inst
gmcs.rdck__L2_I0 \
gmcs.rdck__L3_I0

schGroupInstMgr -win $_nSchema3 -create GroupInst3 -inst
gmcs.rdck__L2_I0 \
gmcs.rdck__L3_I0
```

schNetlistcomOpt

Description

Invoke the *Netlistcom Information* form.

Syntax

```
schNetlistcomOpt
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schNetlistcomOpt
```

schResetDisplayAttr

Description

Reset all the display attributes to the default setting.

Syntax

```
schResetDisplayAttr
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schResetDisplayAttr
```

schResetPRConstraint

Description

Reset the P&R constraint set.

Syntax

```
schResetPRConstraint
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schResetPRConstraint
```

schSDCOpen

Description

Import the SDC File.

Syntax

```
schSDCOpen
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSDCOpen
```

schSetDisplayAttr

Description

Set the display attributes for the type.

Syntax

```
schSetDisplayAttr typeName [[-color color]|[-lineStyle lineStyle]]
```

Arguments

`-color color`

Specify the color.

`-lineStyle lineStyle`

Specify the line style.

`typeName`

Specify the type name. The type names include **Background**, **BlockSymbol**, **RTLSymbol**, **SymbolPort**, **SymbolName**, **ClockSignal**, **VDDSignal**, **VSSSignal**, **NormalSignal**, **GlobalSignal**, **SelectedSet**, **TracedSet**, **BackAnnotate**, **Value0**, **Value1**, **ValueX**, **ValueZ**, **DimColor**, **Preselect**, **TipBackground**, **RubberBanding**, **PureViewMemory**, **FSMBlock**, **FFBlackBox**, **LatchBlackBox**, **MacroBlackBox**, **ValuePropagationConstantNet0**, **ValuePropagationConstantNet1**, **AutoMergeGroupNet**, **FirstSelected**, and **ConnectedPin**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetDisplayAttr Background -lineStyle LineSolid -color red4

schSetDisplayAttr ValuePropagationConstantNet0 -style
dash_dot_dot_l -color ID_BLUE4
```

schSetLibSetByFile

Description

Specify the files for symbol library setting.

Syntax

```
schSetLibSetByFile -novasLibs fileName –novasLibPaths fileName
```

Arguments

`–novasLibPaths fileName`

Specify the file containing the paths of all symbol libraries.

`-novasLibs fileName`

Specify the file containing the names of all symbol libraries.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetLibSetByFile -novasLibs “/home/local/liberty.f”
–novasLibPaths “/home/local/libPathFile.f"
```

NOTE: The **liberty.f** should contain a list of library names following below format:
liba
libb

The **libPathFile.f** should contain a list of paths following below format:
/home/mylib
/home/proj/lib

schSetOptions

Description

Set all the viewing option in *nSchema* window.

Syntax

```
schSetOptions [-win window] {-portName on|off | -instName on|off
|-localNetName on|off | -parameterList on|off | -shortName on|off
|-annotate on|off | -annotateInColor on|off | -annotateSDF on|off
|-annotatePathInfo on|off | -leadingZeros on|off | -preselect
on|off | -autoFit on|off | -highContrastMode on|off | -toolbar
on|off | -msgLine on|off | -hideExtraneousBus on|off | -PGPin
on|off | -delayType delayType | -delayPrecision precision |-
delayScale delayScale}
```

Arguments

`-annotate on|off`

Turn the display of active annotation *on* or *off*.

`-annotateInColor on|off`

Turn annotating signals in color *on* or *off*.

`-annotatePathInfo on|off`

Turn controlling the trace longest/shortest result window to show the path information *on* or *off*.

`-annotateSDF on|off`

Turn the display of the SDF annotation *on* or *off*.

`-autoFit on|off`

Turn fitting the selected object automatically into the viewing area *on* or *off*.

`-delayPrecision` *`precision`*

Specify the value of the **Delay Precision** option. The values are 0.1, 0.01, 0.001, and 0.0001.The default is *0.01*.

`-delayScale` *`delayScale`*

Specify the value of delay scale. The values are 10fs, 100fs, 1ps, 10ps, 100ps, 1n, and 10ns.

`-delayType` *`delayType`*

Specify the delay type. The types are **min**, **typical**, and **max**.

`-hideExtraneousBus on|off`

Turn whether to stop showing the signal in the bus *on* or *off*.

`-highContrastMode on|off`

Turn whether to standout the selected object and trace results from the schematic view *on* or *off*.

`-instName on|off`

Turn the display of the instance name *on* or *off*.

`-leadingZeros on|off`

Turn the display of the leading zeros *on* or *off*.

`-localNetName`

Turn the display of the internally connected net name *on* or *off*.

`-msgLine on|off`

Turn the display of the message line *on* or *off*.

`-parameterList on|off`

Turn the display of the parameters attached to the instance *on* or *off*.

`-PGPin on|off`

Turn the display of the power/ground pins *on* or *off*.

`-portName on|off`

Turn the display of the port name *on* or *off*.

`-preselect on|off`

Turn highlighting the preselect object *on* or *off*.

`-shortName on|off`

Turn the display of the local name in the partial flattened window *on* or *off*.

`-toolbar on|off`

Turn *on* or *off* to display the toolbar.

`-win` *`window`*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetOptions -portName on -annotate on
schSetOptions -win $_nSchema2 -annotateSDF off
```

schSetPRConstraint

Description

Set the P&R constraint set.

Syntax

```
schSetPRConstraint -module moduleName -leftmost | -rightmost
instanceList
```

Arguments

`-leftmost` *`instanceList`*

Specify the instance to be placed at the leftmost position.

`-module` *`moduleName`*

Specify the module name that the constraint applied to.

`-rightmost` *`instanceList`*

Specify the instance to be placed at the rightmost position.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetPRConstraint -module ALUB -leftmost {U233} {U344}
```

schSetPreference

Description

Set all the viewing options in the **Preferences** command in the *nSchema* window.

Syntax

```
schSetPreference [-annotate on|off] [-annotateInColor on|off]
[-autoFit on|off] [-AutoMerge on|off] [-bImportPowerInfo on|off]
[-cellNameCase on|off] [-cellNameToCase Lower|Upper]
[-DefaultAutoMergeSetting filename] [-delayScale delayScale]
[-delayPrecision delayPrecision] [-delayType delayType]
[-detailLevel level] [-detailRTL on|off] [-stopOnFSM on|off]
[-detailRTLBlock off|[on [-detailMux on|off]] [-recogFSM on|off]
[-DisplayAllTracePaths on|off] [-preselectPowerTip on|off]
[-expandGenBlock on|off] [-preselectTip on|off] [-portName on|off]
[-fixFontSize off|[-PGPin on|off]
[-highContrastMode on|off] [-instName on|off] [-leadingZeros
on|off] [-localNetName on|off] [-magnifierRatio ratio]
[-magnifierSize size] [-moduleName on|off]
[-modulePortName on|off] [-msgLine on|off] [-NetConnectedtoPort
on/off] [-parameterList on|off] [-partialDis on|off] [-partialHead
headNum] [-partialTail tailNum] [-pinNameCase on|off]
[-pinNameToCase Lower|Upper]
[-preselect on|off] [-preselectionToolbar on|off]
[-schemaTip on|off] [-sdcGLKPassFF on|off]
[-searchPriority ".lib"|".lib++"|".db"] [-selectionToolbar
on|off] [-shortName on|off] [-showConstantNet on|off]
[-showPassThroughNet on|off]
[-showPropagatedValue on|off] [-skipFFCell on|off] [-skipStateCell
on|off] [-stopOnMOSCell on/off] [-turboLibs libName]
[-SmartTraceDriverLoad on|off]
[-SmartTraceStopAtBranch on|off] [-StopOnCasting on|off]
[-FullHierTip on|off] [-stopOnMacroCell on|off] [-toolbar on|off]
[-stopOnModuleBoundary on|off] [-traceGroupByCell on|off]
[-traceTimeout on|off] [-turnOffMouseGesture on|off]
[-turboLibPaths libPath] [-DisplayLengthwiseTransistor on|off]
[-InferenceLibCell on|off] [-TraceWithPolarity on|off]
[-RegNamingRule "Namingrule"] [-showAllHierPorts on|off]
[-recogMem on|off]
```

Arguments

`-annotate on|off`

Turn the display of the active annotation *on* or *off*.

`-annotateInColor on|off`

Turn annotating signals in color *on* or *off*.

`-autoFit on|off`

Turn fitting the selected object automatically into the viewing area *on* or *off*.

`-AutoMerge on|off`

Turn the **Auto Merge Cell Group When Creating New Schematic Window** option in the **Schematics -> nAnalyzer** page of the **Preferences** command *on* or *off*.

`-bImportPowerInfo on|off`

When this option is *on*, the liberty file is reloaded with power symbol information and the design.

`-cellNameCase on|off`

Turn the **All Cell Names to** option *on* or *off*.

`-cellNameToCase Lower|Upper`

Set the symbol library cell name to either lowercase or uppercase. This option must be used with `-cellNameCase` option.

`-DefaultAutoMergeSetting` *`filename`*

Specify the default file name for the merge settings.

`-delayPrecision` *`precision`*

Specify the value of the **Delay Precision** option. The values are 0.1, 0.01, 0.001, and 0.0001.The default is *0.01*.

`-delayScale` *`delayScale`*

Specify the value of delay scale. The values are **10fs**, **100fs**, **1ps**, **10ps**, **100ps**, **1n**, and **10ns**.

`-delayType` *`delayType`*

Specify the delay type. The types are **min**, **typical**, and **max**.

`-detailLevel` *`level`*

Specify the level limit for expression extraction. This option is valid if `-detailRTLBlock` option is *on*.

`-detailMux on|off`

Turn detecting latch error *on* or *off*. This option is valid if `-detailRTLBlock` option is *on*.

`-detailRTL on|off`

Specify to display the module/entity in detail view.

`-detailRTLBlock on|off`

Turn the display of the detailed RTL functional block *on* or *off*.

`-DisplayAllTracePaths on|off`

Turn the **Display All Trace Paths** option in the **Schematics -> Display Options -> Trace** page of the **Preferences** command *on* or *off*.

`-DisplayLengthwiseTransistor on|off`

Turn the **Display Lengthwise Transistor** option in the **Schematics -> Display Options -> Schematic** page of the **Preferences** command to *on* or *off*. When *on* is specified, transistor symbols are displayed vertically. When *off* is specified, transistor symbols are displayed horizontally. The default is *off*.

`-enableLMBDnd on|off`

Turn the **Enable Button 1 to Drag and Drop** option in the **General** page of the **Tools -> Preferences** command to *on* or *off*. When this option is turned *on*, after you select one or more objects in *nSchema* and *nState* window, you can drag left mouse button from the selected object to do drag-and-drop operation. When this option is turned *off*, you can zoom or perform other mouse operations. The default value of this option is *off*.

`-expandGenBlock on|off`

Specify to expand the generate block and display all individual instances within.

`-fixedSizedFont on|off`

Turn the fixed font size option for net/pin/port names *on* or *off*. If *on*, the option `-fixedSizeFont` *fontName* is required to specify the font name of net/pin/port names. The available combinations of font and size are: **Helvetica 8**, **Helvetica 10**, **Helvetica 12**, **Helvetica 14**, **Courier 18**, and **Courier 24**.

NOTE: The option can be used to set the fix size and font. However, the option is removed from the Verdi GUI.

`-highContrastMode on|off`

Turn whether to standout the selected object and trace results from the schematic view *on* or *off*.

`-FullHierTip on|off`

When this option is turned *on*, full hierarchy name of instance is displayed in a tip when specifying the `-preselectTip` option to show tip. When this option is turned *off*, full hierarchy name of instance is not displayed when specifying the ***-preselectTip*** option to show tip. The default is *off*.

`-InferenceLibCell on|off`

When this option is turned *on* and symbol libraries do not exist, then cells defined with '*celldefine* or '*endcelldefine*, or cells imported with the `-v` or `-y` options are recognized as modules. When this option is turned *on* and symbol libraries exist, cells are defined in symbol libraries, behaving the same when this option is turned *off*. The default is *off*.

`-instName on|off`

Turn the display of the instance name *on* or *off*.

`-leadingZeros on|off`

Turn the display of the leading zeros *on* or *off*.

`-localNetName on|off`

Turn the display of the internally connected net name *on* or *off*.

`-magnifierRatio`

Sets the window ratio (X2, X4, X8, or X16) for the Magnifying Glass.

`-magnifierSize`

Sets the window size (small, medium, or large) for the Magnifying Glass.

`-moduleName on|off`

Turn the display of the module name in the *nSchema* window *on* or *off*.

`-modulePortName on|off`

When this option is *on*, the **View -> Hier. Connector Port Name** command of the *nSchema* window is enabled. When this option is *off*, the command is disabled.

`-msgLine on|off`

Turn the display of the message line *on* or *off*.

`-NetConnectedtoPort on/off`

Enable or disable the **Show Net Connected to Port** option.

`-parameterList on|off`

Turn the display of parameters attached to the instance *on* or *off*.

`-partialDis on|off`

Turn the display of partial signal name *on* or *off*.

`-partialHead` *`headNum`*

Specify the number of the head character of signal to be shown. If a signal name is *system.top.data* and *-partialD* is *on*, *-partialHead* is 2 and *-partialTail* is *3*, then the displayed signal name is *sy...ata*.

`-partialTail` *`tailNum`*

Specify the number of tail character of signal to be shown.

`-PGPin on|off`

Turn the display of the power/ground pins *on* or *off*.

`-pinNameCase on|off`

Turn the **All Pin Names to** option *on* or *off*.

`-pinNameToCase Lower|Upper`

Set the symbol library pin name to either lowercase or uppercase. This option must be used with `-pinNameCase` option.

`-portName on|off`

Turn the display of the port name *on* or *off*.

`-preselect on|off`

Turn highlighting the preselect object *on* or *off*.

`-preselectPowerTip on|off`

Turn the display of power information *on* or *off*. The default is *on*.

`-preselectionToolbar on|off`

When this option is *on*, the **Preselection Message** option displays information related to the preselected object on the toolbar when a new *nSchema* window is opened. When this option is *off*, the preselected message information is not displayed on the toolbar.

`-preselectTip on|off`

Turn the tip display *on* or *off*. The default is *off*.

`-recogFSM on|off`

Specify to recognize an FSM for the selected module/entity.

`recogMem on|off`

Enable/disable to display the memory cells in the nSchema window.

`-RegNamingRule "Namingrule"`

Set the naming rule for RTL registers. Refer to the **Apply Naming Rule for Inferred Registers** option in the **Schematics -> RTL** page of the **Preferences** command for details.

`-schemaTip on|off`

Turn the tip display for the netlist of the instance in the *nSchema* window *on* or *off*.

`-sdcGLKPassFF on|off`

Turn the **Trace through Flip-flop between Source Clock and Generated Clock** option in the **Schematics -> nAnalyzer** page of the **Preferences** command *on* or *off*.

`-searchPriority .lib|.lib++|.db`

Set the symbol file search priority. The options include: **.lib**, **.lib**++, and **.db**.

`-selectionToolbar on|off`

When this option is *on*, the **Selection Message** option displays information related to the selected object on the toolbar when a new *nSchema* window is opened. When this option is *off*, the selected message information is not displayed on the toolbar.

`-shortName on|off`

Turn the display of the local name in the partial flattened window *on* or *off*.

`-showAllHierPorts on|off`

When this option is *on*, all the input/output ports on hierarchical boundaries are displayed in the hierarchical flattened schematic window. When this option is *off*, the connected input/output ports are displayed in the hierarchical flattened schematic window. The default is *off*.

`-showConstantNet on|off`

When this option is *on*, the **Schematic -> Constant Net** command of the *nSchema* window is enabled. When this option is *off*, the command is disabled.

`-showPassThroughNet on|off`

When this option is *on*, all the different scopes of passthrough nets are displayed in a flattened window. The default is *off*.

`-showPropagatedValue on|off`

When this option is *on*, the **Schematic -> Propagated Value** command of the *nSchema* window is enabled. When this option is *off*, the command is disabled.

`-skipFFCell on|off`

When this option is *on*, cells which have logic expression in their definition of ff()/latch() in a liberty file is skipped.

`-skipStateCell on|off`

When this option is *on*, cells generated by power state table is skipped.

`-SmartTraceDriverLoad on|off`

When this option is *on*, *nSchema* automatically traces through single input/output cells (for example, buffers) while tracing a signal's driver or load. Tracing stops on the first multi-input/output cells in the path. When this option is *off*, *nSchema* traces to the first driver or load in the path. The default is *off*.

`-SmartTraceStopAtBranch on|off`

This option only works when the **Smart Trace Driver/Load** option is turned *on*. When the option is *on*, *nSchema* stops tracing when there are multiple driver/load instances. When this option is *off*, *nSchema* traces a signal's driver or load directly. The default is *off*.

`-StopOnCasting on|off`

Turn the **Stop on Casting** option in the **Trace** page of the **Preferences** command *on* or *off*.

`-stopOnFSM on|off`

Turn stopping cone trace at the FSM *on* or *off*.

`-stopOnMacroCell on|off`

Turn stopping *nSchema* clock tree trace on clock pin of macro cells *on* or *off*.

`-stopOnModuleBoundary on|off`

Turn stopping cone trace at the module boundary *on* or *off*.

`-stopOnMOSCell on|off`

Turn stopping cone trace at the MOS cell *on* or *off*.

`-toolbar on|off`

Turn the display of the toolbar *on* or *off*.

`-traceGroupByCell on|off`

Turn the display of group cell types in different black boxes of clock tree view *on* or *off*.

`-traceTimeout on|off`

When this option is turned *on*, the timeout range to stop the tracing when the tracing exceeds the range can be provided. By default, the option is *off*.

`-TraceWithPolarity on|off`

When this option is *on*, tracing in *nSchema* only include paths that contain polarity information. When this option is turned *off*, *nSchema* traces without polarity information. The default is *off*.

`-turboLibPaths` *`libPath`*

Specify paths of symbol library.

`-turboLibs` *`libName`*

Specify the name of symbol libraries.

`-turnOffMouseGesture on|off`

Turn the display of the mouse gesture *on* or *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schSetPreference -portName on -annotate on
schSetPreference -showPropagatedValue on
schSetPreference -showConstantNet on
schSetPreference -schemaTip on
schSetPreference -showPassThroughNet on
schSetPreference -searchPriority ".lib++"
schSetPreference -pinNameCase on -pinNameToCase "Upper"
schSetPreference -DisplayLengthwiseTransistor on
schSetPreference -RegNamingRule "_reg"
schSetPreference -NetConnectedtoPort on
schSetPreference -traceTimeout 300
schSetPreference -traceTimeout off
schSetPreference -selectionToolbar off
schSetPreference -preselectionToolbar off
schSetPreference -turnOffMouseGesture on
schSetPreference -stopOnMOSCell on
schSetPreference -recogMem on
```

Print

schPrint

Description

Print the *nSchema* window content.

Syntax

```
schPrint [-win window] [-header header] [-footer footer][-signLoc
location] [-design designName] [-designer designer] [-descript
description] [-date date] [-ver version] [-pageMode pageMode]
[-autoDiv boolean]{[-scaleFac scalingFactor]|[-rowPage row]
[-colPage col]}[-copy numofCopy] [-orient orientation] [-paper
papersize] [-color]{[-printer printerName]|[-file printFile]
[-printRegion printRegion]|[-sameAs on|off] [-thinBus on|off]}
```

Arguments

`-autoDiv` *`boolean`*

Auto page division or not. The default is *off*. When **autoDiv** is true, the `-scaleFac`, `-colPage`, and `-rowPage` setting are ignored.

`-color`

If specified, use the color printing.

`-colPage` *`col`*

Number of column. The default is *1*.

`-copy` *`numofCopy`*

Specify how many copies to print. The default is *1*.

`-date` *`date`*

Show the date. The default is the date of today.

`-descript` *`description`*

Additional description shown on the signature.

`-design` *`designName`*

Show the design name. If not specified, the current active scope of the current window is the default design name.

`-designer` *`designer`*

Show the designer name. The default is the login user name.

`-file` *`printFile`*

When `-file` option is specified, *nSchema* prints the window content to the specified printfile.

`-footer` *`footer`*

Any printable message shown on the footer.

`-header` *`header`*

Any printable message shown on the header.

`-orient` *`orientation`*

Specify the orientation of the paper. Available values include **landscape** and **portrait**. The default is *landscape*.

`-pageMode single|multiple`

Specify the page mode. When the page mode is **single**, all the multiple page-related settings are ignored.

`-paper` *`papersize`*

Specify the size of printer paper. Available paper sizes include **A4**, **A3**, **A2**, **A1**, **B**, **C**, **D**, and **E**. The default is *A4*.

`-printer` *`printerName`*

If `-printer` is specified, *nSchema* sends the window content to the specified printer. The default printer is *lp*.

`-printRegion` *`printRegion`*

Specify the region to print. Available values include **all**, **currentView**, and **fitToPage**.

`-rowPage` *`row`*

Number of row. The default is *1*.

`-sameAs on|off`

Whether color is as the same as the current *nSchema* window.

`-scaleFac` *`scalingFactor`*

The scaling factor (percentage, %) of 10 ~ 1000. The default is *100*.

`-sigLoc` *`location`*

Specify the signature location. Available values include **none**, **auto**, **top**, **bottom**, **left**, **right**, **lowerLeft**, **lowerRight**, **upperLet**, and **upperRight**.

`-thinBus on|off`

Specify whether the printed bus width is the same as net.

`-ver` *`version`*

Specify the version.

`-win window`

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPrint -win $_nSchema1 -signLoc auto -descript "a demo design"
-ver 1.0  -pageMode multiple  -autoDiv true -printer lp
```

schCapture

Description

Capture current window to an image file in the PNG format.

Syntax

```
schCapture [-win window] [-footer footer] [-region region] -file
fileName
```

Arguments

`-file fileName`

Specify file name to save captured image.

`-footer footer`

Any printable message shown on the footer.

`-region region`

Specify the region to be printed or saved.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCapture -footer "CPU" -file "/home/cpu.png"
```

nWave

Window

wvCloseWindow

Description

Close an *nWave* window.

Syntax

```
wvCloseWindow [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCloseWindow -win $_nWave1
```

wvCreateWindow

Description

Create a new *nWave* window.

Syntax

```
wvCreateWindow [-win window]
```

Value Returned

Waveform window ID.

Example

```
set current_window [wvCreateWindow]
```

wvExit

Description

Exit the stand-alone *nWave* opened by executing the **verdi -nWave** command.

Syntax

```
wvExit
```

Value Returned

None.

Example

```
wvExit
```

wvGetAllWindows

Description

Get all *nWave* window IDs.

Syntax

```
wvGetAllWindows
```

Value Returned

The list window IDs of all *nWave* windows.

Example

```
set wave_window_list wvGetAllWindows
```

wvGetCurrentWindow

Description

Get the current active *nWave* window.

Syntax

```
wvGetCurrentWindow [-primary] [-name]
```

Arguments

`-primary`

Get the primary window. If this option is not specified, get the last *nWave* window.

`-name`

Return the window name. If this option is not specified, return to the window ID.

`-active`

Get the active window. If this option is not specified, get the primary window with the `-primary` option, else get the last *nWave* window.

Value Returned

Current *nWave* window ID or window name.

Example

```
set current_window  wvGetCurrentWindow
wvGetCurrentWindow -primary
wvGetCurrentWindow -name
wvGetCurrentWindow -primary -name
wvGetCurrentWindow -active
```

wvGetFileTimeUnit

Description

Get file time unit.

Syntax

```
wvGetFileTimeUnit [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Time unit (in *G*, *K*, *M*, *m*, *u*, *n*, *p*, or *f* seconds, such as nanosecond or picosecond) if successful; otherwise, returns 0.

Example

```
wvGetFileTimeUnit -win $_nWave1 1ns
```

wvGetXWindowId

Description

It takes the ID returned from **wvCreateWindow**, and returns the X window ID.

Syntax

```
wvGetXWindowId [-id $xwid]
```

Arguments

`-id $xwid`

Tcl window variable ID.

Value Returned

X window ID if successful. Set the command result to *0x0* and return *CMDOBJ_ERROR* otherwise.

Example

```
wvGetXWindowId -id $_nWave1
```

wvGetWindowTimeUnit

Description

Get window time unit.

Syntax

```
wvGetWindowTimeUnit [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Time unit (in *G*, *K*, *M*, *m*, *u*, *n*, *p*, or *f* seconds, such as nanosecond or picosecond) if successful; otherwise, returns 0.

Example

```
wvGetWindowTimeUnit -win $_nWave1 2ps
```

wvLowerWindow

Description

Lays the *nWave* window to the bottom of the window stacking order.

Syntax

```
wvLowerWindow [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvLowerWindow -win $_nWave2
```

wvRaiseWindow

Description

Raises the *nWave* window to the top of the window stacking order.

Syntax

```
wvRaiseWindow [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRaiseWindow -win $_nWave2
```

wvRefresh

Description

Re-read waveform data from the FSDB file.

Syntax

```
wvRefresh [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRefresh -win $_nWave1
```

wvResizeWindow

Description

Resize an *nWave* window according to given width height.

Syntax

```
wvResizeWindow [-win window] x y width height
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`width`

Width in screen unit.

`height`

Height in screen unit.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvResizeWindow -win $_nWave1 300 400
```

wvSetCurrentWindow

Description

Set specified waveform window ID as current window.

Syntax

```
wvSetCurrentWindow window
```

Value Returned

Current *nWave* window ID.

Example

```
wvSetCurrentWindow $_nWave1
```

wvSetPrimaryWindow

Description

Set an *nWav*e window as the primary window.

Syntax

```
wvSetPrimaryWindow [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetPrimaryWindow -win $_nWave1
This primary window is used as the synchronized window with other
applications.
```

wvSplitWindow

Description

Split or stop split *nWave* window.

Syntax

```
wvSplitWindow [-win window] option
```

Arguments

`option`

on | off to control split or stop split. The default is on.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSplitWindow -win $_nWave1 on
```

wvSyncAllWaveform

Description

Synchronize the cursor/marker, viewing time range, and vertical scrolling for multiple *nWave* windows.

Syntax

```
wvSyncVerticalScroll [-win window] [-cursor_marker on|off]
[-horizontal_range on|off] [-vertical_scroll on|off]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window. The other window is synchronized with this invoking *nWave* window.

`-cursor_marker`

Turn the **Window -> Sync All Waveforms by -> Cursor/Marker** *on* or *off*.

`-horizontal_range`

Turn the **Window -> Sync All Waveforms by -> Horizontal Range** *on* or *off*.

`-vertical_scroll`

Turn the **Window -> Sync All Waveforms by -> Vertical Scroll** option *on* or *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSyncAllWaveform -win $_nWave2 -cursor on -horizontal_range off
This command is only valid when there are two or more than 2
windows
```

wvSyncWindow

Description

Set *nWave* windows synchronized in cursor time and viewing time range.

Syntax

```
wvSyncWindow [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window. The other window is synchronized with this invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSyncWindow -win $_nWave1
```

This command is only valid when there are two and only two *nWave* windows.

wvTileWindow

Description

Tile *nWave* windows.

Syntax

```
wvTileWindow [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window. And this invoking window is placed on the top site.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvTileWindow -win $_nWave1
This command is only valid when there are two and only two nWave
windows.
```

File

wvCloseFile

Description

Close all or individual waveform file(s) from the current *nWave* window.

Syntax

```
wvCloseFile [-win window] [fileNameList]
```

Arguments

`fileNameList`

Specify the waveform file(s) to be closed in the *Close Files* form.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
wvCloseFile -win $_nWave1
```

Closes all of the waveform files.

```
wvCloseFile -win $_nWave1 {/tmp/file1.fsdb} {/tmp/file2.fsdb}
```

Closes the waveform files */tmp/file1.fsdb* and */tmp/file2.fsdb*.

wvConvertFile

Description

Convert the file in any supported format to FSDB file.

Syntax

```
wvConvertFile [-win window] [-o fsdbFileName] fileName
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-o` *`fsdbFileName`*

Output of FSDB file name. If not specified, *fileName.FSDB* is the default output FSDB file name.

`fileName`

Specified FSDB file name.

Value Returned

FSDB file name if successful; otherwise, returns 0.

Example

```
wvConvertFile verilog.dump
```

wvIsFileOpen

Description

Check whether an FSDB file is opened.

Syntax

```
wvIsFileOpen [-win window] fullFileName
```

Arguments

`fullFileName`

Specify the FSDB file's full name to check its existence.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if opened; otherwise, returns 0.

Example

```
wvIsFileOpen -win $_nWave2 "/dq2/qa/demo/54/verilog/rtl/rtl.fsdb"
```

wvOpenFile

Description

Load an FSDB file into *nWave* window.

Syntax

`wvOpenFile [-win` *`window`*`] [-time` *`time1 time2`*`] [-time_`*`unit timeUnit`*`]`
`[-extractSameNet][-ruleFile` *`ruleFilename`*`]` *`fileName`* `[-mul` *`filename1 filename2`*`]`

Arguments

`-extractSameNet`

If specified, extract the same net cross hierarchy also.

`fileName`

Specify a FSDB file name.

`-mul` *`filename1 filename2`*

Specify multiple FSDB file names.

`-ruleFile` *`ruleFilename`*

If specified, bus creation uses the specified rule in rule file.

`-time` *`time1 time2`*

Specify the time range to load.

`-time_unit` *`timeUnit`*

Specify the time unit in fs, ps, ns, us, ms, or s.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

FSDB file name if successful; otherwise, returns 0.

Example

```
wvOpenFile verilog.fsdb
wvOpenFile -time 10000 2000 verilog.fsdb
wvOpenFile -time 0 100000 -extractSameNet
wvOpenFile -win $_nWave2 -time 3 10 -time_unit 10us \
{/verify1/regression/demo/waveform/rtl/fsdb/v2.0/dump50.fsdb}
```

wvReloadFile

Description

Reload active FSDB files in all *nWave* windows.

Syntax

```
wvReloadFile -win window [-reopen from to]
```

Arguments

`from`

Specify the begin time of *nWave*.

`to`

Specify the end time of *nWave*.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvReloadFile
```

wvSaveVirtualFile

Description

Save virtual file format and virtual file name in a virtual file.

Syntax

```
wvSaveVirtualFile [-win window] -format stitch -file fileList
-vfname virtualFileName
```

Arguments

`-file`

File list of virtual FSDB files.

`-format`

Virtual FSDB file format is automatically selected based on input FSDB files.

`-vfname`

File name of virtual files.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSaveVirtualFile -win $_nWave1 -format stitch -file {/home/
file1.fsdb /home/file2.fsdb} -vfname virtual.vf
```

wvSetActiveFile

Description

Set one of the loaded waveform file as the active one.

Syntax

`wvSetActiveFile [-win` *`window`*`] [-applyAnnotation on|off]` *`fileName`*

Arguments

`-applyAnnotation on|off`

Whether *nTrace* or *nSchema* synchronizes the specified active file in *nWave*.

`fileName`

Specified FSDB file name.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

New active waveform file name if successful; otherwise, returns 0.

Example

```
wvSetActiveFile -win $_nWave1 -applyAnnotation on /da2/steven/
test/verilog.fsdb
```

wvSetFileTimeRange

Description

Set the time range to view for an opened FSDB file.

Syntax

```
wvSetFileTimeRange [-win winId] [-time_unit unit] from_time
to_time
```

Arguments

`from_time`

Specify the begin time for viewing.

`-time_unit unit`

Specify the time unit.

`to_time`

Specify the end time for viewing.

`-win winId`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetFileTimeRange -win $_nWave1 -time_unit 1n 500 1000
```

wvShiftFileTime

Description

Specify the delta to be added to the time in the waveform file.

Syntax

```
wvShiftFileTime [-win window] time
```

Argume

`time`

Specify the time scale.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvShiftFileTime 100
```

wvVirtualFileEditorClose

Description

Close the virtual file.

Syntax

```
wvVirtualFileEditorClose [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvVirtualFileEditorClose -win $_nWave1
```

wvVirtualFileEditorOpen

Description

Open the virtual file.

Syntax

```
wvVirtualFileEditorOpen [-win window] [-vfname virtualFileName]
```

Arguments

`-vfname`

File name of virtual files.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvVirtualFileEditorOpen -win $_nWave1 -vfname virtual.vf
```

Scope

wvGetScope

Description

Get all subscopes by given scope.

Syntax

```
wvGetScope [-win window] [-delim delim] [scopeName]
```

Arguments

`-delim delim`

Delimiter of the specified signal name. The default is '/'.

`scopeName`

Specified scope name. If not specified. The default is top scope.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

List of sub-scope names.

Example

```
wvGetScope
```

wvGetSignalsByScope

Description

Get all signals by given scope.

Syntax

```
wvGetSignalsByScope [-win window] [-delim delim] [scopeName]
```

Arguments

`-delim delim`

Delimiter of the specified signal name. The default is '/'.

`scopeName`

Specified scope name. If not specified. The default is top scope.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

List of signal names in this scope.

Example

```
wvGetSignalsByScope
```

wvSetHierDelimiter

Description

Set hierarchy delimiter.

Syntax

```
wvSetHierDelimiter [-win window] -delim delim
```

Arguments

`-delim` *`delim`*

Specify the hierarchy delimiter string.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetHierDelimiter {/}
```

Select

wvDeselectAll

Description

Deselect all the selected on the specified window.

Syntax

```
wvDeselectAll [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDeselectAll -win $_nWave1
```

wvSelectAll

Description

Select all signals in the *nWave* window.

Syntax

```
wvSelectAll [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectAll -win $_nWave1
```

wvSelectAnalogs

Description

Select analog signals.

Syntax

```
wvSelectAnalogs [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectAnalogs
```

wvSelectGroup

Description

Select group in the *nWave* window.

Syntax

```
wvSelectGroup [-win window] [-toggle] listGroupName
```

Arguments

`listGroupName`

Specify a list of group names to be selected.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectGroup -win $_nWave1  GA GB GZ
```

wvSelectSignal

Description

Select signals in the *nWave* window.

Syntax

```
wvSelectSignal [-win window] [-delim delim] [-toggle]{signalList |
(group index)}
```

Arguments

`-delim delim`

Delimiter of the specified signal name.

`(group index)`

Specify the position where the index is a list of positions related to the group.

`signalList`

A list of the signal names.

`-toggle`

Toggle the selected. Deselect if it is in the selected set. Otherwise, add into the select set.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Number of objects selected.

Example

```
wvSelectSignal -win $_nWave1  -delim . {system.i_CPU.clock} {(GA
1 2 3)} {(GB 2)} {system.iCPU.reset}
```

wvSelectStuckSignals

Description

Select and highlight stuck signals (i.e. signals without value changes from the cursor time to the specified marker time) in the *nWave* signal pane.

Syntax

```
wvSelectStuckSignals [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectStuckSignals -win $_nWave1
```

Signal

wvAddSignal

Description

Add signals to the *nWave* window.

Syntax

```
wvAddSignal [-win window] [-delim delim] [-clear]
{signalName [-color color] | [-shiftSignal signal] [-sdfSignal
localName] [-risingDelay delay timescale -fallingDelay delay
timescale]} | {-group {groupName {signalName [-color color]}}}
[-scope scopeName]
wvAddSignal [-win window] [-delim delim] {-v variableFullName}
wvAddSignal [-win window] [-delim delim] {-pw fullSignalName}
wvAddSignal [-win window] [-delim delim] -scope fullScopeName
wvAddSignal [-win window] [-delim delim] -scope fullScopeName
[-type input|output|inout|net|register|other]
wvAddSignal [-win window] "-mem 2DMemoryArray"
wvAddSignal [-marker] -group {{groupName {signalName [-color
color]}}} [-scope scopeName]
wvAddSignal [-win window] [delim delim] {-ss fullSignalName}
wvAddSignal [-win window] [delim delim] {-sn fullSignalName}
```

Arguments

`-clear`

Clear the *nWave* window before adding signals.

`-color color`

Specify the color for the signal name.

`-delim delim`

Delimiter of the specified signal name. The default is '/'.

`-fallingDelay delay timescale`

Specify the delay and time scale for falling edge.

`-group`

Add the specified list of signals to the specified group name.

`{groupName}`

Add the signals to this specified group name.

NOTE: If the group name is not specified, the signal is added to the current position.

`-marker`

When this option is specified with the **-group** option, signals and the specified group is added with reference to the marker position. When this option is specified without the **-group** option is not specified, a group name is created.

`-mem`

Show the two-dimensional (2D) memory array evaluated (not in simulator-dumped FSDB) by Verdi on *nWave*. This option is only for 2D array.

`-pw` *`fullSignalName`*

Specify the runtime power state signal to be added and evaluated.

`-risingDelay` *`delay timescale`*

Specify the delay and time scale for rising edge.

`-scope` *`fullScopeName`*

Specify the scope where signals are added from.

`-sdfSignal` *`localName`*

Specify the local name for the shifted signal.

`-shiftSignal` *`signal`*

Specify the signal to shift.

`{`*`signalName`*`}`

Specify the signal to add into the current position with the **-color**, **-shiftSignal**, **-sdfSignal**, **-risingDelay**, and **-fallingDelay** attributes, if any are specified.

`-sn` *`fullSignalName`*

Specify the inserted signal as the runtime power supply net signal to be evaluated.

`–ss` *`fullSignalName`*

Specify the power supply set signal to be added and evaluated.

`-type input|output|inout|net|register|other`

Specify the type of signals to be added in the selected scope. The types are **input**, **output**, **inout**, **net**, **register**, and **other**. Multiple types can be specified together with a space as the separator.

NOTE: This option must be used with the **-scope** option.

`-v` *`variableFullName`*

Specify the full variable signal name.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddSignal -group {GA  {test/mem_c/arbiter/MEM_RD}
{test/mem_c/arbiter/MEM}} {GB {/test/A} {/test/B}}

wvAddSignal {/system/I_cpu/I_ALUB/U288/A1} -shiftSignal {A1}
-sdfSignal {U288.A1} -risingDelay 28.0 1p -fallingDelay 2.00 1p

wvAddSignal {/test/C} -color ID_YELLOW8  {/test/D}

wvAddSignal -v {top/v1} -v {top/v2}

wvAddSignal -scope top.i_cpu

wvAddSignal -win $_nWave2 "-mem system.i_pram.macron[2:0]"

wvAddSignal -win $_nWave3 {-pw /$power_root/@{\system.PDsystem}}

wvAddSignal -win $_nWave1 -scope /system -type inout register

wvAddSignal -marker -group {GA {test/mem_c/arbiter/MEM_RD} {test/
mem_c/arbiter/MEM}} {GB {/test/A} {/test/B}}

wvAddSignal -win $_nWave3 {-ss /$power_root/$power_supplyset}

wvAddSignal -win $_nWave2 {-sn /$power_root/$power_net/\$VDD_TOP}
```

wvAddAllSignals

Description

Get all signals from the active FSDB file and add to the *nWave* window.

Syntax

```
wvAddAllSignals [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddAllSignals
```

wvClearAll

Description

Removes all signals and groups from *nWave* and adds group name *G1* automatically in the signal window after executing this command.

Syntax

```
wvClearAll -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvClearAll $_nWave1
```

wvCollapseToParent

Description

Collapses members to it's parent signal.

Syntax

```
wvCollapseToParent [-win window] [{(group index)}]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCollapseToParent -win $_nWave1 { ("G1" 4) }
```

wvCreateOverlapValue

Description

Annotate color on a signal waveform based on a specified time frame.

Syntax

```
wvCreateOverlapValue [-win window] [-delim delimiter] [-color
colorID] signalName {(startTime endTime string)}
```

Arguments

`-color` *`colorID`*

Specify the color of the overlapped area.

`-delim` *`delimiter`*

Delimiter of the specified signal name.

`{(startTime endTime string)}`

Specify the start time, the end time, and the string.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCreateOverlapValue -win $_nWave1 -color BLUE -delim / {/system/
R_W} {1000 1500 _} {1800 2000 text}
```

wvDecSignalHeight

Description

Decreases the height of selected signals.

Syntax

```
wvDecSignalHeight [-win windowId]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDecSignalHeight -win $_nWave1
```

wvDeleteSignal

Description

Delete the specified signals.

Syntax

```
wvDeleteSignal [-win window] signalList | {(group index)}
```

Arguments

`(group index)`

If specified, only apply to the signal at the positions specified. Otherwise, apply to selected signals.

`signalList`

A list of the signal names.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDeleteSignal {(A 1 3)} {(B 2)}
wvDeleteSignal {/system/reset}
```

wvDumpSignalListToFile

Description

Dump all scopes or signals in an FSDB file to a text file.

Syntax

```
wvDumpSignalListToFile -win window [-scope|-signal]
[-delim delim] -file filename
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the specified scope or signal name. The default is '/'.

`-file` *`filename`*

Specify the output file name.

`-scope`

If specified, all scopes in an FSDB file are output to a text file. It is optional.

`-signal`

If specified, all signals in an FSDB file are output to a text file. It is optional. This option is used as the default if no output type is specified.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDumpSignalListToFile -win $_nWave1 -scope -file scope_list.txt
wvDumpSignalListToFile -win $_nWave1 -scope -delim . -file
scope_list.txt
```

```
wvDumpSignalListToFile -win $_nWave1 -signal -file
signal_list.txt
wvDumpSignalListToFile -win $_nWave1 -signal -delim . -file
signal_list.txt
wvDumpSignalListToFile -win $_nWave1 -delim . -file
signal_list.txt
```

wvExtractSelSignals

Description

Extract selected signals to a specified FSDB file.

Syntax

```
wvExtractSelSignals -win window -verilog|-vhdl [-ignoreRuntime]
[-src sourceFsdbFile] [-out outputFsdbFile]
[-bt beginTime[timeUnit]] [-et endTime[timeUnit]]
[-options otherOptions] [-resolution {valueUnit}]
```

Arguments

`-bt beginTime[timeUnit]`

Specify the begin time to be extracted. If not specified, the source FSDB begin time is used.

`-et endTime[timeUnit]`

Specify the end time to be extracted. If not specified, the source FSDB end time is used.

`-ignoreRuntime`

Ignore runtime signals when extracting selected signals. The default is *off*.

`-options otherOptions`

Specify the options which are sent to *fsdbextract* directly.

`-out outputFsdbFile`

Specify the output FSDB file for extracted signal(s).

`-resolution {valueUnit}`

Specify the resolution of an HSpice format file and the unit in *fs*, *ps*, *ns*, *us*, *ms*, or *s*. This option is only for an HSpice format file (or an FSDB file generated from HSpice).

`-src FsdbFile`

Specify the source FSDB file to be extracted.

`-verilog|vhdl`

Verilog or VHDL format.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvExtractSelSignals -win $_nWave1 -src full.fsdb -out
time_100_200.fsdb -bt 100 -et 200 -options "-f conf.file"

wvExtractSelSignals -win $_nWave2 -src \
{/verify1/regression/SPS0192141/ORISE2fsdb.fsdb} -out \
{/verdi/home/license/rtl.vcd.fsdb} -verilog -bt \
0.000000ns -et 160.000000ns -resolution {1n}
```

wvFindSignal

Description

Find signal by specified name.

Syntax

```
wvFindSignal [-win window] [-delim delim] [direction
-forward|-backward][-matchPrefix] signalName
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the specified signal name. The default is '/'.

`direction -forward|-backward`

Specify the search direction from the current marker position. The default is **-forward**.

`-matchPrefix`

Find all signals with the specified string as the prefix.

`-noMatchCase`

When specified, a case-sensitive search is not performed.

`-next`

Specify the next signal in the signal pane.

`-previous`

Specify the previous signal in the signal pane from the current signal selected.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvFindSignal -delim "."  -backward {top.i1.signalA}
wvFindSignal -delim "." -matchPrefix {top.i1.signalB}
```

wvGetSelectedPureSignals

Description

Return the names for all selected signals. Selected groups are not included in the list. Both the file name and signal name may be included.

Syntax

```
wvGetSelectedPureSignals [-win window] [-file_name]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-file_name`

When specified, pre-pend the FSDB file's full name to the signal name returned.

Value Returned

All selected signals' name list, not including selected groups.

Example

If the following signals "/system/clock", "/system/data", and "/system/en" are loaded, but only "/system/clock" and "/system/en" are selected, and the following is specified:

```
wvGetSelectedPureSignals -win $_nWave1
```

The return result is "/system/clock /system/en".

wvIncSignalHeight

Description

Increases the height of selected signals.

Syntax

```
wvIncSignalHeight [-win windowId]
```

Arguments

`-win windowID`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvIncSignalHeight -win $_nWave1
```

wvIsSignalExistInWaveform

Description

Check if a signal or scope exists in the waveform file.

Syntax

```
wvIsSignalExistInWaveform -win window
[[-scope scopePath]|[-signal] signalPath] [-delim delim]
```

Arguments

`-delim delim`

Specify the delimiter of the specified scope or signal name. The default is '/'.

`-scope scopePath`

If specified, check if the scope exists in the waveform file. It is optional.

`-signal signalPath`

If specified, check if the signal exists in the waveform file. It is optional. A signal path must be specified if neither **-scope** or **-signal** is specified. This option is used as the default if no type is specified.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvIsSignalExistInWaveform -win $_nWave1 -scope /top/A
wvIsSignalExistInWaveform -win $_nWave1 -scope top.A -delim .
wvIsSignalExistInWaveform -win $_nWave1 -signal /top/A/B
wvIsSignalExistInWaveform -win $_nWave1 top.A.B -delim .
```

wvRenameSignal

Description

Rename the specified signal to a new name.

Syntax

```
wvRenameSignal [-win window] [-delim delim] [oldName] newName
```

Arguments

`-delim` *`delim`*

Delimiter of the specified signal name. The default is '/'.

`oldName`

The signal to be renamed. If not specified, use the selected signal.

`newName`

New signal name.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRenameSignal {/top/B[2]}
wvRenameSignal {/top/A} {/top/C}
```

wvReportSelSignals

Description

Report selected signals in a specified report (*.frpt*) file.

Syntax

```
wvReportSelSignals -win window [-src FsdbFile] [-out frptFile]
[-bt beginTime[timeUnit]] [-et endTime[timeUnit]] [-options
otherOptions]
```

Arguments

`-bt beginTime[timeUnit]`

Specify the begin time to be extracted. If not specified, the source FSDB begin time is used.

`-et endTime[timeUnit]`

Specify the end time to be extracted. If not specified, the source FSDB end time is used.

`-options otherOptions`

Specify the options which are sent to *fsdbreport* directly.

`-out frptFile`

Specify the output report (*.frpt*) file.

`-src FsdbFile`

Specify the source FSDB file to report.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvReportSelSignals -win $_nWave1 -src a.fsdb -out report.rpt -bt
0ns -et 1000ns
```

wvRestoreSignal

Description

Restore signals from the specified file which was created by the **wvSaveSignal** command.

Syntax

```
wvRestoreSignal [-win window] [-openDumpFile] [-overWriteAutoAlias
on|off] fileName
```

Arguments

`-appendSignals on|off`

If specified as *on*, the signals from the RC file are appended in the *nWave* window; otherwise the signals are deleted and then signals are added from the RC file into the *nWave* window.

`fileName`

Specify the file to be restored.

`-openDumpFile`

If specified, use the saved FSDB file as the active file in stead of the current open FSDB file. Otherwise, use the current open FSDB file.

`-win window`

Specify the window ID of the invoking *nWave* window.

`-overWriteAutoAlias on|off`

If specified as *on*, the signal RC file is restored with the **Overwrite Autoalias with RC** option enabled; otherwise the **Overwrite Autoalias with RC** option is disabled.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRestoreSignal signal_for_debug.sav
wvRestoreSignal -win $_nWave2 "/a/b/c/1.rc" -overWriteAutoAlias
off
```

wvSaveSignal

Description

Save signals shown on *nWave* window and FSDB file name to a file.

Syntax

```
wvSaveSignal [-win window] [-group {optionList}] fileName
```

Arguments

`fileName`

Specify output file name.

`-group`

Save the specified groups only.

`optionList`

It includes the following: [-cursor] [-marker][-aliastable | -aliasreference][-usermarker][-digitalLOsig] [-analogLOsig] [-event] [-createbus] [-dtoasig] [-atodsig] [-renamedsig] [-comment] [-layout]

Default Value:

-cursor -marker -aliastable -usermarker -digitalLOsig -analogLOsig -event -createbus -dtoasig -atodsig -renamedsig -comment -layout

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
wvSaveSignal -group {GA GB GC} group_for_debug.sav
wvSaveSignal signal_for_debug.sav
```

wvSaveSignalRC

Description

Save signals shown on *nWave* window and FSDB file name to the *rc* resource file.

Syntax

```
wvSaveSignalRC [-win window] [-layout] [-cursor] [-marker]
[-aliastable] [-usermarker][-digitalLOsig] [-analogLOsig]
[-event] [-createbus] [-dtoasig] [-atodsig] [-renamedsig]
[-comment] [-scopehier] [-fsdbinfo] [-abspath]
[-no_natural_bus_range] [-save_scope_with_macro]
```

Arguments

`-abspath`
Save a signal file with the full file path related to the current directory.

`-aliastable`
Including the alias table properties of the signals.

`-analogLOsig`
Including the analog logical operations signals properties of the signals.

`-atodsig`
Including the analog to digital signals properties of the signals.

`-comment`
Including the comments properties of the signals.

`-createbus`
Including the created buses properties of the signals.

`-cursor`
Including the cursor properties of the signals.

`-digitalLOsig`
Including the digital logical operations signals properties of the signals.

`-dtoasig`
Including the digital to analog signals properties of the signals.

`-event`
Including the event properties of the signals.

`-fsdbinfo`
Treat FSDB file information as comments.

`-layout`
Including the window layout properties of the signals.

`-marker`
Including the marker properties of the signals.

`-no_natural_bus_range`

Do not save bit range of natural buses.

`-renamedsig`

Including the renamed signals properties of the signals.

`-save_scope_with_macro`

When specified, the scope is replaced with the specified macro (virtual top) in the *signal.rc* file.

`-scopehier`

Including Get Signals form status and scope hierarchal properties in the *Get Signals* form.

`-usermarker`

Including the user-defined marker properties of the signals.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Default Value:

-cursor -marker -aliastable -usermarker -digitalLOsig -analogLOsig -event -createbus -dtoasig -atodsig -renamedsig -comment -scopehier

NOTE: When saving signal RC with this option **-fsdbinfo** *off* and restoring the same file, some warning information may appear such as signals did not exist since signals might exist in other FSDB files, not current active FSDB file.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
wvSaveSignalRC -win $_nWave2 -layout -cursor -marker -aliastable
-usermarker -digitalLOsig -analogLOsig -event -createbus -dtoasig
-atodsig -renamedsig -comment -scopehier -fsdbinfo -transinfo
-abspath -save_scope_with_macro "/top"
```

wvShiftSignalTime

Description

Specify the delta to be added to the time in the selected signal(s). New signals are created to replace the selected signal(s).

Syntax

```
wvShiftSignalTime [-win window] shiftTime
```

Arguments

`shiftTime`

Specify the shift time for all signals.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

List of the new shifted signals if successful; otherwise, returns 0.

Example

```
wvShiftSignalTime -win $_nWave1 50
```

wvSplitInout

Description

Splits the selected EVCD signal into input signals and output signals.

Syntax

```
wvSplitInout -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSplitInout –win $_nWave1
```

wvSignalReport

Description

Display signal event counts (including rising/falling) within specified time range.

Syntax

```
wvSignalReport [-win window] -add signalList
wvSignalReport [-win window] -delete signalList
wvSignalReport [-win window] [-syncDelta on|off] | [-from time]
[-to time]
wvSignalReport [-win window] -tofile fileName
```

Arguments

`-from time`

The start time. If not specified, use cursor time instead. Should be used with **-to time**.

`-syncDelta on|off`

Use cursor/marker as time range. If this option is on, **-from** and **-to** are ignored.

`-to time`

The end time. If not specified, use marker time instead. Should be used with **-from time**.

`-tofile fileName`

Specify output file name to save event information shown on form.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSignalReport -win $_nWave1 -add {system.i_cpu.clk} {system.clk}
{system.i_cpu_I_CCU.clk}
wvSignalReport -win $_nWave1 -delete {system.clk}
wvSignalReport -win $_nWave1 -from 1000 -to 5000
wvSignalReport -win $_nWave1 -tofile {/dar/integ/Report/clk.txt}
```

wvUnknownSaveResult

Description

Save unrecognizable signal list to file.

Syntax

```
wvUnknownSaveResult [-win window] [-clear] -file fileName
```

Arguments

`-clear`

Option to clean up old information.

`-file fileName`

Specify the file name to save this information.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvUnknownSaveResult -win $_nWave1 -clear -file unknow.list
```

Bus

wvAddFullBus

Description

Add and display the entire bus for a bus member or a partial bus.

Syntax

```
wvAddFullBus -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddFullBus -win $_nWave1
```

wvBusAdjust

Description

Adjust the selected bus(es).

Syntax

```
wvBusAdjust -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvBusAdjust -win $_nWave1
```

wvBusInvert

Description

Bus invert for selected signals.

Syntax

```
wvBusInvert [-win window] [{(group index)}]
```

Arguments

(group index)

If specified, only apply to the signal at the position indicated. Otherwise, apply to selected signals.

-win window

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
wvBusInvert {(A 1)} {(B 2 3 4)}
wvBusInvert
```

wvBusReverse

Description

Reverse bus for selected signals.

Syntax

```
wvBusReverse [-win window] [{(group index)}] [-range starBit
endBit]
```

Arguments

(group index)

If specified, only apply to the signal at the position indicated. Otherwise, apply to selected signals.

`-range` *`starBit endBit`*

Specify bit range of these signals to be reversed.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
wvBusReverse {(A 1)} {(B 2 3 4)}
wvBusReverse
```

wvCreateBus

Description

Create a bus.

Syntax

```
wvCreateBus [-win window] [-delim delim] [-high value] [-low value]
busName signalList [-prefix]
```

Arguments

`busName`

Specify the bus name.

`-delim` *`delim`*

Delimiter of the specified signal name. The default is '/'.

`-high` *`value`*

High threshold of analog signal value.

`-low` *`value`*

Low threshold of analog signal value.

`-prefix`

When this option is *on*, the full hierarchy scope name is added as a prefix to each analog signal when creating a new bus.

`signalList`

A list of signal names.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

Bus ID.

Example

```
wvCreateBus -win $_nWave1 -high 1.500000 -low 0.300000 -prefix
true \ "NewBus\[3:0\]" "i263.ibuffers.iobuf4.v\(dpre_ltch\)" \
"i263.ibuffers.iobuf5.v\(dpre_ltch\)" \
"i263.ibuffers.iobuf6.v\(dpre_ltch\)" \
"i263.ibuffers.ionuf7.v\(dpre_ltch\)"
```

wvCreateBundle

Description

Use this command to bundle signal creation.

Syntax

```
wvCreateBundle -win window [-delim delim] -bComposite TRUE|FALSE
bundleName signalList
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-delim` *`delim`*

Specified delimiter for later signal names.

`-bComposite TRUE|FALSE`

The way to create bundle signal.
TRUE: Create a composite signal.
FALSE: Create a bus signal.

`bundleName`

The full name of bundle signal.

`signalList`

The signal full name list of bundle members.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCreateBundle -win $_nWave3 -bComposite FALSE "/system/i_cpu/
i_ALUB/b_T\[4:2\]" "/system/i_cpu/i_ALUB/T4" "/system/i_cpu/
i_ALUB/T3" "/system/i_cpu/i_ALUB/T2"

wvCreateBundle -win $_nWave3 -bComposite TRUE "/system/i_cpu/
i_ALUB/Bundle_net" "/system/i_cpu/i_ALUB/IR\[1:0\]"
```

wvCreateBusOpen

Description

Open the *Create Bus* form.

Syntax

```
wvCreateBusOpen [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCreateBusOpen -win $_nWave1
```

wvExpandBus

Description

Expand the specified bus signal.

Syntax

```
wvExpandBus [-win window] [{(group index)}] [-range left right]
[-sub_bus sub_size] [-display_from_LSB]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`(`*`group index`*`)`

If specified, only apply to the signal at the position specified. Otherwise, apply to selected signals.

`-range` *`left right`*

Specify the bit range.

`-sub_bus` *`sub_size`*

Specify the bit size to form sub-buses.

`-display_from_LSB`

Show sub-bus from LSB.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvExpandBus {(GA 3)} -range 2 3
wvExpandBus -win $_nWave1 {(GA 3)} -range 7 0 -sub_bus 3
-display_from_LSB
```

wvPartialBus

Description

Replace current bus with the partial bit range bus signal.

Syntax

`wvPartialBus [-win` *`window`*`] [{(`*`group index`*`)}] -range` *`left right`*

Arguments

`(`*`group index`*`)`

If specified, only apply to the signal at the position specified. Otherwise, apply to selected signals.

`-range` *`left right`*

Specify the bit range.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

New bus name if successful; otherwise, returns 0.

Example

```
set bus1 [wvPartialBus {(GA 3)} -range 3 0]
($bus1 is alu[3:0])
```

wvSetBusAdjust

Description

Adjust bus settings.

Syntax

```
wvSetBusAdjust [-win window] -prefix prefixString -bits bitNum
[-operation shift|replace|insert|delete] [-value high|low]
[-direction left|right] [-replace]
```

Arguments

`-prefix prefixString`

The prefix for the signals you selected.

`-bits bitNum`

The total bits you want to deal with the selected signals.

`-operation shift|replace|insert|delete`

Select either **Shift**, **Replace**, **Insert**, or **Delete** radio button to operate the selected signals.

`-value [high|low]`

For operations of **Shift**, **Replace**, or **Insert**, you can set the value of the new bits to either **Logic High** or **Logic Low**.

`-direction left|right`

Select either **From Left** or **From Right** radio button for direction of the selected signals.

`-replace`

Replace the selected signals from the signal window with the new bus signals.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetBusAdjust -win $_nWave1 -prefix myBus -bits 2 -operation
shift -value high -direction left -replace
```

Event

wvAddComplexEvent

Description

Add complex event to event window.

Syntax

```
wvAddComplexEvent [-win window] eventName eventDefintion
```

Arguments

`eventName`

Name of the event to be added.

`eventDefintion`

Event definition string.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddComplexEvent -win $_nWave1 {complext_event1} \
{
    COMPLEX-EVENT  event1  {
      LEVEL 0  {
          timer1: START  timer2: STOP
           IF   ( event0 OCCURS  2)  THEN terminate
      }
      LEVEL 1  {
            timer1: STOP  timer2: STOP
            IF   ( counter1 >  1)  THEN trigger
      }
    }
}
```

wvAddEvent

Description

Add event to event window.

Syntax

```
wvAddEvent [-win window] eventName eventString
```

Arguments

`eventName`

Specify the name of the event to be added.

`eventString`

Specify the event declaration string.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddEvent -win $_nWave1 {event0} {"/system/clock" === 'b0}
```

wvAppendVerilogExpression

Description

Load a previously saved logical operation file.

Syntax

```
wvAppendVerilogExpression [-win window] -file filename
```

Arguments

`-file filename`

Specify the full path name of the saved logical operation file.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAppendVerilogExpression -win $_nWave2 -file ./2.exp
```

wvCaptureEvent

Description

Capture a specific event into the waveform window.

Syntax

```
wvCaptureEvent [-win window] eventName [-begin begin] [-end end]
[-oneshot] -captureTimerCounter
```

Arguments

`-begin begin`

Specify the time range to capture.

`-captureTimerCounter`

Capture the available timer(s) and counter(s) when capturing an event.

`eventName`

Specify the name of the event to be modified.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCaptureEvent -win $_nWave1 {counte1} -begin 0.00 -end 15.0M \
-captureTimerCounter
```

wvDeleteEvent

Description

Delete an event/complex event in event window.

Syntax

```
wvCaptureEvent [-win window] eventName
```

Arguments

`eventName`

Specify the name of the event to be modified.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDeleteEvent -win $_nWave1 {event3}
```

wvGetAllEvents

Description

Query all events' definition list.

Syntax

```
wvGetAllEvents [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Event or sequence name list if successful; otherwise, returns 0.

Example

```
wvGetAllEvents -win $_nWave1 event1
```

wvGetEventDefinition

Description

Query an event definition.

Syntax

```
wvGetEventDefinition [-win window] eventName
```

Arguments

`eventName`

Specify the name of the event to be modified.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Definition of this event if successful; otherwise, returns 0.

Example

```
wvGetEventDefinition -win $_nWave1 event1
```

wvModifyComplexEvent

Description

Modify the complex event definition in event window.

Syntax

```
wvModifyComplexEvent [-win window] eventName eventDefinition
```

Arguments

`eventName`

Specify the name of the event to be modified.

`eventDefinition`

Specify the new event definition content.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvModifyComplexEvent -win $_nWave1 {complex_event1}
{
    COMPLEX-EVENT  event1  {
      LEVEL 0  {
          timer1: START  timer2: STOP
           IF   ( event0 OCCURS  1)  THEN terminate
      }
      LEVEL 1  {
            timer1: STOP  timer2: STOP
            IF   ( counter1 >  2)  THEN trigger
      }
    }
}
```

wvModifyEvent

Description

Modify event in event window.

Syntax

`wvModifyEvent [-win` *`window`*`]` *`eventName eventString`*

Arguments

`eventName`

Specify the name of the event to be modified.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvModifyEvent -win $_nWave1 {event2} {"/system/addr[7:0]" !==
'h21}
```

wvRestoreEvent

Description

Restore events/complex events to event window from file.

Syntax

```
wvRestoreEvent [-win window] fileName
```

Arguments

`fileName`

Specify restored file name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRestoreEvent -win $_nWave1 /tmp/evt0
```

wvSaveEvent

Description

Save event window content to a file.

Syntax

```
wvSaveEvent [-win window] fileName
```

Arguments

`fileName`

Specify output file name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

File name if successful; otherwise, returns 0.

Example

```
wvSaveEvent -win $_nWave1 /tmp/evt0
```

wvSaveEventLog

Description

Log the executive details of a complex event.

Syntax

```
wvSaveEventLog [-win window] "eventName" -atTime traceTime -toFile
fileName
```

Arguments

`-atTime traceTime`

Specify the time to trace.

`eventName`

Name of the event to be saved.

`-toFile fileName`

Save the log to the specified file.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSaveEventLog -win $_nWave1 "event1" -atTime 700 -toFile
“./event_log.rpt”
```

wvSaveVerilogExpression

Description

Save the logical operation to a file.

Syntax

```
wvSaveVerilogExpression [-win window] -file filename
```

Arguments

`-file filename`

Specify the logical operation file name to save to.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSaveVerilogExpression -win $_nWave1 -file ./1.exp
```

wvVerilogEvent

Description

Create Verilog event signals.

Syntax

```
wvVerilogEvent [-win window] eventName eventString
```

Arguments

`eventName`

Specify the event signal name.

`eventString`

Specify the event string.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Event ID.

Example

```
set current_event [wvVerilogEvent event1 {"/system/clock" == 1}]
```

wvVerilogExpression

Description

Create or update a Verilog logical expression signal.

Syntax

```
wvVerilogExpression [-win window] [-capture|-modify] [-delim
delim] signalName expressionString
```

Arguments

`-capture`

Capture the signal to the *nWave* window.

`-delim delim`

Specify the delimiter of the specified signal name. The default is '/'.

`expressionString`

Specify the logical expression string.

`-modify`

Modify the existing expression. This option is not specified when the `-capture` option is specified.

`signalName`

Specify the expression signal name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvVerilogExpression -win $_nWave1 "WORK\[7:0\]" "\"/system/
data\[7:0\]\""
wvVerilogExpression -win $_nWave2 -capture "ab\[1:0\]"
wvVerilogExpression -win $_nWave1 -modify {"addr_clock\[7:0\]"}
{"\"/system/addr\[7:0\]\"+" "\"/system/clock\""}
```

Radix

wvGetRadix

Description

Get the radix of signal(s). If no signal is specified, get the selected radix of signal(s).

Syntax

```
wvGetRadix [-win window] [-delim delim] {signalList|(group index)}
```

Arguments

`-delim delim`

Specify the delimiter of the specified signal name.

`(group index)`

Specify the position where the indice is a list of positions related to the group.

`signalList`

Specify a list of the signal names.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

List of the notation and format.

Example

```
set sigradix [wvGetRadix -win $_nWave1  -delim .
{system.i_CPU.clock} {system.iCPU.reset}]
$sigradix is {Signed Bin} {2Com Bin}
```

wvSetRadix

Description

Set radix for displaying signal values.

Syntax

```
wvSetRadix [-win window] [notation] [-format format]
[listSignalName | {(group index)}]
wvSetRadix [-win window] [-delim delim] [notation] [-format format
|
-localFormat format] listSignalName
wvSetRadix -format [Bin|Dex|...|Enum] -localFormat
[Bin|Dex|...|Enum]
```

Arguments

`-delim` *`delim`*

Delimiter of the specified signal name. The default is '/'.

`-format` *`format`*

Signal value format: Bin | Oct | Hex | Dec | ASCII | Enum.

`-localFormat` *`format`*

Signal value format only for local change: Bin | Oct | Hex | Dec | ASCII | Enum.

`(`*`group index`*`)`

If specified, only apply to the signal at the positions specified. Otherwise, apply to selected signals.

`listSignalName`

Specify list of the signal names which radix changes globally.

`notation`

Signal value notation style: -Unsigned | -2Com | -1Com | -Mag.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetRadix -format Oct  {(GA 1 3)} {(GB 2)}
wvSetRadix -2Com -format Oct
wvSetRadix -2Com -format Bin {/system/signal_a}
wvSetRadix -win $_nWave1 -format Enum {("G1" 11)}
wvSetRadix -win $_nWave1 -localFormat Enum {("G1" 14)}
```

Alias in Waveforms

NOTE: The "wv"-prefixed Tcl commands and "alias"-prefixed Tcl commands have similar functions. The only difference is the "alias"-prefixed Tcl commands only work for the *Alias Editor* form and do not update the *nWave* window; the "wv"-prefixed Tcl commands update the *nWave* window after executing the Tcl commands.

NOTE: The **wvAddAliasProgram**, **wvRemoveAlias**, **wvSetAlias**, and **wvUnsetAlias** Tcl commands do not have the corresponding "alias"-prefixed commands.

wvAddAliasFile

Description

Add alias from a file. Also see the corresponding **aliasAddAliasFile** Tcl command.

Syntax

```
wvAddAliasFile [-win window] fileName
```

Arguments

`fileName`

Alias file name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddAliasFile cpu.alias
```

wvAddAliasTable

Description

Add a new alias map table. Also see the corresponding **aliasAddAliasTable** Tcl command.

Syntax

```
wvAddAliasTable -table aliasTable [-row aliasName value [bgcolor]]
```

Arguments

`-table aliasTable`

Specify the alias table name.

`-row aliasName value`

Specify one alias map entry.

`bgcolor`

Specify the background color.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddAliasTable -table fsm_0
wvAddAliasTable -table fsm_0 -row Reset "2'b00"
```

wvAddSliceTable

Description

Add a new slice table. Also see the corresponding **aliasAddSliceTable** Tcl command.

Syntax

```
wvAddSliceTable -table sliceTable [-row busRange Radix -order
value]
```

Arguments

`-table sliceTable`

Specify the slice table name.

`-row bugRange Radix`

Specify the slice table row and radix. *alias_table* can be specified as the radix.

`-order value`

The slice table row order.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddSliceTable -table slice1 -row 5 Binary -order 1
wvAddSliceTable -table slice1 -row 6 alias_tab1 -order 2
```

wvDeleteAliasTable

Description

Delete alias map tables. Also see the corresponding **aliasDeleteAliasTable** Tcl command.

Syntax

```
wvDeleteAliasTable [-table aliasTable] [-row aliasName] [-all]
```

Arguments

`-table aliasTable`

Alias table name.

`-row aliasName`

Specify the alias map entry to be removed from alias table.

`-all`

Delete all alias tables.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDeleteAliasTable -table fsm_0
wvDeleteAliasTable -table fsm_0 -row Reset
wvDeleteAliasTable -all
```

wvDeleteSliceTable

Description

Delete slice tables. Also see the corresponding **aliasDeleteSliceTable** Tcl command.

Syntax

```
wvDeleteSliceTable [-table aliasTable] [-row aliasName] [-all]
```

Arguments

`-table` *`aliasTable`*

Slice table name.

`-row` *`aliasName`*

Specify the slice entry to be removed from slice table.

`-all`

Delete all slice tables.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDeleteSliceTable -table fsm_0
wvDeleteSliceTable -table fsm_0 -row Reset
wvDeleteSliceTable -all
```

wvModifyAliasTable

Description

Modify the map entry of the alias table. Also see the corresponding **aliasModifyAliasTable** Tcl command.

Syntax

```
wvModifyAliasTable -table aliasTableName -row aliasName value
modifedAliasName modifiedValue [bgcolor]
```

Arguments

`-table aliasTableName`

Alias table name.

`-row aliasName value`

Specify the alias map entry to be modified.

`modifiedAliasName`

The modified alias map name.

`modifiedAliasValue`

The modified alias value.

`bgcolor`

Specify the background color.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvModifyAliasTable -table fsm_0 -row Reset "3'b000"
```

wvModifySliceTable

Description

Modify the map entry of slice table. Also see the corresponding **aliasModifySliceTable** Tcl command.

Syntax

```
wvModifySliceTable -table sliceTable -row oldBusRange oldRadix
newBusRange newRadix -order newOrder
```

Arguments

`-table sliceTableName`

Specify the modified slice table name.

`-row oldBusRange oldRadix newBusRange newRadix`

Specify the old and new slice row and radix.

`-order newOrder`

The modified row order in the slice table.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvModifySliceTable -table slice1 -row 5 Binary 4 Octal -order 5
```

wvSaveAliasTable

Description

Save alias table to the specified file. Also see the corresponding **aliasSaveAliasTable** Tcl command.

Syntax

```
wvSaveAliasTable -file fileName
```

Arguments

`-file fileName`

If specified, save the alias table content to *fileName*. Otherwise, save to current open file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSaveAliasTable -file fsm.alias
```

wvAddAliasProgram

Description

Add alias from an executable program.

Syntax

```
wvAddAliasProgram [-win window] programName
```

Arguments

`programName`

Specify the program name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddAliasProgram adder
```

wvRemoveAlias

Description

Remove alias.

Syntax

```
wvRemoveAlias [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRemoveAlias
```

wvSetAliasTable

Description

Set alias table to signal(s).

Syntax

```
wvSetAliasTable [-win window] [-signal {(group index)}] -table
aliasTable [-global]
```

Arguments

`-signal {(group index)}`

If specified, alias table is applied to the specified signal position. Otherwise, apply to selected.

`-table aliasTable`

Specify the alias table name.

`-global`

If specified, the alias table is applied globally.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetAliasTable -win $_nWave1 -signal {(G1 1) (G2 2)} -table
fsm_0
-global
```

wvUnsetAliasTable

Description

Unset alias table to signal(s).

Syntax

```
wvUnsetAliasTable [-win window] [-signal {(group index)}][-global]
```

Arguments

`-signal {(group index)}`

If specified, alias table is removed from the specified signal position or the selected ones.

`-global`

If specified, the alias table is applied globally.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvUnsetAliasTable -win $_nWave1 -global -signal {(G1 3)}
```

Alias Editor

NOTE: The "wv"-prefixed Tcl commands and "alias"-prefixed Tcl commands have similar functions. The only difference is the "alias"-prefixed Tcl commands only work for the *Alias Editor* form and do not update the *nWave* window; the "wv"-prefixed Tcl commands update the *nWave* window after executing the Tcl commands.

aliasAddAliasFile

Description

Add alias from file. Also see the corresponding **wvAddAliasFile** Tcl command.

Syntax

```
aliasAddAliasFile filename
```

Arguments

`filename`

Alias file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasAddAliasFile cpu.alias
```

aliasAddAliasTable

Description

Add a new alias map table. Also see the corresponding **wvAddAliasTable** Tcl command.

Syntax

```
aliasAddAliasTable -table aliasTable [-row aliasName value
[bgcolor]]
```

Arguments

`-table aliasTable`

Specify the alias table name.

`-row aliasName value`

Specify one alias map entry.

`bgcolor`

Specify the background color.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasAddAliasTable -table fsm_0
aliasAddAliasTable -table fsm_0 -row Reset "2'b00"
```

aliasAddSliceTable

Description

Add a new slice table. Also see the corresponding **wvAddSliceTable** Tcl command.

Syntax

`aliasAddSliceTable -table sliceTable [-row busRange Radix -order value]`

Arguments

`-table sliceTable`

Specify the slice table name.

`-row bugRange Radix`

Specify the slice table row and radix. *alias_table* can be specified as the radix.

`-order value`

The slice table row order.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasAddSliceTable -table slice1 -row 5 Binary -order 1
aliasAddSliceTable -table slice1 -row 6 alias_tab1 -order 2
```

aliasAddCondAliasTable

Description

Add a new conditional alias map.

Syntax

```
aliasAddCondAliasTable -table CondAliasTable [-row CondSignal
AliasTable] [-order value]
```

Arguments

`-order value`

The conditional alias table row order.

`-row CondSignal AliasTable`

Specify the condition and the corresponding alias/slice table.

`-table CondAliasTable`

Specify the conditional alias table name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasAddCondAliasTable -table condition_alias_tab1
aliasAddCondAliasTable -table condition_alias_tab1 -row
"/logical_exp" "alias_tab1" -order 0
```

aliasDeleteAliasTable

Description

Delete alias map tables. Also see the corresponding **wvDeleteAliasTable** Tcl command.

Syntax

```
aliasDeleteAliasTable [-table aliasTable] [-row aliasName] [-all]
```

Arguments

`-table` *`aliasTable`*

Alias table name.

`-row` *`aliasName`*

Specify the alias map entry to be removed from the alias table.

`-all`

Delete all alias tables.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasDeleteAliasTable -table fsm_0
aliasDeleteAliasTable -table fsm_0 -row Reset
aliasDeleteAliasTable -all
```

aliasDeleteCondAliasTable

Description

Delete conditional alias tables.

Syntax

```
aliasDeleteCondAliasTable [-table CondAliasTable] [-row CondSignal
AliasTable] [-order value] [-all]
```

Arguments

`-all`

Delete all conditional alias tables.

`-order value`

Specify the condition order.

`-row CondSignal AliasTable`

Specify the condition and the corresponding alias/slice table.

`-table CondAliasTable`

Specify the conditional alias table name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasDeleteCondAliasTable -table condition_alias_tab1
aliasDeleteCondAliasTable -table condition_alias_tab1 -row
"/logical_exp" "alias_tab1" -order 0
aliasDeleteCondAliasTable -table condition_alias_tab1 -all
```

aliasDeleteSliceTable

Description

Delete slice tables. Also see the corresponding **wvDeleteSliceTable** Tcl command.

Syntax

```
aliasDeleteSliceTable [-table aliasTable] [-row aliasName] [-all]
```

Arguments

`-table aliasTable`

Slice table name.

`-row aliasName`

Specify the slice entry to be removed from the slice table.

`-all`

Delete all slice tables.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasDeleteSliceTable -table fsm_0
aliasDeleteSliceTable -table fsm_0 -row Reset
aliasDeleteSliceTable -all
```

aliasModifyAliasTable

Description

Modify the map entry of the alias table. Also see the corresponding **wvModifyAliasTable** Tcl command.

Syntax

```
aliasModifyAliasTable -table aliasTableName -row aliasName value
modifedAliasName modifiedValue [bgcolor]
```

Arguments

`-table` *`aliasTableName`*

Alias table name.

`-row` *`aliasName value`*

Specify the alias map entry to be modified.

`modifiedAliasName`

The modified alias map name.

`modifiedAliasValue`

The modified alias value.

`bgcolor`

Specify the background color.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasModifyAliasTable -table fsm_0 -row Reset "3'b000"
```

aliasModifyCondAliasTable

Description

Modify the map entry of conditional alias tables.

Syntax

```
aliasModifyCondAliasTable [-table CondAliasTable] [-row
oldCondSignal oldAliasTable newCondSignal newAliasTable] [-order
oldOrder newOrder]
```

Arguments

`-table` *`CondAliasTable`*

Specify the conditional alias table name.

`-row` *`oldCondSignal oldAliasTable newCondSignal newAliasTable`*

Specify the condition and the corresponding alias/slice table to be modified.

`-order` *`oldOrder newOrder`*

The modified conditional alias table order.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasModifyCondAliasTable -table condition_alias_tab1
aliasModifyCondAliasTable -table condition_alias_tab1 -row
"/logical_exp" "alias_tab1" "/logical_exp" "alias_tab2"-order 0 1
```

aliasModifySliceTable

Description

Modify the map entry of the slice table. Also see the corresponding **wvModifySliceTable** Tcl command.

Syntax

```
aliasModifySliceTable -table sliceTable -row oldBusRange oldRadix
newBusRange newRadix -order newOrder
```

Arguments

`-table` *`sliceTableName`*

Specify the modified slice table name.

`-row` *`oldBusRange oldRadix newBusRange newRadix`*

Specify the old and new slice row and radix.

`-order` *`newOrder`*

The modified row order in the slice table.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasModifySliceTable -table slice1 -row 5 Binary 4 Octal -order
5
```

aliasSaveAliasTable

Description

Save the alias table to the specified file. Also see the corresponding **wvSaveAliasTable** Tcl command.

Syntax

`aliasSaveAliasTable -file` *`fileName`*

Arguments

`-file` *`fileName`*

If specified, save the alias table content to *fileName*. Otherwise, save to current open file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
aliasSaveAliasTable -file fsm.alias
```

Transaction

wvAddSelectedToAnalysisWnd

Description

Add the selected transaction/message streams to the *Analysis* window.

Syntax

```
wvAddSelectedToAnalysisWnd [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddSelectedToAnalysisWnd -win $_nWave2
```

wvClearColorize

Description

Clear the applied colorization results in *nWave*.

Syntax

```
wvClearColorize -win window -message|-transaction
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`-message|-transaction`

Clear message or transaction streams.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvClearColorize -win $_nWave2 -message
```

wvClearFilter

Description

Clear the applied filtering results in *nWave*.

Syntax

```
wvClearFilter -win window -message|-transaction
```

Arguments

`-message|-transaction`

Clear message or transaction streams.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvClearFilter -win $_nWave2 -message
```

wvColorize

Description

Apply the specified colorize rule table to *nWave*.

Syntax

```
wvColorize -win window -table {tableName} -case on|off
```

Arguments

`-table {tableName}`

Specify the table name to apply.

`-case on|off`

Turn the case sensitivity when applying rules *on* or *off*.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvColorize -win $_nWave2 -table {Colorize_Table_0} -case off
```

wvDispHideAttr

Description

Display or hide attributes of the selected transaction/message stream.

Syntax

```
wvDispHideAttr [-win window] [-msg|-trans]
```

Arguments

`-msg|-trans`

Specify either transaction or message streams.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDispHideAttr -win $_nWave2 -trans
```

wvExpandAttribute

Description

Expand variable attributes for the selected transaction/message stream.

Syntax

```
wvExpandAttribute [-win window] -signal {stream_name}
-attr_id_list {attr_id_1} {attr_id_2}... -prefix {prefix}
```

Arguments

`-attr_id_list {attr_id_1} {attr_id_2}...`

The attribute ID list to be created as signals.

`-prefix {prefix}`

Specify the prefix for newly created signals' name if necessary. It can be empty if there is no prefix.

`-signal {stream_name}`

The full name of the signal to be expanded.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvExpandAttribute -win $_nWave2 -signal {/msg_root/\$root
.\cpu_ref_instr_cover_callback::exec_instr} -attr_id_list {532}
{533} -prefix {(\$root
.\cpu_ref_instr_cover_callback::exec_instr):}
wvExpandAttribute -win $_nWave2 -signal {/msg_root/\$root
.\cpu_ref_instr_cover_callback::exec_instr} -attr_id_list {532}
-prefix {}
```

wvExpandShrinkMessage

Description

Expand overlapped messages or shrink expanded messages.

Syntax

```
wvExpandShrinkMessage [-win winID]
```

Arguments

`-win winID`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvExpandShrinkMessage -win $_Wave1
```

wvExpandShrinkTransaction

Description

Expand overlapped transactions or shrink expanded transactions.

Syntax

```
wvExpandShrinkTransaction [-win winID]
```

Arguments

`-win winID`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvExpandShrinkTransaction -win $_Wave1
```

wvFilter

Description

Apply the specified filter rule table to *nWave*.

Syntax

```
wvFilter -win window -table {tableName} -case on|off
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`-table {tableName}`

Specify the table name to apply.

`-case on|off`

Turn the case sensitivity when applying rules *on* or *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvFilter -win $_nWave2 -table {Filter_Table_0} -case off
```

wvGetAttributeList

Description

Get the attribute list for the specified transaction from the active file.

Syntax

```
wvGetAttributeList [-win window] -transId
```

Argument

`-win window`

Specify the window ID of the invoking *nWave* window.

`-transId`

Specify the transaction ID.

Value Returned

1 if successful, followed by the attribute list of the specified transaction. If no corresponding attributes are found in the active file, return 0.

Example

```
wvGetAttributeList -win $_nWave1 -transId 51
```

wvGetAttributeValue

Description

Get the attribute value for the attribute of the specified transaction from the active file.

Syntax

```
wvGetAttributeValue [-win window] -transId -attrName
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-transId`

Specify the transaction ID.

`-attrName`

Specify the attribute name.

Value Returned

1 if successful, followed by the specified attribute value. If no specified attributes or no specified transactions are found in the active file, return 0.

Example

```
wvGetAttributeValue -win $_nWave1 -transId 48 -attrName
"mda\[0\]"
wvGetAttributeValue -win $_nWave1 -transId 51 -attrName "Msg"
wvGetAttributeValue -win $_nWave1 -transId 48 -attrName
Call_Stack
```

wvGetRelatedTransactionIdList

Description

Get the ID list of the specified transaction's related transactions from the active file.

Syntax

```
wvGetRelatedTransactionIdList [-win window] -transId
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-transId`

Specify the transaction ID.

Value Returned

1 if successful, followed by the ID list of the specified transaction's related transactions. If no related transaction are found, return 0.

Example

```
wvGetRelatedTransactionIdList -win $_nWave1 -transId 3
```

If successful, "1 5 8" is returned. 1 is for success. The related ID list is “5” and "8".

wvGetSelectedTransactionId

Description

Return to the selected transaction ID.

Syntax

```
wvGetSelectedTransactionId [-win window]
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful, followed by the selected transaction ID. If no transactions are selected, return 0.

Example

```
wvGetSelectedTransactionId -win $_nWave1
```

wvGetStreamName

Description

Get a stream (signal) name for the specified transaction.

Syntax

```
wvGetStreamName [-win window] -transId
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-transId`

Specify the transaction ID.

Value Returned

1 if successful, followed by the stream name of the specified transaction. If no corresponding streams are found in the active file, return 0.

Example

```
wvGetStreamName -win $_nWave1 -transId 51
```

wvSelectTransaction

Description

Select a transaction.

Syntax

```
wvSelectTransaction [-win window] -atTag time signalName [-transId
transId] streamName
```

Arguments

`-atTag`

Select the specified time and transaction of the stream.

`-transId`

The transaction ID of the transaction.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectTransaction -win $_nWave1 -atTag 339.068564 “/Top/fiber0
wvSelectTransaction -win $_nWave1 -atTag 100.5 -transId 10 "/
BusTop/MyAHB_1/AhbTransaction"
```

wvTpfCloseForm

Description

Close the transaction property form.

Syntax

`wvTpfCloseForm -win` *`window`*

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvTpfCloseForm -win $_nWave1
```

wvTpfDisplayForm

Description

Open the transaction property form.

Syntax

```
wvTpfDisplayForm [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvTpfDisplayForm -win $_nWave1"
```

wvTpfSetActivePage

Description

Set the active page in the transaction property form.

Syntax

```
wvTpfSetActivePage [-win winID] -page
```

Arguments

`-page`

Specify the page name.

`-win winID`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvTpfSetActivePage -win $_nWave1 -page Attributes
```

wvTpfSetTransAttrRadix

Description

Set the radix for the specified transaction attribute radix.

Syntax

```
wvTpfSetTransAttrRadix -win window -attrName attrName
-dataTypeName dataTypeName -format
```

Arguments

`-attrName` *`attrName`*

Specify the attribute name.

`-dataTypeName` *`dataTypeName`*

Specify the data type name. Use -attrName and -dataTypeName together to identify attributes of transaction streams.

`-format`

Specify the format [Binary | Octal | Hexadecimal | Udecimal | SDecimal].

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvTpfSetTransAttrRadix -win $_nWave1 -attrName b -dataTypeName
sv_reg -format Hexadecimal
```

Group

wvAddGroup

Description

Add a group to the waveform window.

Syntax

```
wvAddGroup [-win window] groupName
```

Arguments

`groupName`

Specify the added group name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddGroup GA
```

wvCollapseBus

Description

Collapse a bus.

Syntax

```
wvCollapseBus [-win window] [{(group index)} || -name {name}]
```

Arguments

`(group index)`

If specified, only apply to the signal at the position specified. Otherwise, apply to selected signals.

`-name name`

Specify the full name of bus.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
wvCollapseBus -win $_nWave1
```
Use selected bus to collapse. Only one bus is selected at one time.

```
wvCollapseBus -win $_nWave1  -name {/system/addr[7:0]}
wvCollapseBus -win $_nWave1  {(G1 3)}
```

wvCollapseGroup

Description

Hide all signals in the specified group.

Syntax

```
wvCollapseGroup [-win window] groupName
```

Arguments

`groupName`

Specify the group name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCollapseGroup  GA
```

wvExpandAllGroups

Description

Show all signals in all groups.

Syntax

```
wvExpandAllGroups -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful, 0 otherwise.

Example

```
wvExpandAllGroups -win $_nWave1
```

wvExpandGroup

Description

Show all signals in the specified group.

Syntax

```
wvExpandGroup [-win window] groupName
```

Arguments

`groupName`

Specify the group name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvExpandGroup  GA
```

wvGoToGroup

Description

Scroll window to the specified group.

Syntax

```
wvGoToGroup [-win window] groupName
```

Arguments

`groupName`

Specify the group name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGoToGroup GA
```

wvIsGroupExist

Description

Check whether a group exists.

Syntax

```
wvIsGroupExist [-win window] groupFullName
```

Arguments

`groupFullName`

The full name of a target group.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if exists, 0 otherwise

Example

```
wvIsGroupExist -win $_nWave1 {G1/test/test-1}
```

wvRenameGroup

Description

Rename the selected group.

Syntax

```
wvRenameGroup [-win window] [oldName] newName
```

Arguments

`oldName`

If specified, only apply to the specified group. Otherwise, apply to selected group.

`newName`

New group name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRenameGroup  GA GNew
wvRenameGroup GNew
```

Event Sequence Window

wvAddSignalsToEventSequence

Description

Add signals to the *Event Sequence* window.

Syntax

```
wvAddSignalsToEventSequence [-win window] [-delim delim]
(listSignalName|-all)
```

Arguments

`-delim` *delim*

Specify the delimiter of the signal name.

`listSignalName`

A list of signal name.

`-win` *window*

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddSignalsToEventSequence -win $_nWave1 {/system/clock} {/
system/cpu/clock}
```

wvCloseEventSequence

Description

Close the *Event Sequence* window.

Syntax

```
wvCloseEventSequence [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCloseEventSequence -win $_nWaveEvtSeq1
```

wvDropToEventSequence

Description

Drag and drop signals to the *Event Sequence* window.

Syntax

```
wvDropToEventSequence [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDropToEventSequence -win $_nWaveEvtSeq0
```

wvOpenEventSequence

Description

Invoke an *Event Sequence* window.

Syntax

```
wvOpenEventSequence [-win window] [-signal (select|all)] [-sort]
[-range]
```

Arguments

`-range`

If specified, multiple time ranges between cursor and marker time is shown on window.

`-signal select|all`

Add selected signal or all displayed signals to the *Event Sequence* window.

`-sort`

Option to sort by sequence.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Top window ID if successful; otherwise, returns 0.

Example

```
wvOpenEventSequence
```

wvRemoveSignalsFromEventSequence

Description

Remove the signals from the *Event Sequence* window.

Syntax

```
wvRemoveSignalsFromGlitchWindow [-win window] [-delim delim]
listSignalname
```

Arguments

`-delim delim`

Specify the delimiter of the signal name.

`listSignalName`

A list of signal name.

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRemoveSignalsFromEventSequence -delim {.}{system.clock}
{system.cpu.clock}
```

wvEventSequenceCapture

Description

Capture current window to an image file in PNG format.

Syntax

```
wvEventSequenceCapture [-win window] [-footer footer] [-region
region] -file fileName
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`-footer footer`

Any printable message shown on the footer.

`-region region`

Specify the region to be printed or saved.

`-file fileName`

Specify file name to save captured image.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceCapture -footer "eve_seq2" -file "/home/
eve_seq2.png"
```

wvEventSequenceGoToTime

Description

Go to specified time.

Syntax

```
wvGlitchWindowGoToTime [-win window] time
```

Arguments

`time`

Specify time.

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 jump time if successful; otherwise, returns 0.

Example

```
wvEventSequenceGoToTime 100.0
```

wvEventSequenceMerge

Description

When the option is *on*, continuous cells with the same value of the same row is merged.

Syntax

```
wvEventSequenceMerge [-win window] on|off
```

Arguments

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceMerge -win $_nWave1 on
```

wvEventSequenceSearchBy

Description

Specify search mode in *Event Sequence* window.

Syntax

```
wvEventSequenceSearchBy [-win window] -both|-glitch
```

Arguments

`-both`

Specify the search mode by any value changes.

`-glitch`

Specify the search mode by glitches.

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceSearchBy -glitch
```

wvEventSequenceSearchPrev

Description

Search previous event time.

Syntax

```
wvEventSequenceSearchPrev [-win window] [-case] [value]
```

Arguments

`-case`

Perform case-sensitive searching on the specified *value*.

`value`

If specified, it jumps to next time with this *value*.

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceSearchPrev
```

wvEventSequenceSearchNext

Description

Search next event time.

