ModelSim®

Advanced Verification and Debugging

SE

Graphical Interface Reference

Version 6.0b

Published: 15/Nov/04

Mentor Graphics®

This document is for information and instruction purposes. Mentor Graphics reserves the right to make changes in specifications and other information contained in this publication without prior notice, and the reader should, in all cases, consult Mentor Graphics to determine whether any changes have been made.

The terms and conditions governing the sale and licensing of Mentor Graphics products are set forth in written agreements between Mentor Graphics and its customers. No representation or other affirmation of fact contained in this publication shall be deemed to be a warranty or give rise to any liability of Mentor Graphics whatsoever.

MENTOR GRAPHICS MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

MENTOR GRAPHICS SHALL NOT BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATED TO THIS PUBLICATION OR THE INFORMATION CONTAINED IN IT, EVEN IF MENTOR GRAPHICS CORPORATION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

RESTRICTED RIGHTS LEGEND 03/97

U.S. Government Restricted Rights. The SOFTWARE and documentation have been developed entirely at private expense and are commercial computer software provided with restricted rights. Use, duplication or disclosure by the U.S. Government or a U.S. Government subcontractor is subject to the restrictions set forth in the license agreement provided with the software pursuant to DFARS 227.7202-3(a) or as set forth in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clause at FAR 52.227-19, as applicable.

Contractor/manufacturer is:

Mentor Graphics Corporation

8005 S.W. Boeckman Road, Wilsonville, Oregon 97070-7777.

This is an unpublished work of Mentor Graphics Corporation.

Contacting ModelSim Support

Telephone: 503.685.0820

Toll-Free Telephone: 877-744-6699

Website: www.model.com

Support: www.model.com/support

Technical support and updates

Support

Model Technology online and email technical support options, maintenance renewal, and links to international support contacts:
www.model.com/support/default.asp

Mentor Graphics support:
www.mentor.com/supportnet

Updates

Access to the most current version of ModelSim:
www.model.com/downloads/default.asp

Latest version email

Place your name on our list for email notification of news and updates:
www.model.com/products/informant.asp

Where to find our documentation

ModelSim documentation is available from our website at www.model.com/support or in the following formats and locations:

Document	Format	How to get it
ModelSim Installation & Licensing Guide	paper	shipped with Model*Sim*
	PDF	select **Help > Documentation**; also available from the Support page of our web site: www.model.com
ModelSim Quick Guide (command and feature quick-reference)	paper	shipped with Model*Sim*
	PDF	select **Help > Documentation**, also available from the Support page of our web site: www.model.com
ModelSim Tutorial	PDF, HTML	select **Help > Documentation**; also available from the Support page of our web site: www.model.com
ModelSim User's Manual	PDF, HTML	select **Help > Documentation**
ModelSim Command Reference	PDF, HTML	select **Help > Documentation**
ModelSim GUI Reference	PDF, HTML	select **Help > Documentation**
Foreign Language Interface Reference	PDF, HTML	select **Help > Documentation**
Std_DevelopersKit User's Manual	PDF	www.model.com/support/documentation/BOOK/sdk_um.pdf The Standard Developer's Kit is for use with Mentor Graphics QuickHDL.
Command Help	ASCII	type `help [command name]` at the prompt in the Transcript pane
Error message help	ASCII	type `verror <msgNum>` at the Transcript or shell prompt
Tcl Man Pages (Tcl manual)	HTML	select **Help > Tcl Man Pages**, or find *contents.htm* in *\modeltech\docs\tcl_help_html*
Technotes	HTML	select Technotes dropdown on www.model.com/support

Table of Contents

Technical support and updates GR-3
Where to find our documentation GR-4

1 - Simulator windows (GR-9)

Introduction GR-11
Design object icons and their meaning GR-12
Setting fonts GR-13
Main window GR-14
Workspace GR-15
Transcript GR-16
Multiple document interface (MDI) frame GR-17
Main window status bar GR-19
Main window menu bar GR-20
Main window toolbar GR-34
Main window dialogs GR-37
Waveform Compare dialogs GR-88
Active Processes pane GR-108
Process status GR-108
Active Processes dialogs GR-109
Assertions pane GR-110
Assertions pane columns GR-110
Hiding/showing columns in the Assertions pane GR-112
Assertions pane dialogs GR-113
Code coverage panes GR-116
Workspace pane GR-116
Missed Coverage pane GR-120
Current Exclusions pane GR-121
Instance Coverage pane GR-122
Details pane GR-123
Objects pane toggle coverage GR-125
Code coverage toolbar GR-127
Dataflow window GR-128
Objects you can view GR-128
Dataflow window menu bar GR-129
Dataflow window toolbar GR-132
Dataflow window dialogs GR-135
Functional Coverage pane GR-143
Functional Coverage window columns GR-143
Hiding/showing columns in the Functional Coverage window GR-144
Functional Coverage pane dialogs GR-145
List window GR-153
Objects you can view GR-153
List window menu bar GR-154
The List window context menu GR-155
List window dialogs GR-156

Locals pane GR-166
Locals dialogs GR-167
Memory windows GR-169
Memories you can view GR-170
Viewing memory contents GR-171
Direct address navigation GR-172
Splitting the memory contents window GR-172
Memory popup menu commands GR-173
Memory dialogs GR-175
Objects pane GR-184
Filtering the objects list GR-185
Objects dialogs GR-186
Profile panes GR-192
Profile pane columns GR-193
Hiding/showing columns in the Profile panes GR-194
Profiler popup menu commands GR-195
Profiler toolbar GR-196
Profiler dialogs GR-197
Source window GR-199
Opening source files GR-199
Displaying multiple source files GR-200
Language templates GR-201
Setting file-line breakpoints GR-203
Checking object values and descriptions GR-203
Finding and replacing in a Source window GR-204
Marking lines with bookmarks GR-204
Customizing the Source window GR-205
Source window menus GR-206
Watch pane GR-208
Objects you can view GR-208
Adding objects to the pane GR-208
Expanding objects to show individual bits GR-209
Grouping and ungrouping objects GR-209
Saving and reloading format files GR-209
Other Watch pane commands GR-209
Wave window GR-211
Wave window panes GR-213
Objects you can view GR-214
Wave window menu bar GR-216
Wave window toolbar GR-220
Waveform editor toolbar GR-222
Wave window dialogs GR-225
Customizing the GUI layout GR-258
Moving, docking, and undocking panes GR-258
Zooming panes GR-260
Columnar information display GR-262
Quick access toolbars GR-263
Creating and managing breakpoints GR-264

Signal breakpoints GR-264
File-line breakpoints GR-264

2 - Setting GUI preferences (GR-265)

ModelSim GUI preferences GR-266
Setting variables from the GUI GR-267
Setting variables from the command line GR-267

Index

1 - Simulator windows

Chapter contents

Introduction GR-11
Design object icons and their meaning GR-12

Setting fonts GR-13
Font scaling GR-13
Controlling fonts in an X-session GR-13

Main window GR-14
Main window menu bar GR-20
Main window toolbar GR-34
Main window status bar GR-19
Main window dialogs GR-37

Active Processes pane GR-108
Active Processes dialogs GR-109

Assertions pane GR-110
Assertions pane dialogs GR-113

Code coverage panes GR-116
Code coverage toolbar GR-127

Dataflow window GR-128
Dataflow window menu bar GR-129
Dataflow window toolbar GR-132
Dataflow window dialogs GR-135

Functional Coverage pane GR-143
Functional Coverage pane dialogs GR-145

List window GR-153
List window menu bar. GR-154
List window dialogs GR-156

Locals pane. GR-166
Locals dialogs GR-167

Memory windows GR-169
Memory popup menu commands GR-173
Memory dialogs GR-175

Objects pane GR-184
Objects dialogs GR-186

Profile panes GR-192
Profiler popup menu commands GR-195
Profiler dialogs GR-197
Profiler toolbar GR-196

Source window. GR-199

Watch pane. GR-208

Adding objects to the pane GR-208
Expanding objects to show individual bits GR-209
Grouping and ungrouping objects. GR-209

Wave window GR-211
Wave window menu bar GR-216
Wave window toolbar. GR-220
Wave window dialogs. GR-225

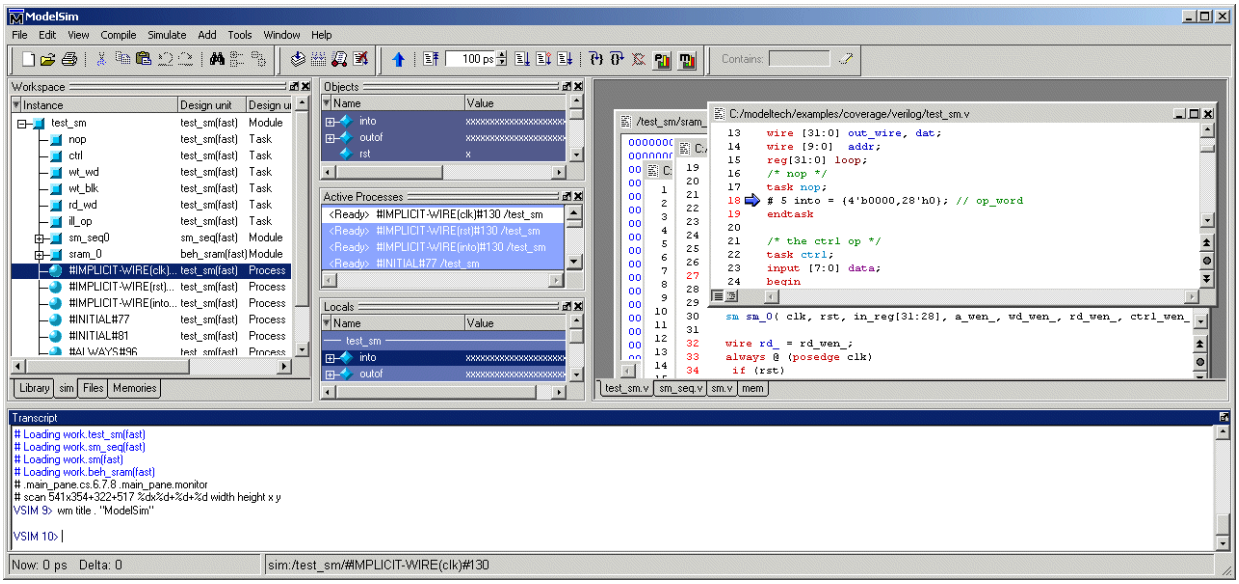
Customizing the GUI layout GR-258
Moving, docking, and undocking panes GR-258
Columnar information display GR-262
Quick access toolbars GR-263

Creating and managing breakpoints GR-264

This chapter describes the various windows, menus, and commands that comprise the ModelSim Graphical User Interface (GUI). Chapters earlier in the User's Manual also discuss the GUI but are organized more in a task-based format as opposed to the reference structure of this appendix.

Introduction

ModelSim's graphical user interface (GUI) consists of various windows that give access to parts of your design and numerous debugging tools. Some of the windows display as panes within the ModelSim Main window, some display as windows in the Multiple Document Interface (MDI) frame, and some display as standalone windows.



The following table summarizes all of the available windows and panes.

Window/pane name	Description	More details
Main	central GUI access point	"Main window" (GR-14)
Active Processes	displays all processes that are scheduled to run during the current simulation cycle	"Active Processes pane" (GR-108)
Assertions	manages PSL assertions	"Assertions pane" (GR-110)
Code coverage	a collection of panes that display code coverage data	"Code coverage panes" (GR-116)
Dataflow	displays "physical" connectivity and lets you trace events (causality)	"Dataflow window" (GR-128)
Functional Coverage	manages PSL cover directives	"Functional Coverage pane" (GR-143)
List	shows waveform data in a tabular format	"List window" (GR-153)
Locals	displays data objects that are immediately visible at the current PC of the selected process	"Locals pane" (GR-166)

Window/pane name	Description	More details
Memory	a Workspace tab and MDI windows that show memories and their contents	"Memory windows" (GR-169)
Watch	displays signal or variable values at the current simulation time	"Watch pane" (GR-208)
Objects	displays all declared data objects in the current scope	"Objects pane" (GR-184)
Profile	two panes that display performance and memory profiling data	"Profile panes" (GR-192)
Source	a text editor for viewing and editing HDL, SystemC, DO, etc. files	"Source window" (GR-199)
Transcript	keeps a running history of commands and messages and provides a command-line interface	"Transcript" (GR-16)
Wave	displays waveforms	"Wave window" (GR-211)
Workspace	provides easy access to projects, libraries, compiled design units, etc.	"Workspace" (GR-15)

The windows and panes are customizable in that you can position and size them as you see fit, and ModelSim will remember your settings upon subsequent invocations. See "Customizing the GUI layout" (GR-258) for more details.

Design object icons and their meaning

The color and shape of icons convey information about the language and type of a design object. Here is a list of icon colors and the languages they indicate:

icon color	language
light blue	Verilog
dark blue	VHDL
green	SystemC
magenta	PSL
orange	virtual object

Here is a list of icon shapes and the design object types they indicate:

icon shape	example	design object type
square		blocks (entity/architecture, module, SC module, etc.)

icon shape	example	design object type
circle		process
diamond		valued object (signals, nets, registers, SystemC channel, PSL endpoint, etc.)
caution sign		comparison object
triangle		PSL assertion
up arrow		PSL cover directive
diamond with red dot		an editable waveform created with the waveform editor

Setting fonts

You may need to adjust font settings to accommodate the aspect ratios of wide screen and double screen displays or to handle launching ModelSim from an X-session.

Font scaling

To change font scaling, select **Tools > Options > Adjust Font Scaling**. You'll need a ruler to complete the instructions in the lower right corner of the dialog. When you have entered the pixel and inches information, click OK to close the dialog. Then, restart ModelSim to see the change. This is a one time setting; you shouldn't have to set it again unless you change display resolution or the hardware (monitor or video card). The font scaling applies to Windows and UNIX operating systems. On UNIX systems, the font scaling is stored based on the $DISPLAY environment variable.

Controlling fonts in an X-session

When executed via an X-session (e.g., Exceed, VNC), ModelSim uses font definitions from the .Xdefaults file. To ensure that the fonts look correct, create a .Xdefaults file with the following lines:

```
vsim*Font: -adobe-courier-medium-r-normal--*-120-*-*-*-*-*
vsim*SystemFont: -adobe-courier-medium-r-normal--*-120-*-*-*-*-*
vsim*StandardFont: -adobe-courier-medium-r-normal--*-120-*-*-*-*-*
vsim*MenuFont: -adobe-courier-medium-r-normal--*-120-*-*-*-*-*
```

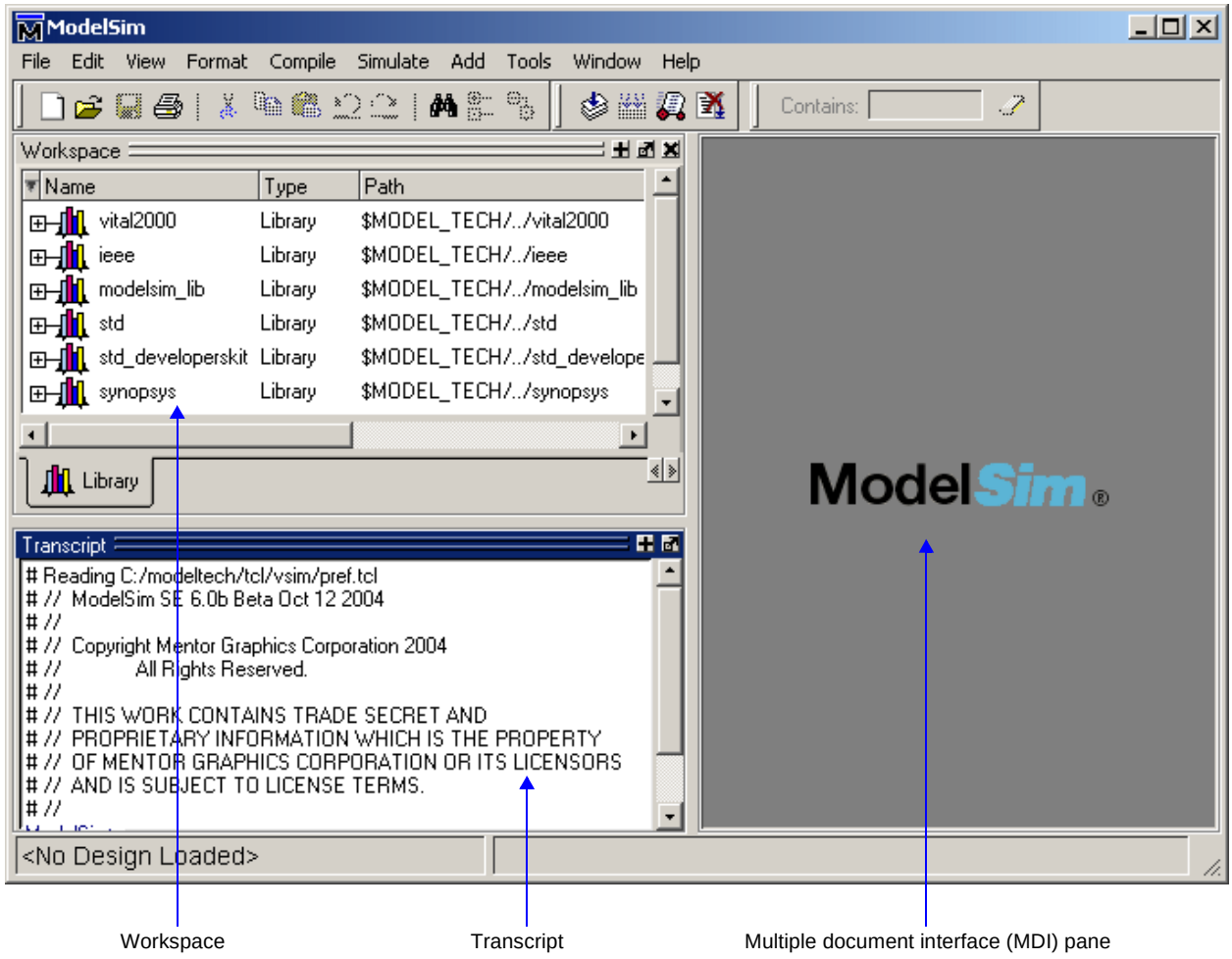
Alternatively, you can choose a different font. Use the program "xlsfonts" to identify which fonts are available on your system.

Also, the following command can be used to update the X resources if you make changes to the .Xdefaults and wish to use those changes on a UNIX machine:

```
xrdb -merge .Xdefaults
```

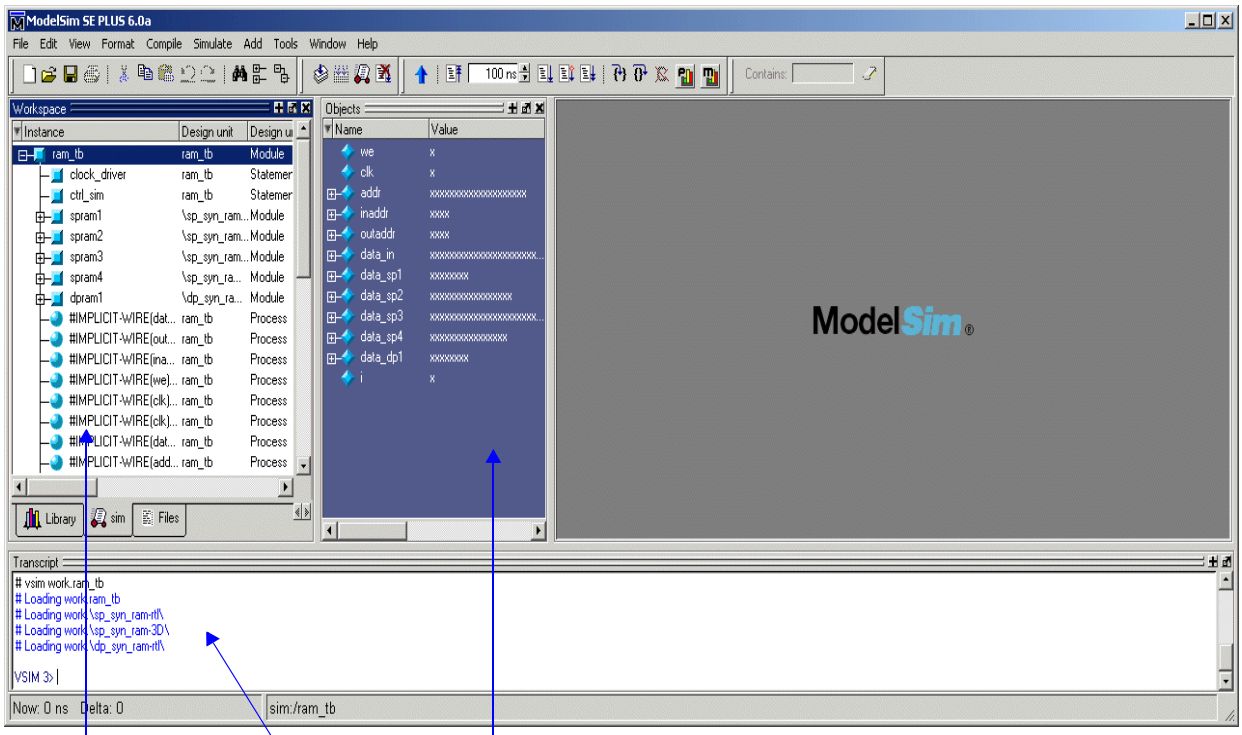
Main window

The primary access point in the ModelSim GUI is called the Main window. Here is what the Main window looks like the very first time you start the tool:



The Main window provides convenient access to design libraries and objects, source files, debugging commands, simulation status messages, etc.

When you load a design, or bring up debugging tools, ModelSim adds additional panes or opens new windows. For example, here is the Main window after loading a simple design.



Workspace tabs organize design elements in a hierarchical tree structure

The Transcript pane reports status and provides a command-line interface

The Objects pane displays data objects in the current scope

Notice some of the elements that appear:

- Workspace tabs organize and display design objects in a hierarchical tree format
- The Transcript pane tracks command history and messages and provides a command-line interface where you can enter ModelSim commands
- The Objects pane displays design objects such as signals, nets, generics, etc. in the current design scope

Workspace

The Workspace provides convenient access to projects, libraries, design files, compiled design units, simulation/dataset structures, and Waveform Comparison objects. It can be hidden or displayed by selecting **View > Workspace** (Main window).

The Workspace can display the types of tabs listed below.

- **Project tab**
  Shows all files that are included in the open project. See *Chapter 2 - Projects* in the *ModelSim User's Manual* for details.

- **Library tab**
  Shows design libraries and compiled design units. See "Managing library contents" (UM-61) for details.
- **Structure tabs**
  Shows a hierarchical view of the active simulation and any open datasets. There is one tab for the current simulation (named "sim") and one tab for each open dataset. See "Viewing dataset structure" (UM-228) for details.

  An entry is created by each object within the design. When you select a region in a structure tab, it becomes the *current region* and is highlighted. The Source window (GR-199) and Objects pane (GR-184) change dynamically to reflect the information for that region. This feature provides a useful method for finding the source code for a selected region because the system keeps track of the pathname where the source is located and displays it automatically, without the need for you to provide the pathname.

  Also, when you select a region in the structure pane, the "Active Processes pane" (GR-108) is updated. The Active Processes window will in turn update the Locals pane (GR-166).
- **Files tab**
  Shows the source files for the loaded design.
- **Memories tab**
  Shows a hierarchical list of all memories in the design. To display this tab, select **View > Debug Windows > Memory**. When you select a memory on the tab, a memory contents page opens in the MDI frame. See "Memory windows" (GR-169).
- **Compare tab**
  Shows comparison objects that were created by doing a waveform comparison. See *Chapter 9 - Waveform analysis* for details.

Transcript

The Transcript portion of the Main window maintains a running history of commands that are invoked and messages that occur as you work with ModelSim. When a simulation is running, the Transcript displays a VSIM prompt, allowing you to enter command-line commands from within the graphic interface.

You can scroll backward and forward through the current work history by using the vertical scrollbar. You can also use arrow keys to recall previous commands, or copy and paste using the mouse within the window (see "Main and Source window mouse and keyboard shortcuts" (UM-605) for details).

Saving the transcript file

Variable settings determine the filename used for saving the transcript. If either **PrefMain(file)** in the *modelsim.tcl* file or **TranscriptFile** in the *modelsim.ini* file is set, then the transcript output is logged to the specified file. By default the **TranscriptFile** variable in *modelsim.ini* is set to *transcript*. If either variable is set, the transcript contents are always saved and no explicit saving is necessary.

If you would like to save an additional copy of the transcript with a different filename, click in the Transcript pane and then select **File > Save As**, or **File > Save**. The initial save must be made with the **Save As** selection, which stores the filename in the Tcl variable **PrefMain(saveFile)**. Subsequent saves can be made with the **Save** selection. Since no

automatic saves are performed for this file, it is written only when you invoke a **Save** command. The file is written to the specified directory and records the contents of the transcript at the time of the save.

Using the saved transcript as a macro (DO file)

Saved transcript files can be used as macros (DO files). See the **do** command (CR-151) for more information.

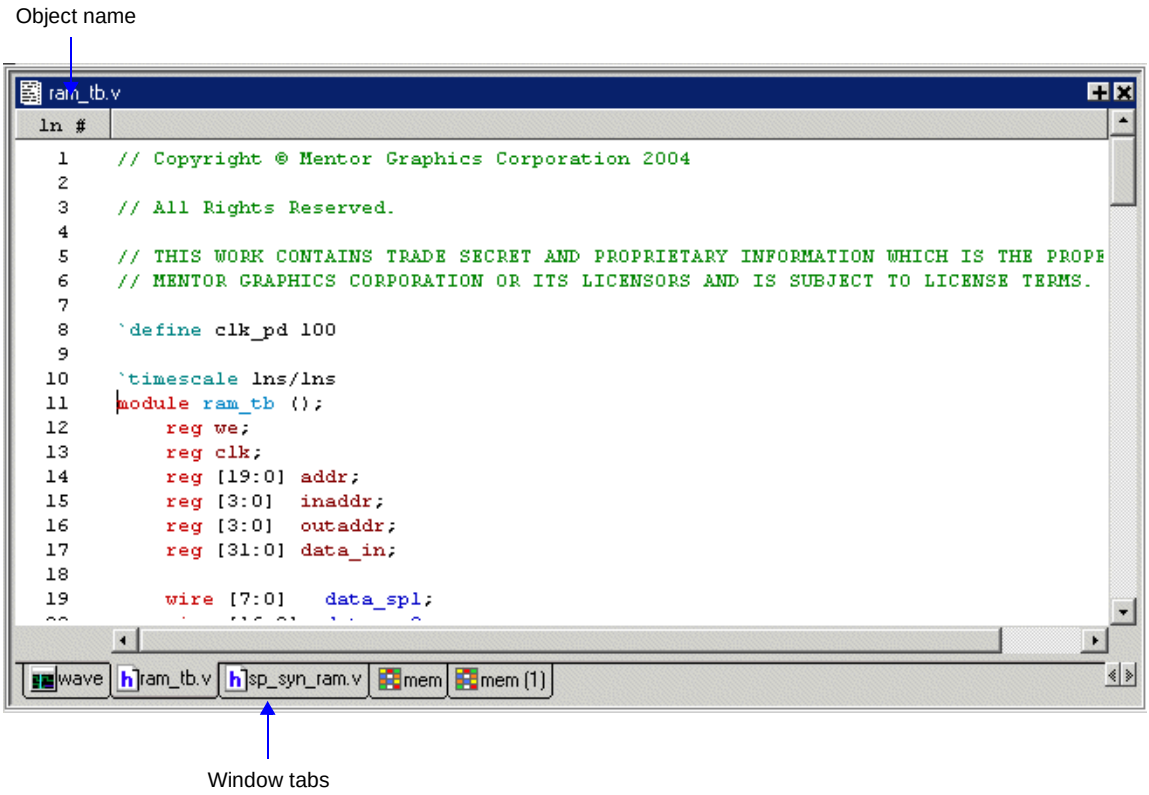
Disabling creation of the transcript file

You can disable the creation of the transcript file by using the following ModelSim command immediately after ModelSim starts:

```
transcript file ""
```

Multiple document interface (MDI) frame

The MDI frame is an area in the Main window where source editor, memory content, and wave windows display. The frame allows multiple windows to be displayed simultaneously, as shown below. A tab appears for each window.

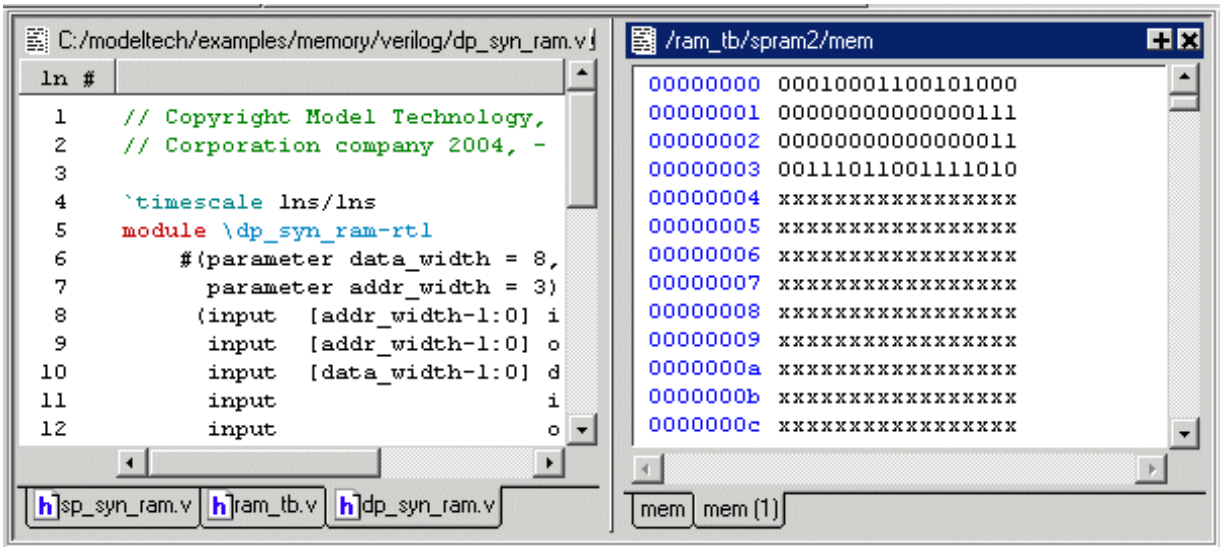


The object name is displayed in the title bar at the top of the window. You can switch between the windows by clicking on a tab.