Syntax

```
wvEventSequenceSearchNext [-win window] [-case] [value]
```

Arguments

`-case`

Perform case-sensitive searching on the specified *value*.

`value`

Jump to next time with this *value*.

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceSearchNext
```

wvEventSequenceSelect

Description

Specify selected rows in *Event Sequence* window.

Syntax

```
wvEventSequenceSelect [-win window] [rows]
```

Arguments

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

`rows`

The selected rows.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceSelect 1 2 5
```

```
wvEventSequenceSelect
```

wvEventSequenceSelectAll

Description

Select all signals on the *Event Sequence* window.

Syntax

```
wvEventSequenceSelectAll [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceSelectAll -win $_nWave1
```

wvEventSequenceSetGridMode

Description

This option sets the grid mode.

Syntax

```
wvEventSequenceSetGridMode [-win evtseqwinid]
```

Arguments

`-win` *`evtseqwinid`*

Specify the window ID of the *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceSetGridMode -win $_nWaveEvtSeq0
```

wvEventSequenceSetWaveformMode

Description

This option sets the waveform mode.

Syntax

```
wvEventSequenceSetWaveformMode [-win evtseqwinid]
```

Arguments

`-win` *`evtseqwinid`*

Specify the window ID of the *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
WvEventSequenceSetWaveformMode -win $_nWaveEvtSeq0
```

wvEventSequenceSort

Description

Re-order the event sequence by turning *on* or *off* sequence sorting.

Syntax

```
wvEventSequenceSort [-win window] on|off
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceSort -win $_nWave1 on
```

wvEventSequenceSyncCursor

Description

Turn synchronizing the cursor time with the *nWave* window *on* or *off*.

Syntax

```
wvEventSequenceSyncCursor [-win window] on|off
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvEventSequenceSyncCursor -win $_nWave1 on
```

wvEventSequenceZoomIn

Description

This option increases the cell width by a factor of 2.

Syntax

```
wvEventSequenceZoomIn [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

Cell width if successful; otherwise, returns 0.

Example

```
wvEventSequenceZoomIn
```

wvEventSequenceZoomOut

Description

This option expands the cell width by a factor of 2.

Syntax

```
wvEventSequenceZoomOut [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Event Sequence* window.

Value Returned

Cell width if successful; otherwise, returns 0.

Example

```
wvEventSequenceZoomOut
```

wvSaveEventSequence

Description

Save content on the *Event Sequence* window to a specified file.

Syntax

```
wvSaveEventSequence [-win window] fileName
```

Arguments

`-win window`

Specify the window ID of the invoking *Event Sequence* window.

`fileName`

Specify the output file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSaveEventSequence -win $_nWaveEvtSeq3 "./2. txt
```

Comment

wvAddComment

Description

Add a comment row.

Syntax

```
wvAddComment [-win window] [commentName]
```

Arguments

`commentName`

Name of the added comment shown on the signal window. The default is **comment**.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Comment row ID if successful; otherwise, returns 0.

Example

```
set comment_row [wvAddComment "myComment"]
```

wvAddCommentBox

Description

Add a comment box to comment row.

Syntax

```
wvAddCommentBox [-win window] -rowid commentRowId -style boxStyle
[-activetrace groupName signalName] [-delim delimiter]
-left leftTime  -right rightTime  [-dy distance] [-h height]
[-attach time] [-attachdy distance] comment
```

Arguments

`-activetrace`

Add additional information about the specified signal to the comment box.

`-attach` *`attachTime`*

Specify the attach time (only effective to the *squareAttach* style).

`-attachdy` *`attDistance`*

Specify the attach distance to the comment row bottom (only effective to the *squareAttach* style).

`comment`

Comment string.

`-delim`

Specify the delimiter for the **-activetrace** option. The options are a period (.) or slash (/).

`-dy` *`distance`*

Specify the distance in pixels between upper bound and comment row bottom.

`-h` *`height`*

Specify the height of the comment box.

`-left` *`leftTime`*

Specify the time for left bound.

`-right` *`rightTime`*

Specify the time for right bound.

`-rowid` *`commentRowId`*

The comment row ID returned by **wvAddComment** command.

`-style`

Specify the style of the comment box.
square | attachSquare | period | arrowPeriod

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddCommentBox -win $_nWave1 -delim "." -activetrace {G1}
{system.R_W} -left \ 100.000000 -right 300.000000 {Comment Box4}
```

wvDelCommentBox

Description

Delete a comment box.

Syntax

```
wvDelCommentBox [-win window] -rowid commentRowId -boxid
commentBoxId
```

Arguments

`-rowid` *`commentRowId`*

The commenr row ID returned by **wvAddComment** command.

`-boxid` *`commentBoxId`*

The comment box ID returned by creation command.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDelCommentBox -rowid $comment_row -boxid $comment_box
```

wvDeleteComment

Description

Remove active trace comments.

Syntax

```
wvDeleteComment [-win window] -activetrace -group
```

Arguments

`-activetrace`

Remove all active trace comments.

`-group`

Remove active assertion trace details from comments in the specified group.

Example

```
wvDeleteComment -win $_nWave1 -activetrace
```

wvLockAllCommentBoxes

Description

Lock all the comment boxes by Tcl. If a comment box is locked, it cannot be moved/cut/modified in the comment row. That is, you cannot select it and make it editable by mouse. You can click right-click option on comment box to popup a menu and press button to change lock mode. Therefore, you can still modify a comment box by unlocking it after calling Tcl **wvLockAllCommentBoxes**. The lock mode of new added comment boxes is unlocked. That is, you can do any manipulation to them as well as usual. Also, you can lock a new added box by using the right-click option, if necessary.

Syntax

```
wvLockAllCommentBoxes [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvLockAllCommentBoxes -win $_nWave1
```

wvModifyCommentBox

Description

Modify a comment box.

Syntax

```
wvModifyCommentBox [-win window] -rowid commentRowId -boxid boxid
{-left leftTime | -right rightTime | -dy distance | -h height
|-attach attachTime | -attachdy attDistance} comment
```

Arguments

`-attach attachTime`

Specify the attach time.

`-attachdy attDistance`

Specify the attach distance to the comment row bottom.

`comment`

Comment string.

`-dy distance`

Specify the distance in pixels between upper bound and comment row bottom.

`-h height`

Specify the height of the comment box.

`-left leftTime`

Specify the left time for left bound time.

`-right rightTime`

Specify the right time for right bound time.

`-rowid commentRowId`

The commenr row ID returned by **wvAddComment** command.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Comment box ID.

Example

```
wvModifyCommentBox -rowid $comment_row -boxid $comment_box -left
50.0
```

wvRenameComment

Description

Rename the name of the comment row.

Syntax

```
wvRenameComment [-win window] -rowid commentRowId commentName
```

Arguments

`-rowid commentRowId`

The comment row ID returned by **wvAddComment** command.

`commentName`

Specify the new name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Comment raw ID if successful; otherwise, returns 0.

Example

```
wvRenameComment -rowid $comment_row -boxid "new_comment"
```

wvUnlockAllCommentBoxes

Description

Inverse of **wvLockAllCommentBoxes** command. Unlocks all comment boxes.

Syntax

```
wvUnlockAllCommentBoxes [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvUnlockAllCommentBoxes -win $_nWave1
```

View

wvAddCompressTimeRange

Description

Insert compressed dump-off time ranges.

Syntax

```
wvAddCompressTimeRange -win window [from_time to_time|dumpOff]
```

Arguments

`dumpOff`

Display all dump off time ranges.

`from_time`

Specify the start time of the compressed time range.

`to_time`

Specify the end time of the compressed time range.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddCompressTimeRange -win $_nWave1 100 130
```

wvAutoInsertDumpoffs

Description

Automatically collapse all dump-off regions and insert the compressed region into the *Compress Time Range* form.

Syntax

```
wvAutoInsertDumpoffs -win window on|off
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`on|off`

Specify whether to automatically insert the dump-off ranges.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAutoInsertDumpoffs -win $_nWave2 off
```

wvBusWaveform

Description

Specify bus waveform display type for the selected signals.

Syntax

```
wvBusWaveform [-win window] -analog|-digital
```

Arguments

`-analog`

Analog display.

`-digital`

Digital display.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvBusWaveform -digital
```

wvCenterCursor

Description

Move waveform to the time range centered by the current cursor time.

Syntax

```
wvCenterCursor [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCenterCursor
```

wvCenterMarker

Description

Move waveform to the time range centered by the current marker time.

Syntax

```
wvCenterMarker [-win window] [markerName]
```

Arguments

`markerName`

Specify the marker name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCenterMarker
```

wvClearAssertionAnalysisMask

Description

Clear the assertion analysis mask in the waveform pane.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvClearAssertionAnalysisMask
```

wvClearGridLine

Description

Clear all triangle markers for the specified signal. Only one signal name may be specified for each call.

Syntax

```
wvCreateGridLine [-win winId] [-delim delimiter] signalname
```

Arguments

`-delim delimiter`

The delimiter of the specified signal name.

`signalName`

Specify the signal name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvClearGridLine -win $_nWave1 {/system/R_W}
wvClearGridLine -win $_nWave1 -delim "." {system.R_W}
```

wvCollapseCompressTimeRange

Description

Collapse the compressed dump-off time range.

Syntax

```
wvCollapseCompressTimeRange -win window from_time to_time
left|right
```

Arguments

`from_time`

Specify the start time of the compressed time range.

`left|right`

Collapse the compressed dump-off time range to the left or the right.

`to_time`

Specify the end time of the compressed time range.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCollapseCompressTimeRange -win $_nWave2 500 1000 left
```

wvCollapseTime

Description

Collapse an area with transition sequences or ordered regions.

Syntax

```
wvCollapseTime [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCollapseTime -win $_nWave1
```

wvCreateGridLine

Description

Add triangle markers on the waveform pane.

Syntax

```
wvCreateGridLine [-win winId] [-delim delimiter] [-pattern
TRIANGLE|REV_TRIANGLE] [-color colorId] [-append] [-group]
signalName {(time)}
```

Arguments

`-append`

Do not replace previously created triangle markers.

`-color colorId`

Specify the color name.

`-delim delimiter`

The delimiter of the specified signal name.

`-group`

Specify the group name.

`-pattern TRIANGLE|REV_TRIANGLE`

Specify the pattern of triangle markers as either TRIANGLE or REV_TRIANGLE.

`signalName`

Specify the signal name.

`time`

Specify the time. Multiple times can be specified.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCreateGridLine -win $_nWave1 -pattern TRIANGLE -color ID_RED5
-append -group {G1} {/system/R_W} 100 200 300 400 500 600 700
wvCreateGridLine -win $_nWave1 -delim "." -pattern REV_TRIANGLE
-color ID_RED5 system.clock 500 600
```

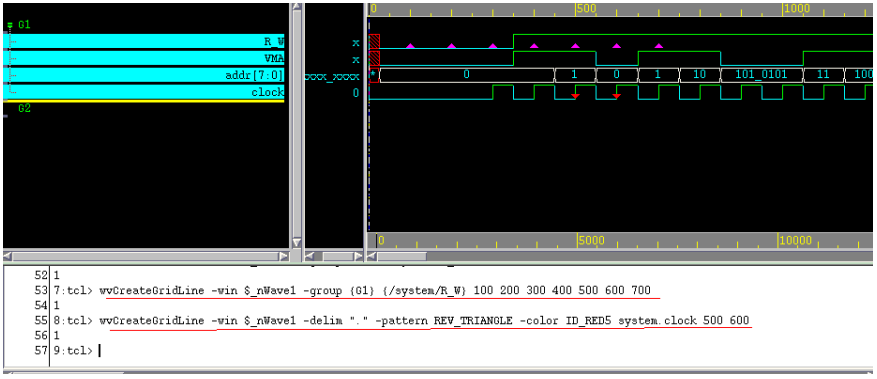


Figure: Create Grid Lines in nWave

wvDeleteCompressTimeRange

Description

Delete compressed dump-off time ranges.

Syntax

```
wvDeleteCompressTimeRange -win window [from_time to_time | all]
```

Arguments

`all`

Remove all dump off time ranges from the display.

`from_time`

Specify the start time of the compressed time range.

`to_time`

Specify the end time of the compressed time range.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDeleteCompressTimeRange -win $_nWave1 100 130
wvDeleteCompressTimeRange -win $_nWave1 all
```

wvDeleteMarker

Description

Delete the user-created marker.

Syntax

```
wvDeleteMarker -win window markerName1 markerName2...markerNameN
```

Arguments

`markerName1 markerName2...markerNameN`

Specify the markers to be deleted.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDeleteMarker -win $_nWave1 "marker1"
wvDeleteMarker -win $_nWave1 "marker1" "marker2"
```

wvDisplayGridCount

Description

Specify whether to display the grid count or not.

Syntax

`wvDisplayGridCount [-win` *`window`*`] [-off]`

The default is on.

Arguments

`-off`

Turn off the grid count display.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDisplayGridCount -off
```

wvExpandCompressTimeRange

Description

Expand the compressed dump-off time range.

Syntax

`wvExpandCompressTimeRange -win` *`window`* `from_time to_time left|right`

Arguments

`from_time`

Specify the start time of the compressed time range.

`left|right`

Expand the compressed dump-off time range to the left or the right.

`to_time`

Specify the end time of the compressed time range.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvExpandCompressTimeRange -win $_nWave2 500 1000 right
```

wvExpandTime

Description

Expand an area with transition sequences or ordered regions.

Syntax

```
wvExpandTime [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvExpandTime -win $_nWave1
```

wvFitSelected

Description

Fit the selected in the window.

Syntax

```
wvFit [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvFitSelected
```

wvGetCursor

Description

Query current cursor time on specified *nWave* window.

Syntax

```
wvGetCursor [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 factor x time_unit if successful; otherwise, returns 0.

Example

```
wvGetCursor -win $_nWave1
```

It returns *1 0.35 x 1.3ps*.

wvGetMarker

Description

Query current marker time on specified *nWave* window.

Syntax

```
wvGetMarker [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 factor x time_unit if successful; otherwise, returns 0.

Example

```
wvGetMarker -win $_nWave1
```

It returns *1 6400 x 0.0001ps*.

wvGetViewTimeRange

Description

Obtain the time range of the current view.

Syntax

```
wvGetViewTimeRange [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

The following data is returned if successful; otherwise, returns 0:

1 startTime endTime

Example

```
wvGetViewTimeRange -win $_nWave2
```

wvGridBothEdge

Description

Draw grid lines on both rising and falling edges of the selected signal.

Syntax

```
wvGridBothEdge [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGridBothEdge -win $_nWave2
```

wvGridCycleTime

Description

Draw vertical grid lines in the waveform window for the specified time interval.

Syntax

```
wvGridCycleTime [-win window] -from fromTime -to toTime -cycleTime
cycleTime
```

Arguments

`-cycleTime` *`cycleTime`*

Specify the time interval for drawing grid lines.

`-from fromTime`
Specify the start time for drawing grid lines.

`-to toTime`
Specify the end time for drawing grid lines.

`-win window`
Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGridCycleTime -from 100 -to 10000 -cycleTime 50
```

wvGridFallingEdge

Description

Draw grids at falling edge.

Syntax

```
wvGridFallingEdge [-win window]
```

Arguments

`-win window`
Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGridFallingEdge -win $_nWave2
```

wvGridRisingEdge

Description

Draw grids at rising edge.

Syntax

```
wvGridRisingEdge [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGridRisingEdge -win $_nWave2
```

wvGridSetLockCount

Description

When this option is turned on, the grid number is fixed. When this option is *off*, the grid number can be changed.

Syntax

```
wvGridSetLockCount [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGridSetLockCount -win $_nWave2 -off
```

wvGridSetStartNum

Description

Specify the starting index number of grid lines associated with the **Cursor** in the waveform pane.

Syntax

```
wvGridSetStartNum [-win window] num
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGridSetStartNum -win $_nWave2 1
```

wvJumpCursorToGridNum

Description

When this option is *on*, the cursor jumps to the specified index number for the grid line.

Syntax

```
wvJumpCursorToGridNum [-win winId] [-off] num
```

Arguments

`-win winId`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvJumpCursorToGridNum -win $_nWave1 8
wvJumpCursorToGridNum -win $_nWave1 -off
```

wvLastView

Description

Restore zoom area and cursor time in last view.

Syntax

```
wvLastView [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvLastView
```

wvPanDown

Description

Pan 1/2 down of current view area to the current window.

Syntax

```
wvPanDown [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvPanDown
```

wvPanLeft

Description

Pan 1/2 left of current view area to the current window.

Syntax

```
wvPanLeft [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvPanLeft
```

wvPanRight

Description

Pan 1/2 right of current view area to the current window.

Syntax

```
wvPanRight [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvPanReft
```

wvPanUp

Description

Pan 1/2 up of current view area to the current window.

Syntax

```
wvPanUp [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvPanUp
```

wvRemoveGrid

Description

Remove all grids.

Syntax

```
wvRemoveGrid [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRemoveGrid
```

wvReportMarker

Description

Save the marker information to file.

Syntax

```
wvReportMarker [-win window] -toFile fileName
```

Arguments

`-toFile fileName`

Specify the name of report file.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvReportMarker -win $_nWave1 -toFile marker.rpt
```

wvRestoreMarker

Description

Load marker information from the previously saved marker file (*.*rpt*).

Syntax

```
wvRestoreMarker -win window -file fileName
```

Arguments

`-file` *`fileName`*

Specify the previously saved marker file to load.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRestoreMarker -win $_nWave1 -file /verdi/temp/1.rpt
```

wvSelectUserMarker

Description

Select the marker.

Syntax

```
wvSelectUserMarker [-win window] -name markerName
```

Arguments

`-name` *`markerName`*

Specify the marker.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectUserMarker -win $_nWave2 -name "m1"
```

wvSetCursor

Description

Set cursor time.

Syntax

```
wvSetCursor   [-win window] -begin|-end|time [-snap {group index}]
```

Arguments

`-begin`

Jump to beginning simulation time.

`-end`

Jump to end of simulation time.

`-snap`

If specified, the cursor is snapped to value change of specified signal.

`time`

Specify the new cursor time.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetCursor  -win $_nWave1 100.0
wvSetCursor -win $_nWave1 -begin
wvSetCursor -win $_nWave1 -end
wvSetCurosr -win $_nWave1 23003 -snap {G1 1}
```

wvSetMarker

Description

Set the time, color and line style of the marker.

Syntax

```
wvSetMarker [-win window] [-keepViewRange] [-name userMarkerName]
time [color] [lineStyle]
```

Arguments

`color lineStyle`

Specify the color and line style of the marker.

`-keepViewRange`

Fix the view range in the *nWave* window. The view range is not changed automatically to keep the marker in the middle of the view range.

`-name userMarkerName`

Specify the marker name.time

Specify the marker time.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetMarker -win $_nWave1 100.0
wvSetMarker -win $_nWave1 -name "a" 3 ID_YELLOW8 solid
```

wvSetMarkerOption

Description

Set the marker options in the *nWave* window.

Syntax

```
wvSetMarkerOption [-win windowId] [-frequency on/off] [-refmarker
marker_name] [-absolute on/off] [-adjacent on/off] [-relative on/
off] [-stickMarker on/off]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`-frequency`
on: Displays the difference as the frequency value.

off: Displays the difference as the time value.

`-refmarker`
Specifies the name of the reference marker.

`-absolute`
on: Displays the time of the marker.

off: Hides the time of the marker.

`-adjacent`
on: Displays the difference between the adjacent markers.

off: Hides the difference between the adjacent markers.

`-relative`
on: Displays the difference between the marker and the reference marker.

off: Hides the difference between the marker and the reference marker.

`-stickMarker`

on: Turns on the **Waveform -> Marker -> Stick Cursor/Marker on Waveform** option.

off: Turns off the **Waveform -> Marker -> Stick Cursor/Marker on Waveform** option.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetMarkerOption -win $_nWave2 -frequency on -refmarker "m1"
wvSetMarkerOption -win $_nWave2 -adjacent on -relative off
```

wvScrollDown

Description

Scroll down.

Syntax

```
wvScrollDown [-win window] [delta]
```

Arguments

`delta`

Specify the number of the rows to scroll. The default is **1**.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvScrollDown
```

wvScrollUp

Description

Scroll up.

Syntax

```
wvScrollUp [-win window] [delta]
```

Arguments

`delta`

Specify the number of the rows to scroll. The default is **1**.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvScrollUp
```

wvShowDeltaFreq

Description

Switch to show the frequency or delta time between the cursor and marker.

Syntax

```
wvShowDeltaFreq -win window -delta|-freq
```

Arguments

`-delta|-freq`

Show the frequency or delta time.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvShowDeltaFreq -win $_nWave2 -freq
```

wvSortSignal

Description

Sort all signals in the selected group(s), including sub-group signals, by their first transition (value change).

Syntax

```
wvSortSignal [-win winId] [-first_tran] {group_name 1} {group_name
2}...
```

Arguments

`-first_tran`

Sort signals by the first transition (value change).

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSortSignal -win $_nWave2 -first_tran {G1} {G2}
```

wvSortSignalByName

Description

Sort signals in ascending or descending order.

Syntax

```
wvSortSignalByName [-win winId] [-asc| -desc] [-group group_name]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-asc`

Sort signals by name in the ascending order.

`-desc`

Sort signals by name in the descending order.

`-group`

Sort signals in the specified group and its sub-group.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSortSignalByName -win $_nWave3 -asc
```

wvUnselectUserMarker

Description

Unselect the marker.

Syntax

```
wvUnselectUserMarker [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvUnselectUserMarker -win $_nWave2
```

wvZoom

Description

Zoom to the specify time range.

Syntax

```
wvZoom [-win window] time1 time2
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`time1`

Zoom from time.

`time2`

Zoom to time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvZoom 100.0 200.0
```

wvZoomAll

Description

Zoom the time to full range.

Syntax

```
wvZoomAll [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvZoomAll
```

wvZoomCursorMarker

Description

Zoom between cursor and marker.

Syntax

```
wvZoomCursorMarker [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvZoomCursorMarker
```

wvZoomIn

Description

Zoom into 1/2 of current viewing time range.

Syntax

```
wvZoomIn [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvZoomIn
```

wvZoomOut

Description

Zoom out to double of the current viewing time range.

Syntax

```
wvZoomOut [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvZoomOut
```

Edit

wvCopy

Description

Copy the selected signals to clipboard.

Syntax

```
wvCopy [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCopy
```

wvCopyFilePathToClipboard

Description

Copy the full file path to clipboard.

Syntax

```
wvCopyFilePathToClipboard -win window -fileId fileId
```

Arguments

`-fileId` *`fileId`*

Specify the file ID of the invoking file.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCopyFilePathToClipboard -win $_nWave1 -fileId 1
```

wvCut

Description

Cut the selected signals to the clipboard.

Syntax

```
wvCut [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCut
```

wvMoveSelected

Description

Move selected signals/groups to the signal cursor position.

Syntax

```
wvMoveSelected [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvMoveSelected
```

wvPaste

Description

Paste the signals from the clipboard.

Syntax

```
wvPaste [-win window]
```

Arguments

`-win window`

.Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvPaste
```

wvSetPosition

Description

Set position for the window of the signal.

Syntax

```
wvSetPosition [-win window] {(group index | last)}
```

Arguments

`(group index)`

Specify the index position in the group.

(group last)

Specify the position to the last line of the group.

`-win` *window*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetPosition {(G1 6)}
wvSetPosition {(G1 last)}
```

wvUndo

Description

Reverse execution to the previous editing command.

Syntax

```
wvUndo [-win window]
```

Arguments

`-win` *window*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvUndo
```

Search

wvGetSearchNextTime

Description

Same action as **wvSearchNext**, but no cursor time is changed, and no window redraw is performed.

Syntax

```
wvGetSearchNextTime -win window
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 and the current cursor time if successful; 0 otherwise.

Example

```
set time [wvGetSearchNextTime]
```
(the $time is 1 1000, if successful)

wvGetSearchPrevTime

Description

Same action as **wvSearchPrev**, but no cursor time is changed, and no window redraw is performed.

Syntax

```
wvGetSearchPrevTime -win window
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 and the current cursor time if successful; 0 otherwise.

Example

```
set time [wvGetSearchPrevTime -win $_nWave1]
```
(the $time is 1 1000, if successful)

wvGoToTime

Description

Go to specified time.

Syntax

```
wvGoToTime -win window [-marker] time
```

Arguments

`time`

Specify time.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGoToTime 100.0
```

wvSearchDown

Description

Search downward in signals or trigger signals using current search mode.

Syntax

```
wvSearchDown [-win nWaveWinId]
```

Arguments

`-win nWaveWinId`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSearchDown -win $_nWave1
```

wvSearchNext

Description

Search forward in time using current search mode.

Syntax

```
wvSearchNext [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 cursor time if successful; otherwise, returns 0.

Example

```
set time [wvSearchNext]
```
(the $time is 1 1000, if successful)

wvSearchPrev

Description

Search backward in time using current search mode.

Syntax

```
wvSearchPrev [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 cursor time if successful; otherwise, returns 0.

Example

```
set time [wvSearchPrev]
```
(the $time is 1 1000, if successful)

wvSearchUp

Description

Search upward in signals or trigger signals using current search mode.

Syntax

```
wvSearchUp [-win nWaveWinId]
```

Arguments

`-win` *`nWaveWinId`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSearchUp -win $_nWave1
```

wvSearchNextBySignal

Description

Jump to the nearest next value change time by specified signal and time, and then start searching. Glitches are skipped and the specified signal is not added into *nWave*. Note that the signal name should escape TCL specific characters, such as "\{}[]()$.

Syntax

```
wvSearchNextBySignal -win window -delim .|/ signalName -time
reference_time
```

Arguments

`-delim .|/`

The delimiter of the signal name. It should be "/" or "."

`signalName`

Specify the signal to search.

`-time reference_time`

The specified reference time with the current time unit.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 + time_value; otherwise, returns 0.

Example

```
wvSearchNextBySignal  -win $_nWave1 -delim . "system.addr\[7:0\]"
-time 200
```

wvSearchPrevBySignal

Description

Jump to the nearest previous value change time by specified signal and time, and then start searching. Glitches are skipped and the specified signal is not added into *nWave*. Note that the signal name should escape TCL specific characters, such as "\{}[]()$.

Syntax

```
wvSearchPrevBySignal -win window -delim .|/ signalName -time
reference_time
```

Arguments

`-delim .|/`

The delimiter of the signal name. It should be "/" or "."

`signalName`

Specify the signal to search.

`-time reference_time`

The specified reference time with the current time unit.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 + time_value; otherwise, returns 0.

Example

```
wvSearchPrevBySignal -win $_nWave1 -delim / "system/addr\[7:0\]"
-time 200
```

wvSetSearchConstraint

Description

Set constraints for the search value.

Syntax

```
wvSetSearchConstraint [-win window] [-valuestable comparator]
deltaTime [-occur times]
```

Arguments

`comparator`

<= | >= | [none]

`-valuestable`

Focus search on value changes that have specified stable time constraint only.

`-occur times`

Specify times of occurrence.

`deltaTime`

Specify delta stable time.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetSearchConstraint -valuestable <= 0  //search glitch only
```

wvSetSearchMode

Description

Specify search mode.

Syntax

```
wvSetSearchMode [-win window] -negedge|-posedge|-anyChange|-event
eventName|-value value|-misMatch|-analog value [tolerance]|[-case
on|off]|[-X2Y on|off]|[-skipGlitch on|off]|-attr {ruleString}
wvSetSearchMode [-win window] -analog
```

Arguments

`-anyChange`

Search by any value change.

`-analog value`

Specify the analog value and tolerance. The default tolerance is **zero**.

`-attr {ruleString}`

Specify a set of attribute rules to be searched. The rule string should be surrounded by braces.

`-case on|off`

Turn the case-sensitive searching *on* or *off*. The default is *off*.

`-event eventName`

Specify the event name.

`-mismatch`

Search by mismatches.

`-negedge`

Search by falling edge.

`-posedge`

Search by rising edge.

`-value` *`value`*

Specify the bus value. For 50, wildcard is accepted as search value. For 5.1, you can also specify transition value, such as 2->5 indicates searching transition from 2 to 5. To use it, you need to turn on **-X2Y** option as well.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-X2Y on|off`

To treat **-value** as transition value or not. The default is *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetSearchMode -negedge
wvSetSearchMode -analog 4.0 3.0
wvSetSearchMode -value "2->5" -X2Y off #search value "2->5"
wvSetSearchMode -value "2->5" -X2Y on #search transition 2 to 5
wvSetSearchMode -win $_nWave2 -attr {"Label"=="*test*"} -case on
```

Analog

wvAnalogExpression

Description

Create a signal using a Verilog logical expression.

Syntax

```
wvAnalogExpression [-win window] [-delim delim] signalName
expressionString
```

Arguments

`-delim delim`

Delimiter of the specified signal name. The default is '/'.

`expressionString`

Analog expression string.

`signalName`

Specify signal name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAnalogExpression {"v5" + "v7"}
```

wvAnalogToDigital

Description

Convert the specified analog signal to digital signal by specifying high and low threshold.

Syntax

```
wvAnalogToDigital [-win winID] (-prefix prefixName | [-delim
delim] -from analogSignal -to digitalSignal) -prefixscopename
TRUE|FALSE -high value -low value
```

Arguments

`-delim delim`

Delimiter of the specified signal name. The default is '/'.

`analogSignal`

Specify the analog signal to be converted.

`digitalSignal`

Specify the converted digital signal name.

`-prefixscopename TRUE|FALSE`

When set to TRUE, the scope name is added as a prefix for analog to digital signals. When set to FALSE, no prefix is added.

`-high value`

Specify high threshold.

`-low value`

Specify low threshold.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAnalogToDigital -high 2.5 -low 2.0 v5 dv5
wvAnalogToDigital -win $_nWave1 -prefixscopename TRUE -prefix
{AtoD_} -high 2.800000 -low 0.500000
```

wvAverageMinMaxRMS

Description

Compute the Min/Max/Average/RMS values of the selected signals.

Syntax

```
wvAverageMinMaxRMS [-win window] [-tofile fileName]
```

Arguments

`-tofile fileName`

Specify the output file name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAverageMinMaxRMS
```

wvGetSignalAMMR

Description

Get signal's Average, Min, Max, RMS.

Syntax

```
wvGetSignalAMMR [-win window] [-delim delim] [signalList]
```

Arguments

`-delim delim`

Delimiter of the specified signal name. The default is '/'.

`signalList`

A list of the signals.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Signal's Average, Min, Max, RMS list

Example

```
set ammr [wvGetSignalAMMR analog_sig1]
```

wvOverlay

Description

Copy and overlap the selected analog signals to a new row.

Syntax

```
wvOverlay [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvOverlay
```

wvRulerGrid

Description

Assign ruler and grid display options of the selected signals.

Syntax

```
wvRulerGrid [-win window] {-rulerX on|off | -rulerY on|off | -gridX
on|off | -gridY on|off | -reference value}
```

Arguments

`-gridX on|off`

Option to display the grid in X direction.

`-gridY on|off`

Option to display the grid in Y direction.

`-reference value`

Specify the reference analog value.

`-rulerX on|off`

Option to display the ruler in X direction.

`-rulerY on|off`

Option to display the ruler in Y direction.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRulerGrid -ruler X on -rulerY off -gridX on -gridY 0.5
-reference 2.5
```

wvVerticalFit

Description

Set the Y value view range to the full value range for the selected signals.

Syntax

```
wvVerticalFit [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvVerticalFit
```

wvWaveSlew

Description

Measure delta time by two specified signal values automatically for currently selected analog signal.

Syntax

```
wvWaveSlew [-win window] [-leftValue leftValue] [-rightValue
rightValue]
```

Arguments

`-leftValue leftValue`

Specify left slew value to measure. If not specified, no left measure to be performed.

`-rightValue rightValue`

Specify right slew value to measure. If not specified, no right measure to be performed.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

leftTime=time_on_left_slew_value rightTime=time_on_right_value

Example

```
set waveslew [wvWaveSlew -leftValue 2.35]
```
(the $waveslew is "leftValue=300.0n)

wvZoomValue

Description

Assign the Y value view range.

Syntax

```
wvZoomValue [-win window] [-range upper|lower] [-full wvZoomValue]
[-reset] [-node {("groupName" index)}]
```

Arguments

`-full` *`wvZoomValue`*

Zoom to full view.

`-node {("`*`groupName`*`" `*`index`*`)}`

Zoom the specified node vertically. The node position is specified by the index, a list of positions related to the group name. If this option is not specified, default is selected node(s).

`-range upper|lower`

Zoom area from **upper** to **lower**.

`-reset`

Reset zoom value to original range.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

None.

Example

```
wvZoomValue -full
wvZoomValue -reset
wvZoomValue -win $_nWave1 -node {("G1" 1)} -range 134.357143
66.306122
```

Compare

wvCompareClearResult

Description

Clears all of the comparisons (red hatch marks) from the waveform.

Syntax

```
wvCompareClearResult [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCompareClearResult -win $_nWave1
```

wvCompareDisplayed

Description

Compare currently displayed signals.

Syntax

```
wvCompareDisplayed [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCompareDisplayed
```

wvCompareResult

Description

Save all comparison results.

Syntax

```
wvCompareResult [-win window] [-clear] -file fileName
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`-clear`

Reset all of the content of comparison results.

`-file fileName`

Specified file name to save all kept comparison result up to now.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCompareResult -file compare.out
```

wvCompareSelected

Description

Compare currently selected signals.

Syntax

```
wvCompareSelected [-win window] [-two]
```

Arguments

`-two`

If specified, compare 2 selected signals.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCompareSelected
```

wvCompareSelectedToFile

Description

Compare the selected signals with the specified FSDB file in the same waveform window.

Syntax

```
wvCompareSelectedToFile -win window -file fullFilename
```

Arguments

`-file fullFilename`

Specify the full FSDB file name to compare.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCompareSelectedToFile -win $_nWave2 -file "/verdi/verilog/gate/
gate.fsdb"
```

wvCompareSignalsFromFile

Description

Compare the signals that are listed in the file.

Syntax

```
wvCompareSignalsFromFile [-win window] fileName
```

Arguments

`fileName`

Specify the file name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCompareSignalsFromFile  sig.list
```

wvCompareTwoGroups

Description

Compare each of the signals of two groups.

Syntax

```
wvCompareTwoGroups [-win window] groupName [-secondWin window]
groupName [-hide]
```

Arguments

`-hide`

Specify to hide the *Comparison Result* forms on *nWave*.

`-secondWindow window`

The other window to be compared with.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCompareTwoGroups  G1 G2
wvCompareTwoGroups  G1 G2 -hide
```

wvComparisonOptions

Description

Set the comparison options.

Syntax

```
wvComparisonOptions [-win nWavewinId] [-delim delim] [-time t1 t2]
[-x on|off] [-z on|off] [-maxErrorPerSignal n] [-totalError n]
[-clock1 name rising|falling [-start time] [-width time]]
[-cycle1 time [-delay time] [-width time]][-expression1 name
true|false] [-clock2 name rising|falling [-start time] [-width
time]][-cycle2 time [-delay time] [-width time]]
[-expression2 name true|false][-fullrange] [-tolerance value]
[-casesensitive on|off] [-busValueTolerance value]
```

Arguments

`-busValueTolerance value`

Specify the value for the bus value mismatch tolerance. The value is based on the time units defined by the Window Time Unit in *nWave*.

`-casesensitive on|off`

Turn the case sensitive searching on the specified value *on* or *off*.

`-clock1 name rising|falling`

Specify the clock signal comparison condition for the first signal of the comparison signal pair. The clock signal name and clock edge are necessary to be specified. The **-delay** and **-width** options are optional.

`-clock2 name rising|falling`

Specify the clock signal comparison condition for the second signal of the comparison signal pair. The clock signal name and clock edge are necessary to be specified. The **-start** and **-width** options are optional.

`-cycle1 time`

Specify the cycle information for the first signal of the comparison signal pair. *time* is necessary to be specified. **-delay** and **-width** are optional.

`-cycle2 time`

Specify the cycle information for the second signal of the comparison signal pair. *time* is necessary to be specified. The **-delay** and **-width** options are optional.

`-delay time`

Specify the time offset of the sampling.

`-delim delim`

Specify the delimiter of the specified signal names. The default is '/'.

`expression1 name true|false`

Specify the trigger expression for the first signal of the comparison signal pair. The name of the expression and condition need to be specified.

`-expression2 name true|false`

Specify the trigger expression for the second signal of the comparison signal pair. The name of the expression and condition need to be specified.

`-fullrange`

Span the comparison across the full time range.

`-maxErrorPerSignal n`

Specify the number of errors per signal to stop the comparison.

`-no1`

Logged in the Tcl command file when conditional settings number 1 have not been set.

`-no2`

Logged in the Tcl command file when conditional settings number 2 have not been set.

`-start time`

Specify the time offset of the sampling.

`-time t1 t2`

Specify the time range.

`-tolerance value`

Specify the maximum number for the mismatch tolerance value.

`-totalError n`

Specify the number of total errors to stop the comparison.

`-width time`

Specify the width of the sampling.

`-win window`

Specify the window ID of the invoking *nWave* window.

`-x on|off`

Turn whether to compare the X value *on* or *off*.

`-z on|off`

Turn the Z value comparison *on* or *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetComparisonOption -time 1000 2000 -clock1 clock1
rising-cycle1 50 -expression1 false_cond1 true -clock2 clock2
falling -cycle2 50
-expression2 false_cond1 true
wvComparisonOptions -win $_nWave1 -busValueTolerance 1
```

wvMessageTypes

Description

Select the error types to be displayed.

Syntax

```
wvMessageTypes [-win window] {-cmperr on|off}| -timerr on|off |
-buserr on|off | -highZ on|off | -ustate on|off | -timwarn on|off
| -blkdly on|off | -powdig on|off | -multog on|off | -mulpul
on|off}
```

Arguments

`-blkdly on|off`

Turn the black delay characterization error *on* or *off*.

`-buserr on|off`

Turn the bus contention error *on* or *off*.

`-cmperr on|off`

Turn the comparison error *on* or *off*.

`-highZ on|off`
Turn the high impedance error *on* or *off*.

`-mulpul on|off`
Turn the multiple-pulse error *on* or *off*.

`-multog on|off`
Turn the multiple-toggle error *on* or *off*.

`-powdig on|off`
Turn the power diagnosis *on* or *off*.

`-timerr on|off`
Turn the timing error *on* or *off*.

`-timwarn on|off`
Turn the timing warning *on* or *off*.

`-ustate on|off`
Turn the uncertain state condition *on* or *off*.

`-win` *`window`*
Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvMessageTypes -highZ on
```

wvSelectComparisonErrors

Description

Select the signals that with comparison errors.

Syntax

```
wvSelectComparisonErrors [-win window]
```

Arguments

`-win` *`window`*
Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectComparisonErrors
```

wvSelectMessages

Description

Select the signals with NanoSim's message.

Syntax

```
wvSelectMessages [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectMessages
```

wvSetErrorViewingOptions

Description

Set the error viewing options.

Syntax

```
wvSetErrorViewingOptions [-win window] -dispMark on|off |-dispDesc
on|off
```

Arguments

`-dispMark on|off`

Turn the display of error mark *on* or *off*.

`-dispDesc on|off`

Turn the display of error description *on* or *off*.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetErrorViewingOptions -dispMark on
```

Property Result

wvRptDumpProperty

Description

Save the property results in textual form to the specified file. If you use batch mode (i.e. using tcl commands instead of involving the GUI form), additional parameters for sorting and filtering can be applied.

Syntax

```
wvRptDumpProperty [parameters for wvRptSortProperty and
wvRptFilterProperty] -filename file
```

Arguments

`-filename` *`file`*

Specify the file name of the dumped file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRptDumpProperty -status failure -orderBy Instance -descend \
-filename result.txt
```

wvRptShowResultForm

Description

Show the property result form with all inserted property results.

Syntax

```
wvRptShowResultForm [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRptShowResultForm -win $_nWave2
```

Error

wvSelectErrIndicator

Description

Select the error indicator by time.

Syntax

```
wvSelectErrIndicator [-win windowId] -time errorTime
```

Arguments

`-win window`

Specify the window ID of the invoking nWave window

`-time`

Specify the error occurred time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSelectErrIndicator -win $_nWave2 -time 420
```

wvSearchErrIndicator

Description

Search forward/previous error indicator from the selected error indicator.

Syntax

```
wvSearchErrIndicator [-win windowId] -next | -previous
```

Arguments

`-win window`

Specify the window ID of the invoking nWave window.

`-next window`

Search the forward error indicator from the selected error.

`-previous window`

Search the previous error indicator from the selected error.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSearchErrIndicator -win $_nWave2 -next
```

wvSaveErrIndicatorAsMarker

Description

Save the selected error indicator as a marker in the Marker form.

Syntax

```
wvSaveErrIndicatorAsMarker [-win windowId]
```

Arguments

`-win window`

Specify the window ID of the invoking nWave window.

Value Returned

1 if successful; otherwise, returns 0

Example

```
wvSaveErrIndicatorAsMarker -win $_nWave2
```

Toggle Coverage

wvMultiToggleCoverageReport

Description

Calculate toggle coverage and generate a report for multiple FSDB files.

Syntax

```
wvMultiToggleCoverageReport [-win evtseqwinid] -file fileList
[scope {scopeList}] -timeRange {timeRangeList} -criterion
[Full|Partial|AnyChange] -skipGlitch [TRUE|FALSE] -skipBusDetails
[TRUE|FALSE] -toFile fileName
```

Arguments

`-criterion`

Specify the type of toggle criterion.

`-file` *`fileList`*

Specify the list of FSDB files.

`-scope`

Specify the scope(s) with full hierarchy name for toggle coverage report. If not specified, all signals in all files is calculated. If the wildcard character "*" is appended after a scope (i.e. system/CHILD1/*), it means the current scope and all sub-scopes are calculated.

`-skipGlitch`

When this option is *on* (TRUE), glitches on signals are skipped. When this option is *off* (FALSE), glitches on signals are counted.

`-skipBusDetails`

When this option is *on* (TRUE), members of a bus are not expanded. When this option is *off* (FALSE), members of a bus are expanded.

`-toFile` *`fileName`*

Specify the output file name.

`-timeRange`

Specify the time range for the toggle coverage report.

`-win`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvMultiToggleCoverageReport -win $_nWave1 -file {{file1} {file2}}
-scope {{{/system/CHILD1} {/system/CHILD2}} {{/system/I_cpu/
I_PCU} {/system/I_pram}}} -timeRange{{0,12500} {1000,10000}}
-criterion Full -skipGlitch FALSE -toFile TC1.rpt
wvMultiToggleCoverageReport -win $_nWave1 -file {{file1} {file2}}
-scope {{{/system/CHILD1/*} {/system/CHILD2/*}} {{/system/I_cpu/
I_PCU} {/system/I_pram/*}}} -timeRange{{0,12500} {1000,10000}}
-criterion Full -skipGlitch FALSE -toFile TC1.rpt
wvMultiToggleCoverageReport -win $_nWave1 -file { \
{./verilog.fsdb}{./rtl.fsdb}} \ -scope { {()}{()} } -timeRange {
{0,12500}{0,12500} } -criterion \ Full -skipGlitch FALSE
-skipBusDetails TRUE -toFile \ "./report.rpt"
```

wvToggleCoverageReport

Description

Calculate toggle coverage and generate a report for an FSDB file.

Syntax

```
wvToggleCoverageReport [-win evtseqwinid] [scope {scopeList}]
-from {fromTime} -to {toTime} -criterion [Full|Partial|AnyChange]
-skipGlitch [TRUE|FALSE] -skipBusDetails [TRUE|FALSE] -toFile
fileName
```

Arguments

`-criterion`

The type of toggle criterion.

`-from`

Specify the calculation begin time for the toggle coverage report.

`-scope`

Specify the scope(s) with full hierarchy name for toggle coverage report. If not specified, all signals in all files are calculated.

`-skipGlitch`

When this option is *on* (TRUE), glitches on signals are skipped. When this option is *off* (FALSE), glitches on signals are counted.

`-skipBusDetails`

When this option is *on* (TRUE), members of a bus are not expanded. When this option is *off* (FALSE), members of a bus are expanded.

`-to`

Specify the calculation end time for the toggle coverage report.

`-toFile` *`fileName`*

Specify the output file name.

`-win`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvToggleCoverageReport -win $_nWave1 -scope {{/system/CHILD1} {/
system/CHILD2}} -from 0 -to 12500 -criterion Full -skipGlitch
TRUE
-toFile TC1.rpt
wvToggleCoverageReport -win $_nWave1 -from 0 -to 12500 -criterion
Full \ -skipGlitch FALSE -skipBusDetails TRUE -toFile \
"./report.rpt"
```

Configuration

wvChangeDisplayAttr

Description

Change the display color/linewidth/linestyle of the selected analog signals.

Syntax

```
wvChangeDisplayAttr [-win window] [-h height] [-c color]
[-lw lineWidth] [-ls lineType] [-pointStyle]]
```

Arguments

`-c|-color color`

Specify the color name.

NOTE: Only the **-c** argument gets logged.

`-h|-height height`

Specify the signal height.

NOTE: Only the **-h** argument gets logged.

`-ls|-lineStyle lineType`

Specify the line style.

NOTE: Only the **-ls** argument gets logged.

`-lw|-lineWidth lineWidth`

Specify the line width.

NOTE: Only the **-lw** argument gets logged.

`-pointStyle`

Draw the analog signal with the point style. This option only affects the newly added signals.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvChangeDisplayAttr -win $_nWave1 -color red4
```

wvDisplayRuler

Description

An option of show/hide the ruler in the bottom of the *nWave* window.

Syntax

```
wvDisplayRuler [-win window] -bottom on|off
```

Arguments

`-bottom on|off`

Specify whether to show or hide the ruler in the bottom of waveform window.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDisplayRuler -win $_nWave1 -bottom off
```

wvGetFileTimeRange

Description

Get time range of the active file.

Syntax

```
wvGetFileTimeRange [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Min/max time range in the active file.

Example

```
wvGetFileTimeRange
```

wvGetDisplaySignals

Description

Get signal list in signal's window.

Syntax

```
wvGetDisplaySignals [-win window] [-all] [-group listGroup]
```

Arguments

`-all`

Get all signals in signal window. The default only gets the signals on the screen.

`-group listGroup`

If specified, return the display signals of the specified group(s).

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Signal list.

Example

```
wvGetDisplaySignals -group {GA} {GB}
```

wvGetDeltaY

Description

Get delta Y value of analog signal.

Syntax

```
wvGetDeltaY [-win window] [-delim delim] signalName [-range pos1 pos2]
```

Arguments

`-delim delim`

Specify the delimiter of the specified signal name. The default is '/'.

`-range pos1 pos2`

Specify the time range. The default is cursor/marker position.

`signalName`

Specify the analog signal name.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGetDeltaY  {system.i_cpu.anasig1} -delim '.'
```

wvGetFileTimeUnit

Description

Get file time unit from current active file.

Syntax

```
wvGetFileTimeUnit [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Time unit (in *G*, *K*, *M*, *m*, *u*, *n*, *p*, or *f* seconds, such as nanosecond or picosecond) if successful; otherwise, returns 0.

Example

```
wvGetFileTimeUnit -win $_nWave1
1ns
```

wvGetSigValueByTime

Description

Get signal's value at a given time.

Syntax

```
wvGetSigValueByTime [-win window] [-delim delim] signalName
[-format format] [-lz] [-time time]
```

Arguments

`-delim delim`

Specify the delimiter of the specified signal name. The default is '/'.-format *format*

Signal value format: Bin | Oct | Hex | Ascii | UDec | 2Com | 1Com | SMag

`-lz`

Display the leading zeros.

`signalName`

Specify the signal name to be queried.

`-time time`

Time (based on the current window time unit). The default is cursor time.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

Value of the signal.

Example

```
wvGetSigValueByTime {system/i_cpu/clock} -format Bin
```

wvGetWindowTimeUnit

Description

Get window time unit.

Syntax

```
wvGetWindowTimeUnit [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

Time unit (in *G*, *K*, *M*, *m*, *u*, *n*, *p*, or *f* seconds, such as nanosecond or picosecond) if successful; otherwise, returns 0.

Example

```
wvGetWindowTimeUnit -win $_nWave1 2ps
```

wvSetDblClkActiveTrace

Description

Enable/disable active trace when double-clicking on the waveform pane.

Syntax

```
wvSetDblClkActiveTrace -win $windowId on|off
```

Arguments

`on|off`

Specify whether to enable (on) or disable (off) active trace by double-click in the waveform pane.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window. If not specified, current window Id is set.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetDblClkActiveTrace -win $_nWave2 off
wvSetDblClkActiveTrace off
```

wvSetDefaultValue

Description

Set the default values.

Syntax

```
wvSetDefaultValue [-win window] [-digitalHeight height]
[-analogHeight height] [-minErrorLen length] [-deli delim]
[-analogFormat Auto | Scientific | Engineer] [-conHierDeli delim]
[-analogPrecision digits] [-scroll scrollPercentage]
[-radix Bin | Oct | Hex | Dec | ASCII | Float] [-zoom_ratio value]
[-dragZoomTolerance value] [-windowTimeUnit scale]
[-groupRuleFile fileName] [-maxTransExpandedLayer]
```

Arguments

`-analogFormat Auto | Scientific | Engineer`

Specify the format of the analog signal. The available formats are: **Auto**, **Scientific**, and **Engineer**.

`-analogHeight` *`height`*

Specify the height of the analog signal.

`-analogPrecision` *`digits`*

Specify the analog precision and scale.

`-conHeirDeli` *`delim`*

Convert hierarchical delimiter.

`-deli` *`delim`*

Specify the delim of the loaded FSDB file.

`-digitalHeight` *`height`*

Specify the height of digital signal.

`-dragZoomTolerance` *`value`*

Specify the waveform window drag zoom tolerance.

`-groupRuleFile` *`fileName`*

Specify the path for forming bus rule file.

`-maxTransExpandedLayer`

Specify the maximum number of layers that are expanded for overlapped transactions.

`-minErrorLen` *`length`*

Specify the minimum length of the error display.

`-radix Bin | Oct | Hex | Dec | ASCII | Float`

Specify the default radix format for bus signal value. The available formats are: **Bin**, **Oct**, **Hex**, **Dec**, **ASCII**, and **Float**.

`-scroll` *`scrollPercentage`*

Specify the scroll percentage when pan by the arrow key.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-windowTimeUnit` *`scale`*

Specify the window time unit.

`-zoom_ratio` *`value`*

Change the initial zoom ratio of the waveform window and set the percentage by 100%, 50%, or 10% respectively.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetDefaultValue -digitalHeight 20
wvSetDefaultValue -scroll 20
wvSetDefaultValue -maxTransExpandedLayer 10
```

wvSetDisplayAttr

Description

Set the display attributes on the *nWave* window.

Syntax

```
wvSetDisplayAttr layerName -color color -linewidth width
-linestyle style -stipple stipple | {windowName -color color -font
font}
```

windowName: *legendWindow* | *valueWindow* | *curveWindow* | *groupName* | *rulerValue* | *analogRulerValue* | *commentString*

Arguments

`layerName`

Specify the layer name. The layer name is available in the *wave.rc* resource file.

`-color color`

Specify the color.

`-linewidth width`

Specify the line width.

`-stipple stipple`

Specify the stipple.

`-font font`

Specify the font for the specified type.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetDisplayAttr computedAnnotationColor -color ID_PURPLE6
```

wvSetFileTimeScale

Description

Set file time scale.

Syntax

```
wvSetFileTimeScale [-win window] time unit
```

Arguments

`time`

Time scale.

`unit`

Time unit.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetFileTimeScale 100 ns
```

wvSetOptions

Description

Set options in the *nWave* window.

Syntax

```
wvSetOptions [-win window] [-hierName on|off] [-signalType on|off]
[-snap on|off] [-leadingZeros on|off] [-fixedDelta on|off]
[-autoUpdate on|off] [-deltaY on|off] [-cursorCenter on|off]
[-filesTimeRange on|off] [-autoFit on|off] [-autoColorPattern
on|off] [-dispDelta on|off] [-dispGlitch on|off] [-caseinsensitive
on|off] [-isoMask on|off] [-hiSig on|off] [-drivingPowerOffMask
on|off] [-retMask on|off] [-regionMode on|off]
```

Arguments

`-autoColorPattern on|off`

Overlay signal with a distinctive layer assigned to each analog signal.

`-autoFit on|off`

Overlay signal is stretched to occupy the whole window.

`-autoUpdate on|off`

Automatically update waveform periodically.

`-caseinsensitive on|off`

Match signal name without case-sensitivity check.

`-cursorCenter on|off`

Keep cursor in the center.

`-deltaY on|off`

Turn the display of all the analog signals' cursor/marker value in the value window *on* or *off*.

`-dispDelta on|off`

Turn the display of the cursor/marker delta value on the toolbar *on* or *off*.

`-dispGlitch on|off`

Turn the display of the glitch *on* or *off*.

`-drivingPowerOffMask on|off`

Turn the display of the driving power off range mask in the waveform pane *on* or *off*.

`-filesTimeRange on|off`

Turn the display of all files' union time range *on* or *off*.

`-fixedDelta on|off`

Fix the delta time between the cursor and marker.

`-hierName on|off`

Turn the display of the hierarchical name *on* or *off*.

`-hiSig on|off`

Turn the highlight of the waveform of a selected signal *on* or *off*.

`-isoMask on|off`

Turn the display of the Isolation range mask in the waveform pane *on* or *off*.

`-leadingZeros on|off`

Turn the display of leading zeros for signal values *on* or *off*.

`-regionMode on|off`

Expand in region mode or sequence mode. *on* is in region mode. *off* is in sequence mode. The default is *off*.

`-retMask on|off`

Turn the display of the Retention range mask in the waveform pane *on* or *off*.

`-signalType on|off`

Turn the display of the signal type *on* or *off*.

`-snap on|off`

Snap mouse click to the next value change.

`-win` *window*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetOptions -dispGlitch on
wvSetOptions -win $_nWave2 -isoMask on
wvSetOptions -win $_nWave2 -hiSig off
```

wvSetPreference

Description

Set global reference in the *nWave* window.

Syntax

```
wvSetPreference [-win window][-leftalign on|off] [-showdelta
on|off] [-deselect on|off] [-IncHSPICE on|off] [-overwrite on|off]
[-getAllSignal on|off] [-gravitysnap on|off] [-dispTran on|off]
[-dispPackedSignalAsBus on|off]
[-dispRadix on|off][-labelMarker on|off] [-resolveSymlk on|off]
[-syncBrowserDir on|off] [-dispInOut on|off] [-dispNetReg on|off]
[-dispVerilogVHDL on|off] [-dispModule on|off] [-traceX on|off]
[-hiSig on|off][-dispFile on|off] [-signalType on[text|icon]|off]
[-stickMarker on|off] [-dispSpace on|off] [-busSize on|off]
[-reloadActive on|off] [-valueBeforeCursor on|off]
[-restoreFromActiveF on|off][-normalize on|off]
[-convertInc on|off] [-hierName on|off]
[-fittedTransHeight on|off] [-hierScopeName on|off]
[-inst_array on|off] [-vhdlnaming on|off] [-orgVar on|off]
[-varUL upper|lower] [-overlap on|off] [-tipInSignalWin on|off]
[-tipInCurveWin on|off] [-keepMarkerAtEndTimeOfTrans on|off]
[-displayLSBsFirst on|off] [-paintSpecificColorPattern on|off]
[-caseinsensitive on|off] [-vhdlvariable on|off]
[-signalvscroll on|off] [-hier_signal_name on|off]
[-partial_hier on|off] [-partial_head nHead] [-partial_tail nTail]
[-value_cursor_marker on|off] [-leadingzero on|off] [-cursorcenter
on|off] [-orgScope on|off] [-displayglitch on|off] [-formAllMem
on|off] [-snapcursor on|off] [-extendLastTick on|off]
[-showPortForInst on|off] [-readValueChangeOnDemandForDrawing
on|off] [-showWndCtntWhileResizing on|off] [-pointStyle on|off]
[-displayAttrNam on|off] [-expandOverlappedTrans on|off]
[-truncateFilePath on|off] [-displayTransBySpecificColor on|off]
[-dispFileName on|off] [-dispFileNumber on|off]
```

```
[-verilogStrengthDispType type] [-expandOverlappedTrans on|off]
[-formBusFromPartSelects on|off] [-filterPropVacuousSuccess
on|off][-reloadCurWin on|off] [-msg_disp_callstack on|off]
[-msg_disp_full on|off] [-msg_disp_label_only on|off]
[-msg_severity_color on|off][-limit_msg_width on|off]
[-msg_max_width value] [-auto_save {on|off interval directory
fileName}] [-dispSamplePoint on|off] [-msg_adjust_width_by_label
on|off] [-retMask on|off] [-denseBlockDrawing on|off]
[-loadScopesOnDemand on|off [num]] [-syncCursorMarker on|off]
[-syncHorizontalRange on|off] [-syncVerticalScroll on|off]
```

Arguments

`-auto_save {on|off interval directory fileName}`

When *on* is specified, automatically save the signals at the specified time interval. The default is *off*.

`-busSize on|off`

When *on* is specified, fit the analog bus by the bus size. The default is *off*.

`-converInc on|off`

When *on* is specified, convert the VCD in the increasing mode. The default is *off*.

`-denseBlockDrawing on|off`

When *on* is specified, display the dense block drawing signals for fast drawing of VHDL records in the waveform pane. The default is *on*.

`-deselect on|off`

When *on* is specified, deselect the selected signal. The default is *off*.

`-dispFile on|off`

When *on* is specified, display the file name before the signal in the signal pane if more than one file are loaded. The default is *off*.

`-dispInOut on|off`

When *on* is specified, display the input/output signal in different color. The default is *off*.

`-displayglitch on|off`

When *on* is specified, display the glitch. The default is *off*.

`-displayLSBFirst on|off`

When *on* is specified, display the LSB value first with limited space. The default is *off*.

`-dispModule on|off`

When *on* is specified, display the module name in the *Get Signal* form.

`-dispPackedSignalAsBus on|off`

When *on* is specified, display packed signals in bitwise manner.

`-dispRadix on|off`

When *on* is specified, display the radix in the value window. The default is *off*.

`-dispSamplePoint on|off`

When *on* is specified, display the sample point for the attribute signals. The default is *on*.

`-dispSpace on|off`

When *on* is specified, display the '_' for each 4 digits of the signal value. The default is *off*.

`-dispTran on|off`

When *on* is specified, display the transition in the value window. The default is *off*.

`-expandOverlappedTrans`

When *on* is specified, expand the overlapped transactions. When *off* is specified, shrink the expanded transactions. The default is *off*.

`-extendLastTick on|off`

When *on* is specified, extend the last time tick. The default is *off*.

`-filterPropVacuousSuccess on|off`

When *on* is specified, filter the threads with value string "vacuous-success" and "vacuous-match" on the *nWave* window. The default is *off*.

`-formAllMem on|off`

When *on* is specified, form all the memory signals when opening the FSDB file. The default is *off*.

`-formBusFromPartSelects on|off`

When *on* is specified, reconstruct and display the entire bus if the part selects of a bus are dumped to the FSDB file. The default is *off*.

`-getAllSignal on|off`

When *on* is specified, confirm to get all signals. The default is *on*.

`-gravitysnap on|off`

When this option is turned *on*, and the **nWave -> Waveform -> Snap Cursor/Marker to Transitions** command is turned *on*, if the cursor/marker is close enough to the transition, snap to the transition occurs. If not, the cursor/marker is placed at the position where the mouse is pointing. When this option is turned *off*, and the **nWave -> Waveform -> Snap Cursor/**

Marker to Transitions command is turned *on*, the cursor or marker always automatically snaps to the nearest value change position. The default is *off*.

`-hier_signal_name on|off`

When *on* is specified, show the full hierarchical signal name. The default is *off*.

`-incHSPICE on|off`

When *on* is specified, open an HSPICE file incrementally. The default is *on*.

`-labelMarker on|off`

When *on* is specified, label the marker. The default is *on*.

`-leftAlign on|off`

When *on* is specified, set the left alignment. The default is *off*.

`-limit_msg_width on|off`

When *on* is specified, limit the width of the messages to the specified characters. The default is *off*.

`-loadScopesOnDemand on|off [num]`

When enabled, it loads the FSDB scope on demand to improve the performance of opening the FSDB file when the opened file has more than 'num' millions scopes.

`-msg_adjust_width_by_label on|off`

When *on* is specified, adjust the width of messages to the width of the label name. The default is *off*.

`-msg_disp_callstack on|off`

When *on* is specified, display the call stack with full sections or last section. The default is *off*.

`-msg_disp_full on|off`

When *on* is specified, display the call stack with full sections. When *off* is specified, display the call stack with last section.

`-msg_disp_label_only on|off`

When *on* is specified, display the transaction/message labels. The default is *off*.

`-msg_max_width` *`value`*

Specify the value of the maximum width of the messages.

`-msg_severity_color on|off`

When *on* is specified, apply the predefined severity color to the newly added message streams and the message streams that have not been applied with the user-defined color. The default is *off*.

`-orgVar on|off`

When *on* is specified, keep the original variable name. The default is *off*.

`-overwrite on|off`

When *on* is specified, confirm to overwrite. The default is *on*.

`-paintSpecificColorPattern on|off`

When *on* is specified, paint the waveform with specified color/pattern. The default is *off*.

`-partial_head` *`nHead`*

Specify the number of scopes from the beginning of a hierarchical signal name to show in the signal pane.

`-partial_hier on|off`

When *on* is specified, show the partial signal name. The default is *off*.

NOTE: The **-hier_signal_name** option must be turned on to enable the **-partial_hier on|off** option.

`-partial_tail` *`nTail`*

Specify the number of scopes from the end of a hierarchical signal name to show in the signal pane.

`-piecewiseLinear on|off`

When *on* is specified, display the piecewise waveform for analog signal. The default is *off*.

`-pointStyle on|off`

When *on* is specified, draw the analog signal in point style. The default is *off*.

`-readValueChangeOnDemandForDrawing on|off`

When *on* is specified, load the signal value change on demand. The default is *off*.

`-reloadActive on|off`

When *on* is specified, reload the active file of the invoking window only. The default is *off*.

`-reloadCurWin on|off`

When *on* is specified, only the files in the current *nWave* window are reloaded. The default is *off*.

`-resolveSymlk on|off`

When *on* is specified, resolve the symbolic link when opening FSDB file. The default is *on*.

`-restoreFromActiveFile on|off`

When *on* is specified, restore signals from the current active file. The default is *on*.

`-retMask on|off`

When *on* is specified, mask the retention range for HDL signals according to the Retention condition applied. The default is *off*.

`-showDelta on|off`

When *on* is specified, show the delta time value. The default is *on*.

`-showWndCtntWhileResizing on|off`

When *on* is specified, show the window content during when the *nWave* window is resized. The default is *off*.

`-signalType on[text|icon]|off`

When *on* is specified, set to display the signal type in text or icon in the signal pane. The default is *off*.

`-stickMarker on|off`

When this option is turned *on*, the preference option S**tick Cursor/Marker on Waveform** preference option in the **Waveform -> View Options -> Waveform Pane - > General** page is enabled. When this option is turned *off*, the preference option S**tick Cursor/Marker on Waveform** preference option in the **Waveform -> View Options -> Waveform Pane - > General** page is disabled

`-syncBrowserDir on|off`

When *on* is specified, the **Synchronize File Browser Directory** option in the **General -> Configure GUI** page is enabled. When this option is turned *on*, the file browser directory is synchronized to the previously opened file browser directory. When this option is turned *off*, the file browser directory in each form operates independently. The default is *on*.

`-syncCursorMarker`

When *on* is specified, it automatically invokes the **Window -> Sync All Waveforms by -> Cursor/Marker** command for the new *nWave* window. The default for this option is *off*.

`-syncHorizontalRange`

When *on* is specified, it automatically invokes the **Window -> Sync All Waveforms by -> Horizontal Range** command for the new *nWave* window. The default for this option is *off*.

`-syncVerticalScroll`

When *on* is specified, it automatically invokes the **Window -> Sync All Waveforms by -> Vertical Scroll** command for the new *nWave* window. The default for this option is *off*.

`-truncateFilePath`

When *on* is specified, truncate the file path in the window title. The default is *on*.

`-valueBeforeCursor on|off`

When *on* is specified, display the value before the cursor time. The default is *off*.

`-varUL upper|lower`

When *on* is specified, set to use uppercase or lower case for the variable name. The default is *off*.

`-verilogStrengthDispType type`

Specify the type by that to display the Verilog strength information. The types are: **Type1** and **Type2**.

`-vhdlnaming on|off`

When *on* is specified, set the VHDL naming style in conversion. The default is *off*.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetPreference -dispSpace on
wvSetPreference -expandOverlappedTrans on
wvSetPreference -formBusFromPartSelects on
wvSetPreference -hier_signal_name on -partial_hier on
-partial_head 1 -partial_tail 1
wvSetPreference -msg_disp_callstack on -msg_disp_full off
wvSetPreference -msg_disp_label_only on -limit_msg_width on
wvSetPreference -auto_save {on 10 "/verdi/home/sylvia_wei/temp"
"novas_autosave_sig2"}
wvSetPreference -dispSamplePoint off -msg_disp_label_only on
```

```
wvSetPreference -loadScopesOnDemand on 5
wvSetPreference -loadScopesOnDemand off
wvSetPreference -win $_nWave2 -syncCursorMarker on
-syncVerticalScroll off
```

wvSetSpacing

Description

Set space between signals.

Syntax

`wvSetSpacing [-win window] space`

Arguments

`space`

Specify spacing between signals.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetSpacing 6
```

wvSetWindowTimeUnit

Description

Set window time unit.

Syntax

`wvSetWindowTimeUnit [-win window] time unit`

Arguments

`time`

Time scale.

`unit`

Time unit.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSetWindowTimeUnit 0.01 ns
```

wvSwitchDisplayAttr

Description

Automatically switch the display color/linewidth/linestyle of the selected signals.

Syntax

```
wvSwitchDisplayAttr [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvSwitchDisplayAttr -win $_nWave1
```

Print

wvCapture

Description

Capture current window to an image file in PNG format.

Syntax

```
wvCapture [-win window] [-footer footer] [-region region] -file
fileName
```

Arguments

`-footer footer`

Specify the message to be shown on the footer.

`-region region`

Specify the region to be printed or saved.

`-file fileName`

Specify file name to save captured image.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCapture -footer "CPU" -file "/home/cpu.png"
```

wvPrint

Description

Print the *nWave* window content.

Syntax

```
wvPrint [-win window] [-header header] [-footer footer]
[-value on|off] [-legend on|off] [-inclinewidth
on|off][-stretchsignal on|off] [-signal signalOption]
[-selected][-start startTime] [-end endTime][-slice pageTime]
[-bbground on|off][-orient orientation]
[-paper papersize][-printallpages on|off][-bypaperlayout
on|off][-pageFrom pageNum][-pageTo pageNum][-color
on|off]{(-printer printerName) | (-file printFile)}[-fittopage
on|off]
```

Arguments

`-bbground on|off`

When set to *on*, the background color is black or gray depending on whether the **-color** option is set to *on* or *off*. When set to *off*, the background color is white. The default it *off*.

- When this option and the **-color** options are set to *on* simultaneously, the background is black.
- When this option is set to on but the **-color** option is set to *off*, the background is gray.

`-bypaperlayout on|off`

When set to *on*, print the signal fitting the paper layout. When set to *off*, print the signal ignoring the paper layout. The default it *off*.

NOTE: This option is enabled after one signal option is specified from the select signal list. The signal options are **All Signals (Time First)**, **All Signals (Signal First)**, and **Selected Signals**.

`-color on|off`

When set to *on*, print multiple colors on a color printer. When set to *off*, print in gray-scale. The default it *off*.

`-end endTime`

Specify the end time to print.

`-file printFile`

Specify the name that *nWave* prints the window content to.

`-fittopage on|off`

When set to *on*, fit the print range to the paper size. When set to *off*, print the signal without fitting the print range to the paper size. The default is *off*.

`-footer footer`

Specify the information to be presented on the bottom of the printout.

`-header` *`header`*

Specify the information to be presented on the top of the printout.

`-inclinewidth on|off`

When set to *on*, increase the signal line width by 1 pt. When set to *off*, keep the original signal line width. The default is *off*.

`-legend on|off`

When set to *on*, print the signal style legend. When set to *off*, do not print the legend. The default is *on*.

`-orient landscape|portrait`

Specify the orientation of the paper. The options are **landscape** and **portrait**.

`-pageFrom` *`pageNum`*

Specify the start page to print.

`-pageTo` *`pageNum`*

Specify the end page to print.

`-paper` *`papersize`*

Specify the paper size for the printed page. The size options are **Letter**, **Legal**, **A4**, **A3**, **A2**, **A1**, **A0**, **B**, **C**, **D**, **E**, and **User-defined**.

`-printer` *`printerName`*

Specify the printer *nWave* sends the window content to. The default is **lp**.

`-printallpages on|off`

When set to *on*, print the entire file. When set to *off*, print the specified pages. The default is *on*.

`-selected`

Print only the selected signals.

`-signal` *`signalOption`*

Print the signals specified by the select signal option. The signal options are **Displayed Signals**, **All Signals (Time First)**, **All Signals (Signal First)**, and **Selected Signals**.

`-slice` *`pageTime`*

Specify the time range for each page.

`-start` *`startTime`*

Specify the start time to print.

`-stretchsignal on|off`

When set to *on*, stretch the signal. When set to *off*, do not stretch the signal. The default is *off*.

`-value on|off`

When set to *on*, print the signal value. When set to *off*, do not print the signal value. The default is *off*.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvPrint -paper A4 -color false -printer
```

See Also

nWave, **File -> Print**

Access

wvGetActiveFile

Description

Get the file handler to access the signal/scope information.

Syntax

```
wvGetActiveFile [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

FSDB file handler if successful; otherwise, returns 0.

Example

```
set fhdl [wvGetActiveFile]
```

wvGetActiveFileName

Description

Get the active file name.

Syntax

```
wvGetActiveFileName [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

Active FSDB file name if exists, empty returned otherwise.

Example

```
wvGetActiveFileName -win $_nWave1
```

wvIterChildScope

Description

Get the child scope handler to access child scope under the specified top scope.

Syntax

```
wvIterChildScope -file fsdbfileHandler -delim delim topScopename
```

Arguments

`-file fsdbfileHandler`

FSDB file handler.

`-delim delim`

Hierarchy delimiter of the scope name.

`topScopename`

Name of specified top scope.

Value Returned

Scope handler if successful; otherwise, returns 0.

Example

```
set scopehdl [wvIterChildScope -file $fhdl $topscope]
```

wvIterScopeSignal

Description

Get the signal handler to access the signals under the specified scope.

Syntax

```
wvIterScopeSignal -file fsdbfileHandler -delim delim scope
```

Arguments

`-file fsdbfileHandler`

FSDB file handler.

`-delim` *`delim`*

Hierarchy delimiter of the scope name.

`scope`

Scope name.

Value Returned

Signal handler if successful; otherwise, returns 0.

Example

```
set signalhdll [wvIterScopeSignal -file $fhdl -delim / $onescope]
```

wvIterTopScope

Description

Get top scope handler to access the top scopes.

Syntax

`wvIterTopScope` *`fsdbfileHandler`*

Arguments

`fsdbfileHandler`

The FSDB file handler returned by the **wvGetActiveFile** command.

Value Returned

Top scope handler if successful; otherwise, returns 0.

Example

```
set tophdl [wvIterTopScope $fhdl]
```

wvScopeIterNext

Description

Get next child scope from the handler.

Syntax

```
wvScopeIterNext scopeHandler
```

Arguments

`scopeHandler`

Scope handler returned from *wvIterChildScope*.

Value Returned

Scope if successful; otherwise, returns 0.

Example

```
set onescope [wvScopeIterNext $scope_handler]
```

wvSignalIterNext

Description

Get next signal from the signal handler.

Syntax

```
wvSignalIterNext signalHandler
```

Arguments

`signalHandler`

Signal handler returned by the **wvIterScopeSignal** command.

Value Returned

Signal if successful; otherwise, returns 0.

Example

```
set onesignal [wvSignalIterNext $signalhdl]
```

wvTopScopeIterNext

Description

Get next top scope.

Syntax

```
wvTopScopeIterNext topscopeHandler
```

Arguments

`topscopeHandler`

The top scope iteration handler returned from **wvIterTopScope**.

Value Returned

Top scope if successful; otherwise, returns 0.

Example

```
set topscope [wvTopScopeIterNext]
```

Get Signal

wvGetSelectedSignals

Description

Get selected signals in the *nWave* window.

Syntax

```
wvGetSelectedSignals [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

A list of selected signal names if successful; otherwise, returns 0.

Example

```
wvGetSelectedSignals -win $_nWave1
```

wvGetSignalClose

Description

Close the *Get Signal* form.

Syntax

```
wvGetSignalClose [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGetSignalClose -win $nwave1
```

wvGetSignalDumpAU

Description

Dump the signal data in the *nWave* window.

Syntax

```
wvGetSignalDumpAU [-win window] [-signal|-tree|-subscope]
```

Arguments

`-signal|-tree|-subscope`

Specify what you want to dump from one or more of the following:

- **-signal:** Dump all found signals in the signal box.
- **-tree:** Dump the hierarchical scope tree.
- **-subscope:** Dump all found sub-scopes in the scope box.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGetSignalDumpAU -win $_nWave1 -signal -tree -subscope
```

wvGetSignalOpen

Description

Open the *Get Signal* form.

Syntax

```
wvGetSignalOpen [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGetSignalOpen -win $nwave1
```

wvGetSignalSetOptions

Description

Change the options of the *Get Signal* form.

Syntax

```
wvGetSignalSetOptions [-win window] [-all on|off] | [-output
on|off] | [-input on|off] | [-inout on|off] | [-buffer on|off] |
[-linkage on|off]) [-net on|off] | [-register on|off] |
[-transaction on|off] | [-property on|off] | [-retention on|off] |
[-isolation on|off] | [-level_shifted on|off] | [-boundary_port
on|off] | [-matchcase on|off] | [-allscope on|off] |
[-searchByRegExpr on|off] | [-order on|off] | [-subprogram on|off]
| [-dragfollowcol on|off] | [-hierarchybox on|off] | [-subscopebox
on|off] | [-searchscope on|off] | [-searchsignal on|off] |
[-checksignal on|off] | [-checksignalnum integer] |
[-filtersynthinstance on|off [-filterOutInstance on|off]] |
[-synthinstanceprefix Prefix1 [Prefix2, Prefix3...]] |
[-filtersynthnet on|off [-filterOutNet on|off]] |
[-synthnetprefix Prefix1 [Prefix2, Prefix3...]] | [-overlap
on|off] | [-collapsesubtree on|off] | [-rowMajor on|off]
```

Arguments

`-all on|off`

List all the signals.

`-allscope on|off`

Search all the scopes for **Find Signal** field.

`-boundary_port on|off`

List the Boundary Port signals in power designs.

`-buffer on|off`

List the buffer signals, VHDL only.

`-checksignal on|off`

Turn the **Ask If Signals Over** option *on* or *off*.

`-checksignalnum` *`integer`*

Enter an integer to be set as the maximum signals searched.

`-collapsesubtree on|off`

Turn the **Collapse All Sub-tree Nodes When Parent Node is Collapsed option** *on* or *off*.

`-dragfollowcol on|off`

Turn the **Drag Selection Follow Column** option *on* or *off*.

`-filterOutInstance on|off`

When specified *on*, filter out the scopes with prefix. When specified *off*, find the scopes with prefix. The default is *on*.

`-filterOutNet on|off`

When specified *on*, filter out the synthesized nets with prefix. When specified *off*, find the synthesized nets with prefix. The default is *on*.

`-filtersynthinstance on|off`

Turn the **Filter Scope(s) with Prefix** option *on* or *off*.

`-filtersynthnet on|off` *`[-filterOutNet on|off]`*

Turn the **Filter Synthesized Nets with Prefix** option *on* or *off*.

`-hierarchybox on|off`

Turn the **Display Design Hierarchy Box** option *on* or *off*.

`-inout on|off`

List the inout signals.

`-input on|off`

List the input signals.

`-isolation on|off`

List the Isolation signals in the power designs.

`-level_shifted on|off`

List the Level-shifted signals in power designs.

`-linkage on|off`

List the linkage signals, VHDL only.

`-matchcase on|off`

Turn the match case option for **Find Signal** field on *or off*.

`-net on|off`

List the net data type signals.

`-order on|off`

Display the signals in the selection order .

`-output on|off`

List the output signals.

`-overlap on|off`

Turn the **Overlap Added Analog Signals** option *on* or *off*.

`-property on|off`

List the property signals.

`-register on|off`

List the register data type signals.

`-retention on|off`

List the Retention signals in power designs.

`-rowMajor on|off`

Specify *on* or *off* to display signals in the Signal List pane of the *Get Signals* form in a row or column order.

`-searchscope on|off`

Turn the **Search Scopes Dynamically** option *on* or *off*.

`-searchsignal on|off`

Turn the **Search Signals Dynamically** option *on* or *off*.

`-subscopebox on|off`

Turn the **Display Subscope Box** option *on* or *off*.

`-subprogram on|off`

Turn the **Display Sub-programs** option *on* or *off*.

`-synthinstanceprefix` *`Prefix1 [Prefix2, Prefix3...]`*

When **-filtersynthinstance** is specified, the filter scope(s)/instance(s) based on the specified prefix(es). Multiple prefixes can be specified by using a comma or space as the separator.

`-synthnetprefix` *`Prefix1 [Prefix2, Prefix3...]`*

When **-filtersynthnet** is specified, the filter synthesized nets based on the specified prefix(es). Multiple prefixes can be specified by using a comma or space as the separator.

`-transaction on|off`

List the transaction signals.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGetSignalSetOptions -inout -matchcase
wvGetSignalSetOptions -win $_nWave1 -searchByRegExpr on
wvGetSignalSetOptions -win $_nWave2 -filtersynthinstance on
wvGetSignalSetOptions -win $_nWave3 -synthinstanceprefix "u23,
u25"
wvGetSignalSetOptions -win $_nWave2 -filtersynthnet on
wvGetSignalSetOptions -win $_nWave2 -synthnetprefix "n17 n19"
wvGetSignalSetOptions -win $_nWave2 -property on -transaction on
wvGetSignalSetOptions -win $_nWave2 -isolation on -retention on
-level_shifted off
```

wvGetSignalSetSignalFilter

Description

Set a filter in **Find Signal** to skip the specific characters.

Syntax

```
wvGetSignalSetSignalFilter [-win window] filter
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvGetSignalSetSignalFilter -win $_nWave1 "I\(*"
wvGetSignalDumpAU -win $_nWave1 -signal
wvGetSignalSetSignalFilter -win $_nWave1 “pipe*”
wvGetSignalSetSignalFilter -win $_nWave1 “mda*\[2\]\[**\]”
```

Use this command and extent command **wvGetSignalDumpAU** to dump current signal list with setting filter.

Power

wvAddIsoControl

Description

Add signals that are applied with the same Isolation command to *nWave*.

Syntax

```
wvAddIsoControl -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddIsoControl -win $_nWave2 -signal "/system/i_pram/R_W"
```

wvAddPowerMode

Description

Add the specified power domain signal(s) to *nWave*.

Syntax

```
wvAddPowerMode -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddPowerMode -win $_nWave2 -signal "/system/i_cpu/i_CCU/
mux_sel\[2:0\]"
```

wvAddPowerSignals

Description

Add signals that are applied with the same Isolation/Level-shifted command to *nWave*.

Syntax

```
wvAddPowerSignals -win window -iso|-lvs -signal fullName
```

Arguments

`-iso|-lvs`

Specify the signal type as Isolation or Level-shifted signals.

`-signal fullName`

Specify the full name of the signal.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddPowerSignals -win $_nWave2 -iso -signal {/system/i_pram/VMA}
```

wvAddIsoInstru

Description

Add the instrumented Isolation signals to *nWave*.

Syntax

```
wvAddIsoInstru -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal to add its instrumented signals.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddIsoInstru -win $_nWave3 -signal "/system/receiver/
in1\[7:0\]"
```

wvAddRetControls

Description

Add the control signals for the specified Retention signal to *nWave*.

Syntax

```
wvAddRetControls -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddRetControls -win $_nWave2 -signal "/system/i_cpu/i_CCU/
bus_mode\[2:0\]"
```

wvAddRetInstru

Description

Add the instrumented Retention signals to *nWave*.

Syntax

```
wvAddRetInstru -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal to add its instrumented signals.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddRetInstru -win $_nWave3 -signal "/system/i_cpu/i_CCU/
bus_mode\[2:0\]"
```

wvAddSPAInstru

Description

Add the instrumented SPA signals to *nWave*.

Syntax

```
wvAddSPAInstru -win window -signal fullName
```

Arguments

`-signal` *`fullName`*

Specify the full name of the signal to add its instrumented signals.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddSPAInstru -win $_nWave3 -signal "/tb/top/i_sub1/i"
```

wvAddSRSNInstru

Description

Add the instrumented SRSN signals to *nWave*.

Syntax

`wvAddSRSNInstru -win` *`window`* `-signal` *`fullName`*

Arguments

`-signal` *`fullName`*

Specify the full name of the signal to add its instrumented signals.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddSRSNInstru -win $_nWave3 -signal "/tb/top/i_sub1/i"
```

wvAddSupplyNet

Description

Add the supply net for the specified signal to *nWave*.

Syntax

```
wvAddSupplyNet [-win windowId] -signal fullName
[-all|-iso|-ret|-srsn|-spa]
```

Arguments

`-all`
Add the supply net for all affected types. This is the default option.

`-iso`
Add the supply net for Isolation rules.

`-ret`
Add the supply net for Retention rules.

`-signal fullName`
Specify the signal to add the supply net.

`-spa`
Add the supply net for SPA rules.

`-srsn`
Add the supply net for SRSN rules.

`-win window`
Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddSupplyNet -win $_nWave2 -signal "/system/a" -all
```

wvFindIsoCmd

Description

Locate the Isolation command that applies to the Isolation signal in the power source code pane.

Syntax

```
wvFindIsoCmd -win window –signal fullName
```

Arguments

`–signal fullName`

Specify the signal to find the corresponding Isolation command.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvFindIsoCmd –win $_nWave3 –signal “/system/receiver/in1\[7:0\]”
```

wvFindLvsCmd

Description

Locate the Level-shifted command that applies to the Level-shifted signal in the power source code pane.

Syntax

```
wvFindLvsCmd -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvFindLvsCmd -win $_nWave2 -signal "/system/i_pram/VMA"
```

wvFindRetCmd

Description

Locate the Retention command that applies to the Retention signal in the power source code pane.

Syntax

```
wvFindRetCmd -win window -signal fullName
```

Arguments

`-signal` *`fullName`*

Specify the full name of the signal.

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvFindRetCmd -win $_nWave2 -signal "/system/i_cpu/i_CCU/
bus_mode\[2:0\]"
```

wvFindSPACmd

Description

Locate the SPA command that applies to the SPA signal in the power source code pane.

Syntax

```
wvFindSPACmd -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvFindSPACmd -win $_nWave2 -signal "/tb/top/i_sub1/i"
```

wvFindSRSNCmd

Description

Locate the SRSN command that applies to the SRSN signal in the power source code pane.

Syntax

```
wvFindSRSNCmd -win window -signal fullName
```

Arguments

`-signal fullName`

Specify the full name of the signal.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvFindSRSNCmd -win $_nWave2 -signal "/tb/system/reset_cpu"
```

wvShowDrivingInfo

Description

Add runtime signal(s) to display the full driving information of the selected signal.

Syntax

```
wvShowDrivingInfo -win window -signal fullName
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`-signal fullName`

Specify the signal to add its full driving information.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvShowDrivingInfo -win $_nWave2 -signal "/system/a"
```

wvShowDrvSupplySet

Description

Add the driver supply set of the selected SPA-affected signal to *nWave*.

Syntax

```
wvShowDrvSupplySet -win window -signal fullName
```

Arguments

`-win window`

Specify the window ID of the invoking *nWave* window.

`-signal fullName`

Specify the signal to add the driver supply set.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvShowDrvSupplySet -win $_nWave2 -signal "/system/a"
```

wvShowPowerState

Description

Add runtime signal(s) to display transitions of the corresponding power modes.

Syntax

```
wvShowPowerState -win window -signal fullName
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-signal` *`fullName`*

Specify the full name of the signal.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvShowPowerState -win $_nWave2 -signal "/system/i_cpu/i_CCU/
mux_sel\[2:0\]"
```

wvShowRcvSupplySet

Description

Add the receiver supply set of the selected SPA-affected signal to *nWave*.

Syntax

```
wvShowRcvSupplySet -win window -signal fullName
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

`-signal` *`fullName`*

Specify the signal to add the receiver supply set.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvShowRcvSupplySet -win $_nWave2 -signal "/system/a"
```

Right-Click Commands

wvAddCounterSignal

Description

Add a counter signal for the selected signal.

Syntax

```
wvAddCounterSignal [-win window] [-type AnyChange|Rising|Falling]
```

Arguments

`-type AnyChange|Rising|Falling`

Specify the type for the counter signal.

`-win window`

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddCounterSignal -win $_nWave1 -type Rising
```

wvAddTriggerSignal

Description

Add trigger signals.

Syntax

```
wvAddTriggerSignal [-win window] signalList
```

Arguments

`signalList`

Specify the signal list.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvAddTriggerSignal -win $_nWave1 {file0/:/system/addr[7:0]}
{/system/clock}
```

wvCopySignalFullPathToClipboard

Description

This command copies the full hierarchical name(s) of the selected signal(s) into the clipboard buffer.

Syntax

```
wvCopySignalFullPathToClipboard -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *nWave* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvCopySignalFullPathToClipboard -win $_nWave2
```

wvDeleteTriggerSignal

Description

Delete trigger signals.

Syntax

```
wvDeleteTriggerSignal [-win window] signalList
```

Arguments

`signalList`

Specify the signal list.

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvDeleteTriggerSignal -win $_nWave1 {file0/:/system/addr[7:0]} {/
system/clock}
```

wvRMBAddToNewGroup

Description

Turns on/off the **Add to New Group** option in the right-click **Add to Waveform** option.

Syntax

```
wvRMBAddToNewGroup -on/-off
```

Arguments

`-on`

Turns on the **Add to New Group** option in the right-click **Add to Waveform** option.

`-off`

Turns off the **Add to New Group** option in the right-click **Add to Waveform** option.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvRMBAddToNewGroup -on
```

wvShowDriverSignals

Description

Show the final destination driver of the selected signal in *nWave*.

Syntax

```
wvShowDriverSignals -win window { signal_fullName }
```

Arguments

`signal_fullName`

Specify the full name of the signal.

`-win` *`window`*

Specify the window ID of the invoking waveform pane.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvShowDriverSignals -win $_nWave2 { /system/FirstDataOutRdy }
```

wvShowLoadSignals

Description

Show the final destination load of the selected signal in *nWave*.

Syntax

```
wvShowLoadSignals -win window {signal_full_name}
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking waveform pane.

`signal_full_name`

Specify the full name of the signal.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
wvShowLoadSignals -win $_nWave2 { /system/FirstDataOutRdy }
```

nState

Window

fsmCloseWindow

Description

Closes the *nState* window.

Syntax

```
fsmCloseWindow [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmCloseWindow -win $_nState1
```

fsmCreateWindow

Description

Duplicates the *nState* window.

Syntax

```
fsmCreateWindow [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns the *nState* window ID.

Example

```
fsmCreateWindow -win $_nState1
```

fsmCreatePartial

Description

Creates the partial FSM from the selected state(s).

Syntax

```
fsmCreatePartial [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns the *nState* window ID.

Example

```
fsmCreatePartial-win $_nState1window
```

fsmDuplicateWindow

Description

Duplicates the specified window.

Syntax

```
fsmDuplicateWindow [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmDuplicateWindow -win $_nState8
```

fsmGetAllWindows

Description

Gets all *nState* window IDs.

Syntax

```
fsmGetAllWindows
```

Value Returned

A list of window IDs for all *nState* windows if successful; otherwise, returns 0.

Example

```
set fsm_window_list fsmGetAllWindows
```

fsmGetCurrentWindow

Description

Gets the active *nState* window.

Syntax

```
fsmGetCurrentWindow
```

Value Returned

Returns the *nState* window ID.

Example

```
set current_window [fsmGetCurrentWindow]
```

fsmGetXWindowId

Description

Returns the X window ID for the ID returned from the `fsmCreateWindow` command.

Syntax

```
fsmGetXWindowId [-id $xwid]
```

Arguments

`-id` *`$xwid`*

Tcl window variable ID.

Value Returned

Returns the X window id if successful; otherwise, sets the result of the command to `0x0` and returns `CMDOBJ_ERROR`.

Example

```
fsmGetXWindowId -id $_nState1
```

fsmResizeWindow

Description

Resizes the *nState* window.

Syntax

```
fsmResizeWindow [-win window] x y width height
```

Arguments

`height`

Specifies the window height.

`width`
Specifies the window width.

`-win` *`window`*
Specifies the window id of the invoking *nState* window.

`x y`
Specifies the starting location.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmResizeWindow -win $_nState1 10 10 400 300
```

fsmSetCurrentWindow

Description

Sets the specified *nState* window ID as the current window.

Syntax

```
fsmSetCurrentWindow window
```

Value Returned

Returns the *nState* window ID.

Example

```
fsmSetCurrentWindow $_nState1
```

Property

fsmGetMachineProperty

Description

Gets the machine property in the specified window.

Syntax

```
fsmGetMachineProperty [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmGetMachineProperty -win $_nState8
```

fsmGetPort

Description

Get the name list of the selected port.

Syntax

```
fsmGetPort [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

Lists of port names if successful; otherwise, returns 0.

Example

```
fsmGetPort StartFSM2
```

fsmGetPortProperty

Description

Lists the properties of the selected port. If the type (for example, io) is not specified, lists all properties of the port.

Syntax

```
fsmGetPortProperty [-win window] [-io] [-where] [portName]
```

Arguments

`-io`

Gets the port direction, input, and output.

`portName`

Lists the properties of the port name. The default value is the currently selected port (only one selected port is acceptable).

`-where`

Returns the name of the connected finite state machine.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Lists of properties if successful; otherwise, returns 0.

Example

```
fsmGetPortProperty -where
```

fsmGetState

Description

Gets the name list of the selected states.

NOTE: This command only works on one *nState* window. It finds the states that match the input value.

Syntax

```
fsmGetState [-win window] [-selected] [-value number]
```

Arguments

`-selected`

Selects the state to get the name list.

`-value number`

Specifies the corresponding value of the state.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns the state name list if successful; otherwise, returns 0.

Example

Assume the `xyz` state is selected in the *nState* window, issuing the following command returns `xyz`:

```
fsmGetState -selected -win 3
```

Assume you have state *others* currently displayed in *nState*, issuing the following command returns `ST1`:

```
fsmGetState -value 1
```

fsmGetStateProperty

Description

Lists the properties of the state. If the type (for example, value, action) is not specified, lists all properties of the state.

Syntax

```
fsmGetStateProperty [-win window] [-value] [-action] [stateName]
```

Arguments

`-action`

Gets the action of the state, for example, `StartFSM3=1;` `FirstDataOutRdy=1;`.

`stateName`

Specifies the state. The default is the currently selected state.

NOTE: You can select only one transition for each time.

`-value`

Gets the value of the state.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns the lists of properties if successful; otherwise, returns 0.

Example

```
fsmGetStateProperty stateA -action
```

fsmGetTransition

Description

Gets the name list of the selected transition.

Syntax

```
fsmGetTransition [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Lists transition(s) if successful; otherwise, returns 0.

Example

```
fsmGetTransition -win $_nState1
```

fsmGetTransitionProperty

Description

Lists properties of the selected transition. If the type (for example, priority, action) is not specified, lists all properties of the transition.

Syntax

```
fsmGetTransitionProperty [-win window] [-priority] [-action]
[-condition] [transition_name]
```

Arguments

`-action`

Gets the action of the transition, for example, `StartFSM1=1;`.

`-condition`

Gets the condition of the transition, for example, `FirstDataInRdy`.

`-priority`

Gets the priority of the transition, for example, `2`.

`transition_name`

Specifies the state name of the transition. The default is the currently selected transition.

NOTE: You can select only one transition for each time.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns the list of properties if successful; otherwise, returns 0.

Example

```
fsmGetTransitionProperty
```

fsmReport

Description

Analyzes the report.

Syntax

```
fsmReport [-win window] [-tofile fileName] [-state] [-transition]
[-all]
```

Arguments

`-all`

Lists the source code. Also, lists the state and transition analysis report if the state animation is *on*.
If none is specified, the list source code is the default report.

`-state`

Generates the state analysis report.

NOTE: The FSDB file must be loaded and the state animation must be turned *on* first).

`-tofile fileName`

Specifies the file name saved to the disk.

`-transition`

Generates the transition analysis report.

NOTE: The FSDB file must be loaded and the state animation must be turned *on* first).

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

When `-tofile` is specified, returns 1 if successful to save it to the disk; otherwise, returns 0. When `-tofile` is not specified, returns the analysis report directly.

Example

```
fsmReport -tofile /tmp/report.out -all
```

fsmSetPortProperty

Description

Sets the properties of the port.

Syntax

```
fsmSetPortProperty [-win window] -animationLink on|off portName
```

Arguments

`-animationLink on|off`

When this option is turned *on*, it shows the related link(s) dynamically. When this option is turned *off*, it does not show the related link(s) dynamically. The default is *on*.

`portName`

Specifies the port name.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSetPortProperty StartFSM2 -animationLink off
```

Analysis

fsmAddSequence

Description

Sets the search sequence.

Syntax

```
fsmAddSequence [-win window] seqName seqList
```

Arguments

`seqList`

Generates the list of state names with optional occurring times separated by space.

`seqName`

Specifies the sequence name.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmAddSequence logic_seq stateA stateB:* stateA:4
```

fsmAnalyzeFSM

Description

Extracts the *nState* analysis report in the batch mode. You can read the previous analysis report and the report is used to check the coverage incrementally.

Syntax

```
fsmAnalyzeFSM -instance instanceName [-fsdb fsdbFile]
[-import_rpt reportFile] [-output accumulatedReportFile]
```

Arguments

`-fsdb` *`fsdbFile`*

Specifies the FSDB file that contains simulation results to be analyzed.

NOTE: If the FSDB file is not specified, the loaded simulation result is analyzed.

`-import_rpt` *`reportFile`*

Specifies the analysis report file to be imported.

`-instance` *`instanceName`*

Specifies the full hierarchical name of the FSM instance.

`-output` *`accumulatedReportFile`*

Specifies the output file name.

Value Returned

Returns 1 if successfully; otherwise, returns 0.

Example

```
fsmAnalyzeFSM -instance system.MASTER.fsm_master:FSM0:31:105:FSM\
              -output $env(TURBO_AU_DIR)/rpt1.log \
              -import_rpt /rd40a/chsu/SPSqa22035/masterFSM_1.log
```

fsmDeleteSequence

Description

Deletes the search sequence(s).

Syntax

`fsmDeleteSequence [-win` *`window`*`] [`*`seqName`* `| -all]`

Arguments

`-all`

Deletes all existing sequences.

`seqName`

Specifies the sequence to be deleted.

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmDeleteSequence -win $_nState1 -all
```

fsmGetSequence

Description

Gets the sequence name list or the definition of the given sequence name.

Syntax

```
fsmGetSequence [-win window] seqName
```

Arguments

`seqName`

Specifies the sequence name.

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Lists the definition of the given sequence name or all existing sequence names if successful; otherwise, returns 0.

Example

```
fsmGetSequence Unknown
```

fsmGetTime

Description

Gets the base time for the simulation search.

Syntax

```
fsmGetTime [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns the current time value.

Example

```
fsmGetTime
```

fsmJumpBack

Description

In the animation mode, if the current state or the next state of the current state transition does not exist in a partial FSM window, `fsmJumpBack` is used to move the cursor time forward. In this case, a transition exists on that cursor time and the next state of the transition exists in the partial FSM. If no such cursor time exists in the remaining simulation results, the cursor time is not changed. This command only works on a partial FSM window.

Syntax

```
fsmJumpBack [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmJumpBack -win $_nState1
```

fsmOpen

Description

Sets the scope in the specified window.

Syntax

```
fsmOpen [-lib libraryName][-module moduleName]
[-instance instanceName] [-win window]
```

Arguments

`-instance instanceName`

Specifies the instance.

`-lib libraryName`

Specifies the library.

`-module moduleName`

Specifies the module.

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmOpen -lib "DebussyLib"
-module "fsm_child2(@1)#__fsm__0"
-instance "system.CHILD2.fsm_child2(@1):FSM0:34:101:FSM"
-win $_nState8
```

fsmSearchNext

Description

Searches the next object based on the `fsmSetSearchMode` command.

Syntax

```
fsmSearchNext [-win window]
```

Arguments

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

1 and first found state name and time if successful; otherwise, returns 0.

Example

```
fsmSearchNext
```

fsmSearchPrev

Description

Searches the previous object based on the `fsmSetSearchMode` command.

Syntax

```
fsmSearchPrev [-win window]
```

Arguments

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

1 and first found state name and time if successful; otherwise, returns 0.

Example

```
fsmSearchPrev
```

fsmSetSearchMode

Description

Sets the mode for simulation search.

Syntax

```
fsmSetSearchMode [-win window] -state stateName | -sequence
seqName| -transition transitionName
```

Arguments

`-sequence seqName`

Searches the specified sequence name.

`-state stateName`

Searches the specified state name.

`-transition transitionName`

Searches the specified transition name.

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSetSearchMode -state stateA
```

fsmSetStateDelay

Description

Sets the state delay time in the specified window.

Syntax

```
fsmSetStateDelay delaytime [-win window]
```

Arguments

`delaytime`

Specifies the delay time. The unit is *ns*.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSetStateDelay 10 -win $_nState8
```

fsmSetTime

Description

Sets the base time for the simulation search.

Syntax

```
fsmSetTime [-win window] time
```

Arguments

`time`

Specifies the cursor time in the integer type.

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Time value if successful (a floating value is truncated to an integer); otherwise, returns -1.

Example

```
fsmSetTime 550
```

Select

fsmAddSignalToWave

Description

Adds the state and clock signal(s) for the synchronous design to the *nWave* window.

Syntax

```
fsmAddSignalToWave [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmAddSignalToWave -win $StateMachine2
```

fsmDeselectAll

Description

Deselects all selected objects on the specified window.

Syntax

```
fsmDeselectAll [-win window] [-sticky | -nonSticky]
```

Arguments

`-nonSticky`

Deselects all non-sticky objects (states or transitions).

NOTE: If the state or transition is not specified, all objects selected previously is deselected.

`-sticky`

Deselects all sticky objects (ports).

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmDeselectAll -win $_nState1
```

fsmFindSignal

Description

Finds states and transitions of a signal that contains a reference to the signal in the FSM of the *nState* window. The states and transitions found is put into the selection set of the *nState* window.

Syntax

```
fsmFindSignal [-win window] signalName
```

Arguments

`signalName`

Specifies the signal to be found.

NOTE: The name must be under the scope of FSM rather than a full hierarchical name. Wildcard is in the name. In such case, all signal names under the scope of the FSM that match the pattern is returned.

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

List of pairs whose first element is a full hierarchical signal name that matches the *signalName* and the other element is a list of states and transitions that contains a reference to the first element. An empty list is returned if the signal is not found or an error is encountered.

Example

```
fsmFindSignal -win system.MASTER.Reset
```

fsmFindState

Description

Finds the specified state in the selected window.

Syntax

```
fsmFindState statename [-win window]
```

Arguments

`statename`

Specifies the state.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmFindState Others -win $_nState8
```

fsmSelect

Description

Selects the objects in the *nState* window.

Syntax

```
fsmSelect [-win window] [-add | -delete] {-state stateName|
-transition transitionName|-port portName}
```

Arguments

`-add`

Adds the object to the selected set.

`-delete`

Removes the object from the selected set.

`-port portName`

Specifies the port name.

`-state stateName`

Specifies the state name.

`-transition transitionName`

Specifies the transition name.

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSelect -win $StateMachine2 -add -state ST3
```

fsmSelectAll

Description

Selects all objects in the specified window.

Syntax

```
fsmSelectAll [-win window]
```

Arguments

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSelectAll -win $_nState5
```

fsmSelectAllStates

Description

Selects all states in the specified window.

Syntax

```
fsmSelectAllStates [-win window]
```

Arguments

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSelectAllStates -win $_nState4
```

fsmSelectAllTrans

Description

Selects all transitions in the specified window.

Syntax

```
fsmSelectAllTrans [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSelectAllTrans -win $_nState5
```

Expand

fsmAddStates

Description

Adds the state to the partial FSM *nState* window.

Syntax

```
fsmAddStates [-win window] state1 [state2]...
```

Arguments

state1 [*state2*]

Specifies the state to be added to the window.

-win *window*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmAddStates -win $StateMachine2 ST3
```

fsmDeleteStates

Description

Removes the selected states from the partial FSM *nState* window.

Syntax

```
fsmDeleteStates [-win window]
```

Arguments

-win *window*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmDeleteState  -win $_nState1
```

fsmExpandNext

Description

Expands the next partial FSM *nState* windows of the currently selected state.

Syntax

```
fsmExpandNext [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmExpandNext -win $_nState1
```

fsmExpandPrev

Description

Expands the previous partial FSM *nState* windows of the currently selected state.

Syntax

```
fsmExpandPrev [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmExpandPrev -win $_nState1
```

fsmExpandAll

Description

Expands both the previous and next partial FSM *nState* windows of the currently selected state.

Syntax

```
fsmExpandAll [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmExpandAll -win $_nState1
```

fsmUndo

Description

Undoes the last action in the partial FSM *nState* window.

Syntax

```
fsmUndo [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmUndo -win $_nState1
```

View

fsmFit

Description

Fits the current design in the window.

Syntax

```
fsmFit [-win window][-selected]
```

Arguments

`-selected`

Fits the current design into the selected window.

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmFit
fsmFit -selected
```

fsmLastView

Description

Switches to the previous view area.

Syntax

```
fsmLastView [-win window]
```

Arguments

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmLastView
```

fsmLineDown

Description

Moves the current view area down.

Syntax

```
fsmLineDown [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmLineDown
```

fsmLineLeft

Description

Moves the current view area to the left.

Syntax

```
fsmLineLeft [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmLineLeft
```

fsmLineRight

Description

Moves the current view area to the right.

Syntax

```
fsmLineRight [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmLineRight
```

fsmLineUp

Description

Moves the current view area up.

Syntax

```
fsmLineUp [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmLineUp
```

fsmPanDown

Description

Pans halfway down the current view area in the current window.

Syntax

```
fsmPanDown [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmPanDown
```

fsmPanLeft

Description

Pans halfway left of the current window.

Syntax

```
fsmPanLeft [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmPanLeft
```

fsmPanRight

Description

Pan halfway right of the current window.

Syntax

```
fsmPanRight [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmPanRight
```

fsmPanUp

Description

Pan halfway up the current view area in the current window.

Syntax

```
fsmPanUp [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmPanUp
```

fsmZoom

Zooms the view area of the invoking *nState* window.

Syntax

```
fsmZoomIn [-win window] point1_X point1_Y point2_X point2_Y
```

Arguments

`point1_X point1_Y point2_X point2_Y`

Specifies the coordinates of the *nState* window.

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmZoom -1375 621 -334 1105 -win $_nState8
```

fsmZoomIn

Zooms the viewing area to half the magnification of the previous view in both the horizontal and vertical directions in the invoking *nState* window.

Syntax

```
fsmZoomIn [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmZoomIn
```

fsmZoomOut

Description

Zooms the viewing area to two times the magnification of the previous view from the center point in both the horizontal and vertical directions in the invoking *nState* window.

Syntax

```
fsmZoomOut [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmZoomOut
```

Configuration

fsmResetDisplayAttr

Description

Resets all display attributes to the default.

Syntax

```
fsmResetDisplayAttr
```

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmResetDisplayAttr
```

fsmSetDisplayAttr

Description

Sets the display attributes.

Syntax

```
fsmSetDisplayAttr typeName {[-fillColor color] | [-lineColor
color] | [-lineWidth width] | [-showStateAction on|off] |
[-showTransAction on|off] | [-showTransCond on|off] | [-StateLabel
NAME|VALUE|BOTH] | [-StateValueRadix ORIG|BIN|OCT|DEC|HEX] |
[-textColor color] | [-tip on|off]}
```

Arguments

`-fillColor` *`color`*

Specifies the background color. The default is *black*.

`-lineColor` *`color`*

Specifies the line color. The default is *blue*.

`-lineWidth` *`width`*

Specifies the line width in the range from 1 to 5.

`-showStateAction on|off`

Turns the display of the state action *on* or *off*.

`-showTransAction on|off`

Turns the display of the transition action *on* or *off*.

`-showTransCond on|off`

Turns the display of the transition condition *on* or *off*.

`-StateLabel NAME|VALUE|BOTH`

Specifies the label to be displayed in a state in the *nState* window. The options include NAME, VALUE, and BOTH.

`-StateValueRadix ORIG|BIN|OCT|DEC|HEX`

Specifies the state value format to display in a state in the *nState* window. The options include:

- ORIG: The state value format is displayed the same as the *Machine Properties* form.
- BIN: The state value is displayed in the Binary format.
- OCT: The state value is displayed in the Octal format.
- DEC: The state value is displayed in the Decimal format.
- HEX: The state value is displayed in the Hexadecimal format.

`-textColor` *`colorId`*

Specifies the text color.

`-tip on|off`

Turns the display of tips *on* or *off*.

`typeName`

Specifies the type to display on the FSM session in the `novas.rc` resource file.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSetDisplayAttr {InitState} -lineWidth 2 -lineColor red1ID_RED4
fsmSetDisplayAttr -StateLabel VALUE
fsmSetDisplayAttr -StateValueRadix HEX
```

fsmSetOptions

Description

Sets the viewing options in the *nState* window.

Syntax

```
fsmSetOptions [-win window] [-junction on|off] [-toolbar on|off]
[-msgLine on|off] [-stateAction on|off] [-transitionCondition
on|off][-transitionAction on|off] [-largeFont on|off]
```

Arguments

`-junction on|off`

Turns the display of the junction *on* or *off*. The default is *off*.

`-largeFont on|off`

Turns the display of the large font *on* or *off*. The default is *off*.

`-msgLine on|off`

Turns the display of the message line *on* or *off*. The default is *on*.

`-stateAction on|off`

Turns the display of the state action *on* or *off*. The default is *off*.

`-transitionAction on|off`

Turns the display of the transition action *on* or *off*. The default is *off*.

`-transitionCondition on|off`

Turns the display of the transition condition *on* or *off*. The default is *off*.

`-toolbar on|off`

Turns the display of the toolbar *on* or *off*. The default is *off*.

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmSetOptions -junction on
```

Print

fsmCapture

Description

Captures the current window and saves as an image file in the PNG format.

Syntax

```
fsmCapture [-win window] [-footer footer] [-region region]
-file fileName
```

Arguments

`-file fileName`

Specifies the file to save the captured image.

`-footer footer`

Specifies the message to be shown on the footer.

`-region region`

Specifies the region to be printed or saved.

`-win window`

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmCapture -footer "CPU" -file "/home/cpu.png"
```

fsmPrint

Description

Prints the content of the invoking *nState* window.

Syntax

```
fsmPrint [-win window] [-header header] [-footer footer]
[-marginLeft location] [-marginRight designName][-marginTop
location] [-marginBottom designName][-pageFrom fromPage] [-pageTo
toPage][-copy numofcopy] [-orient orientation][-paper papersize]
[-color color]{[-printer printerName]|[-file printFile]}
```

Arguments

`-color color`

Turns the color printing *on*.

`-copy numofcopy`

Specifies the number of copies to be printed. The default is *1*.

`-file printFile`

Prints the window content to the specified file name.

`-footer footer`

Specifies the message to be shown on the footer.

`-header header`

Specifies the message to be shown on the header.

`-marginBottom designName`

Specifies the length of the bottom margin. The default is *0.5 inch*.

`-marginLeft location`

Specifies the length of the left margin. The default is *0.5 inch*.

`-marginRight designName`

Specifies the length of the right margin. The default is *0.5 inch*.

`-marginTop location`

Specifies the length of the top margin. The default is *0.5 inch*.

`-orient orientation`

Specifies the source of the paper. The orientation options are *landscape* and *portrait*.

`-pageFrom fromPage`

Specifies the start page to print.

`-pageTo` *`toPage`*

Specifies the end page to print.

`-paper` *`papersize`*

Specifies the paper size for the printed page. The size options are Letter, Legal, A4, A3, A2, A1, A0, B, C, D, E, and User-defined.

`-printer` *`printerName`*

Specifies the printer to which *nState* sends the window content. The default printer is *lp*.

`-win` *`window`*

Specifies the window id of the invoking *nState* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
fsmPrint -win $State1 -header "a demo design" -printer lpr
```

Temporal Flow View

NOTE: Before specifying the Temporal Flow View commands, the **debLoadSimResult** command must be specified to load the FSDB file and the **tfgBehaviorAnalysis** command must be specified to perform behavior analysis.

tfgAddBookMark

Description

Adds a bookmark item.

NOTE: The `tfgAddBookMark` command replaces the `tsAddBookmark` command.

Syntax

```
tfgAddBookMark -win window -name name
```

Argument

`-name` *`name`*

Specifies the bookmark name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgAddBookMark -win $_tFlowView2 -name "bm1"
```

tfgAddRefSignals

Description

Adds reference signal to the *Temporal Flow View* window.

NOTE: The `tfgAddRefSignals` command replaces the `tsAddRefSignals` command.

Syntax

```
tfgAddRefSignals -win window -ref signame_with_time
```

Argument

`-ref`

Specifies the reference signal.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgAddRefSignals -win $_tFlowView -ref "top.cpu.bus#100"
```

tfgBack

Description

Moves the cursor back to the previous snapshot of the window.

NOTE: The `tfgBack` command replaces the `tsBack` command.

Syntax

```
tfgBack -win window
```

Argument

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgBack -win $_tFlowView2
```

tfgBehaviorAnalysis

Description

Performs Behavior Analysis.

NOTE: The `tfgBehaviorAnalysis` command replaces the `tsBehaviorAnalysis` command.

Syntax

```
tfgBehaviorAnalysis [-bdb_load] [-bdb_incr] [-bdb_load_scope]
[-scope] [-bas {{scope1}{scope2}...{scopeN}}] [-cellModel
0|1|2|3|4] [-incr] [-bboxEmptyModule 0|1] [-loopUnroll num]
[-clockSkew value] [-bboxModule 0|1] [-bboxModuleFile]
[-symLibCellList] [-symLibCellListFile] [-simModuleList]
[-simModuleListFile] [-bboxLib][-bboxLibFile] [-apimem]
[-bboxSysTaskFile] [-macroCell] [-macroCellFile] [-mergeRange]
[-traceflattenedMDA]
```

Argument

`-apimem`

Specifies a file containing the PLI memory definition.

`-bas {{scope1}{scope2}...{scopeN}}`

Specifies one or more scopes on which *Behavior Analysis* is applied. If both `-bas` and `-scope` are set, the `-scope` setting will be ignored.

`-bboxEmptyModule 0|1`

When set to 1, empty modules are set as black boxes.

`-bboxLib`

Specifies the library in which all modules/functions are set as black boxes.

`-bboxLibFile`

Specifies a file containing a set of libraries in which all modules/functions are set as black boxes.

`-bboxModule 0|1`

Specifies modules to be set as black boxes.

`-bboxModuleFile`

Specifies a file containing a set of modules that are set as black boxes.

`-bboxSysTaskFile`

Specifies a file containing a set of system tasks that are set as black boxes.

`-bdb_incr`

Loads the Behavior Database (BDB) incrementally. This option works only when the `-bdb_load` option is specified.

`-bdb_load`

Specifies the Behavior Database (BDB) file to perform the Behavior Analysis. This option must be the first option; otherwise, it is ignored. If the first option is `-bdb_load`, other Behavior Analysis options except `-bdb_incr` and `-bdb_load_scope` are ignored.

`-bdb_load_scope`

Specifies the path to load the top module of the Behavior Database (BDB) file. This option works only when the **-bdb_load** option is specified.

`-cellModel 0|1|2|3|4`

Specifies the source for a library cell definition. The following options are available:

0: symbol library first

1: symbol library only

2: simulation library only

3: simulation library first

4: automatic mode

`-clockSkew` *`value`*

Specifies the maximum value for the clock skew (typically in gate level design after clock the tree synthesis). The value should be 0 or a positive integer.

`-incr`

Performs incremental Behavior Analysis.

`-loopUnroll` *`num`*

Specifies the maximum number of for loops to unroll. The number should be 0 or a positive integer.

`-macroCell`

Specifies the modules to be treated as macro cell.

`-macroCellFile`

Specifies a file containing a set of modules that are treated as macro cell.

`-mergeRange`

Specifies whether to merge range in SFG (display).

`-scope`

Specifies the scope on which the *Behavior Analysis* is applied.

`-simModuleList`

Treats the listed cell using the simulation model.

`-simModuleListFile`

Treats the listed cell using the simulation model.

`-symLibCellList`

Treats the cells listed as symbol library cells.

`-symLibCellListFile`

Treats the listed cells as symbol library cells.

`-traceflattenedMDA`

Specifies whether to display multiple dimensional arrays as memories or flattened multiple dimensional arrays into buses.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgBehaviorAnalysis -bdb_load work.lib++/work.bdb

tfgBehaviorAnalysis -bdb_load work.lib++/work.bdb -bdb_incr

tfgBehaviorAnalysis -bdb_load work.lib++/work.bdb -bdb_load_scope
{system.i_cpu}

tfgBehaviorAnalysis -bdb_load work.lib++/work.bdb -bdb_load_scope
{system.i_cpu} -bdb_incr

tfgBehaviorAnalysis -scope {wave} -loopUnroll 0 -clockSkew 1
-bboxEmptyModule 1 -cellModel 0
```

```
tfgBehaviorAnalysis -bas {{system.CHILD1} {system.CHILD2}}
-clockSkew 0 -loopUnroll 0 -bboxEmptyModule 0  -cellModel 0
```

tfgBehaviorAnalysisAfterLoadDesign

Performs Behavior Analysis immediately after loading a design. This Tcl command is a recorded command for the operation of command line options (`-ba`/`-bas`) and does not have an equivalent GUI command. It is almost the same as `tfgBehaviorAnalysis`. Only `-scope` or `-bas` option needs to be specified (other options are set as the default value).

NOTE: The `tfgBehaviorAnalysisAfterLoadDesign` command replaces the `tsBehaviorAnalysisAfterLoadDesign` command.

Syntax

```
tfgBehaviorAnalysisAfterLoadDesign [-scope] [-bas scope(s)]
[-cellModel 0|1|2|3] [-incr] [-bboxEmptyModule 0|1]
[-loopUnroll num] [-clockSkew value] [-bboxModule 0|1]
[-bboxModuleFile] [-symLibCellList] [-symLibCellListFile]
[-simModuleList] [-simModuleListFile] [-bboxLib][-bboxLibFile]
[-apimem] [-bboxSysTaskFile] [-macroCell] [-macroCellFile]
[-mergeRange] [-traceflattenedMDA]
```

Argument

`-apimem`

Specifies a file containing the PLI memory definition.

`-bas` *`scope(s)`*

Specifies one or more scopes on which the *Behavior Analysis* is applied. If both `-bas` and `-scope` are set, the `-scope` setting is ignored.

`-bboxEmptyModule 0|1`

When set to 1, empty modules are set as black boxes.

`-bboxLib`

Specifies the library in which all modules/functions are set as black boxes.

`-bboxLibFile`

Specifies a file containing a set of libraries in which all modules/functions are set as black boxes.

`-bboxModule 0|1`

Specifies modules to be set as black boxes.

`-bboxModuleFile`

Specifies a file containing a set of modules that are set as black boxes.

`-bboxSysTaskFile`

Specifies a file containing a set of system tasks that are set as black boxes.

`-cellModel 0|1|2|3`

Specifies the source of the definition of a library cell. The following options are available:

0: symbol library first

1: symbol library only

2: simulation library first

3: simulation library only

`-clockSkew` *`value`*

Specifies the maximum value for clock skew (typically in gate level design after clock tree synthesis). The value should be 0 or a positive integer.

`-incr`

Performs incremental Behavior Analysis.

`-loopUnroll` *`num`*

Specifies the maximum number of for loops to unroll. The number should be 0 or a positive integer.

`-macroCell`

Specifies modules to be treated as macro cell.

`-macroCellFile`

Specifies a file containing a set of modules that are treated as macro cell.

`-mergeRange`

Specifies whether to merge range in SFG (display).

`-scope`

Specifies the scope on which the *Behavior Analysis* is applied.

`-simModuleList`

Treats the listed cell using the simulation model.

`-simModuleListFile`

Treats the listed cell using the simulation model.

`-symLibCellList`

Treats the cells listed as symbol library cells.

`-symLibCellListFile`

Treats the listed cells as symbol library cells.

`-traceflattenedMDA`

Specifies whether to display multiple dimensional arrays as memories or flattened multiple dimensional arrays into buses.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
Verdi s.v -ba -bas "\tb.tb "
```

The recorded Tcl commands are as follows:

```
...
tfgBehaviorAnalysisAfterLoadDesign -incr -bas "{\tb.tb }"
-clockSkew 0 -loopUnroll 0 -bboxEmptyModule 0 -cellModel 0
-bboxIgnoreProtected 0
```

tfgClearAllDecisionPtr

Description

Clears all decision points in the *Temporal Flow View* window.

Syntax

`tfgClearAllDecisionPtr –win` *`window`*

Argument

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgClearAllDecisionPtr -win $_tFlowView3
```

tfgClockHighlight

Description

Highlights the specified clock.

NOTE: The `tfgClockHighlight` command replaces the `tsClockHighlight` command.

Syntax

```
tfgClockHighlight -win window -color color -clock clock
```

Argument

`-clock` *`clock`*

Specifies the clock to be highlighted.

`-color` *`color`*

Specifies the color.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgClockHighlight -win $_tFlowView2 -color red -clock "top.clk"
```

tfgCloseViewer

Description

Closes an existing *Temporal Flow View* window.

NOTE: The **tfgCloseViewer** command replaces the **tsCloseViewer** command.

Syntax

```
tfgCloseViewer -win window
```

Argument

`-win window`

Specifies the existing *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgCloseViewer -win $_tFlowView2
```

tfgConfigBookMark

Description

Configures bookmarks in the *Temporal Flow View* window.

NOTE: The **tfgConfigBookMark** command replaces the **tsConfigBookMark** command.

Syntax

```
tfgConfigBookMark -win window [-jumpMRU on|off] [-keepRatio
on|off] [-abortOnHidden on|off] [-defaultName on|off]
[-overrideLRU on|off] [-maxShortCut 0|3|9]
```

Argument

`-abortOnHidden on|off`

Specifies whether to go to a bookmark associated with a hidden node. When this option is turned *on*, the flow view display is not updated to show hidden bookmarks. When this option is turned *off*, the flow view display is updated to expand hidden nodes and the associated bookmark. The default is *off*.

`-defaultName on|off`

Names the bookmarks at creation. When this option is turned *on*, the name defaults to the full hierarchical path and time for the node. When this option is turned *off*, the bookmark name can be changed via a form. The default is *off*.

`-jumpMRU on|off`

Jumps using the most recently used (MRU) bookmark order. When *on*, the last selected bookmark becomes index 1 and the remaining bookmarks shift

down one row. When *off*, the bookmarks maintain the original selection order (last selected is the first row, first selected is the last row). The default is *off*.

`-keepRatio on|off`

Jumps to the bookmark keeping the current zoom ratio (scroll the signals only) in the flow view. When *on*, the current zoom ratio is maintained regardless of the zoom ratio saved with the bookmark. When *off*, the zoom ratio is changed to match that saved with the bookmark. The default is *off*.

`-maxShortCut 0|3|9`

Specifies the number of bookmarks to be saved as shortcuts. Options are 0, 3, and 9. The default is *3*. The order in the bookmark cache is determined by the most recently used policy.

`-overrideLRU on|off`

Specifies the method for removing bookmarks when the maximum number of 32 bookmarks is exceeded. When *on*, the least recently used (LRU) entry is automatically replaced. When *off*, if the maximum value is reached, a message window opens and another bookmark cannot be added until an existing bookmark is deleted. The default is *off*.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgConfigBookMark -win $_tFlowView2 -defaultName on
```

tfgCreateSrcWindow

Description

Creates a source code window.

NOTE: The **tfgCreateSrcWindow** command replaces the **tsCreateSrcWindow** command.

Syntax

```
tfgCreateSrcWindow
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgCreateSrcWindow
```

tfgDelBookMark

Description

Deletes the bookmark item.

NOTE: The **tfgDelBookMark** command replaces the **tsDelBookmark** command.

Syntax

```
tfgDelBookMark -win window -mru number
```

Argument

`-mru number`

Specifies the bookmark index.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDelBookMark -win $_tFlowView2 -mru 1
```

tfgDeselectAll

Description

De-selects all.

NOTE: The **tfgDeselectAll** command replaces the **tsDeselectAll** command.

Syntax

```
tfgDeselectAll -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDeselectAll -win $_tFlowView2
```

tfgDisplayAllOnNWave

Description

Shows all Temporal Flow View signals on the *nWave* window.

NOTE: The **tfgDisplayAllOnNWave** command replaces the **tsDisplayAllOnNWave** command.

Syntax

```
tfgDisplayAllOnNWave -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDisplayAllOnNWave -win $_tFlowView2
```

tfgDisplayWaveformMismatchOnNWave

Description

Traces the FSDB mismatch and displays the results on *nWave* window.

NOTE: The **tfgDisplayWaveformMismatchOnNWave** command replaces the **tsDisplayWaveformMismatchOnNWave** command.

Syntax

```
tfgDisplayWaveformMismatchOnNWave -win window
```

Argument

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDisplayWaveformMismatchOnNWave -win $_tFlowView2
```

tfgDisplaySelectedWaveformMismatchFolderOnNWave

Description

Displays mismatch waveforms of the function blocks output pins.

Syntax

```
tfgDisplaySelectedWaveformMismatchFolderOnNWave -win <win_ID>
{<folderFullName#time>}
```

Argument

`-win window`

Specifies the current *Temporal Flow View* window.

`-folder`

Specifies the output pin's name of the function block.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDisplaySelectedWaveformMismatchFolderOnNWave -win $_tFlowView4
{tb.myreg.out[3:0]#400}
```

If the trace is in the transition-based mode, then you can add #T after the time as follows:

```
tfgDisplaySelectedWaveformMismatchFolderOnNWave -win $_tFlowView4
{tb.myreg.out[3:0]#400#T}
```

tfgDisplaySelectedWaveformMismatchNodeOnNWave

Description

Displays mismatch waveforms of the function blocks input pins.

Syntax

```
tfgDisplaySelectedWaveformMismatchNodeOnNWave -win <win_ID>
-folder {<folderFullName#time>} {NodeFullName}
```

Argument

`-win window`

Specifies the current *Temporal Flow View* window.

`-folder`

Specifies the input pin's name of the function block.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDisplaySelectedWaveformMismatchNodeOnNWave -win $_tFlowView4
-folder {tb.myreg.out[2:1]#400} {tb.myreg.sel}
```

If the trace is in the transition-based mode, then you can add #T after the time as follows:

```
tfgDisplaySelectedWaveformMismatchNodeOnNWave -win $_tFlowView4
-folder {tb.myreg.out[2:1]#400#T} {tb.myreg.sel}
```

tfgDrag

Description

Drags the selected signals in the *Temporal Flow View* window.

Syntax

```
tfgDrag -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDrag -win $_tFlowView3
```

tfgDrop

Description

Drops the selected signals from other window to the *Temporal Flow View* window.

Syntax

```
tfgDrop -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDrop -win $_tFlowView4
```

tfgDumpLibCellInfo

Description

Writes pin information for the specified library to a file.

NOTE: The `tfgDumpLibCellInfo` command replaces the `tsDumpLibCellInfo` command.

Syntax

```
tfgDumpLibCellInfo -filename filename -libname libraryName
-libpath libraryPath
```

Argument

`-filename filename`

Specifies the file that the library information is saved to.

`-libname libraryName`

Specifies the name of the library.

`-libpath libraryPath`

Specifies the path of the library.

Example

```
tfgDumpLibCellInfo -filename cellinfo.log -libname
tc280c_gck.lib_NOMIN15 -libpath ./SymDB
```

tfgEditBookMark

Description

Edits the bookmark item.

NOTE: The `tfgEditBookMark` command replaces the `tsEditBookMark` command.

Syntax

```
tfgEditBookMark -win window -mru number -name new_name
```

Argument

`-mru number`

Specifies the bookmark index.

`-name new_name`

Specifies the new bookmark name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgEditBookMark -win $_tFlowView2 -mru 1 -name "bm1"
```

tfgFit

Description

Fits to the whole view.

NOTE: The `tfgFit` command replaces the `tsFit` command.

Syntax

```
tfgFit -win window
```

Argument

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFit -win $_tFlowView2
```

tfgFolderClick

Description

Clicks the specified folder on *Temporal Flow View*.

NOTE: The `tfgFolderClick` command replaces the `tsFolderClick` command.

Syntax

```
tfgFolderClick -win window folder_name
```

Argument

`folder_name`

Specifies the folder name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderClick -win $_tFlowView2 "top.cpu#50"
```

tfgFolderCollapse

Description

Collapses the specified folder in the *Temporal Flow View* window.

Syntax

```
tfgFolderCollapse –win window {folderName}
```

Argument

`folderName`

Specifies the folder name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderCollapse -win $_tFlowView5 {top.aux. pin1 #1000}
```

tfgFolderDisplay

Description

Displays the specified folder.

NOTE: The `tfgFolderDisplay` command replaces the `tsFolderDisplay` command.

Syntax

```
tfgFolderDisplay -win window [folder_name]
```

Argument

`folder_name`

Specifies the folder name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderDisplay -win $_tFlowView2 "top.cpu#50"
```

tfgFolderDisplayFaninRegistersOnNWave

Description

Traces the specified folder by register and shows the result in the *nWave* window.

NOTE: The `tfgFolderDisplayFaninRegistersOnNWave` command replaces the `tsFolderDisplayFaninRegistersOnNWave` command.

Syntax

```
tfgFolderDisplayFaninRegistersOnNWave -win window [folder_name]
[-active]
```

Argument

`-active`

Traces the active only.

`folder_name`

Specifies the folder name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderDisplayFaninRegistersOnNWave -win $_tFlowView2
"top.cpu#50" -active
```

tfgFolderDisplayFaninSignalsOnNWave

Description

Traces the specified folder by statement and shows result on *nWave*.

NOTE: The `tfgFolderDisplayFaninSignalsOnNWave` command replaces the `tsFolderDisplayFaninSignalsOnNWave` command.

Syntax

```
tfgFolderDisplayFaninSignalsOnNWave -win window [folder_name]
[-active]
```

Argument

`-active`

Traces the active only.

`folder_name`

Specifies the folder name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderDisplayFaninSignalsOnNWave -win $_tFlowView2
"top.cpu#50" -active
```

tfgFolderDisplayOnNWave

Description

Synchronizes *Temporal Flow View* with *nWave* (generated by *nWave* menu).

NOTE: The `tfgFolderDisplayOnNWave` command replaces the `tsFolderDisplayOnNWave` command.

Syntax

```
tfgFolderDisplayOnNWave -win window [folder_name]
```

Argument

`folder_name`

Specifies the folder name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderDisplayOnNWave -win $_tFlowView2 "top.cpu#50"
```

tfgFolderExpand

Description

Expands the specified folder in the *Temporal Flow View* window.

Syntax

```
tfgFolderExpand –win window {folderName}
```

Argument

`folderName`

Specifies the folder name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderExpand -win $_tFlowView5 {top.aux. pin1 #1000}
```

tfgFolderMove

Description

Moves the specified folder.

NOTE: The `tfgFolderMove` command replaces the `tsFolderMove` command.

Syntax

```
tfgFolderMove -win window [folder_name] -level number -pos number
```

Argument

`folder_name`

Specifies the folder name.

`-level number`

Specifies the position (x axis).

`-pos number`

Specifies the position (y axis).

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderMove -win $_tFlowView2 "top.cpu#50" -pos 10
```

tfgFolderUndisplay

Description

Hides the specified folder.

NOTE: The `tfgFolderUndisplay` command replaces the `tsFolderUndisplay` command.

Syntax

```
tfgFolderUndisplay -win window [folder_name]
```

Argument

`folder_name`

Specifies the folder name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgFolderUndisplay -win $_tFlowView2 "top.cpu#50"
```

tfgForward

Description

Forwards to the next snapshot of window.

NOTE: The `tfgForward` command replaces the `tsForward` command.

Syntax

```
tfgForward -win window
```

Argument

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgForward -win $_tFlowView2
```

tfgGenerate

Description

Creates a *Temporal Flow View* window.

NOTE: The `tfgGenerate` command replaces the `tsGenerate` command.

Syntax

```
tfgGenerate -scope -ref -startWithStmt -schFG [-bdb_load]
[-bdb_incr] [-bdb_load_scope] [-cellModel 0|1|2|3] [-incr]
[-bboxEmptyModule 0|1] [-loopUnroll num] [-clockSkew value]
[-bboxModule 0|1] [-bboxModuleFile] [-symLibCellList]
[-symLibCellListFile] [-simModuleList] [-simModuleListFile]
[-bboxLib] [-bboxLibFile] [-apimem] [-bboxSysTaskFile]
[-macroCell] [-macroCellFile] [-mergeRange] [-traceflattenedMDA]
```

Argument

`-apimem`

Specifies a file containing the PLI memory definition.

`-bboxEmptyModule 0|1`

When set to 1, empty modules are set as black boxes.

`-bboxLib`

Specifies the library in which all modules/functions are set as black boxes.

`-bboxLibFile`

Specifies a file containing a set of the library in which all modules/functions are set as black boxes.

`-bboxModule 0|1`

Specifies modules to be set as black boxes.

`-bboxModuleFile`

Specifies a file containing a set of modules that are set as black boxes.

`-bboxSysTaskFile`

Specifies a file containing a set of system tasks that are set as black boxes.

`-bdb_incr`

Loads the Behavior Database (BDB) incrementally. This option works only when the `-bdb_load` option is specified.

`-bdb_load`

Specifies the Behavior Database (BDB) file to perform Behavior Analysis. This option must be the first option; otherwise, it is ignored. If the first option is `-bdb_load`, other Behavior Analysis options except `-bdb_incr` and `-bdb_load_scope` are ignored.

`-bdb_load_scope`

Specifies the path to load the top module of the Behavior Database (BDB) file. This option works only when the `-bdb_load` option is specified.

`-cellModel 0|1|2|3|4`

Specifies the source for a library cell definition. The following options are available:

0: symbol library first

1: symbol library only

2: simulation library only

3: simulation library first

4: automatic mode

`-clockSkew` *`value`*

Specifies the maximum value for clock skew (typically in gate level design after clock tree synthesis). The value should be 0 or a positive integer.

`-incr`

Performs incremental Behavior Analysis and opens the *Temporal Flow View* window.

`-loopUnroll` *`num`*

Specifies the maximum number of for loops to unroll. The number should be 0 or a positive integer.

`-macroCell`

Specifies modules to be treated as macro cell.

`-macroCellFile`

Specifies a file containing a set of modules that are treated as macro cell.

`-mergeRange`

Specifies whether to merge range in SFG (display).

`-ref`

Specifies the reference signal which is added to a Temporal Flow View.

`-schFG`

This option must be used with `-startWithStmt`. If this Tcl command contains `-startWithStmt -schFG`, creates Temporal Flow View.

`-scope`

Specifies the scope on which the *Behavior Analysis* is applied.

`-simModuleList`

Treats the listed cell using simulation model.

`-simModuleListFile`

Treats the listed cell using simulation model.

`-startWithStmt`

Creates a *Compact Temporal Flow View*.

`-symLibCellList`

Treats the cells listed as symbol library cells.

`-symLibCellListFile`

Treats the listed cells as symbol library cells.

`-traceflattenedMDA`

Specifies whether to display multiple dimensional arrays as memories or flattened multiple dimensional arrays into buses.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgGenerate -bdb_load work.lib++/work.bdb -ref signal#time

tfgGenerate -bdb_load work.lib++/work.bdb -bdb_incr -ref
signal#time

tfgGenerate -bdb_load work.lib++/work.bdb -bdb_load_scope
{system.i_cpu} -ref signal#time

tfgGenerate -bdb_load work.lib++/work.bdb -bdb_load_scope
{system.i_cpu} -bdb_incr -ref signal#time

tfgGenerate -scope {wave} -ref
{wave.TEST_CFD3SFX1_wave.C001.QN#39200.01} -startWithStmt -schFG
-loopUnroll 0 -clockSkew 0 -bboxEmptyModule 1 -cellModel 0
```

tfgGetWindow

Description

Gets the *Temporal Flow View* window address and returns its value.

NOTE: The `tfgGetWindow` command replaces the `tsGetWindow` command.

Syntax

```
set windId [tfgGetWindow]
```

NOTE: The open and close brackets ([]) for `tfgGetWindow` is required.

Argument

`winId`

A user defined variable storing the address of `tfgGetWindow` to return.

Value Returned

Window address if successful; otherwise, returns 0.

Example

```
srcSourceCodeView
  srcResizeWindow 16 117 804 500
  debImport "-f" "run.f"
  debLoadSimResult /verdi/home/jack_hung/jack/case/rtl.fsdb
  tfgGenerate -scope {system} -ref {system.i_cpu.i_ALUB.ALU#2500}
-startWithStm
-schFG -loopUnroll 0 -clockSkew 0 -bboxEmptyModul
  set winId [tfgGetWindow]
  tfgCloseViewer -win $winId
```

`set winId [tfgGetWindow]` means assign the return value of *tfgGetWindow* to the variable *winId*.

In `tfgCloseViewer -win $winId`, *winId* starting with the dollar sign ($) means get the content stored in the variable *winId*.

tfgGotoBookMark

Description

Goes to the bookmark item.

NOTE: The `tfgGotoBookMark` command replaces the `tsGotoBookmark` command.

Syntax

```
tfgGotoBookMark -win window -mru number
```

Argument

`-mru number`

Specifies the bookmark index.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgGotoBookMark -win $_tFlowView2 -mru 1
```

tfgLevelClick

Description

Clicks the interval of Temporal Flow View.

NOTE: The `tfgLevelClick` command replaces the `tsLevelClick` command.

Syntax

```
tfgLevelClick -win window -level levelNum
```

Argument

`-level levelNum`

Specifies the number of intervals.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgLevelClick -win $_tFlowView2 -level 999999
```

tfgLoadForTrmis

Description

Loads the second FSDB file for trace mismatch.

NOTE: The `tfgLoadForTrmis` command replaces the `tsLoadForTrmis` command.

Syntax

```
tfgLoadForTrmis [filename]
```

Argument

`filename`

Specifies the FSDB file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgLoadForTrmis mismatch.fsdb
```

tfgMemActTraceFromNTrace

Description

Memory trace (generated from *nTrace*).

NOTE: The `tfgMemActTraceFromNTrace` command replaces the `tsMemActTraceFromNTrace` command.

Syntax

```
tfgMemActTraceFromNTrace -win window -sig sig_name -index number
-time number
```

Argument

`-index number`

Specifies the memory address.

`-sig sig_name`

Specifies the memory name.

`-time number`

Specifies the time.

`-win window`

Specifies the memory window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgMemActTraceFromNTrace -win $_nTrace1 -sig "top.mem" -index 16
-time 1000
```

tfgMemClose

Description

Closes the memory window.

NOTE: The `tfgMemClose` command replaces the `tsMemClose` command.

Syntax

```
tfgMemClose -win window
```

Argument

`-win` *window*

Specifies the memory window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgMemClose -win $_tMemory1
```

tfgMemSaveContent

Description

Saves the memory content.

NOTE: The `tfgMemSaveContent` command replaces the `tsMemSaveContent` command.

Syntax

```
tfgMemSaveContent -win window [filename]
```

Argument

`filename`

Specifies the file name.

`-win` *window*

Specifies the memory window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgMemSaveContent -win $_tMemory1 memory.data
```

tfgMemSetOpt

Description

Sets the memory window options.

NOTE: The `tfgMemSetOpt` command replaces the `tsMemSetOpt` command.

Syntax

```
tfgMemSetOpt -win window -numWords number -addrColWidth number
-valueColWidth number
```

Argument

`-addrColWidth` *`number`*

Specifies the address column width.

`-numWords` *`number`*

Specifies the number of words.

`-valueColWidth` *`number`*

Specifies the value column width.

`-win` *`window`*

Specifies the memory window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgMemSetOpt -win $_tMemory1 -numWords 16 -addrColWidth 100
-valueColWidth 100
```

tfgMemSyncCursorTime

Description

Synchronizes the cursor time.

NOTE: The **tfgMemSyncCursorTime** command replaces the **tsMemSyncCursorTime** command.

Syntax

```
tfgMemSyncCursorTime -win window on|off
```

Argument

`-win window on|off`

Specifies the memory window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgMemSyncCursorTime -win $_tMemory1 on
```

tfgNewViewer

Description

Creates a new *Temporal Flow View* from the existing window with the same root nodes.

NOTE: The `tfgNewViewer` command replaces the `tsNewViewer` command.

Syntax

```
tfgNewViewer -win window
```

Argument

`-win window`

Specifies the existing *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNewViewer -win $_tFlowView2
```

tfgNodeAddAliasFile

Description

Adds the alias name from file.

NOTE: The `tfgNodeAddAliasFile` command replaces the `tsNodeAddAliasFile` command.

Syntax

```
tfgNodeAddAliasFile -win window -folder folder_name -file filename
[signal_name]
```

Argument

`-file` *`filename`*

Specifies the alias file name.

`-folder` *`folder_name`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
TfgNodeAddAliasFile -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -file name.alias
```

tfgNodeAddAliasProgram

Description

Adds the alias name from program.

NOTE: The `tfgNodeAddAliasProgram` command replaces the `tsNodeAddAliasProgram` command.

Syntax

```
tfgNodeAddAliasProgram -win window -folder folder_name
-program [signal_name]
```

Argument

`-folder folder_name`

Specifies the folder name.

`-program`

Specifies the alias program name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeAddAliasProgram -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -program nWave
```

tfgNodeClick

Description

Clicks on the specified signal on Temporal Flow View.

NOTE: The `tfgNodeClick` command replaces the `tsNodeClick` command.

Syntax

```
tfgNodeClick -win window -folder folder_name [signal_name]
```

Argument

`-folder folder_name`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeClick -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNodeDisplay

Description

Displays the specified fan-in signal.

NOTE: The `tfgNodeDisplay` command replaces the `tsNodeDisplay` command.

Syntax

```
tfgNodeDisplay -win window -folder folderName [signal_name]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplay -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNodeDisplayClkOnNWave

Description

Displays the clock of given signal on *nWave*.

NOTE: The `tfgNodeDisplayClkOnNWave` command replaces the `tsNodeDisplayClkOnNWave` command.

Syntax

```
tfgNodeDisplayClkOnNWave -win window -folder folderName
[signal_name]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayClkOnNWave -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNodeDisplayFaninFit

Description

Fits the *Temporal Flow View* window to the specified signal.

NOTE: The `tfgNodeDisplayFaninFit` command replaces the `tsNodeDisplayFaninFit` command.

Syntax

```
tfgNodeDisplayFaninFit -win window -folder folderName
[signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayFaninFit -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.a"
```

tfgNodeDisplayFaninFitUndo

Description

Undo fit *Temporal Flow View* window to the specified signal.

NOTE: The `tfgNodeDisplayFaninFitUndo` command replaces the `tsNodeDisplayFaninFitUndo` command.

Syntax

```
tfgNodeDisplayFaninFitUndo -win window
```

Argument

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayFaninFitUndo -win $_tFlowView2
```

tfgNodeDisplayFaninRegister

Description

Traces one level fan-in register cone on the register flow graph.

NOTE: The `tfgNodeDisplayFaninRegister` command replaces the `tsNodeNodeDisplayFaninRegister` command.

Syntax

```
tfgNodeDisplayFaninRegister -win window -folder folderName
[signal_name]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayFaninRegister -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.a"
```

tfgNodeDisplayFaninRegisters

Description

Traces one level fan-in register cone on the register flow graph.

NOTE: The `tfgNodeDisplayFaninRegisters` command replaces the `tsNodeDisplayFaninRegisters` command.

Syntax

```
tfgNodeDisplayFaninRegisters -win window -folder folderName
[signal_name]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayFaninRegisters -win $_tFlowView2 -folder
"top.cpu#50" "top.cpu.a"
```

tfgNodeDisplayFaninRegistersOnNWave

Description

Traces the fan-in register and displays the results on *nWave*.

NOTE: The `tfgNodeDisplayFaninRegistersOnNWave` command replaces the `tsNodeDisplayFaninRegistersOnNWave` command.

Syntax

```
tfgNodeDisplayFaninRegistersOnNWave -win window -folder
folder_name [-active] [signal_name]
```

Argument

`-active`

Specifies trace active nodes only.

`-folder folder_name`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayFaninRegistersOnNWave -win $_tFlowView2 -folder
"top.cpu#50" "top.cpu.data" -active
```

tfgNodeDisplayFaninSignal

Description

Traces one level fan-in signal on the statement flow graph.

NOTE: The `tfgNodeDisplayFaninSignal` command replaces the `tsNodeDisplayFaninSignal` command.

Syntax

```
tfgNodeDisplayFaninSignal -win window -folder folderName
[signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayFaninSignal -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.a"
```

tfgNodeDisplayFaninSignals

Description

Traces one level fan-in signals on statement flow graph.

NOTE: The `tfgNodeDisplayFaninSignals` command replaces the `tsNodeDisplayFaninSignals` command.

Syntax

```
tfgNodeDisplayFaninSignals -win window -folder folderName
[signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayFaninSignals -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.a"
```

tfgNodeDisplayFaninSignalsOnNWave

Description

Traces fan-in signals and shows result on *nWave*.

NOTE: The `tfgNodeDisplayFaninSignalsOnNWave` command replaces the `tsNodeDisplayFaninSignalsOnNWave` command.

Syntax

```
tfgNodeDisplayFaninSignalsOnNWave -win window -folder folderName
[-active] [signal_name]
```

Argument

`-active`

Specifies trace active nodes only.

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayFaninSignalsOnNWave -win $_tFlowView2 -folder
"top.cpu#50" "top.cpu.data" -active
```

tfgNodeDisplayOneLevelActive

Description

Traces one level active fan-in signals on the statement flow graph.

NOTE: The `tfgNodeDisplayOneLevelActive` command replaces the `tsNodeDisplayOneLevelActive` command.

Syntax

```
tfgNodeDisplayOneLevelActive -win window -folder folderName
signal_name
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayOneLevelActive -win $_tFlowView2 -folder
"top.cpu#50" "top.cpu.a"
```

tfgNodeDisplayOnNWave

Description

Shows the specified signal on *nWave*.

NOTE: The **tfgNodeDisplayOnNWave** command replaces the **tsNodeDisplayOnNWave** command.

Syntax

```
tfgNodeDisplayOnNWave -win window -folder folder_name
[signal_name]
```

Argument

`-folder folder_name`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayOnNWave -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNodeDisplaySyncSignalOnNWave

Description

Shows the synchronized signal on *nWave*.

NOTE: The `tfgNodeDisplaySyncSignalOnNWave` command replaces the `tsNodeDisplaySyncSignalOnNWave` command.

Syntax

```
tfgNodeDisplaySyncSignalOnNWave -win window -folder folderName
[signalName]
```

Argument

`-folder folderName`

Specifies the folder name.

`signalName`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplaySyncSignalOnNWave -win $_tFlowView2
-folder "top.cpu#50" "top.cpu.data"
```

tfgNodeDisplayWholeHoldGroup

Description

Displays the whole fan-in group of the hold register.

Syntax

```
tfgNodeDisplayWholeHoldGroup -win window -folder {folderName}
{signalName}
```

Argument

`-folder folderName`

Specifies the folder name.

`signalName`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeDisplayWholeHoldGroup -win $_tFlowView5
-folder {top.cpu#50} {top.cpu.data}
```

tfgNodeFaninManager

Description

Opens the fan-in manager window.

NOTE: The `tfgNodeFaninManager` command replaces the `tsNodeFaninManager` command.

Syntax

```
tfgNodeFaninManager -win window -folder folderName [signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeFaninManager -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNodeOpenRegisterFlowGraph

Description

Creates a new register flow view and uses the specified signal as root.

NOTE: The `tfgNodeOpenRegisterFlowGraph` command replaces the `tsNodeOpenRegisterFlowGraph` command.

Syntax

```
tfgNodeOpenRegisterFlowGraph -win window -folder folderName
[signal_name]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeOpenRegisterFlowGraph -win $_tFlowView2 -folder
"top.cpu#50" "top.cpu.data"
```

tfgNodeOpenStmtFlowGraph

Description

Creates a new statement flow view and uses the specified signal as root.

NOTE: The `tfgNodeOpenStmtFlowGraph` command replaces the `tsNodeOpenStmtFlowGraph` command.

Syntax

```
tfgNodeOpenStmtFlowGraph -win window -folder folderName
[signal_name] [-semafg]
```

Argument

`-folder folderName`

Specifies the folder name.

`-semafg`

Creates a new schematic flow view.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeOpenStmtFlowGraph -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -semafg
```

tfgNodePartialBusTrace

Description

Traces the partial bus.

NOTE: The `tfgNodePartialBusTrace` command replaces the `tsNodePartialBusTrace` command.

Syntax

```
tfgNodePartialBusTrace -win window -folder folderName
[signal_name] -startBit number -endBit number
```

Argument

`-endBit`

Specifies the end bit range.

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-startBit`

Specifies the start bit range.

`-win`

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodePartialBusTracer -win $_tFlowView2 -startBit 31 -endBit 24
-folder "top.cpu#50" "top.cpu.bus"
```

tfgNodeRegroup

Description

Regroups a bus grouped node that has been ungrouped.

NOTE: The `tfgNodeRegroup` command replaces the `tsNodeRegroup` command.

Syntax

```
tfgNodeRegroup -win window -folder folderName [signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeRegroup -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNodeRemoveAlias

Description

Removes the alias.

NOTE: The `tfgNodeRemoveAlias` command replaces the `tsNodeRemoveAlias` command.

Syntax

```
tfgNodeRemoveAlias -win window -folder folderName [signal_name]
```

Argument

`folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeRemoveAlias -win $_tFlowView2 -folder
"top.cpu#50"."top.cpu.data"
```

tfgNodeScTrActValue

Description

Traces the scan-chain active value.

NOTE: The `tfgNodeScTrActValue` command replaces the `tsNodeScTrActValue` command.

Syntax

```
tfgNodeScTrActValue -win window -folder folderName [signal_name]
[-wholePath]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-wholePath`

Traces the whole scan-chain path.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeScTrActValue -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -wholePath
```

tfgNodeSetBackgroundColor

Description

Sets the background color of the specified fan-in signal.

NOTE: The `tfgNodeSetBackgroundColor` command replaces the `tsNodeSetBackgroundColor` command.

Syntax

```
tfgNodeSetBackgroundColor -win window -folder folderName
[signal_name] -color color_name
```

Argument

`-color`

Specifies background color.

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeSetBackgroundColor -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -color red
```

tfgNodeSetForegroundColor

Description

Sets the foreground color of the specified fan-in signal.

NOTE: The `tfgNodeSetForegroundColor` command replaces the `tsNodeSetForegroundColor` command.

Syntax

```
tfgNodeSetForegroundColor -win window -folder folderName
[signal_name] -color color_name
```

Argument

`-color`

Specifies the background color.

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeSetForegroundColor -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -color red
```

tfgNodeSetNotation

Description

Sets the node notation.

NOTE: The `tfgNodeSetNotation` command replaces the `tsNodeSetNotation` command.

Syntax

```
tfgNodeSetNotation -win window -folder folderName -format radix
[signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`-format radix`

Specifies the radix [unsigned/2comp/1comp/magnitude].

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeSetNotation -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -format unsigned
```

tfgNodeSetRadix

Description

Sets the node radix.

NOTE: The `tfgNodeSetRadix` command replaces the `tsNodeSetRadix` command.

Syntax

```
tfgNodeSetRadix -win window -folder folderName -format radix
[signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`-format radix`

Specifies the radix [bin/oct/hex/dec/asc/754].

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeSetRadix -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -format bin
```

tfgNodeShowActiveStatement

Description

Traces one level fan-in signals on statement flow graph.

NOTE: The `tfgNodeShowActiveStatement` command replaces the `tsNodeShowActiveStatement` command.

Syntax

```
tfgNodeShowActiveStatement -win window -folder folderName
[signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeShowActiveStatement -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.a"
```

tfgNodeTraceActTrans

Description

Traces the active transition on the statement flow view.

NOTE: The `tfgNodeTraceActTrans` command replaces the `tsNodeTraceActiveTrans` command.

Syntax

```
tfgNodeTraceActTrans -win window -folder folderName [signalName]
[-stopLevel number|-stopTime number|-stopAtReg]
```

Argument

`-folder folderName`

Specifies the folder name.

`signalName`

Specifies the signal name.

`-stopAtReg`

Specifies whether to stop at the register boundary.

`-stopLevel number`

Specifies the stop level.

`-stopTime number`

Specifies the stop time.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeTraceActTrans -win $_tFlowView2 -folder "top.cpu#50"
-stopLevel 10 "top.cpu.data"
```

tfgNodeTraceActiveValue

Description

Traces active value on the statement flow view.

NOTE: The `tfgNodeTraceActiveValue` command replaces the `tsNodeTraceActiveValue` command.

Syntax

```
tfgNodeTraceActiveValue -win window -folder folderName
[signal_name] [-stopLevel number| -stopTime number]
```

```
[-trSameValue TRUE|FALSE | -trSingleAct TRUE|FALSE | -trSingleTran
TRUE|FALSE | -stopTrCtrlHasTran TRUE|FALSE]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-stopLevel` *`number`*

Specifies the stop level.

`-stopTime` *`number`*

Specifies the stop time.

`-stopTrCtrlHasTran TRUE|FALSE`

Stops when control signal has transition.

`-trSameValue TRUE|FALSE`

Traces the node with the same value as the fan-out signal.

`-trSingleAct TRUE|FALSE`

Traces the fan-in node when there is only one active signal.

`-trSingleTran TRUE|FALSE`

Traces the fan-in node when there is only one transition signal.

`-win` *`window`*

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeTraceActiveValue -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -stopLevel 10 -trSameValue TRUE
```

tfgNodeTraceAgainByShowingDetails

Description

Traces again by showing cycle by cycle details.

NOTE: The `tfgNodeTraceAgainByShowingDetails` command replaces the `tsNodeTraceAgainByShowingDetails` command.

Syntax

```
tfgNodeTraceAgainByShowingDetails -win window -folder folderName
[signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeTraceAgainByShowingDetails -win $_tFlowView2 -folder
"top.cpu#50" "top.cpu.data"
```

tfgNodeTraceDriver

Description

Traces node driver and zooms in to specified node.

NOTE: The `tfgNodeTraceDriver` command replaces the `tsNodeTraceDriver` command.

Syntax

```
tfgNodeTraceDriver -win window -folder folderName [signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeTraceDriver -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNodeTraceThisRegister

Description

Traces active value on the *Temporal Register View*.

NOTE: The `tfgNodeTraceThisRegister` command replaces the `tsNodeTraceThisRegister` command.

Syntax

```
tfgNodeTraceThisRegister -win window -folder folderName
[signal_name] [-stopLevel number|-stopTime number]
[-trSameValue TRUE|FALSE | -trSingleAct TRUE|FALSE |
-trSingleTran TRUE|FALSE | -stopTrCtrlHasTran TRUE|FALSE]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-stopLevel` *number*

Specifies the stop level.

`-stopTime` *number*

Specifies the stop time.

`-stopTrCtrlHasTran TRUE|FALSE`

Stops when control signal has transition.

`-trSameValue TRUE|FALSE`

Traces the node with the same value as the fan-out signal.

`-trSingleAct TRUE|FALSE`

Traces the fan-in node when there is only one active signal.

`-trSingleTran TRUE|FALSE`

Traces the fan-in node when there is only one transition signal.

`-win` *window*

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeTraceThisRegister -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -stopLevel 10 -trSameValue TRUE
```

tfgNodeTrMisByComp

Description

Traces active mismatch node of the specified node.

NOTE: The `tfgNodeTrMisByComp` command replaces the `tsNodeTrMisByComp` command.

Syntax

```
tfgNodeTrMisByComp -win window -folder folderName [signal_name]
[-stopLevel number|-stopTime number] [-JmpToEarliest]
```

Argument

`-folder folderName`

Specifies the folder name.

`-JmpToEarliest`

Jumps to the earliest mismatch time and continues trace.

`signal_name`

Specifies the signal name.

`-stopLevel`

Specifies the stop by level.

`-stopTime`

Specifies the stop by time.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeTrMisByComp -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -stopLevel 10
```

tfgNodeUndisplay

Description

Hides the specified fan-in signal.

NOTE: The `tfgNodeUndisplay` command replaces the `tsNodeUndisplay` command.

Syntax

```
tfgNodeUndisplay -win window -folder folderName [signal_name]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeUndisplay -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNodeUngroup

Description

Ungroups a bus grouped node.

NOTE: The `tfgNodeUngroup` command replaces the `tsNodeUngroup` command.

Syntax

```
tfgNodeUngroup -win window -folder folderName [signal_name]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNodeUngroup -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgNWNodeAddSpreadSheet

Description

Adds signal with time to spreadsheet (generated from *nWave* menu).

NOTE: The `tfgNWNodeAddSpreadSheet` command replaces the `tsNWNodeAddSpreadSheet` command.

Syntax

```
tfgNWNodeDisplayCtrl [signal_name list]
```

Argument

`signal_name`

Specifies the signal name list.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNWNodeAddSpreadSheet "top.cpu.a#50" "top.cpu.b#100"
```

tfgNWNodeDisplayClk

Description

Shows the clock of the given signal by the *nWave* (generated by *nWave* menu).

NOTE: The `tfgNWNodeDisplayClk` command replaces the `tsNWNodeDisplayClk` command.

Syntax

```
tfgNWNodeDisplayClk -win window -folder folderName
[signal_name_with_time]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signal_name_with_time`

Specifies the signal name and time.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNWNodeDisplayClk "top.cpu.data#50"
```

tfgNWNodeDisplayCtrl

Description

Shows the control signal by *nWave* (generated by *nWave* menu).

NOTE: The `tfgNNWNodeDisplayCtrl` command replaces the `tsNWNodeDisplayCtrl` command.

Syntax

```
tfgNWNodeDisplayCtrl [signal_name_with_time]
```

Argument

`signal_name_with_time`

Specifies the signal name and time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNWNodeDisplayCtrl "top.cpu.data#50"
```

tfgNWNodeDisplayDP

Description

Shows the data path by *nWave* (generated by *nWave* menu).

NOTE: The `tfgNWNodeDisplayDP` command replaces the `tsNWNodeDisplayDP` command.

Syntax

```
tfgNWNodeDisplayDP [signal_name_with_time]
```

Argument

`signal_name_with_time`

Specifies the signal name and time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNWNodeDisplayDP "top.cpu.data#50"
```

tfgNWNodeDisplayFaninRegister

Description

Shows the fan-in registers by *nWave* (generated by the *nWave* menu).

NOTE: The `tfgNWNodeDisplayFaninRegister` command replaces the `tsNWNodeDisplayFaninRegister` command.

Syntax

```
tfgNWNodeDisplayFaninRegister [signal_name_with_time] [-active]
[-out tclList] [-format inout|out]
```

Argument

`-active`

Active node only.

`-format inout|out`

Specifies the output format for the dump results. The formats include: **inout** and **out**. The **inout** format shows the result list that contains the input signal name and the result signal name as *input_signal result1_signal result2_signal...resultN_signal*. The **out** format shows the results list that contains the result signal name as *result1_signal result2_signal...resultN_signal*.

`-out tclList`

Specifies an existing Tcl list to save the trace results to.

`signal_name_with_time`

Specifies the signal name and time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debLoadSimResult /dq2/qa/demo/61/verilog/rtl/rtl.fsdb
tfgBehaviorAnalysis -incr
set results_list {}
tfgNWNodeDisplayFaninRegister
{system.i_cpu.i_ALUB.i_alu.a[7:0]#950} -out results_list
puts $results_list

set result_list {}
tfgNWNodeDisplayFaninRegister "top.cpu.data#50" -out result_list
-format inout
```

tfgNWNodeDisplayFaninSignal

Description

Shows the fan-in signals on *nWave* (generated by the *nWave* menu).

NOTE: The **tfgNWNodeDisplayFaninSignal** command replaces the **tsNWNodeDisplayFaninSignal** command.

Syntax

```
tfgNWNodeDisplayFaninSignal [signal_name_with_time] [-active]
```

Argument

`-active`

Active node only.

`signal_name_with_time`

Specifies the signal name and time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNWNodeDisplayFaninSignal "top.cpu.data#50" -active
```

tfgNWNodeDisplayTraceActive

Description

Traces active value and shows result on *nWave* (generated by *nWave* menu).

NOTE: The `tfgNWNodeDisplayTraceActive` command replaces the `tsNWNodeDisplayTraceActive` command.

Syntax

```
tfgNWNodeDisplayTraceActive [signal_name_with_time] -mode
[reg|signal] [-stopLevel number|-stopTime number]
```

Argument

`-mode reg|signal`

Trace by register or signal.

`signal_name_with_time`

Specifies the signal name and time.

`-stopLevel number`

Specifies the stop level.

`-stopTime number`

Specifies the stop at time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgNWNodeDisplayTraceActive "top.cpu#50" -mode signal -stopLevel
10
```

tfgOpenRegisterFlowGraph

Description

Creates a new RFG and adds all root signals on the given window to it.

NOTE: The `tfgOpenRegisterFlowGraph` command replaces the `tsOpenRegisterFlowGraph` command.

Syntax

```
tfgOpenRegisterFlowGraph -win window
```

Argument

`-win` *`window`*

Specifies the window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgOpenRegisterFlowGraph -win $_tFlowView2
```

tfgPrint

Description

Prints the content in the *Temporal Flow View* window.

Syntax

```
tfgPrint [-win window] [-header header] [-footer footer]
-orient landscape|portrait -paper paperSize {[-file printFile]|
[-printer printName]}
```

Argument

`[-file printFile`

Specifies the file that the *Temporal Flow View* prints the window content to.

`-footer footer`

Specifies the information to be presented on the bottom of the printout.

`-header header`

Specifies the information to be presented on the top of the printout.

`-orient landscape|portrait`

Specifies the source of the paper. The orientation options are **landscape** and **portrait**.

`-paper paperSize`

Specifies the paper size for the printed page. The size options are **A4**, **A3**, **A2**, **A1**, **A0**, **L**, **B**, **C**, **D**, and **E**.

`-printer printName`

Specifies the printer that the *Temporal Flow View* sends the window content to. The default is **lp**.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgPrint -win $_tFlowView4 -orient landscape -paper L –printer lp
```

tfgRefresh

Description

Refreshes all signals in the *Temporal Flow View* window.

Syntax

```
tfgRefresh –win window
```

Argument

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgRefresh -win $_tFlowView3
```

tfgResetNodesColor

Description

Resets the color of the specified nodes.

NOTE: The `tfgResetNodesColor` command replaces the `tsResetNodesColor` command.

Syntax

```
tfgResetNodesColor -win window -folder folderName [signalName]
```

Argument

`-folder` *`folderName`*

Specifies the folder name.

`signalName`

Specifies the signal name.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgResetNodesColor -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgSCHDisplayAll

Description

Adds the current trace results from the *Temporal Flow View* window to the existing *nSchema* flattened window. If an *nSchema* flattened window does not exist, the current trace results are added to a new *nSchema* flattened window.

NOTE: The `tfgSCHDisplayAll` command replaces the `tsSCHDisplayAll` command.

Syntax

```
tfgSCHDisplayAll -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSCHDisplayAll -win $_tFlowView2
```

tfgSCHDisplayFaninRegister

Description

Shows the fan-in register cone of given signal on a new *nSchema* window.

NOTE: The `tfgSCHDisplayFaninRegister` command replaces the `tsSCHDisplayFaninRegister` command.

Syntax

```
tfgSCHDisplayFaninRegister -win window -folder folderName
[signalName]
```

Argument

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSCHDisplayFaninRegister -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.a"
```

tfgSCHShowDetailRTL

Description

Shows detailed RTL on a functional block of *nSchema*.

NOTE: The **tfgSCHShowDetailRTL** command replaces the **tsSCHShowDetailRTL** command.

Syntax

```
tfgSCHShowDetailRTL -win window -folder folderName [signal_name]
```

Argument

`-folder foldername`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSCHShowDetailRTL -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data"
```

tfgScteCloseWin

Description

Closes the scan-chain timing error report window.

NOTE: The `tfgScteCloseWin` command replaces the `tsScteCloseWin` command.

Syntax

```
tfgScteCloseWin -win window [filename]
```

Argument

`filename`

Specifies the file name.

`-win window`

Specifies the scan-chain timing error report window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgScteCloseWin -win $_tScteRst1
```

tfgSetNodesColor

Description

Sets the specified nodes with given color.

NOTE: The `tfgSetNodesColor` command replaces the `tsSetNodesColor` command.

Syntax

```
tfgSetNodesColor -win window -folder folderName [signal_name]
-color colorName
```

Argument

`-color colorName`

Specifies the background color.

`-folder folderName`

Specifies the folder name.

`signal_name`

Specifies the signal name.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSetNodesColor -win $_tFlowView2 -folder "top.cpu#50"
"top.cpu.data" -color red
```

tfgSetOption

Description

Sets the specified options in the *Temporal Flow View* window.

NOTE: The `tfgSetOption` command replaces the `tsSetOption` command.

Syntax

```
tfgSetOption -win window [-activeNodeOnly on|off] [-powerCellName
on|off] [-activeDataNodeOnly on|off] [-valueXNodeOnly on|off]
[-trigPathOnly on|off] [-showSrcOnTip on|off] [-hierName on|off]
[-sigName on|off] [-sigValue on|off] [-portName on|off] [-showTip
on|off] [-showHoldCycles on|off] [-showDetailsOfHold on|off]
[-syncWithNWave on|off] [-viewMap on|off] [-pan on|off]
[-showNSchema on|off] [-showNWave on|off]
```

Argument

`-activeDataNodeOnly on|off`

Turns the display of active data nodes *on* or *off*.

`-activeNodeOnly on|off`

Turns the display of active nodes *on* or *off*.

`-hierName on|off`

Turns the display of hierarchy names *on* or *off*.

`-pan on|off`

Turns the display of pan mode *on* or *off*.

`-portName on|off`

Turns the display of port names *on* or *off*.

`-powerCellName on|off`

Turns the display of power cell names *on* or *off*.

`-showDetailsOfHold on|off`

Turns the display of hold register details *on* or *off*.

`-showHoldCycles on|off`

Turns the display of hold cycles *on* or *off*.

`-showNSchema on|off`

Turns the display of adding selected signals in the *Temporal Flow View* window to the *nSchema* window *on* or *off*.

`-showNWave on|off`

Turns the display of adding selected signals in the *Temporal Flow View* window to the *nWave* window *on* or *off*.

`-showSrcOnTip on|off`

Turns the display of tips for selected signals in the *Temporal Flow View* window to the source code frame *on* or *off*.

`-showTip on|off`

Turns the display of tips *on* or *off*. Settings for the tip display location are available in the Tip section of the **Temporal Flow View -> View -> Display** page of the *Preferences* form.

`-sigName on|off`

Turns the display of signal names *on* or *off*.

`-sigValue on|off`

Turns the display of signal values *on* or *off*.

`-syncWithNWave on|off`

Turns the synchronization between the *Temporal Flow View* window and the *nWave* window *on* or *off*.

`-trigPathOnly on|off`

Turns the display of triggering paths *on* or *off*.

`-valueXNodeOnly on|off`

Turns the display of values with 'x' node *on* or *off*.

`-viewMap on|off`

Turns the display of view map *on* or *off*.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSetOption -win $_tFlowView2 -activeNodeOnly on
tfgSetOption -win $_tFlowView3 -powerCellName on
```

tfgSetPreference

Description

Sets *Temporal Flow View* preference setting that is updated to *novas.rc*.

NOTE: The `tfgSetPreference` command replaces the `tsSetPreference` command.

Syntax

```
tfgSetPreference [-tfgValueComputed colorId][-tfgValueNonComputed
colorId] [-tsAutoAdjustMargin true|false [leftMargin|rightMargin
Number]] [-trmisOption 1|2] [-delaySampling delayTime]
```

Argument

`-delaySampling delayTime`

Specifies the value (percentage of clock cycle) of delay sampling.

`-rightMargin/-leftMargin`

Specifies the right/left margin value for viewing. Used with `-tsAutoAdjustMargin false`.

`-tfgValueComputed`

Specifies the color for computed signals.

`-tfgValueNonComputed`

Specifies the color for non-computed signals.

NOTE: The Tcl options for specifying the color of computed signals and non-computed signals are only supported by *Siloti*.

`-trmisOption 1|2`

Specifies whether to provide an opportunity to modify the trace mismatch options every time the **Behavior Trace for Waveform Mismatch** command in the *Temporal Flow View* window is invoked. Default is *1*.

1: Opens the form to modify the trace mismatch options.

2: Performs trace mismatch with default option values.

`-tsAutoAdjustMargin true|false`

Turns on/off automatic margin adjustment on *Temporal Flow View*. True is the value for the enabled option. False is the value for the disabled option.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSetPreference -tfgValueComputed ID_YELLOW5
-tfgValueNonComputed ID_PURPLE7

tfgSetPreference -tsAutoAdjustMargin FALSE -leftMargin 70
-rightMargin 50

tfgSetPreference -trmisDelay 50 -trmisOption 2
```

tfgSetWindow

Description

Sets the given window as the active *Temporal Flow View*.

NOTE: The `tfgSetWindow` command replaces the `tsSetWindow` command.

Syntax

```
tfgSetWindow window
```

Argument

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

Active *Temporal Flow View* window.

Example

```
tfgSetWindow $_tFlowView3
```

tfgSetWindowTimeUnit

Description

Sets the options.

NOTE: The **tfgSetWindowTimeUnit** command replaces the **tsSetWindowTimeUnit** command.

Syntax

```
tfgSetWindowTimeUnit -win window [scale] [unit]
```

Argument

`scale`

Specifies the scale.

`unit`

Specifies the time unit.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSetWindowTimeUnit -win $_tFlowView2 10 ns
```

tfgShowBMEditor

Description

Shows the bookmark editor.

NOTE: The `tfgShowBMEditor` command replaces the `tsShowBMEditor` command.

Syntax

```
tfgShowBMEditor -win window
```

Argument

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgShowBMEditor -win $_tFlowView2
```

tfgShowBMNameDlg

Description

Adds or closes the bookmark name dialog.

NOTE: The `tfgShowBMNameDlg` command replaces the `tsShowBMNameDlg` command.

Syntax

```
tfgShowBMNameDlg -win window [-add] [-close]
```

Argument

`-add`

Adds the bookmark name dialog.

`-close`

Closes the bookmark name dialog.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgShowBMNameDlg -win $_tFlowView2 -add
```

tfgSkipBA

Description

Skips Behavior Analysis before it is triggered by the `debImport` Tcl command. When Behavior Analysis options are specified before loading the design using the `debImport` Tcl command, `debImport` triggers Behavior Analysis. `tfgSkipBA` can be executed before `debImport` to skip performing Behavior Analysis.

NOTE: The `tfgSkipBA` command replaces the `tsSkipBA` command.

Syntax

```
tfgSkipBA
```

Argument

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSkipBA
debImport "-lib" "work"
```

tfgSpreadSheetAdd

Description

Adds signal with time to the spreadsheet.

NOTE: The `tfgSpreadSheetAdd` command replaces the `tsSpreadSheetAdd` command.

Syntax

```
tfgSpreadSheetAdd [signal_name list]
```

Argument

`signal_name` *`list`*

Specifies the signal name list.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetAdd "top.cpu.a#50" "top.cpu.b#100"
```

tfgSpreadSheetClear

Description

Clears the spreadsheet.

NOTE: The `tfgSpreadSheetClear` command replaces the `tsSpreadSheetClear` command.

Syntax

```
tfgSpreadSheetClear
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetClear
```

tfgSpreadSheetDel

Description

Removes signal with time from the spreadsheet.

NOTE: The `tfgSpreadSheetDel` command replaces the `tsSpreadSheetDel` command.

Syntax

```
tfgSpreadSheetDel [signalName list]
```

Argument

`signal_name` *`list`*

Specifies the signal name list.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetDel "top.cpu.a#50" "top.cpu.b#100"
```

tfgSpreadSheetDown

Description

Moves the specified column down.

NOTE: The `tfgSpreadSheetDown` command replaces the `tsSpreadSheetDown` command.

Syntax

```
tfgSpreadSheetDown [signal_symbol]
```

Argument

`signal_symbol`

Specifies the signal symbol.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetDown "top.a#100"
```

tfgSpreadSheetExec

Description

Executes the spreadsheet expression.

NOTE: The `tfgSpreadSheetExec` command replaces the `tsSpreadSheetExec` command.

Syntax

```
tfgSpreadSheetExec [expression]
```

Argument

`expression`

Specifies the expression.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetExec "top.a#100 + 10"
```

tfgSpreadSheetRename

Description

Modifies the name on the spreadsheet.

NOTE: The `tfgSpreadSheetRename` command replaces the `tsSpreadSheetRename` command.

Syntax

```
tfgSpreadSheetRename [originalName] [newName]
```

Argument

`new_name`

Specifies the new name.

`original_name`

Specifies the original name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetRename "top.cpu.original#50" "top.cpu.name#50"
```

tfgSpreadSheetSave

Description

Saves the spreadsheet content to file.

Syntax

```
tfgSpreadSheetSave –file filename
```

Argument

`–file` *`filename`*

Specifies the file to save the spread sheet content to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetSave -file result.txt
```

tfgSpreadSheetSetRadix

Description

Sets the signal symbol radix.

NOTE: The `tfgSpreadSheetSetRadix` command replaces the `tsSpreadSheetSetRadix` command.

Syntax

```
tfgSpreadSheetSetRadix [signalSymbol] -format radix
```

Argument

`-format`

Specifies the radix [bin/oct/hex/dec/asc/754].

`signal_symbol`

Specifies the signal symbol.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetSetRadix "top.a#100" -format bin
```

tfgSpreadSheetUp

Description

Moves the specified column up.

NOTE: The `tfgSpreadSheetUp` command replaces the `tsSpreadSheetUp` command.

Syntax

```
tfgSpreadSheetUp [signalSymbol]
```

Argument

`signal_symbol`

Specifies the signal symbol.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSpreadSheetUp "top.a#100"
```

tfgSyncNWave

Description

Synchronizes *Temporal Flow View* with *nWave* (generated by *nWave* menu).

NOTE: The `tfgSyncNWave` command replaces the `tsSyncNWave` command.

Syntax

```
tfgSyncNWave -win window
```

Argument

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgSyncNWave -win $_tFlowView2
```

tfgTraceMemory

Description

Traces memory and shows it on memory window.

NOTE: The `tfgTraceMemory` command replaces the `tsTraceMemory` command.

Syntax

```
tfgTraceMemory -win window -sig sig_name -startAddr number
-endAddr number -time number -stopTime number
```

Argument

`-endAddr number`

Specifies the end address.

`-sig sig_name`

Specifies the memory net.

`-startAddr number`

Specifies the start address.

`-stopTime number`

Specifies the stop time.

`-time number`

Specifies the begin time.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTraceMemory -win $_tMemory1 -sig "top.mem" -startAddr 0
-endAddr 1023 -time 100 -stopTime 0
```

tfgTraceMemoryInit

Description

Traces memory initialization and creates memory window.

NOTE: The `tfgTraceMemoryInit` command replaces the `tsTraceMemoryInit` command.

Syntax

```
tfgTraceMemoryInit -win window -sig sig_name -startAddr number
-endAddr number
```

Argument

`-endAddr number`

Specifies the end address.

`-sig sig_name`

Specifies the memory net.

`-startAddr` *`number`*

Specifies the start address.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTraceMemoryInit -win $_tFlowView2 -sig "top.mem" -startAddr 0
-endAddr 1023
```

tfgTrPwrX

Description

Tracks down the source of the unknown power state.

NOTE: The `tfgTrPwrX` command replaces the `tsTrPwrX` command.

Syntax

```
tfgTrPwrX -pd PowerDomainName -time time -state
```

Argument

`-pd` *`PowerDomainName`*

Specifies the power domain name.

`-state`

Traces unknown power state.

`-time` *`time`*

Specifies the time to trace.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrPwrX -pd {/$power_root/@{\$TOP.L1.PD1 }} -time 152.57 -state
```

tfgTrX

Description

Traces the active X.

NOTE: The `tfgTrX` command replaces the `tsTrX` command.

Syntax

```
tfgTrX -win window -folder folder_name [-showOnTFG|-showOnNWave]
[-maxLev number] [-traceNonTrigX] -time [signal_name]
[-batchInput filename] [-batchOutput filename] [-showTracePath]
[-noBBox] [-vcOnly] [-causeCnt number] [-snapVC]
```

NOTE: The `-showTracePath`, `-noBBox`, `-vcOnly`, `-causeCnt number`, and `-snapVC` options can only be enabled in batch mode.

Argument

`-batchInput filename`

Specifies the signal list with X value.

`-batchOutput filename`

Specifies the file name for the dumped results.

`-causeCnt number`

Traces until the specified number of causes are found.

`-folder folder_name`

Specifies the folder name.

`-maxLev`

Specifies the maximum level.

`-noBBox`

Stops when the black box boundary is reached.

`signal_name`

Specifies the signal name.

`-showOnNWave`

Shows the trace results on the *nWave* window.

`-showOnTFG`

Shows the trace results on the *Temporal Flow View* window.

`-showTracePath`

Shows the trace path.

`-snapVC`

Snaps to the value change time and continues tracing.

`-time`

Specifies the root signal time.

`-traceNonTrigX`

Traces the non-triggering X when no triggering path with X exists.

`-vcOnly`

Traces the value change path only.

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrX -win $_tFlowView2 -folder "top.cpu#50" "top.cpu.data"
-noBBox -causeCnt 0
```

tfgTrXConfig

Description

Displays the low possibility X causes in the *Trace Active X Results* window.

Syntax

`tfgTrXConfig –win` *`window`* `–showLowPos on|off`

Argument

`–showLowPos on|off`

When this option is turned *on*, displays the low possibility X causes in the *Trace Active X Results* window. When this option is turned *off*, does not display the low possibility X causes in the *Trace Active X Results* window. The default is *off*.

`-win` *`window`*

Specifies the *Trace Active X Results* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrXConfig -win $_tTraceXRst3 –showLowPos on
```

tfgTrXDrag

Description

Drags the selected signals in the *Trace Active X Results* window.

Syntax

```
tfgTrXDrag -win window
```

Argument

`-win` *window*

Specifies the *Trace Active X Results* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrXDrag -win $_tTraceXRst3
```

tfgTrXHighlightPDML

Description

Highlights the condition signals in the power manager window.

Syntax

```
tfgTrXHighlightPDML –win window
```

Argument

`-win window`

Specifies the *Trace Active X Results* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrXHighlightPDML -win $_tTraceXRst3
```

tfgTrXNextNCauses

Description

Continues to trace the next X cause (user specified N causes and want to trace other causes).

NOTE: The `tfgTrXNextNCauses` command replaces the `tsTrXNextNCauses` command.

Syntax

```
tfgTrXNextNCauses -win window
```

Argument

`-win window`

Specifies the trace X result window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrXNextNCauses -win $_tTraceXRst1
```

tfgTrXResume

Description

Resumes and continues to trace the next X cause.

NOTE: The `tfgTrXResume` command replaces the `tsTrXResume` command.

Syntax

```
tfgTrXResume -win window
```

Argument

`-win` *window*

Specifies the trace X result window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrXResume -win $_tTraceXRst1
```

tfgTrXRstClose

Description

Closes the trace X report window.

NOTE: The `tfgTrXRstClose` command replaces the `tsTrXRstClose` command.

Syntax

```
tfgTrXRstClose -win window
```

Argument

`-win` *window*

Specifies the trace X result window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrXRstClose -win $_tTraceXRst1
```

tfgTrXSave

Description

Saves the trace X results to a file.

NOTE: The `tfgTrXSave` command replaces the `tsTrXSave` command.

Syntax

```
tfgTrXSave -win window [filename]
```

Argument

`filename`

Specifies the file name.

`-win window`

Specifies the trace X result window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrXSave -win $_tTraceXRst1 traceX.rst
```

tfgTrXSelect

Description

Selects the signals in the *Trace Active X Results* window.

Syntax

```
tfgTrXSelect –win window -line lineNumber
```

Argument

`-line lineNumber`

Specifies the line in the result grid of the *Trace Active X Results* window.

`-win window`

Specifies the *Trace Active X Results* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrXSelect -win $_tTraceXRst3 -line 4
```

tfgTrXToggleHighlightPath

Description

Turns the **Highlight Selected Trace Path** toggle command *on* or *off*. When this toggle command is turned *on*, execute this Tcl command again to turn it *off* and vice versa.

Syntax

```
tfgTrXToggleHighlightPath –win window
```

Argument

`-win window`

Specifies the *Trace Active X Results* window.

Value Returned

Always returns 1.

Example

```
tfgTrXToggleHighlightPath -win $_tTraceXRst3
```

tfgVHZoomIn

Description

Stores the current status and zooms in for vertical and horizontal direction.

NOTE: The `tfgVHZoomIn` command replaces the `tsVHZoomIn` command.

Syntax

```
tfgVHZoomIn -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgVHZoomIn -win $_tFlowView2
```

tfgVHZoomOut

Description

Stores the current status and zooms in for vertical and horizontal direction.

NOTE: The `tfgVHZoomOut` command replaces the **tsVHZoomOut** command.

Syntax

```
tfgVHZoomOut -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgVHZoomOut -win $_tFlowView2
```

tfgVZoomIn

Description

Expands the space between nodes in order to vertically zoom in.

NOTE: The `tfgVZoomIn` command replaces the `tsVZoomIn` command.

Syntax

```
tfgVZoomIn -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgVZoomIn -win $_tFlowView2
```

tfgVZoomOut

Description

Expands the space between nodes in order to vertically zoom out.

NOTE: The `tfgVZoomOut` command replaces the `tsVZoomOut` command.

Syntax

```
tfgVZoomOut -win window
```

Argument

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgVZoomOut -win $_tFlowView2
```

tfgZoom

Description

Zooms to the specified time range.

NOTE: The `tfgZoom` command replaces the `tsZoom` command.

Syntax

```
tfgZoom -win window [minimum_time] [maximum_time]
```

Argument

`maximum_time`

Specifies the maximum time range.

`minimum_time`

Specifies the minimum time range.

`-win window`

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgZoom -win $_tFlowView2 20 100
```

tfgZoomIn

Description

Zooms in for vertical direction.

NOTE: The `tfgZoomIn` command replaces the `tsZoomIn` command.

Syntax

```
tfgZoomIn -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgZoomIn -win $_tFlowView2
```

tfgZoomOut

Description

Zooms out for vertical direction.