Organizing windows with tab groups

The MDI can quickly become unwieldy if many windows are open. You can create "tab groups" to help organize the windows. A tab group is a collection of tabs that are separated from other groups of tabs.

The graphic below shows how the collection of files in the picture above could be organized into two tab groups.



The commands for creating and organizing tab groups are accessed by right-clicking on any window tab. The table below describes the commands associated with tab groups:

Command	Description
New Tab Group	Creates a new tab group containing the selected tab
Move Next Group	Moves the selected tab to the next group in the MDI
Move Prev Group	Moves the selected tab to the previous group in the MDI
View > Vertical / Horizontal	Arranges tab groups top-to-bottom (vertical) or right-to-left (horizontal)

Note that you can also move the tabs within a tab group by dragging them with the middle mouse button.

Main window status bar

Fields at the bottom of the Main window provide the following information about the current simulation:

Field	Description
Project	name of the current project
Now	the current simulation time
Delta	the current simulation iteration number
Profile Samples	the number of profile samples collected during the current simulation
Memory	the total memory used during the current simulation
environment	name of the current context (object selected in the active Structure tab of the Workspace)
line/column	line and column numbers of the cursor in the active Source window

Main window menu bar

The following commands are available from the Main window menu bar. Many of the commands become enabled/disabled depending upon which pane of the window is currently active.

Some commands behave differently depending upon which window pane is active. For example, if a Source window is active, the **File > Open** command opens a standard Open dialog. However, if a memory list is active, the command opens the Load Memory dialog.

File menu

New	provides these options: Folder – create a new folder in the current directory Source – create a VHDL, Verilog, or Other source file Project – create a new project Library – create a new design library and mapping; see "Create a New Library dialog" (GR-38) Window – create a new window of the specified type
Open	open a file; specify type by clicking Files of Type drop-down list
Load	executes or loads a previously saved format file (e.g. wave.do, waveedit.do, watch.do) in the Wave or Watch windows
Close	close a file or simulation
Import	provides these options: Library – import FPGA libraries; see "Importing FPGA libraries" (UM-69) EVCD – import a previously saved extended EVCD file; see "Signal mapping and importing EVCD files" (GR-295); this item is enabled only when a Wave window is active in the MDI frame of the Main window
Export	provides these options: Waveform – export a created waveform; see "Exporting waveforms to a stimulus file" (GR-294); Image – saves a bitmap image of the Wave window. These items are enabled only when a Wave window is active in the MDI frame of the Main window
Save	saves datasets, waveform formats, or waveform edits – depending upon which pane is active. If the Wave window is active in the MDI frame, this command will open the Save Format dialog, which will allow the user to save waveform formats, waveform edits, or both. If the Workspace pane is active and the sim tab selected, this command allows the user to save the current dataset as a .wlf file.
Save As	save a file with a new name; the file to be saved depends upon which pane is active (e.g., if the Transcript pane is active, this command saves the transcript)

Report	produce a textual report from the active pane
Change Directory	change to a different working directory; disabled if you have a project or dataset open or a simulation running
Use Source	specify an alternative file to use for the current source file; this alternative source mapping exists for the current simulation only
Source Directory	add to a list of directories to search for source files; you can set this permanently using the **SourceDir** variable in the *modelsim.tcl* file
Environment	provides different options depending upon which pane is selected: Follow Current Dataset – update the Objects pane based on the current dataset Fix to dataset <name> – fix the Objects pane content to a specific dataset Follow Process Selection – update the Locals pane content based on the selection in the "Active Processes pane" (GR-108) Fix to Current Process – fix the Locals pane content to the current process; do not update Follow Context Selection – update the pane contents based upon the selection in the structure tab of the Workspace Fix to Current Context – maintain the current view; do not update
Add to Project	provides these options: New File – add new files to the open Project; see "Step 2 — Adding items to the project" (UM-41) Existing File – add existing files to the open Project; see "Step 2 — Adding items to the project" (UM-41) Simulation Configuration – add an object representing a design unit(s) and its associated simulation options; see "Creating a Simulation Configuration" (UM-48) Folder – add an organization folder to the current project; see "Organizing projects with folders" (UM-50)
Page Setup	configure page settings for printing waveforms; this item is enabled only when a wave window is active in the MDI frame of the Main window (Windows only)
Print	print the contents of the Transcript pane, a Source window, or an embedded wave window
Print Postscript	save or print the active Source file or wave window as a Postscript file (Windows only)
Recent Directories	displays a list of the most recent working directories
Recent Projects	displays a list of the most recent projects
Quit	quit ModelSim

Edit menu

Undo	undo the last edit
Redo	redo the previously undone edit
Cut	cut the selected text
Copy	copy the selected text
Paste	paste the previously cut or copied text
Delete	deletes an object from the active Wave window in the MDI frame
Clear	clear the Transcript pane
Select All	select all text in the active window
Unselect All	deselect all text in the active window
Expand	provides these options: Expand Selected – expand the hierarchy of the selected instance Collapse Selected – collapse the hierarchy of the selected instance Expand All – expand the hierarchy of all instances in the active window Collapse All – collapse the hierarchy of all instances in the active window
Advanced	provides these options: Force – force a value on the item selected in the Objects pane; see "Force Selected Signal dialog" (GR-186) Noforce – cancel a previous force command Clock – define a clock; see "Define Clock dialog" (GR-188) Change – change the value of the selected object Comment/Uncomment Selected – add or remove comment characters from the lines selected in the active Source window Examine – display the current value of the object selected in the active Source window; same as the **examine** (CR-162) command Describe – display information about the object selected in the active Source window; same as the **describe** command (CR-147) Drivers – list the names of all drivers of the object selected in the active Source window Readers – list the names of all readers of the object selected in the active Source window
Wave	this menu is enabled only when a wave window is active in the MDI frame of the Main window; most of the options on this menu relate to Waveform Editor (see "Editing waveforms" (GR-290); Edit Cursor – edit the name or time of the selected cursor Delete Cursor – delete the selected cursor Wave Signal Search – search the wave window for a signal name or value, or an expression

Find	search the active window for the specified text string
Replace	find and replace text in the active window
Find in Files	search for text in saved files
Previous Coverage Miss	find the previous line with missed coverage in the active Source window; see "Viewing coverage data in the Source window" (UM-341)
Next Coverage Miss	find the next line with missed coverage in the active Source window; see "Viewing coverage data in the Source window" (UM-341)

View menu

Debug Windows	open various debugging windows/panes
Sort	sort the Objects or Locals pane
Filter	select the type of objects displayed in the Objects, Locals, or Memory list panes
Justify	justify left or right the object names in the Locals window
Split screen	split the active memory contents window; see "Splitting the memory contents window" (GR-172)
Datasets	open the Dataset Browser to open, close, rename, or activate a dataset
Wave	this menu is enabled only when a wave window is active in the MDI frame of the Main window provides these options: Object Declaration – open the source file in the Source window and highlight the declaration for the currently selected object Refresh Display – clear the Wave window, empty the file cache, and rebuild the window from scratch Zoom – choose various zoom commands Mouse Mode – toggle mouse pointer between Select Mode (click left mouse button to select, drag with middle mouse button to zoom) and Zoom Mode (drag with left mouse button to zoom, click middle mouse button to select) Cursors – choose a cursor to go to from a list of available cursors Bookmarks – choose a bookmark to go to from a list of available bookmarks See "Code coverage panes" (GR-116) for details on these panes.

Code Coverage	provides these options: Current Exclusions – hide or show the Exclusions pane Missed Coverage – hide or show the Missed Coverage pane Instance Coverage – hide or show the Instance Coverage pane Details – hide or show the Details pane See "Code coverage panes" (GR-116) for details on these panes.
Profile	provides these options: View – hide or show Profile pane View Details – hide or show Profile Details pane
Source	provides these options: Show line numbers–toggle display of line numbers Show language templates–toggle display of the Language templates (GR-201) pane Read Only–toggles read only status of selected source file
Workspace	hide or show the Workspace (GR-15) pane
Show Tabs	toggle display of window tabs in MDI frame
Tabbed MDI	toggle MDI frame mode from tab-based mode to floating window mode; see "Multiple document interface (MDI) frame" (GR-17) for more information
Goto	jump to the specified line number or memory address
Encoding	select from alphabetical list of encoding names that enable proper display of character representations used by various operating systems or file systems, such as Unicode, ASCII, or Shift-JIS.
Project Settings	show information about the open project
Properties	show information about the object selected in the Workspace

Format menu

The options on this menu are enabled only when a wave window is active in the MDI frame of the Main window. To view a wave window in the MDI frame, use the **view mdiwave** command.

Radix	set the selected objects' radix
Format	set the waveform format for the selected objects – Literal, Logic, Event, Analog
Color	set the color for the selected objects from a color palette
Height	set the waveform height in pixels for the selected objects

Compile menu

Compile	compile source files; not enabled if you have a project open
Compile Options	set various compile options; see "Compiler Options dialog" (GR-60); disabled if you have a project open
SystemC Link	collects the object files created in the different design libraries, and uses them to build a shared library (.so) in the current work library
Compile All	compile all files in the open project; see "Step 3 — Compiling the files" (UM-43) for details; disabled if you don't have a project open
Compile Selected	compile the files selected in the project tab; disabled if you don't have a project open
Compile Order	set the compile order of the files in the open project; see "Changing compile order" (UM-46) for details; disabled if you don't have a project open
Compile Report	report on the compilation history of the selected file(s) in the project; disabled if you don't have a project open
Compile Summary	report on the compilation history of all files in the project; disabled if you don't have a project open

Simulate menu

Design Optimization	open the Design Optimization dialog to configure simulation optimizations
Start Simulation	load the selected design unit; see "Start Simulation dialog" (GR-76)
Runtime Options	set various simulation runtime options; see "Runtime Options dialog" (GR-85)
Run	provides these options: Run <default> – run simulation for one default run length; change the run length with **Simulate > Runtime Options**, or use the Run Length text box on the toolbar Run -All – run simulation until you stop it; see also the **run** command (CR-252) Continue – continue the simulation; see also the **run** command (CR-252) and the **-continue** option Run -Next – run to the next event time Step – single-step the simulator; see also the **step** command (CR-272) Step -Over – execute without single-stepping through a subprogram call Restart – reload the design elements and reset the simulation time to zero; only design elements that have changed are reloaded; you specify whether to maintain the following after restart–List and Wave window environment, breakpoints, logged signals, virtual definitions, and assertion and functional coverage settings; see also the **restart** command (CR-246)
Break	stop the current simulation run
End Simulation	quit the current simulation run

Add menu

Wave	add objects from the selected pane to the Wave window; objects vary depending on which pane is active–for example, if the Assertions pane is selected, you can add assertions, whereas if the Objects pane is selected, you can add signals
List	add objects from the selected pane to the List window; objects vary depending on which pane is active–for example, if the Assertions pane is selected, you can add assertions, whereas if the Objects pane is selected, you can add signals

Log	add objects from the selected pane to the active log file; objects vary depending on which pane is active–for example, if the Assertions pane is selected, you can add assertions, whereas if the Objects pane is selected, you can add signals
Divider	insert a divider in the Wave window; this item is enabled only when a Wave window is active in the MDI frame of the Main window
Breakpoint	add a breakpoint on the selected signal; see "Signal breakpoints" (GR-264); this item is enabled only when a wave window is active in the MDI frame of the Main window
Bookmark	add a bookmark with the current zoom range and scroll location; see "Saving zoom range and scroll position with bookmarks" (UM-250); this item is enabled only when a wave window is active in the MDI frame of the Main window
Cursor	add a cursor to the Wave window; "Saving zoom range and scroll position with bookmarks" (UM-250); this item is enabled only when a wave window is active in the MDI frame of the Main window
Window Pane	split the pathname, values and waveform window panes to provide room for a new waveset; this item is enabled only when a wave window is active in the MDI frame of the Main window

Tools menu

Waveform Compare	see "Waveform Compare sub-menu" (GR-29)
Code Coverage	see "Code Coverage sub-menu" (GR-30)
Functional Coverage	see "Functional Coverage sub-menu" (GR-30)
Toggle Coverage	add toggle coverage tracking to the Objects pane; see "Enabling Toggle coverage" (UM-343)
Profile	see "Profile sub-menu" (GR-31)
Breakpoints	open the Breakpoints dialog box; see "Setting file-line breakpoints" (GR-203) for details
Bookmarks	add, edit, delete, and goto bookmarks; see "Saving zoom range and scroll position with bookmarks" (UM-250)
Dataset Snapshot	enable periodic saving of simulation data to a WLF file; see "Saving at intervals with Dataset Snapshot" (UM-231)

Combine Signals	combine the selected objects into a user-defined bus; see "Combining objects/creating busses" (UM-264)
C Debug	see "C Debug sub-menu" (GR-31)
Execute Macro	call and execute a .do or .tcl macro file
Macro Helper	**UNIX only** - invoke the Macro Helper tool; see "Macro helper" (UM-492)
Tcl Debugger	invoke the Tcl debugger, TDebug; see "The Tcl Debugger" (UM-493)
TclPro Debugger	invoke TclPro Debugger by Scriptics®, if installed; see "TclPro Debugger" (UM-497)
Options (all options are set for the current session only)	provides these options: Adjust Font Scaling – set font scaling factor when using dual monitors; you must restart ModelSim after setting the font scaling Transcript File – set a transcript file to save for this session only Command History – set a file for saving command history only, no comments Save File – set filename for Save Transcript, and Save Transcript As Saved Lines – limit the number of lines saved in the transcript (default is 5000) Line Prefix – specify the comment prefix for the transcript Update Rate – specify the update frequency for the Main window status bar ModelSim Prompt – change the title of the ModelSim prompt VSIM Prompt – change the title of the VSIM prompt Paused Prompt – change the title of the Paused prompt HTML Viewer – specify the path to your browser; used for displaying online help PDF Viewer – specify the path to your PDF viewer; used for displaying documentation Examine Now – sets the examine command to read values at the current simulation time Examine Current Cursor – sets the examine command to read values at the active cursor time Wave Preferences – format the Wave window; see "Window Preferences dialog" (GR-255) Drag and Drop Preferences – set the action to be performed when a file is dragged and dropped into the Project, Transcript or Wave windows
Edit Preferences	set various preference variables; see "Preference variables located in Tcl files" (UM-540) for more information
Save Preferences	save current ModelSim settings to a Tcl preference file; see "Preference variables located in Tcl files" (UM-540) for more information

Waveform Compare sub-menu

Start Comparison	start a comparison; see "Start Comparison dialog" (GR-243)
Comparison Wizard	use the Comparison Wizard to configure a comparison; see "Comparison Wizard" (UM-271)
Run Comparison	compute the number of differences from time zero to the end of the simulation run, from time zero until the maximum total number of differences per signal limit is reached, or from time zero until the maximum total number of differences for all signals compared is reached
Add	provides these options: Compare by Signal – designate signals for the comparison; see "Add Signal Options dialog" (GR-244) Compare by Region – designate a reference region for the comparison; see "Add Comparison by Region dialog" (GR-245) Clocks – Define clocks for a clocked comparison; see "Add Clocks dialog" (GR-246)
Options	define waveform comparison options; see "Comparison Options dialog" (GR-247)
Differences	provides these options: Clear – clear all differences and reset the comparison Show – display differences in the Main window transcript Save – save differences to a file that can be reloaded later
Rules	provides these options: Show – display rules used to set up the comparison Save – save rules to a file so you can recreate the comparison later
Reload	open previously saved differences and rules files; see "Saving and reloading comparison results" (UM-282)

Code Coverage sub-menu

Load	load a previously saved code coverage analysis; see "Saving and reloading coverage data" (UM-354)
Save	save current code coverage data; see "Saving and reloading coverage data" (UM-354)
Reports	produce textual output of code coverage statistics; see "Reporting coverage data" (UM-350)
Clear	clear code coverage data from the active database
Hide coverage data	hide/show the statement coverage column in a Source window
Hide branch coverage	hide/show the branch coverage column in a Source window
Show coverage numbers	display numbers rather than graphics in the coverage columns of a Source window
Show coverage by Instance	display counts only for the instance selected in the Workspace structure tab

Functional Coverage sub-menu

Design Unit Mode	when selected, the functional coverage pane is organized by design unit
Save	save current functional coverage data; see "Saving functional coverage data" (UM-395)
Reload	reload a previously saved functional coverage data file; see "Reloading/merging functional coverage data" (UM-396)
Reports	create reports on functional coverage data; see "Reporting functional coverage statistics" (UM-391)
Configure	configure coverage directives; see "Configuring functional coverage directives" (UM-386)
Filter	filter directives from the Functional Coverage pane; see "Filtering data in the pane" (UM-388)
Comment	add a comment to the selected directive(s)
Clear	clear data from the active functional coverage database; see "Clearing functional coverage data" (UM-397)

Profile sub-menu

Performance	enable statistical performance sampling; see A statistical sampling profiler (UM-318)
Memory	enable memory allocation profiling; see A memory allocation profiler (UM-318)
Collapse Sections	report collapsed processes and functions
Clear Profile Data	clear all statistical performance and memory allocation data; see Collecting memory allocation and performance data (UM-321)
Profile Report	enable the Profile Report dialog; see Reporting profiler results (UM-331)

C Debug sub-menu

Start C Debug	turn on C Debug so you can set breakpoints and step through C code
C Debug setup	specify the location of your gdb installation; see "C Debug setup dialog" (GR-99)
Enable auto step	configure C Debug to run in auto-step mode; see "Identifying all registered function calls" (UM-407)
Run	provide access to step, step-over, run-continue, and run-finish commands
Quit C Debug	turn off C Debug; do this before exiting ModelSim
Init mode	configure C Debug to run in initialization mode; see "Debugging functions during elaboration" (UM-410)
Complete load	cancel initialization mode and complete loading the rest of your design
Auto find bp	set breakpoints at all the FLI/PLI/VPI function entry points that are known (registered) when you make this menu selection
Info bp	list all currently set breakpoints including the source file names, line numbers, and breakpoint ids
Show	show the values of the local variables and arguments of the current C function

Traceback	identify the HDL source line from which the C function was called, if known; when running in initialization mode, no HDL information is available, and this command will list only the gdb traceback stack
C Interrupt	"re-activate" the C debugger when you are stopped in HDL code
Command entry	open a command prompt dialog so you can enter commands even if the CDBG> prompt is inaccessible;
Refresh	reopen a C source file if you close the Source window inadvertently while stopped in the C debugger

Window menu

Initial Layout	restore all windows to the size and placement of the initial full-screen layout
Cascade	cascade all open windows
Tile Horizontally	tile all open windows horizontally
Tile Vertically	tile all open windows vertically
Icon Children	icon all but the Main window
Icon All	icon all windows
Deicon All	restore all windows
Customize	add a button to either the tool or status bar of the specified window; see "Customize Toolbar dialog" (GR-106)
<window_name>	list of up to nine open windows including one for each file opened in the Source window; use the Windows menu item to see a complete list
Windows	open a dialog with a complete list of open windows

Help menu

About ModelSim	display ModelSim application information (e.g., software version)
Release Notes	view current release notes
Welcome Menu	open the Welcome screen
HTML/PDF Documentation	open and read ModelSim documentation in PDF or HTML format; PDF files can be read with a free Adobe Acrobat reader available on the ModelSim installation CD or from www.adobe.com
Tcl Help	open the Tcl command reference (man pages) in Windows help format
Tcl Syntax	open Tcl syntax documentation in a browser
Tcl Man Pages	open the Tcl /Tk 8.3 manual in HTML format
Technotes	select a technical note to view from the drop-down list

Main window toolbar

Buttons on the Main window toolbar give you quick access to various ModelSim commands and functions.

Main window toolbar buttons		
Button	**Menu equivalent**	**Command equivalents**
New File create a new source file	File > New > Source	
Open open the Open File dialog	File > Open	
Save save the contents of the active pane	File > Save	
Print open the Print dialog	File > Print	
Cut cut the selected text to the clipboard	Edit > Cut	
Copy copy the selected text to the clipboard	Edit > Copy	
Paste paste the clipboard text	Edit > Paste	
Undo undo the last edit	Edit > Undo	
Redo redo the last undone edit	Edit > Redo	

Main window toolbar buttons		
Button	**Menu equivalent**	**Command equivalents**
Find find text in the active window	Edit > Find	
Collapse All collapse all instances in the active window	Edit > Expand > Collapse All	
Expand All expand all instance in the active window	Edit > Expand > Expand All	
Compile open the Compile Source Files dialog to select files for compilation	Compile > Compile	**vcom** (CR-311) or **vlog** (CR-358)
Compile All compile all files in the open project	Compile > Compile All	**vcom** (CR-311) or **vlog** (CR-358)
Simulate load the selected design unit or simulation configuration object	Simulate > Start Simulation	**vsim** (CR-373)
Break stop the current simulation run	Simulate > Break	
Environment up move up one level in the design hierarchy		
Restart reload the design elements and reset the simulation time to zero, with the option of maintaining various settings and objects	Simulate > Run > Restart	**restart** (CR-246)
Run Length specify the run length for the current simulation	Simulate > Runtime Options	**run** (CR-252)

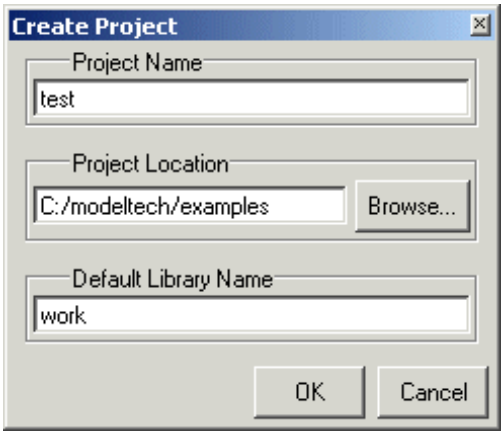
Main window toolbar buttons		
Button	**Menu equivalent**	**Command equivalents**
Run run the current simulation for the specified run length	Simulate > Run > Run <default_run_length>	**run** (CR-252)
Continue Run continue the current simulation run until the end of the specified run length or until it hits a breakpoint or specified break event	Simulate > Run > Continue	**run -continue**
Run -All run the current simulation forever, or until it hits a breakpoint or specified break event	Simulate > Run > Run -All	**run** (CR-252) **-all**
Step step the current simulation to the next statement	Simulate > Run > Step	**step** (CR-272)
Step Over HDL statements are executed but treated as simple statements instead of entered and traced line by line	Simulate > Run > Step -Over	**step** (CR-272) **-over**
C Interrupt reactivates the C debugger when stopped in HDL code	Tools > C Debug > C Interrupt	**cdbg** (CR-79) **interrupt**
Memory Profiling enable collection of memory usage data	Tools > Profile > Memory	
Performance Profiling enable collection of statistical performance data	Tools > Profile > Performance	
Contains filter items in Objects pane		

Main window dialogs

This section describes the dialogs that are accessed via the Main window menu bar. The dialogs are listed in the order in which they appear on the menus, top-to-bottom and left-to-right (i.e., starting with the File menu and progressing across the menu bar). Not all dialogs are documented (e.g., Change Directory dialog).

Create Project dialog

Purpose	Menu command	Additional information
Create a new project	**File** > **New** > **Project**	"Getting started with projects" (UM-40)

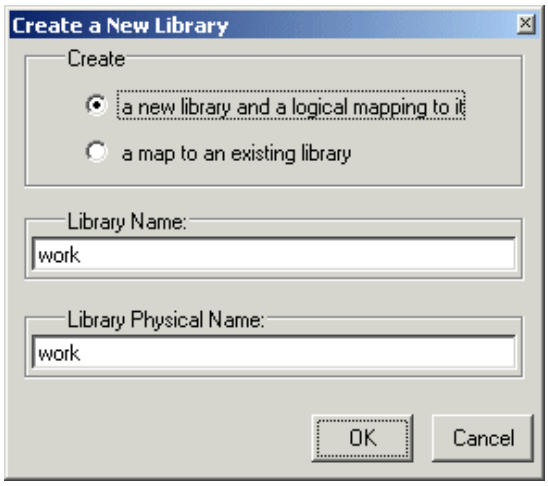


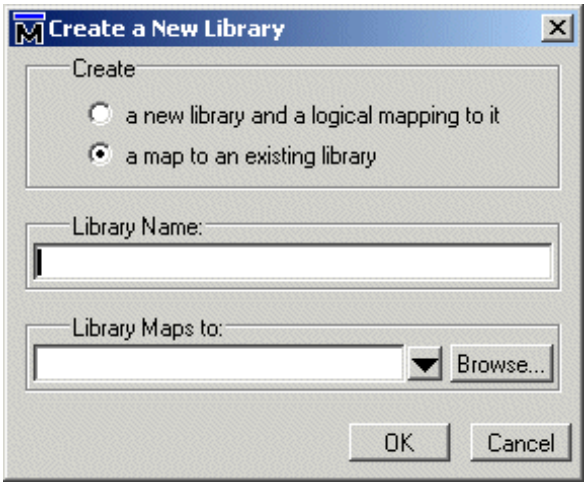
The Create Project dialog includes these options:

- **Project Name**
  The name of the new project.
- **Project Location**
  The directory in which the project *.mpf* file will be created.
- **Default Library Name**
  The name of the working library. See "Working library versus resource libraries" (UM-58) for more details on work libraries. You can generally leave the **Default Library Name** set to "work." The name you specify will be used to create a working library subdirectory within the Project Location.

Create a New Library dialog

Purpose	Menu command	Additional information
Create a new library	**File > New > Library**	"Working with design libraries" (UM-60)



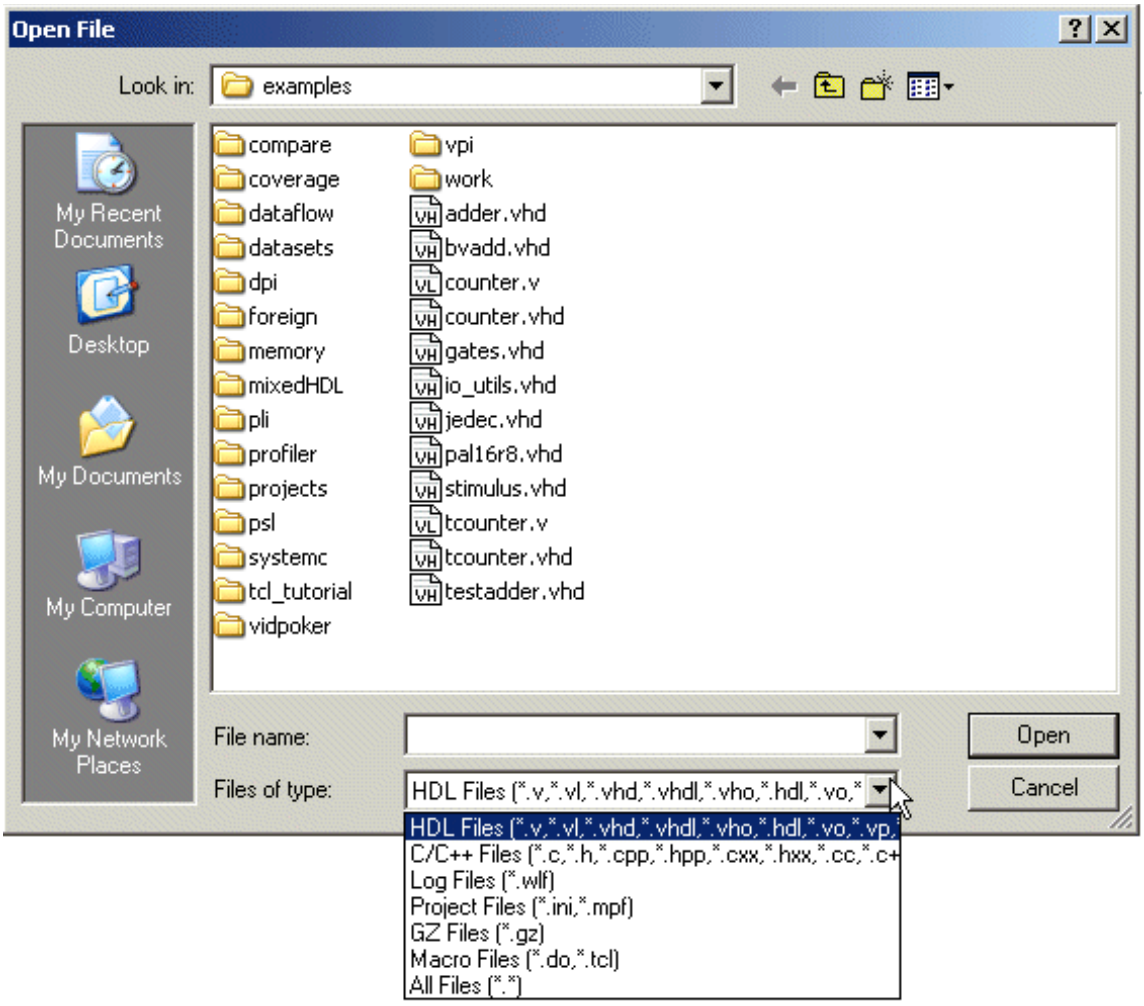


The Create a New Library dialog includes these options:

- **Create a new library and a logical mapping to it**
  Type the new library name into the **Library Name** field. This creates a library sub-directory in your current working directory, initially mapped to itself. Once created, the mapped library is easily remapped to a different library.
- **Create a map to an existing library**
  Type the new library name into the **Library Name** field, then type into the **Library Maps to** field or **Browse** to select a library for the mapping.
- **Library Name**
  The name of the new library.
- **Library Physical Name**
  The physical path to the library. Visible only when **Create a new library and a logical mapping to it** is selected.
- **Library Maps to**
  Type or **Browse** for a mapping for the specified library. This field is visible and can be changed only when the **Create a map to an existing library** option is selected.