NOTE: The `tfgZoomOut` command replaces the `tsZoomOut` command.

Syntax

```
tfgZoomOut -win window
```

Argument

`-win` *window*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgZoomOut -win $_tFlowView2
```

tfgDehighlightAll

Description

Cancels the highlight of all signals.

NOTE: The `tfgDehighlightAll` command replaces the `tsDehighlightAll` command.

Syntax

```
tfgDehighlightAll -win window
```

Argument

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgDehighlightAll -win $_tFlowView2
```

tsCaptureView

Description

Captures the *Temporal Flow View* window.

Syntax

```
tsCaptureView -win window
```

Argument

`-win` *`window`*

Specifies the *Temporal Flow View* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tsCaptureView -win $_tFlowView2
```

nAnalyzer

Clock Analyzer

schCheckClockCTSSetting

Description

Checks different CTS constraints between different clock sources.

Syntax

```
schCheckClockCTSSetting [-checkSignalCTSConflict] [-sigBase
source [-instStop instStopTypeList] [-instIgnore
instIgnoreTypeList] [-instPassthrough passthroughTypeList]]
[-selectSignal signalList]
schCheckClockCTSSetting [-save filename]
schCheckClockCTSSetting [-dump]
schCheckClockCTSSetting [-setType CTSSourceType]
schCheckClockCTSSetting [-setToCTS]
```

Argument

`-checkSignalCTSConflict`

Checks conflicts in the selected clock roots. You must set this parameter.

`-sigBase`

Specifies local CTS settings of a root clock source.

`-selectSignal`

Specifies a list of selected clock sources to be exported.

`-save filename`

Saves to the specified file.

`-dump`

Dumps the current tree list.

`-setType`

Changes the CTS type of the specified CTS constraints for the clock source. Options for the CTS type are Stop, Ignore, Passthrough and Unset.

`-setToCTS`

Applies the latest CTS constraint changes and check the conflicts between the CTS constraints again.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schCheckClockCTSSetting -checkSignalCTSConflict -sigBase \
system.i_cpu.i_CCU.C6_reg.Q -instStop "system.i_cpu.i_ALUB.U282"
\"A" -instStop "system.i_cpu.i_ALUB.U282" "B" -instStop \
"system.i_cpu.i_ALUB.U282" "C" -instStop
"system.i_cpu.i_ALUB.U282" \ "D" -instStop
"system.i_cpu.i_ALUB.U282" "Z" -instIgnore \
"system.i_cpu.i_ALUB.i_alu" "zero" -instIgnore \
"system.i_cpu.i_ALUB.i_alu" "b\[7:0\]" -instIgnore \
"system.i_cpu.i_ALUB.i_alu" "a\[7:0\]" -instIgnore \
"system.i_cpu.i_ALUB.i_alu" "cin" -instIgnore \
"system.i_cpu.i_ALUB.i_alu" "carry" -instIgnore \
"system.i_cpu.i_ALUB.i_alu" "select\[2:0\]" -instPassthrough \
"system.i_cpu.i_ALUB.\\IXR_reg\[6\] " "CD" -instPassthrough \
"system.i_cpu.i_ALUB.\\IXR_reg\[6\] " "CP" -instPassthrough \
"system.i_cpu.i_ALUB.\\IXR_reg\[6\] " "D" -instPassthrough \
"system.i_cpu.i_ALUB.\\IXR_reg\[6\] " "Q" -selectSignal \
"system.i_cpu.i_CCU.C6_reg.Q" "system.i_cpu.i_ALUB.U289.D" \
"system.i_cpu.i_ALUB.U289.C" \
"system.i_cpu.i_ALUB.\\IXR_reg\[5\] .Q" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .Q" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .D" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .CP"

schCheckClockCTSSetting -dump

schCheckClockCTSSetting -setType "system.i_cpu.i_ALUB.U251.Z" \
"system.i_cpu.i_ALUB.n789" "Stop Point"

schCheckClockCTSSetting -setToCTS

schCheckClockCTSSetting -save \ test.txt
```

schClkCheckCrossPathHideSelect

Description

Filters or not to filter the checked crossing path(s) or register pair(s) in the *Crossing Paths* form.

Syntax

```
schClkCheckCrossPathHideSelect [-win window][-filter on|off]
```

Argument

`-filter on|off`

Filters all checked path and pair or not.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schClkCheckCrossPathHideSelect -win $_CheckCrossingPaths4 -filter
on
```

schClkCheckCrossPathSelectAll

schClkCheckCrossPathUnselectAll

Description

Sets all crossing paths or register pairs as check or uncheck.

Syntax

```
schClkCheckCrossPathSelectAll [-win window]
schClkCheckCrossPathUnselectAll [-win window]
```

Argument

-win

Specifies the window id.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schClkCheckCrossPathSelectAll -win $_CheckCrossingPaths4
schClkCheckCrossPathUnselectAll -win $_CheckCrossingPaths4
```

schClkCheckCrossPathSetFalsePathFilterRule

Description

Sets the top design, delimiter and false path list in a SDC file.

Syntax

```
schClkCheckCrossPathSetFalsePathFilterRule [-sdcFile filename]
[-design topDesign] [-delim delimiter]
[-check checkedFalsePathList] [-dump FalsePathList] [-reload
filename]
```

Argument

`-sdcFile filename`

Specifies a SDC file. The file must already exist in the filter string list added by schCrossPathFilter with -addFilter parameter.

`-design`

Specifies the top design of the SDC file.

`-delim`

Specifies the delimiter for the SDC file.

`-check`

Specifies the false path list in the **False Path** tab of the *Filter Rule Settings* form.

`-dump`

Dumps the false paths in the **False Path List** to the **False Path** tab of the *Filter Rule Settings* form.

`-reload`

Forces to reload false path from a SDC file. If this attribute is not set, it tries to load false path from memory/swap file first. If it cannot be found in the memory/swap file, a SDC file is loaded from the same file.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schClkCheckCrossPathSetFalsePathFilterRule -sdcFile \ "test.sdc"\
-design "system" -delim "/" -check 0 1 4
```

schClkDomainFilter

Description

Filter names or resolve types from the clock domains form.

Syntax and Arguments

```
schClkDomainFilter -win $_nSchema2 -nameCheck s1 s2 -nameUnCheck
s3 s4 -resolveTypeUnCheck 2 3 -resolveTypeCheck 1 4 5 6 7 8 9 11
12 14
```

NOTE: Every resolve type has its own representative number. When logging a resolve type to Tcl, only the number is logged. The following is the list of the equivalent numbers:
Resolved (Primary Input) = 1
Resolved (User Specified) = 2
Resolved (Register Output) = 3
Resolved (Gated Clock) = 4
Unresolved (Combinational Logic) = 5
Unresolved (Feedback loop) = 6
Unresolved (Floating) = 7
Multi-Resolved = 8
Floating = 9
Resolved = 11
Unresolved = 12
Resolved (Node) = 14

`-clockSourceCheck <clock_source>`

Checks the clock source type. Available types are “Known Clock Source” and “Not Known Clock Source”.

`-clockSourceUnCheck <clock_source>`

Unchecks the clock source type. Available types are “Known Clock Source” and “Not Known Clock Source”.

`-dump`

Dumps the contents of the current *Clock Domain* window.

`-loadFile`

Loads the file that contains the filters.

`-nameCheck`

Specifies the name of the clock domains or clock sources to be added and checked.

`-notapply`

Only adds the filters to the Filter form but do not apply on the *Clock Domain* window. If you apply, this attribute is not logged.

`-resolveTypeCheck`

Specifies the resolve types to be added and checked.

`-resolveTypeUnCheck`

Specifies the resolve types to be added and checked.

`-saveFile`

Saves the checked and unchecked filters to a file.

`-ScopeUnCheck`

Specifies the name of the clock domains or clock sources to be added but not checked.

`-win`

Specifies the window id.

Value Returned

Returns 1 if successful; otherwise returns 0.

Example

```
schClkDomainFilter -win $_nSchema2 -nameUnCheck system -nameCheck
system.i_cpu.i_ALUB -resolveTypeUnCheck 1 -resolveTypeCheck 2 3 4
5 6 7 8 9 11 12 14 -notapply
```

Every resolve type number must be assigned to either `-resolveTypeUnCheck` or `-resolveTypeCheck`. For example, if the value of `-resolveTypeUnCheck` is 2 3, the value of `-resolveTypeCheck` is 1, 4, 5, 6, 7, 8, 9, 11, 12, 14. Another example, if `-resolveTypeUnCheck` is 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 14, then `-resolveTypeCheck` is not logged, because all the numbers are specified to the `-resolveTypeUnCheck` attribute.

```
schClkDomainFilter -win $_nSchema2 -nameCheck s1 s2 nameUnCheck
s3 s4 -resolveTypeUnCheck 2 3 -resolveTypeCheck  1 4 5 6 7 8 9 11
12 14 -notapply
```

```
schClkDomainFilter -win $_nSchema2 -save "save1.is"
schClkDomainFilter -win $_nSchema2 -load "save1.is"
schClkDomainFilter -win $_nSchema2 -dump
```

Note that when you save filters without clicking the **Apply/OK** button, the command above is logged with `-notapply`, then log `-save` as follows: `"schClkDomainFilter -win $_nSchema2 -save "save1.is"`

```
schClkDomainFilter -win $_nSchema2 -clockSourceCheck "Known Clock
Source" -clockSourceUnCheck \ "Not Known Clock Source"
```

schClkDomainFind

Description

Finds the clock domain or clock source from the *Clock Domain* form based on the specified name.

Syntax and Arguments

`-win`

Specifies the window id.

`-name`

Specifies the clock source or clock domain you want to find.

`-up`

Finds the previous result.

`-down`

Finds the next result.

`-dump`

Dumps the found result.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkDomainFind -win $_nSchema2 -up
schClkDomainFind -win $_nSchema2 -name TEST.SYSCLK -down
schClkDomainFind -win $_nSchema2 -dump
```

Before finding the next result, `schShowClockDomain` must be logged before `schClkDomainFind`. `schShowClockDomain` displays the current node and node.

```
schShowClockDomain -win $xxx -sourceNode | -riseDomain  |
-fallDomain -selectNode yyy
```

-selectNode: Specifies that the selected node is highlighted.

-sourceNode: Specifies that current node is a source.

-riseDomain: Specifies that current node is a rising domain

-fallDomain: Specifies that current node is a falling domain

Example of using `schClkDomainFind` and `schShowClockDomain`:

When you click the **Next** button, the following two commands are logged:

```
schShowClockDomain -win $_nSchema2 -sourceNode -selectNode
system.i_cpu.i_CCU.C5
schClkDomainFind -win $_nSchema2 -name * -down
```

When you click the **Previous** button, if there is not any value for the `-name` attribute, the following command is logged:

```
schClkDomainFind -win $_nSchema2 -name  -up
schClkDomainFind -win $_nSchema2 -dump
```

schClkExtractCDCloseWindow

Description

Closes the *Clock Domains* window.

Syntax

`schClkExtractCDCloseWindow [-win` *`windowID`*`]`

Arguments

`-win windowID`

The *Clock Domains* window ID.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkExtractCDCloseWindow -win $_ExtractClockDomains2
```

schClkExtractCDExtractGeneratedClock

Description

Finds all generated clocks of the known clock sources in the **Clock Source** table of the *Clock Tree Setting* form and shows the results in the *Extract Generated Clock List* form.

Syntax

```
schClkExtractCDExtractGeneratedClock [-dump]
```

Argument

`-dump`

Dumps results.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schClkExtractCDExtractGeneratedClock -dump
```

schClkExtractCDGroupClkSource

Description

Groups the specified root clock source in the *Clock Domains* window.

Syntax

```
schClkExtractCDGroupClkSource [-win windowID] [-addClockGroup
newGroupName] [-rootClkSrc rootClockSourcesList]
```

Arguments

`-addClockGroup` *`newGroupName`*

Creates a new clock group.

`-rootClkSrc` *`rootClockSourceList`*

Specifies the clock source where the clock group belongs. `rootClockSourceList` refers to the root clock source list, including clock source domain number.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkExtractCDGroupClkSource -win $_ExtractClockDomains3
-addClockGroup "NewGroup4" -rootClkSrc "2" "3"
```

schClkExtractCDMoveClkSourceToGroup

Description

Moves a clock source, which might be in an existing clock group, to another group.

Syntax

```
schClkExtractCDMoveClkSourceToGroup  [-win windowID]
[-moveClkSrcToGroup sourceDomainNumber [-clkFromGroup fromGroup]
-clkToGroup toGroup]
```

Arguments

`-moveClkSrcToGroup` *`sourceDomainNumber`*

Moves a clock source from a group, which might be in a group already, to another group. `sourceDomainNumber` refers to the domain number of the clock source to be moved.

`-clkFromGroup` *`fromGroup`*

Specifies the group where the clock source is, if this clock source is already in a group. `fromGroup` refers to name of the group where the clock source is moved.

`-clkToGroup` *`toGroup`*

Specifies the group where the clock source is to move to. `toGroup` refers to the name of the group where the clock source is to move to.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkExtractCDMoveClkSourceToGroup -win $_ExtractClockDomains3
-moveClkSrcToGroup "1" -clkFromGroup "NewGroup6" -clkToGroup
"NewGroup7"
schClkExtractCDMoveClkSourceToGroup -win $_ExtractClockDomains3
-moveClkSrcToGroup "4" -clkToGroup "NewGroup7"
```

schClkExtractCDSetClockRoot

Description

Sets the specified clock source as the clock source root in the *Clock Domains* window.

Syntax

```
schClkExtractCDSetClockRoot [-src clockSourceList] [-dump]
```

Argument

`-src` *`clockSourceList`*

Specifies the list of clock sources.

`-dump`

Dumps the clock root and generated clock information.

Value Returned

Returns 1 if at least one clock root is set; otherwise, returns 0.

Example

```
schClkExtractCDSetClockRoot -src "system.clock"
"system.i_cpu.i_ALUB.S1" -dump

schClkExtractCDSetClockRoot -win $_ExtractClockDomains2 -src
"BBB.clkclk->BBB.clka" "BBB.clkclk->BBB.clkb"
```

The value after the `-src` attribute means a clock source is driven by clock signals and it is expressed with `->`. On the left side of `->` is the clock source and on the right side of `->` are the signals driving the clock source. In the example above,

both the resolved clock signals, `BBB.clka` and `BBB.clkb`, drive the clock source, `BBB.clkclk`.

schClkExtractCDSetDerivedClock

Description

Sets the specified clock pin or signal as the generated clock of the clock root in the *Clock Domains* window.

Syntax

```
schClkExtractCDSetDerivedClock [-win window] [-derivedNode]
```

Argument

`-win`

Specifies the window id.

`-derivedNode`

Specifies the selected clock pin or signal to be set as a generated clock. If the selection is a clock pin, the number for sub-domain ID is shown; if the selection is a signal, the clock source name is shown.

Value Returned

Returns 1 if at least one clock root is set; otherwise, returns 0.

Example

```
schClkExtractCDSetDivideClock -win $_ExtractClockDomains2
-deviveNode \ "system.i_cpu.i_CCU.C19" "system.i_cpu.i_CCU.C5"
\"system.i_cpu.i_CCU.C6"

schClkExtractCDSetDivideClock -win $_ExtractClockDomains2 -
derivedNode 1.1.r 1.2.f 1.3.f
```

schClkExtractCDUngroupClkSource

Description

Ungroups a clock group or removes a clock source from a group.

Syntax

```
schClkExtractCDUngroupClkSource [-win windowID][-delClockGroup
groupNameList][-ungroupClkSrc sourceDomainNo -clkGroup
rootGroupName]
```

Arguments

`-delClockGroup` *`groupNameList`*

Specifies clock groups to be ungrouped. `groupNameList` refers to the group to be removed.

`-ungroupClkSrc` *`sourceDomainNo`*

Specifies the clock source to be ungrouped. `sourceDomainNo` refers to the domain number of the clock source.

`-clkGroup` *`rootGroupName`*

Specifies the clock group where the clock source is located. `rootGroupName` refers to the group where the clock source is located.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkExtractCDUngroupClkSource -win $_ExtractClockDomains3
-delClockGroup "NewGroup4" "NewGroup5"

schClkExtractCDUngroupClkSource -ungroupClkSrc "2" -clkGroup
"NewGroup4"
```

schClkStatCapture

Description

Captures the desired viewing area in the statistics window and saves in the PNG format.

Syntax

```
schClkStatCapture [-win window] [-file filename]]
```

Argument

`-win window`

Specifies the window id of the *Statistics* window.

`-file filename`

Specifies the filename to be saved.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkStatCapture -win $_nClkStatistics3 -file "test.png" -region
-1 -1 0 0
```

schClkStatCloseWindow

Description

Closes a statistics window.

Syntax

```
schClkStatCloseWindow [-win window]
```

Argument

`-win window`

Specifies the window id of the *Statistics* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkStatCloseWindow -win $_nClkStatistics3
```

schClkStatDumpAU

Description

Information from the statistics window such as Histogram Label, Value-Axis (Y-Axis), Category-Axis (X-Axis), Bar, Bar Label, etc. is dumped to the `turbo.log` file.

Syntax

```
schClkStatDumpAU [-win window]
```

Argument

`-win` *`window`*

Specifies the window id of the *Statistics* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkStatDumpAU -win $_nClkStatistics4
```

schClkStatPreference

Description

Configures color settings for all statistics windows.

Syntax

```
schClkStatPreference [-barColor color1 color2] [-selectedSetColor
color] [-backGroundColor color]
```

Argument

`-barColor color1 color2`

Sets the color for the first bar group and second bar group.

`-selectedSetColor color`

Sets the color for the outline of selected bars.

`-backGroundColor color`

Sets the background color of the statistics window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkStatPreference -barColor ID_RED6 ID_BLUE6 -selectedSetColor
ID_BLACK -backGroundColor ID_RED5
```

schClkStatSetPartition

Description

Sets the partition number of histogram in the statistics window.

Syntax

```
schClkStatSetPartition [-win window] [-divide number]
```

Argument

`-divide number`

Specifies the partition number in the *Statistics* window.

`-win window`

Specifies the window id of the *Statistics* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkStatSetPartition -win $_nClkStatistics3 -divide 5
```

schClkStatSetInsertionType

Description

Sets the insertion delay type with **Delay from Source** or **Delay from Level** to calculate delay value.

Syntax

```
schClkStatSetInsertionType [-win window]
[-insertionSource|-insertionLevel]
```

Argument

`-win window`

Specifies the window id of the *Statistics* window.

`-insertionSource`

Specifies the insertion delay type as Delay from Source.

`-insertionLevel`

Specifies the insertion delay type as Delay from Level.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkStatSetInsertionType -win $_nClkStatistics3 -insertionLevel
schClkStatSetInsertionType -win $_nClkStatistics3
-insertionSource
```

schClkStatSetSDF

Description

Sets delay scale, delay type, and delay precision for bars in the *Statistics* window.

Syntax

```
schClkStatSetSDF [-win window] [-delayType sdfType]
[-delayScale sdfScale] [-delayPrecision sdfPrecision]
```

Argument

`-delayType sdfPrecision`

Specifies the delay precision of bars in the *Statistics* window. The precision is the same as the SDF precision.

`-delayType sdfScale`

Specifies the delay scale of bars in the *Statistics* window. The scale is the same as the SDF scale.

`-delayType sdfType`

Specifies the delay type of bars in the *Statistics* window. The type is the same as the SDF type.

`-win window`

Specifies the window id of the *Statistics* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkStatSetSDF -win $_nClkStatistics3 -delayType min
schClkStatSetSDF -win $_nClkStatistics3 -delayScale 100fs
schClkStatSetSDF -win $_nClkStatistics3 -delayPrecision 0.01

schClkStatSetSDF -win $_nClkStatistics3 -delayType max\
-delayScale 100ps -delayPrecision 0.001
```

schClockSkew

Description

Displays the clock skew for the specified clock source.

Syntax

```
schClockSkew [-win window] [-delim value] -clockSource clockSource
[-pathNum #path] [-shortestPath | -longestPath] [-showOn
schema|text] [-order order1 order2...] [-maxSkew skewValue]
[-negedge|-posedge]
```

Arguments

`-clockSource clockSource`

Specifies the clock source to measure skew.

`-delim` *`value`*

Specifies the delimiter for the instance name.

`-longestPath`

Uses the longest path to measure skew. This argument cannot be specified with `-shortestPath`.

`-negedge`

Shows the connection to negative trigger register(s) only. This argument cannot be specified with `-posedge`.

`-order` *`order1 order2...`*

Specifies the order of paths to be shown on *nSchema* or text (the order is based on all found paths).

`-pathNum` *`pathNum`*

Specifies the number of paths to be reported.

`-posedge`

Shows the connection to positive trigger register(s) only. This argument can not be specified with `-negedge`.

`-shortestPath`

Uses the shortest path to measure skew. This argument cannot be specified with `-longestPath`.

`-showOn schema|text`

Displays the results on the *nSchema* window or in the text form.

`-win` *`window`*

Specifies the window id of the invoking *nSchema* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClockSkew -win $_nSchema2 -clockSource "system.clock"\
-delim "." -pathNum -1 -longestPath -maxSkew "0.000000"
schClockSkew -win $_nSchema2 -posedge
schClockSkew -win $_nSchema2 -showOn schema -order 9 48 62 63 64
```

schClockTreeBrowser

Description

Finds an instance in the *Clock Tree Browser* window.

Syntax

```
schClockTreeBrowser -win $ win_id -select inst_name
```

Arguments

`-win`

Specifies the window id of the invoking *nSchema* window.

`-select`

Specifies the instance name to be found.

`-loadCTSAtt "xxx.cts" -append`

Loads the CTS attributes to the **CTS** tab of the *Clock Tree Setting* form.

`-saveCTSAtt "xxx.cts"`

Saves the CTS attributes to the **CTS** tab of the *Clock Tree Setting* form.

`-showCTS`

Shows the *Clock Tree Setting* form.

`-closeCTS`

Closes the *Clock Tree Setting* form.

Example

```
schClockTreeBrowser -win $_nSchema2 -select
"system.CHILD1.FSM1_SEQ"
```

NOTE: In the Verdi 6.1 version, the `schLoadClockSetting` and `schSaveClockSetting` commands replace the `-loadCTSAtt` and `-saveCTSAtt` options of the `schClockTreeBrowser` command respectively.

schClkTreeBrowserCloseWindow

Description

Closes the *Clock Tree Browser* window.

Syntax

```
schClkTreeBrowserCloseWindow  [-win windowID]
```

Arguments

`-win windowID`

The *Clock Tree Browser* window ID.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserCloseWindow -win $_ClockTreeBrowser5
```

schClkTreeBrowserCollapseAllTreeAndSubtree

Description

Collapses all trees and sub-trees in the *Clock Tree Browser* window.

Syntax

```
schClkTreeBrowserCollapseAllTreeAndSubtree [-win window]
```

Argument

`-win window`

Specifies the window id of the invoking *Clock Tree Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserCollapseAllTreeAndSubtree -win
$_ClockTreeBrowser2
```

schClkTreeBrowserCTSConstraintChecker

Description

This command supports the following four functions:

1. Finds the specified node by cell type, cell name, or instance name
2. Checks convergence nodes
3. Lists all nodes without leaf points resulting from the ignore setting
4. Lists the first node in the clock tree that drives only to non-clock leaves

Syntax

```
schClkTreeBrowserCTSConstraintChecker -win window -extract
[-findNode findNodeArguments] [-checkConvergenceNode
convergenceArguments] [-checkNoLeafNode]
[-checkNonClkEndNodeBranches off|on [-checkPrimaryOutput on|off]
[-checkFloating on|off] [-checkStorageNonClk on|off]]


schClkTreeBrowserCTSConstraintChecker -win window -save filename
[-findNode] [-checkConvergenceNode] [-checkNoLeafNode]


schClkTreeBrowserCTSConstraintChecker -win window -refresh
[-select group] [-showPassThroughGateClk] [-dump]
```

Argument

`-win`

Specifies the window id.

`-extract`

Extracts nodes with the specified arguments. You can specify one or more of the following options:

- `-findNode`: Extracts "Find Nodes".
- `-checkConvergenceNode`: Extracts "Check Convergence Nodes".
- `-checkNoLeafNode`: Extracts "List Nodes without Leaves".

`-save` *`filename`*

Saves the current results to the specified file. You can specify one or more of the following options:

- `-findNode`: Saves the results of "Find Nodes" to the specified file.
- `-checkConvergenceNode`: Saves the results of "Check Convergence Nodes" to the specified file.

- `-checkNoLeafNode`: Saves the results of "List Nodes without Leaves" to the specified file.

`-refresh`

Refreshes the results from the **CTS Constraint Checker** command. You can specify one or more of the following options:

- `-select group`: Switches to the specified group where the group can be "Find Nodes", "Check Convergence Nodes", or "List Nodes without Leaves".
- `-showPassThroughGateClk`: Enables the **Show Passthrough Gated Clock** option. If this argument is not specified, the option is turned off.
- `-dump`: Dumps all results in "Find Nodes", "Check Convergence Nodes", and "List Nodes without Leaves" to the specified log.

`-findNode` *`findNodeArguments`*

Finds the specified node by cell type, cell name, or instance name.
The `findNodeArguments` argument includes
`[-instName name] [-cellName name] [-ff] [-latch] [-macro] [-outputPort] [-floating] [-duplication] [-stopPin] [-loop] [-datapin] [-xor] [-gateClk]`

`-ffTrigger`, `-latchTrigger`, and `-macroTrigger` are for trigger types of flip-flop, latch and macro respectively. When `-ffTrigger`, `-latchTrigger`, and `-macroTrigger` are added, one or more trigger types of "positive", "negative" and "unknown"are specified and take effect.

`-checkConvergenceNode` *`convergenceArguments`*

Check convergence nodes.
The `convergenceArguments` include:
`[-tree1 name] [-tree2 name]`

`[-mode1 name] [-mode2 name] [-specifiedmode ModeName]`

- `-tree1`: The name of the first listed clock tree for convergence check.
- `-tree2`: The name of the second listed clock tree for convergence check.
- `-mode1`: The mode name of the first clock tree for convergence check.
- `-mode2`: The mode name of the second clock tree for convergence check.
- `-specifiedmode`: Enables to apply the specified mode to the first and second clock trees for convergence check.

`-checkNoLeafNode`

Lists all nodes without leaf points resulting from the ignore setting.

`-checkNonClkEndNodeBranches`

Checks branch nodes with the non-clock end nodes. You can specify your intended non-clock leaf nodes from `-checkStorageNonClk` (starts the checking from the non-clock port of storage), `-checkPrimaryOutput` (starts the checking from the primary output) and `-checkFloating` (starts the checking from the floating pin). `-checkStorageNonClk`, `-checkPrimaryOutput` and `-checkFloating` do not have any dependencies, so you can specify one or more of the options simultaneously.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schClkTreeBrowserCTSConstraintChecker -win $_ClockTreeBrowser2
-extract -instName \"*" -findNode -checkConvergenceNode -tree1 \
"system.i_cpu.i_ALUB.CH\[0\]" "system.i_cpu.i_ALUB.n789" \
-tree2 "system.i_cpu.i_ALUB.n737" "system.i_cpu.i_ALUB.n789" \
-checkNoLeafNode

schClkTreeBrowserCTSConstraintChecker -win $_ClockTreeBrowser2
-save \ golden.log \ -delim . -findNode -checkConvergenceNode
-checkNoLeafNode

schClkTreeBrowserCTSConstraintChecker -win $_ClockTreeBrowser2
-select \ "system.i_cpu.i_ALUB.n789" -refresh -dump

schClkTreeBrowserCTSConstraintChecker -win $_ClockTreeBrowser4
-extract -instName "*" -findNode -checkNonClkEndNodeBranches
-checkPrimaryOutput -checkFloating -checkStorageNonClk

- schClkTreeBrowserCTSConstraintChecker -win $_ClockTreeBrowser3
-extract -instName "*" -findNode -checkConvergenceNode -tree1
"test.CLK0" "test.CLK1" -tree2 "test.CLK0" "test.CLK1"
-specifiedmode "default"

schClkTreeBrowserCTSConstraintChecker -win $_ClockTreeBrowser3
-extract -ff \ -ffTrigger "positive,unknown," -latch
-latchTrigger \"positive,negative," -macro -instName "*"
-findNode
```

Try to find if any flip-flops with positive or unknown types, any latches with positive or negative types, and all macros.

schClkTreeBrowserDelayHistogram

Description

Creates an insertion delay histogram in a new statistics window.

Syntax

```
schClkTreeBrowserDelayHistogram [-win window]
```

Argument

`-win` *`window`*

Specifies the window id of the invoking *Clock Tree Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserDelayHistogram -win $_ClockTreeBrowser3
```

schClkTreeBrowserDumpClkTreeBrowser

Description

Dumps information, including the current columns on the *Clock Tree Browser* window, to a file.

Equivalent GUI Operation: The **File** -> **Save Clock Tree** -> **Save All Info** command on the *Clock Tree Browser* window.

Syntax

```
schClkTreeBrowserDumpClkTreeBrowser [-win windowID] [-file
fullPathFileName] [-delim delimSymbol]
```

Arguments

`-win` *`windowID`*

The *Clock Tree Browser* window ID.

`-file` *`fullPathFileName`*

The file name with full path.

`-delim delimSymbol`

The delimiter used for "Full Instance Name" in the log file.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserDumpClkTreeBrowser -win $_ClockTreeBrowser5
-file /verdi/home/tmp/tmp.log -delim "."
```

schClkTreeBrowserDumpWithAstroFormat

Description

Dumps the clock tree from the *Clock Tree Browser* window to a file using the *Astro* format.

Syntax

```
schClkTreeBrowserDumpWithAstroFormat [-win windowID] [-file
FileName] [-delim delimChar]
```

Argument

`-win windowID`

The *Clock Tree Browser* window ID.

`-file fileName`

Specifies the file name with full path.

`-delim delimChar`

Specifies the delimiter character.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schClkTreeBrowserDumpWithAstroFormat -win $_ClockTreeBrowser4
-file $env(TURBO_AU_DIR)/SPSqa71709.log -delim "."
```

schClkTreeBrowserFind

Description

Specifies the query criteria to find the desired tree node. The found clock tree node is highlighted in the *Clock Tree Browser* window or in the *nSchema* window.

Syntax

```
schClkTreeBrowserFind [-win windowID] -all [-instName
InstanceName] [AttributeList]
```

Argument

`-win windowID`

The *Clock Tree Browser* window ID.

`-instName InstanceName`

Specifies the instance name to query in the **Instance** field. The default string for instance name is "*".

`AttributeList`

Specifies the cell name with the `-cellName CellName` option or instance type options with `-ff`, `-latch`, `-macro`, `-outputPort`, `-floating`, `-duplication`, `-stopPin`, `-throughPin`, `-loop`, `-datapin`, `-xor`, `-gateClk`.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schClkTreeBrowserFind -win $_ClockTreeBrowser3 -all -instName
"system.i_cpu.*" -ff -latch -macro -cellName "CCU*"
```

The query criteria above are as follows:

1. The prefix for the **Instance** field is "`system.i_cpu`."
2. The prefix for the **Cell Name** field is "`CCU`" or "`Flip-flop`" or "`Latch`" or "`Marco`" as the instance type.

schClkTreeBrowserHighlightPoint

Description

Highlights colors for instances with stop points, through points, convergent points, and exception type points are specified and are highlighted in the extracted clock tree in the *nSchema* window.

Syntax

```
schClkTreeBrowserHighlightPoint -win [windowID] -StopEnable on|off
-StopColor stopColor -ThroughEnable on|off
-ThroughColor throughColor -ConvergentEnable on|off
-ConvergentColor convergentColor -ExceptedEnable on|off
[-dont_size_cells exceptedColor1] [-dont_buffer_nets
exceptedColor2] [-dont_touch_subtrees exceptedColor3]
```

Argument

`-win`

Specifies the window id.

`-StopEnable on|off`

Enables or disables the color setting of the CTS stop type instance in *nSchema*. The default is *off*.

`-StopColor stopColor`

Specifies the color of the CTS stop type instance.

`-ThroughEnable on|off`

Enables or disables the color setting of the CTS Through type instance in *nSchema*. The default is *off*.

`-ThroughColor throughColor`

Specifies the color of CTS through type instance.

`-ConvergentEnable on|off`

Enables or disables the color setting of the convergent instance in *nSchema*. The default is *off*.

`-ConvergentColor convergentColor`

Specifies the color of the convergent instance.

`-ExceptedEnable on|off`

Enables or disables the color setting of the ICC CTS exception type instance in *nSchema*. The default is *off*.

`-dont_size_cells` *`exceptedColor1`*

Specifies the color of the ICC CTS `don't_size_cell` exception type instance.

`-dont_buffer_nets` *`exceptedColor2`*

Specifies the color of the ICC CTS `don't_buffer_nets` exception type instance.

`-dont_touch_subtrees` *`exceptedColor3`*

Specifies the color of the ICC CTS `don't_touch_subtrees` exception type instance color.

Example

```
schClkTreeBrowserHighlightPoint -win $_ClockTreeBrowser3
-StopEnable off -StopColor ID_YELLOW3 -ThroughEnable off
-ThroughColor ID_BLUE5 -ConvergentEnable off -ConvergentColor
ID_GREEN3 -ExceptedEnable on -dont_size_cells ID_ORANGE3
-dont_buffer_nets ID_ORANGE5 -dont_touch_subtrees ID_GRAY5
```

schClkTreeBrowserLevelHistogram

Description

Creates a level histogram window.

Syntax

```
schClkTreeBrowserLevelHistogram [-win window]
```

Argument

`-win` *`window`*

Specifies the window id of the invoking *Clock Tree Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserLevelHistogram -win $_ClockTreeBrowser2
```

schClkTreeBrowserListLvlInfo

Description

Executes the command to open the *User-defined Level Information* form, which lists cell types and their orders from the instance of the clock source to the selected instance.

Syntax

```
schClkTreeBrowserListLvlInfo [-win ClkBrowserWndId] [-cell
CellNameList] [-save|-load filename]
```

Argument

`-win ClkBrowserWndId`

Specifies the window id.

`-cell cellNameList`

Shows the user-defined cell list, which is separated with spaces and listed in order of the user-defined cell name.

`-save filename`

Saves the current user-defined cell list to a file.

`-load filename`

Loads the user-defined cell list from a file.

Example

```
schClkTreeBrowserListLvlInfo -win $_ClockTreeBrowser4

schClkTreeBrowserSetUserDefinedCell -cell INVX1 INVX2 CLKBUFX2

schClkTreeBrowserSetUserDefinedCell -save /prod1/home/test

schClkTreeBrowserSetUserDefinedCell -load /prod1/home/test
```

schClkTreeBrowserLoadClockTreeDatabase

Description

Loads the specified clock tree extraction results file to the *Clock Tree Browser* window.

Syntax

```
schClkTreeBrowserLoadClockTreeDatabase -file filename
```

Argument

`-file filename`

Specifies the file name to load to the *Clock Tree Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserLoadClockTreeDatabase -file "test.log"
```

schClkTreeBrowserSaveClockTreeDatabase

Description

Saves the *Clock Tree Browser* window contents.

Syntax

```
schClkTreeBrowserSaveClockTreeDatabase -win windowID -file
filename
```

Argument

`-win windowID`

Specifies the window id of the invoking *Clock Tree Browser* window.

`-file filename`

Specifies the file name to save the clock tree extraction results and CTS settings.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserSaveClockTreeDatabase -win $_ClockTreeBrowser2
-file "test.log"
```

schClkTreeBrowserSaveClockTree

Description

Dumps all the paths from the clock source to each register on the *Clock Tree Browser* window to a file. The path information includes the net name, connected instance name, and port name.

Equivalent GUI Operation: The **File** -> **Save Clock Tree** -> **Save Path Info** command on the *Clock Tree Browser* window.

Syntax

```
schClkTreeBrowserSaveClockTree -win windowID -file
fullPathFileName
```

Arguments

`-win` *`windowID`*

The *Clock Tree Browser* window ID.

`-file` *`fullPathFileName`*

Specifies the file name with its full path.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserSaveClockTree -win $_ClockTreeBrowser3 -file \
/verdi/home/tmp/tmp.log
```

schClkTreeBrowserSetCTSFromShowList

Description

Sets an instance pin as a CTS constraint.

Syntax

```
schClkTreeBrowserSetCTSFromShowList [-win window] [-InstStop|
-InstIgnore|-InstPassthrough|-InstUnset [comment] [instance]
[pin]] [-signal source] [-updateTypeCol]
```

Argument

`-InstStop|-InstIgnore|-InstPassthrough|-InstUnset` *`comment instance pin`*

Specifies a CTS constraint type for an instance pin. The instance pin is shown in its instance name, pin name, and comments. The CTS constraint types include the following (you can also leave it blank and not set a constraint type):

- `-InstStop`: Sets as stop type.
- `-InstIgnore`: Sets as ignore type.
- `-InstPassthrough`: Sets as passthrough type.
- `-InstUnset`: Unsets it.

`-signal` *`source`*

Specifies the clock source to set as the local CTS constraint. If this parameter is not set, the global CTS constraint is set.

`-updateTypeCol` *`Constraint`*

Sets the argument to update the settings in the *Constraint* column of the tables in *CTS Constraint Checker Result* form.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schClkTreeBrowserSetCTSFromShowList -InstStop "FF"
"system.i_cpu.i_PCU.\\TDB_reg\[0\] " "GN" -signal "system.clock"
-updateTypeCol (Set "system.i_cpu.i_PCU.\TDB_reg[0] .GN" as local
CTS constraint of "system.clock" by stop type and comment is
"FF").
```

```
schClkTreeBrowserSetCTSFromShowList -InstStop "FF"
"system.i_cpu.i_PCU.\\TDB_reg\[0\] " "GN" -updateTypeCol
(Set "system.i_cpu.i_PCU.\TDB_reg[0] .GN" as global CTS
constraint by stop type and comment is "FF").

schClkTreeBrowserSetCTSFromShowList -InstStop ""
"system.i_cpu.i_PCU.\\TDB_reg\[0\] " "GN"
(Set "system.i_cpu.i_PCU.\TDB_reg[0] .GN" as global CTS
constraint by stop type without comment).

schClkTreeBrowserSetCTSFromShowList -InstUnset ""
"system.i_cpu.i_PCU.\\TDB_reg\[0\] " "GN" -signal "system.clock"
(Unset "system.i_cpu.i_PCU.\TDB_reg[0] .GN" from local CTS
constraint of "system.clock").
```

schClkTreeBrowserSetViolationThreshold

Description

Sets a threshold value for the selected node. At least one instance must be selected first.

Syntax

```
schClkTreeBrowserSetViolationThreshold [-win window]
[-setThreshold value]
```

Argument

`-win` *`window`*

Specifies the window id of the invoking *Clock Tree Browser* window.

`-setThreshold` *`value`*

Sets the threshold value for the selected instances.

Value Returned

Base point with the threshold value listed if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserSetViolationThreshold -win $_ClockTreeBrowser3
-setThreshold 3.5
```

schClkTreeBrowserShowList

Description

Shows the CTS list by the clock source group and sets it to the local CTS constraints by the current selected group.

Syntax

```
schClkTreeBrowserShowList [-selectSignal name [-dump]]
schClkTreeBrowserShowList [-InstStop|-InstIgnore|-InstPassthrough
pin] [-signal name]
```

Argument

`-selectSignal`

Specifies the clock source group to show.

`-dump`

Dumps the current data in the table of the *List for CTS Settings* form.

`-signal`

Specifies the clock source used to set local CTS constraints. The `-InstStop`, `-InstIgnore`, and `-InstPassthrough` parameters are the same for the CTS setting functionality.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserShowList -selectSignal "system.clock"

schClkTreeBrowserSetCTSFromShowList -InstStop "FF" \
"system.i_cpu.i_PCU.\\TDB_reg\[0\] " "GN" -signal "system.clock"
```

schClkTreeBrowserShowTips

Description

Shows the detailed information of the *Cell Name, Type, Full Instance Name, Number of Register, Total Number of Instances to Leaves, Number of Instances to Leaves, Total Current Level Number, Current Level, Total Min Level to Leaves, Min Level to Leaves, Total Max Level to Leaves, Max Level to Leaves, Delay from Source, and Delay to Leaves* columns in the *Clock Tree Browser* window. To enable their information, enable the **Show Tip** command from the right-click option menu.

Syntax

```
schClkTreeBrowserShowTips [-win windowID] [-ShowTips on|off]
```

Argument

`-win`

Specifies the window id of the invoking *Clock Tree Browser* window.

`-showTips on|off`

Turns the **Show Tip** command *on* or *off*.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserShowTips -win $_ClockTreeBrowser5 -ShowTips on
```

schClkTreeBrowserSwitchMode

Description

Switches to different modes in the *Clock Tree Browser* window.

Syntax

```
schClkTreeBrowserSwitchMode -win windowId -mode modeName
```

Argument

`-win`

Specifies the window id for mode switching.

`-mode`

Specifies the mode name to switch the mode.

Example

```
schClkTreeBrowserSwitchMode -win $_ClockTreeBrowser3 -name Mode1
```

schClkTreeBrowserSyncSelectedInstance

Description

This toggle command turns the instance synchronization between the *Clock Tree Browser* window and *nTrace on* or *off*.

Syntax

```
schClkTreeBrowserSwitchMode -win windowId -nTrace on|off
```

Argument

`-nTrace on|off`

Specifies whether to synchronize the instance selection between the *Clock Tree Browser* window and *nTrace*. The default is *off*.

`-win windowId`

Specifies the window id of the invoking *Clock Tree Browser* window.

Example

```
schClkTreeBrowserSyncSelectedInstance -win $_ClockTreeBrowser4
-nTrace on
```

schClkTreeBrowserViolationCheck

Description

Checks from the base point for instances with insertion delay values greater than the threshold.

Syntax

```
schClkTreeBrowserViolationCheck [-win window]
```

Argument

`-win` *`window`*

Specifies the window id of the invoking *Clock Tree Browser* window.

Value Returned

Violation instance listed if successful; otherwise, returns 0.

Example

```
schClkTreeBrowserViolationCheck -win $_ClockTreeBrowser3
```

schClkTreeBrowserWnd

Description

The `schClkTreeBrowserWnd` command is used to specify highlight colors for markers, selected rows, and violation points in the *Clock Tree Browser* window, or set and unset markers, or dumps delay and level margin from the specified base instance to the currently selected instance.

Description - Highlight Color

Specifies highlight colors for markers, selected rows, and violation points.

Syntax - Highlight Color

```
schClkTreeBrowserWnd [-MarkerColor color] [-SelectedColor color]
[-ViolationColor color]
```

Argument - Highlight Color

`-MarkerColor` *`color`*

Specifies the highlight color for markers in the *Clock Tree Browser* window.

`-SelectedColor` *`color`*

Specifies the highlight color for selected rows in the *Clock Tree Browser* window.

`-ViolationColor` *`color`*

Specifies the highlight color for violation points in the *Clock Tree Browser* window.

Value Returned - Highlight Color

Returns 1 if successful; otherwise, returns 0.

Example - Highlight Color

```
schClkTreeBrowserWnd -MarkerColor ID_RED4
schClkTreeBrowserWnd -SelectedColor ID_YELLOW7
schClkTreeBrowserWnd -ViolationColor ID_RED5
```

Description - Set Marker

Sets the marker on an instance to calculate delay and level margins.

Syntax - Set Marker

```
schClkTreeBrowserWnd [-win window] [-setmarker instance]
```

Argument - Set Marker

`-win` *`window`*

Specifies the window id of the invoking *Clock Tree Browser* window.

`-setmarker instance`

The instance to set as the marker on the clock tree.

Value Returned - Set Marker

Marked instance name if successful; otherwise, returns 0.

Example - Set Marker

```
schClkTreeBrowserWnd -win $_ClockTreeBrowser3 -setmarker
"system.i_cpu.i_ALUB.i_alu.U105@#system.i_cpu.i_PCU.U216"
```

Description - Unset Marker

Unsets the marker.

Syntax - Unset Marker

```
schClkTreeBrowserWnd [-win window] [-unsetmarker]
```

Argument - Unset Marker

`-win window`

Specifies the window id of the invoking *Clock Tree Browser* window.

Value Returned - Unset Marker

Returns 1 if successful; otherwise, returns 0.

Example - Unset Marker

```
schClkTreeBrowserWnd -win $_ClockTreeBrowser3 -unsetmarker
```

Description - Dump Delay and Level

Dumps delay and level margins from the specified base instance to the currently selected instance. You need to select one instance first and then, use this command to dump.

Syntax - Dump Delay and Level

```
schClkTreeBrowserWnd [-win window] [-dumpDeltaMark]
```

Argument - Dump Delay and Level

`-win window`

Specifies the window id of the invoking *Clock Tree Browser* window.

`-dumpDeltaMark`

Dumps delay and level margin information.

Value Returned - Dump Delay and Level

Delay and level margin information if successful; otherwise, returns 0.

Example - Dump Delay and Level

```
schClkTreeBrowserWnd -win $_ClockTreeBrowser3 -setmarker \

"system.i_cpu.i_ALUB.i_alu.U105@#system.i_cpu.i_PCU.U216"
schClkTreeBrowserWnd -win $_ClockTreeBrowser3 -select \

"system.i_cpu.i_ALUB.i_alu.U105@#system.i_cpu.i_PCU.\\PC_reg\[3\]
"
```

```
schClkTreeBrowserWnd -win $_ClockTreeBrowser3 -dumpDeltaMark

After executing these commands, the following values would be
returned for the dump delay and level results:
21 schClkTreeBrowserWnd Delay Delta: 435.00 Level Delta: 2
```

schClkTreeEditMode

Description

Edits the mode list.

Syntax

```
schClkTreeEditMode [-add modeName] [-delete modeNameList]
[-mode modeName [-cell|-signal name value radix] [-loadSDC
SDCFileName][-delete name]] [-load|-save modeFileName] [-update
modeName]
```

Argument

`-add modeName`

Specifies a mode to add to the mode list.

`-delete modeNameList`

Deletes mode(s) from the mode name list.

`-mode modeName`

Specifies the mode for editing.

`[-cell|-signal Name Value Radix]`

Specifies the cell port or full hierarchical signal name (`name`), the constant value (`value`), and the radix value (`radix`) for the selected cell (`-cell`) or signal (`-signal`).

`-loadSDC SDCFileName`

Loads an SDC file name to the mode specified with `-mode`.

`-delete name`

Deletes the constant settings from the mode list.

NOTE: **-delete name** and **-delete modeNameList** are different. If there is a **-update** default, the mode name list will be deleted; if there is no **-update** default, the constant settings in the mode list will be deleted.

`-load|-save modeFileName`

Loads or saves the mode list from/to the specified file name.

`-update modeName`

Updates the contents of the specified mode.

Example

```
schClkTreeEditMode -win $_nSchema3 -add Mode0
```

Adds a new mode as `Mode0`.

```
schClkTreeEditMode -win $_nSchema3 -mode default -load
SPSqa77806_LoadModeList_Mode0.sdc
```

Loads contents of the SDC file (`file name: SPSqa77806_LoadModeList_Mode0.sdc`) to the default mode.

```
schClkTreeEditMode -win $_nSchema3 -update Mode0 -name
"system.i_cpu.i_ALUB.n740" -value 0 -radix Binary -signal
```

Sets the signal name, radix, and value in `Mode0`.

```
schClkTreeEditMode -win $_nSchema2 -update default -name
"system.clock" -delete
```

Deletes a setting (`setting name: system.clock`) in the default mode.

```
schClkTreeEditMode -win $_nSchema2 -delete Mode1
```

Deletes the setting in `Mode1`.

schConfigTemplate

Description

Configures desired synchronizer templates.

Syntax

```
schConfigTemplate [-win window] [-tmplName_<index>
<template_name>] [-connectName_<index> <connectivity list>]
[-connectType_<index> <connectivity type>]
```

NOTE: The number of list in `-connectName_X` and the number of list in `-connectType_X` must be the same.
The number of index of `-connectName_X` and `-connect_X` must also be the same.

Argument

`-connectName_<index> <connectivity list>`

Specifies the name list of a flop-flop or logic cell. The name can be cell name or cell type depending on the cell type (See description of `-connectType`). Each object in the list can contain multiple names separated by colon `':'`. An `OR` connector in this name list is the same as the `'|'` symbol in the default template. For example:

```
Template sync_1 Type_A : Reg(start) -> Comb* -> FF|Latch
-> Inverter|Buffer|Assignment -> Reg(end)
```

`-connectType_<index> <connectivity type>]`

Specifies a list of connect types. The legal types and their acceptable names follow:

`cell`: Acceptable name is the cell name of a storage type cell.

`flipflop`: Acceptable name is "`Flip-Flop`" or "`Latch`".

"`oneOf`" and "`anyOf`": Acceptable name is "`Comb`", "`Mux`", "`And`", "`Or`", "`Buffer`", "`Inverter`", or "`Assignment`".

"`seqOf`": Acceptable name is "`Comb`", "`Mux`", "`And`", "`Or`", "`Buffer`", "`Inverter`", "`Assignment`", or "`Comb*`".

"`logiccell`": Acceptable name is cell name of a combinational logic cell.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schConfigTemplate -win $_nSchema2 -tmplName_0 "user_sync"
-connectName_0 "And" "Flip-Flop" "Buffer" -connectType_0 "oneOf"
"flipflop" "oneOf"

schConfigTemplate -win $_nSchema2 -tmplName_0 "my_sync"
-connectName_0 Inverter" "FD2" "Inverter" "FD1" "Comb"
-connectType_0 "oneOf" "cell" "oneOf" "cell" "anyOf"

schConfigTemplate -win $_nSchema2 -tmplName_0 "sync_test"
-connectName_0 "Inverter:Buffer:And" "Flip-Flop:Latch" "Comb"
-connectType_0 "seqOf" "flipflop" "anyOf"

schConfigTemplate -win $_nSchema2 -tmplName_0 "T1" -connectName_0
"OR4" -connectType_0 "logiccell" -tmplName_1 "T2" -connectName_1
"AN3" "IV" -connectType_1 "logiccell" "logiccell"

schConfigTemplate -win $_nSchema2 -tmplName_0 "user_sync" -

schConfigTemplate -win $_nSchema2 -tmplName_0 "LogicCell2"
-connectName_0 "NR2" "IV" "ND2" -connectType_0 "logiccell"
```

```
"logiccell" "logiccell" -tmplName_1 "LogicCell1" -connect "AO2"
"AN2" "ND2" -connectType_1 "logiccell" "logiccell"
"logiccell"
```

schCreateWindow

Description

Extracts a clock tree from the specified SDC file(s), the CTS file(s), or the setting file(s).

Syntax

```
schCreateWindow -clockTree [-settingFile] [-sdcFile]
[-ctsAstro2005File] [-ctsAstro2007File] [-ctsApolloFile]
[-ctsFeFile] [-ctsIccFile] [-append] [-checkValidation] [-design
designName] [-delim delimiter]
```

Argument

`-append`

Loads file(s) with the append mode. This option is supported only for SDC and CTS file(s) loading.

`-checkValidation`

Loads file(s) with checking validation.

`-clockTree`

Generates the *Clock Tree Browser* window.

`-ctsApolloFile`

Specifies the Apollo CTS file to import.

`-ctsAstro2005File`

Specifies the Astro CTS file in the 2005 version to import.

`-ctsAstro2007File`

Specifies the Astro CTS file in the 2007 version to import.

`-ctsFeFile`

Specifies the First Encounter CTS file to import.

`-ctsIccFile`

Specifies the ICC CTS file to import.

`-design designName`

Specifies the SDC/CTS top design name. If not specified, the default is the first top design. This option is specified more than once.

`-delim` *`delimiter`*

Specifies the SDC/CTS delimiter. If not specified, the default is "/". This option is specified more than once.

`-sdcFile`

Specifies the SDC file to import.

`-settingFile`

Specifies the setting file to import. When this option is specified, all settings are cleared and the specified setting file is imported. If the specified option is not `-settingFile`, the later specified files have higher priority than the previous file, when setting conflicts occur.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schCreateWindow -clockTree -settingFile "A" -sdcFile "B" \
-sdcFile "C" -ctsAstro2005File "D"

A has A.1. source   Clk1
      A.2. constant Clk2  value:0
      A.3. CTS      instA.portI  stop
B has B.1. source   Clk1
      B.2. constant Clk2  value:1       //conflict with A.2
      B.3. constant Clk3  value:1
C has C.1. source   Clk1
      C.2. constant Clk3  value:1       //conflict with B.3
D has D.1. CTS      instA.portI  ignore //conflict with A.3

The clock tree setting will have:
A.1. source   Clk1
B.2. constant Clk2  value:1
C.2. constant Clk3  value:1
D.1. CTS      instA.portI  ignore

schCreateWindow -clockTree -sdcFile sdcA -design "top" -delim "/"
-sdcFile sdcB -design "system" -delim "@" -sdcFile sdcC
-ctsAstro2005File ctsA
```

schCrossingPath

Description

Checks crossing paths between the specified clock domains.

Syntax

```
schCrossingPath [-win window] [-fromSrc [src1] [src2]...] [-toSrc
[src1] [src2]...] [-fromDomainNo [domain1][domain2]...]
[-toDomainNo [domain1][domain2]...] [-synchronizer [synchronizer
name1]] [-synchronizer [synchronizer name2]]...
[-chkSubDomain on|off] [-checkConvergence on|off]
[-checkDivergence on|off] [-checkReconvergence on|off] [-passScope
scopeList]
```

Arguments

`-win window`

Specifies the window id of the invoking *nSchema* window.

`-fromDomainNo`

Specifies the ID number(s) of the sub-domains belonging to the selected from domain.

`-toDomainNo`

Specifies the ID number(s) of the sub-domains belonging to the selected to domain.

`-chkSubDomain`

When *on*, checks whether crossing paths between sub-domains belong to the selected domains. When *off*, do not check sub-domains. If this parameter is not assigned, the default is *on*.

`-synchronizer`

Specifies synchronizer template set on the **Synchronizer Setting** tab of the *Check Crossing Paths* form.

`-fromSrc`

Specifies the *from* clock source.

`-toSrc`

Specifies the *to* clock source.

`-checkConvergence`

Checks for potential convergence type structure problems after crossing the path extraction.

`-checkDivergence`

Checks for potential divergence type structure problems after crossing the path extraction.

`-checkReconvergence`

Checks for potential re-convergence type structure problems after crossing the path extraction.

`-passScope` *`scopeList`*

Lists the specified scope names. `<->` is used to separate *scope1* and *scope2*, such as "system.i_pram<->system.i_cpu.i_ALUB". "*" is used to represent all scopes, such as "system.i_pram<->*".

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schCrossingPath \
-fromSrc "system.clock" "system.i_cpu.i_CCU.C19"
"system.i_cpu.i_CCU.C5" "system.i_cpu.i_CCU.C6"
"system.i_cpu.i_ALUB.S1" \
-fromDomainNo "1.f" "1.r" "1.1.r" "1.2.f" "1.5.r" "2.f" "2.2" \
-toSrc "system.clock" "system.i_cpu.i_CCU.C19"
"system.i_cpu.i_CCU.C5" "system.i_cpu.i_CCU.C6"
"system.i_cpu.i_ALUB.S1"  \
-toDomainNo "1.f" "1.r" "1.1.r" "1.2.f" "1.5.r" "2.f" "2.2" \
-chkSubDomain on

schCrossingPath \
-fromSrc "system.clock" "system.i_cpu.i_CCU.C19"
"system.i_cpu.i_ALUB.S1" \
-fromDomainNo "1.r" "1.2.r" \
-toSrc "system.clock" "system.i_cpu.i_CCU.C19"
"system.i_cpu.i_CCU.C5" "system.i_cpu.i_CCU.C6"
"system.i_cpu.i_ALUB.S1" \
-toDomainNo "1.r" "1.1.3.r" \
-synchronizer "\(Template sync_configuration\)NewXXX\(AO2 ->
AN2-> ND2\)" \-checkConvergence \-checkDivergence \
-checkReconvergence

schCrossingPath -fromSrc "bug.clk" "bug.clk2" -toSrc "bug.clk"
"bug.clk2" -synchronizer "\(Template DMux\)"
```

Specifies the synchronizer template as *Template DMux*\.

```
schCrossingPath -fromSrc "system.clock" "system.i_cpu.i_ALUB.S1"
-toSrc "system.clock" "system.i_cpu.i_ALUB.S1" -passScope
"system.i_pram <->system.i_cpu.i_ALUB" -chkSubDomain off
```

schCrossPathFilter

Description

Specifies the filter rules for the **Crossing Paths**, **CDC Correlation Details**, and **Register Pair Details** tables in the *Crossing Paths* form.

Syntax

```
schCrossPathFilter [-win window] [-filter on [-addFilter
FilterRuleString] [-updateFilter FilterRuleString] [-delFilter
FilterRuleString] [-exchangeFilter FilterRuleString1
FilterRuleString2] [-addCDCCorrelationFilter FilterRuleString]
[-updateCDCCorrelationFilter FilterRuleString]
[-delCDCCorrelationFilter FilterRuleString]
[-exchangeCDCCorrelationFilter FilterRuleString1
FilterRuleString2] [-filterSyncPath on|off]
[-filterNoConvergencePath on|off]
[-filterNoDivergencePath on|off] [-filterNoReconvergencePath
on|off] [-filterConvergenceProblem on|off]
[-filterDivergenceProblem on|off]
[-filterReconvergenceProblem on|off]]
[-load filename] [-save filename [SaveRuleStrings]]
```

The `SaveRuleStrings` syntax includes the following:

```
-saveFilterSyncPath
-saveFilterNoConvergencePath
-saveFilterNoDivergencePath
-saveFilterNoReconvergencePath
-saveFilterConvergenceProblem
-saveFilterDivergenceProblem
-saveFilterReconvergenceProblem
```

Argument

`-win`

Specifies the window id.

`-addFilter` *`FilterRuleString`*

Specifies a filter rule to add.

`-delFilter` *`FilterRuleString`*

Specifies a filter rule to delete.

`-updateFilter` *`FilterRuleString1 FilterRuleString2`*

Edit a filter rule by updating from *filter rule 1* to *filter rule 2*.

`-exchangeFilter` *`FilterRuleString1 FilterRuleString2`*

Exchanges the priority order between *filter rule 1* and *filter rule 2*.

`-addCDCCorrelationFilter` *`FilterRuleString`*

Specifies a CDC correlation filter rule to add.

`-updateCDCCorrelationFilter` *`FilterRuleString`*

Edits a CDC correlation filter rule by updating from *filter rule 1* to *filter rule 2*.

`-exchangeCDCCorrelationFilter` *`FilterRuleString1 FilterRuleString2`*

Exchanges the priority order between *filter rule 1* and *filter rule 2* for CDC correlation filter rules.

`-delCDCCorrelationFilter` *`FilterRuleString`*

Deletes a CDC correlation filter rule.

`-filterNoConvergencePath`

Adds the option to filter the paths without convergence in the *Crossing Paths* table.

`-filterNoDivergencePath`

Adds the option to filter the paths without divergence in the *Crossing Paths* table.

`-filterNoReconvergencePath`

Adds the option to filter the paths without re-convergence in the *Crossing Paths* table.

`-filterConvergenceProblem`

Adds this option to filter convergence cases in the *CDC Correlation Details* table.

`-filterDivergenceProblem`

Adds this option to filter divergence cases in the *CDC Correlation Details* table.

`-filterReconvergenceProblem`

Adds this option to filter the re-convergence cases in the *CDC Correlation Details* table.

`-save` *`filename`* `SaveRuleStrings`

Saves filter rules for the *CDC Correlation Details* table to a `.flt` file.

```
-saveFilterSyncPath
-saveFilterNoConvergencePath
-saveFilterNoDivergencePath
-saveFilterNoReconvergencePath
-saveFilterConvergenceProblem
```

```
-saveFilterDivergenceProblem
-saveFilterReconvergenceProblem
```

`-load filename`

Loads filter rules for the *CDC Correlation Details* table from a file.

Filter Rule String:

The format of `filterRuleString` in the previous descriptions includes data source names, filter conditions, data values, filter actions, and filter status. Filter actions and status are joined to the rule string with a `','` character separator.

- Available data source types: "`FromRegister`", "`ToRegister`", "`FromRegisterDataSignal`", "`ToRegisterDataSignal`", "`Synchronizer`", and "`Level`"
- Available filter condition types: "`==`", "`!=`", "`>`", "`<`"
- Available filter connectors: "`AND`", "`OR`"
- Available filter status: "`on`", "`off`"
- Available filter actions: "`Hide`" or a color name

For example, assume you have the following data source and conditions:

```
FromRegister != system_i.cpu.i_ALUB.\ACC_tmp_reg[1]
Synchronizer == AN2->NR2
Level > 15
```

Connector setting is "`AND`". The filter action is set to visible with highlight color RED and this filter rule is in use. The complete filter rule is as follows:

```
"(FromRegister != system_i.cpu.i_ALUB.\ACC_tmp_reg[1]) AND
(Synchronizer == AN2->NR2) AND (Level > 15),on,ID_RED"
```

As another example, assume you have the following data source and conditions:

```
ToRegister == system_i.cpu.i_ALUB.\ACC_tmp_reg[1]
Level == 100
```

Connector setting is "`OR`". The filter action is set to hide, but this filter rule is not in use so the complete filter rule string is as follows:

```
"(ToRegister == system_i.cpu.i_ALUB.\ACC_tmp_reg[1]) AND (Level
== 100),off,Hide"
```

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schCrossPathFilter -win $_nSchema1 -addFilter  \
"(ToRegister == system.i_cpu.i_ALUB.\\ACC_reg*) AND (Level <
100),off,ID_BLACK"

schCrossPathFilter -win $_nSchema1 -updateFilter \"(FromRegister
== system.i_cpu.i_ALUB.\\ACC_tmp_reg*) AND (ToRegister ==
system.i_cpu.i_ALUB.\\ACC_reg*),on,ID_BLACK" \ "(FromRegister ==
system.i_cpu.i_ALUB.\\ACC_tmp_reg*) AND (ToRegister ==
system.i_cpu.i_ALUB.\\ACC_reg*),off,ID_BLACK"
```

NOTE: Highlight color of the filter rule in the *Register Pairs* table can use the Tcl command.

Use "`schShowCrossingPath [-win window] -dump`" to dump. If the row is visible and is set to a highlight color except `ID_BLACK`, the color is printed at the end of the row. Here is an example below:

```
***** BEGIN DUMP RegPairList *****
system.i_cpu.i_ALUB.\IXR_reg[0]  system.i_cpu.i_PCU.\IDR_reg[2]
69 ID_BLUE5
system.i_cpu.i_ALUB.\IXR_reg[0]  system.i_cpu.i_PCU.\IDR_reg[1]
39 ID_BLUE5
system.i_cpu.i_ALUB.\IXR_reg[0]  system.i_cpu.i_PCU.\IDR_reg[0]
4 ID_BLUE5
system.i_cpu.i_ALUB.\IXR_reg[1]  system.i_cpu.i_PCU.\IDR_reg[7]
191
system.i_cpu.i_ALUB.\IXR_reg[1]  system.i_cpu.i_PCU.\IDR_reg[6]
165
.
.
.

schCrossPathFilter -win $_nSchema1 -addFilter "FromRegister ==
xxx,on,Hide"
-addCDCCorrelationFilter  "Name ==/!= xxx,on,Hide"
-exchangeCDCCorrelationFilter  "Name ==/!= xxx,on,Hide"
-delCDCCorrelationFilter  "Name ==/!= xxx,on,Hide"
-filterSyncPath on
-filterNoConvergencePath off
-filterNoDivergencePath on
-filterNoReconvergencePath on
-filterConvergenceProblem off
-filterDivergenceProblem on
-filterReconvergenceProblem on

schCrossPathFilter -win $_nSchema1 -save "/prod1/home/magic_tu/
SOL2/deb6.1/test.fil"

schCrossPathFilter -win $_nSchema1 -load "/prod1/home/magic_tu/
SOL2/deb6.1/test.fil"
```

schCrossPathSetColumn

Description

Configures the visible column settings in the *Crossing Path, CDC Correlation Details,* and *Register Pair Details* table of the *Crossing Paths* form.

Syntax

```
schCrossPathSetColumn [-setPathColumn columnName1 columnName2
columnName3...] [-setCDCCorrelationColumn columnName1 columnName2
columnName3...]
```

Argument

`-setPathColumn columnName1 columnName2 columnName3...`

Specifies the column names that are shown in the *Crossing Paths* table in order from left to right.

Legal column name for the *Crossing Paths* table are: "*Path*", "*From Clock Source*", "*To Clock Source*", "*With Synchronizer*", "*Without Synchronizer*", "*Convergence*", "*Divergence*", "*Reconvergence*" and "*Filtered/Colored Register Pair Num*".

Legal column names for the *Register Pair Details* table are: "*From Register*", "*To Register*", "*From Register Data Signal*", "*To Register Data Signal*", "*Synchronizer*" and "*Level*".

`-setCDCCorrelationColumn`

Specifies the column names that will be shown in the *CDC Correlation Details* table in order from left to right.

Legal column name for the *CDC Correlation Details* table are: "*Problem Type*", "I*nstance/Signal Name*" and "*Path Number*".

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schCrossPathSetColumn -setPathColumn "Path" "From Clock Source"
"To Clock Source"
```

The example shows the columns "*Path*", "*From Clock Source*", and "*To Clock Source*" in the *Crossing Paths* table from left to right.

```
schCrossPathSetColumn -setColumn "From Register" "To Register"
"Synchronizer" "Level"
```

The example shows the columns "*From Register*", "*To Register*", "*Synchronizer*", and "*Level*" in the *Register Pair Details* table from left to right.

```
schCrossPathSetColumn -setColumn "Synchronizer" "Level" "To
Register Data Signal"
```

The example shows the columns "*Synchronizer*", "*Level*" and "*To Register Data Signal*" in the *Register Pair Details* table from left to right.

```
schCrossPathSetColumn -setCDCCorrelationColumn "Problem Type"
"Instance/Signal Name" "Path Number"
```

The example shows the columns "*Problem Type*", "*Instance/Signal Name*", and "*Path Number*" in the *CDC Correlation Details* window from left to right.

schCrossPathSort

Description

Sorts the visible columns in the *Register Pair Details* table of the *Crossing Paths* window.

Syntax

```
schCrossPathSort [-win window] [-sort column]
```

Argument

`-win window`

Specifies the window id of the *Crossing Paths* window.

`-sort column`

Sorts the columns that are shown in the *Register Pair Details* table in order. In the *Detail View* mode, the columns include *Synchronizer*, *Level, From Register, To Register, From Register Data Signal,* and *To Register Data Signal* columns. In the *Fan-in View* mode, the columns include: *To Register* and *Fan-in Number* columns. In the *Fan-out View* mode, the columns include: *From Register* and *Fan-out Number* columns.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schShowCrossingPath -sortRegPair "To Register Data Signal"

schShowCrossingPath -sortRegPair "From Register"
```

schEditClkSrc

Description

Edits the clock source tree in the *Clock Tree Setting* form, including dumping the clock source tree in the **CTS** tab and editing the clock tree grouping in the **SDC Settings** tab.

Syntax

```
schEditClkSrc [-dumpCTSClockTree]
schEditClkSrc [-delClockGroup groupNameList] [-ungroupClkSrc
clockSource -clkGroup clockSourceRoot] [-addClockGroup
newGroupName -rootClkSrc sourceListToGroup][-moveClkSrcToGroup
clockSource [-clkFromGroup fromGroup] -clkToGroup toGroup]
```

Arguments

`-dumpCTSClockTree`

Dumps results to the `turbo.log` file in the `log` directory.

`-delClockGroup` *`groupNameList`*

Ungroups the selected clock group.

`-ungroupClkSrc` *`clockSource`*

Ungroups a clock source from a group.

`-clkGroup` *`clockSourceRoot`*

Specifies the root clock source where the grouped clock source belongs.

`-addClockGroup` *`newGroupName`*

Creates a new clock group.

`-rootClkSrc` *`sourceListToGroup`*

Specifies a list of clock sources to be grouped.

`-moveClkSrcToGroup` *`clockSource`*

Moves a clock source from a group (if the group already exists) to another group.

`-clkFromGroup` *`fromGroup`*

Specifies the group where the clock source comes from if this clock source already exists in a group. *fromGroup* refers to name of the group where the clock source comes from.

`-clkToGroup` *`toGroup`*

Specifies the group where the clock source is moved. *toGroup* refers to the name of the group where the clock source is moved.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schEditClkSrc -dumpCTSClockTree

schEditClkSrc -delClockGroup "CLK_GROUP1" "NewGroup1"

schEditClkSrc -ungroupClkSrc "system.i_CCU.C6_reg.Q" -clkGroup
"NewGroup3"
schEditClkSrc -addClockGroup "NewGroup2" -rootClkSrc
"system.clock"  "system.reset"
schEditClkSrc -moveClkSrcToGroup "system.reset" -clkFromGroup
"NewGroup2" -clkToGroup "NewGroup3"

schEditClkSrc -moveClkSrcToGroup "system.i_CCU.C6_reg.Q"
-clkToGroup "NewGroup2"
```

schExportClockCTSSetting

Description

Exports all local CTS settings and global CTS settings to a file in the Astro, Apollo, FirstEncounter, or ICC (IC Compiler) format.

Syntax

```
schExportClockCTSSetting [-exportCTSFile fileName]
[-design topDesign] [-delim delimiter] [-vendor vendorType]
[-sigBase clockSource] [-selectSignal signalName]
[option_CTSsetting_list]
```

Argument

`-exportCTSFile` *`fileName`*

Specifies the file name to export.

`-delim` *`delimiter`*

Specifies the delimiter.

`-design` *`topDesign`*

Specifies the name of the top design.

`option_CTSsetting_list`

Specifies the instance attribute when the clock tree is extracted. The clock tree settings include:

- `-instStop inst_stop_type_list`
  Specifies the instance to be stopped when the clock tree is extracted.
- `-instIgnore inst_ignore_type_list`
  Specifies the instance to be ignored when the clock tree is extracted.
- `-instPassthrough passthrough_type_list`
  Specifies the instance to be passed through when the clock tree is extracted.

`-selectSignal` *`signalName`*

Specifies one or more clock sources to be exported.

`-sigBase` *`clockSource`*

Specifies the local CTS settings of a root clock source.

`-vendor` *`vendorType`*

Specifies the vendor's format, `Astro 2005`, `Astro 2007`, `Apollo`, `FirstEncounter`, or `ICC`, to export.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schExportClockCTSSetting -exportCTSFile "test.fe" \
-design "system" -delim "/" -vendor "FirstEncounter" \
-sigBase system.i_cpu.i_CCU.C6_reg.Q \
-instStop "system.i_cpu.i_ALUB.U282" "A" \
-instStop "system.i_cpu.i_ALUB.U282" "B" \
-instStop "system.i_cpu.i_ALUB.U282" "C" \
-instStop "system.i_cpu.i_ALUB.U282" "D" \
-instStop "system.i_cpu.i_ALUB.U282" "Z" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "zero" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "b\[7:0\]" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "a\[7:0\]" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "cin" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "carry" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "select\[2:0\]" \
-instPassthrough "system.i_cpu.i_ALUB.\\IXR_reg\[6\] " "CD" \
-instPassthrough "system.i_cpu.i_ALUB.\\IXR_reg\[6\] " "CP" \
```

```
-instPassthrough "system.i_cpu.i_ALUB.\\IXR_reg\[6\] " "D" \
-instPassthrough "system.i_cpu.i_ALUB.\\IXR_reg\[6\] " "Q" \
-selectSignal "system.i_cpu.i_CCU.C6_reg.Q" \
"system.i_cpu.i_ALUB.U289.D" \
"system.i_cpu.i_ALUB.U289.C" \
"system.i_cpu.i_ALUB.\\IXR_reg\[5\] .Q" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .Q" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .D" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .CP"

schExportClockCTSSetting -exportCTSFile "test.fe" \
-design "system" -delim "/" -vendor "Astro" \
-sigBase system.i_cpu.i_CCU.C6_reg.Q \
-instStop "system.i_cpu.i_ALUB.U282" "A" \
-instStop "system.i_cpu.i_ALUB.U282" "B" \
-instStop "system.i_cpu.i_ALUB.U282" "C" \
-instStop "system.i_cpu.i_ALUB.U282" \ "D" \
-instStop "system.i_cpu.i_ALUB.U282" "Z" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "zero" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "b\[7:0\]" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "a\[7:0\]" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "cin" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "carry" \
-instIgnore "system.i_cpu.i_ALUB.i_alu" "select\[2:0\]" \
-selectSignal "system.i_cpu.i_CCU.C6_reg.Q" \
"system.i_cpu.i_ALUB.U289.D" \
"system.i_cpu.i_ALUB.U289.C" \
"system.i_cpu.i_ALUB.\\IXR_reg\[5\] .Q" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .Q" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .D" \
"system.i_cpu.i_ALUB.\\IXR_reg\[4\] .CP"

schExportClockCTSSetting -exportCTSFile \
"$env(TURBO_AU_DIR)/golden.log" \
-design "top" -delim "/" -vendor "ICC" -sigBase "top.clk" \
-instStop "top.eco_i3" "A" -instStop "top.eco_i0" "A" \
-instIgnore "top.eco_i1" "J" -instIgnore "top.eco_i0" "B" \
-instIgnore "top.eco_i3" "Z" -instPassthrough "top.eco_i4" "D" \
-instPassthrough "top.eco_i1" "Q" \
-instPassthrough "top.eco_i4" "CP" "Invert" \
-instPassthrough "top.eco_i2" "A" "Non-invert" \
-selectSignal "top.clk"
```

schExtractClockDomain

Description

Extracts all clock domains for the design.

Syntax

```
schExtractClockDomain [-win window] [-cksour clockSourceList]
[-clkSrcOnly on|off] [-reglist on|off] [-stopOnMacroCell on|off]
[-loadFile filename] [-saveFile filename] [-sdcFile filename
[-checkValidation]] [-mergeCell MergeCellOption] [-latchEnAsClk
on|off] [-loadFile filename [-checkValidation]] [-saveSdcFile
filename] [-design designName] [-delim delim] [-constant
constantName constValue] [-constCellPort cellPortName constValue]
```

Arguments

`-win window`

Specifies the window id of the invoking *nSchema* window.

`-cksour clockSourceList`

Specifies the clock source.

`-clkSrcOnly on|off`

This command line option, corresponding to the **Show Clock Source Only** option in the **Options** tab of the *Clock Tree Setting* form, controls extraction results. When the `clkSrcOnly` option is *on*, gated clock domain is merged to its source clock domain; otherwise, gated clock domain is listed. The default option is *on*.

`-reglist on|off`

The command line option, corresponding to the **Extract Fan-in Register List for Unresolved Check Domain** option in the **Options** tab of the *Clock Tree Setting* form, specifies whether the fan-in register list of an unresolved clock domain should be kept during extraction. When the `reglist` option is *on*, the list is kept; otherwise, the list is not kept.

`-stopOnMacroCell on|off`

The command-line option, corresponding to the **Treat Macro Clock Ports as Clock Source** option in the **Options** tab of the *Clock Tree Setting* form, specifies whether *nSchema* stops on the clock pin of macro cells during clock tree tracing actions.

`-loadFile filename`

Specifies the file to be loaded.

`-saveFile` *`filename`*

Specifies the file to be saved.

`-sdcFile` *`filename`* `[-checkValidation]`

Confirms whether the constant cell port, constant signal, or known clock source in the SDC file to be loaded exists in the current design.

`-loadFile` *`filename`* `[-checkValidation]`

Confirms whether the constant cell port, constant signal, or known clock source in the file to be loaded exists in the current design.

`-mergeCell` *`MergeCellOption`*

Specifies whether to merge gated clock cells (`GATECLK`), combinational logic cells (`CombLogic`), or both (`GATECLK CombLogic`).

`-latchEnAsClk on|off`

Specifies whether to ignore latch-enable ports during the clock domain extraction. This option is enabled by default.

`-saveSdcFile` *`filename`*

Specifies the file name to save.

`-design` *`designName`*

Specifies the top design name of the SDC file to be saved.

`-delim delim`

Specifies the delimiter of the SDC file to be saved.

`-constant` *`signalName constValue`*

Specifies a constant signal with an assigned value to find the clock source. The signal name is defined by a full hierarchal path. The format of `constValue` is a radix type Decimal, Hex, Octal or Binary with a number or "Don't care".

`-constCellPort` *`cellPortName constValue`*

Specifies a constant cell port with an assigned value to find the clock source. The cell port name is defined by the name of the cell appended with "." and the name of the port. For example, if you have a cell called "AN25" and a port called "A", then specify "AN25.A" as the cell port name. The format of `constValue` is a radix type Decimal, Hex, Octal, or Binary with a number or "Don't care".

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schExtractClockDomain -win $_nSchema2 -cksour
-constant signalName
(Binary|Hex|Octal|Decimal|dontcare)[value]-constant ....
schExtractClockDomain -win $_nSchema2 -cksour "system.clock"
"system.i_cpu.i_ALUB.n750"
-constant "system.i_cpu.i_ALUB.n750" dontcare
-constant "system.i_cpu.i_ALUB.n751" Binary 111
-cksour "netName" "netName" "netName"....
-cksour "system.clock" "system.i_cpu.i_ALUB.n750"
schExtractClockDomain -win $_nSchema2 -reglist on
schExtractClockDomain -win $_nSchema2 -clkSrcOnly on -mergeCell
GATECLK CombLogic
```

`MergeCellOption` allows you to specify what kind of cell should be merged when the `clkSrcOnly` option is *on*.

`MergeCellOption` is available only when the `clkSrcOnly` option is *on*. You can specify if you want to merge both `GATECLK` and/or `CombLogic`. At least one of the two options should be enabled when `clkSrcOnly` is on; otherwise, both `GATECLK` and `CombLogic` are set to be merged by default.

```
schExtractClockDomain -saveSdcFile "test.sdc" \
-design "system.i_cpu" \
-delim "/"\
-cksour "system.i_cpu.clock" "system.i_cpu.R_W"
"system.i_cpu.reset" \
-constant "system.i_cpu.reset" Binary 0 \
-constant "system.i_cpu.i_PCU.\\TR\[2\] " dontcare \
-constCellPort "MUX21H.A" dontcare \
-constCellPort "AN25.A" dontcare

schExtractClockDomain -clkSrcOnly on -mergeCell GATECLK CombLogic
-latchEnAsClk \
off -reglist off -separateRegToMultiClkDmn off \
-traceMultiResolveSource on
```

schFindClockSource

Description

Finds the clock source from a selected instance or signal with the user-specified constant setting.

Syntax

```
schFindClockSource [-win window] -signal signalName [-constant
signalName <const value>...] [-constCellPort cellName
<const value>...][-newSchema]

schFindClockSource [-win window] -inst instName [-constant
signalName <const_value>...] [-constCellPort cellName
<const value>...][-newSchema]

schFindClockSource [-win window] [-loadFile filename
[-checkValidation]]

schFindClockSource [-win window] [-saveFile filename]
```

Argument

`-win` *`window`*

Specifies the window id of the invoking *nSchema* window.

`-signal` *`signalName`*

Finds the clock source from the specified signal.

`-inst` *`instName`*

Finds the clock source from the specified instance.

`-constant signalName <const_value>`

Specifies a constant signal with an assigned value to use for finding the clock source. The format of `const_value` is a radix type Decimal, Hex, Octal, or Binary with a number or "Don't care". This argument is the same as `-constant` in the `schExtractClockDomain` command.

`-constCellPort` *`cellName`* `<const value>`

Specifies a constant cell port with an assigned value to use for finding the clock source. The format of the `const` value is a radix type Decimal, Hex, Octal, or Binary with a number or "Don't care". This argument is the same as `-constCellPort` in the `schExtractClockDomain` command.

`-loadFile filename [-checkValidation]`

Loads the settings for a constant signal and a constant cell port from the `*.cs` file. If the `-checkValidation` argument is set, the setting file is checked for errors first.

`-saveFile` *`filename`*

Saves the current constant signal and constant cell port setting to a file.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schFindClockSource -win $_nSchema2 -signal "test.b" -constant
"test.eco_n1" Binary 0 \
           -constant "test.eco_n2" Hex 1 -constant "test.c"
dontcare -newSchema
schFindClockSource -win $_nSchema2 -loadFile "./signal.cs"
-checkValidation
```

schHilightClockDomain

Description

Saves/Loads the highlighted clock domains to/from a user-specified file.

Syntax

```
schHilightClockDomain -win window_id
schHilightClockDomain -win window_id -set domain_number color
schHilightClockDomain -win window_id  -saveFile xxxx | -loadFile
xxx
schHilightClockDomain -win $_nSchema2  -syncColor [on/off]
schHilightClockDomain -win $_nSchema2  -regColor [on/off]
```

Arguments

`-win`

Specifies the window id.

`-set`

Sets color to the domain whose number of domain is `domain_number`.

`-saveFile`

Specifies the path and file name to save.

`-loadFile`

Specifies the path and file name to load.

`-syncColor`

If the value is *on*, the button is toggled *on*; if the value is *off*, the button is toggled *off*.

`-regColor`

If the value is on, the button is toggled on; if the value is off, the button is toggled off.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schExtractClockDomain -win $_nSchema2
schHilightClockDomain -win $_nSchema2
schHilightClockDomain -win $_nSchema2 -set 1 ID_GRAY4 2 ID_RED5 3
ID_ORANGE5 4 ID_YELLOW5 5 ID_GREEN5 6 ID_CYAN5 7 ID_BLUE5 8
ID_PURPLE5 9 ID_GRAY4 10 ID_RED5
schHilightClockDomain -win $_nSchema2 -loadFile/proj/home/files/
forLoad1.sc
schHilightClockDomain -win $_nSchema2 -saveFile forSave1.sc.

1. schHilightClockDomain -win $_nSchema3 -syncColor on
schHilightClockDomain -win $_nSchema3 -synchronizerColor
ID_PURPLE6
2. schHilightClockDomain -win $_nSchema3 -syncColor off
```

schHighLightClockTree

Description

Highlights the clock net from the selected instance, signal, and instance port.

Syntax

```
schHighLightClockTree [-win window]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nSchema* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schHighLightClockTree -win $schWindow
```

schUnHighLightClockTree

Description

Unhighlights the specified clock source net.

Syntax

```
schUnHighLightClockTree [-win window] -clockSource clockSource
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nSchema* window.

`-clockSource` *`clockSource`*

Specifies the clock source to be unhighlighted.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schUnHighLightClockTree -clockSource system.clock
```

schImportClockCTSSetting

Description

Imports all local CTS settings and global CTS settings to a file in the Astro, Apollo, FirstEncounter, or ICC (IC Compiler) format.

Syntax

```
schImportClockCTSSetting [-importCTSFile fileName] -append
[-design topDesign] [-delim delimiter] [-vendor vendorType]
```

Argument

`-importCTSFile fileName`

Specifies the file name to import.

`-append`

Appends a new constraint in a file, instead of replacing it.

`-design topDesign`

Specifies the name of the top design.

`-delim delimiter`

Specifies the delimiter.

`-vendor vendorType`

Specifies the vendor's format, "Astro 2005", "Astro 2007", "Apollo", "FirstEncounter", or "ICC" to import.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schImportClockCTSSetting -importCTSFile "demo5.icc" \
-append -design "top" -delim "/" -vendor "ICC"
```

schLoadClockDB

Description

Loads the clock domains from a `.cdb` (Clock Database) file.

Syntax

```
schLoadClockDB [-win window] -file filename
```

Arguments

`-win window`

Specifies the window id of the invoking *nSchema* window.

`-file filename`

Specifies file name to be loaded from the clock database.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schLoadClockDB CPU.cdb
```

schLoadClockSetting

Description

Loads the clock setting to the **CTS** tab of the *Clock Tree Setting* form.

Syntax

```
schLoadClockSetting -loadCTSAtt "xxx.cts" -append
```

Arguments

`-loadCTSAtt "xxx.cts"`

Loads the CTS settings from the `xxx.cts` file to the **CTS** tab of the *Clock Tree Setting* form.

`-append`

Appends the CTS settings from the new file to the original CTS settings on the **CTS** tab of the *Clock Tree Setting* form. Without this argument, the

original CTS settings are cleaned out and reset. With this argument, the original CTS settings are kept and the new settings are added.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schLoadClockSetting -loadCTSAtt "1.cts" -append
```

NOTE: In the Verdi 6.1 version, the `schLoadClockSetting` command replaces the `-loadCTSAtt` option of the `schClockTreeBrowser` command.

schSaveClockDB

Description

Saves the clock domains result as a `.cdb` (Clock Database) file.

Syntax

```
schSaveClockDB [-win window] -file filename
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nSchema* window.

`-file` *`filename`*

Specifies file name to be saved to the clock database.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schSaveClockDB CPU.cdb
```

schSaveClockSDCSetting

Description

Exports the current constant settings in the *Clock Tree Setting* form to an SDC format file.

Syntax

```
schSaveClockSDCSetting [-saveSdcFile file] [-noClkSrcRefresh]
[-design designName] [-delim delim] [-signal clockSourceList]
[-constant SignalName constantValue] [-constCellPort cellName
constValue]
```

Argument

`-saveSdcFile` *`file`*

Specifies the file name to export.

`-noClkSrcRefresh`

If this attribute is set, settings in the current *Clock Source* table are not cleared. All current data for clock sources and generated clock sources is exported.

The remaining parameters, such as `-design`, `-delim`, `-signal`, `-constant`, and `-constCellPort` are described in `schExtractClockDomain`.

Value Returned

Returns 1 if successful, 0 otherwise

Example

```
schSaveClockSDCSetting -saveSdcFile "test.sdc" \
-design "system.i_cpu" -delim "/"
-signal "system.i_cpu.clock" "system.i_cpu.R_W"
"system.i_cpu.i_CCU.C6_reg.Q" \
-constant "system.i_cpu.reset" Binary 0 \
-constant "system.i_cpu.addr\[777:999\]" Binary 1 \
-constant "system.i_cpu.i_CCU.C6" dontcare \
-constCellPort "MUX21H.A" dontcare \
-constCellPort "AN25.A" dontcare
```

schSaveClockSetting

Description

Saves the clock settings on the CTS tab of the *Clock Tree Setting* form.

Syntax

```
schSaveClockSetting -saveCTSAtt "xxx.cts"
```

Arguments

`-saveCTSAtt "xxx.cts"`

Saves the current CTS setting into a file.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schSaveClockSetting -saveCTSAtt "1.cts"
```

NOTE: In the Verdi 6.1 version, the **schSaveClockSetting** command replaces the **-saveCTSAtt** option of the **schClockTreeBrowser** command.

schSaveClockTree

Description

Saves clock tree information to specified output file.

Syntax

```
schSaveClockTree [-win window] -file filename
```

Arguments

`-win window`

Specifies the window id of the invoking *nSchema* window.

`-file filename`

Specifies file name the clock tree information is saved to.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schExpschSaveClockTree clock_tree.out
```

schSetFalsePath

Description

Sets the false path from the *Crossing Paths* window or the *Timing Report* form.

Syntax

```
schSetFalsePath [-win window] [[-addPath|-delPath] "type" "from"
"to"]
schSetFalsePath [-win window] [-exportSDC filename] [-design
"design"] [-delim delimiter] [-exportFilterSetting "xxx.flt"]
```

Argument

`-addPath|-delPath "type" "from" "to"`

Specifies a false path to add or delete. The false path type is *Clock, Cell* (instance) or *InstPin* (instant pin). "`from`" refers to the end point of the path. "`to`" refers to the start point of the path.

`-exportFilterSetting "xxx.flt"`

Saves filter settings for the *Check Crossing Paths* form to a file that is loaded in the *Filter* form opened by the **Tools -> Filter** command in the *Crossing Paths* window.

`-exportSDC filename`

Specifies the SDC filename to be exported. "`-design`" and "`-delim`" parameters have the same export SDC functionality in the `schExtractClockDomain` and the `schSaveClockSDCSetting` commands.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schSetFalsePath -win $_nSchema2 -addPath "InstPin"
"system.i_cpu.i_CCU.\\bus_mode_reg\[2\] .CP"
"system.i_cpu.i_ALUB.\\ACC_reg\[7\] .D"

schSetFalsePath -win $_nSchema2 -delPath "Clock" "system.clock"
"system.i_cpu.i_CCU.C6"

schSetFalsePath -win $_nSchema2 -exportSDC "test.sdc" -design
"system" -delim "/"
```

schShowClockDomain

Description

Lists all currently loaded clock domains.

Syntax

```
schShowClockDomain [-win window] [-domainId domain id list | all]
[-src clockSource] -showOn shortText|text|schema]
[-dumpShortText filename]
```

Arguments

`-win` *`window`*

Specifies the window id of the invoking *nSchema* window.

`-domainId` *`domain id list`* `|` `all`

Lists the clock domain ID.

`-src clockSource`

Lists all clock domains from this source.

`-showOn shortText|text|schema`

The `shortText` option shows the clock domain summary in the `File Viewer` as a short list. The `text` option shows a detailed instance list for the extracted clocks in `File Viewer` as a long list. The `nSchema` option shows the clocks in the *nSchema* window.

`-dumpShortText` *`filename`*

This option saves the clock domain summary to the specified file. The saved file is always as a short list.

`-driver`

Shows the instance pin.

`-source`

Shows the clock source.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schShowClockDomain -win $_nSchema2 -domainId 1 -src
"system.i_cpu.i_ALUB.clock" -showOn text -dumpShortText dump.log

schShowClockDomain -win $_nSchema2 -driver | -source

schShowClockDomain -domainNo -src "BBB.clkclk->BBB.clka"
"BBB.clkclk->BBB.clkb" -showOn clkTreeBrowser
```

The value after the `-src` attribute means a clock source is driven by clock signals and it is expressed with `->`. On the left side of `->` is the clock source and on the right side of `->` are the signals driving the clock source. In the example above, both the resolved clock signals, `BBB.clka` and `BBB.clkb`, drive the clock source, `BBB.clkclk`.

schShowCrossingPath

Description

Lists the crossing paths between domains. Only one from/to clock domain pair is supported, but you may specify multiple register pairs for the domains. The results are shown in an *nSchema* window as a schematic view or in the file viewer as text.

Syntax

```
schShowCrossingPath [-win window] [-showOn schema|text|shortText]
[-fromId oneDomainId -toId oneDomainId ] [[-includeMid]
[-dumpFile filename] [-rowNum rowNo] [-check|-uncheck -regPair
FROMregisterName->TOregisterName | all] [-lastOnly on|off ]
[-loadFile filename] [-saveFile filename] [-sortCrossPath
columnName] [-sortCDCCorrelation columnName] [-switchView Mode]
```

Arguments

`-win` *`window`*

Specifies the window id.

`-showOn schema|text|shortText`

Specifies where to display the crossing path information.

The `schema` option shows the results in the *nSchema* window.

The `text` option shows the results in the file viewer as the long text mode from the GUI. It displays a detailed instance list with register pairs and instances in between register pairs which includes the 'from' register and the ‘to register’.

The `shortText` option shows the results in the file viewer as the short text mode from the GUI. It only displays the instance list with register pairs.

`-fromId` *`oneDomainId`*

Specifies the start clock domain ID of the selected clock domain. If `-regPair` is not specified, the crossing path is used to set check/uncheck; otherwise, only the register pair is used to set check/uncheck status.

`-toId` *`oneDomainId`*

Specifies the end clock domain ID of the selected clock domain. If `-regPair` is not specified, the crossing path is used to set check/uncheck; otherwise, only the register pair is used to set check/uncheck status.

`-rowNum`

Specifies the row number of the selected clock domain in the *Crossing Path* table.

`-regPair`

Specifies the 'from register' name, the 'to register' name, and the row number of the selected register pair.

`-check/-uncheck`

Sets the crossing path or register pair as checked or unchecked.

`-regPair` *`FROMregisterName->TOregisterName`* `| all`

The name of the FROM register, "->", and the name of the TO register. It can be multiple arguments. `all` means all register pairs of one crossing path.

`-viewMode` *`ModeName`*

Specifies whether to show in fan-in view or fan-out view with the `ModeName` as "FanIn" or "FanOut".

`-includeMid`

Dumps instance information in between register pairs.

`-dumpFile` *`filename`*

Specifies the file name into which the information is dumped.

NOTE: Only `-dumpFile` supports more than one from/to clock domain pair.

`-lastOnly on|off`

When *on*, *nAnalyzer* shows the crossing path list without its previous result. When *off*, *nAnalyzer* appends the new results to the existing crossing path list. The default is *off*.

`-loadFile` *`filename`*

Specifies the file to be loaded.

`-saveFile` *`filename`*

Specifies the file to be saved.

`-sortCrossPath`

Specifies the column in the *Crossing Paths* table and the data is sorted according to the specified column.

`-setCDCCorrelationColumn`

Specifies the columns in the *CDC Correlation Details* table. The columns include *Problem Type, Instance/Signal Name,* and *Path Number*.

`-sortCDCCorrelation` *`columnName`*

Specifies the column in the *CDC Correlation Details* table and the data is sorted according to the specified column.

`-switchView` *`Mode`*

Specifies the viewing mode for the table in the *Register Pair Details* section. The mode options include *Detail View* (specified with `Detail_View`), *Fan-in View* (specified with `Fanin_View`), and *Fan-out View* (specified with `Fanout_View`).

Value Returned

Returns 1 if successful; otherwise, returns 0.

Examples

```
schShowCrossingPath -win $_nSchema2 -showOn schema -fromId 2
-toId 10 -regPair \
"system.i_pram.pram:Always1#Always0:38:43:ComboMemory->system.i_c
pu.i_PCU.\\TDB_reg\[7\] " \
"system.i_pram.pram:Always1#Always0:38:43:ComboMemory->system.i_c
pu.i_PCU.\\TDB_reg\[3\] " \
"system.i_pram.pram:Always1#Always0:38:43:ComboMemory->system.i_c
pu.i_PCU.\\TDB_reg\[1\] "
```

The above example displays three register pair paths of *domain 2* to *domain 10*.

```
schShowCrossingPath -win $_nSchema2 -showOn text -fromId 2 -toId
10 -regPair \
"system.i_pram.pram:Always1#Always0:38:43:ComboMemory->system.i_c
pu.i_PCU.\\TDB_reg\[7\] " \
"system.i_pram.pram:Always1#Always0:38:43:ComboMemory->system.i_c
pu.i_PCU.\\TDB_reg\[3\] " \
"system.i_pram.pram:Always1#Always0:38:43:ComboMemory->system.i_c
pu.i_PCU.\\TDB_reg\[1\] "
```

This example displays the register pairs of domain 2 to domain 10 on file viewer by the long text mode.

```
schShowCrossingPath -win $_nSchema2 -fromId 1 2 3 -toId 4 5
-dumpFile "/a/file1.txt"
```

This example finds and dumps all register pairs of "domain 1, 2, 3 to domain 4, 5" to a file, `/a/file1.txt`. The results are not shown on either *nSchema* or the *File Viewer* window.

```
schShowCrossingPath -win $_nSchema2 -fromId 1 2 3 -toId 4 5
-includeMid -dumpFile "/a/file1.txt"
```

The example finds and dumps all register pairs of "domain 1, 2, 3 to domain 4, 5" to a file, `/a/file1.txt`. The register pair information include the instances in between the 'from' register and 'to' register. The results are not shown on *nSchema* or the *File Viewer*.

```
schCrossingPath -win $_nSchema3 -fromId all -toId all
-synchronizer Temp1 Temp2
schShowCrossingPath -win $_nSchema2 -fromId 1 -toId 2 -regPair
all
```

To set the synchronizer in Tcl, you must run `schCrossingPath` to extract crossing path first, and then use `schShowCrossingPath` to dump the file.

```
schShowCrossingPath -fromId 1 -toId 2 -rowNum 0 -check

schShowCrossingPath -fromId 1 -toId 2 -rowNum 0 -regPair \
"system.i_cpu.i_ALUB.\\ACC_tmp_reg\[0\]
->system.i_cpu.i_ALUB.\\IXR_reg\[7\] ->1" -uncheck

schShowCrossingPath -sortCrossPath "Convergnece"/"Divergence"/
"Re-convergence" -sortCDCCorrelation "Problem Type"/"Instance/
Signal Name"/"Path Number"

schShowCrossingPath -showOn schema/shortText/text -fromId 1 -toId
6 -rowNum 3 -CDCCorrelation "C:usb_device.\\core/epctl/ep0_ctl/
U1761 :2" -regPair "system.i_cpu.i_ALUB.\\ACC_tmp_reg\[0\]
->system.i_cpu.i_PCU.\\PC_reg\[6\] ->1"
```

```
"system.i_cpu.i_ALUB.\\ACC_tmp_reg\[0\]
->system.i_cpu.i_PCU.\\PC_reg\[5\] ->2"

schShowCrossingPath -showOn shortText -fromDomainNo \
17.r -toDomainNo 16.r -rowNum 4 -CDCCorrelation \
"Convergence|usb_device.\\core/epctl/ep0_ctl/U2687 |2"

schShowCrossingPath -switchView "Detail_View"
schShowCrossingPath -switchView "Fanin_View"
schShowCrossingPath -switchView "Fanout_View"

schShowCrossingPath -showOn schema -fromDomainNo \
1.f -toDomainNo 2.f -rowNum 2 -switchView FanOut -regPair 2
```

srcGetCurrentWindow

Description

Returns the window id for *File Viewer* windows which are opened by invoking the **Tools -> Clock Domain** related commands. Note that this Tcl command does not apply to *File Viewer* windows invoked by clicking the **New FileViewer** toolbar icon on the *nTrace* toolbar.

Syntax

```
-srcGetCurrentWindow
```

Value Returned

Return the window id if the current window is clock domain related. Otherwise, return 0.

Example

```
srcGetCurrentWindow
```

Prime Time

schPrimeTime

Description

Sets Prime Time Delay Precision and shows timing path(s) information.

Syntax

```
schPrimeTime [-win window] [-file filePath] [-startPointClkDomain|
-endPointClkDomain][-delayPrecision delayPrecisionValue] [-order
pathLineNo] [-showOn schema | text  [-skewSummaryReport -delimiter
delimiter]]
```

Argument

`-win` *`window`*

Specifies the window id.

`-file` *`filePath`*

Specifies the full file name to open a timing report file (`*.xml` extension). The XML file is created from a variety of tool outputs using the related conversion utility, including *pt2Xml.pl, AMBIT.pl, astrol2Xml.pl, DC.pl, EinsTimer.pl, HAL2Xml.pl, magma.pl, pearl.pl* and *RTLC2Xml.pl*.

`-startPointClkDomain`

Enables the **Clock Domain of Start Point** option in the *Timing Report* form.

`-endPointClkDomain`

Enables the **Clock Domain of End Point** option in the *Timing Report* form.

`-delayPrecision`

Specifies value of the **Delay Precision** option as 0.1, 0.01(default), 0.001, and 0.0001.

`-order`

Indicates the row number of the selected path in the main display area of the *Timing Report* form.

`-showOn text | schema`

Shows the timing paths in the *Text Viewer* or *nSchema* window.

`-skewSummaryReport`

Shows the skew summary in the *Text Viewer* window.

`-delimiter`

Specifies the delimiter to use to separate the hierarchical names of an exported full hierarchical name.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
schPrimeTime -delayPrecision "0.01"
schPrimeTime -win $_nSchema1 -max -startPointClkDomain
schPrimeTime -win $_nSchema1 -max -endPointClkDomain
schPrimeTime -win $_nSchema1 -order 4 -showOn text
-skewSummaryReport -delimiter \ "/"

schPrimeTime -win $nSchema_1 -file "~/Astro.txt"

schPrimeTime -win $nSchema_1 -file "~/RTLCompiler.xml"
```

Switching Analysis

saSaveReport

Description

Dumps the Switching Analysis report to a file.

Syntax

```
saSaveReport -report_id window [-filename outputFile] [-dump_csv]
```

Argument

`-dump_csv`

When specified, output a file in the CSV format or in the XML format for external storage. The default format is XML.

`-filename {outputFile}`

Specifies the output file name. The default file name is *default.report*.

`-report_id window`

Specifies the window id of the *Switching Analysis Report* frame.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
saSaveReport -report_id 7
```
Dumps the query result of the *Switching Analysis* frame with number identification 7 to the default file in the XML file format.

```
saSaveReport -report_id 7 -filename {sa_report.xml}
```
Dumps the query result of the *Switching Analysis* frame with number identification 7 to the specified file in the XML file format.

```
saSaveReport -dump_csv -report_id 7 -filename {sa_report.csv}
```
Dumps the query result of the *Switching Analysis* frame with number identification 7 to the specified file in the CSV file format.

Property Tools

abvGetAssertionInfo

NOTE: There is no equivalent GUI command.

Description

Reports the status and the begin and end times of the evaluated assertion.

Syntax

```
abvGetAssertionInfo -name assertName [-success] [-failure]
[-incomplete]
```

Arguments

`-name` *`assertName`*

Specifies the assertion name.

`-success`

Provides threads with status success as output.

`-failure`

Provides threads with status failure as output.

`-incomplete`

Provides threads with status incomplete as output.

Value Returned

Returns the status and the begin time and end time.

Example

```
abvGetAssertionInfo -name top.sb.ss.a2 -success -incomplete
Output:
success 0 100
success 50 150
success 70 170
incomplete 200 250
```

abvGetAssertionNames

NOTE: There is no equivalent GUI command.

Description

Extracts all SVA assertions from the Verdi Knowledge Database (KDB).

Syntax

```
abvGetAssertionNames [-assert] [-assume] [-cover] [-scope scope]
```

Arguments

`-assert`

Provides assertions of type assert as output.

`-assume`

Provides assertions of type assume as output.

`-cover`

Provides assertions of type cover as output.

`-scope` *`scope`*

Provides all assertions in all levels under the specified scope as output.

Value Returned

Returns a list of assertion names (with the full hierarchy).

Example 1

```
abvGetAssertionNames -assert -assume
```

Output:

```
top.sa.ss.a1 assert
top.sa.ss.am1 assume
top.sb.ss.a1 assert
top.sb.ss.a2 assert
top.sb.ss.am1 assume
top.sb.ss.am2 assume
```

Example 2

```
abvGetAssertionNames -scope top.sa -assert
```

Output:

```
top.sa.ss.a1 assert
```

abvGetSigValueByTime

NOTE: There is no equivalent GUI command.

Description

Returns signal values from the given signal name and the given cursor time or the user-specified time as an argument.

Syntax

```
abvGetSigValueByTime -name signalName [-time time] [-single]
```

Arguments

`-name` *`signalName`*

Specifies the signal name.

`-time` *`value`*

Specifies the cursor time.

`-single`

When specified, the output value expression is not separated by a comma or space and all bits are filled.

Value Returned

Returns the signal value (the format of value is binary).

Example 1

```
top.sb.ss.sigA
(xxxxxxxx(0s)'00001111(10s)'11110000(20s)'11111111(30s))
Current Time: 10s
Input 1: abvGetSigValueByTime -name top.sb.ss.sigA
Output 1: 00001111
Input 2: abvGetSigValueByTime -name -top.sb.ss.sigA -time 0
Output 2: xxxxxxxx
Input 3: abvGetSigValueByTime -name -top.sb.ss.sigA -time -1
Output 3: 11111111
```

Example 2

```
abvGetSigValueByTime -name {fifo.mem} -time 0
Output: {0....0, 0....0,... }

abvGetSigValueByTime -single -name {fifo.mem} -time 0
Output: 00000000000000000000000000000000000000000000000000000000000000000000
```

AssAddToDtl

Description

Adds selected property nodes in the *Evaluator* pane to the *Property Details* table.

Syntax

```
AssAddToDtl -assAddSignal {propName1} {propName2}
```

Arguments

`-assAddSignal`

Adds signals to the *Property Details* table.

Value Returned

None.

Example

```
AssAddToDtl -assAddSignal {system.\i_cpu_
{s1_0s4l,s2l[3:0]} .i_ALUB.genblk[0].cv}
```

assCtrlMsg

Description

Controls the error or warning message in the *Property Tools* window.

Syntax

```
assCtrlMsg -enable messageId
assCtrlMsg -disable messageId
```

Arguments

`-disable` *`messageId`*

Disables the specified error or warning message.

`-enable` *`messageId`*

Enables the specified error or warning message.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
assCtrlMsg -enable IDM_ASS_W_SRC_REFRESH_FOR_FSDB_CHANGE
```

AssDebWin

Description

Sets the signal value radix and the notation format in the *Analyzer* pane of the *Property Tools* window.

Syntax

```
AssDebWin [-setRadix bin|dec|hex|oct|ascii|enum] [-setNotation
unsigned|2Com|1Com|mag]
```

Arguments

`-setRadix`

Sets the selected signal value radix in the Binary, Decimal, Hexadecimal, Octal, ASCII, or Enumerated Literal format.

`-setNotation`

Sets the selected signal value notation in the Unsigned, Signed 2's Complement, Signed 1's Complement, or Signed Magnitude format.

Example

```
AssDebWin -setRadix hex
AssDebWin -setNotation 2Com
```

AssDeleteFsdb

Description

Removes the selected FSDB file from the *FSDB Statistics* table.

Syntax

```
AssDeleteFsdb {file_name}
```

Example

```
AssDeleteFsdb {/verify1/regression/SVA/AssertTools/DynamicDump/
F7-unbounded_checker.fsdb.vf}
```

AssEvalChosen

Description

Adds or deletes property signals.

Syntax

```
AssEvalChosen [-add {{signal_name}} | -del {{signal_name}} | -
Addall | -scope {{scope}}]
```

Arguments

`-add {{signal_name}}`

Adds specified property signals to the evaluator list.

`-del {{signal_name}}`

Deletes specified property signals from the evaluator list.

`-Addall`

Adds all property signals to the evaluator list.

`-scope {{scope}}`

Adds all property signals in the specified scope to the evaluator list.

Value Returned

Returns 1 if successful, otherwise, returns 0.

Example

```
AssEvalChosen -add {{system.i_cpu.i_ALUB.ALU_ZERO}}
AssEvalChosen -del {{system.i_cpu.i_ALUB.ALU_ZERO}}
AssEvalChosen -Addall
AssEvalChosen -scope {{system.i_cpu.i_ALUB}}
```

AssertOpenStatistic

Description

Opens the *Property Tools* window.

Syntax

```
AssertOpenStatistic
```

Example

```
AssertOpenStatistic
```

AssertToolsClose

Description

Closes the *Property Tools* window.

Syntax

```
AssertToolsClose
```

Example

```
AssertToolsClose
```

AssertStatistics

Description

Uses the `AssertStatistics` command-line option for the **Results** tab in the *Property Tools* window.

Syntax

```
AssertStatistics -fsdbStatistics [-selectCell
fileNameRowNameColumnName] [-addSignal] [-setActiveFsdb]

AssertStatistics -propDetails [-selectRow fileNameSignalName
(start_time end_time status)] [-delete] [-deleteAll]
[-activeTrace] [-expand fileNameSignalName] [-collapse
fileNameSignalName] [-addSignal -drop signal_name|-nwave]
-propDetails [-sortProperty -asc | -des]
[-sortScope -asc | -des][-sortFailure -asc | -des]
[-sortSuccess -asc | -des][-sortIncomplete -asc | -des]
[-sortStartTime -asc | -des][-sortEndTime -asc | -des]
[-sortStatus -asc | -des][-sortFsdb -asc | -des]
[-sortType -asc | -des]

AssertStatistics [-specifyTimeRange on|off]
[-specifyByStartTime|-specifyByEndTime fromTimeToTime]
[-ignoreCase on|off] [-useRegExp on|off]
[-filterProperty filterTxt] [-filterFsdb filterTxt]
[-filterScope filterTxt] [-filterFailure on|off]
[-filterSuccess on|off] [-filterIncomplete on|off]
[-filterNoAttempt on|off] [-filterNoChecked on|off]
[-filterAssert on|off] [-filterBoolean on|off]
[-filterCover on|off] [-filterEvent on|off] [-filterForbid on|off]
[-sortStartTime on|off] [-sortEndTime on|off] [-sortStatus on|off]
[-displayModule on|off] [-displayScope on|off]
[-displayFailure on|off] [-displaySuccess on|off]
[-displayIncomplete on|off] [-displayStartTime on|off]
[-displayEndTime on|off] [-displayStatus on|off]
[-displayFsdb on|off] [-displayType on|off]
[-syncCursorTime on|off] [-addSelectedProp on|off]
```

Arguments

`-addSelectedProp on|off`

Specifies on to add the selected property to the *nWave* window.

`-displayEndTime on|off`

Specifies on to display the end time field in *Property Details*.

`-displayFailure on|off`
Specifies on to display the failure field in *Property Details*.

`-displayFsdb on|off`
Specifies on to display the FSDB field in *Property Details*.

`-displayIncomplete on|off`
Specifies on to display the incomplete field in *Property Details*.

`-displayModule on|off`
Specifies on to display the module field in *Property Details*.

`-displayScope on|off`
Specifies on to display the scope field in *Property Details*.

`-displayStartTime on|off`
Specifies on to display the start time field in *Property Details*.

`-displayStatus on|off`
Specifies on to display the status field in *Property Details*.

`-displaySuccess on|off`
Specifies on to display the success field in *Property Details*.

`-displayType on|off`
Specifies on to display the type field in *Property Details*.

`-filterAssert on|off`
Specifies on to show the assert type property.

`-filterAssume on|off`
Specifies on to show the assume type property.

`-filterBoolean on|off`
Specifies on to show the boolean type property.

`-filterCover on|off`
Specifies on to show the cover type property.

`-filterEvent on|off`
Specifies on to show the event type property.

`-filterFailure on|off`
Specifies on to show the failure threads of property.

`-filterForbid on|off`
Specifies on to show the forbid type property.

`-filterFsdb` *`filterTxt`*

Specifies `on` to filter the Property Details list based on the FSDB file name.

`-filterIncomplete on|off`

Specifies `on` to show the incomplete threads of property.

`-filterNoAttempt on|off`

Specifies `on` to show the no attempt status property.

`-filterNoChecked on|off`

Specifies `on` to show the no checked status property.

`-filterProperty` *`filterTxt`*

Specifies to filter the Property Details list by the property name.

`-filterScope` *`filterTxt`*

Specifies to filter the Property Details list by the scope.

`-filterSuccess on|off`

Specifies `on` to show the success threads of property.

`-fsdbStatistics -addSignal`

Specifies to add the selected assertion signal in the *FSDB Statistics* table to the *Property Details* table.

`-fsdbStatistics -selectCell` *`fileNameRowNameColumnName`*

Specifies to select the cell in the *FSDB Statistics* table.

`-fsdbStatistics -setActiveFsdb`

Specifies to set active FSDB for the selected file.

`-ignoreCase on|off`

When this option is `on`, the case is ignored (that is, ‘a’ finds ‘a’ and ‘A’) in all the filter fields. When this option is `off`, the case is considered (that is, ‘b’ finds ‘b’ only) in all the filter fields.

`-propDetails -activeTrace`

Specifies to switch to the **Analyzer** tab of the *Property Tools* window and proceeds with the assertion debug for an assertion failure.

`-propDetails -addSignal -drop signal_name|-nwave`

Specifies to add the signal dragged from *nWave* or to add selected properties in *nWave* to the *Property Details* table.

`-propDetails -collapse` *`fileName signalName`*

Specifies to collect the assertion folder.

`-propDetails -delete`

Specifies to delete the selected row in the *Property Details* table.

`-propDetails -deleteAll`

Specifies to remove all results in the *Property Details* table.

`-propDetails -expand` *`fileNameSignalName`*

Specifies to expand the assertion folder.

`-propDetails -selectRow` *`fileNameSignalName`* `(start_time end_time status)`

Specifies to select the specified property row with the file name and signal name, or to select the specified thread row of the property with extra start time, end time, and status.

`-propDetails -sortEndTime -asc | -des`

Specifies to sort the end time column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortFailure -asc | -des`

Specifies to sort the failure column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortFsdb -asc | -des`

Specifies to sort the FSDB column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortIncomplete -asc | -des`

Specifies to sort the complete column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortProperty -asc | -des`

Specifies to sort the property column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortScope -asc | -des`

Specifies to sort the scope column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortStartTime -asc | -des`

Specifies to sort the start time column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortStatus -asc | -des`

Specifies to sort the status column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortSuccess -asc | -des`

Specifies to sort the success column in the ascending or descending order in the *Property Details* table.

`-propDetails -sortType -asc | -des`

Specifies to sort the type column in the ascending or descending order in the *Property Details* table.

`-sortEndTime on|off`

Specifies on to sort threads by the end time.

`-sortStartTime on|off`

Specifies on to sort threads by the start time.

`-sortStatus on|off`

Specifies on to sort threads by Status.

`-specifyByStartTime|-specifyByEndTime` *`fromTimeToTime`*

Specifies to filter property attempts whose start or end time is within the specified time range.

`-specifyTimeRange on|off`

Specifies the time range to filter property attempts.

`-syncCursorTime on|off`

Specifies on to synchronize cursor time with the selected property.

`-useRegExp on|off`

Specifies `on` to support additional regular expressions.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
AssertStatistics -fsdbStatistics -addSignal
AssertStatistics -propDetails -deleteAll
AssertStatistics -options -filterSuccess on
```

AssFindAssert

Description

Finds the specified assert in the **Evaluator**, **Property Statistics**, or **Property Details** table.

Syntax

```
AssFindAssert -assertName {prop_name} -findIn
[evaluator|propStat|propDtl] [-next|-prev]
```

Arguments

`-assertName {prop_name}`

Specifies the assert name.

`-findIn [evaluator|propStat|propDtl] [-next|-prev]`

Specifies the area and direction to search for the assert.

Example

```
AssDeleteFsdb {/verify1/regression/SVA/AssertTools/DynamicDump/
F7-unbounded_checker.fsdb.vf}
```

AssSetActiveFsdb

Description

Sets the selected FSDB file as the current active file.

Syntax

```
AssSetActiveFsdb file_name
```

Example

```
AssSetActiveFsdb /verify1/regression/SVA/AssertTools/DynamicDump/
F7-unbounded_checker.fsdb.vf
```

propDtlSelectAll

Description

Selects all property rows in the **Property Details** table.

Syntax

```
propDtlSelectAll
```

Example

```
propDtlSelectAll
```

propRstSaveText

Description

Saves property results to a text file name

Syntax

```
propRstSaveText [-saveFsdbStat|-savePropStat|-savePropDtl]
file name
```

Arguments

`file_name`

Specifies the file name.

`-saveFsdbStat`

Saves the FSDB Statistics data to a file.

`-savePropState`

Saves the Property Statistics data to a file.

`-savePropDtl`

Saves the Property Details data to a file.

Example

```
propRstSaveText -saveFsdbStat -savePropDtl ./default.txt
```

propStatAddSignal

Description

Adds selected properties to the **Property Details** table.

Syntax

```
propStatAddSignal
```

Example

```
propStatAddSignal
```

propStatCloseCovRpt

Description

Closes the *Coverage Report* form.

Example

```
propStatCloseCovRpt
```

propStatCloseOptForm

Description

Closes the *Options* form.

Example

```
propStatCloseOptForm
```

propStatCollapseRow

Description

Collects the selected property in the **Property Statistics** table.

Syntax

```
propStatCollapseRow {scopeList}
```

Example

```
propStatCollapseRow {system.i_cpu.i_ALUB.ALU_ZERO}
```

propStatDrop

Description

Selects property rows in the **Property Statistics** table.

Syntax

```
propStatDrop {prop1_name} {prop2_name}
```

Example

```
propStatDrop {tt.ss.assert_p2}
```

propStatExpandRow

Description

Expands the selected property in the *Property Statistics* table.

Syntax

```
propStatExpandRow {scopeList}
```

Example

```
StatExpandRow {system.i_cpu.i_ALUB.ALU_ZERO}
```

propStatHideProp

Description

Hides the selected property rows in the *Property Statistics* table.

Syntax

```
propStatHideProp
```

Example

```
propStatHideProp
```

propStatOpenCovRpt

Description

Opens the *Coverage Report* form.

Example

```
propStatOpenCovRpt
propStatOpenOptForm
```

Description

Opens the *Options* form.

Example

```
propStatOpenOptForm
```

propStatSaveCovRpt

Description

Saves the coverage report to a file.

Syntax

```
propStatSaveCovRpt file_name
```

Arguments

`file_name`

Specifies the file name.

Example

```
propStatSaveCovRpt ./coverage.rpt
```

propStatSelectRow

Description

Selects a row with a specified file name in the *Property Statistics* table.

Syntax

```
propStatSelectRow [-all | (-file file_name) property_name]
```

Arguments

`-all`

Selects all rows in the *Property Statistics* table.

`(-file file_name) property_name`

Selects a row with a specified file name.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
propStatSelectRow -all
```

propStatSetOptions

Description

Configures Property Statistics settings.

Syntax

```
propStatSetOptions [-propStat_ignoreCase on|off]
[-propStat_useRegExp on|off] [-propStat_filterProperty
filter_text] [-propStat_filterFsdb filter_text]
[-propStat_filterScope filter_text] [-propStat_filterAssert
on|off] [-propStat_filterBoolean on|off] [-propStat_filterCover
on|off] [-propStat_filterEvent on|off] [-propStat_filterForbid
on|off] [-propStat_sort -propStat_sortFailure|
-propStat_sortSuccess|-propStat_sortIncomplete|
-propStat_sortNoAttempt|-propStat_sortNotChecked]
[-propStat_displayModule on|off] [-propStat_displayScope on|off]
[-propStat_displayFailure on|off] [-propStat_displaySuccess
on|off] [-propStat_displayIncomplete on|off]
```

```
[-propStat_displayNoAttempt on|off] [-propStat_displayNotChecked
on|off] [-propStat_displayType on|off]
```

Arguments

`-propStat_ignoreCase on|off`

When this option is *on*, the case is ignored (that is, 'a' finds 'a' and 'A') in all the filter fields. When this option is *off*, the case is considered (that is, 'b' finds 'b' only) in all the filter fields.

`-propStat_useRegExp on|off`

When this option is *on*, additional regular expressions are supported.

`-propStat_filterProperty` *`filter_text`*

Filters the Property Statistics list by the property name.

`-propStat_filterFsdb` *`filter_text`*

Filters the Property Statistics list based on the FSDB file name.

`-propStat_filterScope` *`filter_text`*

Filters the Property Statistics list based on the scope.

`-propStat_filterAssert on|off`

If specified, the assert property is filtered.

`-propStat_filterBoolean on|off`

If specified, the boolean property is filtered.

`-propStat_filterCover on|off`

If specified, the cover property is filtered.

`-propStat_filterEvent on|off`

If specified, the event property is filtered.

`-propStat_filterForbid on|off`

If specified, the forbid property is filtered.

```
-propStat_sort -propStat_sortFailure|-propStat_sortSuccess|
-propStat_sortIncomplete|-propStat_sortNoAttempt|
-propStat_sortNotChecked]
```

Sorts the Failure, Success, Incomplete, No Attempt, or Not Checked column in the *Property Statistics* table.

`-propStat_displayModule on|off`

Specifies whether to display the *Module* column in the *Property Statistics* table.

`-propStat_displayScope on|off`

Specifies whether to display the *Scope* column in the *Property Statistics* table.

`-propStat_displayFailure on|off`

Specifies whether to display the *Failure* column in the *Property Statistics* table.

`-propStat_displaySuccess on|off`

Specifies whether to display the *Success* column in the *Property Statistics* table.

`-propStat_displayIncomplete on|off`

Specifies whether to display the *Incomplete* column in the *Property Statistics* table.

`-propStat_displayNoAttempt on|off`

Specifies whether to display the *No Attempt* column in the *Property Statistics* table.

`-propStat_displayNotChecked on|off`

Specifies whether to display the *Not Checked* column in the *Property Statistics* table.

`-propStat_displayType on|off`

Specifies whether to display the *Type* column in the *Property Statistics* table.

Example

```
propStatSetOptions -propStat_filterAssert off
```

propStatSortCol

Description

Sorts columns in the *Property Statistics* table in the ascending or descending order.

Syntax

```
propStatSortCol [-sortProperty|-sortScope|-sortFailure|
-sortSuccess|-sortIncomplete|-sortNoAttempt|-sortNotchecked]
[-asc|-des]
```

Arguments

`-sortProperty`

Sorts the *Property/FSDB* column in the *Property Statistics* table.

`-sortScope`

Sorts the *Scope* column in the *Property Statistics* table.

`-sortFailure`

Sorts the *Failure* column in the *Property Statistics* table.

`-sortSuccess`

Sorts the *Success* column in the *Property Statistics* table.

`-sortIncomplete`

Sorts the *Incomplete* column in the *Property Statistics* table.

`-sortNoAttempt`

Sorts the *No Attempt* column in the *Property Statistics* table.

`-sortNotchecked`

Sorts the *Not Checked* column in the *Property Statistics* table.

`-asc`

Sorts the columns in the *Property Statistics* table in the ascending order.

`-des`

Sorts the columns in the *Property Statistics* table in the descending order.

Example

```
propStatSortCol -sortFailure -des
```

propStatUnhideAll

Description

Shows all hidden property rows in the *Property Statistics* table.

Syntax

```
propStatUnhideAll
```

Example

```
propStatUnhideAll
```

psmFilterAssertions

Description

Filters the properties by the specified values.

Syntax

```
psmFilterAssertions [-type assertion|cover|assume|*]
[-name propFullName] [-module moduleName] [-scope scopeName]
[-lang PSL|SVA|*] [-IsShowType 0|1] [-IsShowName 0|1]
[-IsShowModule 0|1] [IsShowScope 0|1] [-IsShowCateGory 0|1]
```

Arguments

`-IsShowCateGory 0|1`

Displays the category of the specified property. The default is *0*.

`-IsShowModule 0|1`

Displays the module name of the specified property. The default is *0*.

`-IsShowName 0|1`

Displays the name of the specified property. The default is *1*.

`IsShowScope 0|1`

Displays the scope name of the specified property. The default is *1*.

`-IsShowType 0|1`

Displays the type of the specified property. The default is *1*.

`-lang PSL|SVA|*`

Specifies the language that the property belongs to. The languages include PSL, SVA and "*". The default is "*" (all).

`-module moduleName`

Specifies the module that the property belongs to.

`-name propFullName`

Specifies the property name.

`-scope scopeName`

Specifies the scope that the property belongs to.

`-type assertion|cover|assume|*`

Specifies the type that the property belongs to. The types include assertion, cover, assume and "*". The default is "*" (all).

Example

```
psmFilterAssertions -type {*} -name {*} -module {*} -Scope {*}
-IsShowType 1 -IsShowName 1 -IsShowModule 1 -IsShowScope 1
-IsShowCateGory 1 -lang {PSL}
```

vdac

Description

Evaluates the enabled assertions.

Syntax

```
vdac designFsdbFile -o resultPropFsdbFile [-write 0|1|2]
[-bt time[s|ms|us|ns|ps|fs]] [-et time[s|ms|us|ns|ps|fs]]
[-no_success][-no_filter][-no_cov_fail_filter][-merge]
[-smallrange][-flag_race][-regression][-enable_sv_display
filename]
```

Arguments

`-bt time[s|ms|us|ns|ps|fs]`

Specifies the start time of evaluation. The time units are: s, ms, us, ns, ps, and fs. The default time unit is *ns*.

`-enable_sv_display` *`filename`*

Specifies to enable `$display` in the SystemVerilog assertion. When `filename` is specified, results are saved to this file. Otherwise, results are printed in the console.

`-et time[s|ms|us|ns|ps|fs]`

Specifies the end time of evaluation. The time units are: s, ms, us, ns, ps, and fs. The default time unit is *ns*.

`-flag_race`

Flags the potential race in sampling (PSL only).

`-merge`

Merges the result FSDB with the design FSDB.

`-no_cov_fail_filter`

Stores the cover failure.

`-no_filter`

Stores the trivial success.

`-no_success`

Stores only the failure (no success).

`-smallrange`

Uses the smaller time range of the resulted FSDB and the design FSDB.

`-write 0|1|2`

Specifies the type of data to be written. The options are 0, 1, and 2. When 0 is specified, all data is written. When 1 is specified, all data except local variable is written. When 2 is specified, only the assertion results are written. The default is *0*.

Example

```
vdac top.fsdb -o top_out.fsdb -write 0 -reset_filter
```

```
vdac -enable_sv_display /home1/demo/systemverilog/
evaluate_result_display.log
```

Testbench Browser

srcTBAddBrkPnt

Description

Add breakpoints on the selected source code line or add breakpoints by time in non-interactive debug mode.

Syntax

```
srcTBAddBrkPnt -win window [-line lineNum] [-file fileName]
[-absolute | -relative time]
```

Arguments

`-absolute | -relative time`

Specify the absolute or relative time.

`-file fileName`

Specify the full path name of the file.

`-line lineNum`

Specify the line number.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBAddBrkPnt -win $_nTrace2 -line 6 -file /verdi/temp/thread/
thread.sv
srcTBAddBrkPnt -win $_nTrace2 -relative 200
```

srcTBAddDataView

Description

Add additional Watch tabs under the *Interactive Sim* tab.

Syntax

```
srcTBAddDataView -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBAddDataView -win $_nTrace2
```

srcTBAddToWatchFromSrc

Description

Add selected variables from the *Source Code* pane to the *Watch* tab.

Syntax

```
srcTBAddToWatchFromSrc -win window -tab tabId
```

Arguments

`-tab tabId`

Specify the Watch tab ID.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBAddToWatchFromSrc -win $_nTrace1 -tab 1
```

srcTBBreakPointLoad

Description

Load breakpoints from a saved file.

Syntax

```
srcTBBreakPointLoad -win window -file filename
```

Arguments

`-file` *`filename`*

Specify the full path of the file.

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBBreakPointLoad -win $_nTrace2 -file /verdi/temp/thread/a.bp
```

srcTBBreakPointSave

Description

Save breakpoints into a specified file.

Syntax

```
srcTBBreakPointSave -win window -file fileName
```

Arguments

`-file` *`fileName`*

Specify the full path of the file.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBBreakPointSave -win $_nTrace2 -file /verdi/temp/thread/a.bp
```

srcTBBTreeSelect

Description

Select the tree node path in the *Declaration Hierarchy* pane of the *Testbench Browser* window.

Syntax

```
srcTBBTreeSelect -win window -path "pathName"
```

Arguments

`-path "pathName"`

Specify the path. Double quotes must be used to enclose the path.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBBTreeSelect -win $_nTrace2 -path "Foo::print"
```

srcTBClassViewAction

Description

Set the class tree node in the *Declaration Hierarchy* pane of the *Testbench Browser* window.

Syntax

```
srcTBClassViewAction -win window -path"pathName"
```

Arguments

`-path "pathName"`

Specify the path. Double quotes must be used to enclose the path.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBClassViewAction -win $_nTrace2 -path "Foo::print"
```

srcTBCloseForm

Description

Close the *Testbench Browser* window.

Syntax

```
srcTBCloseForm -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBCloseForm -win $_nTrace2
```

srcTBDeleteDataView

Description

Delete the selected *Watch* tab from the *Interactive Sim* tab.

Syntax

```
srcTBDeleteDataView -win window -tab tabId
```

Arguments

`-tab tabId`

Specify the Watch tab ID.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDeleteDataView -win $_nTrace2 -tab 1
```

srcTBDVAddTo

Description

Add the selected nodes to another or newly created *Watch* tab.

Syntax

```
srcTBDVAddTo -win window -tab tabID -toTab newTabID -root
RootNodeName -path FullPathNodeName -scope FullScopeName
```

Arguments

`-path FullPathNodeName`

The name from root to the changed node.

`-root RootNodeName`

The root node name.

`-scope` *`FullScopeName`*

The scope information of root node (if it is not local tab).

`-tab` *`tabID`*

Specify the current watch tab ID.

`-toTab` *`newTabID`*

Specify the newly created watch tab ID.

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVAddTo -win $_nTrace2 -tab 1 -toTab 2 -root "limit" -path
"limit" -scope "p.unnamed_\$thread_sv_30"
```

srcTBDVMarker

Description

Set or unset the markers of the selected tree node(s) in the *Watch/Local* tab.

Syntax

```
srcTBDVMarker -win window -set | -unsetAll -tab tabID -root
RootNodeName -path FullPathNodeName
```

Arguments

`-path` *`FullPathNodeName`*

The name from root to the changed node.

`-root` *`RootNodeName`*

The root node name.

`-set | -unsetAll`

If **-set** is specified, set the marker of the selected tree node(s) in the *Watch/Local* tab. If **-unsetAll** is specified, remove all markers from the *Watch/Local* tab.

`-tab tabID`

Specify the current watch tab ID.

`-toTab newTabID`

The newly created watch tab ID.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVMarker -win $_nTrace2 -set -tab 1 -root "bar" -path "bar"
```

srcTBDVSelect

Description

Select tree nodes in the *Watch/Local* tab.

Syntax

```
srcTBDVSelect -win window -tab tabId -item {node1Index}
{node2Index}...
```

Arguments

`-item {node1Index} {node2Index}`

The index of the selected tree.

`-tab tabId`

Specify the *Watch/Local* tab ID.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVSelect -win $_nTrace1 -tab 0 -item {4 0 14}
```

srcTBFindScope

Description

Find the declarations in the *Testbench Browser* window.

Syntax

```
srcTBFindScope -win window -name "patternName" [-case]
-next | -prev | -all
```

Arguments

`-case`

Perform case-sensitive searching.

`-name "patternName"`

Specify a search pattern.

`-next | -prev | -all`

Find the next, previous, or all matched declarations.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBFindScope -win $_nTrace2 -name "Foo" -case -next
```

srcTBGoHistory

Description

Go backward or forward in the trace history, and change the view accordingly.

Syntax

```
srcTBGoHistory -win window -forward | -backward
```

Arguments

`-forward | -backward`

Go backward or forward in the trace history.

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBGoHistory -win $_nTrace2 -forward
```

srcTBGotoBrkPnt

Description

Jump to the selected breakpoint and highlight the breakpoint line.

Syntax

`srcTBGotoBrkPnt -win` *`window`* `-index` *`breakpointID`*

Arguments

`-index` *`breakpointID`*

Specify the breakpoint ID.

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBGotoBrkPnt -win $_nTrace1 -index 2
```

srcTBInsertObject

Description

Add the object id to the *Watch* tab.

Syntax

```
srcTBInsertObject -win window -tab tabId -object objId
```

Arguments

`-object objId`

Specify the object ID.

`-tab tabId`

Specify the Watch tab ID.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBInsertObject -win $_nTrace1 -tab 1 -object "p.demo_unit3@1"
```

srcTBJumpDebugPos

Description

Jump to the current debug position and highlight the line.

Syntax

```
srcTBJumpDebugPos -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBJumpDebugPos -win $_nTrace2
```

srcTBMsgDClick

Description

Double-click on the line in the *Message* pane of the *Testbench Browser* window.

Syntax

```
srcTBMsgDClick -win window -line linenumber -word pos
```

Arguments

`-line` *`linenumber`*

Specify the line in the *Message* pane of the *Testbench Browser* window.

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

`-word` *`pos`*

Click a specific word in the line.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBMsgDClick -win $_nTrace2 -line "0" -word "0"
```

srcTBOpenForm

Description

Open the *Testbench Browser* window.

Syntax

```
srcTBOpenForm -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBOpenForm -win $_nTrace2
```

srcTBRenameDataView

Description

Rename the currently used *Watch* tab(s).

Syntax

```
srcTBRenameDataView -win window -tab tabId -name “NewTabName”
```

Arguments

`-name “NewTabName”`

Specify the new tab name.

`-tab tabId`

Specify the Watch tab ID.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBRenameDataView -win $_nTrace2 -tab 1 -name "DataView 1"
```

srcTBSetBrkPnt

Description

Delete or enable/disable the specified or all breakpoints in non-interactive debug mode.

Syntax

```
srcTBSetBrkPnt -win window [-delete] [-enable | -disable]
[-index bpID] [-all]
```

Arguments

`-all`

When specified, all breakpoints in the *Testbench* Browser are removed or enabled/disabled.

`-delete`

Delete breakpoints.

`-enable | -disable`

Enable or disable breakpoints.

`-index bpID`

Specify the breakpoint ID.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetBrkPnt -win $_nTrace2 -delete -index 0
srcTBSetBrkPnt -win $_nTrace2 -enable -all
```

srcTBSetDVRadix

Description

Set the signal value format of the selected nodes in the *Interactive Sim* tab.

Syntax

```
srcTBSetDVRadix -win window [-tab tabID]
-radix Bin | Oct | Dec | Hex [-2Com | -Unsigned] -root RootNodeName
-path FullPathNodeName -scope FullScopeName
```

Arguments

`-path FullPathNodeName`

The name from root to the changed node.

`-radix Bin | Oct | Dec | Hex`

Set the signal value of the selected nodes in **Binary**, **Octal**, **Decimal**, or **Hexadecimal** format.

`-root RootNodeName`

The root node name.

`-scope FullScopeName`

The scope information of root node (if it is not *Local* tab).

`-tab tabID`

Specify the current watch tab ID.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

`-2Com | -Unsigned`

Set the signal value of the selected nodes in **Signed 2's Complement** or **Unsigned** format.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetDVRadix -win $_nTrace2 -radix "Bin" -root "limit" -path
"limit" -scope "p.unnamed_\$thread_sv_30"
srcTBSetDVRadix -win $_nTrace2 -tab 1 -root "A" -path "A/B[2:0]/
B[0]/C" -scope "a.b.c" -radix dec -2Com
```

srcTBSetFullClassViewMode

Description

Display all or partial classes in the *Inheritance View* tab.

Syntax

```
srcTBSetFullClassViewMode -win window -toggle on | off
```

Arguments

`-toggle on | off`

When this option is *on*, all classes from which the selected class extends are displayed in the *Inheritance View* tab. When this option is *off*, the sibling classes which are in the ovm_pkg package with the prefix "ovm_", "avm_", or "vmm_" are filtered and not displayed.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetFullClassViewMode -win $_nTrace2 -toggle off
```

srcTBSetSynHDLSource

Description

Synchronize the HDL source code between *nTrace* and *Testbench Browser*.

Syntax

```
srcTBSetSynHDLSource -win window -enable | -disable
```

Arguments

`-enable | -disable`

Enable or disable the synchronization between *nTrace* and *Testbench Browser*.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetSynHDLSource -win $_nTrace2 -enable
```

srcTBShowDataViewTip

Description

Show the tip of the selected data in the *Watch* tab.

Syntax

```
srcTBShowDataViewTip -win window -showTip on | off
```

Arguments

`-showTip on | off`

Specify whether to display the tip of the selected data.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBShowDataViewTip -win $_nTrace2 -showTip on
```

srcTBShowDeclTreeInstantiation

Description

Find all instantiations of the currently-selected tree node on the *Declaration* tab. The results are shown on the *Message* frame.

Syntax

```
srcTBShowDeclTreeInstantiation
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBShowDeclTreeInstantiation
```

srcTBShowObjRefFromSrc

Description

Show the object reference with the selected class object on the *Source Code* pane.

Syntax

```
ssrcTBShowObjRefFromSrc -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBShowObjRefFromSrc -win $_nTrace1
```

srcTBSrcDClick

Description

Double-click on the line in the *Source Code* pane of the *Testbench Browser* window.

Syntax

```
srcTBSrcDClick -win window -line linenumber -word pos
```

Arguments

`-line` *`linenumber`*

Specify the line in the *Source Code* pane of the *Testbench Browser* window.

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

`-word` *`pos`*

Click a specific word in the line.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSrcDClick -win $_nTrace2 -line "5" -word "12"
```

srcTBTreeAction

Description

Set the active node in the *Declaration Hierarchy* pane of the *Testbench Browser* window.

Syntax

```
srcTBTreeAction -win window -path "pathName"
```

Arguments

`-path "`*`pathName`*`"`

Specify the path. Double quotes must be used to enclose the path.

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTreeAction -win $_nTrace2 -path "Foo::print"
```

tbDebugAddBrkPnt

Description

Add breakpoints on the selected source code line or add breakpoints by time in interactive debug mode.

Syntax

```
tbDebugAddBrkPnt -win window [-line lineNum] [-file fileName]
[-abs | -relative time]
```

Arguments

`-absolute | -relative time`

Specify the absolute or relative time.

`-file fileName`

Specify the full path of the file.

`-line lineNum`

Specify the line number.

`-win window`

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tbDebugAddBrkPnt -line 12 -file /verdi/temp/thread/thread.sv
tbDebugAddBrkPnt -abs 500
```

tbDebugSetBrkPnt

Description

Delete or enable/disable the specified or all breakpoints in interactive debug mode.

Syntax

```
srcTBSetBrkPnt -win window [-delete] [-enable | -disable]
[-index bpID]
```

Arguments

`-delete`

Delete breakpoints.

`-enable | -disable`

Enable or disable breakpoints.

`-index` *`bpID`*

Specify the breakpoint ID.

`-win` *`window`*

Specify the window ID of the invoking *Testbench Browser* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tbDebugSetBrkPnt -delete -index 8
tbDebugSetBrkPnt -enable -index 6
```

tbDebugSetVarInfo

Description

Set the variable information of the selected nodes in the *Interactive Sim* tab.

Syntax

```
tbDebugSetVarInfo -name signalName -value signalValue
-deposit | -freeze
```

Arguments

`-deposit | -freeze`

If the **-deposit** option is specified, the signal value is forced once. If the **-freeze** option is specified, the signal value is the value specified.

`-name` *`signalName`*

The specified signal name.

`-value` *`signalValue`*

Specify the signal value to be forced during Interactive Debug.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tbDebugSetVarInfo -name this.array -value 0 -deposit
```

Interactive Debug

Local and Watch Tabs

srcTBAddDataView

Description

Add additional Watch tabs.

Syntax

```
srcTBAddDataView -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBAddDataView -win $_nTrace1
```

srcTBDeleteDataView

Description

Delete the current *Watch* tab.

Syntax

```
srcTBDeleteDataView -win window tab tabId
```

Arguments

`-tab` *`tabId`*

Specify the *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

srcTBDeleteDataView -win $_nTrace1 -tab 2

srcTBDVCollapse

Description

Collapse tree node(s) on *Local* or *Watch* tab.

Syntax

```
srcTBDVExpand -win window tab tabId [-item {node_index}]
```

Arguments

`-item {node_index}`

Specify the tree node index to expand.

`-tab tabId`

Specify the *Local* or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVCollapse -win $_nTrace1 -tab 0 -item {1}
srcTBDVCollapse -win $_nTrace1 -tab 0
```

srcTBDVDrag

Description

Drag selected tree node(s) from *Local* or *Watch* tab.

Syntax

```
srcTBDVDrag -tab tabId
```

Arguments

`-tab` *`tabId`*

Specify the *Local* or *Watch* tab ID.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVDrag -tab 0
```

srcTBDVDrop

Description

Drop selected tree node(s) to the *Local* or *Watch* tab.

Syntax

```
srcTBDVDrop -tab tabId
```

Arguments

`-tab` *`tabId`*

Specify the Local or Watch tab ID.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVDrop -tab 1
```

srcTBDVExpand

Description

Expand tree node(s) on *Local* or *Watch* tab.

Syntax

```
srcTBDVExpand -win window tab tabId [-item {node_index}]
```

Arguments

`-item {node_index}`

Specify the tree node index to expand.

`-tab tabId`

Specify the *Local* or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVExpand -win $_nTrace1 -tab 0 -item {1}
srcTBDVExpand -win $_nTrace1 -tab 0
```

srcTBDVSelect

Description

Select tree node(s) on *Local* or *Watch* tab.

Syntax

```
srcTBDVSelect -tab tabId -range {node1_index} {node2_index} …
```

Arguments

`-range {node1_index} {node2_index}`

Specify the index of selected tree node.

`-tab tabId`

Specify the *Local* or *Watch* tab ID.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVSelect -tab 0 -range {1 3 2-2} {1 3 3-3} {1 3 1-1} {1 3 0-
0}
```

srcTBDVSort

Description

Sort tree node(s) on *Local* or *Watch* tab.

Syntax

```
srcTBDVSort -ascending | -decending -tab tabId -column columnNo
```

Arguments

`-ascending | -decending`

Specify to sort tree node column in ascending or descending order.

`-column` *`columnNo`*

Specify the number of the column to be sorted.

`-tab` *`tabId`*

Specify the *Local* or *Watch* tab ID.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVSort -ascending -tab 0 -column 0
srcTBDVSort -decending -tab 0 -column 1
```

srcTBRenameDataView

Description

Rename the *Watch* tab.

Syntax

```
srcTBRenameDataView -win window tab tabId -name NewTabName
```

Arguments

`-name NewTabName`

Specify the new tab name.

`-tab tabId`

Specify the *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBRenameDataView -win $_nTrace1 -tab 2 -name "NewWatchName"
```

srcTBSwitchWatchView

Description

Switch the *Watch* tab.

Syntax

```
srcTBSwitchWatchView -win window tab tabId
```

Arguments

`-tab tabId`

Specify the *Local* or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSwitchWatchView -win $_nTrace1 -tab 1
```

Simulation Commands

srcTBAddBrkPnt

Description

Add the breakpoint with specified options.

Syntax

```
srcTBAddBrkPnt -line lineNumber -file filename
[-instance id | -object var | -object_id id] [-cond expression]
[-mindex macroIndex -moffset macroOffset | -poffset protectOffset]
[-start_col startColumn -end_col endColumn]
[-skip number | -once] [-quiet] [-command tclScript]
```

```
srcTBAddBrkPnt -abs time | -relative time [-cond expression] [-skip
number | -once] [-quiet] [-command tclScript]
```

```
srcTBAddBrkPnt -posedge id | -negedge id | -change id
[-cond expression] [-skip number | -once] [-quiet]
[-command tclScript]
```

```
srcTBAddBrkPnt -in name [-end] [-object var | -object_id id]
[-cond expression] [-skip number | -once] [-quiet]
[-command tclScript]
```

```
srcTBAddBrkPnt -assert id [-start | -success | -failure | -end |
-any] [-skip number | -once] [-quiet] [-command tclScript]
```

```
srcTBAddBrkPnt -solver [-once | -serial number | -skip number |
-condition expression] [-class name | -object_id id]
[-solver_cond expression]
```

```
srcTBAddBrkPnt -thread id [-enter | -end] [-cond expression]
[-skip number | -once] [-quiet] [-command tclScript]
```

Arguments

`–abs time`

Specify the absolute time.

`-assert id`

Specify the assertion ID.

`-change id`

Change the signal or object id.

`-class name`

Specify the class name that the built-in randomize method belongs to.

`-command tclScript`

Specify the Tcl script that contains Tcl commands to execute when reaching the stop point.

`-cond expression`

Specify the expression to determine whether the stop point is triggered.

`-end`

Stop pointing to the end of the task or function.

`-end_col endColumn`

Specify the end column-index of the statement.

`-enter | -end`

When the **-enter** option is specified, create or resume the thread. When the **-end** option is specified, terminate the thread.

NOTE: The **-enter** and **-end** options become valid after the **-thread** option is specified.

`-file filename`

Specify file name.

`-in name`

Specify the function or task name.

`-instance id`

Specify the instance path with the hierarchical path name.

`-kill checkpointId`

Delete the specified checkpoint.

`-line lineNumber`

Specify the line number.

`-mindex macroIndex`

Specify the macro index.

`-moffset macroOffset`

Specify the macro offset.

`-negedge id`

Search by falling edge.

`-object_id id`

Specify a class object ID and override the variable specified with the **-object** option.

`-object` *var*
Specify a class variable for the class object.

`-once`
Skip the first breakpoint before stopping the simulation.

`-poffset` *protectOffset*
Specify the protected offset.

`-posedge` *id*
Search by rising edge.

`-quiet`
Disable the printed message that is triggered when reaching the stop point.

`-relative` *time*
Specify the relative time,

`-skip` *number*
Specify the time to skip the breakpoints before stopping the simulation.

`–solver`
Switch to constraint debug mode.

`-start | -success | -failure | -end | -any`
Specify the assertion type. The types include: start, success, failure, end, and any.

`-start_col` *startColumn*
Specify the start column-index of the statement.

`-thread` *id*
Specify the thread ID.

Value Returned

The breakpoint index.

Example

```
srcTBAddBrkPnt -line 90 -file test.sv
srcTBAddBrkPnt -abs 30ns
srcTBAddBrkPnt -change "\\p::xFoo1 @2.i2" -posedge
srcTBAddBrkPnt -in "{p.\xFoo1::new }"
srcTBAddBrkPnt -asser abc -failure
srcTBAddBrkPnt -solver -class ClassA
srcTBAddBrkPnt -thread 2 -end
```

srcTBChkPnt

Description

Add, delete, or rewind the checkpoint.

Syntax

```
srcTBChkPnt -add
srcTBChkPnt -kill checkpointId
srcTBChkPnt -join checkpointId
```

Arguments

`–add`

Add a checkpoint.

`-join` *`checkpointId`*

Rewind the specified checkpoint.

`-kill` *`checkpointId`*

Delete the specified checkpoint.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBChkPnt -add
srcTBChkPnt -kill 2
srcTBChkPnt -join 1
```

srcTBDelAllBrkPnt

Description

Delete all breakpoints.

Syntax

```
srcTBDelAllBrkPnt
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDelAllBrkPnt
```

srcTBDisAllBrkPnt

Description

Disable all breakpoints.

Syntax

```
srcTBDisAllBrkPnt
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDisAllBrkPnt
```

srcTBDumpObject

Description

Dump the specified object to the FSDB file.

Syntax

```
srcTBDumpObject -var object1@object2@...
```

Arguments

`-var` *`object1@object2@...`*

Specify the names of the object. The delimiter is the at sign (@).

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDumpObject -var "p.xFoo1@1.aaaaa"
```

srcTBDumpVar

Description

Dump the specified signal or scope to the FSDB file.

Syntax

```
srcTBDumpVar -var signal1@signal2@...
```

Arguments

`-var signal1@signal2@...`

Specify the names of the signal or scope. The delimiter is the at sign (@).

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDumpVar -var "top.sigA@top.sigB"
```

srcTBEnAllBrkPnt

Description

Enable all breakpoints.

Syntax

```
srcTBEnAllBrkPnt
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBEnAllBrkPnt
```

srcTBInvokeSim

Description

Invoke simulation and interactive debug mode.

Syntax

```
srcTBInvokeSim
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBInvokeSim
```

srcTBLoadBrkPnt

Description

Load the breakpoint from the specified file.

Syntax

```
srcTBLoadBrkPnt -file filename
```

Arguments

`-file` *`filename`*

Specify the file to load the breakpoint from.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBLoadBrkPnt -file abc
```

srcTBOpenViewLog

Description

Display the content of the simulation log file in a new *Source Code* pane.

Syntax

```
srcTBOpenViewLog
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBOpenViewLog
```

srcTBRebuild

Description

This command allows you to perform the following:

- Rebuild KDB and simv binary
- Restart simulator after design rebuild

Syntax

```
-fromVCSScript | -fromCmd rebuild_command –path command_path
[-reloadKDB | -reloadByVCSScript | -reloadByCmd rebuild_command]
[-restartSim restart_options]
```

Arguments

-fromVCSScript

Specify to use VCS script to rebuild KDB and simv binary.

-fromCmd rebuild_command -path command_path

Specify to use custom command or script to rebuild KDB and simv binary.

-reloadKDB

Reload KDB with the same import options after design rebuild.

-reloadByVCSScript

Load KDB using `simv.kdb` generated by VCS.

-reloadByCmd rebuild_command

Use specified command or script to load KDB after design rebuild.

-restartSim restart_options

Specify to restart simulator after design rebuild. It should be followed by the restart option.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBRebuild -fromCmd rebuild –reloadByVCSScript –restart “-sv
test.sv –sml=verdi”
```

srcTBRestoreSimState

Description

Restore the simulation.

Syntax

`srcTBRestoreSimState -file` *`filename`*

Arguments

`-file` *`filename`*

Specify the state file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBRestoreSimState -file abc.bin
```

srcTBRunSim

Description

Run the simulation.

Syntax

```
srcTBRunSim -opt simOption
```

Arguments

`-opt` *`simOption`*

Specify the simulation options.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBRunSim -opt {2ns}
```

srcTBSaveBrkPnt

Description

Save the breakpoint to the specified file.

Syntax

```
srcTBSaveBrkPnt -file filename
```

Arguments

`-file` *`filename`*

Specify the file to save the breakpoint to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSaveBrkPnt -file abc
```

srcTBSaveSimState

Description

Save the simulation state to the specified file.

Syntax

```
srcTBSaveSimState -file filename
```

Arguments

`-file filename`

Specify the state file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSaveSimState -file abc.bin
```

srcTBSetBrkPnt

Description

Enable, disable, or delete the breakpoint.

Syntax

```
srcTBSetBrkPnt -index stopId [-disable | -enable | -delete]
```

Arguments

`-delete`

Delete the breakpoint.

`-enable`

Enable the breakpoint.

`-disable`

Disable the breakpoint.

`-index stopId`

Specify the breakpoint id.

Value Returned

The breakpoint index.

Example

```
srcTBSetBrkPnt -index 2 -disable
srcTBSetBrkPnt -index 2 -delete
```

srcTBSetRebuildOption

Description

Allows you to prefill the custom commands and simulator options for rebuild and restart (**Simulation** -> **Rebuild and Restart** form).

Syntax

```
srcTBSetRebuildOption  [-simOpt "option"] [-rebuildCmd "command"]
[-rebuildDir "directory"]
```

Arguments

-simOpt

Specify the simulator options.

-rebuildCmd

Specify the rebuild command.

-rebuildDir

Specify the rebuild directory. If this argument is not specified, the current working directory is used as the default directory.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetRebuildOption -simOpt “+UVM_TESTNAME=my_test -l
my_sim.log” -rebuildCmd “make vcs_comp” -rebuildDir “../../”
```

srcTBSetVarInfo

Description

Force or deposit a value of a signal or variable.

Syntax

```
srcTBSetVarInfo [-name name] [-value value]
[-freeze | -deposit] [-repeat interval] [-cancel time] [-drive]
[-object_id objId]
```

Arguments

`-cancel time`

Specify the time to release the forced value.

`-deposit`

Overwrite the forced value with a subsequent driver transaction.

`-drive`

Attach a new driver to the signal

NOTE: This option is for VHDL only.

`-freeze`

Freeze the value to the forced value.

`-name name`

Specify the name of the signal or variable.

`-object_id objId`

Specify the object id.

`-repeat interval`

Specify the interval to repeat the waveform.

`-value value`

Specify the value to force or deposit.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetVarInfo -name "b" -value "1" -object_id "\p::xFoo1 @1" -
deposit
```

srcTBSimBreak

Description

Stop the current simulation.

Syntax

```
srcTBSimBreak
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSimBreak
```

srcTBSimKill

Description

Force to end the current simulator process.

Syntax

```
srcTBSimKill
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSimKill
```

srcTBSimQuit

Description

Quit the current simulation.

Syntax

```
srcTBSimQuit
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSimQuit
```

srcTBSimReset

Description

Restart the simulation.

Syntax

```
srcTBSimReset
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSimReset
```

srcTBStartSimParallel

Description

Import the design and invoke the simulation simultaneously.

NOTE: The **srcTBStartSimParallel** command must be used with the **debImport** and s**rcTBInvokeSim** commands.

Syntax

```
srcTBStartSimParallel
```

Value Returned

None.

Example

```
srcTBStartSimParallel
debImport "-sv" "class_in_class.sv"
srcTBInvokeSim
```

srcTBStepIn

Description

Step to the next simulation task or function.

Syntax

```
srcTBStepIn [-tb | –thread tid | –solver] [-re_randomize]
[–dist_num number –dist_cont] [-opt simOption]
```

Arguments

`–dist_cont`

Continue to randomize the distribution based on the result of the last randomization. This option works only when the **-re_randomize** and **–dist_num** options are specified.

`–dist_num` *`number`*

Specify the number of times to randomize the distribution. This option works only when the **-re_randomize** and **–dist_cont** options are specified.

`-opt` *`simOption`*

Specify the simulation option.

`-re_randomize`

Randomize the distribution again. This option works only when the **-solver** options is specified.

`–solver`

Switch to constraint debug mode.

`-tb`

Step into the testbench.

`-thread` *`tid`*

Specify the ID of the thread where the simulation steps proceed.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStepIn  -tb
srcTBStepIn  -thread 2
srcTBStepIn  -sovler
```

srcTBStepNext

Description

Step to the next simulation task or function.

Syntax

`srcTBStepNext [-thread` *`tid`*`] [-opt` *`simOption`*`]`

Arguments

`-opt simOption`

Specify the simulation option.

`-thread tid`

Specify the ID of the thread where the simulation steps proceed.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStepNext
srcTBStepNext -opt {-end}
srcTBStepNext -thread 2
```

srcTBStepOut

Description

Stop the current simulation task or function.

Syntax

```
srcTBStepOut [-opt simOption]
```

Arguments

`-opt simOption`

Specify the simulation option.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStepOut -opt "-reverse"
```

srcTBUnSetVarInfo

Description

Release the value of a signal or variable.

Syntax

```
srcTBUnSetVarInfo [-name name]
```

Arguments

`-name` *`name`*

Specify the name of the signal or variable.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBUnSetVarInfo -name top.sigA
```

Class Tab

srcTBClassBrowser

Description

Select an object in the *Class* pane.

Syntax

```
srcTBClassBrowser -treeSel -select objectName
-treeExpand objectName -treeCollapse objectName -showDef
```

Arguments

`-select objectName`

Specify the desired object. This option works only when the **-treeSel** option is specified.

`-showDef`

Show the definition of the currently selected object.

`-treeCollapse objectName`

Collapse the hierarchy tree that the selected object belongs to.

`-treeExpand objectName`

Expand the hierarchy tree that the selected object belongs to.

`-treeSel`

Select the object from the hierarchy tree.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBClassBrowser -treeSel -select "vmm_unit\(_vcs_vmm\)"
srcTBClassBrowser srcTBClassBrowser -treeCollapse
"vmm_unit\(_vcs_vmm\)"
srcTBClassBrowser -treeExpand "vmm_unit\(_vcs_vmm\)"
srcTBClassBrowser -showDef
```

srcTBFindClassBrowserNext

Description

Search forward or backward for the next node in the *Class* tab based on the specified string.

Syntax

```
srcTBFindClassBrowserNext -name string [-prev]
```

Arguments

`-name string`

Specify the string to search for.

`-prev`

Search for the next node backward.

NOTE: If this option is not specified, search for the next node forward.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBFindClassBrowserNext -name "test" -prev
```

Member Tab

srcTBFindMemberViewNext

Description

Search forward or backward for the next node in the *Member* tab based on the specified string.

Syntax

```
srcTBFindMemberViewNext -name string [-prev]
```

Arguments

`-name string`

Specify the string to search for.

`-prev`

Search for the next node backward.

NOTE: If this option is not specified, search for the next node forward.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBFindMemberViewNext  -name "test" -prev
```

srcTBMemberView

Description

Select an object in the *Member* pane.

Syntax

```
srcTBMemberView -treeSel -select objectName -treeExpand objectName
-treeCollapse objectName -showDef
```

Arguments

`-select objectName`

Specify the desired object. This option works only when the **-treeSel** option is specified.

`-showDef`

Show the definition of the currently selected object.

`-treeCollapse objectName`

Collapse the hierarchy tree that the selected object belongs to.

`-treeExpand objectName`

Expand the hierarchy tree that the selected object belongs to.

`-treeSel`

Select the object from the hierarchy tree.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBMemberView -treeSel -select "Variables.exp"
srcTBMemberView -treeExpand "Variables"
srcTBMemberView -treeCollapse "Variables"
srcTBMemberView -showDef
```

srcTBMemberViewFilter

Description

Set the filter string and the filter type in the *Member* tab.

Syntax

```
srcTBMemberViewFilter -pattern patternStr -type type1\type2
```

Arguments

`-pattern patternStr`

Specify the pattern to filter.

`-type type1 type2`

Specify the type to filter.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBMemberViewFilter -pattern "test" -type basemember \
staticmember localmember protmember pubmember \
iterconstr orderconstr regularconst
```

Reference View

srcTBReference

Description

Search forward or backward for the next node in the *Member* tab based on the specified string.

Syntax

```
srcTBReference -select objectName -addreference -showSource
-addobject -showInObj -path filterPattern -objectId filterPattern
-update
```

Arguments

`-addobject`

Add the selected object to the *Watch* tab.

`-addreference`

Add the selected object reference to the *Watch* tab.

`-objectId filterPattern`

Set the pattern string for the object ID filter.

`-path filterPattern`

Set the pattern string for the path filter.

`-select objectName`

Select the desired object in the *Reference* tab.

`-showInObj`

Show the selected object in the object browser.

`-showSource`

Display the source code of the selected object.

`-update`

Force to update the *Reference* Tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBReference -select "P.vmm_log.known"
srcTBReference -addreference
srcTBReference -showSource
srcTBReference -addobject
srcTBReference -showInObj
srcTBReference -path "*test
srcTBReference -objectId "test*"
srcTBReference -update
```

Object Tab

srcTBFindObjectBrowserNext

Description

Search forward or backward for the next node in the *Object* tab based on the specified string.

Syntax

```
srcTBFindObjectBrowserNext -name string [-prev]
```

Arguments

`-name string`

Specify the string to search for.

`-prev`

Search for the next node backward.

NOTE: If this option is not specified, search for the next node forward.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBFindObjectBrowserNext -name "test" -prev
```

srcTBObjectBrowser

Description

Select an object in the *Object* tab.

Syntax

```
srcTBObjectBrowser -treeSel -select objectName
-treeExpand objectName -treeCollapse objectName -showDef
```

Arguments

`-select objectName`

Specify the desired object. This option works only when the **-treeExpand** option is specified.

`-showDef`

Show the definition of the currently selected object.

`-treeCollapse objectName`

Collapse the hierarchy tree that the selected object belongs to.

`-treeExpand objectName`

Expand the hierarchy tree that the selected object belongs to.

`-treeSel`

Select the object from the hierarchy tree.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBObjectBrowser -treeSel -select "_vcs_vmm.vmm_debug"
srcTBObjectBrowser -treeExpand "_vcs_vmm.vmm_debug"
srcTBClassBrowser -treeCollapse "_vcs_vmm.vmm_data"
srcTBObjectBrowser -showDef
```

srcTBObjectBrowserFilter

Description

Set the filter string and the filter type in the *Object* tab.

Syntax

```
srcTBObjectBrowserFilter -pattern string -classpattern string
-type type1\type2
```

Arguments

`-classpattern string`

Specify the pattern of the class type to filter.

`-pattern string`

Specify the pattern to filter.

`-type` *`type1 type2`*

Specify the type to filter. The options incude: **package_class_vars**, **module_class_vars**, **class_static_vars**, **task_static_vars**, **task_auto_vars**, **all_variables**, **base_members**, **null_reference**, **empty_arrays**, and **all**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBObjectBrowserFilter -pattern "ab*" -classpattern \ "vmm*"
-type package_class_vars module_class_vars
```

srcTBObjectBrowserMode

Description

Switch object browser mode.

Syntax

```
srcTBObjectBrowserMode -type modeName
```

Arguments

`-type` *`modeName`*

Specify the mode to switch to. The options include: **allobjects**, **uvmcomponents**, **uvmobjects**, **ovmcomponents**, **ovmobjects**, **vmmcomponents**, and **vmmobjects**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBObjectBrowserMode -type uvmobjects
```

Stack Tab

srcTBStackBrowser

Description

Select an object in the *Stack* Tab.

Syntax

```
srcTBClassBrowser -select objectName -active -activate objectName
-stack -thread
```

Arguments

`-activate objectName`

Activate the selected object.

`-active`

Activate the object selection.

`-select objectName`

Specify the desired object.

`-stack`

Specify the stack view in Thread mode.

`-thread`

Specify the thread view in Thread mode.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStackBrowser -select "P@0" -active -stack
srcTBStackBrowser -active -activate "P@0" -thread
```

srcTBStackFind

Description

Search forward or backward for the next node in the *Stack* tab based on the specified string.

Syntax

```
srcTBFindClassBrowserNext -prev -pattern string
```

Arguments

`-pattern` *`string`*

Specify the pattern string to search for.

`-prev`

Search for the next node backward.

NOTE: If this option is not specified, search for the next node forward.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStackFind -prev -pattern "uname*"
```

srcTBStackNameFilter

Description

Enable or disable the name filter in the *Stack* tab.

Syntax

```
srcTBStackNameFilter [-off]
```

Arguments

`-off`

Disable the name filter in the *Stack* tab.

NOTE: If the *-off* option is not specified, enable the name filter in the *Stack* tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStackNameFilter -off
```

srcTBStackSwitchMode

Description

Switch the *Stack* tab between active mode and thread mode.

Syntax

```
srcTBStackSwitchMode [-thread]
```

Arguments

`-thread`

Switch the *Stack* tab to thread mode.

NOTE: If the **-thread** option is not specified, switch to active mode.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStackSwitchMode -thread
```

Constraint_Debug Tab

srcTBConstrDbgAnnotate

Description

Annotate original randomize results in the Rerand Solver view.

Syntax

```
srcTBConstrDbgAnnotate
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgAnnotate
```

srcTBConstrDbgDisplaySourceCode

Description

Display the source code of the selected object.

Syntax

```
srcTBConstrDbgDisplaySourceCode -view viewName
```

Arguments

`-view` *`viewName`*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgDisplaySourceCode -view Solver
```

srcTBConstrDbgSplitMerge

Description

Split the Constraint views including *Solver*, *Relation*, *Solver_R*, *Relation_R*, *Dist_R*, and *Histogram_R* to separate or merge views.

Syntax

```
srcTBConstrDbgSplitMerge
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgSplitMerge
```

srcTBConstrDbgViewMode

Description

Switch mode for the *Constraint Debug* tab.

Syntax

```
srcTBConstrDbgViewMode -mode modeName
```

Arguments

`-mode` *`modeName`*

Specify the mode to switch to. The options include: **Origin**, **Rerand**, and **All**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgViewMode -mode Origin
```

srcTBSolverTreeFilterType

Description

Input filter string or select/deselect filter type in filter drop down menu in *Constraint Debug* pane.

Syntax

```
srcTBSolverTreeFilterType -pattern patternStr -type type1\type2...
```

Arguments

`-pattern patternStr`

Specify the pattern to filter.

`-type type1 type2`

Specify the type to filter. The options include: **on_block**, **off_block**, **soft_dropped_cnst**, **soft_hornored_cnst**, and **nonsoft_cnst**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSolverTreeFilterType -pattern "test*" -type on_block \
off_block soft_dropped_cnst soft_hornored_cnst
```

srcTBSolverTreeSearchNext

Description

Search forward or backward for the next node in the *Constraint Debug* tab based on the specified string.

Syntax

```
srcTBSolverTreeSearchNext -prev -pattern string
-type solved variable | constraint block
```

Arguments

`-pattern string`

Specify the pattern string to search for.

`-prev`

Search for the next node backward.

NOTE: If this option is not specified, search for the next node forward.

`-type solved variable | constraint block`

Specify the type to search for. The options include: **Solved Variable** and **Constraint Block**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSolverTreeSearchNext -type "Constraint Block" -pattern "abc"
-prev
```

srcTBStackShowClass

Description

Show the related class of the select object.

Syntax

```
srcTBStackShowClass -active
```

Arguments

`-view` *`viewName`*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgShowInClassBrowser -view Solver_R
```

tbvConstrDbg

Description

Select an object in the *Constraint Debug* tab.

Syntax

```
tbvConstrDbg [-select] [-expand] [-collapse] -view viewName
-name itemName -type typeName -parent name
```

Arguments

`-collapse`

Collapse the selected tree.

`-expand`

Expand the selected tree.

`-name itemName`

Specify the object.

`-parent name`

Specify the parent name of the object.

`-select`

Specify the desired object.

`-type typeName`

Specify the type of the selected item. The options include: **NoType**, **Partition**, **Folder**, **RandVar, Block**, **Constraint**, **DistVar**, and **DistConstraint**.

`-view viewName`

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tbvConstrDbg -select -view Solver -name "Partition:2" -type
Partition
tbvConstrDbg -expand -view Solver -name "i" -type RandVar -parent
"Partition:1"
```

```
tbvConstrDbg -collapse -view Solver -name "i" -type RandVar -
parent "Partition:1"
```

Right-click Commands

srcTBAddToWatchFromSrc

Description

Add the selected variable on the source view to the *Watch* tab.

Syntax

```
srcTBAddToWatchFromSrc -win window -tab tabId
```

Arguments

`-tab tabId`

Specify the Watch tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBAddToWatchFromSrc -win $_nTrace1 -tab 1
```

srcTBConstrAdd

Description

Create the new constraint.

Syntax

```
srcTBConstrAdd -expr exprList -obj classVar -block cnstrName
```

Arguments

`-block cnstrName`

Specify the constraint name.

`-expr exprList`

Specify the constraint expression list.

`-obj classVar`

Specify the class variable name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrAdd -expr {a>=10; b>=6; } -obj {this} -block {newCnstr}
```

srcTBConstrDbgAddToSearch

Description

Add the string to search.

Syntax

```
srcTBConstrDbgAddToSearch -view viewName -pattern string
-type type
```

Arguments

`-pattern string`

Specify the pattern to search.

`-type type`

Specify the item type to search.

`-view viewName`

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgAddToSearch -view Solver_R -pattern "bob\[1\]" -type
"Solved Variable"
```

srcTBConstrDbgCollapseAll

Description

Collapse the selected tree.

Syntax

```
srcTBConstrDbgCollapseAll -view viewName
```

Arguments

`-view` *viewName*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgCollapseAll -view Solver
```

srcTBConstrDbgExpandNodeByLevelAll

Description

Expand the hierarchy tree by levels.

Syntax

```
srcTBConstrDbgExpandNodeByLevelAll -view viewName
```

Arguments

`-view` *viewName*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgExpandNodeByLevelAll -view Solver
```

srcTBConstrDbgExtractTestcase

Description

Extract test cases for the selected item.

Syntax

```
srcTBConstrDbgExtractTestcase -dir directory
```

Arguments

`-dir` *`directory`*

Specify the directory to save the test cases to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgExtractTestcase -dir "/remote/directory/to/save"
```

srcTBConstrDbgSaveChange

Description

Save constraint changes to the specified file.

Syntax

```
srcTBConstrDbgSaveChange -file filename
```

Arguments

`-file` *`filename`*

Specify the file to save constraint changes to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgSaveChange -file "/remote/vgrnd67/zfyang/work/
constraint_changes.txt"
```

srcTBConstrDbgSetRadixBin

Description

Set the radix of selected item to be in Binary format.

Syntax

```
srcTBConstrDbgSetRadixBin -view viewName
```

Arguments

`-view` *`viewName`*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgSetRadixBin -view Solver
```

srcTBConstrDbgSetRadixDec

Description

Set the radix of selected item to be in Decimal format.

Syntax

```
srcTBConstrDbgSetRadixDec -view viewName
```

Arguments

`-view` *`viewName`*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgSetRadixDec-view Solver
```

srcTBConstrDbgSetRadixHex

Description

Set the radix of selected item to be in Hexadecimal format.

Syntax

```
srcTBConstrDbgSetRadixHex -view viewName
```

Arguments

`-view` *`viewName`*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgSetRadixHex-view Solver
```

srcTBConstrDbgSetRadixOct

Description

Set the radix of selected item to be in Octal format.

Syntax

```
srcTBConstrDbgSetRadixOct -view viewName
```

Arguments

`-view` *`viewName`*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgSetRadixOct-view Solver
```

srcTBConstrDbgSetRadixUnsigned

Description

Set the radix of selected item to be in unsigned format.

Syntax

```
srcTBConstrDbgSetRadixUnsigned -view viewName
```

Arguments

`-view` *`viewName`*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgSetRadixUnsigned -view Solver
```

srcTBConstrDbgSetRadix2Complement

Description

Set the radix of selected item to be in Signed 2's Complement format.

Syntax

```
srcTBConstrDbgSetRadixUnsigned -view viewName
```

Arguments

`-view` *viewName*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgSetRadix2Complement -view Solver
```

srcTBConstrDbgShowInLocalView

Description

Show the related item in the *Local* tab.

Syntax

```
srcTBConstrDbgShowInLocalView -view viewName
```

Arguments

`-view` *viewName*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgShowInLocalView -view Solver
```

srcTBConstrDbgShowNavigate

Description

Show the **Navigation** text field in the *Constraint Debug* tab.

Syntax

```
srcTBConstrDbgShowNavigate
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgShowNavigate
```

srcTBConstrDbgShowRelation

Description

Show related items for the variable.

Syntax

```
srcTBConstrDbgShowRelation -view viewName -var variable
```

Arguments

`-view` *`viewName`*

Specify the view for the selected item. The options include: **Solver**, **Relation**, **Solver_R**, **Relation_R**, **Dist_R**, and **Histogram_R**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgShowRelation -view Solver -var "i"
```

srcTBConstrDbgShowTip

Description

Show the tool tip in the *Constraint Debug* tab.

Syntax

```
srcTBConstrDbgShowTip
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDbgShowTip
```

srcTBConstrDisable

Description

Disable the constraint expression.

Syntax

```
srcTBConstrDisable -id exprId -obj classVar -block cnstrName
```

Arguments

`-block` *`cnstrName`*

Specify the constraint name.

`-id` *`exprId`*

Specify the constraint expression ID.

`-obj` *`classVar`*

Specify the class variable name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrDisable -id 0 -obj {bar1} -block {c_simple}
```

srcTBConstrEnable

Description

Enable the constraint expression.

Syntax

```
srcTBConstrEnable -id exprId -obj classVar -block cnstrName
```

Arguments

`-block cnstrName`

Specify the constraint name.

`-id exprId`

Specify the constraint expression ID.

`-obj classVar`

Specify the class variable name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBConstrEnable -id 0 -obj {bar1} -block {c_simple}
```

srcTBDataViewFilter

Description

Filter the tree in the *Local* or *Watch* tab.

Syntax

```
srcTBDataViewFilter -pattern patternStr -type basemember pubmember
protmember localmember otheraccessmember allaccessmember
constraintmember objectmember parametermember randmember
othervaluemember allVariables
-tab tabId
```

Arguments

`-pattern` *`patternStr`*

Specify the pattern to filter.

`-tab` *`tabId`*

Specify the *Local* or *Watch* tab ID.

`-type` *`basemember pubmember protmember localmember otheraccessmember allaccessmember constraintmember objectmember parametermember randmember othervaluemember allVariables`*

Specify the types to keep. The possible types include:

- **basemember:** Keep base class members.
- **pubmember:** Keep public-access variables.
- **protmember:** Keep protected-access variables.
- **localmember:** Keep local-access variables.
- **otheraccessmember:** Keep the access type that is not public, protected, or local.
- **allaccessmember:** Keep all access type.
- **constraintmember:** Keep all constraints.
- **objectmember:** Keep class objects.
- **parametermember:** Keep parameters.
- **randmember:** Keep rand variables.
- **othervaluemember:** Keep all data types that are not constraint, class-object, parameter, nor rand.
- **allVariables:** Keep all data types,

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDataViewFilter -pattern "ii" -type basemember pubmember
protmember localmember otheraccessmember allaccessmember
constraintmember objectmember parametermember randmember
othervaluemember allVariables -tab 0
```

srcTBDataViewFind

Description

Find variable name in the *Local* or *Watch* tab.

Syntax

```
srcTBDataViewFind -mode current | object | package
srcTBDataViewFind -name pattern -tab tabId
```

Arguments

`-mode current | object | package`

Specify search mode. Possible values include:

- **current:** Search only the current tree without expanding any tree nodes.
- **object:** Search the expand class object tree nodes.
- **package:** Search the expand class objects and package tree nodes.

`-name pattern`

Specify the pattern.

`-tab tabId`

Specify the *Local* or *Watch* tab ID.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDataViewFind -mode current
srcTBDataViewFind -name "ii" -tab 0
```

srcTBDataViewShowNavigate

Description

Enable the **Navigation** text field in the *Local* or *Watch* tab.

Syntax

```
srcTBDataViewShowNavigate -local
```

Arguments

`-local`

Specify the *Local* tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDataViewShowNavigate -local
```

srcTBDeleteAllDataTree

Description

Delete all data trees from the *Watch* tab.

Syntax

```
srcTBDeleteAllDataTree -win window -tab tabId
```

Arguments

`-tab tabId`

Specify the *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDeleteAllDataTree -win $_nTrace1 -tab 1
```

srcTBDeleteDataTree

Description

Delete the specified data tree from the *Watch* tab.

Syntax

```
srcTBDeleteDataTree -win window -tab tabId -tree rootName
```

Arguments

`-tab tabId`

Specify the *Watch* tab ID.

`tree rootName`

Specify the root node name.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDeleteDataTree -win $_nTrace1 -tab 1 -tree "stB"
```

srcTBDVAddObjToWave

Description

Add the specified object in the *Local* or *Watch* tab to the waveform.

Syntax

```
srcTBDVAddObjToWave -win window -tab tabId -item nodeIndex
```

Arguments

`-item nodeIndex`

Specify the tree node index.

`-tab tabID`

Specify the *Local* or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVAddObjToWave -win $_nTrace1 -tab 0 -item {6}
```

srcTBDVAddSignalToWave

Description

Add the specified signal in the *Local* or *Watch* tab to the waveform.

Syntax

```
srcTBDVAddSignalToWave -win window -tab tabId -item nodeIndex
```

Arguments

`-item nodeIndex`

Specify the tree node index.

`-tab tabID`

Specify the *Local* or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVAddSignalToWave -win $_nTrace1 -tab 0 -item {9}
```

srcTBDVAddTo

Description

Add the selected nodes to another or newly created *Watch* tab.

Syntax

```
srcTBDVAddTo -win window -tab tabID -toTab newTabID
-root RootNodeName -path FullPathNodeName -scope FullScopeName
```

Arguments

`-path FullPathNodeName`

The name from root to the changed node.

`-root RootNodeName`

The root node name.

`-scope FullScopeName`

The scope information of root node (if it is not *Local* tab).

`-tab tabID`

Specify the current *Watch* tab ID.

`-toTab newTabID`

Specify the newly created *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVAddTo -win $_nTrace2 -tab 1 -toTab 2 -root "limit" -path
"limit" -scope "p.unnamed_\$thread_sv_30"
```

srcTBDVDC

Description

Double-click the selected node on Local or Watch tabs.

Syntax

`srcTBDVDC -win window -tab tabId`

Arguments

`-tab tabId`

Specify the Local or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVDC -win $_nTrace1 -tab 0
```

srcTBDVMarker

Description

Set or unset the markers of the selected tree node(s) in the *Local* or *Watch* tab.

Syntax

```
srcTBDVMarker -win window -set | -unsetAll -tab tabID
-root RootNodeName -path FullPathNodeName
```

Arguments

`-path FullPathNodeName`

The name from root to the changed node.

`-root RootNodeName`

The root node name.

`-set | -unsetAll`

If **-set** is specified, set the marker of the selected tree node(s) in the *Watch/Local* tab. If **-unsetAll** is specified, remove all markers from the *Watch/Local* tab.

`-tab tabID`

Specify the current *Watch* tab ID.

`-toTab newTabID`

The newly created *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVMarker -win $_nTrace2 -set -tab 1 -root "bar" -path "bar"
```

srcTBDVShowDecl

Description

Show the declaration of the selected tree node in the *Local* or *Watch* tab.

Syntax

```
srcTBDVShowDecl -win window -tab tabId
```

Arguments

`-tab tabId`

Specify the Local or Watch tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVShowDecl -win $_nTrace1 -tab 0
```

srcTBDVShowDefine

Description

Show the definition of the selected tree node in the *Local* or *Watch* tab.

Syntax

```
srcTBDVShowDefine -win window -tab tabId
```

Arguments

`-tab tabId`

Specify the *Local* or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVShowDefine -win $_nTrace1 -tab 0
```

srcTBDVShowObjRef

Description

Open the *References* form to show the reference path of the selected tree node in the *Local* or *Watch* tab.

Syntax

```
srcTBDVShowObjRef -win window -tab tabId -item nodeIndex
```

Arguments

`-item nodeIndex`

Specify the tree node index.

`-tab tabId`

Specify the *Local* or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVShowObjRef -win $_nTrace1 -tab 0 -item {6}
```

srcTBDVShowRef

Description

Show the reference of the selected tree node in the *Local* or *Watch* tab in the *Message* pane.

Syntax

```
srcTBDVShowRef -win window -tab tabId
```

Arguments

`-tab tabId`

Specify the *Local* or *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBDVShowRef -win $_nTrace1 -tab 0
```

srcTBInsertDataTree

Description

Drag the selected variable from other view and drop it to the *Watch* tab.

Syntax

```
srcTBInsertDataTree -win window -tab tabId
-tree variableName [-scope | -signal]
```

Arguments

`-scope | -signal`

Specify *-scope* when the selected scope is an instance or scope. Specify *-signal* when the selected scope is a signal.

`-tab tabId`

Specify the *Watch* tab ID.

`-tree variableName`

Specify the variable name to inset to the data tree.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBInsertDataTree -win $_nTrace1 -tab 1 -tree "p.bar.iii3"
srcTBInsertDataTree -win $_nTrace1 -tab 1 -tree "top.a.b"
-scope
srcTBInsertDataTree -win $_nTrace1 -tab 1 -tree "top.a.b.sig"
-signal
```

srcTBInsertObject

Description

Insert the specified class object to the *Watch* tab.

Syntax

```
srcTBInsertObject -win window -tab tabId object objectId
```

Arguments

`object objectId`

Specify the class object ID.

`-tab tabId`

Specify the *Watch* tab ID.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBInsertObject -win $_nTrace1 -tab 1 -object
"\$unit::clsBase@1"
```

srcTBRunToSource

Description

Enable simulation running up to the specified line in the source file.

Syntax

```
srcTBRunToSource -file filename -line lineNumber [-offset
macroOffset] [-reverse]
```

Arguments

`-file` *`filename`*

Specify the file name.

`-line` *`lineNumber`*

Specify the line number.

`-offset` *`macroOffset`*

Specify the macro offset.

`-reverse`

If specified, the simulator goes back to the specified line number in the file.

Value Returned

1 if successful, 0 otherwise

Example

```
srcTBRunToSource -file test.sv -line 20
```

srcTBSetCnstrMode

Description

Set constraint mode for constraint debug.

Syntax

```
srcTBSetCnstrMode -cnstr cnstrName -object classVar -value 1|0
```

Arguments

`-object classVar`

Specify the class variable name.

`-value 1|0`

When this option is set to *1*, turn on the constraint mode. When this option is set to *0*, turn off the constraint mode.

`-cnstr cnstrName`

Specify the constraint name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetCnstrMode -cnstr c_simple -object {bar1} -value {1}
```

srcTBSetDVRadix

Description

Set the signal value format of the selected nodes in the *Local* or *Watch* tab.

Syntax

```
srcTBSetDVRadix -win window [-tab tabID]
-radix Bin | Oct | Dec | Hex [-2Com | -Unsigned] -root RootNodeName
-path FullPathNodeName -scope FullScopeName
```

Arguments

`-path FullPathNodeName`

The name from root to the changed node.

`-radix Bin | Oct | Dec | Hex`

Set the signal value of the selected nodes in **Binary**, **Octal**, **Decimal**, or **Hexadecimal** format.

`-root RootNodeName`

The root node name.

`-scope FullScopeName`

The scope information of root node (if it is not *Local* tab).

`-tab` *`tabID`*

Specify the current *Watch* tab ID.

`-win` *`window`*

Specify the window ID of the invoking source code window.