Open File dialog

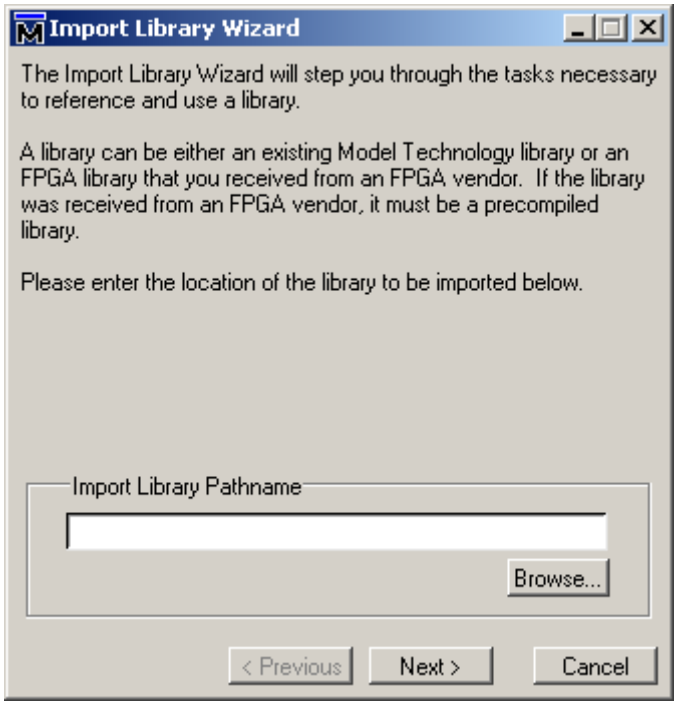
Purpose	Menu command	Additional information
Open a previously saved file	**File > Open**	



The Open File dialog is the standard Open dialog used by your operating system, and it may look slightly different on your system than the one above. The key field in this dialog is **Files of Type**. Select an item here to show the file type you want to open.

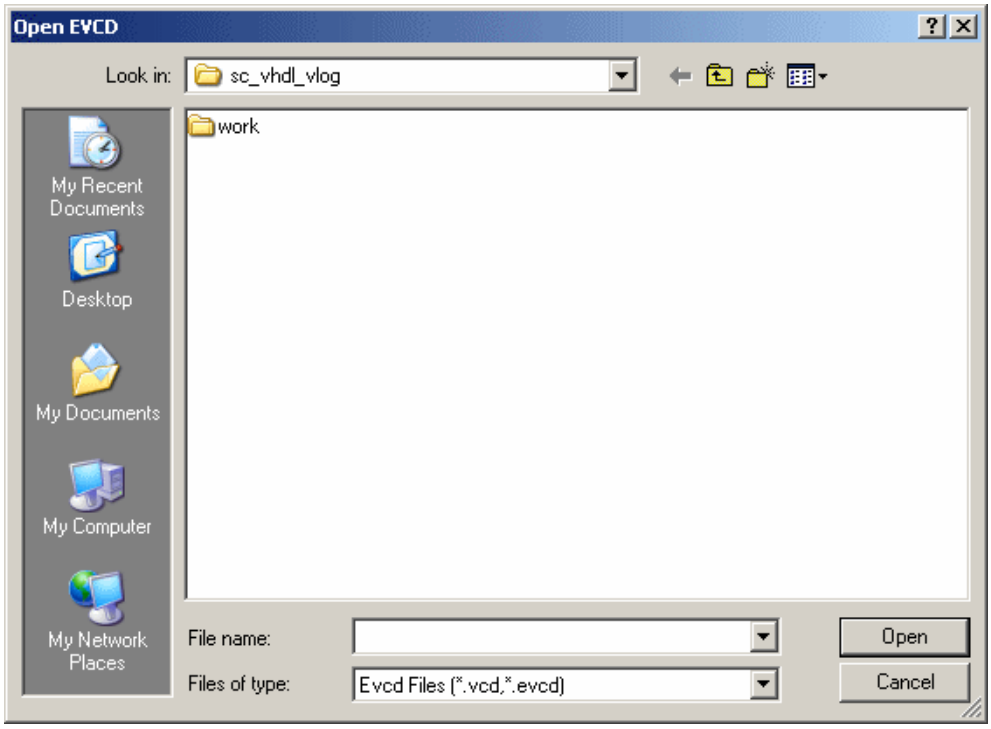
Import Library Wizard dialog

Purpose	Menu command	Additional information
Map to a vendor-supplied library	**File > Import > Library**	"Importing FPGA libraries" (UM-69)



Evcd Import dialog

Purpose	Menu command	Additional information
Import a previously saved EVCD file as stimulus	**File > Import > EVCD**	"Signal mapping and importing EVCD files" (GR-295)

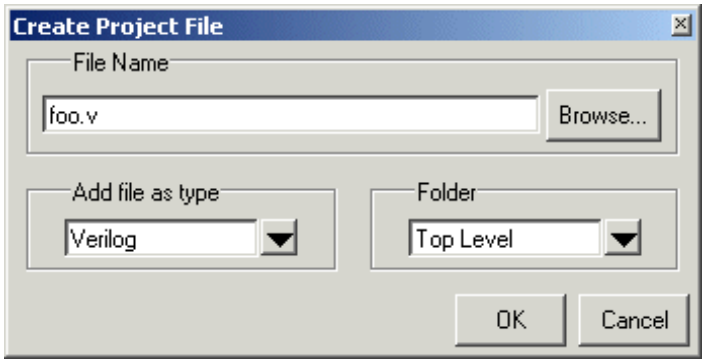


Save As dialog

Purpose	Menu command	Additional information
Save a file for the first time	**File > Save As**	

Create Project File dialog

Purpose	Menu command	Additional information
Add a new file to the open project	**File > Add to Project> New File**	"Create New File" (UM-42)

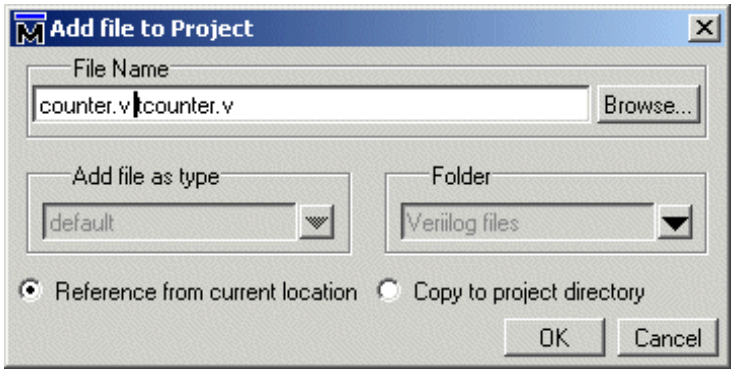


The Create Project File dialog includes these options:

- **File Name**
  The name of the new file.
- **Add file as type**
  Select from a variety of types (e.g., Tcl, Verilog, SDF, etc.).
- **Folder**
  The organization folder in which you want the new file placed. You must first create folders in order to access them here. See "Organizing projects with folders" (UM-50) for details.

Add file to Project dialog

Purpose	Menu command	Additional information
Add an existing file to the open project	**File > Add to Project> Existing File**	"Add Existing File" (UM-42)

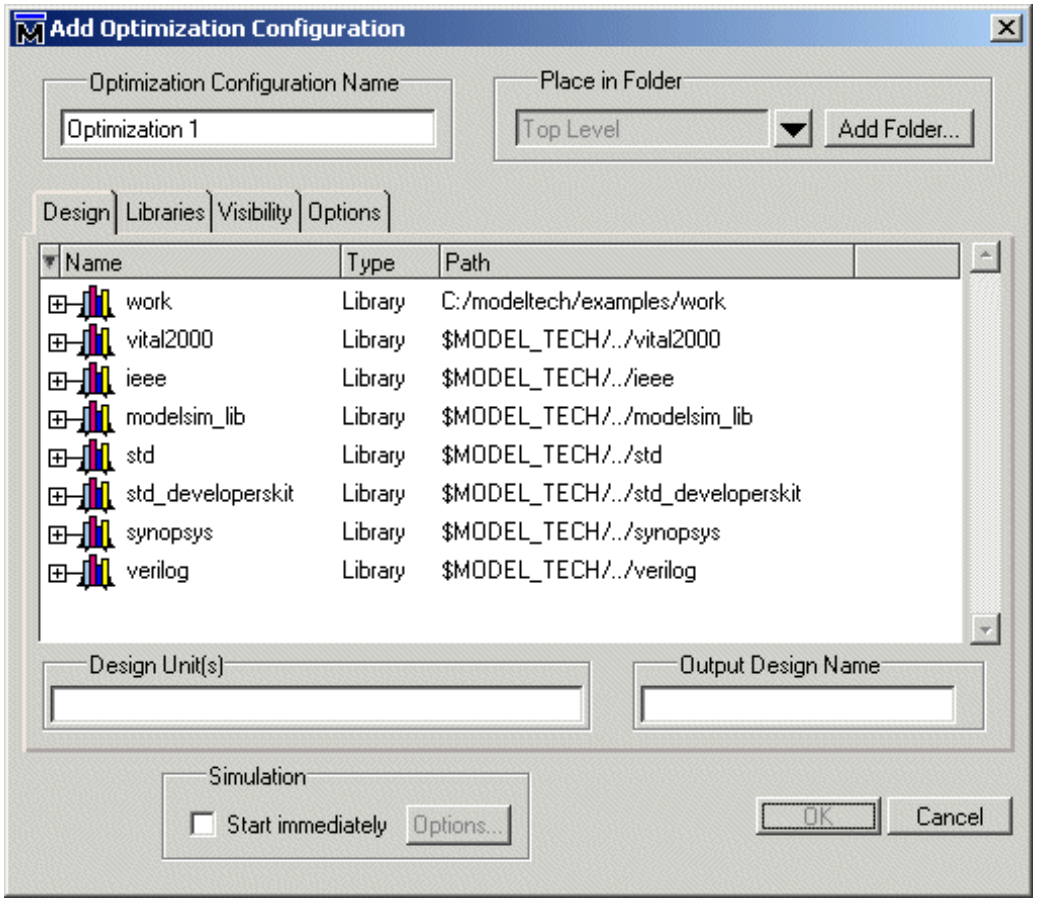


The Add file to Project dialog includes these options:

- **File Name**
  The name of the file to add. You can add multiple files at one time.
- **Add file as type**
  The type of the file. "Default" assigns type based on the file extension (e.g., *.v* is type Verilog).
- **Folder**
  The organization folder in which you want the file placed. You must first create folders in order to access them here. See "Organizing projects with folders" (UM-50) for details.
- **Reference from current location/Copy to project directory**
  Choose whether to reference the file from its current location or to copy it into the project directory.

Optimization Configuration dialog

Purpose	Menu command	Additional information
Add an optimization configuration to the open project	**File > Add to Project> Optimization Configuration**	"Optimization Configurations" (UM-49), "Design Optimization dialog" (GR-70)

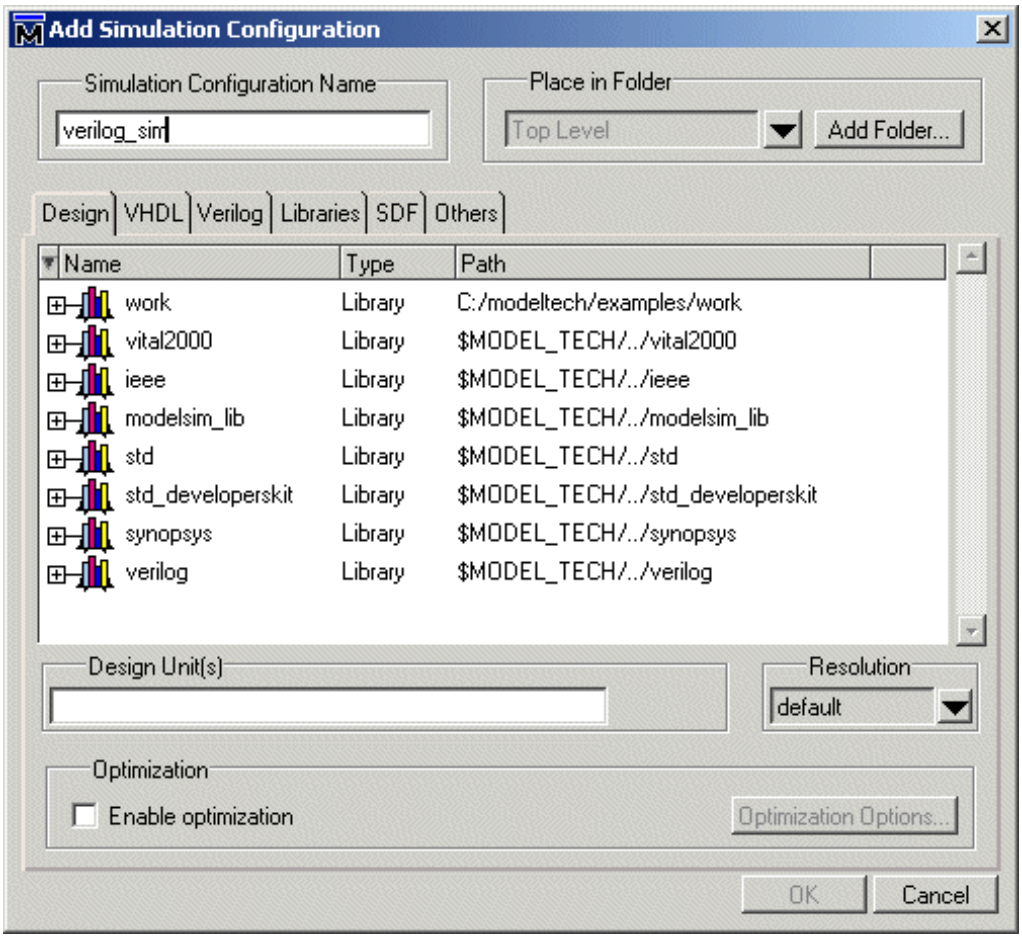


When adding an optimization configuration, you are presented with a modified version of the "Design Optimization dialog" (GR-70) that includes two additional options:

- **Optimization Configuration Name**
  The name you want for the Optimization Configuration.
- **Place in Folder**
  The organization folder in which you want the Optimization Configuration placed. Click Add Folder to create a new folder. See "Organizing projects with folders" (UM-50) for details.

Simulation Configuration dialog

Purpose	Menu command	Additional information
Add a simulation configuration to the open project	**File > Add to Project> Simulation Configuration**	"Creating a Simulation Configuration" (UM-48), "Start Simulation dialog" (GR-76)

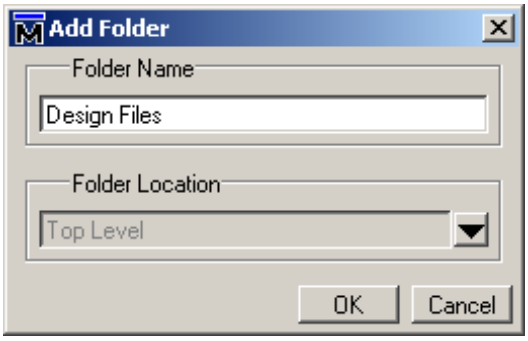


When adding a simulation configuration, you are presented with a modified version of the "Start Simulation dialog" (GR-76) that includes two additional options:

- **Simulation Configuration Name**
  The name you want for the Simulation Configuration.
- **Place in Folder**
  The organization folder in which you want the Simulation Configuration placed. Click Add Folder to create a new folder. See "Organizing projects with folders" (UM-50) for details.

Add Folder dialog

Purpose	Menu command	Additional information
Add a folder to the open project	**File > Add to Project> Folder**	"Organizing projects with folders" (UM-50)

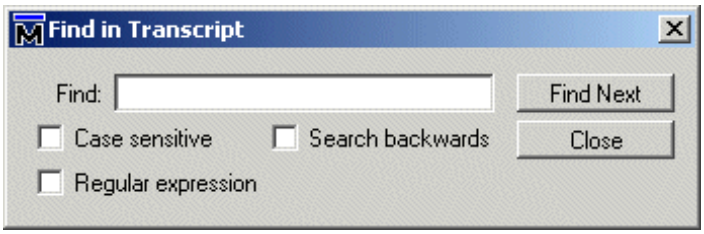


The Add Folder dialog includes these options:

- **Folder Name**
  The name you want for the new folder.
- **Folder Location**
  The organization folder in which you want the new folder placed. The first folder you create can be placed only in "Top Level."

Find in Transcript dialog

Purpose	Menu command	Additional information
Search for text in the current transcript	**Edit > Find**	NA



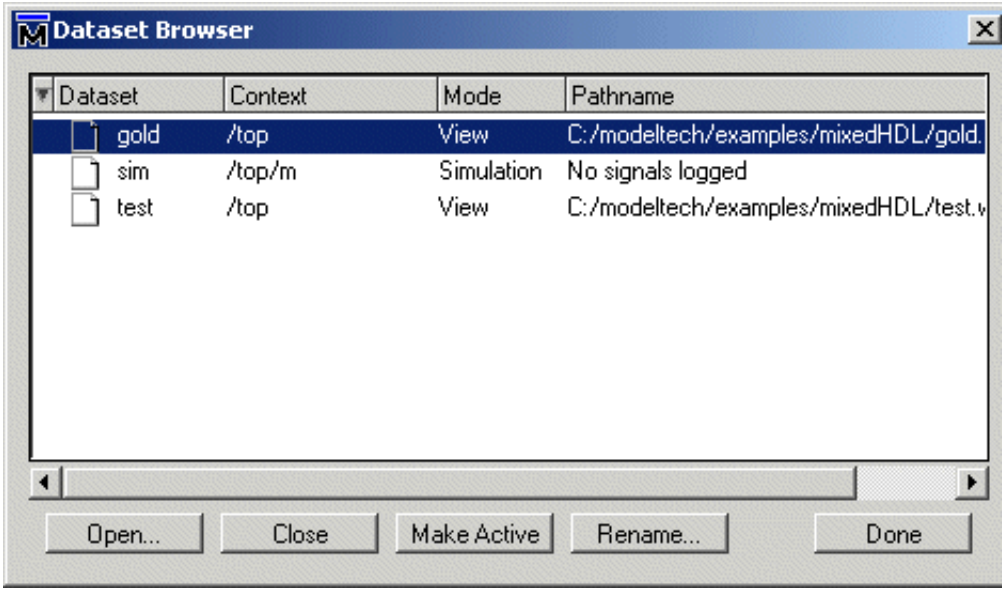
You must activate the Transcript pane by clicking in it before this dialog will be available.

The Find in Transcript dialog includes these options:

- **Find**
  The text string you want to search for in the transcript. Backslashes are used to escape special interpretation of basic regular expression characters. To search explicitly for a backslash character, it is necessary to escape the character. For example, to match \Arch Signal 1\, the pattern \\Arch... is required.
- **Case sensitive**
  If checked, the search will be case sensitive.
- **Regular expression**
  Check this box if you are using regular expression in the search string.
- **Search backwards**
  Search progresses backwards from the current cursor location in the transcript pane.

Dataset Browser dialog

Purpose	Menu command	Additional information
Manage datasets	**View > Datasets**	"Managing multiple datasets" (UM-229)



The Dataset Browser dialog includes these options.

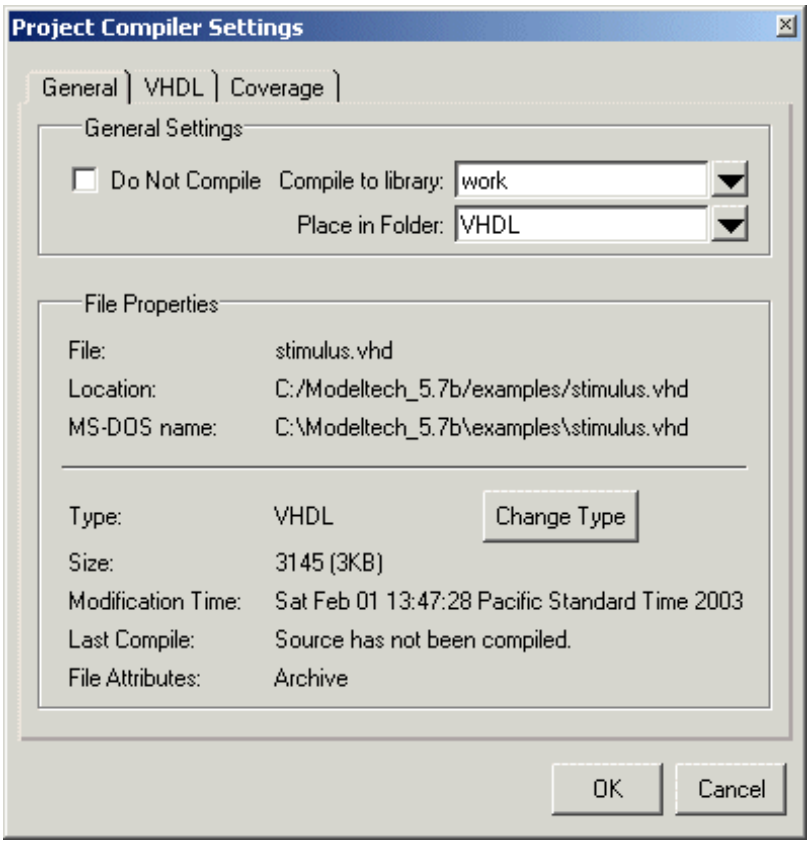
- **Open**
  Opens the Open Dataset dialog (see "Open File dialog" (GR-39)) so you can open additional datasets.
- **Close**
  Closes the selected dataset. This will also remove the dataset's Structure tab in the Main window workspace.
- **Make Active**
  Makes the selected dataset "active." You can also effect this change by double-clicking the dataset name. "Active" dataset means that if you type a region path as part of a command and omit the dataset prefix, the active dataset will be assumed. It is equivalent to typing `env <dataset>:` at the VSIM prompt. The active dataset is displayed at the bottom of the Main window.
- **Rename**
  Allows you to assign a new logical name to the selected dataset.

Project Compiler Settings

Purpose	Menu command	Additional information
Configure settings for project files	**View > Properties**	"Specifying file properties and project settings" (UM-52)

This dialog is accessible only if you have a project open. In addition to selecting **View > Properties**, you can open this dialog by right-clicking a file in the Project tab and selecting **Properties**, or by selecting **Compile > Compile Properties.** The tabs that are displayed will depend on the type of file(s) you select. When you select a SystemC file, only the General tab will be displayed.

General tab

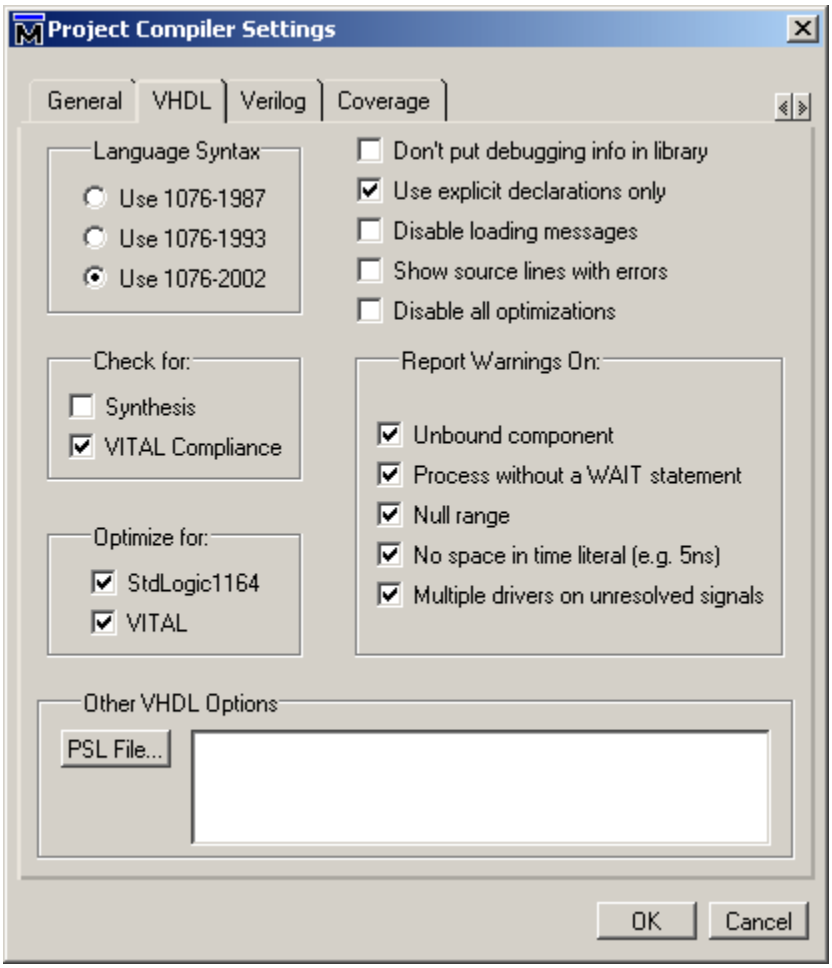


The General tab includes these options:

- **Do Not Compile**
  Determines whether the file is excluded from the compile.
- **Compile to library**
  Specifies to which library you want to compile the file; defaults to the working library.

- **Place in Folder**
  Specifies the folder in which to place the selected file(s). See "Organizing projects with folders" (UM-50) for details on folders.
- **File Properties**
  A variety of information about the selected file (e.g, type, size, path). Displays only if a single file is selected in the Project tab.
- **Change Type**
  Change the type that ModelSim assigns the file. ModelSim uses file extensions (e.g., .v or .vhd) to assign file types.

VHDL tab



- **Language Syntax**
  Specifies which version of the 1076 standard to use when compiling. The default for versions 5.8 and later is 2002. Changing the VHDL standard (UM-539) variable in the *modelsim.ini* file will make the setting permanent.
- **Don't put debugging info in library**
  Models compiled with this option do not use any of the ModelSim debugging features.

Consequently, your user will not be able to see into the model. This also means that you cannot set breakpoints or single step within this code. Don't compile with this option until you are done debugging. Same as the **-nodebug** argument to the **vcom** command (CR-311). See "Protecting source code using -nodebug" (UM-70) for more details. Edit the NoDebug (UM-527) variable in the *modelsim.ini* file to set a permanent default.

- **Use explicit declarations only**
  Used to ignore an error in packages supplied by some other EDA vendors; directs the compiler to resolve ambiguous function overloading in favor of the explicit function definition. Same as the **-explicit** argument to the **vcom** command (CR-311). Edit the Explicit (UM-527) variable in the *modelsim.ini* file to set a permanent default.

  Although it is not intuitively obvious, the = operator is overloaded in the **std_logic_1164** package. All enumeration data types in VHDL get an "implicit" definition for the = operator. So while there is no explicit = operator, there is an implicit one. This implicit declaration can be hidden by an explicit declaration of = in the same package (LRM Section 10.3). However, if another version of the = operator is declared in a different package than that containing the enumeration declaration, and both operators become visible through **use** clauses, neither can be used without explicit naming, for example:

  ```
  ARITHMETIC."="(left, right)
  ```

  This option allows the explicit = operator to hide the implicit one.

- **Disable loading messages**
  Disables loading messages in the Transcript pane. Same as the **-quiet** argument for the **vcom** command (CR-311). Edit the Quiet (UM-527) variable in the *modelsim.ini* file to set a permanent default.

- **Show source lines with errors**
  Causes the compiler to display the relevant lines of code in the transcript. Same as the **-source** argument to the **vcom** command (CR-311). Edit the Show_source (UM-528) variable in the *modelsim.ini* file to set a permanent default.

- **Disable all optimizations**
  Instructs the compiler to remove all optimizations. Same as the **-O0** argument to the **vcom** command (CR-311). Useful when "Measuring code coverage" (UM-333), where optimizations can skew results.

Check for:

- **Synthesis**
  Turns on limited synthesis-rule compliance checking. The checks apply only to signals used (read) by a process; also, the checks understand only combinational logic, not clocked logic. Edit the CheckSynthesis (UM-527) variable in the *modelsim.ini* file to set a permanent default.

- **VITAL Compliance**
  Toggle Vital compliance checking. Edit the NoVitalCheck (UM-527) variable in the *modelsim.ini* file to set a permanent default.

Report Warnings on:

- **Unbound component**
  Flags any component instantiation in the VHDL source code that has no matching entity in a library that is referenced in the source code, either directly or indirectly. Edit the Show_Warning1 (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Process without a WAIT statement**
  Flags any process that does not contain a wait statement or a sensitivity list. Edit the Show_Warning2 (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Null range**
  Flags any null range, such as 0 down to 4. Edit the Show_Warning3 (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **No space in time literal (e.g. 5ns)**
  Flags any time literal that is missing a space between the number and the time unit. Edit the Show_Warning4 (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Multiple drivers on unresolved signals**
  Flags any unresolved signals that have multiple drivers. Edit the Show_Warning5 (UM-528) variable in the *modelsim.ini* file to set a permanent default.

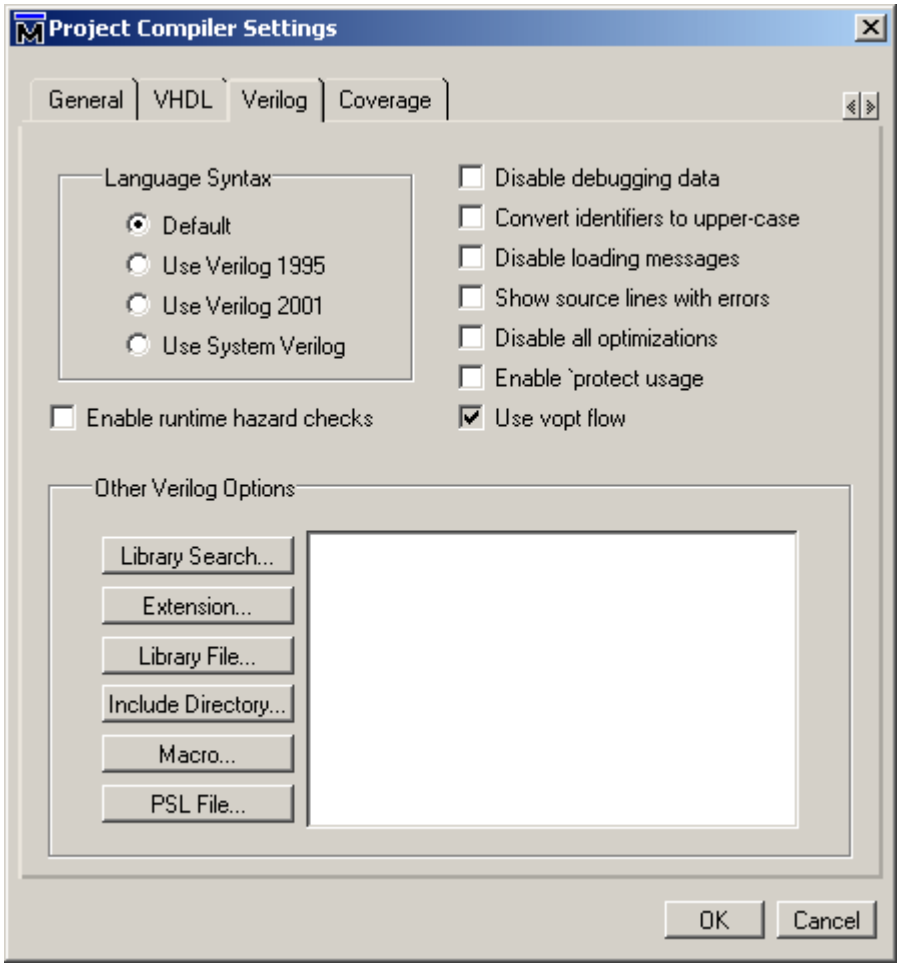
Optimize for:

- **StdLogic1164**
  Causes the compiler to perform special optimizations for speeding up simulation when the multi-value logic package std_logic_1164 is used. Unless you have modified the std_logic_1164 package, this option should always be checked. Edit the Optimize_1164 (UM-527) variable in the *modelsim.ini* file to set a permanent default.
- **VITAL**
  Toggle acceleration of the VITAL packages. Edit the NoVital (UM-527) variable in the *modelsim.ini* file to set a permanent default.