`-2Com | -Unsigned`

Set the signal value of the selected nodes in **Signed 2's Complement** or **Unsigned** format.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetDVRadix -win $_nTrace2 -radix "Bin" -root "limit" -path
"limit" -scope "p.unnamed_\$thread_sv_30"
srcTBSetDVRadix -win $_nTrace2 -tab 1 -root "A" -path "A/B[2:0]/
B[0]/C" -scope "a.b.c" -radix dec -2Com
```

srcTBSetRandMode

Description

Set random mode for constraint debug.

Syntax

`srcTBSetRandMode -var` *`varName`* `-object` ***`classVar`*** `-value 1|0`

Arguments

`-object` ***`classVar`***

Specify the class variable name.

`-value 1|0`

When this option is set to *1*, turn on the random mode. When this option is set to *0*, turn on the random mode.

`-var` *`varName`*

Specify the random variable name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBSetRandMode -var b -object {bar1} -value {1}
```

srcTBShowDataViewTip

Description

Show the tip in the *Local* or *Watch* tab.

Syntax

```
srcTBDeleteAllDataTree -win window -showTip on|off
```

Arguments

`-showTip on|off`

Specify *on* to show the tip. Specify *off* to stop showing the tip.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBShowDataViewTip -win $_nTrace1 -showTip on
```

srcTBShowSourceCode

Description

Display the source code of the selected stack.

Syntax

```
srcTBShowSourceCode -win window -file filename -line lineNumber
```

Arguments

`-file filename`

Specify the file name of source code.

`-line lineNumber`

Specify the line number of the source code to be shown.

`-win window`

Specify the window ID of the invoking source code window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBShowSourceCode -win $_nTrace1 -file "vmm.sv" -line 38087
```

srcTBStackShowClass

Description

Show the related class of the select object.

Syntax

```
srcTBStackShowClass -active
```

Arguments

`-active`

Activate the object selection.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStackShowClass -active
```

srcTBStackShowInfo

Description

Show the information of the selected file in the *Stack* tab.

Syntax

```
srcTBStackShowInfo -line number -scope scopeName -file filename
```

Arguments

`-file filename`

Specify the file.

`-line number`

Specify the line number in the source file.

`-scope scopeName`

Specify the scope to show the information in.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStackShowInfo -line 387 -scope "vmm_log::report" -file
"vmm.sv"
```

srcTBStackShowNavigate

Description

Show the **Navigation** text field in the *Stack* tab.

Syntax

```
srcTBStackShowNavigate [-hide]
```

Arguments

`-hide`

Hide the **Navigation** text field in the *Stack* tab.

NOTE: If this option is not specified, show the **Navigation** text field in the *Stack* tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStackShowNavigate -hide
srcTBStackShowNavigate
```

srcTBStackShowTip

Description

Show the tool tip in the *Stack* tab.

Syntax

```
srcTBStackShowTip [-hide]
```

Arguments

`-hide`

Hide the tool tip in the *Stack* tab.

NOTE: If this option is not specified, show the tool tip in the *Stack* tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBStackShowTip -hide
```

srcTBTBCBAddWatch

Description

Add the elected object to the *Watch* tab.

Syntax

```
srcTBTBCBAddWatch -tab tabId
```

Arguments

`-tab` *`tabId`*

Specify the *Watch* tab to add the object to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBAddWatch -tab 1
```

srcTBTBCBCollapseAll

Description

Collapse the selected tree.

Syntax

```
srcTBTBCBCollapseAll
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBCollapseAll
```

srcTBTBCBDisplaySourceCode

Description

Display the source code of the selected class.

Syntax

```
srcTBTBCBDisplaySourceCode
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBDisplaySourceCode
```

srcTBTBCBExpandTreeByLevelAll

Description

Expand the hierarchy tree by levels.

Syntax

```
srcTBTBCBExpandTreeByLevelAll -level all|2|3|4|5
```

Arguments

`-level all|2|3|4|5`

Specify the number of levels to be expanded.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBExpandTreeByLevelAll -level 3
```

srcTBTBCBSetConstraintBp

Description

Set constraint breakpoints.

Syntax

```
srcTBTBCBSetConstraintBp
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBSetConstraintBp
```

srcTBTBCBShowInfo

Description

Show the information of the selected file in the *Class* tab.

Syntax

```
srcTBTBCBShowInfo
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBShowInfo
```

srcTBTBCBShowMemory

Description

Show the memory of the selected object in the *Class* tab.

Syntax

```
srcTBTBCBShowMemory
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBShowMemory
```

srcTBTBCBShowNavigate

Description

Show the navigation text field in the *Class* pane.

Syntax

```
srcTBTBCBShowNavigate
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBShowNavigate
```

srcTBTBCBShowReferenceCount

Description

Show the reference count of the selected item.

Syntax

```
srcTBTBCBShowReferenceCount
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBShowReferenceCount
```

srcTBTBCBShowTip

Description

Show the tool tip in the *Class* tab.

Syntax

```
srcTBTBCBShowTip
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBCBShowTip
```

srcTBTBMVAddObjToWave

Description

Add the selected object to *nWave* window.

Syntax

```
srcTBTBMVAddObjToWave
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVAddObjToWave
```

srcTBTBMVAddReference

Description

Add the selected object reference to the specified *Watch* tab.

Syntax

```
srcTBTBMVAddWatch -tab tabId
```

Arguments

`-tab` *`tabId`*

Specify the *Watch* tab to add the object reference to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVAddReference -tab 1
```

srcTBTBMVAddWatch

Description

Add the selected object to the specified *Watch* tab.

Syntax

```
srcTBTBMVAddWatch -tab tabId
```

Arguments

`-tab` *`tabId`*

Specify the *Watch* tab to add the object to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVAddWatch -tab 1
```

srcTBTBMVCollapseAll

Description

Collapse the selected tree.

Syntax

```
srcTBTBMVCollapseAll
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVCollapseAll
```

srcTBTBMVDisplaySourceCode

Description

Display the source code of the selected class.

Syntax

```
srcTBTBMVDisplaySourceCode
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVDisplaySourceCode
```

srcTBTBMVDumpObjToFSDB

Description

Dump the selected object to the FSDB file.

Syntax

```
srcTBTBMVDumpObjToFSDB
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVDumpObjToFSDB
```

srcTBTBMVExpandTreeByLevelAll

Description

Expand the hierarchy tree by levels.

Syntax

```
srcTBTBMVExpandTreeByLevelAll -level all|2|3|4|5
```

Arguments

`-level all|2|3|4|5`

Specify the number of levels to be expanded.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVExpandTreeByLevelAll -level 2
```

srcTBTBMVSetBp

Description

Set the breakpoint of the selected object.

Syntax

```
srcTBTBMVSetBp
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVSetBp
```

srcTBTBMVSetConstraintBp

Description

Set constraint breakpoints

Syntax

```
srcTBTBMVSetConstraintBp
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVSetConstraintBp
```

srcTBTBMVShowInClassBrowser

Description

Show the related class of the select item in the *Class* tab.

Syntax

```
srcTBTBMVShowInClassBrowser
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVShowInClassBrowser
```

srcTBTBMVShowInfo

Description

Show the information of the selected file in the *Member* tab.

Syntax

```
srcTBTBMVShowInfo
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVShowInfo
```

srcTBTBMVShowNavigate

Description

Enable the **Navigation** text field in the *Member* tab.

Syntax

```
srcTBTBMVShowNavigate
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVShowNavigate
```

srcTBTBMVShowReference

Description

Show the references of the selected object.

Syntax

```
srcTBTBMVShowReference
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVShowReference
```

srcTBTBMVShowTip

Description

Show the tool tip in the *Member* tab.

Syntax

```
srcTBTBMVShowTip
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVShowTip
```

srcTBTBOBAddObjToWave

Description

Add the specified object to the *nWave* window.

Syntax

```
srcTBTBOBAddObjToWave
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBAddObjToWave
```

srcTBTBOBAddReference

Description

Add the selected object reference to the specified *Watch* tab.

Syntax

```
srcTBTBOBAddReference -tab tabId
```

Arguments

`-tab` *`tabId`*

Specify the *Watch* tab to add the object reference to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBAddReference -tab 1
```

srcTBTBOBAddWatch

Description

Add the selected object to the *Watch* tab.

Syntax

```
srcTBTBOBAddWatch -tab tabId
```

Arguments

`-tab` *`tabId`*

Specify the *Watch* tab to add the object to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBAddWatch
```

srcTBTBOBCollapseAll

Description

Collapse the selected tree.

Syntax

```
srcTBTBOBCollapseAll
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBCollapseAll
```

srcTBTBOBDisplaySourceCode

Description

Display the source code of the selected object.

Syntax

```
srcTBTBOBDisplaySourceCode
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBDisplaySourceCode
```

srcTBTBOBDumpObjToFSDB

Description

Dump the selected object to the FSDB file.

Syntax

```
srcTBTBOBDumpObjToFSDB
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBDumpObjToFSDB
```

srcTBTBOBExpandTreeByLevelAll

Description

Expand the hierarchy tree by levels.

Syntax

```
srcTBTBOBExpandTreeByLevelAll -level all|2|3|4|5
```

Arguments

`-level all|2|3|4|5`

Specify the number of levels to be expanded.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBExpandTreeByLevelAll -level "all"
```

srcTBTBOBShowCreation

Description

Show the create location of the selected object.

Syntax

```
srcTBTBOBShowCreation
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBShowCreation
```

srcTBTBOBShowInClassBrowser

Description

Show the related class of the select item in the *Class* tab.

Syntax

```
srcTBTBOBShowInClassBrowser
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBShowInClassBrowser
```

srcTBTBOBShowInfo

Description

Show the information of the selected file in the *Object* tab.

Syntax

```
srcTBTBMVShowInfo
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBMVShowInfo
```

srcTBTBOBShowMemory

Description

Show the memory of the selected object in the *Object* tab.

Syntax

```
srcTBTBOBShowMemory
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBShowMemory
```

srcTBTBOBShowNavigate

Description

Enable the **Navigation** text field in the *Object* tab.

Syntax

```
srcTBTBOBShowNavigate
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBShowNavigate
```

srcTBTBOBShowReference

Description

Show the reference of the selected object.

Syntax

```
srcTBTBOBShowReference
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBShowReference
```

srcTBTBOBShowTip

Description

Show the tool tip in the *Object* tab.

Syntax

```
srcTBTBOBShowTip
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTBTBOBShowTip
```

Power Manager

debLoadPDML

Description

Load CPF/UPF files.

Syntax

```
debLoadPDML -upf|-upf1.0|-upf2.0|-cpf1.0|-cpf1.1|-cpf2.0 files
[-pdmlPath path] [-upfTop top] [-upf_ground_logic_value_is_1
<0|1>|<yes|no>] [-domainColor configFile] [-powerLib libFileName]
[-powerModel NLP|MVSim|ModelSim|NCSim] [-power=options] [-lp_xml
xmlFileName]
```

Arguments

`-domainColor` *`configFile`*

Specify the configuration file to set the colors of the specified power domains.

`-lp_xml` *`xmlFileName`*

Specify the name of the low-power XML file to import. Refer to the **-lp_xml** option in the *verdi* utility for details.

`-pdmlPath` *`path`*

Specify the working path of the loaded files.

`-power=`*`options`*

The following power compile-time options are supported:

NOTE: The plus (+) symbol can be used to specify multiple options.

=auto_complete supports automatic completion of UPF commands. By default, the automatic completion for UPF commands and command options is disabled. It is recommended to correct the UPF command or command option instead of using **-power=auto_complete**.

=accurate ignores the impact of isolation-related commands.

=apfcompat skips corruption of reals, constants, and signals assigned only in initial blocks.

=dccompat_gen_separator supports using dots as the separator of Verilog generate blocks.

=nofeedthrough disables treating the simple continuous assigns as feedthrough paths.

`-powerLib` *`libFileName`*

Specify the power library file to be loaded.

`-powerModel NLP|MVSim|ModelSim|NCSim`

Specify the simulator when loading the power design. The default is **NLP**.

`-upf_ground_logic_value_is_1 <0|1> | <yes|no>`

Specify the value to treat ground as "ON is 1". When 1 or yes is specified, 1 is considered as ON for ground. When 0 or no is specified, 0 is considered as ON for ground. The default value is *1|yes*.

`-upf|-upf1.0|-upf2.0|-cpf1.0|-cpf1.1|-cpf2.0` *`files`*

Specify the UPF/CPF files to be loaded.

`-upfTop` *`top`*

Specify the top level design instance for UPF files.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
debLoadPDML -pdmlPath /verdi/home/power/cpf_demo -cpf1.1
debLoadPDML -upfTop top  -upf2.0 ./compositePD.upf -domainColor
1.cfg
debLoadPDML -powerLib lib

debLoadPDML -power=accurate+apfcompat

debLoadPDML -lp_xml top.xml -upf2.0 top.upf
```

paImpactFind

Description

Find a specified string in the *Report Power Impacted Signals* form.

Syntax

`paImpactFind "`*`string`*`" -next|-prev`

Arguments

`-next|-prev`

Find the next or previously matched string.

`"string"`

Specify the string to be found in the *Report Power Impacted Signals* form.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
paImpactFind "cpu" -prev
```

paImpactReport

Description

Open the *Report Power Impacted Signals* form and report all the rules in the list.

Syntax

```
paImpactReport [-save filename] [-close] [-showFullName on|off]
```

Arguments

`-close`

Close the *Report Power Impacted Signals* form.

`-save filename`

Save the rules in the *Report Power Impacted Signals* form to the specified text file.

`-showFullName on|off`

Specify whether to show the full name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
paImpactReport -close
paImpactReport -showFullName on
```

paImpactScope

Description

Specify scopes for which to report the power impacted signals.

Syntax

```
paImpactScope [-select nRow1 nRow2 nRow3 …] [-delSelect] [-delAll]
[-addScope] [-delim “strDelim”] [-case on|off] [-incSubscope
on|off] [-drag] [-drop]
```

Arguments

`-addScope` *`scope_full_hier`*

Add the selected scope to the scope list.

`-case on|off`

Turn the case matching for scope search *on* or *off*. The default is *on*.

`-drag`

Drag a scope to the scope list.

`-delAll`

Delete all scopes in the scope list.

`-delim “strDelim”`

Specify the delimiter for scopes. The default is the period (.) character.

`-delSelect`

Delete the selected scope in the scope list.

`-drop`

Drop the selected scope to the scope list.

`-incSubscope on|off`

Specify whether sub-scopes of the selected scope are listed in the scope list. The default is *off*.

`-select` *`nRow1 nRow2 nRow3 …`*

Select the scope in the scope list.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
paImpactScope -select 0 1 2 3
paImpactScope -delSelect
paImpactScope -delAll
paImpactScope –addScope “system.*”
paImpactScope -delim "."
paImpactScope -case on
paImpactScope -incSubscope on
paImpactScope -drag
paImpactScope -drop
```

paImpactSelect

Description

Select the cell(s) in the *Report Power Impacted Signals* form.

Syntax

```
paImpactSelect -row RowNumber -col ColumnNumber
```

Arguments

`-col ColumnNumber`

Specify the column number of the selected cell in the *Report Power Impacted Signals* form.

`-row RowNumber`

Specify the row number of the selected cell in the *Report Power Impacted Signals* form.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
paImpactSelect -row 2 -col 4
```

paImpactSetActiveTab

Description

Set the specified tab as the active tab in the *Report Power Impacted Signals* form.

Syntax

```
paImpactSetActiveTab -tab ImpactedSignalTab|ControlSignalTab
```

Arguments

`-tab ImpactedSignalTab|ControlSignalTab`

Set the Impacted Signals tab or Control Signals tab as the active tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
paImpactSetActiveTab -tab ControlSignalTab
```

paImpactSort

Description

Sort the columns in the *Report Power Impacted Signals* form.

Syntax

```
paImpactSort [-rule|-signal|-fromPD|-fromInst|-toPD|-toInst|
-isoCon|-savCon|-resCon] [-asc|-des]
```

Arguments

`-asc|-des`

Sort the columns in the *Report Power Impacted Signals* form in ascending or descending order.

`-fromInst`

Sort the From Instance column in the Impacted Signals tab.

`-fromPD`

Sort the From Domain column in the Impacted Signals tab.

`-isoCon`

Sort the Isolation Condition column in the Control Signals tab.

`-resCon`

Sort the Restore Condition column in the Control Signals tab.

`-rule`

Sort the Rule column.

`-savCon`

Sort the Save Condition column in the Control Signals tab.

`-signal`

Sort the Signal column in the Impacted Signals tab.

`-toInst`

Sort the To Instance column in the Impacted Signals tab.

`-toPD`

Sort the To Domain column in the Impacted Signals tab.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
paImpactSort -rule -asc
paImpactSort -toPD -des
```

paSetDisplayAttr

Description

Set the color for the selected power aware object.

Syntax

```
paSetDisplayAttr [-default] [PowerDownInstance -color colorName]
[RetentionSignal -color colorName] [IsolationSignal -color
colorName] [LevelShiftedSignal -color colorName] [PowerNet -color
colorName] [GroundNet -color colorName] [SRSNObject -color
colorName]
```

Arguments

`-default`

Set the color for all power aware objects as the default.

`GroundNet -color colorName`

Specify the color of ground nets.

`IsolationSignal -color colorName`

Specify the color of isolation objects.

`LevelShiftedSignal -color` *`colorName`*

Specify the color of level shifted objects.

`PowerDownInstance -color` *`colorName`*

Specify the color of power down instances.

`PowerNet -color` *`colorName`*

Specify the color of power nets.

`RetentionSignal -color` *`colorName`*

Specify the color of retention objects.

`SRSNObject -color` *`colorName`*

Specify the color of SRSN signals.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
paSetDisplayAttr IsolationSignal -color ID_RED4
paSetDisplayAttr -default
paSetDisplayAttr IsolationSignal RetentionSignal
LevelShiftedSignal -color ID_RED4
```

paSetPreference

Description

Set the modes for the layout of the invoking *Power Manager* window.

Syntax

```
paSetPreference [-AnnotateSignal on|off] [-highlightPowerObject
on|off] [-highlightPowerDomain on|off] [-extractIsoSameNet on|off]
[-showVoltage on|off] [-matchPDNodeCaseInsensitive on|off]
[-displayInstrumentedCell on|0ff] [-showAlias on|off]
[-searchHBNodeDynamically on|off] [-showPstAnnot on|off]
[-showSymbol on|off]
```

Arguments

`-AnnotateSignal on|off`

When this option is turned *on*, the power aware design objects are annotated in the *Source Code* pane.

`-displayInstrumentedCell on|0ff`

When this option is turned *on*, isolation cells are displayed in flattened schematic windows.

`-extractIsoSameNet on|off`

When this option is turned *on*, the same nets which are derived from an Isolation signal is marked as ISO in *nTrace*, *nWave*, *Power Manager*, *Temporal Flow View*, or *nSchema*.

`-highlightPowerDomain on|off`

When this option is turned *on*, the power domains are highlighted with specific colors.

`-highlightPowerObject on|off`

When this option is turned *on*, the power aware design objects are highlighted with specific colors.

`-matchPDNodeCaseInsensitive on|off`

When this option is turned *on*, searching or filtering tree nodes is not case-sensitive.

`-searchHBNodeDynamically on|off`

When this option is turned *on*, tree nodes are dynamically searched as the search string is entered.

`-showAlias on|off`

When this option is turned *on*, the nominal condition of the port state name for CPF or power domains for UPF are displayed with the power domain annotation in *nTrace*, *nSchema*, *Temporal Flow View*, or *Power Manager*.

`-showPstAnnot on|off`

When this option is turned *on*, annotation for states in the **add_pst_state** command is displayed. When this option is turned *off*, annotation for states in the **add_pst_state** command is not displayed. The default is *off*.

`-showVoltage on|off`

When this option is turned *on*, the voltage for power domains are displayed with the power domain annotation in *nTrace*, *nSchema*, *Temporal Flow View*, or *Power Manager*.

`-showSymbol on|off`

When this option is turned *on*, the isolation, SRSN, or SPA annotation is displayed as a symbol.

When this option is turned *off*, the isolation, SRSN, or SPA annotation is displayed as a string.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
paSetPreference -showVoltage on
paSetPreference -showSymbol on
paSetPreference -showAlias off
paSetPreference -extractIsoSameNet off
paSetPreference -displayInstrumentedCell on
```

pdmlAddAppliedSigToWave

Description

Add the applied signals in the *Power Manager* window to *nWave*.

Syntax

```
pdmlAddAppliedSigToWave -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlAddAppliedSigToWave -win $_nPDMLCodeBrowser3
```

pdmlBackwardHistory

Description

Backward trace the history, and change the view accordingly.

Syntax

```
pdmlBackwardHistory -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlBackwardHistory -win $_nPDMLCodeBrowser2
```

pdmlCloseWindow

Description

Close the *Power Manager* window.

Syntax

```
ddCloseWindow -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlCloseWindow -win $_nPDMLCodeBrowser3
```

pdmlCollapseNode

Description

Collapse nodes in the *Power Domain Details* pane of the *Power Manager* window.

Syntax

```
pdmlCollapseNode -win window -row rowNumber
```

Arguments

`-row rowNumber`

The position of the signal to be collapsed.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlCollapseNode -win $_nPDMLCodeBrowser3 -row 1
```

pdmlCreateWindow

Description

Open the *Power Manager* window.

Syntax

```
pdmlCreateWindow
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlCreateWindow
```

pdmlExpandNode

Description

Expand nodes in the *Power Domain Details* pane of the *Power Manager* window.

Syntax

```
pdmlExpandNode -win window -row rowNumber
```

Arguments

`-row rowNumber`

The position of the signal to be expanded.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlExpandNode -win $_nPDMLCodeBrowser3 -row 1
```

pdmlFind

Description

Find strings/signals/scopes/domains/power nets in the *Power Manager* window.

Syntax

```
pdmlFind "patternName" -win window [-string|-signal|-scope|
-powerDomain|-powerNet] [-inCurDomain|-inAllDomains]
-case [-next|-prev] [-hideForm]
```

Arguments

`-case`

Perform case-sensitive searching.

`-hideForm`

Close the *PDML -Find* form.

`-next|-prev`
Find the next or previously matched pattern.

"patternName"
Specify the pattern to be found in the *Power Manager* window.

`-string|-signal|-scope|-powerDomain|-powerNet`
Specify the type to be found in the *Power Manager* window.

`-win` *window*
Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlFind "CHILD" -win $_nPDMLCodeBrowser4 -case -next
```

```
pdmlFind "PD" -win $_nPDMLCodeBrowser3 -case -signal -inCurDomain
```

pdmlForwardHistory

Description

Forward trace the history, and change the view accordingly.

Syntax

```
pdmlForwardHistory -win window
```

Arguments

`-win` *window*
Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlForwardHistory -win $_nPDMLCodeBrowser2
```

pdmlGetPDInfo

Description

Search for the full hierarchy names of the related power domain with the full hierarchy name of the object.

Syntax

```
pdmlGetPDInfo [-inst instFullHier] [-signal signalFullHier]
[-delim strDelimiter] [-topPort portFullHier]
[–instPin pinFullHier] [–module moduleName] [–output filename]
```

Arguments

`-delim strDelimiter`

Specify the delimiter of the hierarchy names of the signal or instance. The default is ".".

`-inst instFullHier`

Specify the instance with the full hierarchy name.

`–instPin pinFullHier`

Specify the instance pin with the full hierarchy name.

`–module moduleName`

Specify the module name.

`–output filename`

Specify the file to output the returned power domain information to. The format of the output is: {{port_full_hier:powerDomain1}, {port_full_hier:NF},{port_full_hier:ERR}…}

`-signal signalFullHier`

Specify the signal of the full hierarchy name.

`-topPort portFullHier`

Specify the top level port with the full hierarchy name.

Value Returned

The full hierarchy names of the related power domain. The format of the value is: {{powerDomain1} {powerDomain2} {powerDomain3}...}.

Example

```
pdmlGetPDInfo -inst {top10.sys1} {top10.sys2} -signal
```

```
{top10.sys10.i_cpu.i_PCU.TDB[6]} {top10.sys10.i_cpu.i_PCU.TDB[7]}
-delim "."
```

This command returns the following power domains:

```
{ {PD_sys1} {PD_sys2} {PD_PCU} {PD_PCU} }
```

pdmlHBCollapsedAll

Description

Collapse all tree nodes in the *Power Domain List* frame.

Syntax

```
pdmlHBCollapsedAll -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlHBCollapsedAll -win $_nPDMLCodeBrowser2
```

pdmlHBExpandAll

Description

Expand all tree nodes in the *Power Domain List* frame.

Syntax

```
pdmlHBExpandAll -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlHBExpandAll -win $_nPDMLCodeBrowser2
```

pdmlHBSetFilter

Description

Filter out the tree nodes in the power domain list.

Syntax

```
pdmlHBSetFilter [stringPattern] –case on|off
–type [Domain|Rule|Net|Inst|Signal|Other|All]
```

Arguments

`-case on|off`

Turn the case matching for attributes comparison *on* or *off*. The default is *on*.

`stringPattern`

Specify the search string pattern.

`-type`

Specify the tree node type. The types include: **Domain**, **Rule**, **Net**, **Inst,** **Signal**, **Other**, and **All**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlHBSetFilter “PD_*” –case on -type Domain Rule
```

pdmlListHDLSignal

Description

List all HDL signals used in the CPF/UPF file in the *List HDL Signals* form.

Syntax

```
pdmlListHDLSignal -win window [-save filename]
```

Arguments

`-save filename`

Save the HDL signals in the *List HDL Signals* form to the specified text file.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlListHDLSignal -win $_nPDMLCodeBrowser2
pdmlListHDLSignal -win $_nPDMLCodeBrowser3 -save /verdi/power/
HDLSig.txt
```

pdmlPstFind

Description

Find a string in the power state table.

Syntax

```
pdmlPstFind "string" -win window -next|-prev
```

Arguments

`-next|-prev`

Find the next or previously matched string.

`"string"`

Specify the string to be found in the power state table.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlPstFind "PD" -win $_nPDMLCodeBrowser3 -next
```

pdmlPstSaveAsCSV

Description

Save the power state table to the specified file in CSV file format.

Syntax

```
pdmlPstSaveAsCSV -f filename -openForm -hideForm
```

Arguments

`-f filename`

Specify the file name to save.

`-hideForm`

Hide the *Save As CSV* form.

`-openForm`

Open a form to specify the file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlPstSaveAsCSV -f /mydir/curPst.csv
pdmlPstSaveAsCSV -hideForm
```

pdmlPstSelect

Description

Select the power domain, power mode, power state, condition title or condition expression cell in the power state table.

Syntax

```
pdmlPstSelect [-pst pstName] [-pd pdName|-pm pmName]
[–column col]
```

Arguments

`–column col`

Specify the column of the selected cell in the power state table row or in the power mode row.

`-pd pdName`

Specify the power domain name.

`-pm pmName`

Specify the power mode name.

`-pst pstName`

Specify the power state table.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlPstSelect -pst system.power_mode_table -pd system.PD_TOP
pdmlPstSelect –pst top.generic_mode_table –pm Test3 –column 2
pdmlPstSelect –pst top.generic_mode_table –column 1
```

NOTE: You must set the delimiter for a power design. In the same power design, the unified delimiter should be used.

pdmlPstToggle

Description

Expand or collapse the power state table.

Syntax

```
pdmlPstToggle -win window [-expand|-collect] [-pst pstEncName]
```

Arguments

`-expand|-collect`

Toggle to expand or collect the power state table.

`-pst pstEncName`

Specify the power state table to be expanded or be collapsed. If not specified, the whole power state table is expanded or collapsed.

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlPstToggle -win $_nPDMLCodeBrowser3 -collect
```

```
pdmlPstToggle -win $_nPDMLCodeBrowser3 -expand -pst Top.inst.pst
```

pdmlReload

Description

Reload the design in the *Power Manager* window.

Syntax

`pdmlReload -win` *`window`*

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlReload -win $_nPDMLCodeBrowser4
```

pdmlSeqCheck

Description

Invoke the *Check Power Sequence* form.

Syntax

```
pdmlSeqCheck [-rule ruleId | -all | -all_iso | -all_pd]
[-maxError maxErrorCount] [-save filename] [-restore filename]
[-select ruleId rowNum] -close [-all_pmt] [-pdList "power domain
list"] [-by fromTime] [-et toTime] [-reportByRule on|off] [-
searchNext|-searchPrev "string"]
```

Arguments

`-all`

Enable all rules.

`-all_pmt`

Specify all the checking rules related to the Power Mode Table.

`-all_iso`

Only enable isolation related rules.

`-all_pd`

Only enable power domain related rules.

`-bt fromTime`

Specify the begin time to check the power sequence.

`-close`

Close the *Check Power Sequence* form.

`-et toTime`

Specify the end time to check the power sequence.

`-maxError maxErrorCount`

Specify the maximum number of error report.

`-pdList "power domain list"`

Specify the power domain to check the power sequence. The default is all power domains.

`-reportByRule on|off`

Specify whether to show report by rule or by hierarchy. The default is *on* (that is, by rule).

`-restore filename`

Specify the file to restore the report from.

`-rule ruleId`

Specify the rules to check. The `ruleId` corresponds to the rule name as described in the Power Rules Table.

`-save filename`

Specify the file to save the report to.

`-searchNext|-searchPrev "string"`

Search the next/previous matched item in the report.

`-select ruleId rowNum`

Select the report line with the specified rule and row number in the *Check Power Sequence* form. The `ruleId` corresponds to the rule name as described in the Power Rules Table.

Power Rules Table

Rule Number (ruleId)	Rule Name
100	Isolation Control Signal has X
101	Isolation Control Signal not On when Power Domain Status Off
102	Isolation On but Signal has no Clamp Value
103	Isolation Control never Enabled
200	Power Domain Status Off but Signal Value Change not Corrupt
201	Power Domain Status has X
202	Power Domain Status On but no Signal has Value Change
203	Power Domain Status always On
204	Power Domain Status always Off
300	Power Mode Table has invalid mode
301	Power Mode Table has uncovered mode

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
pdmlSeqCheck
pdmlSeqCheck -rule 101 102 201
pdmlSeqCheck -all_iso -maxError 100
pdmlSeqCheck -save output.rpt
pdmlSeqCheck -restore restore.rpt
pdmlSeqCheck -select 100 1
pdmlSeqCheck -all_pmt -maxError 1000
pdmlSeqCheck -bt 100 -et 200
```

```
pdmlSeqCheck -pdList "TOP.PD1" "TOP.PD2"
pdmlSeqCheck -reportByRule off
pdmlSeqCheck -searchNext "*iso_sig*"
pdmlSeqCheck -searchPrev "*iso_sig*"
```

pdmlSetOption

Description

Show or hide the power state table.

Syntax

```
pdmlSetOption -win window -showPst on|off
```

Arguments

`-showPst on|off`

Show or hide the power state table in the *Power Manager* window by turning this option *on* or *off*.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlSetOption -win $_nPDMLCodeBrowser4 -showPst off
```

pdmlSetPowerDomain

Description

Set the selected power domain as the active power domain.

Syntax

```
pdmlSetPowerDomain "powerDomain" -win window
```

Arguments

"*powerDomain*"

Specify the power domain name.

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlSetPowerDomain "system.PD_FSM" -win $_nPDMLCodeBrowser3
```

pdmlSetViewOption

Description

Show or hide the power state table.

Syntax

```
pdmlSetViewOption -win window [-fullDomainName on|off]
[-showExtent on|off] [-showShutOffCondition on|off]
[-showIsolation on|off] [-showRetention on|off]
[-showLevelShifter on|off] [-showPowerNet on|off] [-showGroundNet
on|off] [-showSecondaryDomain on|off] [-showMappingDomain on|off]
[-showBoundary on|off][-showPst on|off][-showSupplySet on|off]
```

Arguments

`-fullDomainName on|off`

Show or hide the full domain name in the *Power Manager* window by turning this option *on* or *off*.

`-showBoundary on|off`

Show or hide the boundary port signals under the "Extent" statement by turning this option *on* or *off*.

`-showExtent on|off`

Show or hide the extent signals in the *Power Manager* window by turning this option *on* or *off*.

`-showGroundNet on|off`

Show or hide the ground net in the *Power Manager* window by turning this option *on* or *off*.

`-showIsolation on|off`

Show or hide the isolation signals in the *Power Manager* window by turning this option *on* or *off*.

`-showLevelShifter on|off`

Show or hide the level shifted signals in the *Power Manager* window by turning this option *on* or *off*.

`-showPowerNet on|off`

Show or hide the power net in the *Power Manager* window by turning this option *on* or *off*.

`-showMappingDomain on|off`

Show or hide the mapping domain in the *Power Manager* window by turning this option *on* or *off*. This option is only valid for CPF 1.1.

`-showRetention on|off`

Show or hide the retention signals in the *Power Manager* window by turning this option *on* or *off*.

`-showSecondaryDomain on|off`

Show or hide the secondary domain in the *Power Manager* window by turning this option *on* or *off*. This option is only valid for CPF 1.1.

`-showShutOffCondition on|off`

Show or hide the shut off condition in the *Power Manager* window by turning this option *on* or *off*.

`-showPst on|off`

Show or hide the power state table in the *Power Domain Hierarchy* by turning this option *on* or *off*. The default state is *on*. This option is only valid for UPF 1.0.

`-showSupplySet on|off`

Show or hide the supply set in the *Power Manager* window by turning this option *on* or *off*. This option is only valid for UPF 2.0.

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlSetViewOption -win $_nPDMLCodeBrowser4 -fullDomainName on
```

pdmlSetWinMode

Description

Set the modes for the layout of the *Power Manager* window.

Syntax

```
pdmlSetWinMode -win window -mode pstMode|normalMode
```

Arguments

`-mode pstMode|normalMode`

Set the window mode as Power State mode or Normal mode.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlSetWinMode -win $_nPDMLCodeBrowser3 -mode pstMode
```

pdmlShowAnnotation

Description

Display active annotation in the *Power Manager* window.

Syntax

```
pdmlShowAnnotation -win window -annot on|off
```

Arguments

`-annot on|off`

Set the annotation *on* or *off*.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlShowAnnotation -win $_nPDMLCodeBrowser4 -annot on
```

pdmlShowDefinition

Description

Show the definition of the selected signal in the *Power Manager* window.

Syntax

```
pdmlShowDefinition -win window -fileName fileName -lineNo
-powerDomain
```

Arguments

`-fileName` *`fileName`*

Specify the file name.

`-lineNo`

Specify the line number in the *Source Code* pane.

`-powerDomain`

Specify the power domain name.

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlShowDefinition "PD_ALUB" -win $_nPDMLCodeBrowser3 -fileName \
/verdi/power/cpf/demo.cpf -lineNo 28 -powerDomain system.PD_ALUB
```

pdmlShowTraceUnknownReasonCMD

Description

Highlight all conflicting conditions that can cause the power state as error or unknown in the *Source Code* pane and show the reason in the *Message* pane.

Syntax

```
pdmlShowTraceUnknownReasonCMD -powerDomain domainName
-file strFile -lineNo lineNumber -reason reason_-expr
{strExpr1,strExpr2...} -time strTime
```

Arguments

`-expr`

Show conflicting expressions.

`-file strFile`

Specify the file name.

`-lineNo lineNumber`

Specify the line number.

`-powerDomain domainName`

The encoded power domain name.

`-reason reason`

Show the unknown reason.

`-time strTime`

Specify the cursor time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlShowTraceUnknownReasonCMD -powerDomain top.PD_cpu \
                              -file "abc.cpf" \
                              -lineNo  10 \
                              -reason CONFLICT \
                              -expr "sigA and sigB are on"
                              -time 100
```

pdmlSrcAddSigToWave

Description

Add the selected power object(s) to the *nWave* window. The power objects include the power domain, supply net, supply port, supply set, and power state table.

Syntax

```
pdmlSrcAddSigToWave
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlSrcAddSigToWave
```

pdmlTraceActiveDriver

Description

Locate the active driver in the *Power Source Code* frame for the selected power signal.

Syntax

```
pdmlTraceActiveDriver “signalName” -win window
```

Arguments

“signalName”

Specify the signal name.

`-win` *window*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlTraceActiveDriver "TOP.L1.clk" -win $_nPDMLCodeBrowser2
```

pdmlTraceConnectivity

Description

Give the results of both the Trace Power Driver and Trace Power Load commands with a single command.

Syntax

```
pdmlTraceConnectivity “signalName” -win window
```

Arguments

“signalName”

Specify the signal name.

`-win` *window*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlTraceConnectivity "TOP.L1.clk" -win $_nPDMLCodeBrowser2
```

pdmlTraceDriver

Description

Trace all of the commands which impact the HDL signal.

Syntax

```
pdmlTraceDriver “signalName” -win window
```

Arguments

“signalName”

Specify the signal name.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlTraceDriver "TOP.L1.SP" -win $_nPDMLCodeBrowser2
```

pdmlTraceLoad

Description

Trace all of the commands which are impacted by the HDL signal.

Syntax

```
pdmlTraceLoad “signalName” -win window
```

Arguments

“signalName”

Specify the signal name.

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlTraceLoad "TOP.L1.LP" -win $_nPDMLCodeBrowser2
```

pdmlTracePowerConnectivity

Description

Trace the power connectivity in the *Power Source Code* frame for the specified object.

Syntax

```
pdmlTracePowerConnectivity [-powerSupplySet | -powerPort |
-powerNet | -powerDomain | -powerSwitch] objectName
```

Arguments

`-powerSupplySet`

Trace the power driver for the power supply set object.

`-powerPort`

Trace the power driver for the power port object.

`-powerNet`

Trace the power driver for the power net object.

`-powerDomain`

Trace the power driver for the power domain object.

`-powerSwitch`

Trace the power driver for the power switch object.

`objectName`

Specify the object with the full hierarchical name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlTracePowerConnectivity –powerNet TB/vdd
```

pdmlTracePowerDriver

Description

Trace the power driver in the *Power Source Code* frame for the specified object.

Syntax

```
pdmlTracePowerDriver [-powerSupplySet | -powerPort | -powerNet |
-powerDomain | -powerSwitch] objectName
```

Arguments

`-powerSupplySet`

Trace the power driver for the power supply set object.

`-powerPort`
Trace the power driver for the power port object.

`-powerNet`
Trace the power driver for the power net object.

`-powerDomain`
Trace the power driver for the power domain object.

`-powerSwitch`
Trace the power driver for the power switch object.

`objectName`
Specify the object with the full hierarchical name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlTracePowerDriver -powerNet TB/vdd
```

pdmlTracePowerLoad

Description

Trace the power load in the *Power Source Code* frame for the specified object.

Syntax

```
pdmlTracePowerLoad [-powerSupplySet | -powerPort | -powerNet |
-powerDomain | -powerSwitch] objectName
```

Arguments

`-powerSupplySet`
Trace the power driver for the power supply set object.

`-powerPort`
Trace the power driver for the power port object.

`-powerNet`
Trace the power driver for the power net object.

`-powerDomain`
Trace the power driver for the power domain object.

`-powerSwitch`
Trace the power driver for the power switch object.

`objectName`
Specify the object with the full hierarchical name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlTracePowerLoad –powerNet TB/vdd
```

pdmlTracePowerTransition

Description

Trace the power transition in the *Power Source Code* frame for the specified power domain object.

Syntax

```
pdmlTracePowerTransition -powerDomain domainName -time timeString
```

Arguments

`-powerDomain` *`domainName`*
Specify the power domain.

`-time` *`timeString`*
Specify the starting time to trace the power transition.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlTracePowerTransition -powerDomain Island_Vdd -time 0
```

schCreateWindow

Description

Create the *Partial Power Map* window.

Syntax

```
schCreateWindow -win $_nSchema1 [-partialflatpdml]
[-PowerCellCoverInst instanceName] [-PowerCellCoverNet netName
domainName] [-PowerDomain domainName] [-PowerNonCoverNet netName
fromDomain toDomain] [-PowerRule ruleName] [-PowerRuleCoverInst
instanceName ruleName] [-PowerRuleCoverNet netName ruleName]
[-PowerSupplyNet netName] [-PowerSwitch switchName]
```

Arguments

`-partialflatpdml`

Create the *Partial Power Map* window.

`-PowerCellCoverInst` *instanceName*

Specify the power cell cover instance.

`-PowerCellCoverNet` *netName domainName*

Specify the power cell cover net.

`-PowerDomain` *domainName*

Specify the power domain object.

`-PowerNonCoverNet` *netName fromDomain toDomain*

Specify the non-covered net.

`-PowerRule` *ruleName*

Specify the power rule object.

`-PowerRuleCoverInst` *instanceName ruleName*

Specify the power rule cover instance.

`-PowerRuleCoverNet` *netName ruleName*

Specify the power rule cover net.

`-PowerSupplyNet` *netName*

Specify the power supply net object.

`-PowerSwitch` *switchName*

Specify the power switch object.

`-win` *window*

Specify the window ID of the invoking *nSchema* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schCreateWindow -win $_nSchema1 -partialflatpdml
–PowerDomain "PD_MEM" i_cpu/PD_CPU" -PowerSwitch "SW_MEM"
"SW_PCU" -PowerSupplyNet VDD" "VDD_MEM"
-PowerRuleCoverInst "LEV:i_cpu/PD_ALUB:i_cpu/ls_ALUB_OUT" "i_cpu/
ls_ALUB_OUT"

schCreateWindow -win $_nSchema1 –partialflatpdml
–PowerRuleCoverNet i_cpu/PD_PCU:i_cpu/PD_ALUB:i_cpu/ls_PCU_OUT"
"i_cpu/ls_PCU_OUT"

schCreateWindow -win $_nSchema1 -partialflatpdml
-PowerRule "i_cpu/iso_ALUB"

schCreateWindow -win $_nSchema1 -partialflatpdml
–PowerNonCoverNet "NON_COVER_ISO:i_cpu/PD_CCU:i_cpu/PD_ALUB"
"i_cpu/PD_CCU" i_cpu/PD_ALUB"
```

schPdmlAddViewObj

Description

Add the view object to the *Partial Power Map* window.

Syntax

```
schCreateWindow -win window [-PowerDomain domainName]
[-PowerRule ruleName] [-PowerSupplyNet netName] [-PowerSwitch
switchName]
```

Arguments

`-PowerDomain` *`domainName`*

Specify the power domain object.

`-PowerRule` *`ruleName`*

Specify the power rule object.

`-PowerSupplyNet` *`netName`*

Specify the power supply net object.

`-PowerSwitch` *`switchName`*

Specify the power switch object.

`-win` *`window`*

Specify the window ID of the invoking *Partial Power Map* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlAddViewObj -win $_nPowerMap19 -PowerDomain "i_cpu/PD_CCU"
"i_cpu/PD_PCU"
schPdmlAddViewObj -win $_nPowerMap19 -PowerRule "i_cpu/
ls_ALUB_OUT" "i_cpu/ls_PCU_OUT"
schPdmlAddViewObj -win $_nPowerMap19 -PowerSwitch "SW_CPU"
schPdmlAddViewObj -win $_nPowerMap19 -PowerSupplyNet "i_cpu/
iso_PCU" -PowerSwitch "SW_ALUB"
```

schPdmlChangeDisplayAttr

Description

Change the display attributes for the specified objects in the *Power Map* frame.

Syntax

```
schPdmlChangeDisplayAttr [-win window] [-color colorId] [-default]
[-defaultAll]
```

Arguments

`-color` *`colorId`*

Specify the color for the signals or instances.

`-default`

Reset the specified display attributes to the default settings.

`-defaultAll`

Reset all display attributes to the default settings.

`-win` *`window`*

Specify the window ID of the invoking *Power Map* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlChangeDisplayAttr –win $_nPowerMap3 –color ID_RED4
```

schPdmlImpactSigReport

NOTE: This Tcl command is activated after the *PowerMap* window is opened by the **-flatpdml** option of the **schCreateWindow** command.

Description

Search for specified signals in the *nPowerMap - Report Power Impacted Signals* form.

Syntax

```
schPdmlImpactSigReport -win window -search -next|-prev
-all|-rule|-signal|-fromInst|-fromPD|-toInst|-toPD|-value
"searchString"
```

Arguments

`-all`

Search among all types.

`-fromInst`

Search among from instances.

`-fromPD`

Search among from power domains.

`-next`

Find the next string matching the specification.

`-prev`

Find the previous string matching the specification.

`-rule`

Search among rule signals.

`-search "searchString"`

Search the specified string in the *nPowerMap -Report Power Impacted Signals* form.

`-signal`

Search among signals.

`-tab ControlSignalTab|ImpactedSignalTab`
Specify the **Impacted Signals** tab or the **Control Signals** tab.

`-toInst`
Search among to instances.

`-toPD`
Search among to power domains.

`-value`
Search for signal value.

`-win window`
Specify the window ID of the invoking *Power Map* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlImpactSigReport -win $_nPowerMap3 -search -next -rule
"iso"

schPdmlImpactSigReport -win $_nPowerMap3 -tab ControlSignalTab
```

schPdmlRedo

Description

Redo previous undone action.

Syntax

```
schPdmlRedo –win window
```

Arguments

`-win window`
Specify the window ID of the invoking *Power Map* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlRedo -win $_nPowerMap19
```

schPdmlSaveFindResult

Description

Save the searching results of the specified window to a file.

Syntax

```
schPdmlSaveFindResult -win window [-filter string] [-CaseMatch]
-PowerDomain|-PowerRule|-LibPowerCell|-SupplyNet [-select string]
-file filename
```

Arguments

`-CaseMatch`

Perform case-sensitive searching.

`-file filename`

Specify the file name with full path to save the search results to.

`-filter string`

Specify the searching string. The wildcard character * is supported. This option is optional.

`-LibPowerCell`

Search all library power cells.

`-PowerDomain`

Search all power domains.

`-PowerRule`

Search all power rule commands.

`-select string`

Save the selected string to the file.

`-SupplyNet`

Search all supply nets.

`-win window`

Specify the window ID of the window where the searching results are located.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlSaveFindResult -win $_nPowerMap3 -filter "*input"
-PowerDomain -file ./save.txt
```

schPdmlSelect

Description

Select the object in the *Power Map* window.

Syntax

```
schPdmlSelect -win window [-toggle] [-signal signalName |
-inst instanceName | -instpin instpinname]
```

Arguments

`-instpin instpinname`

Specify the instance pin.

`-inst instanceName`

Specify the instance.

`-signal signalName`

Specify the signal.

`-toggle`

Keep the previously selected object.

`-win window`

Specify the window ID of the invoking *Power Map* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlSelect -win $_nPowerMap7 -inst "ISO:PD_PCU:iso_PCU"
```

schPdmlSetOptions

Description

Set viewing options for the *Power Map* window.

Syntax

```
schPdmlSetOptions [-win window] [-CellCoverCheck on|off]
[-CheckCover on|off]
```

Arguments

`-CellCoverCheck on|off`

Turn the checking and display of cell-covered signals *on* or *off*.

`-CheckCover on|off`

Turn the checking and display of rule-covered signals *on* or *off*.

`-win window`

Specify the window ID of the invoking *Power Map* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlSetOptions -win $_nPowerMap3 -CellCoverCheck on
```

schPdmlSetPreference

Description

Set the preference for the *Power Map* window.

Syntax

```
schPdmlSetPreference [-IsoNonCoverChecking on|off]
[-LvsNonCoverChecking on|off] [-CheckPowerMode on|off]
[-CellCoverCheck on|off] [-CheckCover on|off]
```

Arguments

`-CellCoverCheck on|off`

When this option is turned *on*, the *Power Map* window performs cell-covered checking. The default is *off*.

`-CheckCover on|off`

When this option is turned *on*, the *Power Map* window performs rule-covered checking. The default is *off*.

`-CheckPowerMode on|off`

When this option is turned *on*, the domain status and domain voltage defined in the Power mode is checked during Isolation/Level-shifter non-covered checking. The default is *on*.

`-IsoNonCoverChecking on|off`

When this option is turned *on*, the *Power Map* window performs Isolation non-covered checking. The default is *off*.

`-LvsNonCoverChecking on|off`

When this option is turned *on*, the *Power Map* window performs Level-shifter non-covered checking. The default is *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlSetPreference -IsoNonCoverChecking on
schPdmlSetPreference -CheckPowerMode off
```

schPdmlShowPowerNetwork

Description

Show the power domain supply network.

Syntax

`schPdmlShowPowerNetwork –win` *`window`* `–PowerDomain` *`domainNameList`*

Arguments

`–PowerDomain` *`domainNameList`*

Specify one or more power domains.

`-win` *`window`*

Specify the window ID of the invoking *Power Map* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlShowPowerNetwork -win $_nPowerMap19
-PowerDomain "i_cpu/PD_PCU" "i_cpu/PD_ALUB" "i_cpu/PD_CCU"
```

schPdmlTraceConnectivity

Description

Trace the connectivity of the specified switch or domain instance pin.

NOTE: Before specifying this command, the **schPdmlSelect** command must be specified.

Syntax

```
schPdmlTraceConnectivity -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlSelect -win $_nPowerMap19 -instpin "i_cpu/PD_ALUB" "i_cpu/
VDD_ALUB"
schPdmlTraceConnectivity -win $_nPowerMap19
```

schPdmlTraceDriver

Description

Trace the driver of the specified switch or domain instance pin.

NOTE: Before specifying this command, the **schPdmlSelect** command must be specified.

Syntax

```
schPdmlTraceDriver -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlSelect -win $_nPowerMap19 -instpin "i_cpu/PD_ALUB" "i_cpu/
VDD_ALUB"
schPdmlTraceDriver -win $_nPowerMap19
```

schPdmlTraceLoad

Description

Trace the load of the specified switch or domain instance pin.

NOTE: Before specifying this command, the **schPdmlSelect** command must be specified.

Syntax

```
schPdmlTraceLoad -win window
```

Arguments

`-win window`

Specify the window ID of the invoking *Power Manager* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlSelect -win $_nPowerMap19 -instpin "i_cpu/PD_ALUB" "i_cpu/
VDD_ALUB"
schPdmlTraceLoad -win $_nPowerMap19
```

schPdmlUndo

Description

Undo the last action.

Syntax

```
schPdmlUndo –win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Power Map* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
schPdmlUndo -win $_nPowerMap15
```

srcHilightPowerDomain

Description

Set a color for the specified power domain.

Syntax

```
srcHilightPowerDomain [-save fileName] [-load fileName] [-hilight
pdName colorName] [-default]
```

Arguments

-default

Set default colors for all power domains.

-hilight *pdName colorName*

Set the highlight color for the power domain named as "*pdName*" with the color specified as "*colorName*".

-load *fileName*

Load the specified file with power domain color settings.

-save *fileName*

Save the current power domain color settings to the specified file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcHilightPowerDomain -hilight system.PD_PCU ID_PURPLE5
```

srcViewPDMLLogFile

Description

View the CPF/UPF log file.

Syntax

```
srcViewPDMLLogFile
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcViewPDMLLogFile
```

tfgTrPwrX

Description

Identify the source of the unknown or error power state.

NOTE: The **tfgTrPwrX** command replaces the **tsTrPwrX** command.

Syntax

```
tfgTrPwrX -pd powerDomainName -time timeValue -state
```

Arguments

`-pd powerDomainName`

Specify the power domain name.

`-state`

Indicate the trace unknown or error power state.

`-time timeValue`

Specify the time to be traced.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
tfgTrPwrX -pd {/$power_root/\$Island_V1 } -time 550 -state
```

Right-click Commands

pdmlDisplayLibDefine

Description

Display the liberty file for the selected cell in the Power Aware *Source Code* pane. Only single selection is supported.

Syntax

```
pdmlDisplayLibDefine
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
pdmlDisplayLibDefine
```

SmartLog

Open Window/Log Commands

viaCreateLogViewer

Description

Opens the *SmartLog* window and returns the window ID. The window ID can be used to specify the target window for other Tcl commands.

Syntax

```
viaCreateLogViewer
```

Arguments

None

Value Returned

The Window ID if successful; otherwise, returns 0

Example

```
set winId [viaCreateLogViewer]
set logId [viaLogViewerOpenLog -file sim.log -window "$winId"]
```

viaLogViewerOpenLog

Description

Opens the log file in a SmartLog window. Each log occupies a tab in the SmartLog window.

Syntax

```
viaLogViewerOpenLog -file fileName -window winId -hyperRuleFile
hyperFile -parRuleFile parFile1 [parFile2]*
```

Arguments

`-file FileName`

Specifies the log file name.

`-window winId`

Specifies the window ID.

`-hyperRuleFile hyperFile`

Specifies the hyperlink rule file.

`-parRuleFile parFile1 [parFile2]*`

Specifies the partition rule files. You can specify multiple files.

Value Returned

The Log ID if successful; otherwise, returns 0.

Example

```
set winId [viaCreateLogViewer]
set logId [viaLogViewerOpenLog -file sim.log -hyperRuleFile
generic.rc -parRuleFile uvm.rc vcs.rc -window "$winId"]
```

viaLogViewerCloseLog

Description

Closes the specified log file that is opened in the SmartLog window.

Syntax

```
viaLogViewerCloseLog -logID logId -window winId
```

Arguments

`-logID logId`

Specifies the log file that must be closed.

`-window winId`

Specifies the window ID.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerCloseLog -logID $logId -window $winId
```

viaCloseLogViewer

Description

Closes the SmartLog window.

Syntax

```
viaCloseLogViewer -window winId
```

Arguments

`-window winId`

Specifies the window ID.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaCloseLogViewer -window $winId
```

viaLogViewerRefreshLog

Description

Refreshes the log.

Syntax

```
viaLogViewerRefreshLog -logID logId -window winId
```

Arguments

`-logID logId`

Specifies the log ID.

`-window winId`

Specifies the window ID.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerRefreshLog -logID $logId -window $winId
```

viaLogViewerRenameTab

Description

Renames the tab name of the log in the SmartLog window.

Syntax

```
viaLogViewerRenameTab -logID logId -window winId -name name
```

Arguments

`-logID logId`

Specifies the log ID.

`-window winId`

Specifies the window ID.

`-name name`

Specifies the new tab name.

NOTE: The first character of the new name must not be '*'.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerRenameTab -logID $logId -window $winId -name
new_name1
```

Save Commands

viaLogViewerRmbSaveContents

Description

Saves the contents of the active log of the specified SmartLog window.

Syntax

```
viaLogViewerRmbSaveContents -file filename -window winId
```

Arguments

`-file filename`

Specifies the file name in which the contents are saved.

`-window winId`

Specifies the window ID.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerRmbSaveContents -file export.txt -window $winId
```

viaLogViewerRmbSaveSelectedText

Description

Saves the selected text into a file.

Syntax

```
viaLogViewerRmbSaveSelectedText -file filename -text contents -
window winId
```

Arguments

`-file filename`

Specifies the file name in which the contents are saved.

`-text contents`

Specifies the text that must be saved.

`-window winId`

Specifies the window ID.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerRmbSaveSelectedText -file export.txt -text {selected
text is here} -window $winId
```

Hyperlink Commands

Global Tcl Variable Setting

On clicking a hyperlink in SmartLog, some Tcl variables are set automatically; users can apply these Tcl variables for their own Tcl processes.

Global TCL Variables

- **vgifActionLineNum**

  Syntax - `vgifActionLineNum Num: action line number on window`

  Example - `set vgifActionLineNum 6`

- **vgifActionWindow**

  Syntax - `set vgifActionWindow WindowID: type of window (for example, smartLog)`

  Example - `set vgifActionWindow SmartLog`

- **vgifActionTabIndex**

  Syntax - `set vgifActionTabIndex TabID : tab ID on window`

  Example - `set vgifActionTabIndex 1`

- **vgifActionLineStr**

  Syntax - `set vgifActionLineStr StringLine : string of action line`

  Example - `set vgifActionLineStr {Back to file '../../file/run/Design/TopModule.sv'.}`

- **vgifActionFileName**

  Syntax - `set vgifActionFileName File: absolute file path opened on window`

  Example - `set vgifActionFileName {/file/Design/sim.log}`

viaLogViewerOpenTimeHyperlink

Description

If the *SmartLog* window is synchronized to a specific *nWave* window, the *nWave's* cursor synchronizes to a specific time; otherwise, the primary *nWave's* cursor synchronizes to a specific time.

Syntax

```
viaLogViewerOpenTimeHyperlink -logID logId -window windowId -time time -timeUnit timeUnit
```

Arguments

`-logID` *`logId`*

Specifies the log ID of the window.

`-window` *`windowId`*

Specifies the window from which this Tcl command is executed.

`-time` *`time`*

Specifies the time to which the waveform's cursor points.

`-timeUnit` *`timeUnit`*

Specifies the time unit.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerOpenTimeHyperlink -logID $logId -window $winId -time
3400000 -unit ps
```

viaLogViewerOpenFileHyperlink

Description

Opens the file in *SmartLog* or *nTrace* window. If the filename extension is .log or log.gz, the file is opened in the *SmartLog* window. If the file is a pdf file, it is opened in Acrobat Reader; otherwise, the file is opened in the *nTrace* window.

Syntax

```
viaLogViewerOpenFileHyperlink -window windowId -fileName fileName
-lineNum lineNum -linkLinNum linkLinNum
```

Arguments

`-window windowId`

Specifies the window from which this Tcl command is executed.

`-fileName fileName`

Specifies the hyperlink file path on *SmartLog* window.

`-lineNum lineNum`

Specifies the hyperlink line number on the hyperlink file.

`-linkLineNum linkLineNum`

Specifies the action line number on window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerOpenFileHyperlink  -window $winId -fileName
uvm_resource_db.svh -lineNum 202 -linkLineNum 3
```

Navigation Commands

viaLogViewerGoToTime

Description

If the log has valid time information, the time ruler is shown for navigation. This Tcl command can be used to change the time cursor of the log.

Syntax

```
viaLogViewerGoToTime -logID logId -window winId { -next | -prev |
{ -time 1000 } }
```

Arguments

`-logID` *`logId`*

Specifies the log from which this Tcl command is executed.

`-window` *`windowId`*

Specifies the window from which this Tcl command is executed.

`-time` *`time`*

Specifies the time to which the time cursor points.

`-next`

Points the cursor to the next time.

`-prev`

Points the cursor to the previous time.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerGoToTime -logID $logId -window $winId -time 1000
viaLogViewerGoToTime -logID $logId -window $winId -next
viaLogViewerGoToTime -logID $logId -window $winId -prev
```

viaLogViewerGoToBlock

Description

With partition rule, the log can be separated into different blocks in the structure view. This Tcl command is used to navigate in the block base.

Syntax

```
viaLogViewerGoToBlock -logID logId -window winId { -next | -prev }
```

Arguments

`-logID` *logId*

Specifies the log from which this Tcl command is executed.

`-window` *windowId*

Specifies the window from which this Tcl command is executed.

`-next`

Points the cursor to the next time.

`-prev`

Points the cursor to the previous time.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerGoToBlock -logID $logId -window $winId -next
viaLogViewerGoToBlock -logID $logId -window $winId -prev
```

viaLogViewerGoToLine

Description

Moves the cursor to the specified line.

Syntax

```
viaLogViewerGoToLine -logID logId -window winId -line lineNo
```

Arguments

`-logID logId`

Specifies the log from which this command is executed.

`-window windowId`

Specifies the window from which this Tcl command is executed.

`-line lineNo`

Specifies the line number.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerGoToLine -logID $logId -window $winId -line 100
```

viaLogViewerSyncWaveCursor

Description

Synchronizes the active log to a certain waveform.

Syntax

```
viaLogViewerSyncWaveCursor -window winId [-nWave waveName] {-
enable | -disable}
```

Arguments

`-window windowId`

Specifies the window from which this Tcl command is executed.

`-nWave waveName`

Specifies the target *nWave* window by *nWave* window name.

`-enable`

Enables the synchronization.

`-disable`

Disables the synchronization.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerSyncWaveCursor -window $winId -nWave nWave4 -enable
viaLogViewerSyncWaveCursor -window $winId -disable
```

viaLogViewerSetCurrentLog

Description

Switches to a different log.

Syntax

viaLogViewerSetCurrentLog -logID logId -window winId

Arguments

`-logID` *logId*

Specifies the log from which this command is executed.

`-window` *windowId*

Specifies the window from which this Tcl command is executed.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerSetCurrentLog -logID $logId -window $winId
```

Log Setting Commands

viaLogViewerSetLogTimeUnit

Description

The log may include time information without specifying the time unit, such as, “100”. If the log time unit is set as “ps” by the users, *SmartLog* uses “ps” for all the time which has no specified unit.

Syntax

```
viaLogViewerSetLogTimeUnit -logID logId -window winId -multiplier
timeValue -unit {s | ms | us | ns | ps | fs}
```

Arguments

`-logID logId`

Specifies the log from which this command is executed.

`-window windowId`

Specifies the window from which this Tcl command is executed.

`-multiplier timeValue`

Specifies the value of time.

`-unit { s | ms | us | ns | ps | fs }`

Specifies the time unit.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerSetLogTimeUnit -logID $logId -window $winId -
multiplier 10 -unit ps
```

viaLogViewerSetDisplayTimeUnit

Description

The display time unit affects the time ruler and the time column value in structure view. This Tcl command is used to change the display time unit. Note that the display time unit is applicable for the whole window, which means that each log within the window shares the same display time unit.

Syntax

```
viaLogViewerSetDisplayTimeUnit -window winId -multiplier timeValue
-unit { s | ms | us | ns | ps | fs }
```

Arguments

`-window windowId`

Specifies the window from which this Tcl command is executed.

`-multiplier` *`timeValue`*

Specifies the value of time.

`-unit { s | ms | us | ns | ps | fs }`

Specifies the time unit.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerSetDisplayTimeUnit -window $winId -multiplier 10 -
unit ps
viaLogViewerSetDisplayTimeUnit -window $winId -multiplier 100 -
unit ns
```

View Commands

viaLogViewerSwitchContext

Description

Switches the log to a different view - *File* view or *Structure* view.

Syntax

```
viaLogViewerSwitchContext -logID logId -window winId { -fileView |
-structView }
```

Arguments

`-logID` *`logId`*

Specifies the log from which this Tcl command is executed.

`-windowID` *`windowId`*

Specifies the window from which this Tcl command is executed.

`-fileView`

Switches to *File* view.

`-structView`

Switches to *Structure* view.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerSwitchContext -logID $logId -window $winId -fileView
viaLogViewerSwitchContext -logID $logId -window $winId -
structView
```

viaLogViewerSetSearchResultView

Description

Displays or hides the *Search Result* view.

Syntax

```
viaLogViewerSetSearchResultView -logID logId -window windowId {-
show | -hide}
```

Arguments

`-logID` *`logId`*

Specifies the log from which this Tcl command is executed.

`-windowID` *`windowId`*

Specifies the window from which this Tcl command is executed.

`-show`

Shows the *Search Result* view.

`-hide`

Hides the *Search Result* view.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerSetSearchResultView -logID $logId -window $winId -
show
viaLogViewerSetSearchResultView -logID $logId -window $winId -
hide
```

Search/Filter Commands

viaLogViewerGrepStart

Description

Searches for a keyword in the log.

Syntax

```
viaLogViewerGrepStart -logID logId -window winId -key searchText
[-caseInsensitive] { -next | -prev }
```

Arguments

`-logID` *`logId`*

Specifies the log from which this Tcl command is executed.

`-windowID` *`windowId`*

Specifies the window from which this Tcl command is executed.

`-key searchText`

Specifies the keyword to search.

`-caseInsensitive`

Specifies the search as case insensitive; otherwise, the search is case sensitive.

`-next`

Searches for the next result.

`-prev`

Searches for the previous result.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerGrepStart -logID $logId -window $winId -key UVM -
caseInsensitive -next
viaLogViewerGrepStart -logID $logId -window $winId -key OVM -prev
```

viaLogViewerSetMsgFilter

Description

Filters the log by message.

Syntax

```
viaLogViewerSetMsgFilter -logID logId -window winId -type type -
severity level -code code -userType key1 [-userType key2]*
```

Arguments

`-logID logId`

Specifies the log from which this Tcl command is executed.

`-windowID windowId`

Specifies the window from which this Tcl command is executed.

`-type type`

Specifies the type that must be filtered out. This option affects the column <Type> in the structure mode.

`-severity [ Info | Note | Warning | Error | Fatal | Trace | Debug | Verbose | Other ]`

Specifies the severity that must be filtered out. This option affects the column <Severity> in the structure mode.

`-code code`

Specifies the code that you want to display. This option affects the column <Code> in the structure mode.

`-userType keyword`

Specifies the keyword that you want to display. This option affects the user-specified columns in the structure mode.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

- Filter out blocks with type “UVM”

```
viaLogViewerSetMsgFilter -logID $logID -window $winID -type
{UVM} -severity {} -code {} -userType {}
```

- Filter out blocks with severity “Error”

```
viaLogViewerSetMsgFilter -logID $logID -window $winID -type
{} -severity {Error} -code {} -userType {}
```

- Display only blocks with code “LP_ISO_DIS”

```
viaLogViewerSetMsgFilter -logID $logID -window $winID -type
{} -severity {} -code {LP_ISO_DIS} -userType {}
```

- Display only blocks which have the `key1` keyword in the first user specified column

```
viaLogViewerSetMsgFilter -logID $logID -window $winID -type
{} -severity {} -code {} -userType {key1} -userType {}
```

- Display only blocks which have the `key2` keyword in the second user specified column

```
viaLogViewerSetMsgFilter -logID $logID -window $winID -type
{} -severity {} -code {} -userType {} -userType {key2}
```

viaLogViewerSetTimeFilter

Description

Filters the log by time.

Syntax

`viaLogViewerSetTimeFilter -logID` *`logId`* `-window` *`windowId`* `-beginTime` *`time`* `-endTime` *`time`*

Arguments

`-logID` *`logId`*

Specifies the log from which this Tcl command is executed.

`-windowID` *`windowId`*

Specifies the window from which this Tcl command is executed.

`-beginTime time`

Specifies the begin time.

`-endTime time`

Specifies the end time.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerSetTimeFilter -logID $logId -window $winId -beginTime
100 -endTime 200
```

viaLogViewerZoomTime

Description

When the log has blocks with different time, the time ruler is shown for users to navigate by time. To focus on a specific period, you can zoom in and zoom out the time ruler.

Syntax

```
viaLogViewerZoomTime -logID logId -window winId -type { all | in |
out }
```

Arguments

`-logID` *`logId`*

Specifies the log from which this Tcl command is executed.

`-windowID` *`windowId`*

Specifies the window from which this Tcl command is executed.

`-type { all | in | out }`

`all`: Restores the time ruler to full view.

`in`: Zooms in the time ruler.

`out`: Zooms out the time ruler.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
viaLogViewerZoomTime -logID $logId -window $winId -type in
viaLogViewerZoomTime -logID $logId -window $winId -type out
viaLogViewerZoomTime -logID $logId -window $winId -type all
```

Transaction Debug

Transaction and Protocol Analyzer

evbCreateWin

Description

Opens a new *Transaction and Protocol Analyzer* pane.

Syntax

```
evbCreateWin [-file fsdbFilename]
```

Arguments

`-file` *`fsdbFileName`*

Specifies the FSDB file to be opened in a new *Transaction and Protocol Analyzer* pane.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
evbCreateWin -file 1.fsdb
```

Transaction Table View

eaCreateWin

Description

Opens a new *Transaction Table View* pane.

Syntax

```
eaCreateWin [-f fsdbFilename]
```

Arguments

`-f` *`fsdbFilename`*

Specifies the FSDB file to be opened in a new *Transaction Table View* pane.

Value Returned

Returns window ID if successful; 0 if it fails.

Example

```
eaCreate -f "novas.fsdb"
```

eaGetActiveWin

Description

Gets the last active *Transaction Table View* window.

Syntax

```
eaGetActiveWin
```

Arguments

No arguments

Value Returned

Returns window ID if successful; 0 if it fails.

Transaction Relation Navigator

eRelationCreateWindow

Description

Opens a new *Transaction Relation Navigator* pane for the specified event. If a *Transaction Relation Navigator* pane has been opened previously, a new tab in the pane is opened for the specified event; otherwise, a new pane is opened for the specified event.

Syntax

```
eRelationCreateWindow  -eventId eventId -f filename
```

Arguments

`-eventId` *`eventId`*

Specifies the event to be displayed in the *Transaction Relation Navigator* pane.

`-f` *`fsdbFilename`*

Specifies the FSDB file to be opened in a new *Transaction Relation Navigator* pane.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
eRelationCreateWindow -eventId 2000 -f  "top/cases/test.fsdb"
```

Transaction and Protocol Analyzer

verdiInvokePA

Description

Specifies the settings to invoke *Transaction and Protocol Analyzer*.

Syntax

```
verdiInvokePA [-dir dirName] [-vipName vipName] [other_options]
```

Arguments

`-dir dirName`

Specifies the name of the directory with simulation results.

`-vipName vipName`

Specifies the names of the VIPs.

`other_options`

Specifies other command line options that are supported by *Transaction and Protocol Analyzer*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
verdiInvokePA -dir /user/simDir -vip cpu
```

Transaction Analysis

Window

taCreateWindow

Description

Creates a new *Analysis* window.

Syntax

```
taCreateWindow [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the *Analysis* window to be invoked.

Value Returned

The *Analysis* window ID.

Example

```
set current_window [taCreateWindow]
```

taCloseWindow

Description

Closes an *Analysis* window.

Syntax

```
taCloseWindow [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
taCloseWindow -win $_nTrans1
```

taGetCurrentWindow

Description

Gets the current active *Analysis* window.

Syntax

```
taGetCurrentWindow window
```

Value Returned

Current *Analysis* window ID.

Example

```
set current_window [taGetCurrentWindow]
```

taGetFileTimeUnit

Description

Gets the FSDB file time unit.

Syntax

```
taGetFileTimeUnit [-win window]
```

Argument

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

Time unit (in *G*, *K*, *M*, *m*, *u*, *n*, *p*, or *f* seconds, e.g. nanosecond or picosecond) if successful; otherwise, returns 0.

Example

```
taGetFileTimeUnit -win $_nTrans1
1ns
```

taGetWindowTimeUnit

Description

Gets the window time unit.

Syntax

```
taGetWindowTimeUnit [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

Time unit (in *G*, *K*, *M*, *m*, *u*, *n*, *p*, or *f* seconds, e.g. nanosecond or picosecond) if successful; otherwise, returns 0.

Example

```
taGetWindowTimeUnit -win $_nTrans1
2us
```

taStatCapture

Description

Captures the statistics chart from the *Statistics* window.

Syntax

```
taStatCapture [-win window] [-footer footer] [-region region]
-file filename
```

Argument

`-file` *`filename`*

Specifies the file name to save the captured image to.

`-footer` *`footer`*

Specifies the printable message to be shown on the footer.

`-region` *`region`*

Specifies the region to be printed or saved.

`-win` *`window`*

Specifies the window ID of the invoked *Statistics* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taStatCapture -win $_nStatistics8 -file
```

File

taCloseFile

Description

Closes all FSDB files opened in the *Analysis* window.

Syntax

```
taCloseFile [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
taCloseFile trans.fsdb
```

taOpenFile

Description

Loads an FSDB file into the *Analysis* window.

Syntax

```
taOpenFile [-win window] -file fileName
```

Arguments

`-file` *`fileName`*

Specifies the name of an FSDB file containing transactions.

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

FSDB file name if successful; otherwise returns 0.

Example

```
taOpenFile -file trans.fsdb
```

taRestoreSession

Description

Restores the status from the session file.

Syntax

```
taRestoreSession [-win window] -file fileName
```

Arguments

`-file fileName`

Specifies the filename to restore the status.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taRestoreSession -win $_nTrans4   ./aaaa.ta.ses
```

taSaveAs

Description

Saves the current content (this is either all transactions or the results of a filter) into a file with the file extension *txt*.

Syntax

```
taSaveAs [-win window] -file fileName
```

Arguments

`-file fileName`

Specifies the output file name.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taSaveAs -file output_trans.txt
```

taSaveSession

Description

Saves the status into a session file.

Syntax

```
taSaveSession [-win window] -file fileName
```

Arguments

`-file fileName`

Specifies the filename to be saved.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taSaveSession -win $_nTrans4   ./aaaa.ta.ses
```

Stream

taAddAllStreams

Description

Adds all the streams of the opened FSDB files to the *Analysis* window.

Syntax

```
taAddAllStreams [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

{1 stream_list} if adding any streams to the *Analysis* window is successful of adding any streams to *Analysis* window; otherwise, returns 0. The returned stream_list is the list of stream names added into the window; each stream name is enclosed with a bracket.

Example

```
taAddAllStreams
```

taAddStream

Description

Adds a stream to the *Analysis* window.

Syntax

```
taAddStream [-win window] [-delim delim] -stream fullHierName
```

Arguments

`[-delim` *`delim`*`]`

Specifies the delimiter of the stream.

`-stream` *`fullHierName`*

Specifies the stream name with full hierarchy to be added to the *Analysis* window.

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taAddStream -win $_nTrans7 -stream {BusTop/MyAHB_1/AhbTransfer}
```

taDeleteStream

Description

Removes the stream from the specified window.

Syntax

```
taDeleteStream [-win window] -stream fullHierName
```

Arguments

`-stream` *`fullHierName`*

Specifies the stream name with full hierarchy to be deleted from the *Analysis* window.

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taDeleteStream -win $_nTrans7 -stream {BusTop/MyAHB_1/AhbTransaction}
```

taGetAllStreams

Description

Gets all stream names of the opened FSDB files in the *Analysis* window.

Syntax

```
taGetAllStreams [-win window]
```

Argument

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

{1 stream_list} if getting any streams to the *Analysis* window is successful of getting any streams to *Transaction Analyzer* window; otherwise, returns 0. The returned stream_list is the list of stream names got into the window; each stream name is enclosed with a bracket.

Example

`taGetAllStreams`

The return value is {1 {StreamDirectory/AhbCycle} {StreamDirectory/AhbTransaction} {StreamDirectory/AhbTransfer} {StreamDirectory/Config} {StreamDirectory/Error} }

taGetTransactionList

Description

Gets a list of transactions by a time range for the specified stream in the *Analysis* window.

Syntax

```
taGetTransactionList [-win window] [-time timeRange] [-stream streamName]
```

Arguments

`-stream streamName`

Specifies the stream name.

`-time timeRange`

Specifies the time range to get transactions.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

A list of transaction indices.

Example

```
set trans_list [taGetTransactionList -stream addr_phase ]
```

taIterCancel

Description

Cancels the iterator.

Syntax

```
taIterCancel iterateHandler
```

Argument

`iterateHandler`

Specifies the iterator handle.

Value Returned

None.

Example

```
taIterCancel $trans_itr
```

taIterNext

Description

Gets next item from the iterator.

Syntax

```
taIterNext iterateHandler
```

Argument

`iterateHandler`

Specifies the iterator handle.

Value Returned

A transaction index if successful; otherwise, returns 0.

Example

```
set trans_idx [taIterNext $trans_itr]
```

taIterTransaction

Description

Gets an iterator for iterating transactions within a time range in a stream.

Syntax

```
taIterTransaction [-win window] [-time timeRange] -stream streamName
```

Arguments

`-stream` *`streamName`*

Specifies the stream name.

`-time` *`timeRange`*

Specifies the time range to get transactions.

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

An iterator handle if successful; otherwise, returns 0.

Example

```
set trans_itr [taIterTransaction -stream {top/addr_phase}]
```

taMergeStreams

Description

Merges several streams to form a new stream.

Syntax

```
taMergeStreams [-win window] [-name newStreamNames] -streamList streamNameList
```

Arguments

`-name` *`newStreamName`*

Specifies the name of the created stream. If not specified, the name is formed by concatenating selected stream names with '_'.

`-streamList` *`streamNameList`*

Specifies a list of stream names. Each stream name must be enclosed by brace, that is, {stream_name}, to correctly separate the name. If the brace is not specified in the stream list, the space character is applied as the separate-token.

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

{1 the_created_stream_name} if successful; otherwise, returns 0. The created stream name returned is the full path name.

Example

```
taMergeStreams -streamList {{top/stream1} {top/stream2}}
```

taSelectStream

Description

Selects a stream in the *Analysis* window.

Syntax

```
taSelectStream [-win window] -stream streamName
```

Arguments

`-stream` *`streamName`*

Specifies the stream name.

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taSelectStream -stream addr_phase
```

Transaction

taSetActiveTransaction

Description

Sets the active transaction in the *Analysis* window.

Syntax

```
taSetActiveTransaction [-win window] [-stream streamName] -trans transIdx
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-stream streamName`

Specifies a stream name.

`-trans transIdx`

Specifies a transaction index.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taSetActiveTransaction -stream {top/stream1} -trans 100
```

taGetActiveTransaction

Description

Gets the transaction index for the active transaction in the window.

Syntax

```
taGetActiveTransaction [-win window]
```

Argument

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

The active transaction index.

Example

```
set trans_idx [taGetActiveTransaction]
```

taGetTransactionLabel

Description

Gets the label of a transaction.

Syntax

```
taGetTransactionLabel [-win window] [-stream streamName] -trans transIdx
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-stream streamName`

Specifies a stream name.

`-trans transIdx`

Specifies a transaction index.

Value Returned

The label of the transaction.

Example

```
set trans_label [taGetTransactionLabel -trans 100]
```

taGetTransactionBeginTime

Description

Gets the begin time of a transaction.

Syntax

```
taGetTransactionBeginTime [-win window] [-stream streamName] -trans transIdx
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-stream streamName`

Specifies a stream name.

`-trans transIdx`

Specifies a transaction index.

Value Returned

The begin time of the transaction.

Example

```
set begin_time [taGetTransactionBeginTime]
```

taGetTransactionEndTime

Description

Gets the end time of a transaction.

Syntax

```
taGetTransactionEndTime [-win window] [-stream streamName] -trans transIdx
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-stream streamName`

Specifies a stream name.

`-trans transIdx`

Specifies a transaction index.

Value Returned

The end time of the transaction.

Example

```
set end_time [taGetTransactionBeginTime]
```

taHighlightTransactions

Description

Highlights a list of transactions.

Syntax

```
taHighlightTransactions [-win window] [-stream streamName] -color colorName -
transList transLis
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-stream streamName`

A stream name. Switch the specified stream to be active one. If not specified, apply to be the active one.

`-color colorName`

Specify the color. The available color options are:
ID_BLACK, ID_GRAY1, ID_GRAY2, ID_GRAY3,
ID_GRAY4, ID_GRAY5, ID_GRAY6, ID_WHITE,
ID_RED1, ID_RED2, ID_RED3, ID_RED4,
ID_RED5, ID_RED6, ID_RED7, ID_RED8,
ID_ORANGE1, ID_ORANGE2, ID_ORANGE3, ID_ORANGE4,
ID_ORANGE5, ID_ORANGE6, ID_ORANGE7, ID_ORANGE8,
ID_YELLOW1, ID_YELLOW2, ID_YELLOW3, ID_YELLOW4,
ID_YELLOW5, ID_YELLOW6, ID_YELLOW7, ID_YELLOW8,
ID_GREEN1, ID_GREEN2, ID_GREEN3, ID_GREEN4,
ID_GREEN5, ID_GREEN6, ID_GREEN7, ID_GREEN8,
ID_CYAN1, ID_CYAN2, ID_CYAN3, ID_CYAN4,
ID_CYAN5, ID_CYAN6, ID_CYAN7, ID_CYAN8,
ID_BLUE1, ID_BLUE2, ID_BLUE3, ID_BLUE4,
ID_BLUE5, ID_BLUE6, ID_BLUE7, ID_BLUE8,
ID_PURPLE1, ID_PURPLE2, ID_PURPLE3, ID_PURPLE4,
ID_PURPLE5, ID_PURPLE6, ID_PURPLE7, ID_PURPLE8

`-transList transLis`

A list of transaction indices.

Value Returned

{1 *a_list_of_transactions*} if highlighting more than one of the transaction(s) in current *Analysis* window is successful; otherwise, returns 0. The returned transaction list is the list of transactions highlighted in the window.

Example

```
taHighlightTransactions -transList {10 11 12} -color ID_GREEN1
```

taClearHighlightTransactions

Description

Clears the highlight of transactions.

Syntax

```
taClearHighlightTransactions [-win window] [-stream streamName]
[-transList transList]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-stream streamName`

Specifies a stream name. Switch the specified stream to be active one. If not specified, apply to be the active stream.

`-transList transLis`

A list of transaction indices.

Value Returned

1 if successful; 0 otherwise.

Example

```
taClearHighlightTransactions -stream {top/addr_phase} -transList {10 11 12}
```

taClearAllHighlightTransactions

Description

Clears all transaction highlights.

Syntax

```
taClearAllHighlightTransactions [-win window] [-stream streamName]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-stream streamName`

Specifies a stream name. Switch the specified stream to be active one. If not specified, apply to be the active one.

Value Returned

1 if successful; 0 otherwise.

Example

```
taClearAllHighlightTransactions -stream {top/addr_phase}
```

taGetAttributeList

Description

Gets the list of attribute names for transactions in the specified stream.

Syntax

```
taGetAttributeList [-win window] [-stream streamName]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-stream streamName`

Specifies a stream name.

Value Returned

A list of attribute names.

Example

```
set attr_list [taGetAttributeList]
```

taGetAttributeValue

Description

Gets the value of an attribute.

Syntax

```
taGetAttributeValue [-win window] [-stream streamName] -trans transIdx -attr
attrName
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

`-stream` *`streamName`*

Specifies a stream name.

`-trans` *`transIdx`*

Specifies a transaction index.

`-attr` *`attrName`*

Specifies an attribute name.

Value Returned

{1 value_of_the_attribute} if successful; otherwise returns 0.

Example

```
set attr_value [taGetAttributeValue -trans 100 -attr BurstLen]
($attr_value is 1 incr8 if successful)
```

View

addColorRuleTable

Description

Adds one colorize rule table.

Syntaxes

```
addColorRuleTable -name {tableName} -rule {rule_1} {rule_2}...
```

Arguments

`-name {tableName}`

Specifies the table name.

`-rule {rule_1} {rule_2}...`

Specifies one or more rules.

Value Returned

1 if successful; 0 otherwise.

Example

```
addColorRuleTable -name {Colorize_Table_0} -rule \
{n  "Severity"=="4" ID_BLUE5} \
{y   "Severity"<"4"  ID_YELLOW5}  {y "Severity">="6" ID_PURPLE8}
```

addFilterRuleTable

Description

Adds one filter rule table.

Syntaxes

```
addFilterRuleTable -name {tableName} -rule {rule_1} {rule_2}...
```

Arguments

`-name {tableName}`

Specifies the table name.

`-rule {rule 1} {rule_2}...`

Specifies one or more rules.

Value Returned

1 if successful; otherwise returns 0.

Example

```
addFilterRuleTable -name {Filter_Table_0} -rule \
{y        "Label"=="*txa*"        AND}
addFilterRuleTable -name {Filter_Table_1} -rule \
{n   "Label"=="*txa*"        AND} \
{y      "Severity"=="5" AND}
```

defaultPredefColorRuleTab

Description

Resets all colors back to their original default settings.

Value Returned

1 if successful; otherwise returns 0.

Example

```
defaultPredefColorRuleTab
```

taClearColorize

Description

Clears colorization results in the *Analysis* window.

Syntax

```
taClearColorize [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Transaction Analyzer* window.

Example

```
taClearColorize -win $_nTrans3
```

taClearFilter

Description

Clears filtering results in the *Transaction Analyzer* window.

Syntax

```
taClearFilter [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Transaction Analyzer* window.

Example

```
 taClearFilter -win $_nTrans3
```

taConfigureColumn

Description

Configures the column order and visibility in a *Transaction Analyzer* window.

Syntax

```
taConfigureColumn [-win window] columnList
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

`columnList`

A list of column names to be displayed in order. Columns that are not specified are hidden. Reserved column names are:

- Index: Transaction index.
- BeginTime: Transaction begin time.
- EndTime: Transaction end time.

- Label: Transaction label.

Value Returned

1 if successful; 0 otherwise.

Example

```
taConfigureColumn {Command BurstType Master Slave}
```

taFilter

Description

Filters or colorizes the transactions/messages according to the specified rule table.

Syntaxes

```
taFilter [-win window] -filter|-colorize {tableName} [-syncResult] [-
caseSensitive]
```

Arguments

`[-caseSensitive]`

Performs case matching for the attributes comparison.

`-colorize {tableName}`

Colorizes the transactions/messages based on the specified rule table in the *Analysis* window.

`-filter {tableName}`

Filters the transactions/messages based on the specified rule table in the *Analysis* window.

`-syncResult`

Specifies whether to synchronize the filtered results with the waveform pane.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise returns 0.

Example

```
taFilter -win $_nTrans3 -filter {Filter_Table_0} -syncResult
taFilter -win $_nTrans3 -colorize {Colorize Table} -syncResult
```

taFind

Description

Finds a specified transaction/message string in an *Analysis* window.

Syntax

```
taFind {ruleString} [-win window] [-matchcase on|off] [-prev]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`{ruleString}`

Specifies a set of rules to be searched. The rule string should be surrounded by braces.

`-matchcase on|off`

Turns the case matching *on* or *off*. The default is *off*.

`-prev`

Specifies if the pattern should be searched in a backward direction. The default is finding the string in forward direction.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taFind {cpu} -matchcase
taFind {"Label"=="*Sample*"&&"Severity"=="5"} -win $_nTrans2
-prev -matchcase
```

taGetCursor

Description

Gets the current cursor time of an *Analysis* window.

Syntax

```
taGetCursor [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 cursor time if successful; 0 otherwise.

Example

```
set cursor_time [taGetCursor]
($cursor_time is 1 10000 if successful.)
```

taGetMarker

Description

Gets current marker time of an *Analysis* window.

Syntax

```
taGetMarker [-win window] [-name markerName]
```

Arguments

`-name markerName`

A user-defined marker name.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 marker time if successful; 0 otherwise.

Example

```
set marker_time [taGetMarker]
($marker_time is 1 10000 if successful.)
```

taJumpToCursor

Description

Scrolls the window to the current cursor time.

Syntax

```
taJumpToCursor [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
taJumpToCursor
```

taJumpToMarker

Description

Scrolls the window to the current marker time.

Syntax

```
taJumpToMarker [-win window] [markerName]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`markerName`

A user-defined marker name.

Value Returned

1 if successful; 0 otherwise.

Example

```
taJumpToMarker
```

taRmFreezeCol

Description

Unfreezes column in the *Analysis* window. This means all columns can be scrolled.

Syntax

```
taRmFreezeCol [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
taRmFreezeCol -win $_nTrans2
```

taSetAlignment

Description

Sets alignment for the selected column.

Syntaxes

```
taSetAlignment [-win window] -column columnName
-left|-center|-right
```

Arguments

`-column columnName`

Specifies the selected column.

`-left|-center|-right`

Specifies whether the selected column is left-aligned, centered, or right-aligned.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; 0 otherwise.

Examples

```
taSetAlignment -win $_nTrans3 -column "uint64_attr0" -left
```

taSetCursor

Description

Sets the cursor time. If a time is specified, the cursor time is automatically aligned to the nearest begin time of a transaction. If the row index of a transaction is specified, the cursor time is updated to the begin time of the transaction.

Syntaxes

```
taSetCursor [-win window] -time time
taGetCursor [-win window] -row rowNo
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-time time`

Specifies new cursor time.

`-row rowNo`

Specifies the row index of the transaction.

Value Returned

1 if successful; 0 otherwise.

Examples

```
taSetCursor -win $nTrans1 -time 10000
taSetCursor -win $nTrans1 -row 5
taSetMarker -win $nTrans1 -row 15
```

taSetFreezeCol

Description

Freezes column in the *Analysis* window. This is similar to the Excel freeze column function where any columns to the left can not be scrolled; only columns to the right can be scrolled.

Syntax

```
taSetFreezeCol [-win window] -column columnName
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-column columnName`

Specifies the column name.

Value Returned

1 if successful; 0 otherwise.

Example

```
taSetFreezeCol -win $_nTrans2 -column "BeginTime"
```

taSetMarker

Description

Sets the marker time. If a time is specified, the marker time is automatically aligned to the nearest begin time of a transaction. If the row index of a transaction is specified, the marker time is updated to the begin time of the transaction.

Syntaxes

```
taSetMarker [-win window] [-name markerName] -time time
taSetMarker [-win window] [-name markerName] -row rowNo
```

Arguments

`-name markerName`

Specifies a user-defined marker name.

`-row rowNo`

Specifies the row index of the transaction.

`-time time`

Specifies new marker time.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise returns 0.

Examples

```
taSetMarker -win $nTrans1 -time 10000
```

taSort

Description

Sorts transactions by the specified column.

Syntax

```
taSort [-win window] -orderBy columnName [-descend on|off]
```

Arguments

`-descend on|off`

- on: Sort by descending order.
- off: Sort by ascending order.

`-orderBy columnName`

The name of the column. Reserved column names are:

- Index: Transaction index.
- BeginTime: Transaction begin time.
- EndTime: Transaction end time.
- Label: Transaction label.

`-win window`

Specifies the window ID of the invoked *Analysis* window.

NOTE: If `-descend` is not specified, the sorting order is automatically reversed when sorting on the same column, just as the behavior in GUI.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taSort -orderBy BurstLength
```

taSync2Waveform

Description

Option for synchronizing the active transaction between *Analysis* and *nWave* windows.

Syntax

```
taSync2Waveform on|off [-win window]
```

Arguments

`on|off`

Turns synchronization *on* or *off*.

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise returns 0.

Example

```
taSync2Waveform on -win $_nTrans1
```

taSyncCursorTime

Description

Option for synchronizing the global Verdi cursor time in the *Analysis* window.

Syntax

```
taSyncCursorTime [-win window] [on|off]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`on | off`

Turns this option *on* or *off*.

Value Returned

1 if successful; otherwise returns 0.

Example

```
taSyncCursorTime on -win $_nTrans1
```

Radix

taGetRadix

Description

Gets the radix of a column.

Syntax

```
taGetRadix [-win window] -column columnName
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

`-column` *`columnName`*

Specifies a column name.

Value Returned

The current format of the specified attribute.

Example

```
set attr_radix [taGetRadix -column Address]
```

taSetRadix

Description

Sets radix for a column.

Syntax

```
taSetRadix -column columnName -format format
```

Arguments

`-column` *`columnName`*

Specifies a column name.

`-format` *`format`*

Specifies the address display format. The available formats are: Bin, Oct, Hex, Dec, and ASCII.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taSetRadix -column Address -format Hex
```

Transaction Comparison

taCmpSetOptions

Description

Configures comparison criteria.

Syntax

```
taCmpSetOptions [-win window] -baseStream {{fsdb_filename} {full_stream name}}
-cmpStream {{fsdb_filename} {full_stream name}}  -cmpAttrib
{compared_attributes_list} -nameMapping
{{Attribute_name_in_base_stream,attribute_name_in_compared_stream}}[-
timeInBase {from_time, to_time, timeunit} ][-timeInCmp {from_time, to_time,
timeunit} ][-caseSensitive on|off]
[-stopErrCount number][-beginTimeTol time] [-endTimeTol time]
```

Arguments

`-baseStream {{`*`fsdb_filename`*`} {`*`full_stream name`*`}}`

Specifies the base stream name in the FSDB file.

`-beginTimeTol` *`time`*

Specifies the base tolerance for the begin time.

`-caseSensitive on|off`

Turns the case matching for attributes comparison *on* or *off*.

`-cmpAttrib {`*`compared_attributes_list`*`}`

Specifies the list of attribute names to be compared in the base stream.

`-cmpStream {{`*`fsdb_filename`*`} {`*`full_stream name`*`}}`

Specifies the stream name to be compared to in the FSDB file.

`-endTimeTol` *`time`*

Specifies the base tolerance for the end time.

`-nameMapping {{`*`Attribute_name_in_base_stream`*`,`
`attribute_name_in_compared_stream``}}`

Specifies the list of attribute name mapping pairs between base stream and compared stream.

`-stopErrCount` *`number`*

Specifies the maximum number of errors allowed before stopping the comparison.

`-timeInBase {from_time, to_time, timeunit}`

Specifies the base stream time range to be used for comparison. The default is *full range*.

`-timeInCmp {from_time, to_time, timeunit}`

Specifies the time range of the stream being compared.

`-win window`

Specifies the window ID of the invoked *Transaction Comparison* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taCmpSetOptions -baseStream {{/qa/FsdbPool/Transaction/testcases/viewing/
txdb.fsdb} {/MyAHB_1/AhbTransaction}} -cmpStream {{/qa/FsdbPool/Transaction/
testcases/viewing/leon.fsdb} {/Stream_Dir/AhbTransaction}}
taCmpSetOptions -cmpAttrib {BeginTime,Label,Command}    taCmpSetOptions -
nameMapping
{{BeginTime,BT}{Command,Label}{BeatCount,Beat}{Address,StartAddress}} -
timeInBase {5000,6250000,ns} -caseSensitive off
-stopErrCount 1000
```

taCmpDoComparison

Description

Performs the comparison using the criteria set by Tcl command **taCmdSetOptions**.

Syntax

```
taCmpDoComparison [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Transaction Comparison* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taCmpDoComparison
```

taCmpSaveAs

Description

Saves current differences into a file. The file format is .text (*.txt*).

Syntax

```
taCmpSaveAs [-win window] -file filename
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Transaction Comparison* window.

`-file` *`filename`*

Specifies the output file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taCmpSaveAs -file output_trans.txt
```

taCmpCloseWindow

Description

Closes the *Transaction Comparison* window.

Syntax

```
taCmpCloseWindow [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Transaction Comparison* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taCmpCloseWindow -win $_nCompare1
```

taCmpFind

Description

Finds a specified pattern string in a *Transaction Comparison* window.

Syntax

```
taCmpFind [-win window] [-matchcase on|off] [-prev] patternString
```

Arguments

`-matchcase on|off`

Turns the case matching *on* or *off*. The default is *off*.

`-prev`

Specifies if the pattern should be searched in a backwards direction. If not specified, the string is searched in a forward direction.

`PatternString`

The pattern string to be matched.

`-win window`

Specifies the window ID of the invoked *Transaction Comparison* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taCmpFind -matchcase -prev {burst}
```

taCmpConfigureColumn

Description

Configures the column order and visibility in a *Transaction Comparison* window.

Syntax

```
taCmpConfigureColumn [-win window] columnList
```

Arguments

`-win window`

Specifies the window ID of the invoked *Transaction Comparison* window.

`columnList`

Specifies a list of column names to be displayed in order. Columns that are not specified will be hidden.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taCmpConfigureColumn {{Command} {BurstType} {Master} {Slave}}
```

taCmpJumpToDiff

Description

Jumps current view to a specified mismatch area.

Syntax

```
taCmpJumpToDiff [-win window]  -first|-prev|-next|-last
```

Arguments

`-win window`

Specifies the window ID of the invoked *Transaction Comparison* window.

`-first|-prev|-next|-last`

Jump to the first, previous, next, or last mismatch area.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taCmpJumptoDiff -first
```

Data Window

taCreateDataWindow

Description

Creates a *Data Window* with the address and data of the current cursor row in the *Analysis* window.

Syntax

```
taCreateDataWindow
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taCreateDataWindow
```

taDataCloseWindow

Description

Closes the *Data Window*.

Syntax

```
taDataCloseWindow [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the invoked *Data Window*.

Value Returned

1 if successful; otherwise returns 0.

Example

```
taDataCloseWindow -win $_nTransData4
```

taDataSaveAs

Description

Saves the current transaction data into a file with the file extension *txt*.

Syntax

```
taDataSaveAs [-win window] -file fileName
```

Arguments

`-file fileName`

Specifies the output file name.

`-win window`

Specifies the window ID of the invoked *Data Window*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taDataSaveAs -win $_nTransData4 -file "out.txt"
```

taDataFind

Description

Searches for data or address string in the current *Data Window* by inputting the pattern.

Syntax

```
taDataFind [-win window] [-pattern patternString] [-matchtype Addr|Data] [-
next|-prev]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Data Window*.

`-pattern patternString`

A pattern string to be matched. Wildcards are supported.

`-matchtype Addr|Data`

Matches the pattern by either address string or data string.

`-next`

Finds the next matched pattern from the current selected cell.

`-prev`

Finds the previous matched pattern from the current selected cell.

Value Returned

The found address/data string if successful, otherwise returns null.

Example

```
taDataFind -win $_nTransData4 -pattern "*100*" -matchtype Data
-next
```

taDataSetDump

Description

Specifies the method to use to locate the previous/next transaction with the **Prev Dump** and **Next Dump** toolbar icons. Data in the new transaction is refreshed in the *Data Window*.

Syntax

```
taDataSetDump [-win window] [-dumpBy Seq|Read|Write|ReadWrite]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Data Window*.

`-dumpBy Seq|Read|Write|ReadWrite`

Sets the dump type from one of the following types:

- Seq: trace by sequence.
- Read: trace by *Command* category in the *Analysis* window with value *READ*.
- Write: trace by *Command* category in the *Analysis* window with value *WRITE*.
- ReadWrite: trace by *Command* category in the *Analysis* window with value *READ* or *WRITE*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taDataSetDump -win $_nTransData7 -dumpBy Read
```

taDataPrevDump / taDataNextDump

Description

Jumps to previous or next dump by the current dumping type.

Syntaxes

```
taDataPrevDump [-win window]
taDataNextDump [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the invoked *Data Window*.

Value Returned

Data address if successful, otherwise returns -1.

Examples

```
taDataPrevDump -win $_nTransData7
taDataNextDump -win $_nTransData7
```

taDataSyncCursorTime

Description

Synchronizes changes of the current address position in the *Data Window* to the cursor in the *Analysis* window. It applies after every action of the previous/next dump.

Syntax

```
taDataSyncCursorTime [-win window] [-toggle on|off]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Data Window*.

`-toggle on | off`

Turns this option *on* or *off*.

Value Returned

1 if successful; 0 otherwise.

Example

```
taDataSyncCursorTime -win $_nTransData7 -toggle on
```

taDataSetAddrRadix

Description

Sets radix format of address category in the *Data Window*.

Syntaxes

```
taDataSetAddrRadix [-win window] [-format Oct|Hex|Dec]
taDataSetAddrRadix [-win window] [-leadingZero on|off]
taDataSetAddrRadix [-win window] [-prefix on|off]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Data Window*.

`-format Oct|Hex|Dec`

Sets radix format of the address category. The available formats are:
Oct | Hex | Dec

`-leadingZero on|off`

Enables/disables the leading zeros in the address category.

`-prefix on|off`

Enables/disables the prefix of the radix format in the address category.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
taDataSetAddrRadix -win $_nTransData7 -format Hex
taDataSetAddrRadix -win $_nTransData7 -leadingZero on
```

```
taDataSetAddrRadix -win $_nTransData7 -prefix off
```

taDataSetDataRadix

Description

Sets radix format of data category in the *Data Window*.

Syntaxes

```
taDataSetDataRadix [-win window] [-format Bin|Oct|Hex|Dec|ASCII]
taDataSetDataRadix [-win window] [-leadingZero on|off]
taDataSetDataRadix [-win window] [-prefix on|off]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Data Window*.

`-format Bin|Oct|Hex|Dec|ASCII`

Sets radix format of the data category. The available formats are:
Bin | Oct | Hex | Dec | ASCII

`-leadingZero on|off`

Enables/disables the leading zeros in the data category.

`-prefix on|off`

Enables/disables the prefix of the radix format in the data category.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
taDataSetDataRadix -win $_nTransData7 -format Hex
taDataSetDataRadix -win $_nTransData7 -leadingZero on
taDataSetDataRadix -win $_nTransData7 -prefix off
```

taDataOption

Description

Specifies how many words can be shown in one row.

Syntax

```
taDataOption [-win window] [-display dataPerRow]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Data Window*.

`-display` *`dataPerRow`*

Specifies the number of data displayed in each row.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taDataOption -win $_nTransData4 -display 3
```

Statistics Window

taCreateStatWindow

Description

Performs statistical calculation on the *Analysis* window and creates a statistics window.

Syntax

```
taCreateStatWindow [-win window] -fromTime fromTime -toTime toTime -category
categoryAttributeName [-valueAttr valueAttrinuteName]
-calOption option
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-fromTime fromTime`

Specifies the from time.

`-toTime toTime`

Specifies the to time.

`-category categoryAttributeName`

Specifies the category attribute. Transactions having the same value of the attribute are grouped to the same category.

`-valueAttr valueAttrinuteName`

Specifies the value attribute. The argument is needed only when the calculate option is *MIN_MAX_AVG*.

`-calOption option`

The value of option is either *FREQUENCY* or *MIN_MAX_AVG*.

Value Returned

1 if successful; otherwise, returns 0.

Examples

```
taCreateStatWindow -win $_nTrans3 -fromTime 60000 -toTime 1.462e+07 -category
Command -calOption FREQUENCY
```

```
taCreateStatWindow -win $_nTrans3 -fromTime 60000 -toTime 1.462e+07 -category
Command -valueAttr BeatCount -calOption MIN_MAX_AVG
```

taStatExport

Description

Dumps the statistic information in the *Statistics Window* into a text file.

Syntax

```
taStatExport [-win window] -file "fileName"
```

Argument

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-file "fileName"`

Specifies the text file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taStatExport -win $_nStatistics4 -file "/qa/home/chsu/58336.txt"
```

taStatSelect

Description

Selects the specific transaction result data.

Syntax

```
taStatSelect [-win window] -rowIdx rowIndexNumber
```

Argument

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-rowIdx rowIndexNumber`

Specifies the row index number.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taStatSelect -win $_nStatistics4 -rowIdx 3
```

taStatCloseWindow

Description

Closes the Statistics Window.

Syntax

```
taStatCloseWindow [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the invoked *Analysis* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taStatCloseWindow -win $_nTrans3
```

taStatSwitchWndType

Description

Changes the transaction result data display.

Syntax

```
taStatSwitchWndType [-win window] -view [“PieChart”|”BarGraph”|”Text”]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-view [“PieChart”|”BarGraph”|”Text”]`

Specifies the view option as either pie chart, bar graph, or table.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taStatSwitchWndType -win $_nStatistics4 -view “PieChart”
```

taStatDuplicateAs

Description

Duplicates the current statistics window as either a pie chart, bar graph, or table format.

Syntax

```
taStatDuplicateAs [-win window] -view [“PieChart”|”BarGraph”|”Text”]
```

Arguments

`-win window`

Specifies the window ID of the invoked *Analysis* window.

`-view [“PieChart”|”BarGraph”|”Text”]`

Specifies the view option as either pie chart, bar graph, or table.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taStatDuplicateAs -win $_nStatistics4 -view “PieChart”
```

Relationship Window

taRelCloseWindow

Description

Closes the *Relationship Window*.

Syntax

```
taRelCloseWindow [-win window]
```

Argument

`-win` *`window`*

Specifies the window ID of the invoked *Relationship Window*.

Value Returned

1 if successful; otherwise returns 0.

Example

```
taRelCloseWindow -win $_nRelation1
```

taRelFind

Description

Finds a specified pattern string in a *Relationship Window*.

Syntax

```
taRelFind [-win window] [-matchcase on|off] [-prev] patternString
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Relationship Window*.

`-matchcase on|off`

Turns the case matching *on* or *off*. The default is *off*.

`-prev`

Specifies if the pattern should be searched in a backward direction. Default is finding the string in forward direction.

`patternString`

Specifies a pattern string to be matched.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
taRelFind -matchcase -prev burst
```

taRelFilter

Description

Filters the relationships upon the relationship names in the *Relationship Window*.

Syntaxes

`taRelFilter [-win` *`window`*`] -relNameList {`*`relationship name list`*`}`

Arguments

`-win` *`window`*

Specifies the window ID of the invoked *Relationship Window*.

`-relNameList {`*`relationship name list`*`}`

Specifies a list of relationship name(s) to be filtered out in the *Relationship Window*.

Value Returned

1 if successful; 0 otherwise.

Example

```
taRelFilter -relNameList {Child Parent R2}
```

taRelConfigureColumn

Description

Configures the column order and visibility in a *Relationship Window*.

Syntax

```
taRelConfigureColumn [-win window] columnList
```

Arguments

`-win window`

Specifies the window ID of the invoked *Relationship Window*.

`columnList`

Specifies a list of column names to be displayed in order. Columns that are not specified are hidden.

Value Returned

1 if successful; 0 otherwise.

Example

```
taRelConfigureColumn {{Command} {BurstType} {Master} {Slave}}
```

taRelSetAsTop

Description

Sets specified transaction as top and regenerates the relationship tree.

Syntax

```
taRelSetAsTop [-win window] -row rowNumber
```

Argument

`-win window`

Specifies the window ID of the invoked *Relationship Window*.

`-row rowNumber`

Specifies the row index of specified transaction in the *Relationship Window*.

Value Returned

1 if successful; 0 otherwise.

Example

```
taRelSetAsTop -row 5
```

taRelGoToTrans

Description

Jumps to the specified transaction in *Analysis* window.

Syntax

```
taRelGoToTrans [-win window] [-row rowNumber]
```

Argument

`-win window`

Specifies the window ID of the invoked *Relationship Window*.

`-row rowNumber`

Specifies the row index of specified transaction in the *Relationship Window*. If not specified, applies to active transaction.

Value Returned

1 if successful; 0 otherwise.

Example

```
taRelGoToTrans -row 5
```

taRelExpand

Description

Expands the related transactions of specified transaction with *n* level in the *Relationship Window*.

Syntax

```
taRelExpand [-win window] [-row rowNumber] -level N
```

Argument

`-win window`

Specifies the window ID of the invoked *Relationship Window*.

`-row rowNumber`

Specifies the row index of specified transaction in the *Relationship Window*. If not specified, applies the operation to current active transaction.

`-level N`

Specifies the *Nth* level to be expanded.

Value Returned

1 if successful; 0 otherwise.

Example

```
taRelExpand -level 5 -row 14
```

taRelHideRecurNode

Description

Option for hiding recursive node(s) in the *Relationship Window*.

Syntax

```
taRelHideRecurNode [-win window] on|off
```

Argument

`-win window`

Specifies the window ID of the invoked *Relationship Window*.

`on|off`

Specifies whether to turn this command *on* or *off*.

Value Returned

1 if successful; 0 otherwise.

Example

```
taRelHideRecurNode on
```

Transaction Evaluator

srcOpenTransEvalForm

Description

Opens the *Transaction Evaluator* form.

Example

```
srcOpenTransEvalForm
```

srcCloseTransEvalForm

Description

Closes the *Transaction Evaluator* form.

Example

```
srcCloseTransEvalForm
```

srcSelectTransEvalNode

Description

Selects the transaction scope.

Syntax

```
srcSelectTransEvalNode -scope fullpath
```

Arguments

`-scope` *`fullpath`*

Specifies a full path for the transaction.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSelectTransEvalNode -scope Transaction_Set.tb_ex2.dut.traffic_checker
```

srcSetSearchPattern

Description

Specifies the default pattern to search for in the **Transaction SVA Description List**.

Syntax

```
srcSetSearchPattern -pattern {searchPattern}
```

Arguments

`-pattern {searchPattern}`

Specifies the search pattern.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetSearchPattern -pattern {a*}
```

srcSetTransOptions

Description

Sets the options to use in the *Transaction Evaluator* form.

Syntax

```
srcSetTransOptions -searchInSubScope 0|1 -LoadFsdbAfterEvaluate 0|1
```

Arguments

`-searchInSubScope 0|1`

When this option is *on*, all transactions under the scope and its sub-scopes are shown in the **Transaction SVA Description List**. When this option is *off*, the transactions under the selected sub-scope are shown in the **Transaction SVA Description List**. The default is *on*.

`-LoadFsdbAfterEvaluate 0|1`

When this option is *on*, the transaction FSDB is automatically loaded into Verdi after evaluation. When this option is *off*, the transaction FSDB is not loaded into Verdi after evaluation. The default is *on*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcSetTransOptions -searchInSubScope 1 -LoadFsdbAfterEvaluate 1
```

srcTransEvalBuildVirFile

Description

Build a virtual FSDB file.

Syntax

```
srcTransEvalBuildVirFile -virtualFile fileName -fileNum number
-FileList fullPath
```

Arguments

`-virtualFile` *`fileName`*

Specifies the virtual FSDB file name.

`-fileNum` *`number`*

Specifies the number of files to include.

`-FileList` *`fullPath`*

Lists the individual FSDB files to include in the virtual FSDB file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTransEvalBuildVirFile -virtualFile nTX_result.fsdb.vf \
-fileNum 2 -FileList AssertDebWin_design.fsdb nTX_result.fsdb
```

srcTransEvalCommit

Description

Enables the assertions selected in the *Transaction Evaluator* window.

Syntax

```
srcTransEvalCommit
```

Example

```
srcTransEvalCommit
```

srcTransEvalEnable

Description

Adds the transactions based on the input transaction list.

Syntax

```
srcTransEvalEnable [-all]|-TransList {transList}
```

Arguments

`-all`

Selects all transactions in the **Transaction Enable List**. This option is the default.

`-TransList {transList}`

Specifies the transaction(s) in the **Transaction Enable List**.

NOTE: If *-all* and *-TransList {transList}* options are specified simultaneously, only the *-TransList {transList}* option takes effect.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTransEvalEnable -TransList {tb_ex2.dut.bind_inst.rest_seg2_assert }
```

srcTransEvalDisable

Description

Removes the transactions based on the input transaction list.

Syntax

```
srcTransEvalDisable [-all]|-TransList {transList}
```

Arguments

`-all`

Selects all transactions in the **Transaction Enable List**. This option is the default.

`-TransList {transList}`

Specifies the transaction(s) in the **Transaction Enable List**.

NOTE: If the *-all* and *-TransList {transList}* options are specified simultaneously, only the *-TransList {transList}* option will take effect.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcTransEvalDisable -TransList {tb_ex2.dut.bind_inst.rest_seg2_assert }
```

Class Browser

srcCBOpenForm

Description

Opens the *Class Browser* window.

Syntax

```
srcCBOpenForm [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBOpenForm -win $_nTrace2
```

srcCBCloseForm

Description

Closes the *Class Browser* window.

Syntax

```
srcCBCloseForm [-win windows Id]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBCloseForm -win $_nTrace3
```

srcCBTreeSelect

Description

Selects the tree node in the *Class Inheritance* pane of the *Class Browser* window.

Syntax

```
srcCBTreeSelect [-win window] -path nodePath
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *Class Browser* window.

`-path` *`nodePath`*

Specifies the path of the tree node.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBTreeSelect -win $_nTrace3 -path "base.run_third"
```

srcCBTreeAction

Description

Sets the active node in the *Class Inheritance* pane of the *Class Browser* window.

Syntax

```
srcCBTreeAction [-win window] -path nodePath
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

`-path nodePath`

Specifies the path of the active node.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBTreeAction -win $_nTrace3 -path "base.myBase"
```

srcCBTreeDrag

Description

Drags the selected tree node in the Class Inheritance pane to another location.

Syntax

```
srcCBTreeDrag [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCbTreeDrag -win $_nTrace3
```

srcCBTreeDrop

Description

Drags the selected tree node dragged from the *Class Inheritance* pane in another window (that is, the *nBench* window).

Syntax

```
srcCBTreeDrop [-win window]
```

Arguments

`-win` *window*

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBTreeDrop -win $_nTrace3
```

srcCBTreeSetShowDefine

Description

Sets the double-click action as **Show Definition**.

Syntax

```
srcCBTreeSetShowDefine [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBTreeSetShowDefine -win $_nTrace3
```

srcCBTreeSetGotoInstance

Description

Sets the double-click action as **Goto Instantiation**.

Syntax

```
srcCBTreeSetGotoInstance [-win window]
```

Arguments

`-win` *`window`*

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCbTreeSetGotoInstance -win $_nTrace3
```

srcCBTreeShowDefine

Description

Jumps to the type definition of the instance in *nBench*.

Syntax

```
srcCBTreeSetGotoInstance [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBTreeShowDefine -win $_nTrace3
```

srcCBTreeGotoInstance

Description

Jumps to the instantiation of the selected node in *nBench*.

Syntax

```
srcCBTreeGotoInstance [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBTreeGotoInstance -win $_nTrace3
```

srcCBGridClick

Description

Clicks the row in the *Member List* pane of the *Class Browser* window.

Syntax

```
srcCBGridClick [-win window] -column index
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

`-column index`

The index of the selected row in the *Member List* pane of the *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBGridClick -win $_nTrace3
```

srcCBGridDClick

Description

Double-click the row in the *Member List* pane of the *Class Browser* window.

Syntax

```
srcCBGridDClick [-win window]
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBGridDClick -win $_nTrace3
```

srcCBFileViewClick

Description

Click the line in the *Class Definition/Reference* pane of the *Class Browser* window.

Syntax

```
srcCBFileViewClick [-win window] -line linenumber -word pos
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

`-line linenumber`

Specifies the line in the *Source Quick View* pane of the *Class Browser* window.

`-word pos`

Click a specific word in the line.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBFileViewClick -win $_nTrace3 -line "5" -word "2"
```

srcCBFindClass

Description

Finds a class according to a specified condition.

Syntax

```
srcCBFindClass [-win window] -name pattern -case -next|-prev
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

`-name pattern`

Specifies the class name.

`-case`

Performs case-sensitive searching.

`-next|-prev`

Finds the next or previous matched class.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBFindClass -win $_nTrace3 -name "base" -case -next
```

srcCBFindMember

Description

Finds a class member according to a specified condition.

Syntax

```
srcCBFindMember [-win window] -name pattern -case -data -method
-allClass|-curClass -next|-prev
```

Arguments

`-win window`

Specifies the window ID of the invoking *Class Browser* window.

`-name pattern`

Specifies the member name.

`-case`

Performs case-sensitive searching.

`-data`

Specifies if data members on the member list are included in the search.

`-method`

Specifies if method members on the member list are included in the search.

`-allClass|-curClass`

Finds members in the current specified class scope or in all class scopes.

`-next|-prev`

Finds the next or previous matched member.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
SrcCBFindMember -win $_nTrace3 -name "base" -case -next
```

srcCBDumpAu

Description

Dumps all data in the *Class Browser* window.

Syntax

```
srcCBDumpAu [-win window]
```

Arguments

`-win` *window*

Specifies the window ID of the invoking *Class Browser* window.

Value Returned

Returns 1 if successful; otherwise, returns 0.

Example

```
srcCBDumpAu -win $_nTrace3
```

Memory/MDA

Window

nMemCreateWindow

Description

Create a new *Memory* window.

Syntax

```
nMemCreateWindow
```

Value Returned

Memory window ID.

Example

```
set $_myMem0 [nMemCreateWindow]
```

nMemSetCurrentWindow

Description

Set specified memory window ID as current window.

Syntax

```
nMemSetCurrentWindow -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

Value Returned

Current memory window ID.

Example

```
nMemSetCurrentWindow $current_window
```

nMemGetCurrentWindow

Description

Get the current active *Memory* window.

Syntax

```
nMemGetCurrentWindow
```

Value Returned

Current memory window pointer.

Example

```
set $curWinAddr nMemGetCurrentWindow
```

nMemCloseWindow

Description

Close a *Memory* window.

Syntax

```
nMemCloseWindow -win window
```

Arguments

`-win` *window*

Specify the window ID of the invoking *Memory* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemCloseWindow -win $_nMem0
```

nMemUpdate

Description

Update memory data from the FSDB file (Interactive Mode only).

Syntax

```
nMemUpdate -win window
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemUpdate -win $_nMem0
```

File

nMemGetVariable

Description

Display the memory variables in the *Memory* window.

Syntax

```
nMemGetVariable [-win window] -var variableName [-delim delim]
[-from source] [-addrRange addressRange] [-wordBitRange
wordBitRange] [-wordsInOneRow wordNum] [-time time]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

`-var` *`variableName`*

Specify the hierarchical variable name.

`-delim` *`delim`*

The delimiter of the specified variable name; the default is ".".

`-from`

Specify the variable source.

`-addrRange` *`addrRange`*

Specify the address range to display.

`-wordBitRange` *`wordBitRange`*

Specify the word bit range to display. This is only available for 3D variables and above.

`-wordsInOneRow` *`wordNum`*

Specify the number of words to be displayed in one row.

`-time`

Specify the time at which to display the memory contents.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemGetVariable -win $_nMem0 {/top/mda_ins/mda2x3[1:0]}
```

nMemSaveToFile

Description

Save memory variable contents to a file.

Syntax

```
nMemSaveToFile [-win window] -file fileName [-fullTimeRange
TRUE|FALSE]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

`-file` *`fileName`*

Specified the output file name.

`-fullTimeRange`

If specified, dumps for the full FSDB time range are saved; otherwise, only the traced range is saved.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemSaveToFile "mem.txt"
```

nMemSaveSession

Description

Save the current *Memory* window status into a session file.

Syntax

```
nMemSaveSession [-win window] -file fileName [-keynote keynote]
[-varInfo] [-radix] [-notation] [-addrRadix] [-addrColWidth]
[-cellColWidth]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

`-filename` *`fileName`*

Specify the "*.ses*" file name to save the memory contents to.

`-keynote` *`keynote`*

Specify the keynote shown on the header in the session file.

`-varInfo`

If specified, save variable information into a session file.

`-radix`

If specified, save radix information into a session file.

`-notation`

If specified, save notation information into a session file.

`-addrRadix`

If specified, save address radix information into a session file.

`-addrColWidth`

If specified, save address column width information into a session file.

`-cellColWidth`

If specified, save value column width information into a session file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemSaveSession -win $_nMem0 -file /qa/chsu/c.ses -varInfo
```

nMemRestoreSession

Description

Restore the status from the session file.

Syntax

```
nMemRestoreSession [-win window] -file sessionFile
[-dispOptionOnly]
```

Arguments

`-dispOptionOnly`

When specified, apply the display settings only.

`-filename` *`fileName`*

Specify the file name which was saved by the **nMemSaveSession** command.

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemRestoreSession -win $_nMem0 -filename /qa/chsu/c.ses
```

Concatenate

nMemCreateConcatnateVariable

Description

Concatenate two or more memory variables into one memory variable.

Syntax

```
nMemCreateConcatnateVariable -newVar newVariable -srcVarList
variableList -method (word|address)
```

Arguments

`-newVar` *`newVariable`*

Specify the newly created memory variable name.

`-srcVarList` *`variableList`*

Specify the name of variables to be concatenated. Two or more variables are required.

`-method word|address`

Specify concatenating scheme. If the specified method is word, these variables must be of the same address range so that they can be concatenated word by word. If the method chosen is address, these memory variables must be of the same bit width range so that they can be linked as a single variable with larger address range.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemCreateConcatnateVariable -name system.i_cpu.newSignal -method
word -varList {system.i_cpu.data system.i_cpu.i_alub.alu}
```

Slice

nMemCreateSliceVariable

Description

Create a new variable as a slice of an existing variable.

Syntax

```
nMemCreateSliceVariable -newVar newVariable -srcVar sourceVariable
-addrRange addrRange
```

Arguments

`-newVar newVariable`

The new variable name.

`-srcVar sourceVariable`

Existing variable name.

`-addrRange addrRange`

Specify the slice address range.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemCreateSliceVariable -newVar newVariable -srcVar
sourceVariable -addrRange [255:128]
```

Radix

nMemAddAliasFile

Description

Add and apply alias information from a file to the loaded memory variable.

Syntax

```
nMemAddAliasFile [-win window] -file aliasFile
```

Arguments

`-win window`

Specify the window ID of the invoking *Memory* window.

`-file aliasFile`

Alias file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemAddAliasFile cpu.aliass
```

nMemAddAliasProgram

Description

Apply alias from a program to the loaded memory variable.

Syntax

```
nMemAddAliasProgram [-win window] -prog aliasProgram
```

Arguments

`-win window`

Specify the window ID of the invoking *Memory* window.

`-prog aliasProgram`

The program name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
mdaAddAliasProgram alias.pl
```

nMemRemoveAlias

Description

Remove aliases from the selected signals and revert to display their numeric values.

Syntax

```
nMemRemoveAlias [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking *Memory* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemRemoveAlias
```

nMemSetRadix

Description

Set the radix for the memory variable value.

Syntax

```
nMemSetRadix [-win window] [-notation notation] [-format format]
[-alias aliasTableName]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

`-notation` *`notation`*

Specify signal value notation. Notation options are **Signed**, **Unsigned**, **2Com**, **1Com**, and **Mag**.

`-format` *`format`*

Specify format of signal value. Format options are **Bin**, **Oct**, **Hex**, **Dec**, and **ASCII**.

`-alias` *`aliasTableName`*

Specify the alias table name to apply.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemSetRadix -format Hex
```

View

nMemSetTime

Description

Set the memory window display time.

Syntax

```
nMemSetTime [-win window] -time time
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

`-time` *`time`*

Specify the display time.

Value Returned

Current time if successful; otherwise, returns 0.

Example

```
nMemSetTime -time 1000
```

Search

nMemFind

Description

Find the pattern in the *Memory* window.

Syntax

```
nMemFind [-win window] [-prev] -pattern pattern
```

Arguments

`-win window`

Specify the window ID of the invoking *Memory* window.

`-prev`

Set search direction to backward; otherwise, the search direction is forward.

`-pattern pattern`

Specify the search pattern.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemFind -prev -pattern "01111"
```

nMemPrevDump

Description

Jump to the previous time that satisfies the current search mode.

Syntax

```
nMemPrevDump [-win window] [-cellList cellAddressList]
```

Arguments

`-win window`

Specify the window ID of the invoking *Memory* window.

`-cellList` *`cellAddressList`*

If specified, jump to the previous time where the specified cell addresses have value changes. Otherwise, jump to the previous dump time where any of the cells in display range have value changes.

Value Returned

1 and jumped time if successful; otherwise, returns 0.

Example

```
nMemPrevDump -win $_nMem0 -addrList {[3][2], [3][5]}
```

nMemNextDump

Description

Jump to the next time that satisfies the current search mode.

Syntax

```
nMemNextDump [-win window] [-cellList cellAddressList]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking *Memory* window

`-cellList` *`cellAddressList`*

If specified, jump to the next time where the specified cell addresses have value changes. Otherwise, jump to the next dump time where any of the cells in display range have value changes.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemNextDump -win $_nMem0 -addrList {[3][2], [3][5]}
```

Configuration

nMemSetDispOptions

Description

Set display related settings of the *Memory* window.

Syntax

```
nMemSetDispOptions [-win window] [-dispMode displayMode]
[-addrColWidth addrColWidth] [-valueColWidth valueColWidth]
[-wordsInOneRow wordNum] [-sync TRUE|FALSE] [-addrRange addrRange]
[-wordBitRange wordBitRange] [-addrRadix radix] [-fixColumnWidth
TRUE|FALSE] [-showCellBitRangeWithAddr TRUE|FALSE]
```

Arguments

`-addrColWidth` *addressColumnWidth*

Display the address column width.

`-addrRadix` *radix*

Specify the address display format. The *radix* options include **Oct**, **Hex**, and **Dec**.

`-addrRange` *addrRange*

Specify the display address range.

`-dispMode` *displayMode*

Specify the display mode. The *displaymode* includes ADDR_VALUE and ADDR_HINT.

`-fixColumnWidth TRUE|FALSE`

When this option is set to TRUE, the column width is fixed. When this option is set to FALSE, the column width can be changed. The default is FALSE.

`-showCellBitRangeWithAddr TRUE|FALSE`

Set to TRUE to show the cell range with address in the grid window.

`-sync TRUE|FALSE`

Set to TRUE to turn on cursor time synchronization with the *nWave* window.

`-valueColWidth` *valueColWidth*

Display the value column width.

`-wordBitRange` *`wordBitRange`*

Specify the display word-bits range.

`-wordsInOneRow` *`wordNum`*

Specify words to show for one row.

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemSetDispOptions -win $_nMem0 -dispMode ADDR_HINT -addrColWidth
10 -cellColWidth 20 -wordsInOneRow 8 -addrRange [128:0]
-cellBitRange [7:1] -sync TRUE
```

nMemSetPreference

Description

Set the default memory viewing options and apply the settings to the current Window immediately.

Syntax

```
nMemSetPreference [-win window] [-dispMode displayMode]
[-addrColWidth addressColumnWidth] [-valueColWidth valueColWidth]
[-sync] [-fixColumnWidth] [-wordsInOneRow wordNum]
[-showCellBitRangeWithAddr TRUE|FALSE] [-font font]
```

Arguments

`-addrColWidth` *`addressColumnWidth`*

Display the address column width.

`-dispMode` *`displayMode`*

Specify the display mode. The *displaymode* includes ADDR_VALUE and ADDR_HINT.

`-fixColumnWidth`

If specified, the columns are fixed.

`-font` *`font`*

Specify the font.

`-showCellBitRangeWithAddr TRUE|FALSE`

Set to TRUE to show the cell range with address in the grid window.

`-sync`

When specified, turn the cursor time synchronization with the *nWave* window on.

`-valueColWidth` *`valueColWidth`*

Display the value column width.

`-wordsInOneRow` *`wordNum`*

Specify how many words to show for one row.

`-win` *`window`*

Specify the window ID of the invoking *Memory* window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
nMemSetPreference -win $_nMem0 -dispMode ADDR_VALUE
-wordsInOneRow 2 -addrColWidth 10 -valueColWidth 20
```

nECO

Window

schCreateWindow

Description

Creates an *nECO* window from the selected set.

Syntax

```
schCreateWindow [-win window] -eco -all
```

Argument

`-all`

Creates the *nECO* window for all the instances in the current *nSchema* window.

`-eco`

Creates the *nECO* window for the instances in the current *nSchema* window.

`-win window`

Specifies the window ID of the invoking *nSchema* window.

Value Returned

nSchema window ID if successful; 0 otherwise.

Example

```
schCreateWindow -win $current_window -eco
```

ECO Utility

schCellTypeReplace

Description

Replaces all cells at one time.

Syntax

```
schCellTypeReplace -old oldCell -new newCell [-portMapping newport
oldport|tie_up|tie_down] [-addCmt -keynote comment]
```

Argument

`-addCmt`

Adds comment next to netlist change.

`-new newCell`

Browses the cell to replace from the **Cell List**.

`-old oldCell`

Specifies the old cell name.

`-portMapping`

Specifies a port mapping for an undetermined cell.

Value Returned

1 if successful; 0 otherwise.

Example

```
schCellTypeReplace -win $_nSchema2 -old “IV” -new “IVP” -addCmt
-keynote “”
```

schChangeInstScope

Description

Moves the instance to the specified scope with the original connection kept. The changes are reflected in all windows immediately.

Syntax

```
schChangeInstScope [-win window] [-delim] -scope scopeName
[-inst instName] [-keynote comment]
```

Argument

`-delim`

Specifies the delimiter of the instances.

`-inst instName`

Specifies the instance to be moved.

`-keynote comment`

Specifies the comment string for the source code changes.

`-scope scopeName`

Specifies the destination scope.

`-win window`

Specifies the window ID of the invoking *nECO* window.

Value Returned

None.

Example

```
schChangeInstScope -win $current_window -scope "system.CHILD1"
-inst "MASTER" -keynote "Novas ECO updated by shelley Jul 6, 2001
18:16 "
```

schCloneLogic

Description

Duplicates selected logics in non-freeze silicon mode.

Syntax

```
schCloneLogic -scope scope -inst listInst [-prefix] [-retainInput]
[-retainOutput] [-newscope newscope]
```

Argument

`-inst listInst`

Specifies the instances.

`-newscope` *`newscope`*

The destination of the scope for the clone logic.

`-prefix`

Sets the prefix for the new instances

`-retainInput`

Retains the input signal.

`-retainOutput`

Retains the output signal.

`-scope` *`scope`*

The original scope for the clone logic to be cloned from.

Value Returned

1 if successful; 0 otherwise.

Example

```
schCloneLogic -scope “system.i_cpu.i_ALUB” -newscope “system”
-inst “U280” “U279” “U282” -prefix
```

schCloneModule

Description

Clones the instance to a new module and move the instance’s original module to the top level.

Syntax

```
schCloneModule [-win window] {new module-inst pair list}
```

Argument

`{new module-inst pair list}`

Displays the names of the new module and the instance that the new module is cloned from.

`-win` *`window`*

Specifies the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
schCloneModule {sys.LS100_TOP.LINK_LIST_MGT.LLM_A_GAP,eco_GAP}
{sys.LS100_TOP.LINK_LIST_MGT.LLM_B_GAP,eco_GAP1}
```

File

ecoSaveHDL

Description

Saves the source code to the destination directory.

Syntax

```
ecoSaveHDL [-dir dirName][-update][-scope scopeName][-orig]
[-prefix fileprefix]
```

Argument

`-dir dirName`

Specifies the destination directory.

NOTE: The specified directory *dirname* must exist and be writable for saving.

`-orig`

Force writing each file to the original path. The parameter *-dir* and *-orig* should not exist simultaneously.

`-prefix fileprefix`

Specifies the filename prefix of the modified netlist files. When the filename-with-prefix of the new netlist files exist in the path where the files are written, a serial number (starting from 0) is appended to the prefix. This ensures that every new filename in the path is unique. This is optional and only works with the *-orig* option. If this option is not specified, the modified files overwrite the source files.

`-scope scopeName`

Saves the source codes of the hierarchy under this specified scope.

`-update`

Saves the modified netlist files only.

Value Returned

None.

Example

```
ecoSaveHDL -dir "/dar/design/update" -update
```

ecoSaveLogFile

Description

Saves the ECO report.

Syntax

```
ecoSaveLogFile filename [-append]
ecoSaveLogFile -commitLog <file>
```

Argument

`-append`

Appends the specified file.

`-commitLog <file>`

Appends the extra commit report to ECO Report. <file> is the full path name of ECO Report.

`filename`

Specifies the file name.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoSaveLogFile "/dar/design/update/history.txt"
```

ecoSaveScript

Description

Saves the ECO modification to a script file.

Syntax

```
ecoSaveScript [-novas|-avanti|-magma|-astro|-icc[-c]|-iccTcl|
-fe filename] [-append]
```

Argument

`-append`

Adds the new script content to the end of the existing content when saving the ECO script to an existing file.

`-astro` *`filename`*

Saves the modified ECO file in Astro script format to the specified output file.

`-avanti` *`filename`*

Saves the modified ECO file in Avanti script format to the specified output file.

`-c`

Shows the current mode when opening the ECO Window the first time.

`-fe` *`filename`*

Saves the modified ECO file in First Encounter script format to the specified output file.

`-icc` *`filename`*

Saves the modified ECO file in ICC HECO script format to the specified output file.

`-iccTcl` *`filename`*

Saves the modified ECO file in ICC Tcl script format to the specified output file.

`-magma` *`filename`*

Saves the modified ECO file in Magma script format to the specified output file.

`-novas` *`filename`*

Saves the modified ECO file in Novas script format to the specified output file.

Value Returned

1 if successful, 0 otherwise

Example

```
ecoSaveScript -magma "Modification.eco"
```

Edit

ecoAddInst

Description

Adds an instance to the *nECO* window.

Syntax

```
ecoAddInst -win window -sparecell sparecell
-master cellName -inst instName
-consume {"consumed_spare_cell_name""new_instance_name"}
```

Argument

`-consume [{"consumed_spare_cell_name""new_instance_name"}]`

Specifies the consumed spare cells when multiple instances exist in the scope that the spare cell is added to.

`-inst instName`

Specifies the name of the added instance.

`-master cellName`

Specifies the master cell name of the added instances.

`-sparecell spareCell`

Specifies the cell name of the added instances.

`-win window`

Specifies the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoAddInst -sparecell "system.i_cpu.i_spare.spare0" -inst
"system.i_cpu.i_ALUB1.newInst1"-consume
{"system.i_cpu.i_spare.spare1" "system.i_cpu.i_ALUB2.newInst1"}
{"system.i_cpu.i_spare.spare11" "system.i_cpu.i_ALUB3.newInst1"}
```

ecoDeleteInst

Description

Deletes an instance from the *nECO* window.

Syntax

```
ecoDeleteInst [-win window] [-delim delim] [-inst instName]
```

Argument

`-delim delim`

Delimiter of the name_list.

`-inst instName`

If specified, delete the specified instance from *nECO* window. Otherwise, delete the selected instance from *nECO* window.

`-win window`

Specifies the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoDeleteInst -inst {system.i_cpu.u23}
```

ecoReplaceInst

Description

Replaces the master instance keeping the port configuration.

Syntax

```
ecoReplaceInst -win window_id [-delim delim] -master cellname|
-sparecell sparecell -inst instname [-map connections]
[-portMapping newport oldport|tie_up|tie_down]
```

Argument

`-delim delim`

Specifies the delimiter of the name list.

`-inst` *`instName`*

Specifies the name of the instance that will be replaced.

`-map` *`connections`*

Specifies the connection for the new spare cell in the freeze-silicon mode. The format of the connection is: -map "replace-cell-port1 local_netName1_of_port1 hierarchy-port1 hierarchy-port1-dir hierarchy-port2 hierarchy-port2-dir" "replace-cell-port2 local_netName2_of_port2 hierarchy-port2 hierarchy-port2-dir hierarchy-port2 hierarchy-port2-dir"...

NOTE: This option would be logged into the *verdi.cmd* file if the instance is replaced in the *nECO* window. It is not recommended to specify this option for the Tcl scripts because *nECO* makes the connection automatically. If this option is specified, the incorrect connection may cause the Tcl command to fail.

`-master` *`cellname`*

Specifies the name of the cell that will replace the instance in non-freeze silicon mode.

`-portMapping` *`newport`* `oldport|tie_up|tie_down`

Specifies the ports mapping information. Map the new cell ports to the replaced cell ports when the numbers of these cell ports differ. The first element is the new cell port and the second element is the old cell port or net that the new cell port would connect to. The format is: -portMapping "new Port1" "Old Port1" "new Port2" "tie_up" "new Port3" "sig3".

`-sparecell` *`sparecell`*

Specifies the name of the spare cell instance that will replace the instance in freeze-silicon mode.

`-win` *`window`*

Specifies the window ID of the invoking the *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoReplaceInst -win $_nSchema3 -master "AN2"
-inst "system.i_cpu.i_PCU.U237" -portMapping "A" "A" "B" "tie_up"
"Z" "Z"
```

ecoChangeInstScope

Description

Moves an instance to a specified scope. Break original connection.

Syntax

```
ecoChangeInstScope [-delim delim] [-win window] -scope scopeName
[-inst instName] [-new newName]
```

Argument

`-delim delim`

Specifies the delimiter of the name list.

`-inst instName`

If specified, relocate the instance to specified scope. Otherwise, relocate the selected instance.

`-new newName`

Specifies the new name of the scope where the instance would be re-located.

`-scope scopeName`

Specifies the new destination scope where the instance would be re-located.

`-win window`

Specifies the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoChangeInstScope -scope system.i_cpu.i_PCU -inst
system.i_cpu.u23
```

ecoMakeConnection

NOTE: *ecoMakeConnection* is able to describe path information of multiple bit connections.
For example, connect *system.i_cpu.i_CCU.SUB1.P[0:3]* to

system.i_cpu.i_PCU.SUB1.data[0:3] where *system.i_cpu.i_PCU.SUB1.data[3]* already connects to *system.i_cpu.s1*.

Description

Make a connection between two instance ports.

Syntax

```
ecoMakeConnection [-win window] -bitmap bitMap [-delim delim]
[-instport instancePort] [-instpin instancePin] [-map connection
information] -merge -signal fullSignalName -reserve fullSignalName
```

Argument

`-bitmap` *`bitMap`*

Specifies the range for the deleted nets to be merged to when the bit width of two nets are different.

`-delim` *`delim`*

Specifies the delimiter of the name list.

`-instpin` *`instancePin`*

Makes the connection between two specified instance pins.

`-instport` *`instancePort`*

Makes the connection between two specified instance ports.

`-map` *`connection information`*

Creates the port or net with the connection map information.

The `connection-info` format is as follows:

```
" default_net_name [fully_specified_port_list] "
```

where default_net_name refers to the default name for new signals, and fully_specified_signal_list refers to a list of fully specified signals separated by a space. For example:

```
-map " eco_n_1 "
-map "n781 system.i_cpu.i_ALUB.n781 Output
system.i_cpu.i_CCU.n781 Input"
```

The fully specified signal format:

```
port-full-name [port-direction-value]
```

where port-direction-value can be InOut, Input, or Output. For example:

```
system.i_cpu.i_ALUB.eco_n_1 InOut
```

Alternatively, \" may be used to quote the default net name and fully specified signals. By default, *nECO* will add \" to quote both the default net name and fully specified signal. For example,

```
-map " \"eco_n_2\" \"system.i_cpu.i_CCU.n1 Input\"
\"system.i_cpu.n3\" \"system.i_cpu.i_ALUB.n2 Input\" "
```

`-merge`

Merge the specified nets.

`-reserve`

Specify the net to be reserved.

`-signal` *`fullSignalName`*

Specify the nets to be operated on.

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoMakeConnection -win $_nSchema3 -map
"\"n587\" \"system.i_cpu.i_ALUB.i_alu.n587 Output\" "
-instpin \"system.i_cpu.i_ALUB.i_alu.U152" "A"
"system.i_cpu.i_ALUB.zero_flag_reg" "D"

ecoMakeConnection -win nSchema_3 -merge -signal
top.U2.temp_in_0_,top.U2.temp_in_1_ -signal top.U2.a[3:0]
-reserve top.U2.a[3:0] -bitmap 3 2
```

ecoDeleteConnection

Description

Deletes the connection from the instances.

Syntax

```
ecoDeleteConnection [-win window] [-delim delim]
[-instport instancePort] [-instpin instancePin]
[-onmodule inner|outer]
```

Argument

`-delim`

Specifies the delimiter of the instance name.

`-instpin` *`instancePin`*

Deletes the connection of the specified instance pins.

`-instport` *`instancePort`*

Deletes the connection of the specified instance ports.

`-onmodule inner|outer`

Deletes the inner or outer connection of the module ports.

`-win` *`window`*

Specifies the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoDeleteConnection -win $_nSchema4 -instport "CPU.i_CCU"
"\\mprom_out\[1\]" -onmodule "inner"
ecoDeleteConnection -win $_nSchema4 -instpin "CPU.i_CCU.C1_reg"
"Q"
```

ecoReConnect

Description

Reconnects the instance pin or instance port to another net.

Syntax

`ecoReConnect [-win` *`window`*`] -instpin` *`instancePin`* `-new` *`netName`*

Argument

`-instpin` *`instancePin`*

Specifies the hierarchical instance and pin name.

`-new` *`netName`*

Specifies the net name to connect to.

`-win` *`window`*

Specifies the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoReConnect -win $_nSchema4 -instpin
"system.i_cpu.i_CCU.\\CH_reg\[2\] " "Q" -new
"system.i_cpu.i_CCU.CH\[0\]"
```

ecoTieUp

Description

Ties the connection to the power net.

Syntax

```
ecoTieUp [-win window] [-delim] [-instport instanceportList]
[-instpin instancepinList] [-name]
```

Argument

`-delim`

Specifies the delimiter of the instance name.

`-instpin` *instancepinList*

Ties the connection of the specified instance pin(s) to the power net.

`-instport` *instanceportList*

Ties the connection of the specified instance port(s) to the power net.

`-name`

Specifies a name for the VDD or VSS net in ICC (IC Compiler).

`-win` *window*

Specifies the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoTieUp -instport system.i_cpu.u23 Z
```

```
ecoTieUp -win $_nSchema4 -instpin "system.i_cpu.i_ALUB.U263" "B"
-name VDD1
```

ecoTieDown

Description

Tie the connection to the ground.

Syntax

```
ecoTieDown [-win window] [-delim] [-instport instanceportList]
[-instpin instancepinList] [-name]
```

Argument

`-delim`

Delimiter of the instance name.

`-instpin instancepinList`

If specified, tie the connection of specified instance pin(s) to ground net. Otherwise, tie the selected instance pin(s) to ground net.

`-instport instanceportList`

If specified, tie the connection of specified instance port(s) to ground net. Otherwise, tie the selected instance port(s) to ground net.

`-name`

Specify a name for the VDD or VSS net in ICC (IC Compiler).

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoTieDown -instport system.i_cpu.u23 Z
ecoTieDown -win $_nSchema4 -instpin
"system.i_cpu.i_ALUB.carry_flag_reg" "Q" -name VSS1
```

ecoRedo

Description

Reverse the execution to the next state.

Syntax

```
ecoRedo [-win window]
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoRedo
```

ecoUndo

Description

Reverse the execution to the previous state.

Syntax

```
ecoUndo [-win window]
```

Argument

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoUndo
```

ecoRenameNet

Description

Renames the net name with the new net name.

Syntax

```
ecoRenameNet [-win window] [-delim delim] [-old old_name] -new
new_name
```

Argument

`-new`

Specifies the new name to replace with.

`-old`

Replaces the specified net with new name.

`-win window`

Specifies the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoRenameNet -old system.clk -new system.clk2
```

ecoRenameInst

Description

Renames the instance with the new name.

Syntax

```
ecoRenameInst [-win window] [-delim delim] [-inst instName]
-new newName
```

Argument

`-delim delim`

Specifies the delimiter of the instance name.

`-inst instName`

Specifies the instance name.

`-new newName`

Specifies a new name for the instance.

`-win window`

Indicates the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoRenameInst -win $_nSchema3 -inst "system.i_cpu.i_ALUB.eco_i1"
-new "myInst1"
```

ecoConnectToNet

Description

Connect the selected instance pin to the specified instance pin or net in the *nECO* window. Continuous assignment can be handled by **Connect to Net**.

Syntax

```
ecoConnectToNet [-win window] -assign “LHS” “RHS” -edit “New LHS”
“New RHS”
ecoConnectToNet [-win window] -concate concatenatedNetName
-instpin instancePinName [-merge|-reconnect]
ecoConnectToNet –map mapstr -instpin instpin -signal signalName
```

Argument

`-assign “LHS” “RHS”`

Specify the continuous assignment.

`-concate concatenatedNetName`

Specify the concatenated net name for the connection.

`-edit “New LHS” “New RHS”`

Edit the new assignment.

`-instpin instancePinName`

Specify the instance pin.

`-map` *`connection information`*

Create the port or net with the connection map information.

The `connection-info` format is as follows:

```
" default_net_name [fully_specified_port_list] "
```

where default_net_name refers to the default name for new signals, and fully_specified_signal_list refers to a list of fully specified signals separated by a space. For example:

```
-map " eco_n_1 "
-map "n781 system.i_cpu.i_ALUB.n781 Output
system.i_cpu.i_CCU.n781 Input"
```

The fully specified signal format:

```
port-full-name [port-direction-value]
```

where port-direction-value can be InOut, Input, or Output. For example:

```
system.i_cpu.i_ALUB.eco_n_1 InOut
```

Alternatively, \" may be used to quote the default net name and fully specified signals. By default, *nECO* will add \" to quote both the default net name and fully specified signal. For example,

```
-map " \"eco_n_2\" \"system.i_cpu.i_CCU.n1 Input\"
\"system.i_cpu.n3\" \"system.i_cpu.i_ALUB.n2 Input\" "
```

`-merge|-reconnect`

Specify whether to reconnect or merge nets.

`-signal` *`signalName`*

Specify the signal name.

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoConnectToNet -win $_nSchema3 -assign "top.Gx.wire_2\[0:1\]"
"top.Gx.wire_1\[0:1\]" -edit "top.Gx.wire_2\[0:1\]"
“top.Gx.new\[1:0\]”
```

```
ecoConnectToNet -win nSchema_3 -concate
top.U2.temp_in_0_,top.U2.temp_in_1_ -instpin top.u2.in[1:0] -
reconnect
```

ecoTraceDriver

Description

Find the driver(s) of the specified signal or instance. If the specified signal or instance does not exist, the action is applied to the selected signal or instance.

Syntax

```
ecoTraceDriver [-delim delim] [-win window] [-signal signalName|
-inst instName]
```

Argument

`-delim` *`delim`*

Specify the delimiter of the name list.

`-inst` *`instName`*

Specify the lowest hierarchy instance name.

`-signal` *`signalName`*

Specify the signal name.

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

A list of scope and instance name for the first traced result. 1 if successful; 0 otherwise. If the invoking window is a *full hierarchy* window, then 1 if successful; 0 otherwise. If the invoking window is a *flattened* window, then this command is only applied to these two types of window.

Example

```
ecoTraceDriver -win $_nSchema3
```

ecoTraceLoad

Description

Find the load(s) of the specified signal or instance. If the specified signal or instance does not exist, the action is applied to the selected signal or instance.

Syntax

```
ecoTraceLoad [-delim delim] [-win window] [-signal signalName|
-inst instName]
```

Argument

`-delim delim`

Specify the delimiter of the name list.

`-inst instName`

Specify the lowest hierarchy instance name.

`-signal signalName`

Specify the signal name.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

A list of scope and instance name for the first traced result if successful; 0 otherwise. If the invoking window is *full hierarchy* window, then 1 if successful; 0 otherwise. If the invoking window is *flattened* window, then this command is only applied to these two types of window.

Example

```
ecoTraceLoad -win $_nSchema3
```

ecoTraceConnectivity

Description

Find the connectivity(s) of the specified signal or instance. If the specified signal or instance does not exist, the action is applied to the selected signal or instance.

Syntax

```
ecoTraceConnectivity [-delim delim] [-win window]
[-signal signalName|-inst instName]
```

Argument

`-delim delim`

Specify the delimiter of the name list.

`-inst instName`

Specify the lowest hierarchy instance name.

`-signal signalName`

Specify the signal name.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

A list of scope and instance name for the first traced result if successful; 0 otherwise. If the invoking window is *full hierarchy* window, then 1 if successful; 0 otherwise. If the invoking window is *flattened* window, then this command is only applied to these two types of window

Example

```
ecoTraceConnecticity -win $_nSchema3
```

ecoDeselectAll

Description

Deselect all objects.

Syntax

```
ecoDeselectAll [-win window]
```

Argument

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoDeselectAll
```

ecoAddBuffer

Description

Add the buffer instance to a net.

Syntax

```
ecoAddBuffer [-win window] –master masterName –inst instName
–map mapstr -scope scopeName -old oldScopeName –new newScopeName
```

Argument

`–inst instName`

Specify the instance to be added.

`–master masterName`

Specify the master of the instance.

`-multiple additional buffer number`

Specify the additional buffer number.

`-new newScopeName`

Specify the new instance scope name.

`-old oldScopeName`

Specify the old instance scope name.

`-scope scopeName`

Specify the scope name.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoAddBuffer -win $_nSchema5 -master "DFXCN1" -inst
"CPU.i_SPARE.DFXCN1_SPARE_3" \
            -old "CPU.i_ALUB.\\X0\[0\] "
"CPU.i_SPARE.DFXCN1_SPARE_3.SA" \
            "CPU.i_SPARE.DFXCN1_SPARE_3.DA"
"CPU.i_SPARE.DFXCN1_SPARE_3.DB" \
            "CPU.i_SPARE.DFXCN1_SPARE_3.CP"
"CPU.i_SPARE.DFXCN1_SPARE_3.CDN" \
            -new "CPU.i_ALUB.eco_n30"
"CPU.i_SPARE.DFXCN1_SPARE_3.Q" \
            "CPU.i_SPARE.DFXCN1_SPARE_3.QN"
"CPU.i_ALUB.i_alu.\\a\[0\]" -map \
            "\\X0\[0\] CPU.i_ALUB.\\X0\[0\]  Output
CPU.i_SPARE.\\X0\[0\]  Input" \
            "eco_n30 CPU.i_SPARE.eco_n30 Output
CPU.i_ALUB.eco_n30 Input"
ecoAddBuffer -win $_nSchema5 -master "DFXCN1" -inst
"CPU.i_SPARE.DFXCN1_SPARE_3" \
            -multiple 6
```

ecoDelBuf

Description

Delete the buffer instance from the net.

Syntax

```
ecoDelBuf -win window -inst instName -signal signalname
-remove signalname
```

Argument

`-inst instName`

Specify the buffer instance name.

`-remove signalname`

Remove the specified signal that is connected to the instance.

`-signal signalname`

Keep the specified signal that is connected to the instance.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoDelBuf -win $_nSchema2 -inst “top.FUK.J1“ -signal
“top.FUK.wire_1” -remove “top.FUK.U_1”
```

ecoDeleteInst

Description

Delete the selected instance and the connections of its ports. If Freeze Silicon Mode is turned *on*, the deleted instance will be added to the spare cell list. It can handle continuous assignment as well.

Syntax

```
ecoDeleteInst -assign “LHS” “RHS”
```

Argument

`-assign` *“LHS” “RHS”*

The continuous assignment.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoDeleteInst -win $_nSchema2 -assign “top.FUK.wire_2“
“top.FUK.wire_1“
```

ecoCloneModule

Description

Clone the instance to a new module and move the instance’s original module to the top level.

Syntax

```
ecoCloneModule [-win window] {new module-inst pair list}
```

Argument

`{new module-inst pair list}`

Display the names of the new module and the instance that the new module is cloned from.

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoCloneModule -win
$_nSchema3{sys.LS100_TOP.LINK_LIST_MGT.LLM_A_GAP,eco_GAP}
{sys.LS100_TOP.LINK_LIST_MGT.LLM_B_GAP,eco_GAP1}
```

ecoCommit

Description

Commit all the changes to design. Reflect changes to all windows.

Syntax

`ecoCommit [-win` *`window`*`] [-comment comment]`

Argument

`-comment`

Put the comment on the corresponding updated source code. Otherwise, put the default comment *Novas ECO - date* on the source codes.

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoCommit
```

ecoCreatePort

Description

This command creates a port after you enter the **Module Name**, the **Port Name**, and the **Port Direction**.

Syntax

```
ecoCreatePort [-win window] -module “moduleName” -port “portName”
-direction input|output|inout [-instports “instPortName”]
```

Argument

`-direction input|output|inout`

Specify the direction. The direction include: **input**, **output**, or **inout**.

`-instports “instPortName”`

Specify the instant port name.

`-module “moduleName”`

Specify the module name.

`-port “portName”`

Specify the port name.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoCreatePort -win $_nSchema3 -module “ALUB” -port “newp1”
-direction input \
-instports “system.i_cpu.i_ALUB”
```

ecoAddViewObj

Description

Add the instance, signal or connector to the view objects set.

Syntax

```
ecoAddViewObj [-win window] [-delim delim] -inst listInst
-signal listSigname -connector listConnname -path {instancePin1}
{instancePin2}...
```

Argument

`-connector listConnname`

Specify the connector list to be added to the view object set.

`-delim delim`

Specify the delimiter of the instance names.

`-inst listInst`

Specify the instance list to be added to the view object set.

`-path {instancePin1} {instancePin2}...`

Specify all instance pins, including the hierarchy boundary ports, to be added to the view object set.

`-signal listSigname`

Specify the signal list to be added to the view object set.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoAddViewObj -inst {system.i_cpu.alu} {system.i_cpu.U23}
ecoAddViewObj -win $_nECO2 -path top.u1.U113.A
ecoAddViewObj -win $_nECO2 -inst "top.u1.U113"
```

ecoRemoveViewObj

Description

Remove the viewing object.

Syntax

```
ecoRemoveViewObj [-win window] [-delim delim]
[-inst listInst]
```

Argument

`-delim`

Specify the delimiter of the instance name.

`-inst` *`listInst`*

Remove the specified instance from the viewing object set of the *nECO* window.

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoRemoveViewObj -win $current_window -inst {system.i_cpu.alu}
{system.i_cpu.U23}
```

ecoEditConcateNet

Description

Edit the concate net.

Syntax

```
ecoEditConcateNet [-win window] [-signal signalName] index newname
pair list
```

Argument

`Index newname pair list`

Specify the indices of the concate net and mapping new net names.

`-signal` *`signalName`*

Specify the signal name.

`-win` *`window`*

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoEditConcateNet {1, nn1}
```

ecoSplitBus

Description

Specify whether to split the bus during an ECO operation.

Syntax

```
ecoSplitBus [-win window] -hideExtraneousBus on|off
```

Argument

`-hideExtraneousBus on|off`

When this option is turned *on*, set the split bus flag. The default is *off*.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoSplitBus -hideExtraneousBus off -win $_nSchema4
```

ecoMoveObj

Description

Move the object.

Syntax

```
ecoMoveObj [-win window] -offset dx dy
```

Argument

`-offset dx dy`

Specify the move offset.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoMoveObj -win $_nSchema3 -offset 3000 -2000
```

ecoChangePortDir

Description

Change the port direction list reported by the *nECO* after the *nECO* commits.

Syntax

```
ecoChangePortDir port-dir pair list [-comment comment]
```

Argument

`-comment comment`

Put the comment on the corresponding updated source code; otherwise, put the default comment *Novas ECO - date* on the source codes.

`port-dir pair list`

Specify the port and mapping new direction.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
ecoChangePortDir {top.U1.s1,Input}
```

Select

ecoSelect

Description

Select the instances or signals in a window.

Syntax

```
ecoSelect [-win window] [-delim delim] [-toggle]
-instport listInstportname
ecoSelect [-win window] [-delim delim] [-toggle]
-assign assign_LHS assign_RHS
```

Argument

`-assign assign_LHS assign_RHS`

Specify the continuous assignment.

`-delim delim`

Specify the delimiter for the names of the instances or signals.

`-instport listInstportname`

Specify the hierarchical port name.

`-toggle`

Toggle the selected object. Deselect the object when the object is in the selected set; otherwise, add the object into the selected set.

`-win window`

Specify the window ID of the invoking *nECO* window.

Value Returned

Number of objects selected.

Example

```
ecoSelect -win $current_window -toggle
ecoSelect -win $_nECO3 -instport "top.QA"
```

Configuration

ecoSetPreference

Description

Set the preferences for *nECO* window.

Syntax

```
ecoSetPreference [-preNet prefix] [-prePort prefix]
[-preInst prefix] [-format format] [-freezeSilicon off|[on
freezeSiliconOnOptions]] [-spareCellColor color] [-rColor color]
[-novas|-astro|-icc| -magma] [-magmaPower magmaPowerNetName]
[-magmaGround magmaGroundNetName] [-iccPower powerName]
[-iccGround groundName]
```

freezeSiliconOnOptions syntax includes: `[-manageMode cellNumber|instName [-pinManage off|[on [-input statusType] [-output statusType] [-nameRule prefix] [-cellQuantityMode on|off]]`

Argument

`-astro`

Set the current ECO output file mode as Astro script mode in the **Schematics -> ECO -> Script File** page of the **Preferences** command.

`-cellQuantityMode on|off`

Specify whether to control spare cell by number.

`-format`

Specify the default comment format in the **Schematics-> ECO-> General** tab of the **Preferences** command.

`-freezeSilicon on|off`

If the **Freeze Silicon** option in the **Schematics-> ECO-> Freeze Silicon** tab of the **Preferences** command is on, it is in freeze silicon mode. Otherwise, it is in non-freeze silicon mode.

`-icc`

Set the current ECO output file mode as ICC (IC Compiler) script mode in the **Schematics -> ECO -> Script File** page of the **Preferences** command.

`-iccGround` *groundName*

Specify the name of the ICC (IC Compiler) ground net.

`-iccPower` *`powerName`*

Specify the name of the ICC (IC Compiler) power net.

`-input` *`statusType`*

Set the default status for spare cell input pins as tie-up, tie-down or floating in the **Schematics-> ECO-> Freeze Silicon** tab of the **Preferences** command.

`-magma`

Set the current ECO output file mode as Magma script mode in the **Schematics -> ECO -> Script File** page of the **Preferences** command.

`-magmaGround` *`magmaGroundNetName`*

Specify the name of the Magma ground net.

`-manageMode` *`instName|cellNumber`*

Set the spare cell management mode as by cell number (Avant! spare-cell auto selection mode) or by instance number (formal freeze silicon management mode) in the in the **Schematics-> ECO-> Freeze Silicon** tab of the **Preferences** command.

`-magmaPower` *`magmaPowerNetName`*

Specify the name of the Magma power net.

`-nameRule`

Specify the prefix name for the spare cell name in the **Schematics-> ECO-> Freeze Silicon** tab of the **Preferences** command.

`-novas`

Set the current ECO output file mode as Novas script mode in the **Schematics -> ECO -> Script File** page of the **Preferences** command.

`-output` *`statusType`*

Set the default status for spare cell output pins as tie-up, tie-down or floating in the **Schematics-> ECO-> Freeze Silicon** tab of the **Preferences** command.

`-pinManage`

Set the spare cell pin management option in the **Schematics-> ECO-> Freeze Silicon** tab of the **Preferences** command.

`-preInst`

Specify default prefix for the new instance in the **Schematics-> ECO-> General** tab of the **Preferences** command.

`-preNet`

Specify the default prefix for the new net in the **Schematics-> ECO-> General** tab of the **Preferences** command.

`-prePort`

Specify the default prefix for the new port in the **Schematics-> ECO-> General** tab of the **Preferences** command.

`-rColor`

Specify the color for revised components in the **Schematics -> ECO-> Color** tab of the **Preferences** command.

`-spareCellColor`

Specify the color for spare cells in the **Schematics -> ECO -> Color** tab of the **Preferences** command.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoSetPreference -freezeSilicon on -input tieup -output tiedown -
nameRule dummy -format "ECO updated by \$\{UserName\} \$\{Date\}
\$\{Time\} "

ecoSetPreference -magma -magmaPower "VDD" -magmaGround "GND"

ecoSetPreference -iccPower "VDD" -iccGround "GND"
```

ecoSetOptions

Description

Set *nECO* window display options.

Syntax

`ecoSetOptions [-win` *`window`*`] [-fanInOut on|off] [-keepPlace on|off]`

Argument

`-fanInOut on|off`

Display the fan in or fan out number on the instance port.

`-keepPlace on|off`

Specify whether to keep placement after ECO operation.

`-win window`
Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoSetOptions -faninOut on
```

ecoDumpAU

Description

Dump the ECO change information.

Syntax

```
ecoDumpAU [-win window] [-ecoOnly] [-fanInOut]|[-report]
```

Argument

`-ecoOnly`
Dump the ECO change information caused only by the ECO.

`-fanInOut`
Display the fan in or fan out number on the instance port.

`-report`
Dump the ECO report.

`-win window`
Specify the window ID of the invoking *nECO* window.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoDumpAU -fanInOut
```

Spare Cell

ecoAddSpareCell

Description

Add instance(s) to spare cell form.

Syntax

```
ecoAddSpareCell [-inst list_inst]
```

Argument

`-inst`

Specify instances.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoAddSpareCell -inst
"mmaccell_bist_tb.dummy_scope.DUMMY_INST.spare_inst0"
```

ecoDeleteSpareCell

Description

Remove instance(s) from spare cell form.

Syntax

```
ecoDeleteSpareCell [-inst listInst] -cellName [cellName]
```

Argument

`-inst`

Specify instances.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoDeleteSpareCell -inst
"mmaccell_bist_tb.dummy_scope.DUMMY_INST.spare_inst0"
```

ecoLoadSpareCellFile

Description

Load spare cell file to spare cell form.

Syntax

```
ecoLoadSpareCellFile -file fileName
```

Argument

`-file`

Filename.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoLoadSpareCellFile -file "SaveSpareCell.log"
```

ecoSaveSpareCellFile

Description

Save listed spare cells in spare cell form to file.

Syntax

```
ecoSaveSpareCell -file fileName
```

Argument

`-file`

File name.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoSaveSpareCellFile -file "SaveSpareCell.log"
```

ecoSearchSpareCell

Description

Search matched instances and add to spare cell form.

Syntax

```
ecoSearchSpareCell -nameRule "string" -wildcard
```

Argument

`-nameRule`

Specify pattern.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoSearchSpareCell -nameRule "spare_"
```

ecoLoadSpareCellNum

Description

Import the spare cell list with instance name from file.

Syntax

```
ecoLoadSpareCellNum -file fileName
```

Argument

`-file` *`fileName`*

The file which contains the spare cell list with instance name.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoLoadSpareCellNum -file “sp1.cnt”
```

ecoSaveSpareCellNum

Description

Save the number list to a file for further usage of spare cell functions.

Syntax

`ecoSaveSpareCellNum -file` *`fileName`*

Argument

`-file` *`fileName`*

The file which contains the spare cell list with instance name.

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoSaveSpareCellNum -file “sp1.cnt”
```

ecoConfirmSpareCell

Description

Commit the spare cell number list.

Syntax

```
ecoConfirmSpareCell
```

Value Returned

1 if successful; 0 otherwise.

Example

```
ecoConfirmSpareCell
```

nRegister

Window

regOpenWindow

Description

Creates a new register window.

Syntax

```
regCreateWindow
```

Value Returned

Register window ID.

Example

```
set current_window [regOpenWindow]
```

regGetCurrentWindow

Description

Get the current active register window.

Syntax

```
regGetCurrentWindow
```

Value Returned

Current register window ID.

Example

```
set current_window  regGetCurrentWindow
```

regSetCurrentWindow

Description

Sets specified register window ID as current window.

Syntax

```
regSetCurrentWindow window
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSetCurrentWindow $_nReg1
```

regGetAllWindow

Description

Get all register window IDs.

Syntax

```
regGetAllWindow
```

Value Returned

The list window IDs of all register windows.

Example

```
set reg_window_list regGetAllWindow
```

regCloseWindow

Description

Closes a register window.

Syntax

```
regCloseWindow [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regCloseWindow -win $_nReg1
```

File

regRestore

Description

Restores all objects and corresponding attributes to register window from specified file.

Syntax

```
regRestore [-win window] fileName
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`fileName`

Specify file to restore from.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regRestore "case1.register"
```

regSave

Description

Saves all objects and corresponding attributes in register window into a file.

Syntax

```
regSave [-win window] fileName
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`fileName`
Specify file to save into.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSave "case1.register"
```

regReload

Description

Reloads previously opened *nRegister* rc file.

Syntax

```
regReload [-win window]
```

Arguments

`-win` *`window`*
Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regReload -win $_nReg0
```

Edit

regAddSignals

Description

Adds signals to a register window. All the added signals will be in selection list and upper left position can be specified.

Syntax

```
regAddSignals [-win window] [-delim delim] [-point "x,y"]
{signal1} {signal2}... {signaln}
```

Arguments

`-delim delim`

Specify the delimiter of the signal name.

`-point "x,y"`

Specify upper left coordinates of signal(s) box.

`{signal1} {signal2}... {signaln}`

Specify the signal name.

`-win window`

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAddSignals -win $_nReg0 -point "10,20" {top.a} {top.b} {top.c}
```

regUndo

Description

Allows you to undo the last action.

Syntax

```
regUndo [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regUndo -win $_nReg1
```

regRedo

Description

Allows you to redo the previously undone action.

Syntax

```
regRedo [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regRedo -win $_nReg1
```

regCut

Description

Cuts objects in selection set.

Syntax

```
regCut [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regCut -win $_nReg1
```

regCopy

Description

Cut objects in selection set.

Syntax

```
regCopy [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regCopy -win $_nReg1
```

regPaste

Description

Pastes the objects in buffer (from copy) to register window.

Syntax

```
regPaste [-win window]
```

Arguments

`-win` *window*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regPaste -win $_nReg1
```

regMove

Description

Moves selected objects with designated offset.

Syntax

```
regMove [-win window] -offset "x,y"
```

Arguments

`-win` *window*

Specify the window ID of the invoking register window.

`-offset` *“x,y”*

The offset to be moved.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regMove -win $_nReg1 -offset "10,10"
```

regSetOrientation

Description

Sets orientation of selected signal(s) and paragraph(s).

Syntax

```
regSetOrientation [-win window] -orient normal | bottom-up |
upside-down | top-down
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-orient normal | bottom-up | upside-down | top-down`

normal | bottom-up | upside-down | top-down

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSetOrientation -win $_nReg1 -orient upside-down
```

regSetOrder

Description

Changes order of selected objects.

Syntax

```
regSetOrder [-win window]
-order BringToFront | SendToBack | BringForward | SendBackward
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-order BringToFront | SendToBack | BringForward | SendBackward`

Specify what order-changing action to apply to selected objects.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSetOrder -win $_nReg1 -order BringToFront
```

regAlign

Description

Aligns selected objects with specified side.

Syntax

```
regAlign [-win window] -align left | right | top | bottom
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-align left | right | top | bottom`

Specify align side: left | right | top | bottom

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAlign -win $_nReg1 -align top
```

regSetActiveObj

Description

Sets an object as the active one for alignment and order-setting function.

Syntax

```
regSetActiveObj [-win window] -obj objID
```

Arguments

`-obj objID`

Specify the object to set as the active one.

`-win window`

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSetActiveObj -win $_nReg1 -obj $doObj27
```

regAddLine

Description

Adds a line on register window.

Syntax

```
regAddLine [-win window] -color colorID -width lineWidth
-point "x1,y1" "x2,y2"
```

Arguments

`-color colorID`

Specify the color of created line.

`-point “x1,y1” “x2,y2”`

Specify 2 ends of line to be created.

`-width lineWidth`

Specify the width of created line.

`-win window`

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAddLine -win $_nReg1 -color RED -width 2 -point "10,20"
"30,20"
```

regAddRectangle

Description

Adds a rectangle on register window.

Syntax

```
regAddRectangle[-win window] -color colorID -width lineWidth
-fill TRUE|FALSE -bbox "x1,y1" "x2,y2"
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

`-color` *`colorID`*

Specify the color of created rectangle.

`-width` *`lineWidth`*

Specify the width of created rectangle.

`-fill TRUE|FALSE`

Whether to fill the rectangle.

`-bbox` *`“x1,y1” “x2,y2”`*

Specify bounding box of rectangle to be created.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAddRectangle -win $_nReg1_window -color RED -width 2 -fill
TRUE -bbox "10,20" "30,20"
```

regAddEllipse

Description

Adds an ellipse on register window.

Syntax

```
regAddEllipse [-win window] -color colorID -width lineWidth
-fill TRUE|FALSE -bbox "x1,y1" "x2,y2"
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-color colorID`

Specify the color of created ellipse.

`-width lineWidth`

Specify the width of created ellipse.

`-fill TRUE|FALSE`

Whether to fill the ellipse.

`-bbox “x1,y1” “x2,y2”`

Specify bounding box of ellipse to be created.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAddEllipse -win $_nReg1_window -color RED -width 2 -fill TRUE
-bbox "10,20" "30,20"
```

regAddArrow

Description

Adds an arrow on register window.

Syntax

```
regAddEllipse [-win window] -color colorID -width lineWidth
-doubleHead TRUE|FALSE -head "x1,y1" -tail "x2,y2"
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-color colorID`

Specify the color of created arrow.

`-width lineWidth`

Specify the width of created arrow.

`-doubleHead TRUE|FALSE`

Whether to create arrow with double head.

`-head "x1,y1"`

Specify arrowhead vertex.

`-tail "x2,y2"`

Specify arrowtail vertex.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAddArrow -win $_nReg1 -color RED -width 2 -head "10,20" -tail
"30,20"
```

regAddParagraph

Description

Adds a paragraph or updates an existing paragraph.

Syntax

```
regAddParagraph [-win window] -fcolor colorID -bcolor colorID
-font fontID -bbox "x1,y1" "x2,y2" -text textString
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

`-fcolor` *`colorID`*

Specify the foreground color of paragraph.

`-bcolor` *`colorID`*

Specify the background color of paragraph.

`-font` *`fontID`*

Specify the font of paragraph.

`-bbox` *`“x1,y1” “x2,y2”`*

Specify bounding box of paragraph to be created.

`-text` *`textString`*

Specify the paragraph content.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAddParagraph -win $_nReg1 -fcolor BLACK -bcolor WHITE -font
Fixed12 -bbox "10,10" "100,100" -text "this is a test"
```

regLock

Description

Locks register window to prevent any modification.

Syntax

```
regLock [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regLock -win $_nReg1
```

regUnlock

Description

Unlocks register window to continue modification.

Syntax

```
regUnlock [-win window]
```

Arguments

-win window

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regUnlock -win $_nReg1
```

regNoRedraw

Description

Stops all register windows from automatically redrawing. When NoRedraw mode is turned on, register windows will redraw only when executing the regRedraw command. Currently, this NoRedraw scheme applies only to signal content updates. For other actions, register window will still redraw automatically.

Syntax

```
regNoRedraw
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regNoRedraw
```

regRedraw

Description

Forces register window to redraw. Can only be used after calling regNoRedraw.

Syntax

```
regRedraw [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regRedraw -win $_nReg0
```

Widget Objects

regAddWidget

Description

Adds a widget object to *nRegister* window.

Syntax

```
regAddWidget [-win window] -type type -bbox "x1,y1" "x2,y2"
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-type type`

Widget type of the new object. Possible values are **Push Button**, **Text**, and **Option Menu**.

`-bbox "x1,y1" "x2,y2"`

Bounding box of the newly created widget.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAddWidget -win $_nReg0 -type OptMenu -bbox "228,141" "332,176"
```

regWidgetAttributes

Description

Set label, command, or menu items of a widget.

Syntax

```
regWidgetAttributes [-win window] -type type -obj objID
[-label labelString] [-tclCmd tclCommand] [-items itemList]
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-type type`

Widget type of the new object. Possible values are **Push Button**, **Text**, and **Option Menu**.

`-obj objID`

Object ID of the activated widget.

`-label labelString`

Label string of the widget.

`-tclCmd tclCommand`

Tcl command associated to the widget.

`-items itemList`

Menu items for an option menu widget.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regWidgetAttributes -win $_nReg0 -type PushBtn -obj $doObj0 -
label Button3 -tclCmd {wvCreateWindow}
```

regWidgetEvent

Description

Triggers previously defined event associated with a widget object.

Syntax

```
regWidgetEvent [-win window] -type type -obj objID
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-type type`

Object type of the widget object. Possible values are **Push Button**, **Text**, and **Option Menu**.

`-obj` *`objID`*

Object ID of the widget object.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regWidgetEvent -win $_nReg0 -type PushBtn -obj $doObj0
```

Search

regPrev

Description

Jumps to previous value change of selected signals in register window.

Syntax

```
regPrev [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regPrev -win $_nReg0
```

regNext

Description

Jumps to next value change of selected signals in register window.

Syntax

```
regNext [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regNext -win $_nReg0
```

regGoToTime

Description

Navigates to a specified time for all signals.

Syntax

```
regGoToTime [-win window] Time
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`Time`

Time to jump to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regGoToTime -win $_nReg0 30
```

regShiftSignalTime

Description

Specifies time offset from current time for a list of signal objects.

Syntax

```
regShiftSignalTime [-win window] -timeOffset offset -objList
objID1 objID2 objID3...
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-timeOffset` *`offset`*

Time offset to be applied to the signal.

`-objList` *`objID1 objID2 objID3...`*

A list of object IDs of signals to be changed.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regShiftSignalTime -win $_nReg0 -timeOffset 234.234 -objList
$doObj1 $doObj2 $doObj3
```

Select

regAddSelect

Description

Adds selected object by specified point to selection set.

Syntax

```
regAddSelect [-win window] -point "x,y"
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-point "x,y"`

Specify the point to select object.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regAddSelect -win $_nReg0 -point "10,20"
```

regFitAll

Description

Moves all objects while keeping their relative position to the upper left corner of register window.

Syntax

```
regFitAll [-win window]
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regFitAll -win $_nReg0
```

regSelect

Description

Select objects by specified area or specified point.

Syntax

```
regSelect [-win window] -bbox "x1,y1" "x2,y2" | -point "x,y"
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-bbox "x1,y1" "x2,y2"`

Specify the bounding box of the area to be selected.

`-point "x,y"`

Specify the point to select object.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSelect -win $_nReg0 -point "10,20"
regSelect -win $_nReg0 -bbox "10,20" "100,100"
```

regSelectAll

Description

Allows you to select all objects in register window.

Syntax

```
regSelectAll [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSelectAll -win $_nReg0
```

regUnselectAll

Description

Allows you to unselect all objects in current selection set.

Syntax

```
regUnselectAll [-win window]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regUnselectAll -win $_nReg0
```

Configuration

regSetRadix

Description

Allows you to set radix or notation of a signal specified by point.

Syntax

```
regSetRadix [-win window] -point "x,y" [-radix Bin|Oct|Hex|
Dec|Ascii] | [-notation USign|Sign|2Comp|1Comp|SMag]
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-point "x,y"`

Specify signal to change radix or notation by giving coordinates.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSetRadix -win $_nReg0 -radix Bin -point "88,138"
```

regSetAttributes

Description

Allows you to set display attributes of selected objects.

Syntax

```
regSetAttributes [-win window] -type SigVar [-signalColor colorID]
[-valueColor colorID] [-borderColor colorID] [-font fontID]
regSetAttributes [-win window] -type Paragraph
[-fcolor colorID] [-bcolor colorID] [-font fontID]
regSetAttributes [-win window] -type Shape
[-color colorID] [-width width]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

`-signalColor` *`colorID`*

Specify color of signal name for signal objects.

`-valueColor` *`colorID`*

Specify color of value name for signal objects.

`-borderColor` *`colorID`*

Specify color of border for signal objects.

`-fcolor` *`colorID`*

Specify foreground color for paragraph objects.

`-bcolor` *`colorID`*

Specify background color for paragraph objects.

`-color` *`colorID`*

Specify color for shape objects.

`-font` *`fontID`*

Specify font for signal objects or paragraph objects.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSetAttributes -win $_nReg0 -type Shape -color ID_BLUE6 -width
1
```

regSetSignalAttr

Description

Allows you to set display content or attributes of a signal based on its object ID.

Syntax

```
regSetSignalAttr [-win window] -obj objID [-signal signalName]
[-value valueString] [-signalColor colorID] [-valueColor colorID]
[-borderColor colorID] [-font fontID]
```

Arguments

`-win window`

Specify the window ID of the invoking register window.

`-obj objID`

The object ID of the signal to operate on.

`-signal signalName`

Specify signal's name to be displayed.

`-value valueString`

Specify signal's value to be displayed.

`-signalColor colorID`

Specify the color of signal name.

`-valueColor colorID`

Specify the color of signal value.

`-borderColor colorID`

Specify border color of this signal object.

`-font fontID`

Specify the font of this signal object.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regSetSignalAttr -win $_nReg0 -obj $doObj2 -signal {MY_NAME} -
value ZERO -signalColor ID_YELLOW5 -valueColor ID_BLUE5 -
borderColor ID_WHITE -font "Fixed 14"
```

regOption

Description

Allows you to set register window display option.

Syntax

```
regOption [-win window] [-hier on|off] [-lead on|off] [-showTrans
on|off] [-autoExpand on|off] [-searchBy
ByChange|ByRising|ByFalling] [-valueSpace on|off]
[-valueSpaceChar c]
```

Arguments

`-win` *`window`*

Specify the window ID of the invoking register window.

`-hier on|off`

Turn on or turn off hierarchical signal name display option.

`-lead on|off`

Turn the display of leading zero signal names *on* or *off*.

`-showTrans on|off`

Specify **ShowTransition** option for signal display in register window.

`-autoExpand on|off`

Specify **AutoExpand** option for signal display in register window.

`-searchBy ByChange|ByRising|ByFalling`

Specify searching rule of signal values.

`-valueSpace on|off`

Specify whether to turn *on* or *off* value space insertion option.

`-valueSpaceChar c`

Specify which character to use as the value space character.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regOption -win $current_window -hier on
```

Print

regPrint

Description

Prints the register window content.

Syntax

```
regPrint [-win window]
```

Arguments

`-win` *window*

Specify the window ID of the invoking register window.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
regPrint
```

Siloti

General

srcInvokeVerdi

Description

Invokes Verdi mode from the Siloti console.

Syntax

```
srcInvokeVerdi
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
srcInvokeVerdi
```

sidSaveDesignSnapshot

Description

Saves a session file containing a snapshot of the current Siloti environment (design, FSDB files, symbol libraries, novas.rc resource file, restore file and command line options file).

Syntax

```
sidSaveDesignSnapshot filename/_hardcopy.ses -keynote {invoke
directory: dir command line: info current trace: scope memo: info}
```

Arguments

`filename/_hardcopy.ses`

Specify the name for the session file including directory path. The _hardcopy.ses is a temporary file and will be removed when the save session is complete.

`-keynote {invoke directory: dir command line: info current trace: scope memo: info}`

Specify additional information about the design, such as the invoking directory, the command line options, the current tracing scope and general notes.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidSaveDesignSnapshot /myproject/debug.ses/_hardcopy.ses -keynote
{invoke directory: /home/proj command line: -lib work -verdi
current trace: tb_CPUsystem memo:}
```

Essential Signal Analysis

sidEsaDDT

Description

Specifies the options for essential signal analysis.

Syntax

```
sidEsaDDT -db esdb_filename [-dump_seq_libcell_bound 0|1]
[-exclude module] [-excludefile filename] [-info N] [-libcell 0|1]
[-optimalA filename] [-xpinfile filename]
```

Arguments

`-db esdb_filename`

Dumps the Essential Signal Database to the specified file.

`-dump_seq_libcell_bound 0|1`

When this option is set to *1*, dump all signals connected to the sequential library cells. The default is *0*.

`-exclude module`

Excludes the signals in the specified module and dumps its output ports only.

`-excludefile filename`

Excludes the signals in the specified modules in the file and dumps their output ports only.

`-info N`

Specifies the information level for the log. The default is *0*.

`-libcell 0|1`

When this option is set to *1*, dump the signals connected to the library cell as the essential signals. The default is *0*.

`-optimalA filename`

Specifies the text file containing a list of proposed modules to suppress to obtain the optimized Essential Signal Analysis results.

`-xpinfile filename`

Includes the specified module or cell ports in the file.

Value Returned

None.

Example

```
sidEsaDDT -db es
```

Data Expansion

sidDEToggle

Description

Toggle Data Expansion *on* or *off*.

Syntax

```
sidDEToggle ON|OFF
```

Arguments

`ON|OFF`

Turn Data Expansion *on* or *off*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidDEToggle ON
```

sidDESetup

Description

Enables Data Expansion.

Syntax

```
sidDESetup [[-start_time time -end_time time] | [-full_time]]
[-crdbfile filename | -crdb | -g2r_mapping name] [-force filename]
[-on_demand] [-de_enforce] [-multiFSDB]
```

Arguments

`-start_time time`

Specify the start time for Data Expansion.

`-end_time time`

Specify the end time for Data Expansion.

`-full_time`

Use the full time range for Data Expansion.

`-on_demand`

Enable automatic time window mode.

`-de_enforce`

Enable Data Expansion for a large time window.

`-crdbfile` *`filename`*

When specified, use the specified correlation database for gate/rtl mapping.

`-crdb`

When specified, use the correlation database in memory for gate/rtl mapping.

`-g2r_mapping` *`filename`*

When specified, use the specified mapping file(s) for gate/rtl mapping.

`-force` *`filename`*

Specify the force file name.