Other VHDL options:

- **PSL File**
  Click this button to add a PSL file. See *Chapter 14 - PSL Assertions* for additional information about using assertions.
- Enter any other valid **vcom** arguments. See the **vcom** command (CR-311) in the *ModelSim Command Reference* for a complete list.

Verilog tab



The Verilog tab includes the following options:

- **Language Syntax**
  Specify which version of the standard should be used to compile the selected file(s). The default is Verilog 2001. Selecting **Use Verilog 1995** simply disables 2001 keywords. Similarly, selecting **Use SystemVerilog** simply enables SystemVerilog keywords.
- **Enable runtime hazard checks**
  Enables the run-time hazard checking code. Same as the **-hazards** argument to the **vlog** command (CR-358). Edit the Hazard (UM-525) variable in the *modelsim.ini* file to set a permanent default.
- **Disable debugging data**
  Models compiled with this option do not use any of the ModelSim debugging features. Consequently, your user will not be able to see into the model. This also means that you cannot set breakpoints or single step within this code. Don't compile with this option until you are done debugging. Same as the **-nodebug** argument for the **vlog** command (CR-358). See "Protecting source code using -nodebug" (UM-70) for more details. Edit the NoDebug (UM-527) variable in the *modelsim.ini* file to set a permanent default.

- **Convert identifiers to upper-case**
  Converts regular Verilog identifiers to uppercase. Allows case insensitivity for module names. Same as the **-u** argument to the **vlog** command (CR-358). Edit the UpCase (UM-526) variable in the *modelsim.ini* file to set a permanent default.
- **Disable loading messages**
  Disables loading messages in the Transcript pane. Same as the **-quiet** argument for the **vlog** command (CR-358). Edit the Quiet (UM-527) variable in the *modelsim.ini* file to set a permanent default.
- **Show source lines with errors**
  Causes the compiler to display the relevant lines of code in the transcript. Same as the **-source** argument to the **vlog** command (CR-358). Edit the Show_source (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Disable all optimizations**
  Instructs the compiler to remove all optimizations. Same as the **-O0** argument to the **vlog** command (CR-358). Useful when running "Measuring code coverage" (UM-333), where optimizations can skew results.
- **Enable `protect usage**
  Enables encryption of regions of your Verilog source code. See "ModelSim compiler directives" (UM-155) for more details. Same as the **+protect** argument for the **vlog** command (CR-358). Edit the Protect (UM-526) variable in the *modelsim.ini* file to set a permanent default.
- **Use vopt flow**
  Instructs ModelSim that you will be optimizing the design (see "Optimizing Verilog designs" (UM-124) for more information). This prevents the compiler from producing code. Same as the **-vopt** argument to the **vlog** command (CR-358).

Other Verilog Options:

Specify any valid **vlog** command (CR-358) arguments.

- **Library Search**
  Specifies the Verilog source library directory to search for undefined modules. Same as the **-y <library_directory>** argument for the **vlog** command (CR-358).
- **Extension**
  Specifies the suffix of files in the library directory. Multiple suffixes can be used. Same as the **+libext+<suffix>** argument for the **vlog** command (CR-358).
- **Library File**
  Specifies the Verilog source library file to search for undefined modules. Same as the **-v <library_file>** argument for the **vlog** command (CR-358).
- **Include Directory**
  Specifies a directory for files included with the **‘include filename** compiler directive. Same as the **+incdir+<directory>** argument for the **vlog** command (CR-358).
- **Macro**
  Defines a macro equivalent to one created with the **‘define macro_name macro_text** compiler directive. Same as the **+define+<macro_name> [=<macro_text>]** argument for the **vlog** command (CR-358).

- PSL File
  Add a PSL file. See *Chapter 14 - PSL Assertions* for additional information about using assertions.

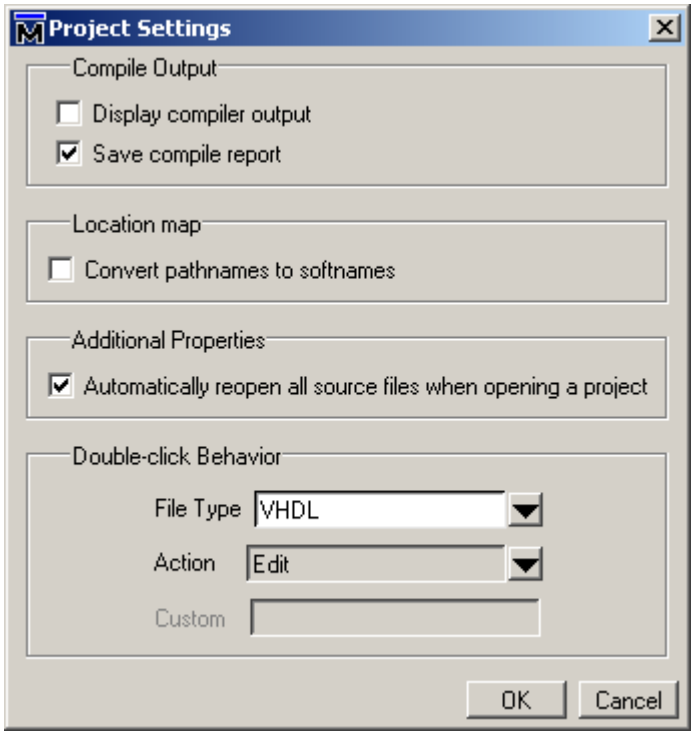
Note: When you specify Other Verilog Options, they are saved into a file called *vlog.opt*. If you do this while a project is open, an OptionFile entry is written into your project file. If you do this when a project is not open, an OptionFile entry is written into the *modelsim.ini* file that you are currently using.

Coverage tab

The definitions for the options on the Coverage tab can be found in "Enabling code coverage" (UM-337).

Project Settings dialog

Purpose	Menu command	Additional information
Configure default project properties	**View > Project Settings**	"Specifying file properties and project settings" (UM-52)



The Project Settings dialog includes these options:

- **Display compiler output**
  Prints verbose compile output to the Transcript. By default verbose output is produced in the Compile Report only.
- **Save compile report**
  Saves verbose compile output to disk. You can access the report by right-clicking a file and selecting **Compile > Compile Report**.
- **Location map**
  Specifies whether physical paths for the project files should be saved as soft paths if they are present in the location map. See "Referencing source files with location maps" (UM-67) for more details on using location maps.
- **Additional Properties**
  Specifies whether all previously opened project source files will be reopened when a project is reopened. Default: "Automatically reopen all source files..." is enabled.

- **Double-click Behavior**
  Specifies the action to take when you double-click a type of file. If you select Custom, you can specify a Tcl command in the text box below the file type.

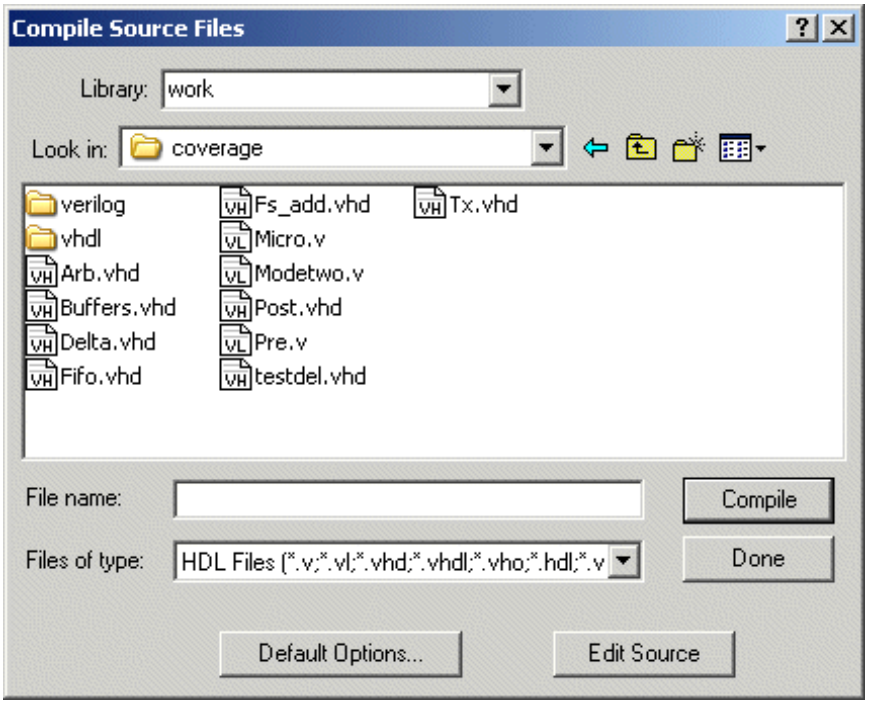
  You can use *%f* for filename substitution. For example, if you want double click on a Tcl file to open the file with Notepad, you would insert the following in the text box:

  ```
  notepad %f
  ```

  ModelSim will substitute the *%f* with the filename that was clicked on, then execute the string.

Compile Source Files dialog

Purpose	Menu command	Additional information
Compile Verilog or VHDL source files	**Compile > Compile** (disabled when a project is open)	"Compiling Verilog files" (UM-114), "Compiling VHDL files" (UM-73), "Compiling SystemC files" (UM-164)



The Compile Source Files dialog includes these options:

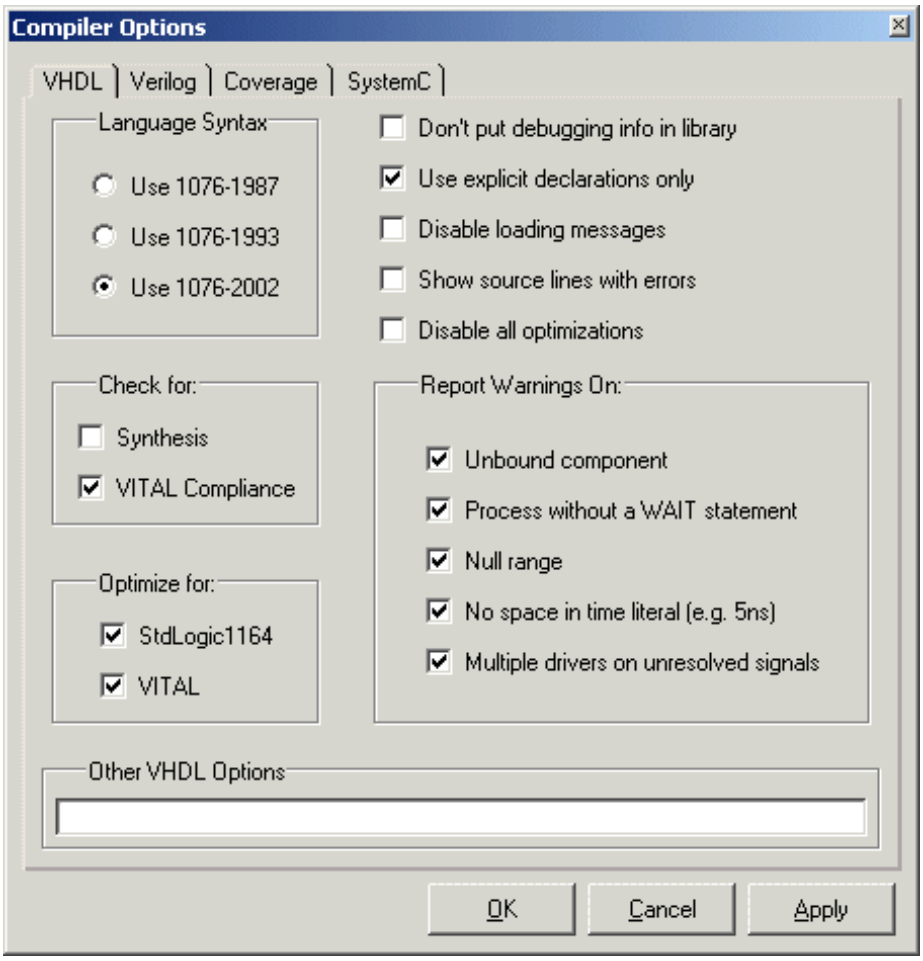
- **Library**
  The library into which you want the source code compiled.
- **Look in**
  The directory you want to look in for source files.
- **File name**
  The name of the file(s) you want to compile. Specify the file name and then click the Compile button. Alternatively, double-click a file in the list to compile it.
- **File of type**
  Filter the list to show only files of a certain type (e.g., HDL files, Verilog files, etc.).
- **Default Options**
  Configure compiler options for the Verilog, VHDL, and SystemC compilers and Code Coverage. See "Compiler Options dialog" (GR-60) for details.
- **Edit Source**
  View or edit the selected file in the Source window.

Compiler Options dialog

Purpose	Menu command	Additional information
Configure compiler options	**Compile** > **Compile Options (**disabled when a project is open)	**vlog** command (CR-358), **vcom** command (CR-311), **sccom** command (CR-254)

Important: Note that changes made in the **Compiler Options** dialog become the default for all future simulations.

VHDL tab



The VHDL tab includes the following options:

- **Language Syntax**
  Specifies which version of the 1076 standard to use when compiling. The default for versions 5.8 and later is 2002. Changing the VHDL standard (UM-539) variable in the *modelsim.ini* file will make the setting permanent.

- **Don’t put debugging info in library**
  Models compiled with this option do not use any of the ModelSim debugging features. Consequently, your user will not be able to see into the model. This also means that you cannot set breakpoints or single step within this code. Don’t compile with this option until you are done debugging. Same as the **-nodebug** argument to the **vcom** command (CR-311). See "Protecting source code using -nodebug" (UM-70) for more details. Edit the NoDebug (UM-527) variable in the *modelsim.ini* file to set a permanent default.
- **Use explicit declarations only**
  Used to ignore an error in packages supplied by some other EDA vendors; directs the compiler to resolve ambiguous function overloading in favor of the explicit function definition. Same as the **-explicit** argument to the **vcom** command (CR-311). Edit the Explicit (UM-527) variable in the *modelsim.ini* file to set a permanent default.

  Although it is not intuitively obvious, the = operator is overloaded in the **std_logic_1164** package. All enumeration data types in VHDL get an “implicit” definition for the = operator. So while there is no explicit = operator, there is an implicit one. This implicit declaration can be hidden by an explicit declaration of = in the same package (LRM Section 10.3). However, if another version of the = operator is declared in a different package than that containing the enumeration declaration, and both operators become visible through **use** clauses, neither can be used without explicit naming, for example:

  ```
  ARITHMETIC.”=”(left, right)
  ```

  This option allows the explicit = operator to hide the implicit one.
- **Disable loading messages**
  Disables loading messages in the Transcript pane. Same as the **-quiet** argument for the **vcom** command (CR-311). Edit the Quiet (UM-527) variable in the *modelsim.ini* file to set a permanent default.
- **Show source lines with errors**
  Causes the compiler to display the relevant lines of code in the transcript. Same as the **-source** argument to the **vcom** command (CR-311). Edit the Show_source (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Disable all optimizations**
  Instructs the compiler to remove all optimizations. Same as the **-O0** argument to the **vcom** command (CR-311). Useful when "Measuring code coverage" (UM-333), where optimizations can skew results.

Check for:

- **Synthesis**
  Turns on limited synthesis-rule compliance checking. The checks apply only to signals used (read) by a process; also, the checks understand only combinational logic, not clocked logic. Edit the CheckSynthesis (UM-527) variable in the *modelsim.ini* file to set a permanent default.
- **VITAL Compliance**
  Toggle Vital compliance checking. Edit the NoVitalCheck (UM-527) variable in the *modelsim.ini* file to set a permanent default.

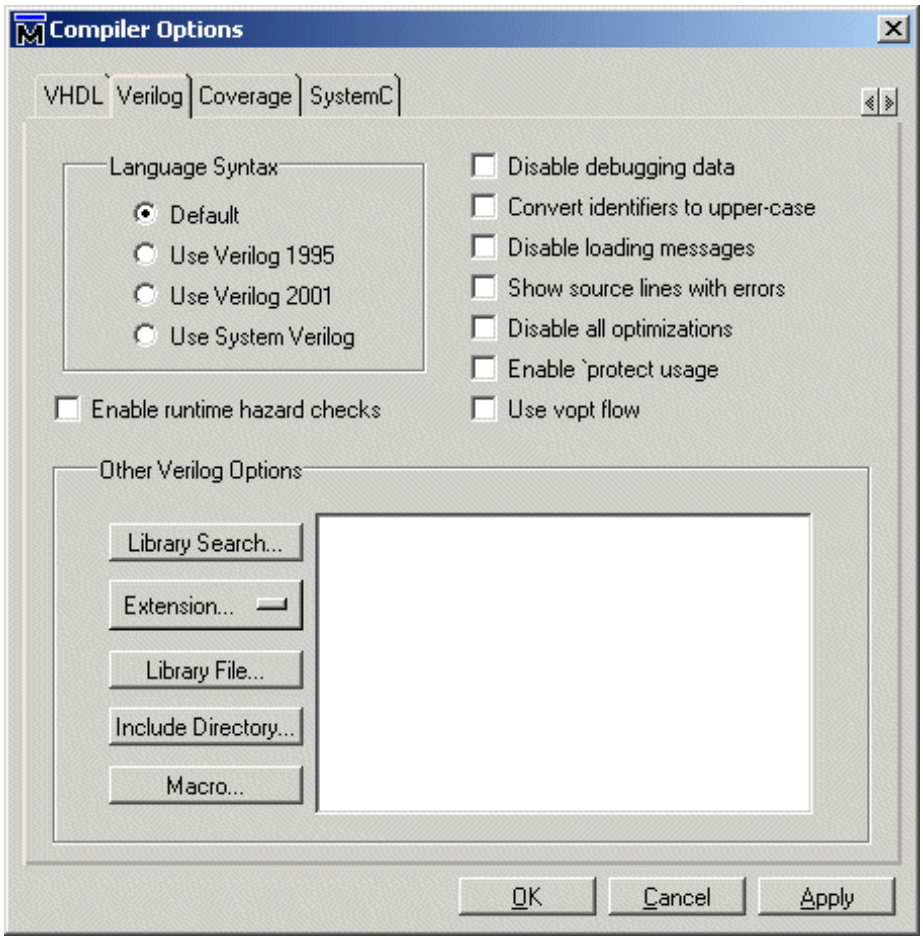
Report Warnings on:

- **Unbound component**
  Flags any component instantiation in the VHDL source code that has no matching entity in a library that is referenced in the source code, either directly or indirectly. Edit the Show_Warning1 (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Process without a WAIT statement**
  Flags any process that does not contain a wait statement or a sensitivity list. Edit the Show_Warning2 (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Null range**
  Flags any null range, such as 0 down to 4. Edit the Show_Warning3 (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **No space in time literal (e.g. 5ns)**
  Flags any time literal that is missing a space between the number and the time unit. Edit the Show_Warning4 (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Multiple drivers on unresolved signals**
  Flags any unresolved signals that have multiple drivers. Edit the Show_Warning5 (UM-528) variable in the *modelsim.ini* file to set a permanent default.

Optimize for:

- **StdLogic1164**
  Causes the compiler to perform special optimizations for speeding up simulation when the multi-value logic package std_logic_1164 is used. Unless you have modified the std_logic_1164 package, this option should always be checked. Edit the Optimize_1164 (UM-527) variable in the *modelsim.ini* file to set a permanent default.
- **VITAL**
  Toggle acceleration of the VITAL packages. Edit the NoVital (UM-527) variable in the *modelsim.ini* file to set a permanent default.
- **Other VHDL options**
  Enter any other valid **vcom** arguments. See the **vcom** command (CR-311) in the *ModelSim Command Reference* for a complete list.

Verilog tab



The Verilog tab includes the following options:

- **Language Syntax**
  Specify which version of the standard should be used to compile the selected file(s). The default is Verilog 2001. Selecting **Use Verilog 1995** simply disables 2001 keywords. Similarly, selecting **Use SystemVerilog** simply enables SystemVerilog keywords.
- **Enable runtime hazard checks**
  Enables the run-time hazard checking code. Same as the **-hazards** argument to the **vlog** command (CR-358). Edit the Hazard (UM-525) variable in the *modelsim.ini* file to set a permanent default.
- **Disable debugging data**
  Models compiled with this option do not use any of the ModelSim debugging features. Consequently, your user will not be able to see into the model. This also means that you cannot set breakpoints or single step within this code. Don't compile with this option until you are done debugging. Same as the **-nodebug** argument for the **vlog** command (CR-358). See "Protecting source code using -nodebug" (UM-70) for more details. Edit the NoDebug (UM-527) variable in the *modelsim.ini* file to set a permanent default.

- **Convert identifiers to upper-case**
  Converts regular Verilog identifiers to uppercase. Allows case insensitivity for module names. Same as the **-u** argument to the **vlog** command (CR-358). Edit the UpCase (UM-526) variable in the *modelsim.ini* file to set a permanent default.
- **Disable loading messages**
  Disables loading messages in the Transcript pane. Same as the **-quiet** argument for the **vlog** command (CR-358). Edit the Quiet (UM-527) variable in the *modelsim.ini* file to set a permanent default.
- **Show source lines with errors**
  Causes the compiler to display the relevant lines of code in the transcript. Same as the **-source** argument to the **vlog** command (CR-358). Edit the Show_source (UM-528) variable in the *modelsim.ini* file to set a permanent default.
- **Disable all optimizations**
  Instructs the compiler to remove all optimizations. Same as the **-O0** argument to the **vlog** command (CR-358). Useful when running "Measuring code coverage" (UM-333), where optimizations can skew results.
- **Enable `protect usage**
  Enables encryption of regions of your Verilog source code. See "ModelSim compiler directives" (UM-155) for more details. Same as the **+protect** argument for the **vlog** command (CR-358). Edit the Protect (UM-526) variable in the *modelsim.ini* file to set a permanent default.
- **Use vopt flow**
  Instructs ModelSim that you will be optimizing the design (see "Optimizing Verilog designs" (UM-124) for more information). This prevents the compiler from producing code. Same as the **-vopt** argument to the **vlog** command (CR-358).

Other Verilog Options:

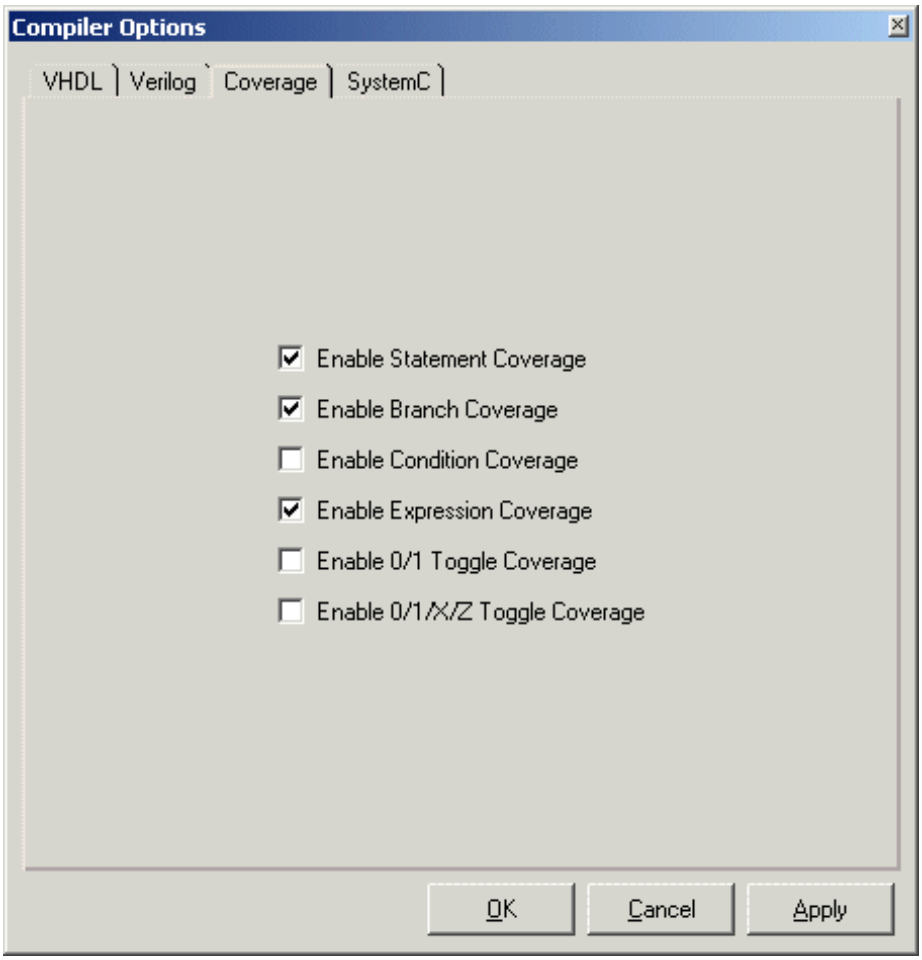
Specify any valid **vlog** command (CR-358) arguments.

- **Library Search**
  Specifies the Verilog source library directory to search for undefined modules. Same as the **-y <library_directory>** argument for the **vlog** command (CR-358).
- **Extension**
  Specifies the suffix of files in the library directory. Multiple suffixes can be used. Same as the **+libext+<suffix>** argument for the **vlog** command (CR-358).
- **Library File**
  Specifies the Verilog source library file to search for undefined modules. Same as the **-v <library_file>** argument for the **vlog** command (CR-358).
- **Include Directory**
  Specifies a directory for files included with the **‘include filename** compiler directive. Same as the **+incdir+<directory>** argument for the **vlog** command (CR-358).
- **Macro**
  Defines a macro equivalent to one created with the **‘define macro_name macro_text**

compiler directive. Same as the **+define+<macro_name> [=<macro_text>]** argument for the **vlog** command (CR-358).

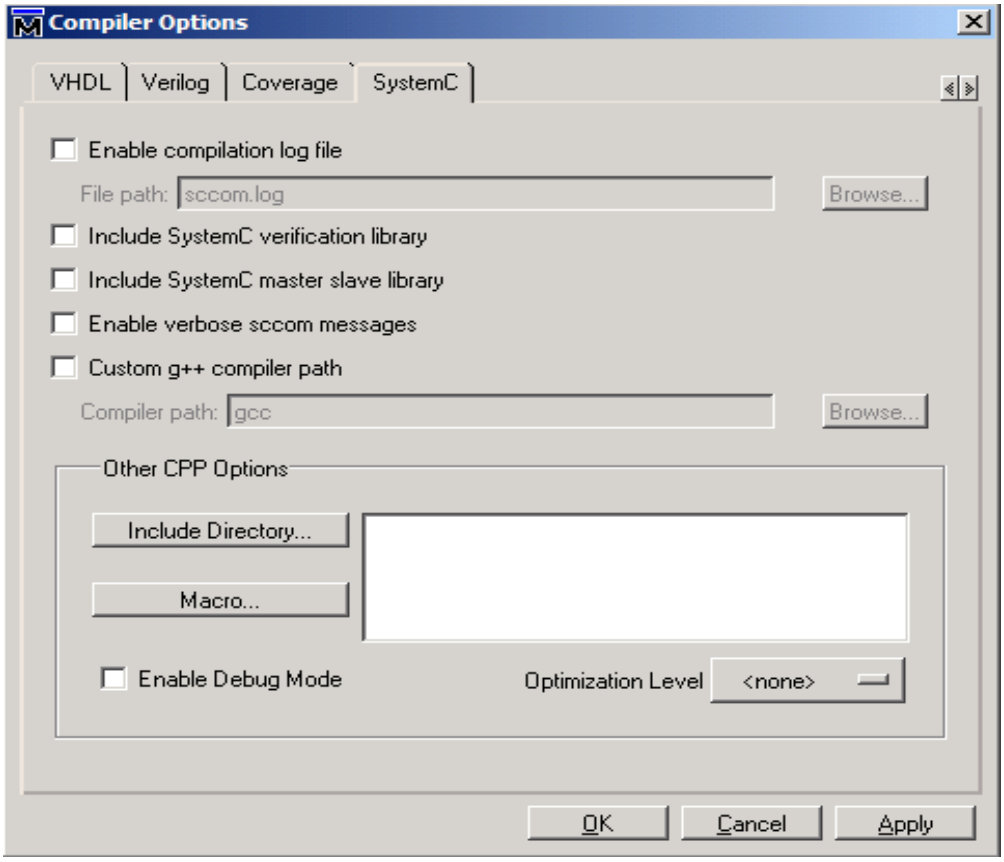
Note: When you specify Other Verilog Options, they are saved into a file called *vlog.opt*. If you do this while a project is open, an OptionFile entry is written into your project file. If you do this when a project is not open, an OptionFile entry is written into the *modelsim.ini* file that you are currently using.

Coverage tab



The Coverage tab lets you select which types of Code Coverage statistics will be collected during the simulation. See "Enabling code coverage" (UM-337) for details.

SystemC tab



The SystemC tab includes the following options:

- **Enable compilation log file**
  Writes the compilation output to a file name, specified in the **File path** field. Same as the **-log** argument to the **sccom** command (CR-254).
- **Include SystemC verification library**
  Includes the SystemC verification library. Same as the **-scv** argument to the **sccom** command (CR-254).
- **Include SystemC master slave library**
  Includes the SystemC master slave library. Same as the **-scms** argument to the **sccom** command (CR-254).
- **Enable verbose sccom messages**
  Echoes subprocess invocations with command arguments. Same as the **-verbose** argument to the **sccom** command (CR-254).
- **Custom g++ compiler path**
  Enables you to specify a path for your custom g++ installation.

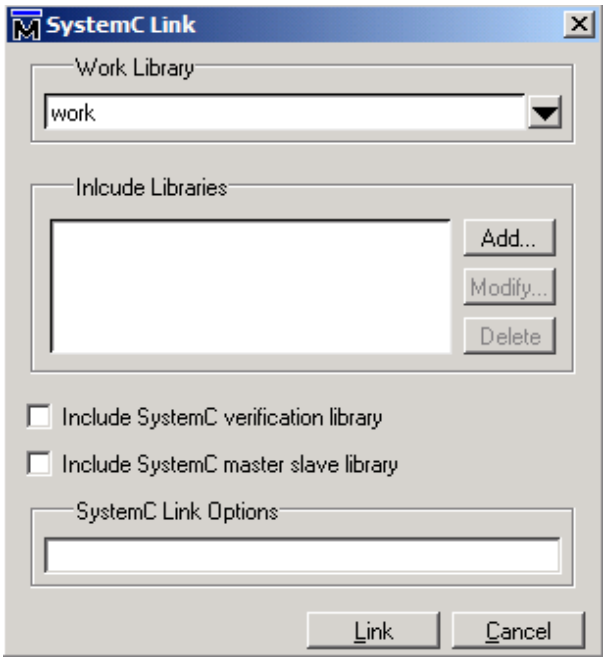
Other CPP Options

Specify any valid g++/aCC compiler options. All options are accepted, with the exception of the **-o** and **-c** options.

- **Include Directory**
  Includes a directory that contains source files. Same as the **-I** argument to g++/aCC.
- **Macro**
  Defines a macro. Same as the **-D** argument to g++/aCC.
- **Enable Debug Mode**
  Compiles SystemC code with debugging information. By default SystemC code is compiled without debugging information. Same as the **-g** argument to g++/aCC.
- **Optimization level**
  Specifies an optimization value you wish to use. By default, no optimization is performed. Same as the **-O#** argument to g++/aCC.

SystemC Link dialog

Purpose	Menu command	Additional information
Build a shared library (.so) in the current work library	**Compile > SystemC Link**	"Linking the compiled source" (UM-172)

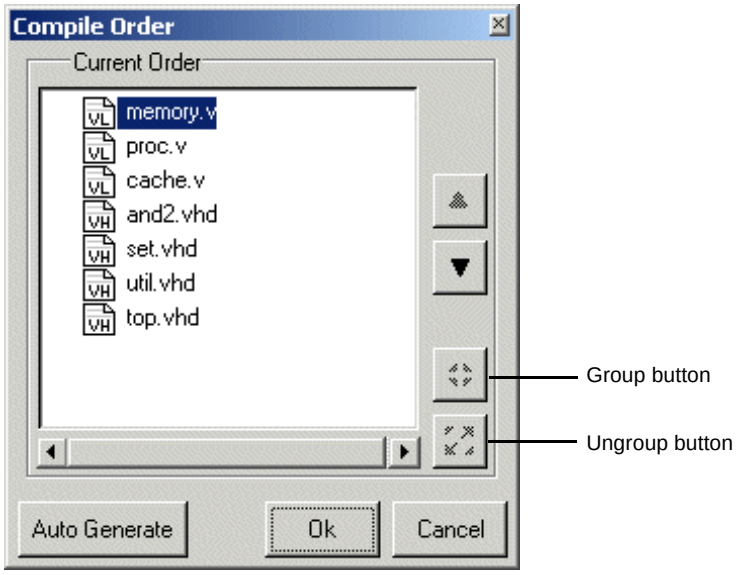


The SystemC Link dialog includes these options:

- **Work Library**
  Sets the work library for the **sccom -link** command. Same as the **-work** argument to the **sccom** command (CR-254).
- **Include Libraries**
  Includes any specified library. You can Add (browse for) a library, Modify the selection of a library, or Delete the library. Same as the **-lib** argument to the **sccom** command (CR-254).
- **Include SystemC verification library**
  Includes the SystemC verification library. Same as the **-scv** argument to the **sccom** command (CR-254).
- **Include SystemC master slave library**
  Includes the SystemC master slave library. Same as the **-scms** argument to the **sccom** command (CR-254).
- **SystemC Link Options**
  Specify any valid g++/aCC linking options. All options are accepted.

Compile Order dialog

Purpose	Menu command	Additional information
Set the order in which files in a project are compiled	**Compile** > **Compile Order** (disabled unless project is open)	"Changing compile order" (UM-46)

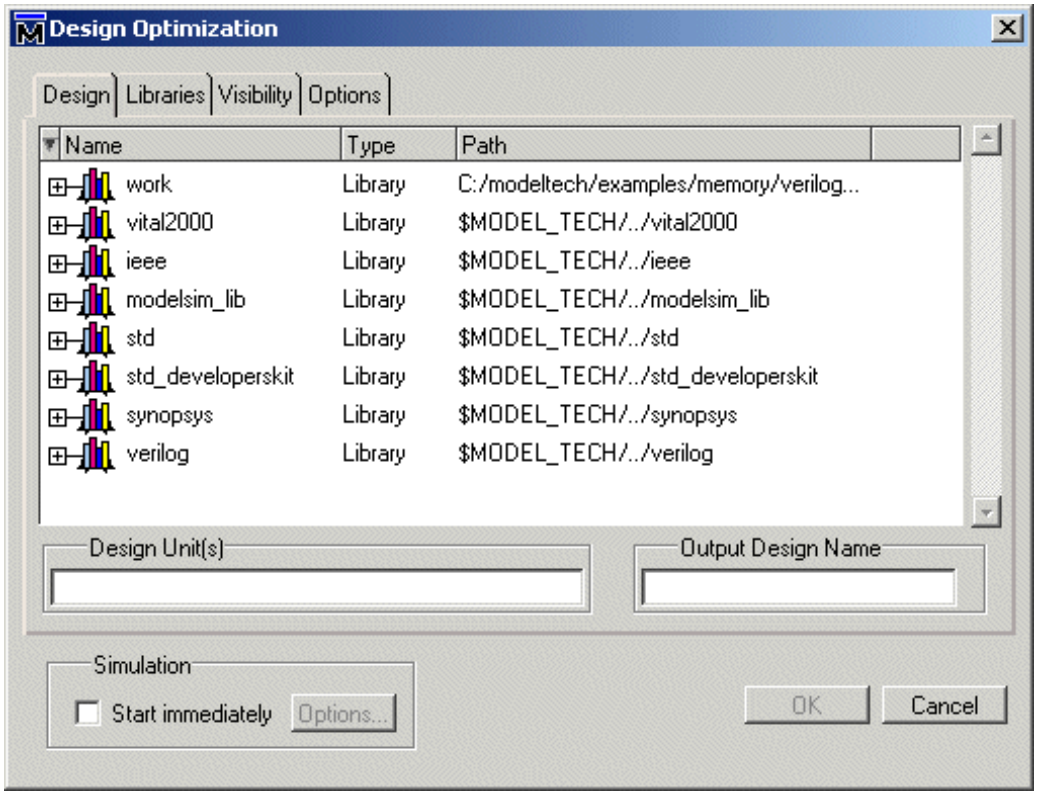


- **Up arrow**
  Move selected file(s) up in compile order.
- **Down arrow**
  Move selected file(s) down in order.
- **Group button**
  Groups several files together so they are sent to the compiler at the same time. See "Grouping files" (UM-47) for further details.
- **Ungroup button**
  Ungroups a previously created group.
- **Auto Generate**
  Determines the correct compile order by making multiple passes over the files. See "Auto-generating compile order" (UM-46) for further details.

Design Optimization dialog

Purpose	Menu command	Additional information
Optimize a design	**Simulate** > **Design Optimization**	"Optimizing Verilog designs" (UM-124), **vopt** command (CR-371)

Design tab



The Design tab includes these options:

- **Design Unit(s)**
  Specifies the top-level design unit to optimize. You can specify names one of two ways:

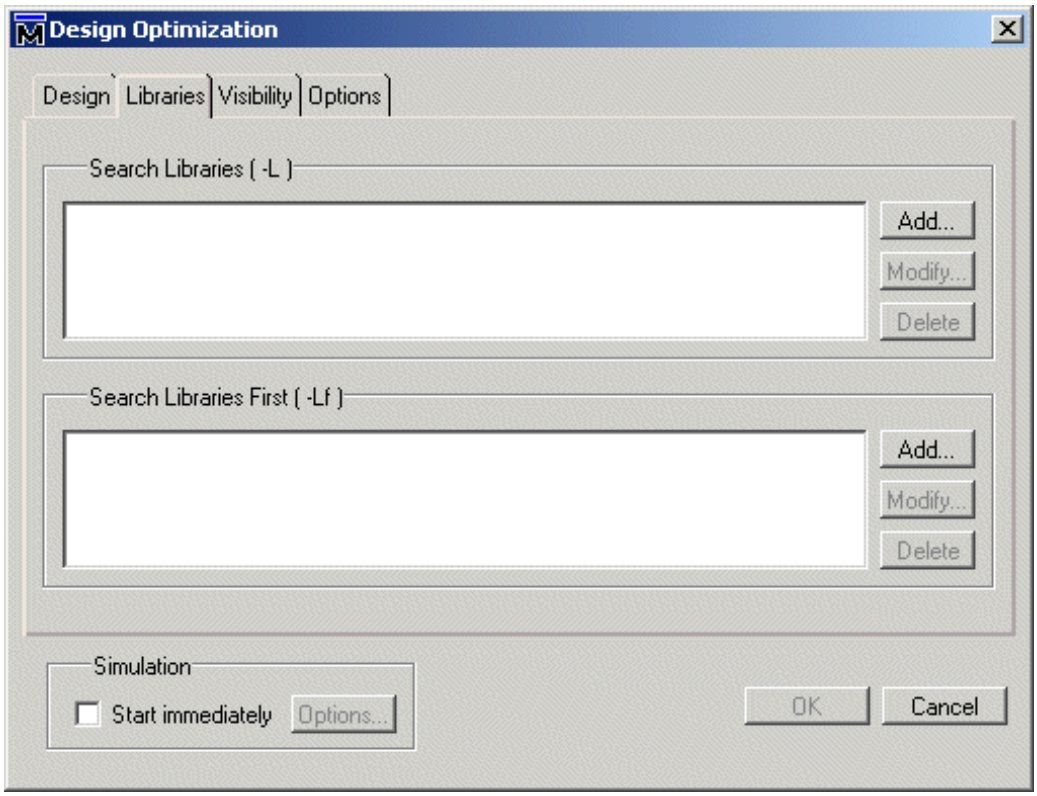
  - Type a design unit name (configuration, module, or entity) into the field, separate additional names with a space. Specify library/design units with the following syntax:

    ```
    [<library_name>.]<design_unit>
    ```

  - Select a design unit from the list. You can select multiple top level design units from the list by using the control key when you click.

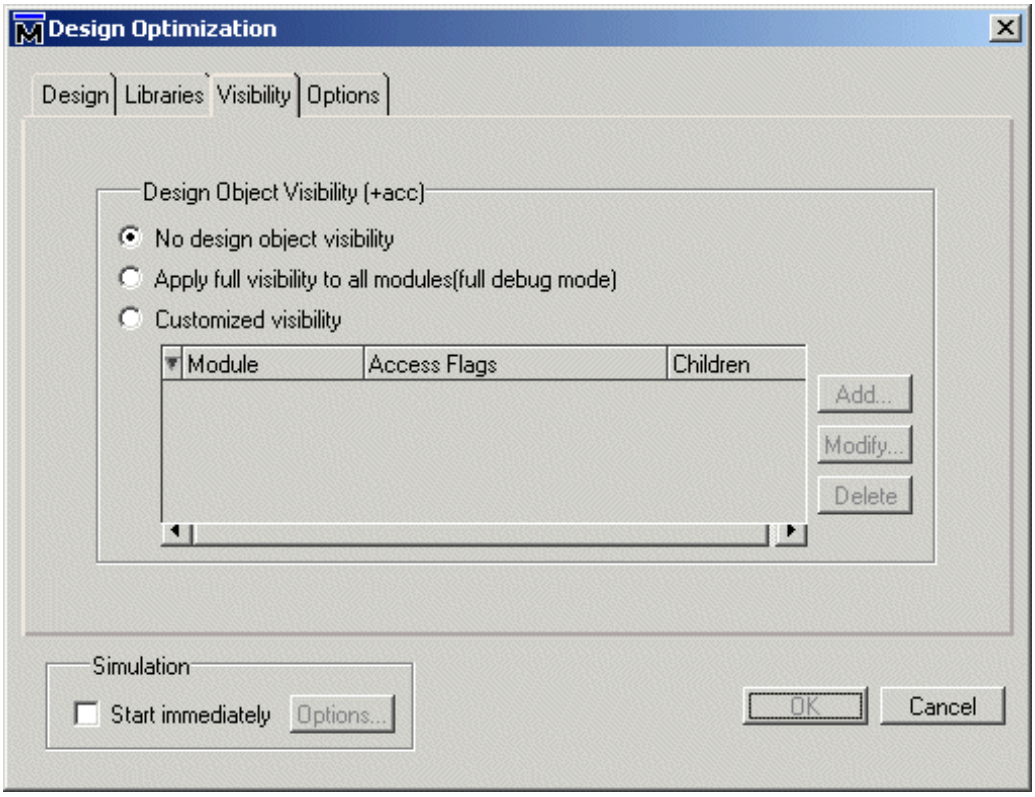
- **Output Design Name** (vopt -o <design_name>)
  Specify a name for the optimized design. This is a required option. See "Naming the optimized design" (UM-125) for more information.
- **Simulation Start Immediately**
  Automatically loads the design after optimization is complete. Click the Options button to specify Simulation options (see "Start Simulation dialog" (GR-76) for a description of the options).

Libraries tab



Use the Libraries tab to override any library search options you specified when you compiled the design. See the "Libraries tab" (GR-80) for further details.

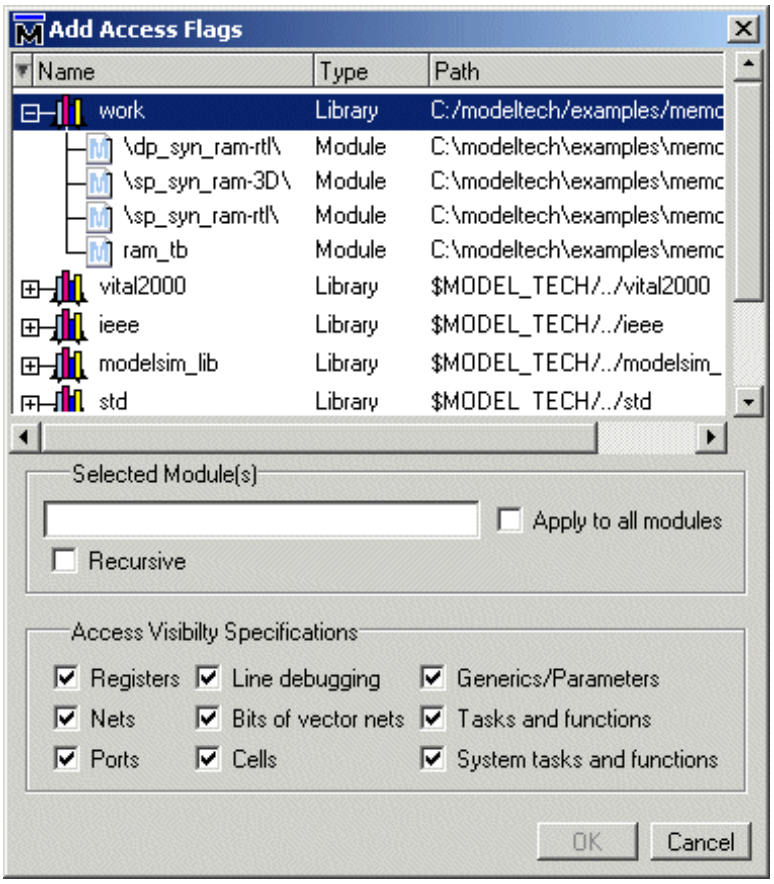
Visibility tab



Use the Visibility tab to selectively enable access to parts of your design. See "Enabling design object visibility with the +acc option" (UM-126) for additional information. The tab includes these options:

- **No design object visibility**
  vopt applies all possible optimizations with no concern for debugging visibility. Many of the nets, ports, and registers are unavailable by name in user interface commands and in the various graphic interface windows. In addition, many of these objects do not have PLI Access handles, potentially affecting the operation of PLI applications.
- **Apply full visibility...**(+acc)
  vopt maintains full access to all design objects. This may substantially reduce simulator performance.

- **Customized visibility**
  Click the **Add** button to open the Add Access Flags dialog.

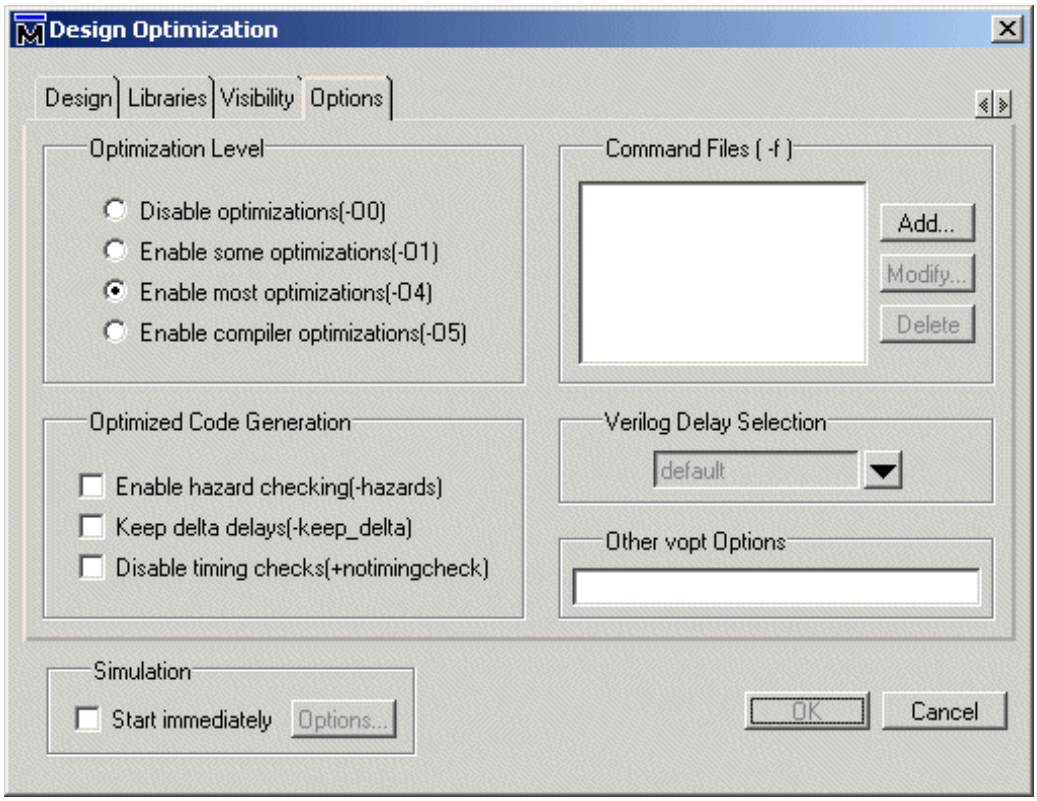


- **Selected Module(s)**
  Specify one or more module names for which you want to add access flags. You can type the names or use the library browser to select modules with the mouse.
- **Recursive**
  Applies flags recursively into sub-regions of the specified module(s).
- **Apply to all modules**
  Applies flags to all modules in the design.
- **Access Visibility Specifications**
  Specify to which design objects you need access. Options include:

Registers (+acc=r)	Enable access to registers (including memories, integer, time, and real types).
Nets (+acc=n)	Enable access to nets.
Ports (+acc=p)	Enable access to ports.

Line debugging (+acc=l)	Enable line number directives and process names for line debugging, profiling, and code coverage.
Bits of vector nets (+acc=b)	Enable access to individual bits of vector nets.
Cells (+acc=c)	Enable access to library cells.
Generics/Parameters (+acc=g)	Enable access to generics and parameters.
Tasks and functions (+acc=t)	Enable access to tasks and functions.
System tasks and functions (+acc=s)	Enable access to system tasks

Options tab



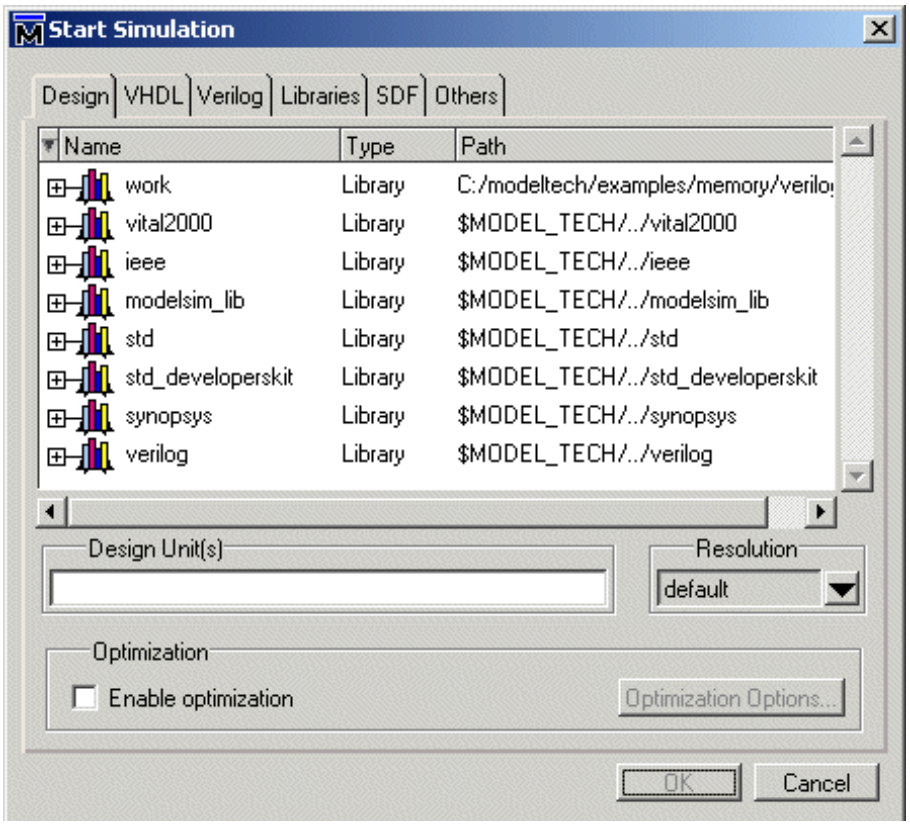
The Options tab includes these options:

- **Optimization Level** (-O)
  Specify the optimization level for the design. This option is ignored for VHDL and SystemC designs.
- **Command Files** (-f)
  Click Add to include one or more command files. A command file is a text file that includes additional command arguments.
- **Enable Hazard Checking** (-hazards)
  Enables hazard checking in Verilog modules. This overrides settings from your Verilog compiler invocation.
- **Keep delta delays** (-keep_delta)
  Disables optimizations that remove delta delays. This overrides settings from your Verilog compiler invocation.
- **Disable Timing Checks in Specify Blocks** (+notimingchecks)
  Disables the timing check system tasks ($setup, $hold,...) in specify blocks. This overrides settings from your Verilog compiler invocation.
- **Verilog Delay Selection** (+mindelays | +typdelays | +maxdelays)
  Use the drop-down menu to select timing for min:typ:max expressions.
- **Other Vopt Options**
  Specify any other **vopt** command (CR-371) arguments.

Start Simulation dialog

Purpose	Menu command	Additional information
Simulate a compiled design unit	**Simulate** > **Start Simulation**	"Verilog simulation" (UM-111), "VHDL simulation" (UM-71), "SystemC simulation" (UM-159), "Mixed-language simulation" (UM-187), **vsim** command (CR-373)

Design tab



The Design tab includes these options:

- **Design Unit(s)**
  Specifies the design unit(s) to simulate. You can specify the top-level design unit in one of two ways:

  - Type a design unit name (configuration, module, entity, optimized design) into the field, separating additional names with a space. Specify library/design units with the following syntax:

    ```
    [<library_name>.]<design_unit>
    ```

  - Select a design unit from the list. You can select multiple design units from the list by using the control key when you click.

- **Resolution**
  (-t [<multiplier>]<time_unit>)
  The drop-down menu sets the simulator time units.

  Simulator time units can be expressed as any of the following:

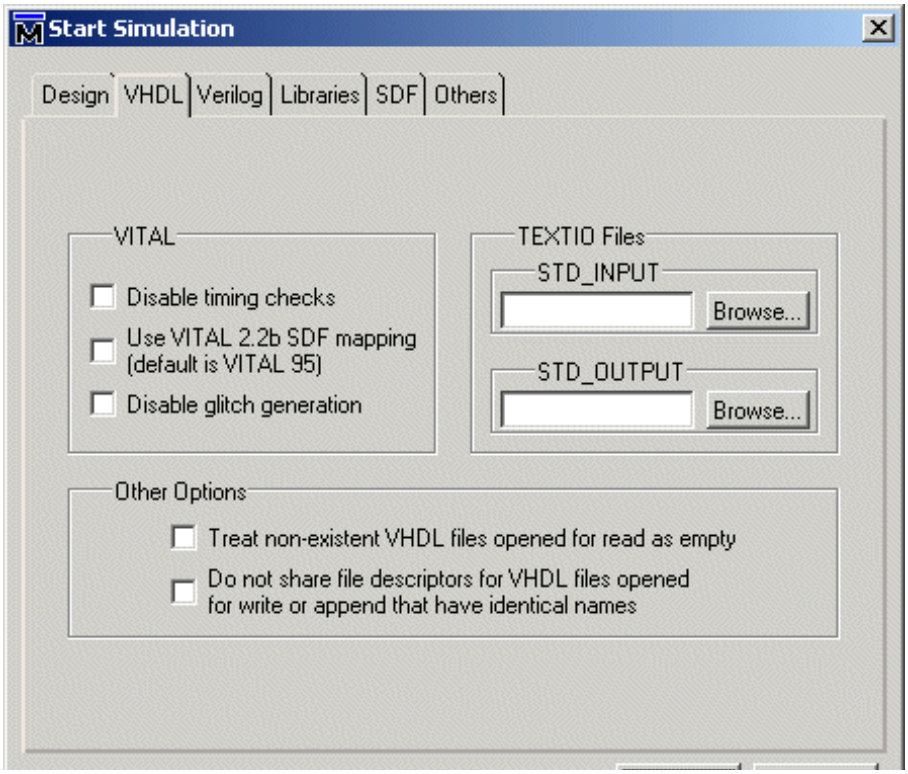
Simulation time units	
1fs, 10fs, or 100fs	femtoseconds
1ps, 10ps, or 100ps	picoseconds
1ns, 10ns, or 100ns	nanoseconds
1us, 10us, or 100us	microseconds
1ms, 10ms, or 100ms	milliseconds
1sec, 10sec, or 100sec	seconds

  See also, "Simulator resolution limit" (UM-78).

- **Enable Optimization**
  Invokes **vopt** on the design prior to loading. Please read "Optimizing Verilog designs" (UM-124) before using this option.

- **Optimization Options**
  Specify optimization options for **vopt**. See "Start Simulation dialog" (GR-76) for details on this dialog.

VHDL tab



The VHDL tab includes these options:

VITAL

- **Disable Timing Checks** (+notimingchecks)
  Disables timing checks generated by VITAL models.
- **Use Vital 2.2b SDF Mapping** (-vital2.2b)
  Selects SDF mapping for VITAL 2.2b (default is Vital95).
- **Disable Glitch Generation** (-noglitch)
  Disables VITAL glitch generation.

TEXTIO files

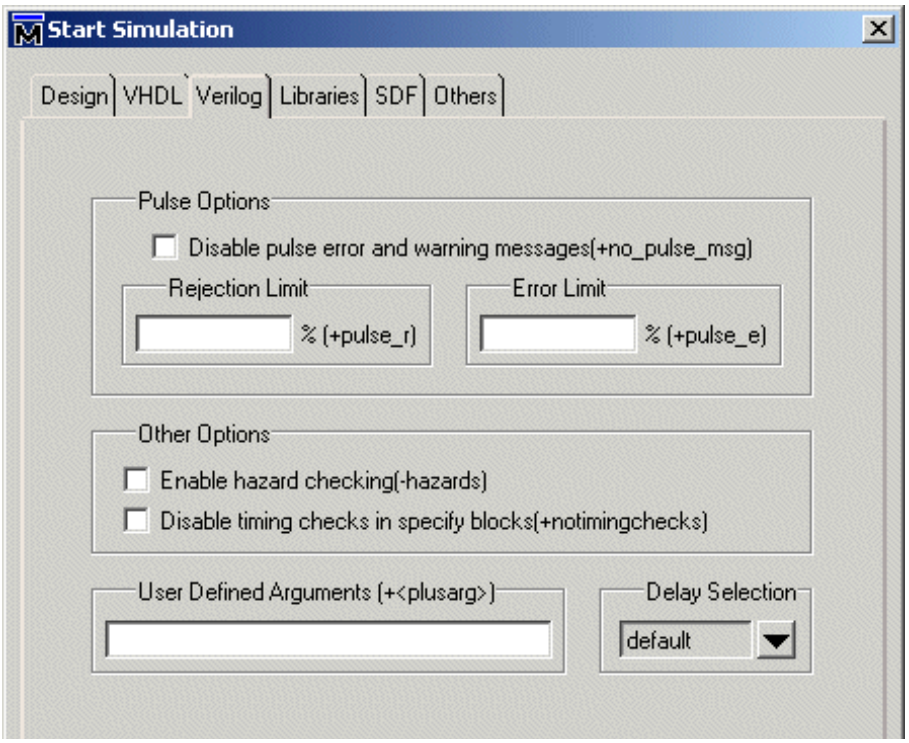
- **STD_INPUT** (-std_input <filename>)
  Specifies the file to use for the VHDL textio STD_INPUT file. Use the **Browse** button to locate a file within your directories.
- **STD_OUTPUT** (-std_output <filename>)
  Specifies the file to use for the VHDL textio STD_OUTPUT file. Use the **Browse** button to locate a file within your directories.

Other Options

- **Treat non-existent VHDL files...**(-absentisempty)
  Specifies that files opened for read that target non-existent files be treated as empty, rather than ModelSim issuing fatal error messages.

- **Do not share file descriptors...**(-nofileshare)
  Turns off file descriptor sharing. By default ModelSim shares a file descriptor for all VHDL files opened for write or append that have identical names.