`-multiFSDB`

Enable Data Expansion support for all Essential Signal Dump (ESD) FSDB files in the same session.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidDESetup -start_time 0 -end_time 1000
```

sidUpdateSignalValue

Description

Updates the signal values in all Siloti/Verdi windows (e.g. *nTrace*, *nWave*, etc.).

Syntax

```
sidUpdateSignalValue
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidUpdateSignalValue
```

Visibility Analysis

sidVisibilityAnalysis

Description

Specify the options for visibility analysis which analyzes the visibility possible with different essential signal files or FSDB files.

Syntax

```
sidVisibilityAnalysis [-dsfile filename|-fsdbfile filename] [-del
string] [-analyze_scope {level} {scope}] [-analyze_scope_file
{filename}] [-exclude_module {module}] [-exclude_module_file
{filename}] [-exclude_nd_module] [-crdbfile filename|-crdb|-
g2r_mapping name] [-va_check_sig signal] [-va_report filename] [-
miss] [-Drv] [-nDrv] [-pDrv] [-set_ff_computable]
```

Arguments

`-dsfile` *`filename`* | `-fsdbfile` *`filename`*

Specify the input essential signal file name or the FSDB file name to analyze.

`-del` *`string`*

Specify the hierarchical delimiter.

`-analyze_scope` *`{level} {scope}`*

Specify one or more level/scope pairs to analyze. The scope should include the full hierarchical path.

`-analyze_scope_file` *`{filename}`*

Specify a filename containing one or more level/scope pairs to analyze.

`-exclude_module` *`{module}`*

Specify one or more module names to exclude from the analysis.

`-exclude_module_file` *`{filename}`*

Specify a filename containing one or more module names to exclude from the analysis.

`-exclude_nd_module`

When specified, the modules (library cells) whose outputs cannot be computed will be excluded.

`-crdbfile` *`filename`*

When specified, use the specified correlation database for gate/rtl mapping.

`-crdb`

When specified, use the correlation database in memory for gate/rtl mapping.

`-g2r_mapping` *`filename`*

When specified, use the specified mapping file(s) for gate/rtl mapping.

`-va_check_sig` *`signal`*

Identify the missing essential signals for the specified signal (include full hierarchical path). Must be used with -miss.

`-va_report` *`filename`*

Specify the file name to save the visibility analysis report results to.

`-miss`

When specified, the missing essential signals are saved to a file. The same file name as the -va_report is used with a .miss_eslist extension.

`-Drv`

When specified, derivable signals are saved to a file. The same file name as the -va_report is used with a .Drv extension.

`-nDrv`

When specified, non-derivable signals are saved to a file. The same file name as the -va_report is used with a .nDrv extension.

`-pDrv`

When specified, potentially derivable signals are saved to a file. The same file name as the -va_report is used with a .pDrv extension.

`-set_ff_computable`

When specified, flip-flops are computable.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidVisibilityAnalysis -dsfile myesa.lst -analyze_scope {0}
{tb.top.cpu} -exclude_module {ALUB}
sidVisibilityAnalysis -fsdbfile myefile.fsdb -va_check_sig
tb.CPUsystem.i_CPUsystem.i_CPU.data -va_report output.txt -miss
```

Gate/RTL Correlation

sidCRAnaCurrentUncorrelate

Description

Shows the current uncorrelated object in the design tree browser.

Syntax

```
sidCRAnaCurrentUncorrelate -gate | -rtl
```

Arguments

`-gate`

Show the current uncorrelated object for the gate design.

`-rtl`

Show the current uncorrelated object for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaCurrentUncorrelate -gate
```

sidCRAnaExpandTreeByLevel

Description

Allows you to expand the design tree browser.

Syntax

```
sidCRAnaExpandTreeByLevel -gate | -rtl -all | -level levelNum
```

Arguments

`-all`

Expand the whole design tree browser.

`-gate`

Expand the design tree browser for the gate design.

`-level` *`levelNum`*

Specify the level to expand the design tree browser.

`-rtl`

Expand the design tree browser for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaExpandTreeByLevel -gate -level 3
sidCRAnaExpandTreeByLevel -rtl -all
```

sidCRAnaFilterAdd

Description

Adds a filter rule to the end of the rule table.

Syntax

`sidCRAnaFilterAdd -gate | -rtl` *`{rule}`*

Arguments

`-gate`

Add a filter rule from the gate design to the end of the rule table.

`-rtl`

Add a filter rule from the RTL design to the end of the rule table.

`{rule}`

Specify the filter rule to add.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaFilterAdd -gate {^n%d}
```

sidCRAnaFilterApply

Description

Apply all filter rules from the rule table and save into the rc file.

Syntax

```
sidCRAnaFilterApply -gate | -rtl
```

Arguments

`-gate`

Apply all filter rules for the gate design.

`-rtl`

Apply all filter rules for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaFilterApply -gate
```

sidCRAnaFilterDel

Description

Deletes a filter rule from the specified position in the rule table.

Syntax

```
sidCRAnaFilterDel -gate | -rtl -pos value
```

Arguments

`-gate`

Delete a filter rule from the gate design.

`-pos` *`value`*

Specify the position in the rule table. Value must be a positive integer.

`-rtl`

Delete a filter rule from the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaFilterDel -rtl -pos 5
```

sidCRAnaFilterDelAll

Description

Deletes all filter rules from the rule table.

Syntax

```
sidCRAnaFilterDelAll -gate | -rtl
```

Arguments

`-gate`

Delete all rules for the gate design.

`-rtl`

Delete all rules for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaFilterDelAll -rtl
```

sidCRAnaFirstUncorrelate

Description

Shows the first uncorrelated object in the design tree browser.

Syntax

```
sidCRAnaFirstUncorrelate -gate | -rtl
```

Arguments

`-gate`

Show the first uncorrelated object for the gate design.

`-rtl`

Show the first uncorrelated object for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaFirstUncorrelate -gate
```

sidCRAnaFullHierName

Description

Shows full hierarchical names or object names on the design tree browser.

Syntax

```
sidCRAnaFullHierName -on | -off
```

Arguments

`-off`

Show the object names.

`-on`

Show the full hierarchical names.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaFullHierName -on
```

sidCRAnaGotoUpHier

Description

Navigates to the upper hierarchy of the selected objects in the design tree browser.

Syntax

```
sidCRAnaGotoUpHier -gate | -rtl
```

Arguments

`-gate`

Go to the upper hierarchy of the selected objects for the gate design.

`-rtl`

Go to the upper hierarchy of the selected objects for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaGotoUpHier -rtl
```

sidCRAnaHighlightSource

Description

Highlights the selected objects in the *nTrace* design tree browsers of both the gate and RTL designs.

Syntax

```
sidCRAnaHighlightSource
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaHighlightSource
```

sidCRAnaImport

Description

Specifies the import data for the *Correlation Analyzer* window.

Syntax

```
sidCRAnaImport -gate_crdb reg_sig | all_sig [seq_inst] [seq_port]
| -gate_fsdb filename | -gate_txt filename
sidCRAnaImport -rtl_crdb reg_sig | all_sig [seq_inst] [seq_port] |
-rtl_fsdb filename | -rtl_txt filename
```

Arguments

`all_sig`

Import all signals in the design. Cannot be used with **reg_sig**.

`-gate_crdb reg_sig | all_sig`

Import data from the CRDB for the gate design using the **reg_sig** or **all_sig** option.

`-gate_fsdb` *`filename`*

Import signals in the specified FSDB file for gate design.

`-gate_txt` *`filename`*

Import signals in the specified text file for gate design.

`reg_sig`

Import register signals only. Cannot be used with **all_reg**.

`-rtl_crdb reg_sig | all_sig`

Import data from the CRDB for the RTL design using the **reg_sig** or **all_sig** option.

`-rtl_fsdb` *`filename`*

Import signals in the specified FSDB file for RTL design.

`-rtl_txt` *`filename`*

Import signals in the specified text file for RTL design.

`seq_inst`

Import sequential library cell instances. This option can be used with **reg_sig**, **all_sig**, or **seq_port**.

`seq_port`

Import sequential library cell ports. This option can be used with **reg_sig**, **all_sig**, or **seq_inst**.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaImport -gate_crdb reg_sig seq_inst seq_port -rtl_crdb
all_sig
sidCRAnaImport -gate_crdb all_sig seq_port -rtl_fsdb filename
```

sidCRAnaLastUncorrelate

Description

Shows the last uncorrelated object in the design tree browser.

Syntax

```
sidCRAnaLastUncorrelate -gate | -rtl
```

Arguments

`-gate`

Show the last uncorrelated object for the gate design.

`-rtl`

Show the last uncorrelated object for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaLastUncorrelate -gate
```

sidCRAnaNextUncorrelate

Description

Shows the next uncorrelated object in the design tree browser.

Syntax

```
sidCRAnaNextUncorrelate -gate | -rtl
```

Arguments

`-gate`

Show the next uncorrelated object for the gate design.

`-rtl`

Show the next uncorrelated object for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaNextUncorrelate -gate
```

sidCRAnaPrevUncorrelate

Description

Shows the previous uncorrelated object in the design tree browser.

Syntax

```
sidCRAnaPrevUncorrelate -gate | -rtl
```

Arguments

`-gate`

Show the previous uncorrelated object for the gate design.

`-rtl`

Show the previous uncorrelated object for the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaPrevUncorrelate -gate
```

sidCRAnaSaveResult

Description

Saves the correlated signals into a file.

Syntax

```
sidCRAnaSaveResult filename [-dumprtl] [-dumpgate]
[-dumpgate_uncor]
```

Arguments

`-dumpgate`

Dump all correlated gate-level signals into a *.gate* file.

`-dumpgate_uncor`

Dump all uncorrelated gate signals into a *.gate.uncor* file.

`-dumprtl`

Dump the correlated RTL signals into a *.rtl* file.

`filename`

Specify the file name to save the mapping results to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaSaveResult -dumpgate correlated.lst
```

sidCRAnaSelect

Description

Selects an object in the design tree browser.

Syntax

sidCRAnaSelect -gate | -rtl *{full_hier_name}*

Arguments

`{full_hier_name}`

Specify the full hierarchical name of the object to select.

`-gate`

Select an object from the gate design.

`-rtl`

Select an object from the RTL design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaSelect -gate {top.u1.u2}
```

sidCRAnaShowAll

Description

Shows all input objects in the design tree browsers.

Syntax

```
sidCRAnaShowAll
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaShowAll
```

sidCRAnaShowFilter

Description

Show/Hide the filter fields.

Syntax

```
sidCRAnaShowFilter -on | -off
```

Arguments

`-off`

Hide the filter fields.

`-on`

Show the filter fields.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaShowFilter -off
```

sidCRAnaShowUncorrelate

Description

Displays the uncorrelated objects in the design tree browsers.

Syntax

```
sidCRAnaShowUncorrelate
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRAnaShowUncorrelate
```

sidCRCloseAnalyzer

Description

Closes the *Correlation Analyzer* window.

Syntax

```
sidCRCloseAnalyzer
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRCloseAnalyzer
```

sidCROpenAnalyzer

Description

Opens the *Correlation Analyzer* window.

Syntax

```
sidCROepnAnalyzer
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROpenAnalyzer
```

sidCRRtl2Gate

Description

Creates a mapping file to correlate the given RTL signals to the corresponding gate-level signals.

NOTE: To enable this command, the environment variable *CR_ANALYZER* must be set to *1*. The default is *0*.

Syntax

```
sidCRRtl2Gate -txt input_signal_list [-dumprtl] [-dumpgate]
[-dumprtl_uncor] [-dumpgate_eq] [-flatten_vec]
-crlRst output_correlation_file
```

Arguments

`-crlRst` *`output_correlation_file`*

Specify the output file name.

`-dumpgate`

Dump the correlated gate-level signals into a *.gate* file.

`-dumpgate_eq`

Dump the equivalent signals of the correlated gate-level signals into the output file.

`-dumprtl`

Dump the correlated RTL signals into a *.rtl* file.

`-dumprtl_uncor`

Dump the uncorrelated RTL signals into a separate *.rtl.uncor* file. For example, *output_correlation_file.rtl.uncor*.

`-flatten_vec`

Output signals in the flattened form. For example, when this option is specified, the *tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[7:0]* signals appear as:

```
tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[0]
tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[1]
```

```
    tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[2]
    tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[3]
    tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[4]
    tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[5]
    tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[6]
    tb_CPUsystem.i_CPUsystem.i_CPU.i_ALUB.IXR[7]
-txt input_signal_list
```

Specify the input file name for the file with RTL signals and/or scopes.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRRtl2Gate -txt rtl_signals.txt -dumprtl -dumpgate \
-dumprtl_uncor -crlRst correlated_gate_signals.txt.
```

sidCRTraceAndCorrelate

Description

Correlates the specified gate level signals to the RTL counterparts. When the correlation from the CRDB file for a signal does not exist, this command traces the drivers and checks whether any driving signal has a correlation in the RTL. The tracing stops at a sequential element or a primary input.

NOTE: Before invoking this command, a gate level design should be loaded in the Siloti system and Behavior Analysis should be complete for the gate level design. A CRDB file should then be prepared for the gate level and RTL designs.

Syntax

```
sidCRTraceAndCorrelate -rtl -txt input_signal_list -user_format
-crlRst output_correlation_file [-dumpgate_uncor]
[-dumprtl_uncor] [-max_rst num]
```

Arguments

`-crlRst output_correlation_file`

Specify the output file name.

`-dumpgate_uncor`

Dump the uncorrelated gate level signals into a separate *.gate.uncor* file. For example, *output_correlation_file.gate.uncor*.

`-dumprtl_uncor`

Dump the uncorrelated RTL level signals into a separate *.rtl.uncor* file. For example, *output_correlation_file.rtl.uncor*.

`-max_rst` *`num`*

Specify the maximum number of results for a single input signal. The default is *1*.

`-rtl`

Specify the RTL level for tracing.

`-txt` *`input_signal_list`*

Specify the input file name.

`-user_format`

Output in a readable text file.

Example

```
sidCRTraceAndCorrelate -txt input_signal_list -crlRst
output_correlation_file -dumpgate_uncor -max_rst 100
sidCRTraceAndCorrelate -rtl -txt input_signal_list -crlRst
output_correlation_file -dumprtl_uncor -max_rst 100
```

Mapping Rules

sidCROpenNamingRule

Description

Allows you to open the *Naming Rule* form.

Syntax

```
sidCROpenNamingRule
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROpenNamingRule
```

sidCRCloseNamingRule

Description

Closes the *Naming Rule* form.

Syntax

```
sidCRCloseNamingRule
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRCloseNamingRule
```

sidCRSaveNamingRules

Description

Saves the naming rules to a file.

Syntax

```
sidCRSaveNamingRules filename [-preference]
```

Arguments

`filename`

Specify the file name to save the naming rules to.

`-preference`

When specified, the file name will be saved to the *novas.rc* resource file as the default naming rule.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRSaveNamingRules rules.txt
```

sidCRApplyNamingRules

Description

Applies the naming rules into the correlation database.

Syntax

```
sidCRApplyNamingRules
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRApplyNamingRules
```

sidCRGenNamingRules

Description

Applies the naming rules into the correlation database and close the *Naming Rule* form.

Syntax

```
sidCRGenNamingRules
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRGenNamingRules
```

sidCROptAddMod

Description

Open a *Module Rule* form.

Syntax

```
sidCROptAddMod {module}
```

Arguments

`{module}`

Specify the module to add.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptAddMod {ALUB}
```

sidCRModCloseNamingRule

Description

Closes the *Module Rule* form.

Syntax

```
sidCRModCloseNamingRule
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRModCloseNamingRule
```

sidCRModGenNamingRule

Description

Applies the naming rules in the specific module and close the *Module Rule* form.

Syntax

```
sidCRModGenNamingRule
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRModGenNamingRule
```

sidCROptEditMod

Description

Edits the specific module rule.

Syntax

```
sidCROptEditMod {module}
```

Arguments

`{module}`

Specify the module to edit.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptEditMod {ALUB}
```

sidCROptDelMod

Description

Deletes the specific module rule.

Syntax

```
sidCROptDelMod {module}
```

Arguments

`{module}`

Specify the module to delete.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptDelMod {ALUB}
```

sidCROptRegExpAdd

Description

Adds a naming rule to the end of the rule table.

Syntax

```
sidCROptRegExpAdd [-mod] {regexprule}
```

Arguments

`-mod`

When specified, the naming rule will be added into the specific module.

`{regexprule}`

Specify the naming rule to add.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptRegExpAdd {^system => dut}
```

sidCROptRegExpDelAll

Description

Deletes all naming rules from the rule table.

Syntax

```
sidCROptRegExpDelAll
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptRegExpDelAll
```

sidCROptRegExpDel

Description

Deletes a naming rule from the specified position in the rule table.

Syntax

```
sidCROptRegExpDel [-mod] -pos value
```

Arguments

`-mod`

When specified, the naming rule will be deleted from the specific module.

`-pos value`

Specify the position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptRegExpDel -mod -pos 2
```

sidCROptRegExpInsert

Description

Inserts a naming rule into the specified position of the rule table. The rule can not be duplicated.

Syntax

```
sidCROptRegExpInsert [-mod] {regexprule} -pos value
```

Arguments

`-mod`

When specified, the naming rule will be added to the specified position into the specific module.

`{regexprule}`

Specify the naming rule to add.

`-pos value`

Specify the position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptRegExpInsert {^system => dut} -pos 2
```

sidCROptRegExpMoveDown

Description

Moves the naming rule down by one row from the specified position in the rule table.

Syntax

```
sidCROptRegExpMoveDown [-mod] -pos value
```

Arguments

`-mod`

When specified, the starting position in the rule table is for the specific module rule.

`-pos value`

Specify the starting position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptRegExpMoveDown [-mod] -pos 3
```

sidCROptRegExpMoveUp

Description

Moves the naming rule up by one row from the specified position in the rule table.

Syntax

```
sidCROptRegExpMoveUp [-mod] -pos value
```

Arguments

`-mod`

When specified, the starting position in the rule table is for the specific module rule.

`-pos value`

Specify the starting position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptRegExpMoveUp -pos 3
```

sidCROptRegExpReplace

Description

Replaces a naming rule into the specified position of the rule table. The rule can not be duplicated.

Syntax

```
sidCROptRegExpReplace [-mod] {regexprule} -pos value
```

Arguments

`-mod`

When specified, the position to replace in the rule table is for the specific module rule.

`{regexprule}`

Specify the naming rule to replace.

`-pos value`

Specify the position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptRegExpReplace {^system => dut} -pos 2
```

sidCROpt

Description

Enables you to setup the pre-defined rules for hierarchical separator, tri-state device and register correlation.

Syntax

```
sidCROpt [-mod] -hierSep {string} | -tri {string} | -reg {string}
```

Arguments

`-mod`

When specified, the pre-defined rules will be added into the specific module.

`-hierSep [string]`

Specify the hierarchical separator string for flattened signals/instances. Only one character is allowed.

`-tri {string}`

Specify the tri-state string format for tri-state instances and output nets.

`-reg {string}`

Specify the register string format for register instances and output nets.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROpt -hierSep {!}
```

sidCROptHierChAdd

Description

Adds a hierarchical change rule to the end of the rule table.

Syntax

```
sidCROptHierChAdd {hierrule}
```

Arguments

`{hierrule}`

Specify the hierarchical change rule to add.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptHierChAdd {system => dut}
```

sidCROptHierChDelAll

Description

Deletes all hierarchical change rules from the rule table.

Syntax

```
sidCROptHierChDelAll
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptHierChDelAll
```

sidCROptHierChDel

Description

Deletes a hierarchical change rule from the specified position in the rule table.

Syntax

```
sidCROptHierChDel -pos value
```

Arguments

`-pos value`

Specify the position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptHierChDel -pos 4
```

sidCROptHierChInsert

Description

Inserts a hierarchical change rule into the specified position of the rule table. The rule can not be duplicated.

Syntax

```
sidCROptHierChInsert {hierrule} -pos value
```

Arguments

`{hierrule}`

Specify the hierarchical change rule to insert.

`-pos` *`value`*

Specify the position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptHierChInsert {system => dut} -pos 2
```

sidCROptHierChMoveDown

Description

Moves the hierarchical change rule down one row from the specified position in the rule table.

Syntax

```
sidCROptHierChMoveDown -pos value
```

Arguments

`-pos` *`value`*

Specify the starting position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptHierChMoveDown -pos 3
```

sidCROptHierChMoveUp

Description

Moves the hierarchical change rule up one row from the specified position in the rule table.

Syntax

```
sidCROptHierChMoveUp -pos value
```

Arguments

`-pos value`

Specify the starting position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptHierChMoveUp -pos 3
```

sidCROptHierChReplace

Description

Replaces a hierarchical change rule into the specified position of the rule table. The rule can not be duplicated.

Syntax

```
sidCROptHierChReplace {hierrule} -pos value
```

Arguments

`{hierrule}`

Specify the hierarchical change rule to replace.

`-pos value`

Specify the position in the rule table. Value must be a positive integer.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptHierChReplace {system => dut} -pos 2
```

sidCROptScanRule

Description

Setup the scan cell rules for scan clock, scan input and scan enable pins.

Syntax

```
sidCROptScanRule -sclk {strings} | -si {strings} | -se {strings}
```

Arguments

`-sclk {strings}`

Specify the scan clock pin names for the scan cell. Use | to separate multiple names.

`-si {strings}`

Specify the scan input pin names for the scan cell. Use | to separate multiple names.

`-se {strings}`

Specify the scan enable pin names for the scan cell. Use | to separate multiple names.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROptScanRule -sclk {SCAN_CLK | SCLK}
```

Test Correlation

sidCRCloseTestCor

Description

Closes the *Test Correlation* window.

Syntax

```
sidCRCloseTestCor
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRCloseTestCor
```

sidCROpenTestCor

Description

Opens the *Test Correlation* window.

Syntax

```
sidCROpenTestCor -allreg | -fsdb filename | -txt filename
```

Arguments

`-allreg`

When specified, all registers will be mapped from the gate to the RTL.

`-fsdb` *`filename`*

When specified, all signals in the specified FSDB file will be mapped from the gate to the RTL.

`-txt` *`filename`*

When specified, all signals in the specified text file will be mapped from the gate to the RTL.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROpenTestCor -allreg
```

sidCRSetFilter

Description

Filters out the input signals in the *Test Correlation* window.

Syntax

```
sidCRSetFilter {string}
```

Arguments

`{string}`

Specify the regular expression strings to filter. Use a space to separate multiple strings.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRSetFilter {i_ALUB}
```

sidCRFilterAdd

Description

Adds the string to filter out the input signals in the Test Correlation window.

Syntax

```
sidCRFilterAdd {string}
```

Arguments

`{string}`

Specify the regular expression strings to filter. Use a space to separate multiple strings.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRFilterAdd {i_ALUB}
```

sidCRFilterDel

Description

Deletes the filter string from the *Test Correlation* window.

Syntax

```
sidCRFilterDel -pos value
```

Arguments

`-pos value`

Specify the value as a positive integer for the row in the filter table of the test mapping window to delete.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRFilterAdd -pos 5
```

sidCRFilterDelAll

Description

Deletes all filter strings from the *Test Correlation* window.

Syntax

```
sidCRFilterDelAll
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRFilterAddDel
```

sidCRShowAll

Description

Displays all input signals in the *Test Correlation* window.

Syntax

```
sidCRShowAll
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRShowAll
```

sidCRShowUncorrelate

Description

Shows the unmapped input signals in the *Test Correlation* window.

Syntax

```
sidCRShowUncorrelate
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRShowUncorrelate
```

sidCRTestAll

Description

Allows you to test all input signals in the *Test Correlation* window.

Syntax

```
sidCRTestAll
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRTestAll
```

sidCRTestUncorrelate

Description

Test all unmapped input signals in the *Test Correlation* window.

Syntax

```
sidCRTestUncorrelate
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRTestUncorrelate
```

sidCRSaveTestCorResult

Description

Saves the correlated signals into a file.

Syntax

```
sidCRSaveTestCorResult [-dumprtl] [-dumpgate] filename
```

Arguments

`-dumprtl`

Dump the correlated RTL signals into a *.rtl* file.

`-dumpgate`

Dump the correlated gate-level signals into a *.gate* file.

`filename`

Specify the file name to save the mapping results to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRSaveTestCorResult -dumpgate mapping.lst
```

Correlation Console

sidCROpenConsole

Description

Opens the *Correlation Console* window.

Syntax

```
sidCROpenConsole
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROpenConsole
```

sidCRCloseConsole

Description

Closes the *Correlation Console* window.

Syntax

```
sidCRCloseConsole
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRCloseConsole
```

sidCRClearHistory

Description

Clears the history table in the *Correlation Console* window.

Syntax

```
sidCRClearHistory
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRClearHistory
```

sidCRCorrelate

Description

Enables you to correlate a gate level signal/scope to the RTL design or a RTL signal/scope to a gate design.

Syntax

```
sidCRCorrelate [-mapBtn] {string}
```

Arguments

`-mapBtn`

When specified, correlate the signal in the text field next to the **Correlate** button. If not specified, correlate the signal that is dropped in the history table.

`{string}`

Specify the hierarchical signal name to correlate.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRCorrelate {system.i_ALUB.clk}
```

sidCRGenRst

Description

Creates a mapping file that correlates the RTL to the gate-level design.

Syntax

```
sidCRGenRst [-allreg|-fsdb filename|-txt filename] [-dumprtl]
[-dumprtl_eq][-dumpgate] -crlRst filename [-excludeModule
filename]
[-top_inport]
```

Arguments

`-allreg|-fsdb filename|-txt filename`

When **-allreg** is specified, all registers will be mapped from the gate to the RTL.

When **-fsdb** is specified, all signals in the specified FSDB file will be mapped from the gate to the RTL.

When **-txt** is specified, all signals in the specified text file will be mapped from the gate to the RTL.

`-crlRst filename`

Save the mapping results to the specified file.

`-dumprtl`

Dump the correlated RTL signals into a *.*rtl* file.

`-dumprtl_eq`

Dump the equivalent signals of the correlated RTL-level signals into the output file.

`-dumpgate`

Dump the correlated gate-level signals into a *.*gate* file.

`-excludeModule` *`filename`*

Signals in the excluded modules will be excluded from the mapping file. All equivalent signals (i.e. same nets or outside nets) of the excluded port signals will be listed in the mapping file.

`-top_inport`

Include input ports that were successfully mapped from the gate-level design to the RTL design in the top scope.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRGenRst -fsdb gate-ess.fsdb -dumprtl -crlRst mapping.rst
```

All gate-level signals in gate-ess.fsdb will be mapped to RTL and the mapping file is saved as mapping.rst. The mapped RTL signals will saved to mapping.rst.rtl.

```
sidCRGenRst -allreg -dumprtl -dumpgate -dumpgate_uncor -dumprtl_eq
-crlRst g2r.map
```

All gate-level register signals in the loaded CRDB are mapped to RTL and the mapping file is saved as `g2r.map`. The equivalent signals of the correlated RTL-level signals are dumped to this file. The mapped RTL signals are saved to `g2r.map.rtl`. The mapped GATE signals are saved to `g2r.map.gate`. The unmapped gate-level register signals are saved to `g2r.map.gate.uncor`.

sidCROpenSlaveProcess

Description

Opens a slave process to load the gate design.

Syntax

```
sidCROpenSlaveProcess -tkName name -clientTkName clientname
[-cmdOpt {options}] [-specBin {bin}] [-appPid 0|1]
```

Arguments

`-appPid 0|1`

When *1* is specified, append the PID (Process ID) to the tkNames of the current process and client process. When *0* is specified, do not append the PID (Process ID) to the tkNames of the current process and client process. The default is *1*.

`-tkName` *`name`*

Specify a tkName for the current process.

`-clientTkName` *`clientname`*

Specify a tkName for the client process.

`-cmdOpt` *`{options}`*

Specify the options for the slave process.

`-specBin` *`{bin}`*

Specify the Siloti executable path.

Value Returned

The master and slave Tk names used if successful; otherwise, returns 0.

Example

```
sidCROpenSlaveProcss -tkName siloti -clientTkName siloti_crl
-cmdOpt {-play test.cmd} -specBin {/home/siloti} -appPid 0
```

sidCRExpandHistory

Description

Expand or collapse the bit-splitting results for the specified row in the history table of the *Correlation Console* window.

Syntax

```
sidCRExpandHistory -pos value
```

Arguments

`-pos` *`value`*

Specify the value as a positive integer for the row in the history table of the *Correlation Console* window to expand or collapse.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRExpandHistory -pos 10
```

sidCRExtractDB

Description

Extracts the information to the correlation database (CRDB).

Syntax

```
sidCRExtractDB -rtl | -gate
```

Arguments

`-rtl`

Specify the extracting abstraction type as RTL.

`-gate`

Specify the extracting abstraction type as gate-level.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRExtractDB -rtl
```

sidCRExtractFSDB

Description

Extracts the correlation database (CRDB) from the FSDB file.

Syntax

```
sidCRExtractFSDB -fsdb filename [-nocase]
```

Arguments

`-fsdb` *`filename`*

Specify the FSDB file name to extract the correlation database from.

`-nocase`

When specified, the extraction will not be case sensitive.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRExtractFSDB -fsdb mysim.fsdb -nocase
```

sidCRLoadDB

Description

Loads the correlation database (CRDB) from a binary file.

Syntax

```
sidCRLoadDB filename
```

Arguments

`filename`

Specify the file name of the correlation database to be loaded.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRLoadDB extracted.crdb
```

sidCRSaveDB

Description

Extracts the information in the correlation database (CRDB) and store it as a binary file.

Syntax

```
sidCRSaveDB filename
```

Arguments

`filename`

Specify the file name to store the information to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRSaveDB extracted.crdb
```

sidCRPrepareDB

Description

Prepares the correlation database (two process mode). This command is only available in the primary Siloti process.

Syntax

```
sidCRPrepareDB
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRPrepareDB
```

sidCRHistory

Description

Highlights the selected scope or signal in the *nTrace* window.

Syntax

```
sidCRHistory -pos selected_row
```

Arguments

`-pos selected_row`

Specify the position of the row selected by cursor.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRHistory -pos 7
```

sidCRShowSchema

Description

Invokes *nSchema* and show the results obtained by the Correlation Console on *nSchema*.

Syntax

```
sidCRShowSchema -pos selected_row
```

Arguments

`-pos selected_row`

Specify the position of the row selected by cursor.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRShowSchema -pos 7
```

sidCRShowFanInSchema

Description

Invokes *nSchema* and show the results obtained by the *Correlation Console* and its fan-in cone on *nSchema*.

Syntax

```
sidCRShowFanInSchema -pos selected_row
```

Arguments

`-pos` *`selected_row`*

Specify the position of the row selected by cursor.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRShowFanInSchema -pos 7
```

sidCRShowFanOutSchema

Description

Invokes *nSchema* and show the results obtained by the *Correlation Console* and its fan-out cone on *nSchema*.

Syntax

```
sidCRShowFanOutSchema -pos selected_row
```

Arguments

`-pos` *`selected_row`*

Specify the position of the row selected by cursor.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRShowFanOutSchema -pos 7
```

sidCROpenDumpResult

Description

Opens a file before dumping data.

Syntax

```
sidCROpenDumpResult {filename}
```

Arguments

{filename}

Specify the file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCROpenDumpResult {dumpFile}
```

sidCRCloseDumpResult

Description

Closes a file after dumping data.

Syntax

```
sidCRCloseDumpResult {filename}
```

Arguments

{filename}

Specify the file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRCloseDumpResult {dumpFile}
```

sidCROpenPrepareDB

Description

Opens the *Prepare CRDB* window.

Syntax

```
sidCRPrepareDB
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRPrepareDB
```

sidCRExitPrepareDB

Description

Closes the *Prepare CRDB* window.

Syntax

```
sidCRExitPrepareDB
```

Arguments

None.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
sidCRExitPrepareDB
```

List X

Window

lxListX

Description

Create the form which display the X/U information.

Syntax

```
lxListX
```

Value Returned

Form ID.

Example

```
lxListX
```

lxClose

Description

Close the form which display the X/U information.

Syntax

```
lxClose
```

Value Returned

None.

Example

```
lxClose
```

File

lxGen

Description

Generate an xloc file with input file name base on the current FSDB file.

Syntax

```
lxGen xlocFileName
```

Arguments

`xlocFileName`

Specify the output xloc file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxGen verilog.fsdb.xloc
```

lxAbortGen

Description

Stop the current generate xloc file process fork by lxGen.

Syntax

```
lxAbortGen
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxAbortGen
```

lxOpenFile

Description

Open an FSDB or xloc file.

Syntax

```
lxOpenFile fileName
```

Arguments

`fileName`

Specify the FSDB or xloc file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxOpenFile verilog.fsdb
```

lxSetStartTime

Description

Set the start time for generated xloc file. The time unit is the original FSDB file time unit.

Syntax

```
lxSetStartTime startTime
```

Arguments

`startTime`

Specify the xloc file start time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxSetStartTime 250
```

lxSetEndTime

Description

Set the end time for generated xloc file. The time unit is the original FSDB file time unit.

Syntax

```
lxSetEndTime endTime
```

Arguments

`endTime`

Specify the xloc file end time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxSetEndTime 6250
```

lxOpenSignalFile

Description

Set the end time for generated xloc file. The time unit is the original FSDB file time unit.

Syntax

```
lxOpenSignalFile signalFile
```

Arguments

`signalFile`

Generate xloc file only for those signals that specified in the signalFile.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxOpenSignalFile kelvin.sig
```

lxSpecifyLogFile

Description

Specify the name of the saved log file.

Syntax

```
lxSpecifyLogFile fileName
```

Arguments

`fileName`

The name of the saved log file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxSpecifyLogFile snapshot_1.rst
```

lxSaveSnapshot

Description

Save the current list data into a file.

Syntax

```
lxSaveSnapshot
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxSaveSnapshot
```

View

lxPlay

Description

Start the List X function and locate the X value at the user-specified time.

Syntax

```
lxPlay [-listbits] [-sig signalName]
```

Arguments

`-listbits`

When specified, only list the bus bits having X values.

`-sig signalName`

Specify the signal or scope's full path name.

Value Returned

1 and the occurring time if successful; otherwise, returns 0.

Example

```
lxPlay -listbits -sig {/tb_CPUsystem/i_CPUsystem/i_pram/
dataout[7:0]}
```

lxSearchPrev

Description

Search backward from the current time to locate previous X's occurring time.

Syntax

```
lxSearchPrev
```

Value Returned

1 and the occurring time if successful; otherwise, returns 0.

Example

```
lxSearchPrev
```

lxSearchNext

Description

Search forward from the current time to locate next X'S occurring time.

Syntax

```
lxSearchNext
```

Value Returned

1 and the occurring time if successful; otherwise, returns 0.

Example

```
lxSearchNext
```

lxFastJump

Description

Directly set the time to the first X occurring time.

Syntax

```
lxFastJump
```

Value Returned

1 and the occurring time if successful; otherwise, returns 0.

Example

```
lxFastJump
```

lxGotoTime

Description

Search the closet X occurring time according to user-specified time. The time unit is the current wave display time unit.

Syntax

```
lxGotoTime time
```

Arguments

`time`

Specify the search time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxGotoTime 350
```

lxSetWidth

Description

Set the minimum width time to filter out some short period 'X'/'U' value noise. The time unit is the current wave display time unit.

Syntax

```
lxSetWidth period
```

Arguments

`period`

Specify the minimum period time to ignore noise.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxSetWidth 350
```

lxStop

Description

Stop the list X function. All the information shown on the *ListX* window is clear.

Syntax

```
lxStop
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
lxStop
```

Configuration

lxSyncWv

Description

Toggle on/off synchronize the ListX time with current *nWave* cursor time mechanism.

Syntax

```
lxSyncWv
```

Value Returned

1 and current status of the synchronization if successful; otherwise, returns 0.

Example

```
lxSyncWv
```

New List X

Window

listxClose

Description

Close the *List X* form.

Syntax

```
listxClose
```

Value Returned

None.

Example

```
listxClose
```

listxListX

Description

Invoke a *List X* form.

Syntax

```
listxListX [-selected|-all]
```

Arguments

`-all`

Display all signals in the *List X* form.

`-selected`

Display only the selected signals in the *List X* form.

Value Returned

None.

Example

```
listxListX
```

File

listxGen

Description

Generate an xloc file with the input file name based on the current FSDB file.

Syntax

```
listxGen
```

Value Returned

None.

Example

```
listxGen
```

listxGenAbort

Description

Stop the current generate X file process.

Syntax

```
listxGenAbort
```

Value Returned

None.

Example

```
listxGenAbort
```

listxGenAddScopeList

Description

Add the scope list from the xloc file.

Syntax

```
listxGenAddScopeList scopeList
```

Arguments

scopeList

Specify the xloc file to add the scope list to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxGenAddScopeList "/TOP/system/transmitter" \
"/TOP/system/receiver"
```

listxGenAddSignalList

Description

Add the signal list to the xloc file.

Syntax

```
lxAddGenSignalList signalList
```

Arguments

signalList

Specify the xloc file to add the signal list to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxGenAddSignalList "win $_nwave3 "/TOP/system/transmitter/b" \
```

```
"/TOP/system/transmitter/c"
```

listxGenDelScopeList

Description

Delete the scope list from the xloc list.

Syntax

```
listxGenDelScopeList [scopeList] [-all]
```

Arguments

`-all`

Delete all scope lists.

`scopeList`

Specify the scope list to delete.

Value Returned

None.

Example

```
listxGenDelScopeList "/TOP/system/transmitter/"
listxGenDelScopeList -all
```

listxGenDelSignalList

Description

Delete the signal list from the xloc list.

Syntax

```
listxGenDelSignalList [signalList] [-all]
```

Arguments

`-all`

Delete all signal lists.

`signalList`

Specify the signal list to delete.

Value Returned

None.

Example

```
listxGenDelSignalList "/TOP/system/transmitter/b"
listxGenDelSignalList -all
```

listxGenEndTime

Description

Set the end time for the generated xloc file. The time unit is the original FSDB file time unit.

Syntax

```
listxGenEndTime endTime
```

Arguments

`endTime`

Specify the xloc file end time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listGenEndTime 10000
```

listxGenLoadScopeFile

Description

Load the scope list from the xloc file.

Syntax

```
listxGenLoadScopeFile fileName
```

Arguments

`fileName`

Specify the xloc file to load the scope list from.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxGenLoadScopeFile scope_list.scope
```

listxGenLoadSignalFile

Description

Load the signal list from the xloc file.

Syntax

```
listxGenLoadSignalFile fileName
```

Arguments

`fileName`

Specify the xloc file to load the signal list from.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxGenLoadSignalFile  signal_list.sig
```

listxGenStartTime

Description

Set the start time for the generated xloc file. The time unit is the original FSDB file time unit.

Syntax

```
listxGenStartTime startTime
```

Arguments

`startTime`

Specify the xloc file start time.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listGenStartTime 0
```

listxGenXloc

Description

Generates a new X file based on the settings in the *Create X File* form.

Syntax

```
listxGenXloc filename
```

Arguments

`filename`

Specified the X file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxGenXloc x_file.xloc
```

listxOpenFile

Description

Open an xloc file as the source file.

Syntax

`listxOpenFile` *`file_name`*

Arguments

`fileName`

Specify the X file to open.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxOpenFile rtl.fsdb.xloc
```

View

listxAddSignalToListX

Description

Add the specified signals to the *List X* form.

Syntax

```
listxAddSignalToListX signalList
```

Arguments

`signalList`

Specify the target signal list to add to the *List X* form.

Value Returned

None.

Example

```
listxAddSignalToListX "/TOP/system/transmitter/" \
"/TOP/system/transmitter/b"
```

listxDelAll

Description

Delete all the target signals and the X value in the *List X* form.

Syntax

```
listxDelAll
```

Value Returned

None.

Example

```
listxDelAll
```

listxDumpAU

Description

Dump the current X results to the log file.

Syntax

```
listxDumpAU
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxDumpAU
```

listxFastJump

Description

Jump to the first X viewing time.

Syntax

```
listxFastJump
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxFastJump
```

listxPlay

Description

Start to list the X value with the specified time range.

Syntax

```
listxPlay -startTime startTime -stepWidth width
```

Arguments

`-startTime` *`startTime`*

Specify the start time.

`-stepWidth` *`width`*

Specify the viewing window width.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxPlay -startTime 0 -stepWidth 100
```

listxSaveListX

Description

Save the list X results to a text file.

Syntax

`listxSaveListX` *`fileName`* `[-delim` *`delim`*`] [-simpleFormat]`

Arguments

`-delim delim`

Specify the delimiter of the full signal name. The default is /.

`fileName`

Specify the file to save the list X results to.

`-simpleFormat`

Save the list X results in the format with the full signal name and the value change time. For example, `system.i_cpu.i_ALUB.ACC[7:0]#6900`.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxSaveListX  output.lxt
```

```
listxSaveListX -delim . -simpleFormat listX.lxt
```

listxSearchNext

Description

Search forward from the current time to locate the occurring time of the next X value.

Syntax

```
listxSearchNext
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxSearchNext
```

listxSearchPrev

Description

Search backward from the current time to locate the occurring time of the previous X value.

Syntax

```
listxSearchPrev
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
listxSearchPrev
```

Configuration

listxSetOptions

Description

Set the list X options.

Syntax

```
listxSetOptions [-ignorePowerOffX on|off] [-displayPowerInfo
on|off] [-listBits on|off] [-skipGlitch on|off] [-syncWave on|off]
[-hierName on|off] [-stableLen length] [-listOnlyFFTriState
on|off]
```

Arguments

`-displayPowerInfo on|off`

When this option is turned *on*, display the power domain related information in the *List X* form. The default is *off*.

`-hierName on|off`

When this option is turned *on*, display the full hierarchy names in the **Name** column. The default is *on*.

`-ignorePowerOffX on|off`

When this option is turned *on*, remove the X value when the power domain is off or all the driver power domains are off. This option can be enabled when the power design is loaded. The default is *off*.

`-listBits on|off`

When this option is turned *on*, display only the bits of the specified signal with X values. The default is *off*.

`-listOnlyFFTriState on|off`

When this option is turned *on*, display only the flip-flop and Tri-state signals. The default is *on*.

`-skipGlitch on|off`

When this option is turned *on*, skip X values caused by glitches. The default is *on*.

`-stableLen` *`length`*

List only the X values that have stable width equal to or larger than the specified time.

`-syncWave on|off`

When this option is turned *on*, synchronize the signals in the selected row with the signals in the primary *nWave* window while synchronizing the X time of the selected row with the cursor time in the primary *nWave* window. The default is *off*.

Value Returned

None.

Example

```
listxSetOptions -ignorePowerOffX on
listxSetOptions -listBits on -syncWave on -stableLen 100
```

NPI

File

npiImport

Description

Imports the design.

Syntax

```
npiImport [-aliasIgnoreHier] [-aliasFile alias_file]
[-lib libName] [-recordhlc cmdFile] [-noInc] [-recordhlc cmdFile]
[-top topModule] [-vtop fileName][-vhdl [-93]|-verilog]
```

Arguments

`-aliasFile` *`alias_file`*

Specify an alias file.

`-aliasIgnoreHier`

Ignore the hierarchy differences when applying alias settings to the signals.

NOTE: This option should be used with the *-aliasFile* option.

`-lib` *`libName`*

Specify the library name.

`-noInc`

Suppress the incremental loading.

`-recordhlc` *`cmdFile`*

Record the command language history to the specified *cmdFile*.

`-top` *`topModule`*

Specify the top module name.

`-vhdl [-93]|-verilog`

Specify the language of the imported design. For VHDL, if *-93* is not specified, *VHDL-87* should be used as the default version of VHDL syntax.

`-vtop` *`fileName`*

Specify virtual top file for import design.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiImport -vhdl -top "system top1" -f run.f
npiImport -vhdl -top "system" -f run.f
npiImport -vhdl -93 -f run.f
npiImport -f run.f
npiImport -f run.f -aliasIgnoreHier -aliasFile "signal.alias"
```

Src

npiSrcFind

Description

Identifies the specified signal or instance through the design and saves to a file.

Syntax

```
npiSrcFind [-delim delim] [-signal|-inst] [-case on|off]
[-fullHierarchy on|off] [-libCell on|off] name -output outFileName
```

Arguments

`-case on|off`

Turn on or off the case matching. The default is *on*.

`-delim delim`

Specify the delimiter of the name list.

`-output outFileName`

Specify the output file name to save the search results to.

`-fullHierarchy on|off`

Turn on or off the search in full hierarchy. The default is *off*.

`-inst`

Search for the specified instance.

`-libCell on|off`

Turn on or off the search from the library cell. The default is *off*.

name

Specify the pattern or name.

`-signal`

Search for the specified signal.

NOTE: When the *-inst* and *-signal* options are specified simultaneously, the *-signal* option will be ignored.

Value Returned

char*

Example

```
set active_scope [npiSrcGetScope]

Output data
[active_scope]
System.i_cpu..i_ALUB
```

npiSrcGetScope

Description

Gets the current active scope.

Syntax

```
npiSrcGetScope
```

Value Returned

char*

Example

```
set active_scope [npiSrcGetScope]

Output data
[active_scope]
System.i_cpu..i_ALUB
```

npiSrcReportAllFileNames

Description

Saves all file names of the design to the output file.

Syntax

```
npiSrcReportAllFileNames [-file filename|-all] [-sort]
-output outFileName
```

Arguments

`-all`

Output all corresponding module lists of each file.

`-file` *`filename`*

Output the module lists of the specified file.

NOTE: When the *-all* and *-file* options are specified simultaneously, the *-file* option will be ignored.

`-output` *`outFilename`*

Specify the output file name.

`-sort`

Sort the output alphabetically.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiSrcReportAllFileNames -file "cpu.v" "design_cpu.list"

[design_cpu.list]
cpu.v
CPU
```

npiSrcReportAllModuleNames

Description

Saves all module names of the design to the output file.

Syntax

```
npiSrcReportAllFileNames [-Module moduleName | -all] [-sort]
-output outFileName
```

Arguments

`-all`

Output all corresponding module lists of each file.

`-Module` *`moduleName`*

Output the instantiated lists of the specified module name.

`-output outFileName`
Specify the output file name.

`-sort`
Sort the output alphabetically.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiSrcReportAllModuleNames -module "CPU" -sort "design_cpu.list"

[design_cpu.list]
CPU
i_cpu
```

npiSrcSetScope

Description

Sets the active scope.

Syntax

```
npiSrcSetScope [-delim delim] scopename
```

Arguments

`-delim delim`
Specify the delimiter of the hierarchical name.

`scopename`
Specify the active scope.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiSrcSetScope -delim {/} {system/I_cpu/I_CCU}
```

npiSrcTraceConnectivity

Description

Saves the connectivity of the specified signal to the output file.

Syntax

```
npiSrcTraceConnectivity [-delim delim] [-r][-range start end]
signalName -output outFileName
```

Arguments

`-delim delim`

Specify the delimiter of the name list.

`-r`

Relate the signal name to the current scope.

`-output outFileName`

Specify the output file name.

`-range start end`

Specify the index range for tracing the partial bit range for the specified bus signal.

`signalName`

Specify the signal name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiSrcTraceConnectivity “system.i_pram.VMA” –output drvLoad.list

[drvLoad.list]
system.i_pram.VMA
         <D>                 system.v(42):     CPU
i_cpu(clock,reset_cpu,VMA,R_W,data,addr);
         <L> system.v(44): pram i_pram(clock,VMA,R_W,addr,data);
system :              1 driver pass-through(s),      1 load
pass-through(s)
        *<D> TopModule.v(54): assign VMA = C0;
    *system.i_cpu :     1 driver(s)
        *<L> pram.v(38): always @(posedge (clock & VMA))
    *system.i_pram :     1 load(s)
```

```
    *Total :         1 driver(s),    1 load(s),    1 driver
pass-through(s),    1 load pass-through(s)
```

npiSrcTraceDriver

Description

Saves the drivers of the specified signal to the output file.

Syntax

```
npiSrcTraceConnectivity [-delim delim] [-r] [-range start end]
signalName -output outFileName
```

Arguments

`-delim delim`

Specify the delimiter of the hierarchical name.

`-output outFileName`

Specify the output file name.

`-r`

Relate the signal name to the current scope.

`-range start end`

Specify the index range for tracing the partial bit range for the specified bus signal.

`signalName`

Specify the signal name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiSrcTraceDriver "top.ccx_inst.pxm0_pcx_req" –range 0 2 –output
drv.list

[drv.list]
top.ccx_inst.pxm0_pcx_req[2:0]

        *<D>        test.v(73):        assign
pxm0_pcx_req=
{cmp_l2clk,cmp_l2clk,cmp_l2clk};
```

```
        <D>  test.v(80):  ccx  ccx_inst(cmp_l2clk,
pxm0_pcx_req,
pxm0_ccx_credit, ccx_pxm0_ctl, ccx_pxm0_data,
ccx_pxm0_req_credit,
ccx_pxm0_resp_credit);
    *top :      1 driver(s),   1 driver pass-through(s)
        <9D> test.v(47): module ccx (input cmp_l2clk, inout
[2:0]
pxm0_pcx_req,  inout  pxm0_ccx_credit,  inout [2:0]
ccx_pxm0_ctl,
inout [2:0] ccx_pxm0_data,  inout [2:0] ccx_pxm0_req_credit,
inout
[2:0] ccx_pxm0_resp_credit );
     top.ccx_inst :     9 driver pass-through(s)
    *Total :       1 driver(s),  10 driver pass-through(s)
```

npiSrcTraceFanIn

Description

Saves the fan-in register of the specified signal to the output file.

Syntax

`npiSrcTraceFanIn -src -sch signalName -output` *`outFileName`*

Arguments

`-output` *`outFileName`*

Specify the output file name.

`signalName`

Specify the signal name.

`-sch`

Find the fan-in registers basing on the net list.

`-src`

Find the fan-in registers basing on the HDB. This option is enabled as default.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiSrcTraceFanIn "system.data \[7:0\] " –output fanIn.list
```

```
[fanIn.list]
system.data[7:0]
        *<FI> pram.v(38~43): always @(posedge (clock & VMA))
    *system.i_pram : 1 result(s)
       *<FI> PCU.v(62~65): always @(negedge C5 or negedge reset)
begin
    *system.i_cpu.i_PCU : 1 result(s)
    *Total : 2 result(s)
```

npiSrcTraceFanOut

Description

Saves the fan-out registers of the specified signal to the output file.

Syntax

```
npiSrcTraceFanOut -src -sch signalName -output outFileName
```

Arguments

`-output` *`outFileName`*

Specify the output file name.

`signalName`

Specify the signal name.

`-sch`

Find the fan-in registers basing on the net list.

`-src`

Find the fan-in registers basing on the HDB. This option is enabled as default.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiSrcTraceFanOut "system.data \[7:0\] " –output fanOut.list
 [fanOut.list]
 system.data[7:0]
         *<FO> pram.v(38~43): always @(posedge (clock & VMA))
     *system.i_pram : 1 result(s)
        *<FO> PCU.v(91~94): always @(data or C1 or error_reset)
     *system.i_cpu.i_PCU : 1 result(s)
     *Total : 2 result(s)
```

npiSrcTraceLoad

Description

Saves the load of the specified signal.

Syntax

```
npiSrcTraceLoad [-delim delim] [-r][-range start end] signalName
-output outFileName
```

Arguments

`-delim delim`

Specify the delimiter of the name list.

`-output outFileName`

Specify the scope list to delete.

`-r`

The specified name is related to the current active scope.

`-range start end`

Specify the index range for tracing the partial bit range for the specified bus signal.

`signalName`

Specify the signal name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiSrcTraceLoad "system.data" –range 0 7 –output load.list
[load.list]
system.data[7:0]
<L> system.v(42): CPU i_cpu(clock,reset_cpu,VMA,R_W,data,addr);
<L> system.v(44): pram i_pram(clock,VMA,R_W,addr,data);
system : 2 load pass-through(s)
<L> TopModule.v(62): , .data(data)
system.i_cpu : 1 load pass-through(s)
*<L> pram.v(42): else macroram[addr]=data;
*system.i_pram : 1 load(s)
*<L> PCU.v(91): always @(data or C1 or error_reset)
*<L> PCU.v(94): else if (C1) TDB = data;
*system.i_cpu.i_PCU : 2 load(s)
*Total : 3 load(s), 3 load pass-through(s)
```

npiSchGet

Description

Gets the property of the instance.

Syntax

```
npiSchGet [-fileLine|-func] –inst instanceName
```

Arguments

`-fileLine`

Specify to output the file line number.

`-func`

Specify to output the function. The list of function names are: **HierModule**, **UnknownCell**, **Not**, **Or**, **And**, **Nor**, **Nand**, **Buffer**, **Xor**, **XNor**, **Combo**, **Macro**, **TriBuffer**, **InBuffer**, **OutBuffer**, **Tristate**, **FlipFlop**, **Latch**, **Delay**, **Mux**, **Assignment**, **FSM**, **Init**, and **InfLatch**.

`–inst instanceName`

Specify the instance name.

Value Returned

char*

Example

```
npiSchGet -func "system.i_cpu.i_ALUB.i_alu.U127

Output format:
if "-func" is specified, output
  FunctionName
if "-fileLine" is specified, output
fileName:LineNo

Output Example:
if "-func" is specified,
   AND
if "-fileLine" is specified,
   system.sv:25
```

npiSchScan

Description

Scans the value of the trace results from NPI schematic tracing commands with the command format: npiSchTracexxx.

Syntax

```
npiScan -driver|-load|-fanin|-fanout|-connectivity|-trace2signal
```

Arguments

`-connectivity`

Specify to scan the value of the trace results from the *npiSchTraceConnectivity* command.

`-driver`

Specify to scan the value of the trace results from the *npiSchTraceDriver* command.

`-fanin`

Specify to scan the value of the trace results from the *npiSchTraceFanIn* command.

`-fanout`

Specify to scan the value of the trace results from the *npiSchTraceFanOut* command.

`-load`

Specify to scan the value of the trace results from the *npiSchTraceLoad* command.

`-trace2signal`

Specify to scan the value of the trace results from the *npiSchTrace2Signals* command.

Value Returned

The object handle pointed by the specified iterator and then iterate the iterator to point to the next object.

Example

```
npiSchTraceDriver -signal {system.i_cpu.i_ALUB.i_alu.add_23.B}
set res [ npiSchScan -driver ]
```

```
while { $res != "" } {
  set drv_net $res
  puts "$res"
  set res [ npiSchScan -driver ]
}
```

The printed results:

```
system.i_cpu.i_ALUB.i_alu.add_23.U13
system.i_cpu.i_ALUB.i_alu.add_23.U14
```

npiSchTrace2Signals

Description

Traces the connectivity between two design signals.

Syntax

```
npiSchTrace2Signals [-delim delim] [-passCom signal1Name
signal2Name...] [-byPassCom signal1Name signal2Name...] [-inst]
-from fromSignalName -to toSignalName
```

Arguments

`-byPassCom` *`signal1Name signal2Name...`*

Stop the trace action from passing the specified signals.

`-delim` *`delim`*

Specify the delimiter of the full hierarchical signal name.

`-from` *`fromSignalName`*

Specify the starting point of full hierarchical signal name.

`-inst`

Return all hierarchical instance names between two signals.

`-passCom` *`signal1Name signal2Name...`*

Compel the trace action to pass the specified signals.

`-to` *`toSignalName`*

Specify the ending point of full hierarchical signal name.

Value Returned

1 if successful; otherwise, returns 0.

NOTE: The *npiSchScan* command can be used to traverse the trace results.

Example

```
npiSchTrace2Signals -delim "." -stopOnFF on -from
"system.i_cpu.i_ALUB.i_alu.n573" -to "system.i_cpu.i_ALUB.ALU"
```

Use npiSchScan to print trace results:

```
system.i_cpu.i_ALUB.i_alu.add_24.A[3]
system.i_cpu.i_ALUB.i_alu.add_24.n48
system.i_cpu.i_ALUB.i_alu.add_24.SUM[3]
```

npiSchTraceDriver

Description

Finds the drivers of the specified design signal.

Syntax

```
npiSchTraceDriver [-delim delim] [-range start end] [-signal
sigName] [-inst instName]
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the name list.

`-inst` *`instName`*

Specify the name of the instance to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If an instance is selected, or the *-inst* option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal to be traced.

Value Returned

1 if successful; otherwise, returns 0.

NOTE: The *npiSchScan* command can be used to traverse the trace results.

Example

```
npiSchTraceDriver -signal {system.i_cpu.i_ALUB.i_alu.add_23.B}
```

Use npiSchScan to print trace results:

```
system.i_cpu.i_ALUB.i_alu.add_23.U13
system.i_cpu.i_ALUB.i_alu.add_23.U14
```

npiSchTraceFanIn

Description

Identifies the fan-in registers of the specified design signal.

Syntax

```
npiSchTraceFanIn [-delim delim] [-range start end]
[-signal signalName] [-inst instName]
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the name list.

`-inst` *`instName`*

Specify the name of the instance to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If an instance is selected, or the *-inst* option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal to be traced.

Value Returned

1 if successful; otherwise, returns 0.

NOTE: The *npiSchScan* command can be used to traverse the trace results.

Example

```
npiSchTraceFanIn -signal {system.i_cpu.i_ALUB.i_alu.add_23.B}
```

Use npiSchScan to print trace results:

```
system.i_cpu.i_ALUB.i_alu.add_23.U13
system.i_cpu.i_ALUB.i_alu.add_23.U14
```

npiSchTraceFanOut

Description

Identifies the fan-out registers of the specified design signal.

Syntax

```
npiSchTraceFanOut [-delim delim] [-range start end]
[-signal signalName] [-inst instName]
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the name list.

`-inst` *`instName`*

Specify the name of the instance to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If an instance is selected, or the *-inst* option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal to be traced.

Value Returned

1 if successful; otherwise, returns 0.

NOTE: The *npiSchScan* command can be used to traverse the trace results.

Example

```
npiSchTraceFanOut -signal {system.i_cpu.i_ALUB.i_alu.add_23.B}
```

Use npiSchScan to print trace results:

```
system.i_cpu.i_ALUB.i_alu.add_23.U13
system.i_cpu.i_ALUB.i_alu.add_23.U14
```

npiSchTraceLoad

Description

Identifies the loads of the specified design signal.

Syntax

```
npiSchTraceFanOut [-delim delim] [-range start end]
[-signal signalName] [-inst instName]
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the name list.

`-inst` *`instName`*

Specify the name of the instance to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If an instance is selected, or the *-inst* option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify the signal to be traced.

Value Returned

1 if successful; otherwise, returns 0.

NOTE: The *npiSchScan* command can be used to traverse the trace results.

Example

```
npiSchTraceLoad -signal {system.i_cpu.i_ALUB.i_alu.add_23.B}
```

Use npiSchScan to print trace results:

```
system.i_cpu.i_ALUB.i_alu.add_23.U13
system.i_cpu.i_ALUB.i_alu.add_23.U14
```

npiTraceConnectivity

Description

Detects the connectivity of the specified design signal.

Syntax

```
npiSchTraceFanOut [-delim delim] [-range start end]
[-signal signalName] [-inst instName]
```

Arguments

`-delim` *`delim`*

Specify the delimiter of the name list.

`-inst` *`instName`*

Specify the name of the instance to be traced.

`-range` *`start end`*

Specify the index range for tracing the partial bit range for the specified bus signal. If an instance is selected, or the *-inst* option is specified, this range option is ignored.

`-signal` *`signalName`*

Specify signal to be traced.

Value Returned

1 if successful; otherwise, returns 0.

NOTE: The *npiSchScan* command can be used to traverse the trace results.

Example

```
npiTraceConnectivity -signal {system.i_cpu.i_ALUB.i_alu.add_23.B}
```

Use npiSchScan to print trace results:

```
system.i_cpu.i_ALUB.i_alu.add_23.U13
system.i_cpu.i_ALUB.i_alu.add_23.U14
```

FSM

npiFsmAnalyze

Description

Analyzes the finite state machines (FSM).

Syntax

```
npiFsmAnalyze -fsm fsmname
```

Arguments

`-fsm` *`fsmname`*

Specify the finite state machines.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiFsmAnalyze
"sys.LS100_TOP.INT_4B5B_RSP.resp_port:FSM0:83:124:FSM"
```

npiFsmGet

Description

Gets all finite state machines (FSM) in the specified scope.

Syntax

```
npiFsmGet -scope scopename
```

Arguments

`-scope` *`scopename`*

Specify the scope.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiFsmGet -scope "sys.LS100_TOP.INT_4B5B_RSP"
```

Use npiFsmScan to print trace results:

```
"sys.LS100_TOP.INT_4B5B_RSP.resp_port:FSM0:83:124:FSM"
"sys.LS100_TOP.INT_4B5B_RSP.resp_port:FSM0:165:194:FSM"
```

npiFsmGetState

Description

Returns all state names of the finite state machines (FSM).

Syntax

```
npiFsmGetState [-fsm fsmname]
```

Arguments

`-fsm` *`fsmname`*

Specify the FSM to report. If this option is not specified, report the last analyzed FSM.

Value Returned

1 if successful; otherwise, returns 0.

NOTE: The *npiFsmScan* command can be used to traverse the trace results.

Example

```
npiFsmGetState -fsm fsm_1
```

Use npiSchScan to print trace results:

```
ST0 ST1 ST2 others
```

npiFsmGetTransition

Description

Returns all transitions of the finite state machines (FSM).

Syntax

```
npiFsmGetTransition [-fsm fsmname]
```

Arguments

`-fsm fsmname`

Specify the FSM to report. If this option is not specified, report the last analyzed FSM.

Value Returned

1 if successful; otherwise, returns 0.

NOTE: The *npiFsmScan* command can be used to traverse the trace results.

Example

```
npiFsmGetTransition -fsm fsm_1
```

Use npiFsmScan to print trace results:

```
T0 T1 T2 T3 T4 T5 T6
```

npiFsmSave

Description

Saves the *nState* analysis report for the finite state machines (FSM).

Syntax

```
npiFsmSave [-fsm fsmname] -output outFileName
```

Arguments

`-fsm fsmname`

Specify the FSM to report. If this option is not specified, report the last analyzed FSM.

`-output` *`outFileName`*

Specify output file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiFsmGet "sys.LS100_TOP.INT_4B5B_RSP.resp_port:FSM0:83:124:FSM"
-output Report.txt
```

npiFsmScan

Description

Scans the value of the trace results from the NPI schematic tracing commands with the command format: npiFsmGetxxx.

Syntax

```
npiFsmScan -fsm|-state|-transition
```

Arguments

`-fsm`

Specify to scan the value of the trace results from the *npiFsmGet* command.

`-state`

Specify to scan the value of the trace results from the *npiFsmGetState* command.

`-transition`

Specify to scan the value of the trace results from the *npiFsmGetTransition* command.

Value Returned

The object handle pointed by the specified iterator and then iterate the iterator to point to the next object.

Example

```
npiFsmScan "sys.LS100_TOP.INT_4B5B_RSP.resp_port:FSM0:83:124:FSM"
-output
```

npiFsmStateProperty

Description

Lists the properties of the state.

Syntax

```
npiFsmStateProperty [-fsm fsmname] -state stateName [-connect]
```

Arguments

`-connect`

Dump all the connections of the specified state. The connection format is:

```
-from
[from-state1]->[from-transition1]->[current-state]
-to
[current-state]->[to-transition1]->[to-state1]
```

`-fsm fsmname`

Specify the FSM to report. If this option is not specified, report the last analyzed FSM.

`-state stateName`

Specify the state.

Value Returned

char*.

Example

```
npiFsmStateProperty -fsm fsm_1 -state ST0
ST0
-value
4'b0001
-action

npiFsmStateProperty ST5
-value
4'b0000
-action
-from
[ST0]->[T5]->[ST5]
-to
[ST5]->[T8]->[ST8]
```

```
[ST5]->[T9]->[ST9]
```

npiFsmTransitionProperty

Description

Lists properties of the transition.

Syntax

```
npiFsmTransitionProperty [-fsm fsmname] -transition transitionName
[-connect]
```

Arguments

`-connect`

Dump all the connections of the specified transition. The connection format is:

```
-connection
[from-state]->[current-transition]->[to-state]
```

`-fsm` *`fsmname`*

Specify the FSM to report. If this option is not specified, report the last analyzed FSM.

`-transition` *`transitionName`*

Specify the transition.

Value Returned

char*.

Example

```
npiFsmTransitionProperty -fsm fsm_1 -transition T0
Output:
T0
-condition
Reset
-priority
0
-action
-connection
[ST0]->[T1]->[ST1]
```

Clock

npiClkAnalyzeCrossingPath

Description

Checks the crossing paths between specified clock domains.

Syntax

```
npiClkAnalyzeCrossingPath [-fromSrc [src1] [src2]...]
[-toSrc [src1] [src2]...] [-fromId [domain1][domain2]...]
[-toId [domain1][domain2]...] [-synchronizer [synchronizer name1]]
[-synchronizer [synchronizer name]]... [-chkSubDomain]
[-checkConvergence] [-checkDivergence]
[-checkReconvergence] [-passScope scopeList]
```

Arguments

`-checkConvergence`

Check the potential convergence type structure problems after crossing path extraction.

`-checkDivergence`

Check the potential divergence type structure problems after crossing path extraction.

`-checkReconvergence`

Check the potential reconvergence type structure problems after crossing path extraction.

`-chkSubDomain`

When this option is turned *on*, check whether the crossing paths between sub-domains belong to the selected domains. When this option is turned *off*, stop checking the sub-domains. The default *on*.

`-fromId [domain1][domain2]...`

Specify the ID numbers of the sub-domains that belong to the selected starting domain. If the value is "all", all domains will be specified.

`-fromSrc`

Specify the starting clock source.

`-passScope scopeList`

List the specified scope names. “<->” is used to separate scope1 and scope2. For example, "system.i_pram<->system.i_cpu.i_ALUB". "*" is used to represent all scopes. For example, "system.i_pram<->*".

`-synchronizer [synchronizer name]`

Specify the synchronizer template set on the **Synchronizer Setting** tab of the *Check Crossing Paths* form.

`-toId`

Specify the ID numbers of the sub-domains that belong to the selected ending domain. If the value is "all", all domains will be specified.

`-toSrc`

Specify the ending clock source.

Value Returned

char*.

Example

```
npiClkAnalyzeCrossingPath \
-fromSrc "system.clock" "system.i_cpu.i_CCU.C19"
"system.i_cpu.i_CCU.C5" "system.i_cpu.i_CCU.C6"
"system.i_cpu.i_ALUB.S1" \
-fromId "1.f" "1.r" "1.1.r" "1.2.f" "1.5.r" "2.f" "2.2" \
-toSrc "system.clock" "system.i_cpu.i_CCU.C19"
"system.i_cpu.i_CCU.C5" "system.i_cpu.i_CCU.C6"
"system.i_cpu.i_ALUB.S1" \
-toId "1.f" "1.r" "1.1.r" "1.2.f" "1.5.r" "2.f" "2.2" \
-chkSubDomain on
```

npiClkAnalyzeTree

Description

Extracts the clock tree with the specified signal name.

Syntax

```
npiClkAnalyzeTree [-signal signalName]|[-ctsAstro2005File|
-ctsAstro2007File|-ctsApolloFile|-ctsFeFile|-ctsIccFile|-sdcFile]
```

Arguments

`-ctsApolloFile`

Specify the Apollo CTS file to import.

`-ctsAstro2005File`
Specify the Astro CTS file in the 2005 version to import.

`-ctsAstro2007File`
Specify the Astro CTS file in the 2007 version to import.

`-ctsFeFile`
Specify the First Encounter CTS file to import.

`-ctsIccFile`
Specify the ICC CTS file to import.

`-sdcFile`
Specify the SDC file to import.

`-signal` *`signalName`*
Specify the signal name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiClkAnalyzeTree -signal top.clk
```

npiClkExtractDomain

Description

Extracts all clock domains for the design.

Syntax

```
npiClkExtractDomain [-sdc sdc_file]
```

Arguments

`-sdc` *`sdc_file`*
Specify the SDC file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiClkExtractDomain -sdc a.sdc
```

npiClkGetSrc

Description

Finds the clock source from the selected instance or the signal with specified constant setting.

Syntax

```
npiClkGetSrc [-signal signalName |-inst instName|
-loadFile filename [-checkValidation]][-constant signalName <const
value>...] [-constCellPort cellName <const value>...]
```

Arguments

`-constant signalName <const_value>`

Specify a constant signal with the assigned value to find the clock source. The format of the const_value can be a radix type Decimal, Hex, Octal or Binary with a number or "don’t care".

`-constCellPort cellName <const value>`

Specify a constant cell port with the assigned value to find the clock source. The format the const value can be a radix type Decimal, Hex, Octal or Binary with a number or "don’t care".

`-inst instName`

Find the clock source from the specified instance.

`-loadFile filename [-checkValidation]`

Load the settings for constant signal and constant cell port from *.cs file. If the *-checkValidation* option is turned *on*, the setting file will be checked for errors first.

`-signal signalName`

Find the clock source from the specified signal.

Value Returned

char*.

Example

```
npiClkGetSrc -signal "test.b" -constant "test.eco_n1" Binary 0
-constant "test.eco_n2" Hex 1 -constant "test.c" dontcare
```

Output format:

```
clkSrcName
```

Output Example:

```
top.Clk
```

npiClkReportCrossingPath

Description

Saves the crossing paths.

Syntax

```
npiClkReportCrossingPath -output outFileName [-fromSrc [src1]
[src2]...] [-toSrc [src1] [src2]...]
[-fromId [domain1][domain2]...] [-toId [domain1][domain2]...]
[-all] [-includeMid]
```

Arguments

`-all`

Report the crossing paths among all domains.

`-fromId`

Specify the ID numbers of the sub-domains that belong to the selected starting domain. If the value is "all", all domains will be specified.

`-fromSrc`

Specify the starting clock source.

`-includeMid`

Dump the instance information between register pairs.

`-output` *`outFileName`*

Specify the output file name.

`-toId`

Specify the ID numbers of the sub-domains that belong to the selected ending domain. If the value is "all", all domains will be specified.

`-toSrc`

Specify the ending clock source.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
dkpClkReportCrossingPath -output Report.txt -all
```

npiClkReportDomain

Description

Saves all clock domains.

Syntax

```
npiClkReportDomain -output outFileName
```

Arguments

`-output` *`outFileName`*

Specify the output file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiClkReportDomain -output Report.txt
```

npiClkReportTree

Description

Saves the clock tree.

Syntax

```
npiClkReportTree -output outFileName
```

Arguments

`-output` *`outFileName`*

Specify the output file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiClkReportTree -output Report.txt
```

npiClockLoadDB

Description

Loads the clock database.

Syntax

`npiClockLoadDB -file` *`dirName`*

Arguments

`-file` *`dirName`*

Specify the directory to load the clock database.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiClockLoadDB -file /rcc2/integ/tmp
```

npiClockSaveDB

Description

Saves the clock database.

Syntax

`npiClockSaveDB -file` *`dirName`*

Arguments

`-file` *`dirName`*

Specify the directory to save the clock database to.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiClockSaveDB -file /rcc2/integ/tmp
```

VC Apps

VC Apps Preliminary Set Up Tcl Commands

viaSetupL1Apps

Description

Loads VC Apps libraries.

NOTE: This command must be called before applying any Design Manipulation Tcl commands.

Syntax

```
viaSetupL1Apps
```

Value Returned

1 if successful; otherwise, returns 0.

Design Manipulation - Apps Database

npiDlSymStr -func via_dm_app_get_db

Description

Loads the sharing design manipulation Apps database. This database is used by Apps including *ChipInt*, *HierMan*, *Disconnect Port*, and *Delete Pass Through*. You can use this database to perform further operations using DM L0 APIs.

Syntax

```
npiDlSymStr -func via_dm_app_get_db
```

Value Returned

On success, the created DMDB handle is returned.

On failure, an empty string is returned.

Example

```
viaSetupL1Apps

## Open Apps and do operations
npiDlSym -func via_int_init -shared
npiDlSym -func via_int_create_module -import_file import_file
npiDlSym -func via_int_exit

## Get Apps sharing DB
set DMDB [npiDlSymStr -func via_dm_app_get_db]
npi_dm_set_master_db -dmdb $DMDB

## Modify Apps result by DM L0 APIs
set instHDL [npi_dm_handle_by_name -name top.u_m1 -scope ""]
npi_dm_rename_object -handle $instHDL -name u_new_m1

## Export designs
npi_dm_write_text_mode -dir RES

debExit
```

Design Manipulation - HierMan

npiDlSym -func via_hm_diff

Description

Calls TkDiff to view differences between two files.

Syntax

```
npiDlSym -func via_hm_diff filePath1 filePath2
```

Arguments

`filePath1`

Specify the path of the original file.

`filePath2`

Specify the path of the modified file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_hm_diff {./top.v} {./HM_DIR/work_dir_path/
top.v}
```

npiDlSym -func via_hm_exit

Description

Exits the HierMan environment and destroys all related objects.

Syntax

```
npiDlSym -func via_hm_exit
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_hm_exit
```

npiDlSym -func via_hm_export

Description

Exports the operation results and write modified files to the work directory.

Syntax

```
npiDlSym -func via_hm_exit[-ansi]
```

Arguments

`-ansi`

Specify the style of the write out port.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_hm_export
```

npiDlSym -func via_hm_group

Description

Groups the target instances under the target scope into the specified group instance, or wraps the target scope into the specified group module. The corresponding group module is created with the given name. The group module will be written out in the specified group module file after being exported.

Syntax

```
npiDlSym -func via_hm_group -target_scope scopeName
<-target_inst instanceList | -wrap_target_scope> -group_mod
moduleName
-group_inst instanceName -group_mod_file filename
```

Arguments

`-group_inst` *`instanceName`*

Specify the group instance.

`-group_mod` *`moduleName`*

Specify the group module.

`-group_mod_file` *`filename`*

Specify the file of the group module.

`-target_inst` *`instanceList`*

Specify the instance lists under the target scope.

`-target_scope` *`scopeName`*

Specify the target scope.

`-wrap_target_scope`

Wraps the target scope into the specified group module.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_hm_group -target_scope {top.U1} -target_inst
{U6} {U7} -group_mod
{invMod} -group_inst {invInst} -group_mod_file {invFile.v}
npiDlSym -func via_hm_group -target_scope {top} –wrap_targt_scope
-group_mod
{wrapMod} -group_inst {wrapInst} -group_mod_file {wrap.v}
```

npiDlSym -func via_hm_init

Description

Initializes the HierMan environment.

Syntax

```
npiDlSym -func via_hm_init [-dir workDir] [-shared]
```

Arguments

`-dir` *`workDir`*

Specify the work directory for HierMan. The default work directory is *"./HM_DIR"*.

`-shared`

Use shared DM Database.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_hm_init -dir HM_TEST -shared
```

npiDlSym -func via_hm_move

Description

Moves the source instance to the destination scope.

Syntax

```
npiDlSym -func via_hm_move -from sourceInstance
-to destinationScope
```

Arguments

`-from sourceInstance`

Specify the source instance to be moved.

`-to destinationScope`

Specify the destination scope.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_hm_move -from {top.U1.U5} -to {top}
```

npiDlSym -func via_hm_rename

Description

Renames the source instance.

Syntax

```
npiDlSym -func via_hm_rename -inst instanceName -name newName
```

Arguments

`-inst instanceName`

Specify the source instance to be renamed.

`-name newName`

Specify the new name for the source instance.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_hm_rename -inst {top.U1.U5} -name {X5}
```

npiDlSym -func via_hm_ungroup

Description

Removes a level of hierarchy for the specified instance. The contents of the instance are brought up to the level as the instance. The specified delimiter is used for renaming during ungroup operation.

Syntax

```
npiDlSym -func via_hm_ungroup -inst instanceName
[-delimiter delimiterLiteral]
```

Arguments

`-delimiter delimiterLiteral`

Specify the delimiter of hierarchical reference. The delimiter is used when renaming is performed. Default delimiter is {.}.

`-inst instanceName`

Specify name of the instance to be ungrouped.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func  via_hm_ungroup -inst {top.inst1.u5} -delimiter
{_}
```

npiDlSym -func via_hm_uniquify

Description

Generates a unique module for the specified instance even if the referenced module of the instance is already unique.

Syntax

```
npiDlSym -func via_hm_uniquify -inst instanceName [-all]
```

Arguments

`-all`

Turn on this option to make the instances unique that are not unique on the hierarchy tree of the specified instance.

`-inst` *`instanceName`*

Specify name of the instance that needs to be made unique.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func  via_hm_uniquify -inst {top.inst1.u5}
```

Design Manipulation - Ungroup

npiDlSym -func via_ung_diff

Description

Calls TkDiff to view differences between two files.

Syntax

```
npiDlSym -func via_ung_diff filePath1 filePath2
```

Arguments

`filePath1`

Specify the path of the original file.

`filePath2`

Specify the path of the modified file.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_ung_diff {./top.v} {./UNG_TEST/work_dir_path/
top.v}
```

npiDlSym -func via_ung_exit

Description

Exits the ungroup environment and destroys all related objects.

Syntax

```
npiDlSym -func via_ung_exit
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_ung_exit
```

npiDlSym -func via_ung_init

Description

Initializes the ungroup environment.

Syntax

```
npiDlSym -func via_ung_init [-dir workDir]
```

Arguments

`-dir workDir`

Specify the work directory for ungroup. The default work directory is *"./UNG_DIR"*.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_ung_init -dir UNG_TEST
```

npiDlSym -func via_ung_ungroup

Description

Ungroups the target instance and write results into the specified directory. The specified delimiter is used to rename during the ungroup process.

Syntax

```
npiDlSym -func via_ung_ungroup -target_inst instanceName
-dir workDir [-delimiter delimiterLiteral]
```

Arguments

`-delimiter delimiterLiteral`

Specify the delimiter for the hierarchical reference. The delimiter is used when renaming. The default delimiter is *{.}*.

`-target_inst` *`instanceName`*

Specify the target instance to be ungrouped.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_ung_ungroup -target_inst {top.U1} -dir
UNG_TEST -delimiter {.}
```

Design Manipulation - Port Tie Constant

npiDlSym -func vaptc_exec_cmd -cmd EXECUTE

Description

Executes port tie constant operation based on the table.

Syntax

```
npiDlSym -func vaptc_exec_cmd -cmd EXECUTE
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func vaptc_exec_cmd -cmd EXECUTE
```

npiDlSym -func vaptc_exec_cmd -cmd DELETE_TABLE_ITEM

Description

Deletes the pending items of the table.

Syntax

```
npiDlSym -func vaptc_exec_cmd -cmd DELETE_TABLE_ITEM -row row_num
```

Arguments

-row rowNumber

Specifies the row number.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func vaptc_exec_cmd -cmd DELETE_TABLE_ITEM -row {2}
```

npiDlSym -func vaptc_exec_cmd -cmd ADD_TABLE_ITEM

Description

Adds the pending items of the table.

Syntax

```
npiDlSym -func vaptc_exec_cmd -cmd ADD_TABLE_ITEM -row row_num
```

Arguments

-row rowNumber

Specifies the row number.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func vaptc_exec_cmd -cmd ADD_TABLE_ITEM –row {2}
```

npiDlSym -func vaptc_exec_cmd -cmd EDIT_TABLE

Description

Edits the pending items of the table.

Syntax

```
npiDlSym -func vaptc_exec_cmd -cmd EDIT_TABLE -row row_num -col
column_num –data data_string
```

Arguments

-row *rowNumber*

Specifies the row number.

`-col columnNumber`

Specifies the column number.

`-data`

Specifies the input data string.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func vaptc_exec_cmd -cmd EDIT_TABLE –row {2} –col {2} –
data {1’b0}
```

npiDlSym -func vaptc_exec_cmd -cmd OPEN_WIN

Description

Opens the port tie constant window.

Syntax

```
npiDlSym -func vaptc_exec_cmd -cmd OPEN_WIN
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func vaptc_exec_cmd -cmd OPEN_WIN
```

npiDlSym -func vaptc_exec_cmd -cmd CLOSE_WIN

Description

Closes the port tie constant window.

Syntax

```
npiDlSym -func vaptc_exec_cmd -cmd CLOSE_WIN
```

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func vaptc_exec_cmd -cmd CLOSE_WIN
```

npiDlSym -func vaptc_exec_cmd -cmd SAVE_SETTING

Description

Saves the port tie constant table setting as the csv file.

Syntax

```
npiDlSym -func vaptc_exec_cmd -cmd SAVE_SETTING -file file_path
```

Arguments

`-file file_path`

Specifies the file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func vaptc_exec_cmd -cmd SAVE_SETTING –file {./config/
setting.csv}
```

npiDlSym -func vaptc_exec_cmd -cmd LOAD_SETTING

Description

Loads the port tie constant setting file.

Syntax

```
npiDlSym -func vaptc_exec_cmd -cmd LOAD_SETTING –file {./config/
setting.csv}
```

Arguments

`-file file_path`

Specifies the file name.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func vaptc_exec_cmd -cmd LOAD_SETTING –file {./config/
setting.csv}
```

Design Manipulation – Rename Instance

npiDlSym -func via_dm_rename_inst

Description

Executes renaming instance operation.

Syntax

```
npiDlSym -func via_dm_rename_inst -inst_file instance_list_file
-prefix prefix_str -postfix postfix_str -out_dir output_dir
```

Arguments

`-inst_file instance_list_file`

Specifies the file which includes the instance list to be renamed. Within file, each line represents the full hierarchy of one instance.

`-prefix`

Specifies the prefix string for renaming rule.

`-postfix`

Specifies the postfix string for renaming rule.

`-out_dir`

Specifies the output directory.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_dm_rename_inst -inst_file {./rename_list.file}
-prefix {prefix_} -postfix {_postfix} -out_dir {out_dir}
```

Design Manipulation – Delete Pass Through

npiDlSym -func via_dm_delete_pass_through_path

Description

Deletes pass through paths.

Syntax

```
npiDlSym -func via_dm_delete_pass_through_path -port_file
port_list_file -out_dir output_dir [-shared]
```

Arguments

`-port_file`

Specifies the file which includes the port list within the pass through path. Within file, each line represents the full hierarchy of one port.

`-out_dir`

Specifies the output directory.

`-shared`

Specifies to use shared DM database.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_dm_delete_pass_through_path -port_file
{port_list.file} -out_dir {output_dir}
```

Design Manipulation – Disconnect Port

npiDlSym -func via_dm_discon

Description

Disconnects the ports.

Syntax

```
npiDlSym -func via_dm_discon -port_file port_list_file -out_dir
output_dir [-shared]
```

Arguments

`-port_file`

Specifies the file which includes the port list to be disconnected. Within file, each line represents the full hierarchy of one port.

`-out_dir`

Specifies the output directory.

`-shared`

Specifies to use shared DM database.

Value Returned

1 if successful; otherwise, returns 0.

Example

```
npiDlSym -func via_dm_discon -port_file {port_list.file} -out_dir
{output_dir}
```

Design Manipulation - ChipInt

ChipInt Basic Tcl Commands

npiDlSym -func via_int_init

Description
Initializes ChipInt. This command must be used before calling other ChipInt TCL commands.

Syntax
```
npiDlSym -func via_int_init [-shared]
```

Arguments
-shared
Specifies to use shared DM database.

Value Returned
1 if successful; otherwise, returns 0.

npiDlSym -func via_int_exit

Description
Exits ChipInt.

Syntax
```
npiDlSym -func via_int_exit
```

Value Returned
1 if successful; otherwise, returns 0.

npiDlSym -func via_int_export

Description
Exports output files.

Syntax

```
npiDlSym -func via_int_export
```

Value Returned

1 if successful; otherwise, returns 0.

Module Creation Tcl Commands

npiDlSym -func via_int_create_module

Description

Creates modules by file. The file describes the input table as csv format. The first line represents the column header. Apart from the first line, each line represents a cell data corresponding to the column header. The following is a sample file:

```
======== Import File Example =========
#Module Name,File Name,Library Cell
new_ip1,new_ip1.v,0
new_ip2,new_ip2.v,1
=====================================
```

Syntax

```
npiDlSym -func via_int_create_module -import_file import_file
```

Arguments

-import_file

Specifies the import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_create_module -import_file import_file
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```

Parameter Creation Tcl Commands

npiDlSym -func via_int_create_parameter

Description

Creates parameters by file. The file describes the input table as csv format. The first line represents the column header. Apart from the first line, each line represents a cell data corresponding to the column header. The following is a sample file:

```
======== Import File Example===================================
#Scope,Name,Local,Data Type,Signing,Packed Dim.,Default Assign
top,P1,1,logic,unsigned,[1:0][2:0],1'b1
top,P2,0,int,signed,,3
================================================================
```

Syntax

```
npiDlSym -func via_int_create_parameter -import_file import_file
```

Arguments

-import_file

Specifies the import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_create_parameter -import_file import_file
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```

npiDlSym -func via_int_create_typeparam

Description

Creates type parameters by file. The file describes the input table as csv format. The first line represents the column header. Apart from the first line, each line represents a cell data corresponding to the column header. The following is a sample file:

```
======== Import File Example ===================================
#Scope,Name,Local,Data Type,Signing,Packed Dim.
top,PT1,1,logic,unsigned,[1:0][2:0]
top,PT2,0,integer,unsigned,
================================================================
```

Syntax

```
npiDlSym -func via_int_create_typeparam -import_file import_file
```

Arguments

-import_file

Specifies the import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_create_typeparam -import_file import_file
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```

Port Creation Tcl Commands

npiDlSym -func via_int_create_port

Description

Creates ports by file. The file describes the input table as csv format. The first line represents the column header. Apart from the first line, each line represents a cell data corresponding to the column header. The following is a sample file:

```
======== Import File Example ===================================
#Scope,Name,Direction,Data Type,Signing,Packed Dim.,Unpacked
Dim.,Net Port,Net Type
top,in1,input,logic,signed,[1:0],[1:0],1,wire
top,out1,output,shortint,signed,,,0,
================================================================
```

Syntax

```
npiDlSym -func via_int_create_port -import_file import_file
```

Arguments

-import_file

Specifies the import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_create_port -import_file import_file
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```

Instance Insertion Tcl Commands

npiDlSym -func via_int_insert_instance

Description

Inserts instances by file. The file describes the input table as csv format. The first line represents the column header. Apart from the first line, each line represents a cell data corresponding to the column header. The following is a sample file:

```
======== Import File Example ==================================
##Scope,Instance,Port,Inserted Module,Inserted Instance,Entry
A,Entry B
chip,ip1,out,mod_buf,buf1,in,out
chip,ip1,in,mod_buf,buf2,out,in
================================================================
```

Syntax

```
npiDlSym -func via_int_insert_instance -import_file import_file
```

Arguments

-import_file

Specifies the import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Tcl Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_insert_instance -import_file import_file
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```

Port-to-Hierarchy Tcl Commands

npiDlSym -func via_int_run_imp_exp

Description

Connects ports to hierarchies by file. Each line defines the source port hierarchy and the export/import port hierarchy. The first token is source port hierarchy and the second token is export/import port hierarchy. The tokens are separated by space. The following is a sample file:

```
======== Import File Example ===================================
## source_port_hierarchy export_import_port_hierarchy
top.u_chip.ip1.in top.top_in
top.u_chip.ip1.out top.top_out
================================================================
```

Syntax

```
npiDlSym -func via_int_run_imp_exp -file_name import_file
```

Arguments

-file_name

Specifies the import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Tcl Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_run_imp_exp -file_name import_file
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```

Module Instantiation Tcl Commands

npiDlSym -func via_int_run_inst

Description

Adds instances by file. Each line of import file describes the basic data units to add an instance. The data units are separated by space.

Syntax of Import File

```
scope_module_name source_module_name [parameter_assignment]
instance_name
```

Example of Import File

```
## scope_module_name source_module_name [parameter_assignment]
instance_name
top ip  inst1  ##without parameter assignment
top ip <PARAM>{ P1:1'b0, PT1:int} inst2 ##with parameter
assignment
top ip <PARAM>{ P1:1'b0, PT1:logic} inst3
```

Syntax of Tcl Command

```
npiDlSym -func via_int_run_inst -file_name import_file
```

Arguments

-file_name

Specify import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Tcl Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_run_inst -file_name import_file
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```

Port-to-Port Tcl Commands

npiDlSym -func via_int_run_connect

Description

Connects ports by file. The import file of port-to-port connection focuses on one instance. The file could be reused for different partitions just with hierarchy macro modification.

Syntax of Import File

The file includes four stages to define different categories. Each stage applies a different syntax to define the user knowledge.

1. <TARGET_INSTANCE>
   a. Stage purpose:
      Sets the target instance to the connected ports.
   b. Stage syntax:
      `target_instance_hierarchy`
2. <MACRO_DEFINE>
   a. Stage purpose:
      Sets hierarchy macro applied in the <MAP_TABLE> stage.
   b. Stage syntax:
      macro_name macro_context
3. <NAME_RULE>
   a. Stage purpose:
      Sets the naming rule and context.
   b. Stage syntax (bold words below represent key word and '|' represents logic "or"):
      prefix|**postfix**|**default**|**none** [name_text]
4. <MAP_TABLE>
   a. Stage purpose:
      Sets port-to-port mapping relationship.
   b. Stage syntax to define port-to-constant connection:
      source_port_name [source_port_index] constant_value
   c. Stage syntax to define port-to-port connection:

source_port_name [source_port_index] dest_port_hierarchy [dest_port_index]

Example of Import File

```
## Each stage defines different scenario described as follows:
## <TARGET_INSTANCE>: set target instance of port connection
## <MACRO_DEFINE>: set macro applied in <MAP_TABLE> stage
## <NAME_RULE>: set naming rule and context
## <MAP_TABLE>: set port-to-port mapping relationship

<TARGET_INSTANCE>
## target_instance_hierarchy
top.u_chip.ip1

<MACRO_DEFINE>
## macro_name macro_context
HIER_0 top.u_chip.ip2
HIER_1 top

<NAME_RULE>
## name_type [name_text]
prefix
<MAP_TABLE>
## source_port_name [source_port_index] dest_port_hierarchy
[dest_port_index]
## source_port_name [source_port_index] constant_value
in $HIER_0.out
out $HIER_1.out
```

Syntax of Tcl Command

```
npiDlSym -func via_int_run_connect -file_name import_file
```

Arguments

-file_name

Specifies the import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Tcl Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_run_connect -file_name import_file
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```

npiDlSym -func via_int_run_con_list

Description

Connects ports by file. The file includes the file path list. Each line includes one file path. Each file path defines the port-to-port connection configuration file of one instance. The extension file name must be ".list", for example, "import_file.list".

Syntax of Import File

```
full_config_file_path_defined_for_instance1
full_config_file_path_defined_for_instance2
.....................................
```

Example of Import File

```
./MapConfigFiles/top.u_chip.ip1_prefix_.cia
./MapConfigFiles/top.u_chip.ip2_prefix_.cia
```

Syntax of Tcl Command

```
npiDlSym -func via_int_run_con_list -file_name import_file.list
```

Arguments

-file_name

Specifies the import file path.

Value Returned

1 if successful; otherwise, returns 0.

Batch Mode Tcl Example

```
viaSetupL1Apps
npiDlSym -func via_int_init
npiDlSym -func via_int_run_con_list -file_name import_file.list
npiDlSym -func via_int_export
npiDlSym -func via_int_exit
debExit
```