Verilog tab



The Verilog tab includes these options:

Pulse Options

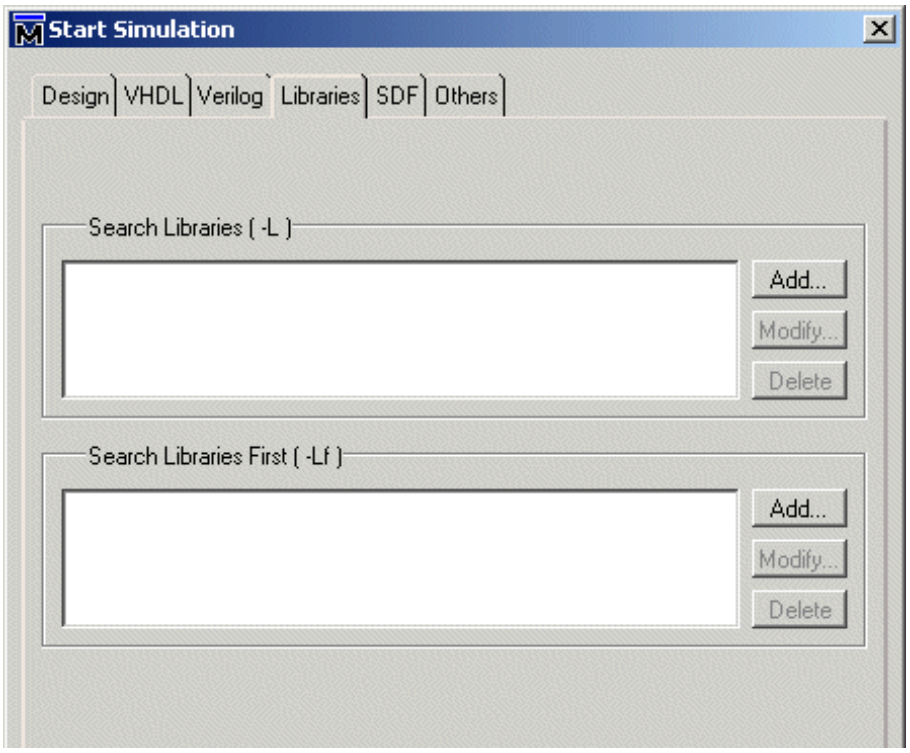
- **Disable pulse error and warning messages** (+no_pulse_msg)
  Disables path pulse error and warning messages.
- **Rejection Limit** (+pulse_r/<percent>)
  Sets the module path pulse rejection limit as a percentage of the path delay.
- **Error Limit** (+pulse_e/<percent>)
  Sets the module path pulse error limit as a percentage of the path delay.

Other Options

- **Enable Hazard Checking** (-hazards)
  Enables hazard checking in Verilog modules.
- **Disable Timing Checks in Specify Blocks** (+notimingchecks)
  Disables the timing check system tasks ($setup, $hold,...) in specify blocks.
- **User Defined Arguments** (+<plusarg>)
  Arguments are preceded with “+”, making them accessible through the Verilog PLI routine **mc_scan_plusargs**. The values specified in this field must have a "+" preceding them or ModelSim may parse them incorrectly.

- **Delay Selection** (+mindelays | +typdelays | +maxdelays)
  Use the drop-down menu to select timing for min:typ:max expressions.

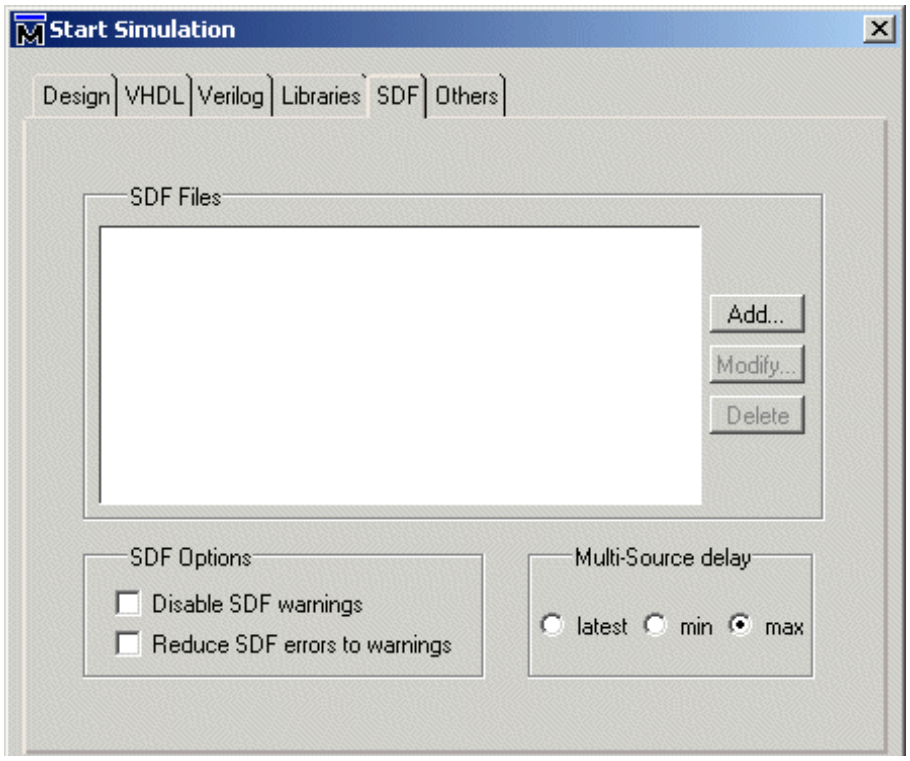
Libraries tab



The Libraries tab includes these options:

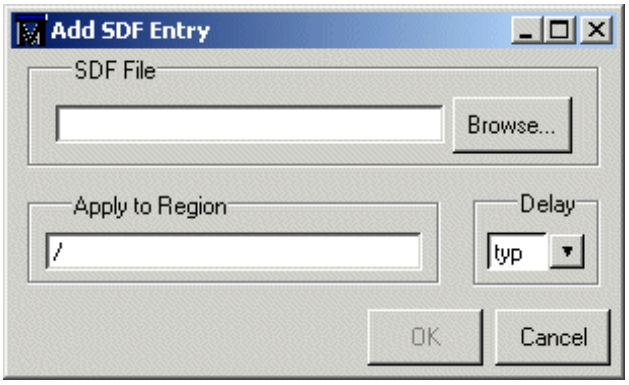
- **Search Libraries** (-L)
  Specifies the libraries to search for design units instantiated from Verilog.
- **Search Libraries First** (-Lf)
  Same as Search Libraries but these libraries are searched before ‘uselib.

SDF tab



The SDF (Standard Delay Format) tab includes these options:

SDF Files



Click the **Add** button to specify the SDF files to load for the current simulation. You may also select an existing file on the listing to **Delete** or **Modify**.

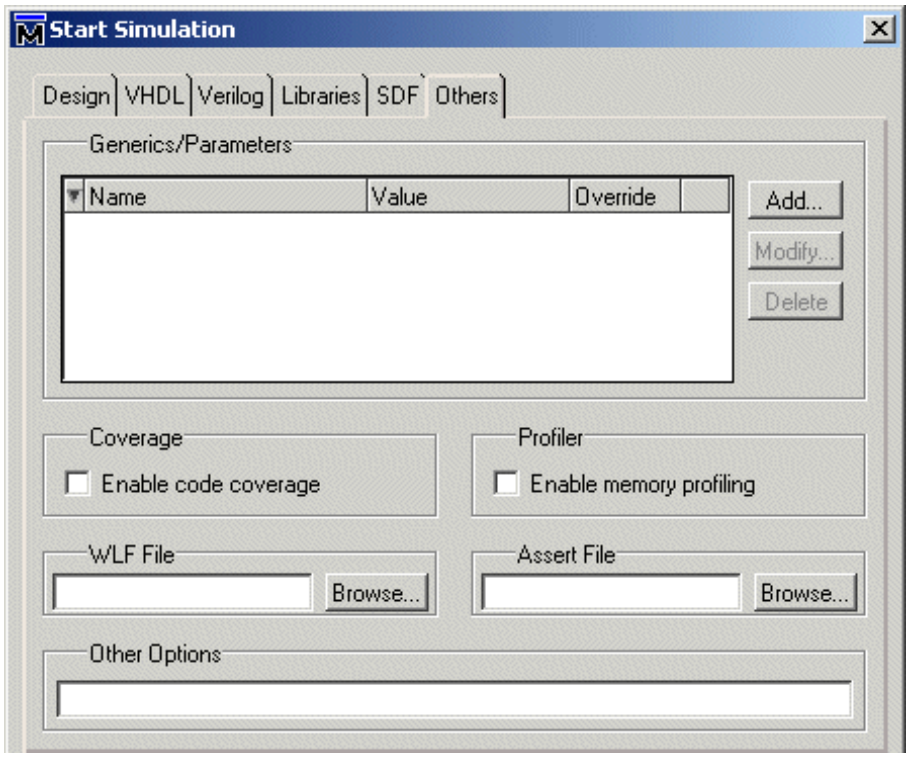
From the **Add SDF Entry** dialog you can set the following options:

- **SDF file** ([<region>] = <sdf_filename>)
  Specifies the SDF file to use for annotation. Use the **Browse** button to locate a file within your directories.
- **Apply to region** ([<region>] = <sdf_filename>)
  Specifies the design region to use with the selected SDF options.
- **Delay** (-sdfmin | -sdftyp | -sdfmax)
  The drop-down menu selects delay timing (min, typ, or max) to be used from the specified SDF file. See also, "Specifying SDF files for simulation" (UM-440).

SDF options

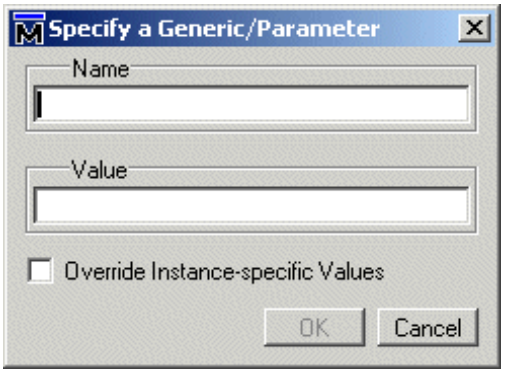
- **Disable SDF warnings** (-sdfnowarn)
  Select to disable warnings from the SDF reader.
- **Reduce SDF errors to warnings** (-sdfnoerror)
  Change SDF errors to warnings so the simulation can continue.
- **Multi-Source Delay** (-multisource_delay <sdf_option>)
  Select **max**, **min**, or **latest** delay. Controls how multiple PORT or INTERCONNECT constructs that terminate at the same port are handled. By default, the Module Input Port Delay (MIPD) is set to the **max** value encountered in the SDF file. Alternatively, you can choose the **min** or **latest** of the values.

Others tab



The Others tab includes these options:

Generics/Parameters



The **Add** button opens a dialog that allows you to specify the value of generics/parameters within the current simulation. You can also select an existing generic/parameter from the list to **Delete** or **Edit**.

From the **Specify a Generic/Parameter** dialog you can set the following options:

- **Name** (-g <Name>=<Value>)
  The name of the generic or parameter. Enter the name as it appears in your source code. Verilog parameters are case sensitive; VHDL generics are case insensitive.
- **Value**
  Specifies a value for all generics/parameters in the design with the given name (above) that have not received explicit values in generic maps (such as top-level generics and generics that would otherwise receive their default value). The value must be appropriate for the declared data type of the generic/parameter. No spaces are allowed in the specification (except within quotes) when specifying a string value.

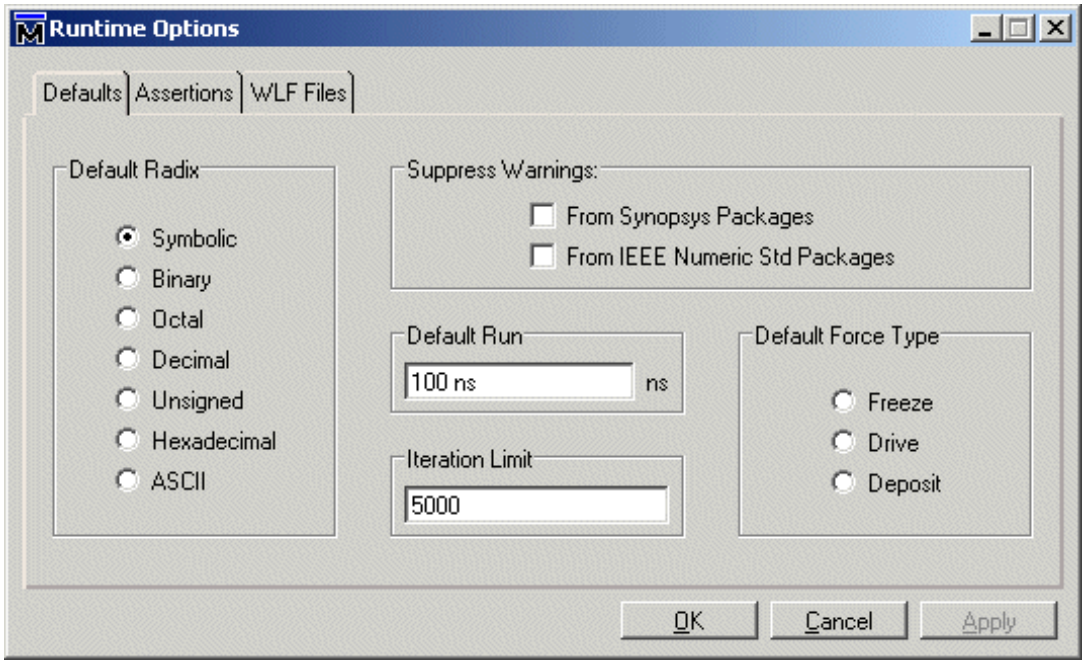
- **Override Instance - specific Values** (-G <Name>=<Value>)
  Select to override generics/parameters that received explicit values in generic maps. The name and value are specified as above. The use of this switch is indicated in the **Override** column of the **Generics/Parameters** list.
- **Enable code coverage** (-coverage)
  Turn on collection of Code Coverage statistics. You must also specify which type of statistics you want when you compile the design. See *Chapter 13 - Measuring code coverage* for more information.
- **Enable memory profiling** (-memprof)
  Causes memory allocation data to be collected during elaboration and simulation. See *Chapter 12 - Profiling performance and memory use* for more information.
- **WLF File** (-wlf <filename>)
  Specify the name of the wave log format (WLF) file to create. The default is vsim.wlf.
- **Assert File** (-assertfile <filename>)
  Designate an alternative file for recording assertion messages. By default assertion messages are output to the file specified by the TranscriptFile variable in the *modelsim.ini* file (see "Creating a transcript file" (UM-537)).
- **Other options**
  Specify any other **vsim** command (CR-373) argument.

Runtime Options dialog

Purpose	Menu command	Additional information
Configure simulation options	**Simulate** > **Runtime Options**	"[vsim] simulator control variables" (UM-529)

Changes made in the **Runtime Options** dialog are written to the active *modelsim.ini* file, if it is writable, and affect the current session as well as all future sessions. If the file is read-only, the changes affect only the current session.

Defaults tab



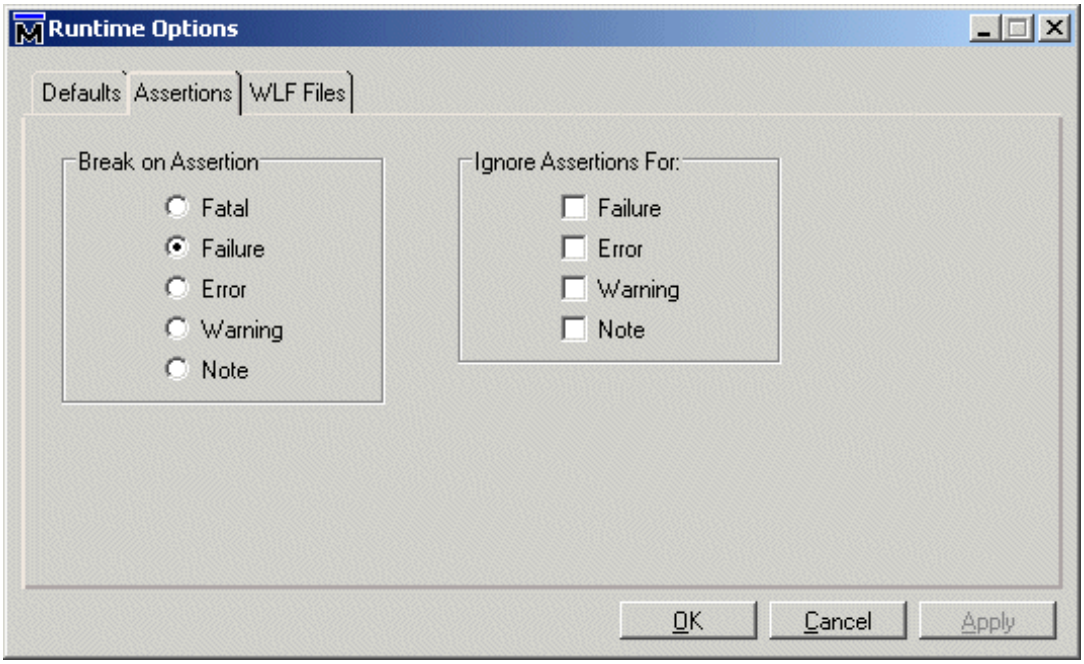
The Defaults tab includes these options:

- **Default Radix**
  Sets the default radix for the current simulation run. You can also use the **radix** (CR-241) command to set the same temporary default. The chosen radix is used for all commands (**force** (CR-180), **examine** (CR-162), **change** (CR-81) are examples) and for displayed values in the Objects, Locals, Dataflow, List, and Wave windows. Corresponding *modelsim.ini* variable is DefaultRadix (UM-531).
- **Suppress Warnings**
  Selecting **From Synopsys Packages** suppresses warnings generated within the accelerated Synopsys std_arith packages. Corresponding *modelsim.ini* variable is StdArithNoWarnings (UM-534).

  Selecting **From IEEE Numeric Std Packages** suppresses warnings generated within the accelerated numeric_std and numeric_bit packages. Corresponding *modelsim.ini* variable is NumericStdNoWarnings (UM-533).

- **Default Run**
  Sets the default run length for the current simulation. Corresponding *modelsim.ini* variable is RunLength (UM-533).
- **Iteration Limit**
  Sets a limit on the number of deltas within the same simulation time unit to prevent infinite looping. Corresponding *modelsim.ini* variable is IterationLimit (UM-532).
- **Default Force Type**
  Selects the default force type for the current simulation. Corresponding *modelsim.ini* variable is DefaultForceKind (UM-531).

Assertions tab



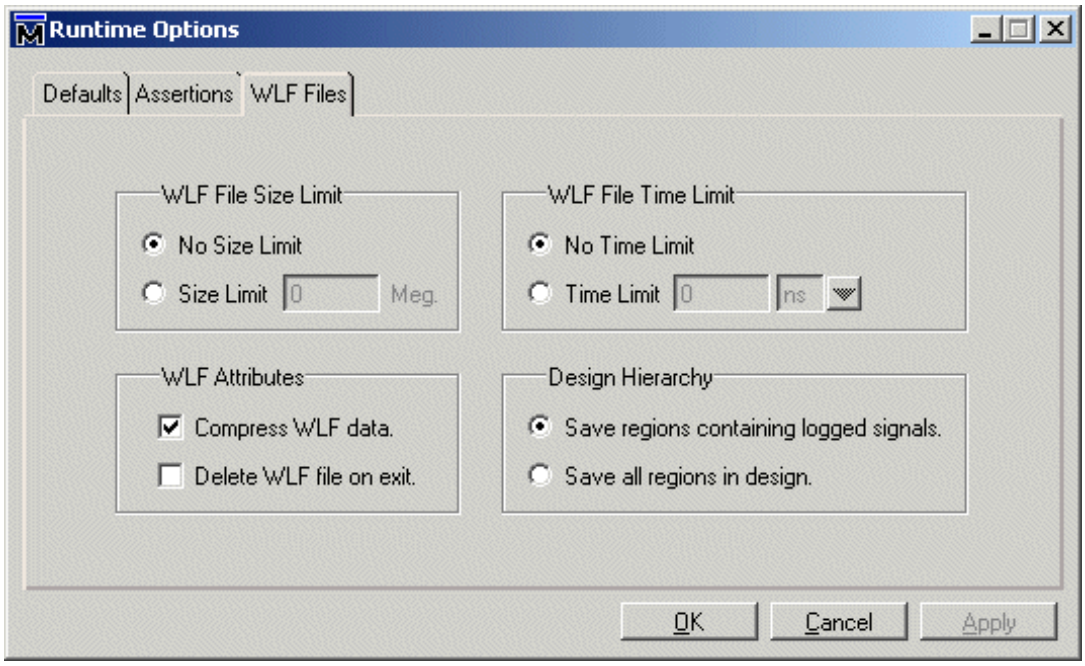
The Assertions tab includes these options:

- **Break on Assertion**
  Selects the assertion severity that will stop simulation. Corresponding *modelsim.ini* variable is BreakOnAssertion (UM-530).
- **Ignore Assertions For**
  Selects the assertion type to ignore for the current simulation. Multiple selections are possible. Corresponding *modelsim.ini* variables are IgnoreFailure, IgnoreError, IgnoreWarning, and IgnoreNote (UM-532).

  When an assertion type is ignored, no message will be printed, nor will the simulation halt (even if break on assertion is set for that type).

Note: Assertions that appear within an instantiation or configuration port map clause conversion function will not stop the simulation regardless of the severity level of the assertion.

WLF Files tab

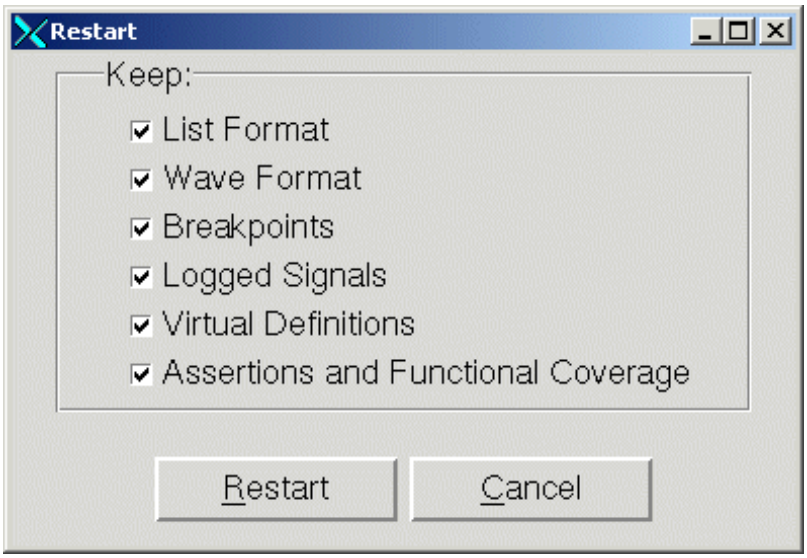


The WLF Files tab includes these options:

- **WLF File Size Limit**
  Limits the WLF file by size (as closely as possible) to the specified number of megabytes. If both size and time limits are specified, the most restrictive is used. Setting it to 0 results in no limit. Corresponding *modelsim.ini* variable is WLFSizeLimit (UM-535).
- **WLF File Time Limit**
  Limits the WLF file by size (as closely as possible) to the specified amount of time. If both time and size limits are specified, the most restrictive is used. Setting it to 0 results in no limit. Corresponding *modelsim.ini* variable is WLFTimeLimit (UM-535).
- **WLF Attributes**
  Specifies whether to compress WLF files and whether to delete the WLF file when the simulation ends. You would typically only disable compression for troubleshooting purposes. Corresponding *modelsim.ini* variables are WLFCompress (UM-535) for compression and WLFDeleteOnQuit (UM-535) for WLF file deletion.
- **Design Hierarchy**
  Specifies whether to save all design hierarchy in the WLF file or only regions containing logged signals. Corresponding *modelsim.ini* variable is WLFSaveAllRegions (UM-535).

Restart dialog

Purpose	Menu command	Additional information
Specify which settings are retained after a restart	**Simulate** > **Run** > **Restart**	**restart** command (CR-246)



The Restart dialog includes the following options:

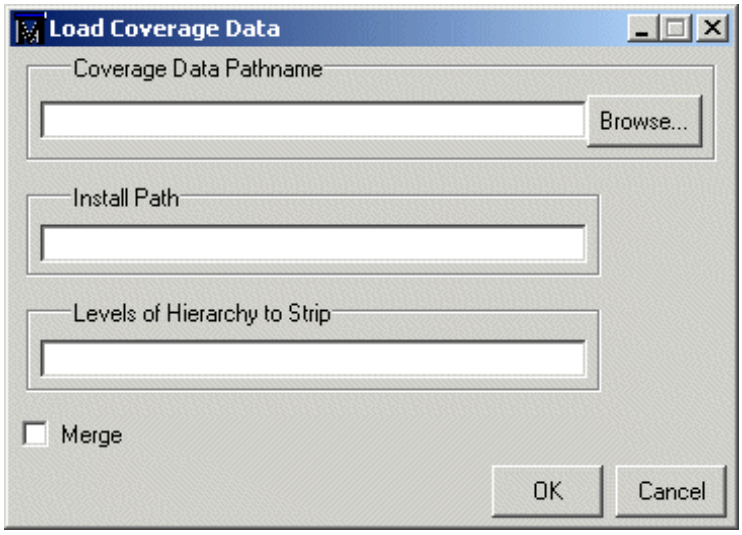
- **List Format**
  Retain all objects in the List window and their formats.
- **Wave Format**
  Retain all objects in the Wave window and their formats.
- **Breakpoints**
  Re-install all breakpoints after simulation is restarted.
- **Logged Signals**
  Retain logging of all currently logged objects.
- **Virtual Signals**
  Retain currently defined virtual definitions (e.g., virtual functions, virtual signals, etc.).
- **Assertions and Functional Coverage**
  Retain assertion and functional coverage settings.

Waveform Compare dialogs

These dialogs, accessed via the **Tools** > **Waveform Compare** menu, are documented in the "Wave window" (GR-211) section.

Load Coverage Data dialog

Purpose	Menu command	Additional information
Reload previously saved coverage data	**Tools > Code Coverage > Load**	"Saving and reloading coverage data" (UM-354)

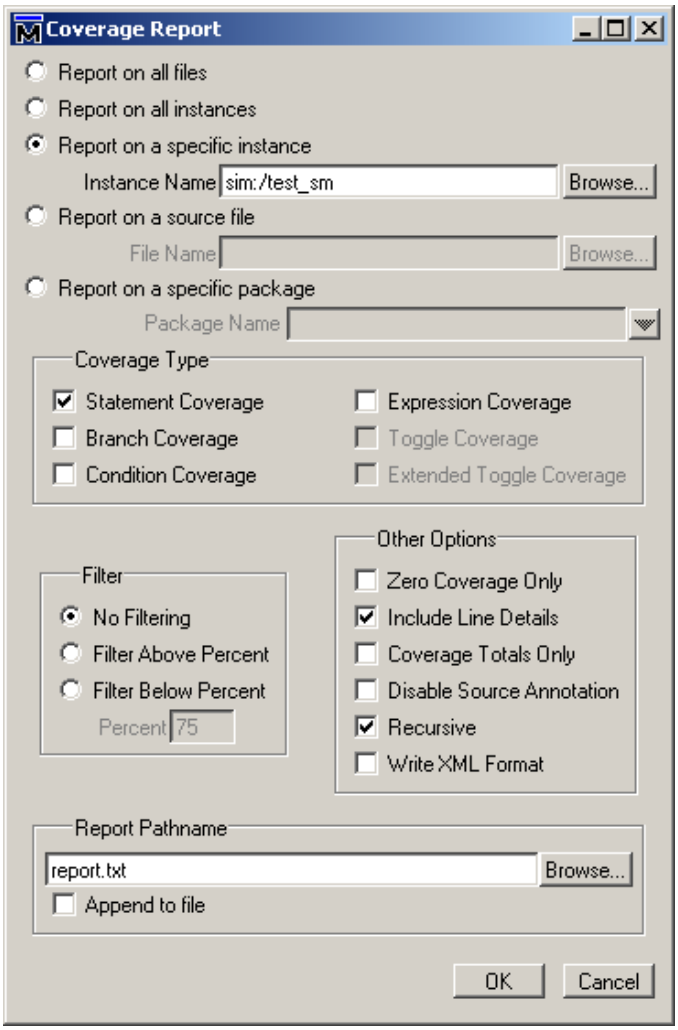


The Load Coverage Data dialog includes the following options:

- **Coverage Data Pathname**
  The pathname to the coverage data file you want to load.
- **Install Path**
  Adds whatever you specify as additional hierarchy on the front end of instance and signal names in the data file. This option allows you to merge coverage results from simulations that have different hierarchies.
- **Levels of Hierarchy to Strip**
  Removes the specified levels of hierarchy from instance and signal names in the data file. Enter an integer. This option allows you to merge coverage results from simulations that have different hierarchies.
- **Merge**
  Merges the saved coverage data with coverage data in the current simulation. If you don't check merge, ModelSim clears existing coverage data before loading the saved data.

Coverage Report dialog

Purpose	Menu command	Additional information
Save textual reports of Code Coverage data	**Tools > Code Coverage > Reports**	"Reporting coverage data" (UM-350)



The Coverage Report dialog includes these options:

- **Report on all files**
  Saves a textual summary for each file in the design.
- **Report on all instances**
  Saves a textual summary for each instance in the design.

- **Report on a specific instance**
  Saves a textual summary for the specified instance. The selected instance automatically appears in the *Instance Name* field. You can browse for other instances.
- **Report on a source file**
  Saves a textual summary for the specified source file. The selected file automatically appears in the *File Name* field. You can browse for other source files.
- **Report on a specific package**
  Saves a textual summary for the specified HDL package. The selected package automatically appears in the *Package Name* field. You can browse for other HDL packages.

Coverage Type

Select the type of coverage to be reported – statement, branch, condition, expression, toggle, and extended toggle coverage.

Filter

Specifies whether to filter the report based on coverage percentage. You can choose to filter objects with coverage above or below a certain percentage.

Other Options

- **Zero Coverage Only**
  Saves a textual summary of statement and branch coverage that includes columns for the number of statements and branches not executed.
- **Include Line Details**
  Saves a detailed textual report of the statement and branch coverage for every line of code.
- **Coverage Totals Only**
  Saves a text report of the coverage totals by files and by instances. Includes total hits and coverage percentages for all active statements and branches.
- **Disable Source Annotation**
  Removes source code from coverage reports.
- **Recursive**
  Reports on the specified instance, and all included instances, recursively.
- **Write XML format**
  Produces output in an XML-structured format. See "XML output" (UM-351) for an example.

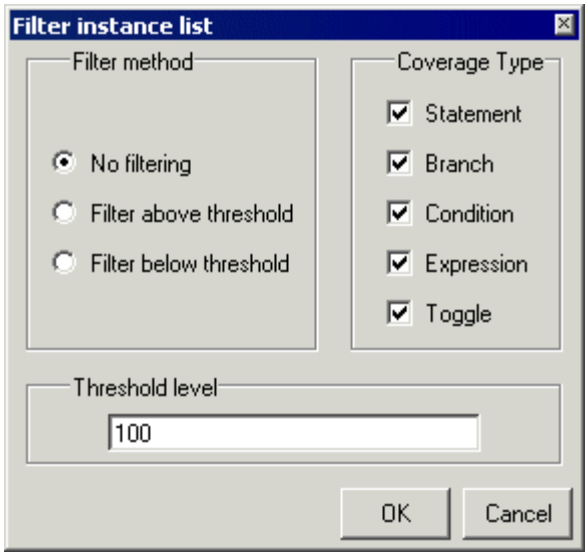
Report Pathname

Specify a pathname for the output file.

- **Append to file**
  Appends output to an existing file rather than overwriting the file.

Filter instance list dialog

Purpose	Menu command	Additional information
Filter coverage statistics	Right-click in Instance Coverage pane > Set Filter	"Filtering coverage data" (UM-346)

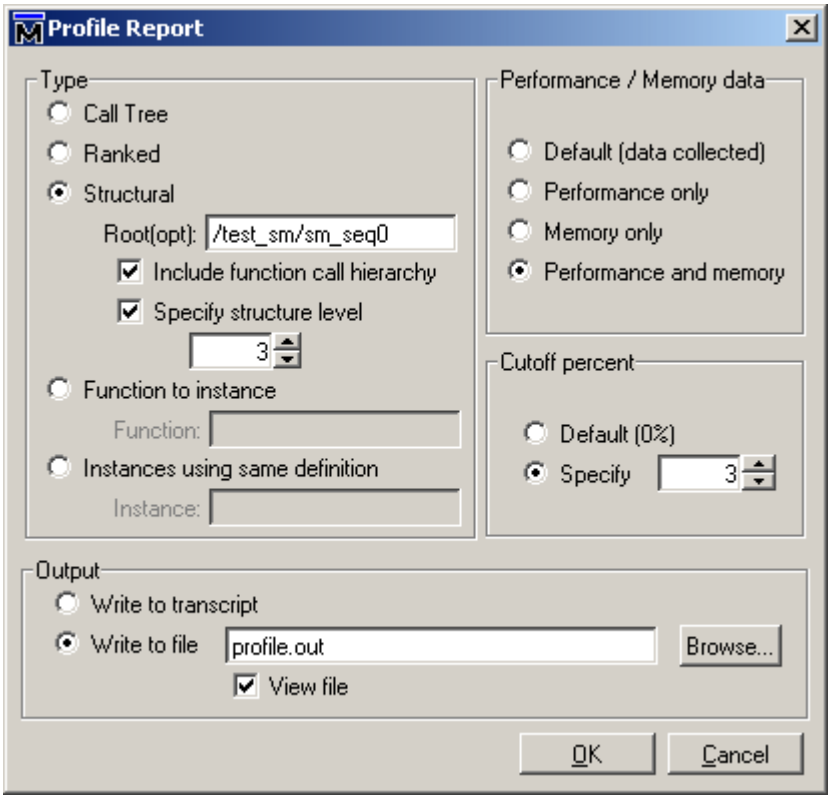


The Filter instance dialog includes these options:

- **Filter method**
  Specifies whether you want to filter objects that exceed the threshold or fall below the threshold.
- **Coverage Type**
  Determines which coverage statistics you want to filter.
- **Threshold level**
  Specifies the percentage above or below which objects are filtered.

Profile Report dialog

Purpose	Menu command	Additional information
Create textual reports from performance and memory profile results	**Tools > Profile > Profile Report**	Reporting profiler results (UM-331) or **profile report** command (CR-231)



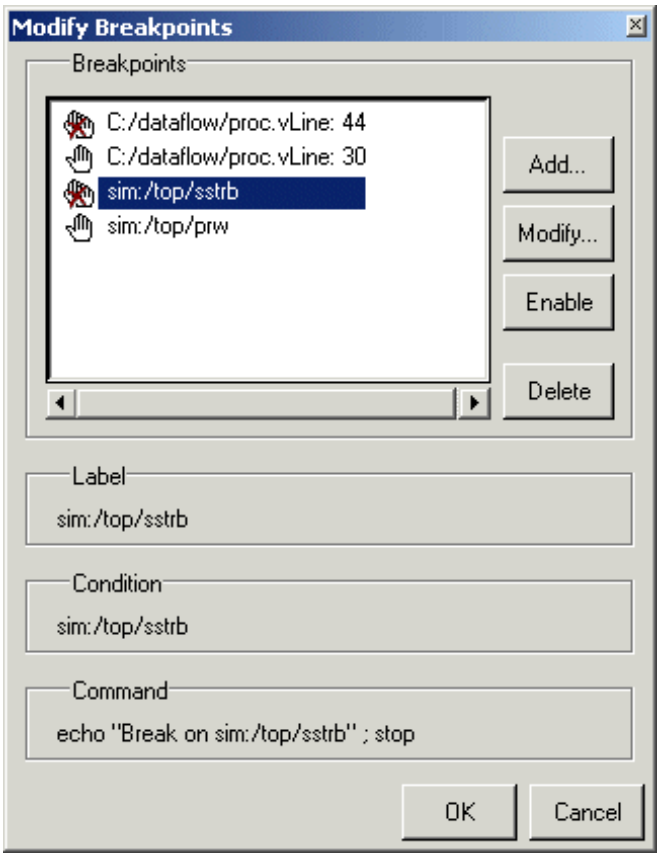
The Profile Report dialog includes the following options:

- **Type**
  Save a textual report from Call Tree, Ranked, and Structural profile data. The Structural option allows you to designate the root instance for the report, include the function call hierarchy, and specify the structure level. You can also create a Function to instance report for the designated function, and a report of Instances using the same definition as the designated instance.
- **Performance/Memory data**
  Elect to save performance profile data only, memory allocation data only, or both.
- **Cutoff percent**
  Report results including and above the designated or Default percentage.

- **Output**
  Writes the textual report directly to the transcript or to a file. Will also display the file immediately after it is created if "View file" is selected.

Modify Breakpoints dialog

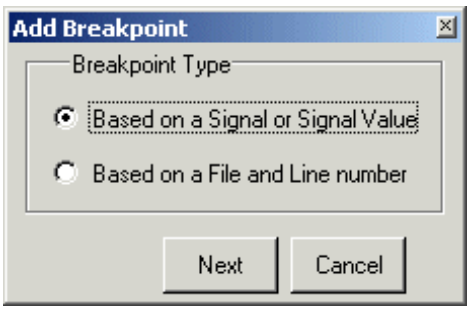
Purpose	Menu command	Additional information
Add or manage signal and file breakpoints	**Tools** > **Breakpoints**	"Creating and managing breakpoints" (GR-264)



The Breakpoints dialog includes these options:

- **Breakpoints**
  List of all existing breakpoints. Breakpoints set from anywhere in the GUI, or from the command line, are listed. A red 'X' through the hand icon means the breakpoint is currently disabled.

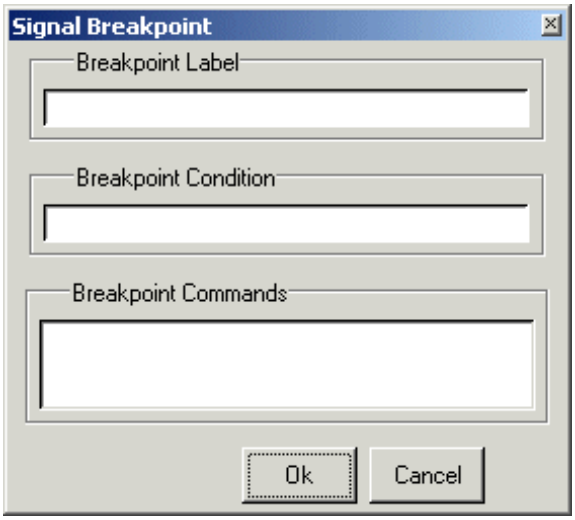
- **Add**
  Create a new signal or file-line breakpoint. When you click Add to add a new breakpoint, you will see the Add Breakpoint dialog. Choose whether to create a signal breakpoint or a file-line breakpoint and then select Next. Depending on which type of breakpoint you are creating, you will see one of the two dialogs described below.



- **Modify**
  Change properties of an existing breakpoint.
- **Disable/Enable**
  De-activate or activate the selected breakpoint.
- **Delete**
  Delete the selected breakpoint.
- **Label**
  Text label of the selected breakpoint. Entered in the Signal Breakpoint or File Breakpoint dialog (described below).
- **Condition**
  The condition under which the breakpoint will be hit. Entered in the Signal Breakpoint or File Breakpoint dialog (described below).
- **Command**
  The command that will be executed when the breakpoint is hit. Entered in the Signal Breakpoint or File Breakpoint dialog (described below).

Signal Breakpoint dialog

Purpose	Menu command	Additional information
Add/modify signal breakpoints	**Tools** > **Breakpoints**	"Creating and managing breakpoints" (UM-239)

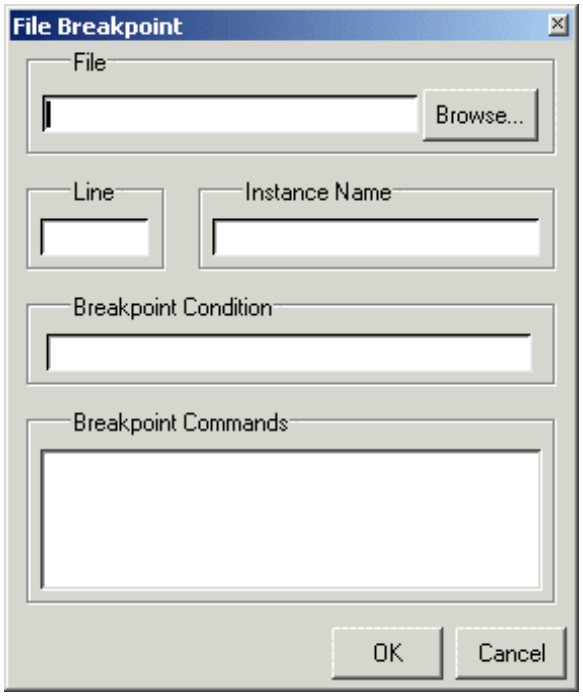


The Signal Breakpoint dialog includes these options:

- **Breakpoint Label**
  Specify an optional text label for the breakpoint.
- **Breakpoint Condition**
  Specify condition(s) to be met for the command(s) to be executed. See the **when** command (CR-407) for more information on creating the condition statement.
- **Breakpoint Commands**
  Specify command(s) to be executed when the condition is met. Any ModelSim or Tcl command or series of commands are valid, with one exception – the **run** command (CR-252) cannot be used.

File Breakpoint dialog

Purpose	Menu command	Additional information
Add/modify file breakpoints	**Tools** > **Breakpoints**	"Creating and managing breakpoints" (UM-239)

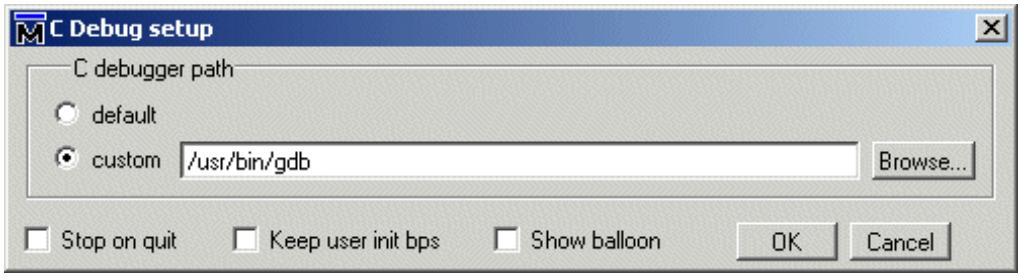


The File Breakpoint dialog includes these options:

- **File**
  Specify the file in which to set the breakpoint.
- **Line**
  Specify the line number on which to set the breakpoint. Note that breakpoints can be set only on executable lines.
- **Instance Name**
  Specify a region in which to apply the breakpoint. If left blank the breakpoint affects every instance in the design.
- **Breakpoint Condition**
  Specify a condition that determines whether the breakpoint is hit.
- **Breakpoint Commands**
  Specify command(s) to be executed when the breakpoint is hit. Any ModelSim or Tcl command or series of commands is valid, with one exception – the **run** command (CR-252) cannot be used.

C Debug setup dialog

Purpose	Menu command	Additional information
Configure C Debug	**Tools > C Debug > C Debug setup**	"Setting up C Debug" (UM-402)



The C Debug setup dialog includes these options:

- **C debugger path**
  Specifies the path to the installed copy of **gdb**. Select "default" to point at the Model Technology supplied gdb or "custom" to point at another installation of gdb. See "Supported platforms and gdb versions" (UM-401) for the supported versions.
- **Stop on quit**
  Allows you to debug functions that get called when the simulator is exiting. See "Debugging functions when quitting simulation" (UM-414) for details.
- **Keep user init bps**
  Leaves enabled any breakpoints you set while running in initialization mode (see "Debugging functions during elaboration" (UM-410)). Normally breakpoints set during initialization mode are disabled once the design is finished loading.
- **Show source balloon**
  Enables name/value popup in the Source window when you hover your mouse pointer over a variable name.

Command entry dialog

Purpose	Menu command	Additional information
Enter C Debug commands	**Tools > C Debug > Command entry**	NA

This dialog lets you enter C Debug commands even if the CDBG> prompt is inaccessible. The GUI prompt may become inaccessible in certain situations (e.g., when debugging FLI LoadDone callback functions)

The Command entry dialog includes this field:

- **Enter command**
  Specify the debugging command to execute.

Tcl debugger

TclDebugger and TclPro Debugger are third-party tools we make available through ModelSim. They are described briefly in the Tcl chapter. See "The Tcl Debugger" (UM-493) and "TclPro Debugger" (UM-497).

Macro dialog

Purpose	Menu command	Additional information
Record a series of mouse movements and key strokes	**Tools** > **Macro Helper**	"Macro helper" (UM-492)

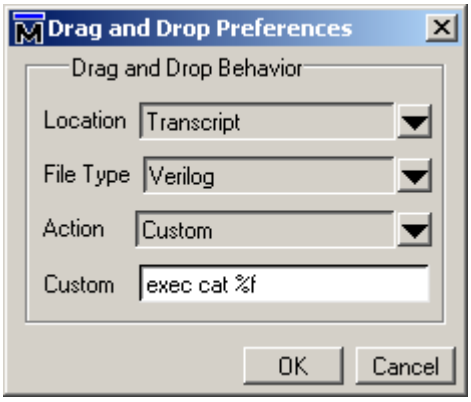
This tool is available for UNIX only (excluding Linux).

The macro dialog includes these options.

- **Macro name**
  Type a filename for the macro. The resulting file can be called from a more complex macro by using the **play** (CR-218) command.
- **Record/Stop**
  Record begins recording and toggles to Stop once a recording begins**.**
- **Insert Pause**
  Inserts a .5 second pause into the macro file. Press the button more than once to add more pause time.
- **Play**
  Play the macro specified in the file name field.

Drag and Drop Preferences dialog

Purpose	Menu command	Additional information
Set the action to be performed when a file is dragged and dropped into the Project, Transcript or Wave windows	**Tools > Options > Drag and Drop Preferences**	



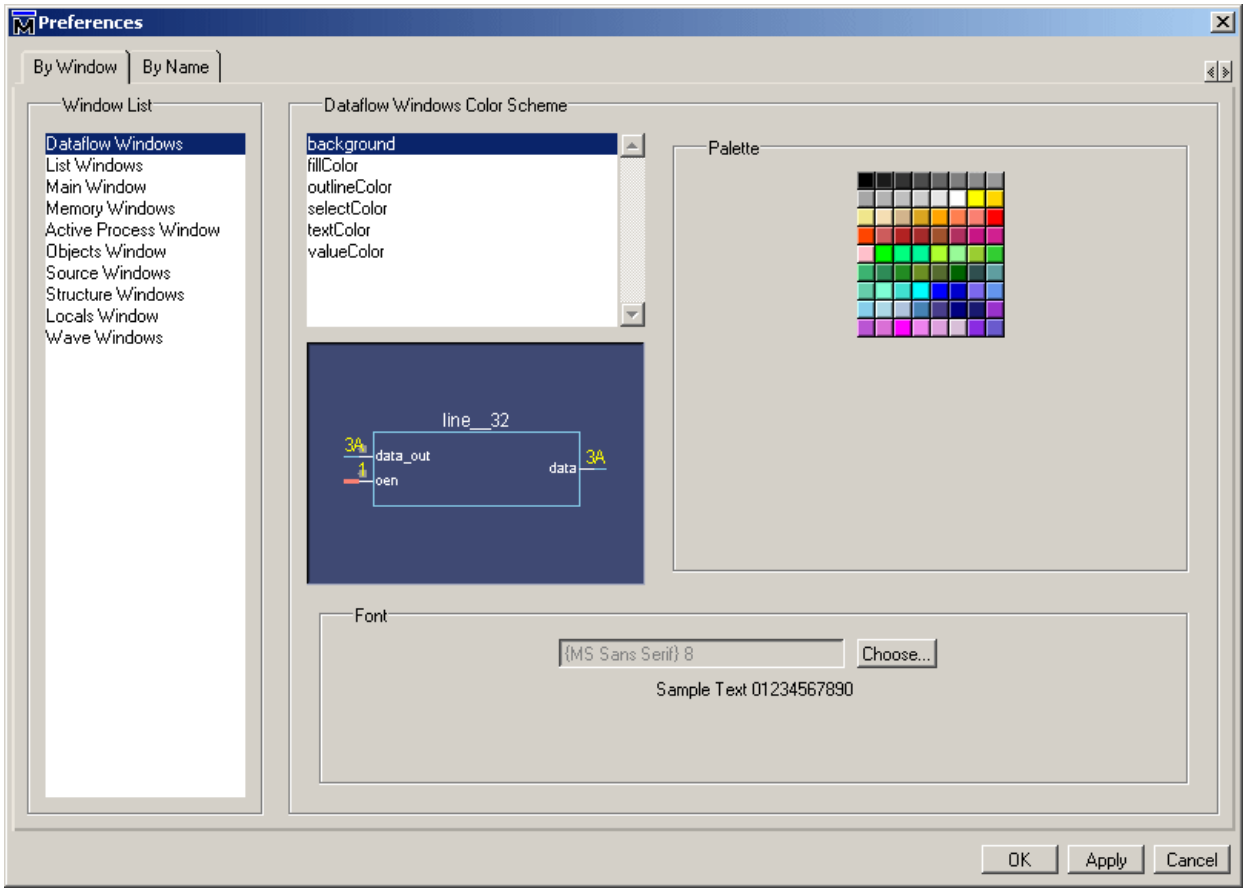
The dialog contains four fields:

- **Location**
  Select the Project pane, the Transcript pane or the Wave window. When a file of the type specified in the File Type field is dragged and dropped into the selected location, the specified action will be performed.
- **File Type**
  Select from a list of file types that includes: Verilog, VHDL, PSL, Text, SystemC, TCL, Macro, VCD, SDF, XML and Logfile. When a file of the type specified is dragged and dropped into the specified location, the specified action will be performed.
- **Action**
  Allows you to select the following actions: Open, Execute, Add to Project, and Custom. When Custom is selected, the Custom field becomes active.
- **Custom**
  Allows you to specify a custom action to be performed when the specified file type is dragged and dropped to the specified location. The action is defined with a Tcl command. If the command requires a pathname of an object, place "%f" in the command. ModelSim will substitute the appropriate pathname and execute the appropriate action.

Preferences dialog

Purpose	Menu command	Additional information
Edit ModelSim Tcl preference variables	**Tools > Edit Preferences**	"Preference variables located in Tcl files" (UM-540)

By Window tab

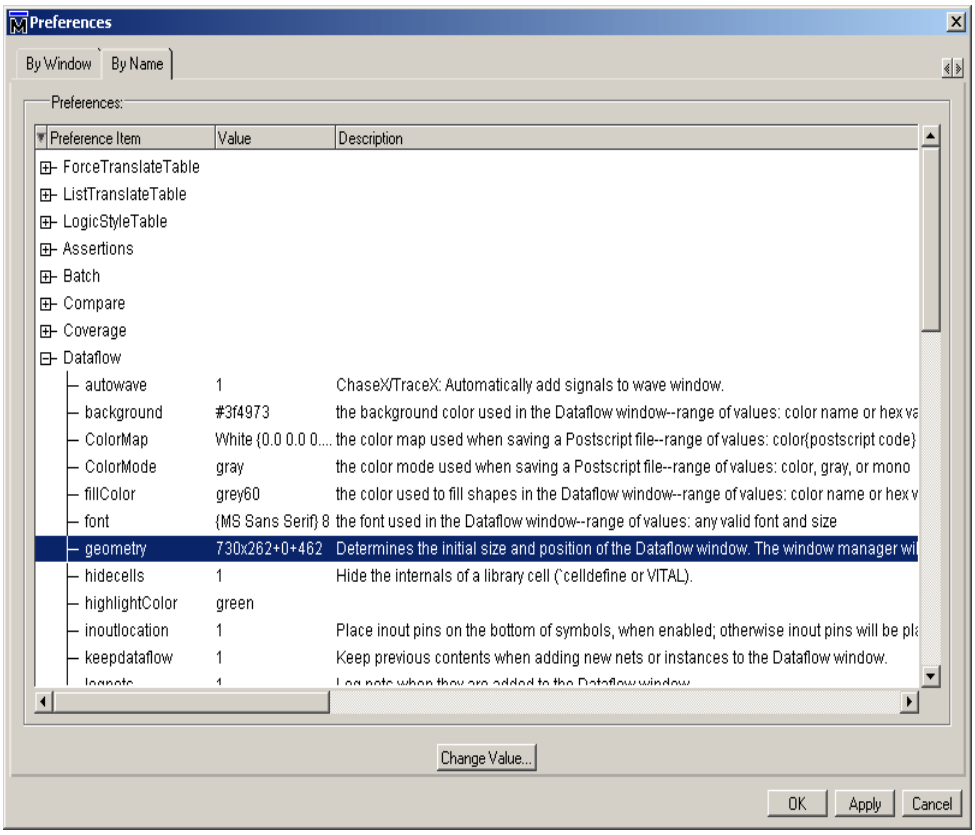


The By Window tab includes the following options:

- **Window List**
  Select a window type.
- **Color Scheme**
  Select a window component of which you want to change the color. Select a color from the Palette.
- **Font**
  Choose a font for text in the selected window type.
- **Apply**
  Apply the changes for the current ModelSim session only.

- **OK**
  Saves the current preference settings to a user preference file that is invoked each time ModelSim is invoked. See "ModelSim GUI preferences" (GR-266) for further details.

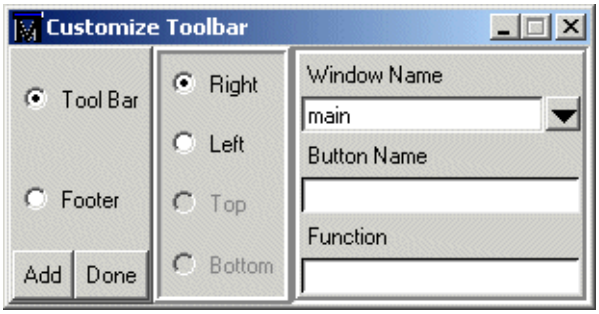
By Name tab



The By Name tab organizes Tcl preference variables by category rather than by window. Select a preference item then click Change Value to edit the variable.

Customize Toolbar dialog

Purpose	Menu command	Additional information
Add user-defined toolbar buttons	**Window > Customize**	NA



The Customize Toolbar dialog includes the following options:

- **Tool Bar**
  Adds the button to a new toolbar in the specified window.
- **Footer**
  Adds the button to the window's status bar.
- **Placement options**
  Justify the button on the right-side, left-side, top/center, or bottom/center of the toolbar or footer.
- **Window Name**
  The window to which you want to add the button.
- **Button Name**
  The text that will appear on the button.
- **Function**
  Any command or macro you might execute from the ModelSim command line. For example, you might want an **add wave** button in the Wave window.

Making the button persistent

User-defined buttons exist only until you close the window unless you add the button code to the window's user hook variable. When you create a button, the underlying commands are echoed in the transcript. You can use these commands to make the button appear every time you invoke the window. Follow these steps:

1. Create a button.
2. Copy the commands from the transcript into a Tcl procedure in the *modelsim.tcl* file. If you don't have a *modelsim.tcl* file already, create a new text file with that name and set the MODELSIM_TCL environment variable to the full path of the *modelsim.tcl* file.

3 Append the procedure name to the window's user_hook Tcl variable. See "Preference variables located in Tcl files" (UM-540) for more information on Tcl preference variables.

An example will help clarify. Say you create a button in the Wave window that adds all signals from the selected region to the Wave window. The button code will look something like this:

```
_add_menu .wave controls right SystemButtonFace black AddWaves {add wave *}
```

You would insert that code into a Tcl procedure in the *modelsim.tcl* file and then append the procedure to the PrefWave(user_hook) variable. The entire entry in the *modelsim.tcl* file would look as follows:

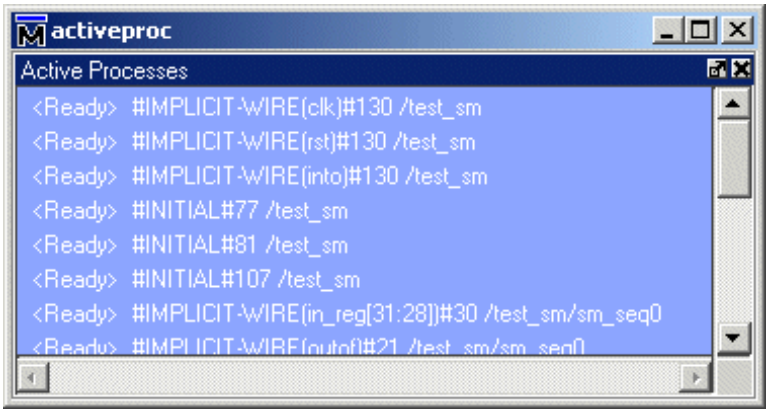
```
proc AddWaves winname {
_add_menu .wave controls right SystemButtonFace black AddWaves {add wave *}
}

lappend PrefWave(user_hook) AddWaves
```

Now, any time you start ModelSim and open the Wave window, it will have a button labeled "AddWaves" that executes the command "add wave *".

Active Processes pane

The Active Processes pane displays a list of HDL and SystemC processes.



Processes are also displayed in the structure tabs of the Main window Workspace. To filter displayed processes in the structure tabs, select **View** > **Filter** > **Processes**.

Process status

Each object in the scrollbox is preceded by one of the following indicators:

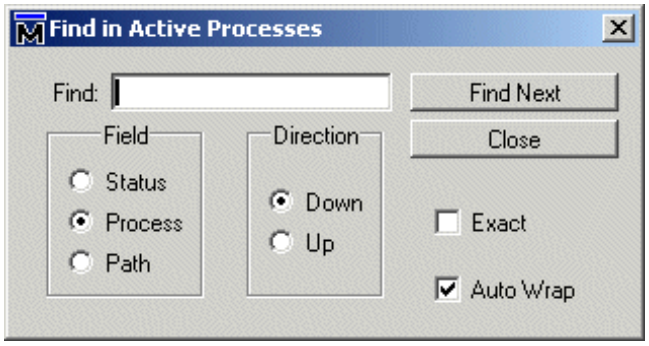
- **<Ready>**
  Indicates that the process is scheduled to be executed within the current delta time. If you select a "Ready" process, it will be executed next by the simulator.
- **<Wait>**
  Indicates that the process is waiting for a VHDL signal or Verilog net or variable to change or for a specified time-out period. SystemC objects cannot be in a Wait state.
- **<Done>**
  Indicates that the process has executed a VHDL wait statement without a time-out or a sensitivity list. The process will not restart during the current simulation run. SystemC objects cannot be in a Done state.

Active Processes dialogs

This section describes the processes-related dialogs that are accessed via the Main window menu bar. Not all dialogs are documented (e.g., File > Save).

Find in Active Processes dialog

Purpose	Menu command	Additional information
Locate object names	**Edit > Find**	NA



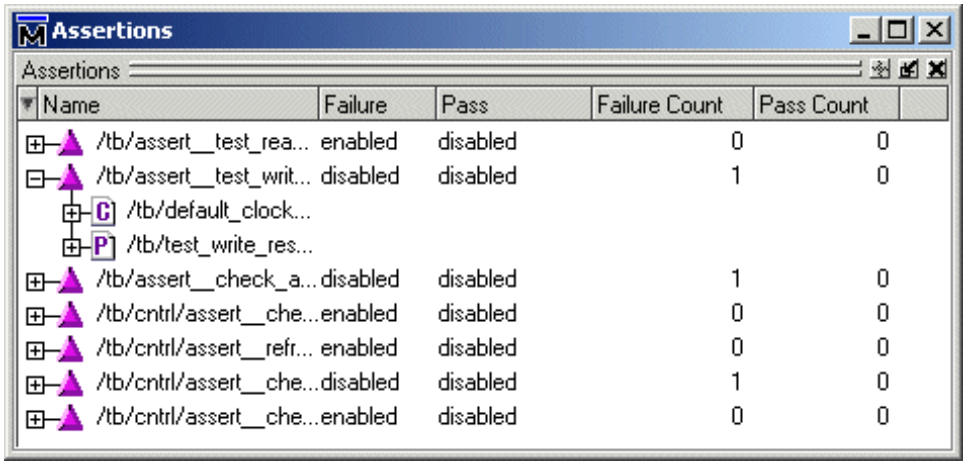
You must activate the Active Processes pane by clicking in it before this dialog will be available.

The Find in Active Processes dialog includes these options:

- **Find**
  Specify the text for which you want to search.
- **Field**
  Specify which column of the pane to search.
- **Direction**
  Specify which direction to start searching.
- **Exact**
  Check **Exact** if you only want to find objects that match your search exactly. For example, searching for "initial" without Exact will find *#INITIAL74#, #INITIAL104#, etc*.
- **Auto Wrap**
  Check **Auto Wrap** to continue the search at the top or bottom of the pane.

Assertions pane

The Assertions pane provides a convenient interface to PSL assertions in the current simulation. For more information on PSL assertions, see *Chapter 14 - PSL Assertions* in the *ModelSim User's Manual*.



The Assertions pane lists all embedded and external assertions that were successfully compiled and simulated during the current session. The plus sign ('+') to the left of the Name column lets you expand the assertion hierarchy to show its elements (properties, sequences, clocks, and HDL signals).

Assertions pane columns

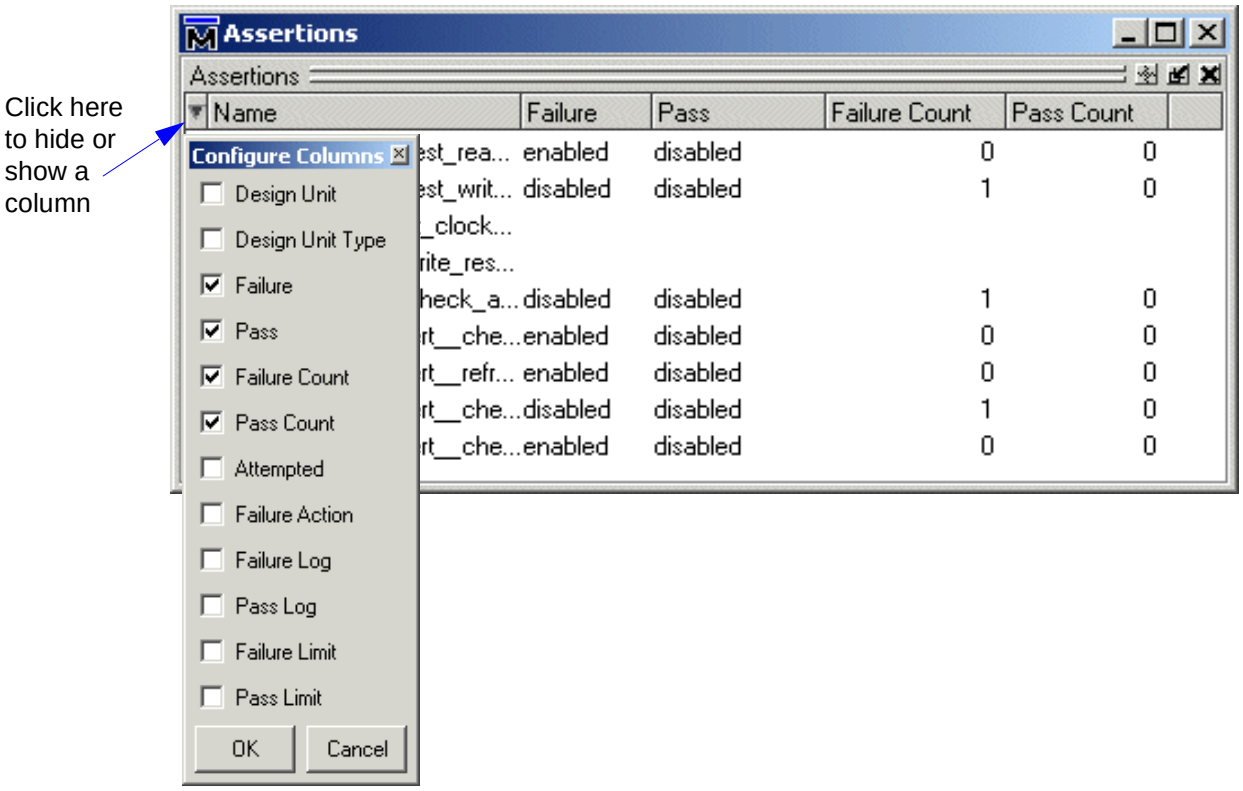
The Assertions pane includes the columns described below, though only five are displayed by default. See "Hiding/showing columns in the Assertions pane" (GR-112) for details on how to show or hide particular columns.

- The **Name** column lists the PSL statement or vunit name you specified in the assertion code. For vunits the individual assertion names are listed under the vunit name. Also, any signal referenced in an assertion will be part of the hierarchy as well. See "Understanding assertion names" (UM-370) for more details on assertion names.
- The **Design Unit** column identifies the design unit to which the assertion is bound. Not displayed by default.
- The **Design Unit Type** column lists the HDL type of the design unit. Not displayed by default.
- The **Failure** column shows "enabled" when failure checking is enabled on the assertion. If the column shows "disabled", ModelSim isn't currently checking that assertion's failures. Once ModelSim reaches the Failure limit, it disables failure checking (see "Setting failure and pass limits" (UM-379)).
- The **Pass** column shows "enabled" when pass checking is enabled on the assertion. If the column shows disabled, ModelSim isn't currently tracking that assertion's checking. Once ModelSim reaches the Pass limit, it disables pass checking (see "Setting failure and pass limits" (UM-379)).

- The **Failure Count** column counts the total number of times the assertion has failed in the current simulation. These counts are maintained between runs unless you reset the count for the assertion.
- The **Pass Count** column counts the total number of times the assertions has passed in the current simulation. These counts are maintained between runs unless you reset the count for the assertion.
- The **Attempted** column shows a green checkmark when an assertion has triggered and a red ’X’ when it has not triggered. Not displayed by default.
- The **Failure Action** column lists the action that ModelSim takes when the assertion passes or fails. Not displayed by default.
- The **Failure Log** column shows "enabled" when failure messages will be logged to the transcript. The column shows "disabled" when failure messages will not be logged to the transcript. Not displayed by default.
- The **Pass Log** column shows "enabled" when pass messages will be logged to the transcript. The column shows "disabled" when pass messages will not be logged to the transcript. Not displayed by default.
- The **Failure Limit** column shows the number of times ModelSim will respond to a failure event on an assertion. Not displayed by default.
- The **Pass Limit** column shows the number of times ModelSim will respond to a pass event on an assertion. Not displayed by default.

Hiding/showing columns in the Assertions pane

You can hide or show any of the columns in the Assertions pane. Click the drop-down arrow on the left-hand side of the dialog and select a column name.



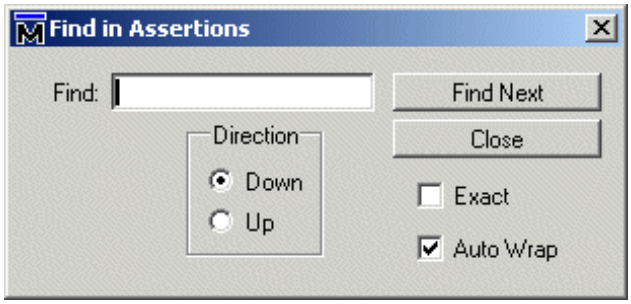
The selection acts as a toggle–select it once to hide a column; select it again to show the column.

Assertions pane dialogs

This section describes the assertions-related dialogs that are accessed via the Main window menu bar.

Find in Assertions dialog

Purpose	Menu command	Additional information
Locate text in the Assertions pane	**Edit > Find**	NA



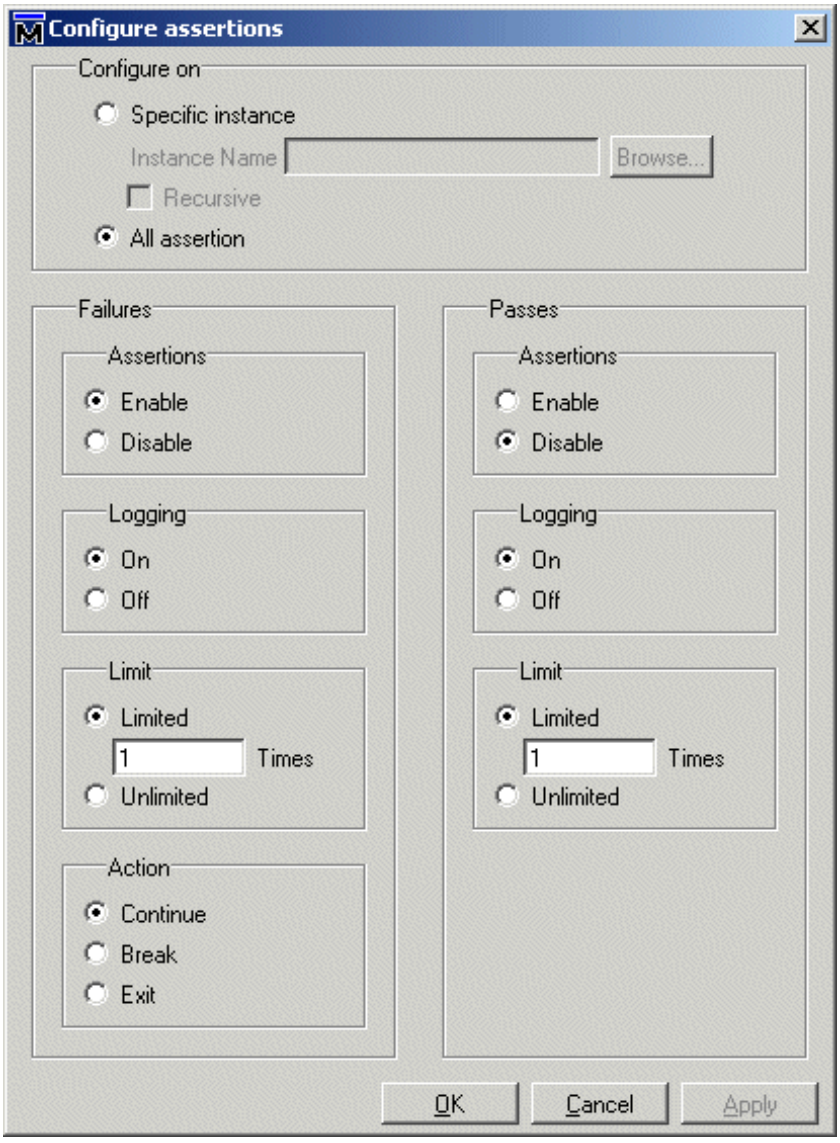
You must activate the Assertions pane by clicking in it before this dialog will be available.

The Find in Assertions dialog includes these options:

- **Find**
  The text you want to locate.
- **Direction**
  Specify the direction to begin searching.
- **Exact**
  Check Exact if you only want to find objects that match your search exactly. For example, searching for "read" without Exact will find *assert_check_read* and *test_read_response*.
- **Auto Wrap**
  Check Auto Wrap to continue the search at the top or bottom of the list.

Configure assertions dialog

Purpose	Menu command	Additional information
Configure assertion behavior	**Edit** > **Advanced** > **Change**	"Managing assertions" (UM-376)



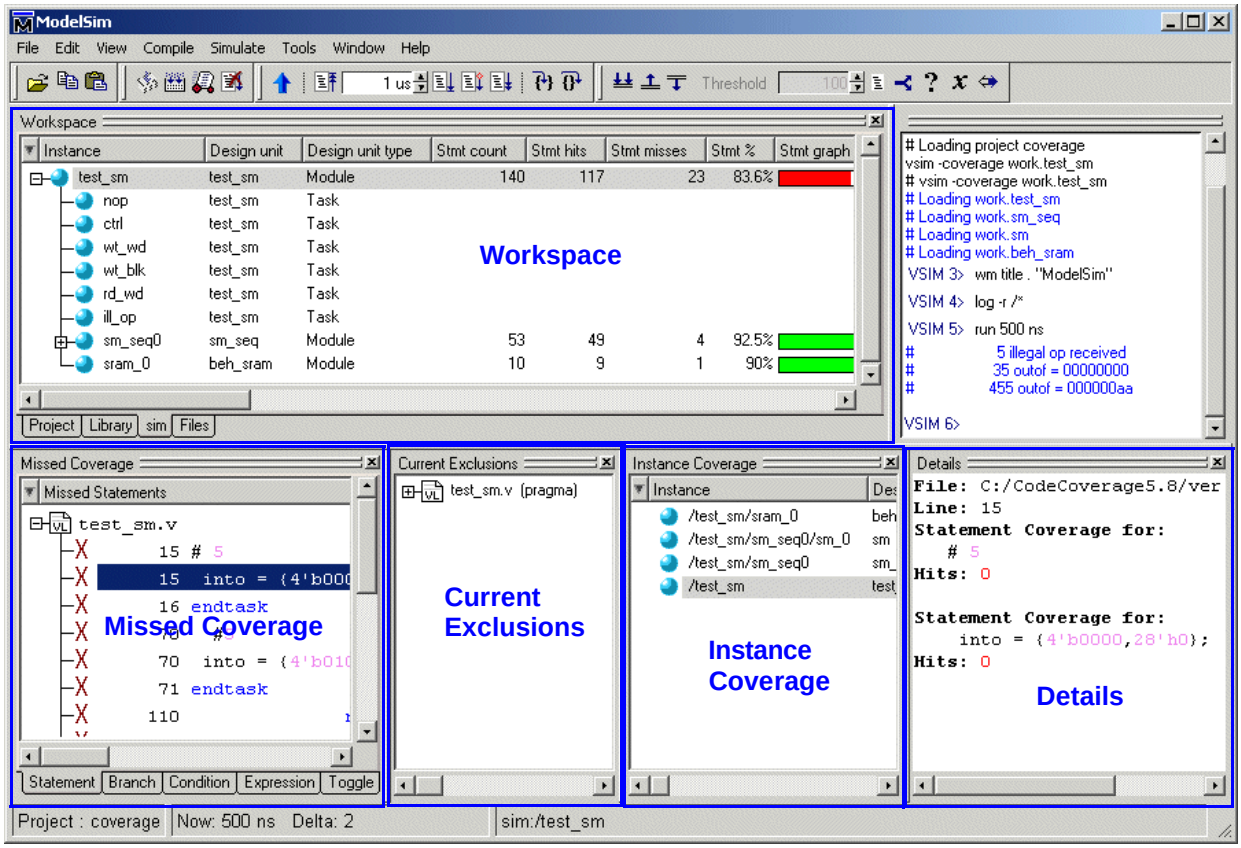
The Configure assertions dialog includes these options:

- **Specific instance**
  Specifies a particular instance whose assertions will be modified by the configuration settings.
- **Recursive**
  Searches for assertions in subregions of the specified instance.
- **All assertions**
  Specifies that all assertions will be modified by the configuration settings.
- **Assertions**
  Enables or disables failure and pass checking for the specified assertions.
- **Logging**
  Enables or disables failure and pass logging for the specified assertions.
- **Limit**
  Determines how many times ModelSim responds to an assertion pass or failure before disabling the assertion. If you select Limited, enter an integer in the Times field.
- **Action**
  Determines what action ModelSim takes when an assertion fails. ’Continue’ logs the assertion failure and proceeds with the simulation; ’Break’ pauses the simulation; ’Exit’ stops and then quits the simulation.

Code coverage panes

When you run simulations with code coverage enabled, a number of panes display in the Main window. These panes dissect and organize the data collected during coverage analysis. Each pane contains context menus (right-click in the pane to access the menus) with commands appropriate to that pane. You can hide and show the panes by selecting **View > Code Coverage**.

For details about using code coverage, see *Chapter 13 - Measuring code coverage* in the *ModelSim User's Manual*.



Workspace pane

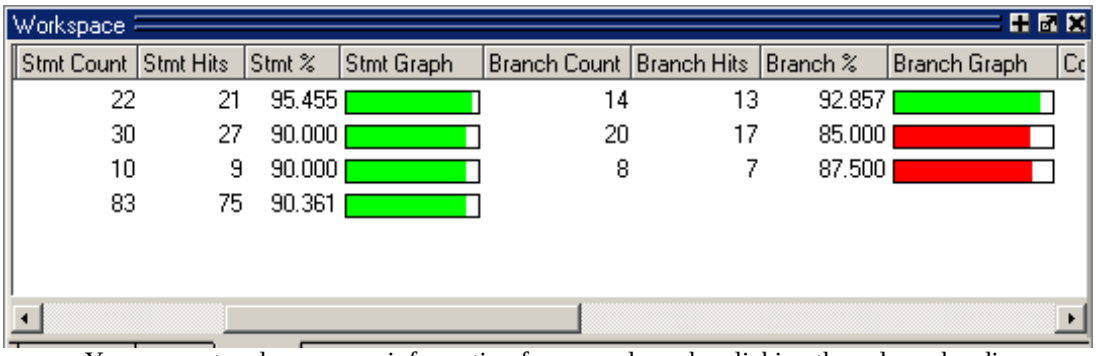
The Workspace pane displays code coverage information in the Files tab and in the structure tabs (e.g., the *sim* tab) that display structure for any datasets being simulated. When coverage is invoked, several columns for displaying coverage data are added to the Workspace pane. You can toggle columns on/off by right-clicking on a column name and

selecting from the context menu that appears. The following code coverage-related columns appear in the Workspace pane:

Column name	Description
Stmt count	in the Files tab, the number of executable statements in each file; in the sim tab, the number of executable statements in each level and all levels under that level
Stmt hits	the number of executable statements that have been executed in the current simulation
Stmt misses	the number of executable statements that were not executed in the current simulation
Stmt %	the current ratio of Stmt hits to Stmt count
Stmt graph	a bar chart displaying the Stmt %; if the percentage is below 90%, the bar is red; 90% or more, the bar is green; you can change this threshold percentage by editing the **PrefCoverage(cutoff)** preference variable
Branch count	in the Files tab, the number of executable branches in each file; in the sim tab, the number of executable branches in each level and all levels under that level
Branch hits	the number of executable branches that have been executed in the current simulation
Branch misses	the number of executable branches that were not executed in the current simulation
Branch %	the current ratio of Branch hits to Branch count
Branch graph	a bar chart displaying the Branch %; if the percentage is below 90%, the bar is red; 90% or more, the bar is green; you can change this threshold percentage by editing the **PrefCoverage(cutoff)** preference variable
Condition rows	in the Files tab, the number of conditions in each file; in the sim tab, the number of conditions in each level and all levels under that level
Condition hits	in the Files tab, the number of times the conditions in a file have been executed; in the sim tab, the number of times the conditions in a level, and all levels under that level, have been executed
Condition misses	in the Files tab, the number of conditions in a file that were not executed; in the sim tab, the number of conditions in a level, and all levels under that level, that were not executed
Condition %	the current ratio of Condition hits to Condition rows

Column name	Description
Condition graph	a bar chart displaying the Condition %; if the percentage is below 90%, the bar is red; 90% or more, the bar is green; you can change this threshold percentage by editing the **PrefCoverage(cutoff)** preference variable
Expression rows	in the Files tab, the number of executable expressions in each file; in the sim tab, the number of executable expressions in each level and all levels subsumed under that level
Expression hits	in the Files tab, the number of times expressions in a file have been executed; in the sim tab, the number of times expressions in a level, and each level under that level, have been executed
Expression misses	in the Files tab, the number of executable expressions in a file that were not executed; in the sim tab, the number of executable expressions in a level, and all levels under that level, that were not executed
Expression %	the current ratio of Expression hits to Expression rows
Expression graph	a bar chart displaying the Expression %; if the percentage is below 90%, the bar is red; 90% or more, the bar is green; you can change this threshold percentage by editing the **PrefCoverage(cutoff)** preference variable
Toggle nodes	the number of points in each instance where the logic will transition from one state to another
Toggle hits	the number of nodes in each instance that have transitioned at least once
Toggle misses	the number of nodes in each instance that have not transitioned at least once
Toggle %	the current ratio of Toggle hits to Toggle nodes
Toggle graph	a bar chart displaying the Toggle %; if the percentage is below 90%, the bar is red; 90% or more, the bar is green; you can change this threshold percentage by editing the **PrefCoverage(cutoff)** preference variable

The diagram below shows a portion of the Workspace window pane with code coverage data displayed.

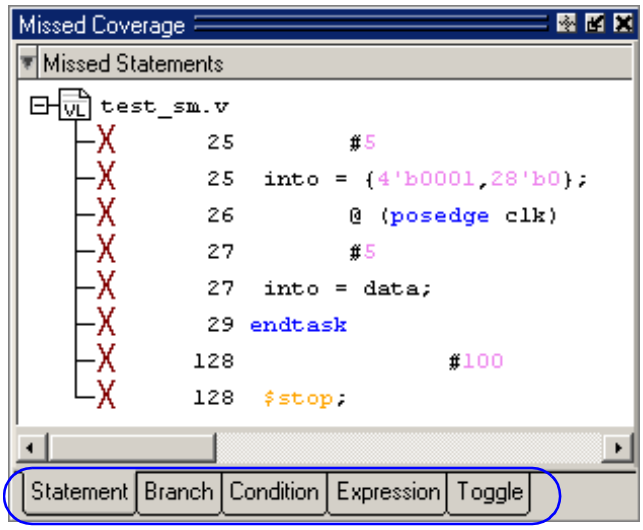


You can sort code coverage information for any column by clicking the column heading. Clicking the column heading again will reverse the order.

Coverage information in the Workspace pane is dynamically linked to the Missed Coverage pane and the Current Exclusions pane. Click the left mouse button on any file in the Workspace pane to display that file's un-executed statements, branches, conditions, expressions, and toggles in the Missed Coverage pane. Lines from the selected file that are excluded from coverage statistics are displayed in the Current Exclusions pane.

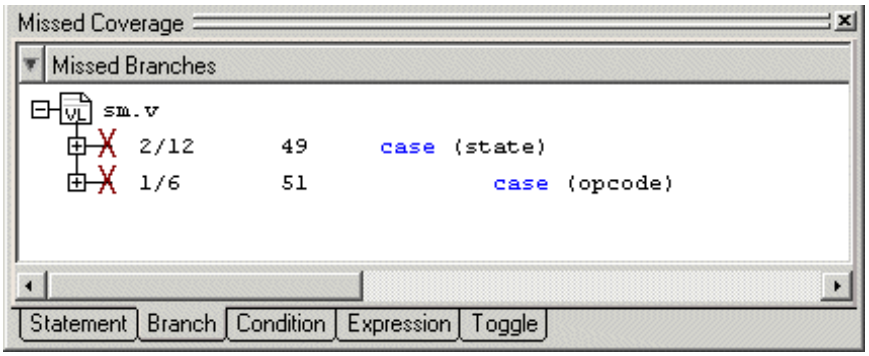
Missed Coverage pane

When you select a file in the Workspace pane, the Missed Coverage pane displays that file's un-executed statements, branches, conditions, and expressions and signals that haven't toggled. The pane includes a tab for each object, as shown below.



Each tab includes a column for the line number and a column for statement, branch, condition, expression, or toggle on that line. The "X" indicates the object was not executed.

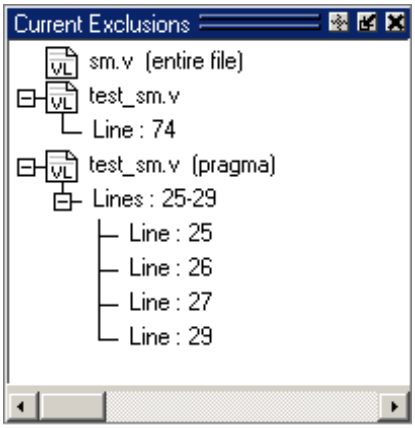
The Branch tab also includes a column for branch code (conditional "if/then/else" and "case" statements). "X_T" indicates that only the true condition of the branch was not executed. "X_F" indicates that only the false condition of the branch was not executed. Fractional numbers indicate how many case statement labels were not executed. For example, if only one of six case labels executed, the Branch tab would indicate "X 1/6."



When you right-click any object in the Statement, Branch, Condition, or Expression tabs you can select **Exclude Selection** or **Exclude Selection for Instance <name>** to exclude the object from coverage statistics and make it appear in the Current Exclusions pane.

Current Exclusions pane

The Current Exclusions pane lists all files and lines that are excluded from coverage statistics. See "Excluding objects from coverage" (UM-347) for more details.

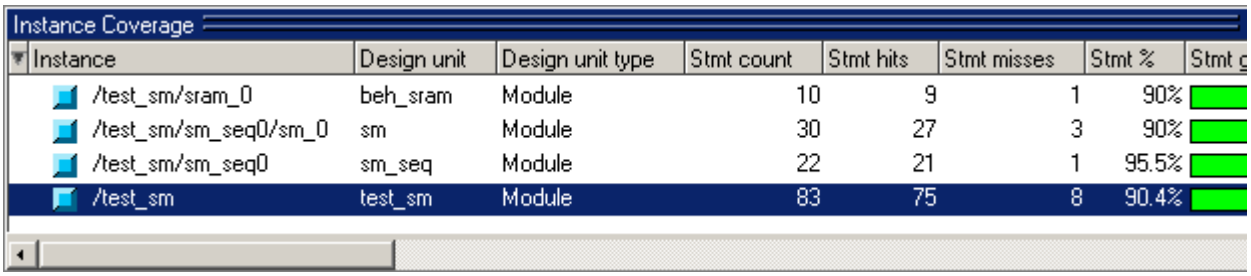


The pane does not display by default. Select **View > Code Coverage > Current Exclusions** to display the it.

Instance Coverage pane

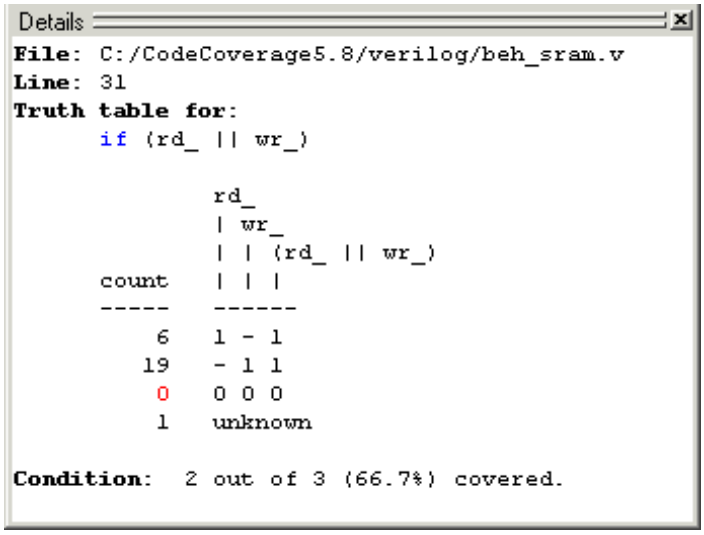
The Instance Coverage pane displays coverage statistics for each instance in a flat, non-hierarchical view. The Instance Coverage pane contains the same code coverage statistics columns as in the "Workspace" (GR-15) pane.

A partial view of the Instance Coverage pane is shown below.

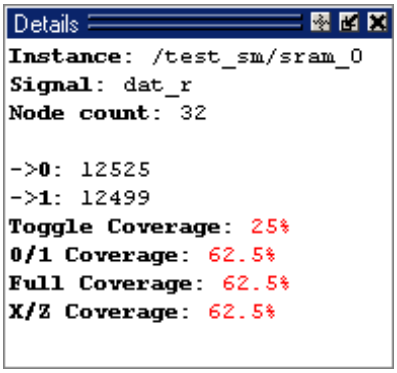


Details pane

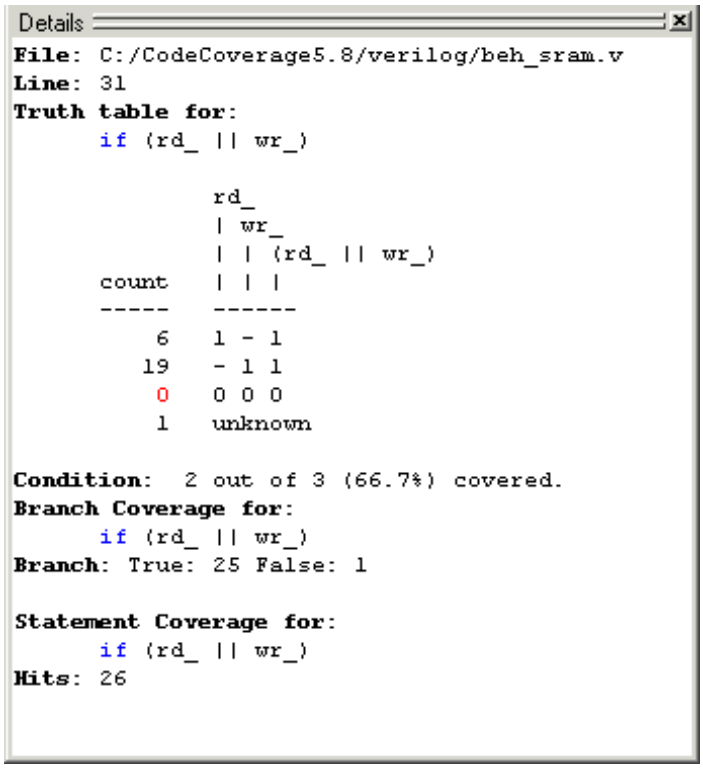
After code coverage is invoked and the simulation is loaded and run, you can turn on the Details pane by selecting **View > Coverage > Details**. The Details pane shows the details of missed coverage. When an object is selected in the Missed Coverage pane, the details of that coverage are displayed in the Details pane. Truth tables will be displayed for condition and expression coverage, as shown here.



Toggle details are displayed as follows:

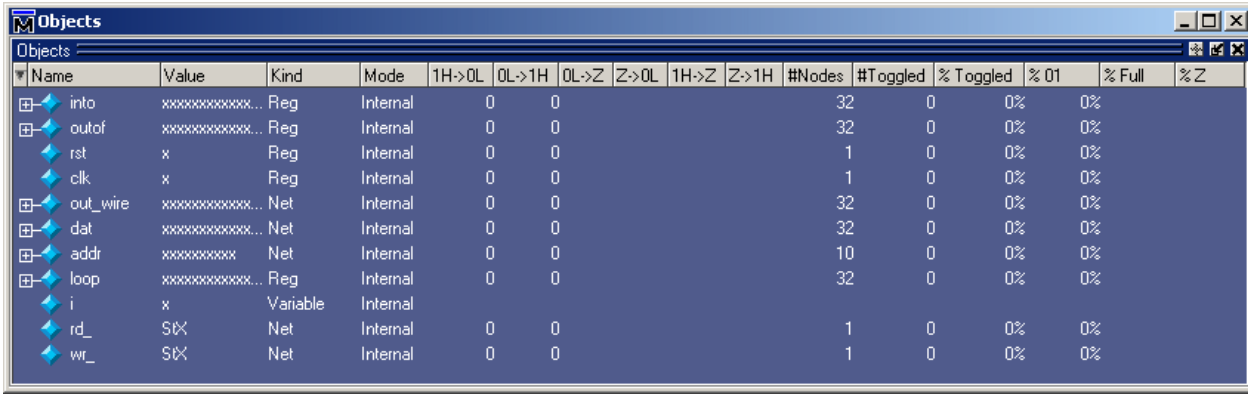


By clicking the left mouse button on the statement Hits column in the Source window, all coverage information for that line will be displayed in the Details pane as shown here:



Objects pane toggle coverage

Toggle coverage data is displayed in the Objects pane in multiple columns, as shown below. There is a column for each of the six transition types.



Right click any column name to toggle a column on or off.

The following table provides a description of the available columns:

Column name	Description
Name	the name of each object in the current region
Value	the current value of each object
Kind	the object type
Mode	the object mode (internal, in, out, etc.)
1H -> 0L	the number of times each object has transitioned from a 1 or a High state to a 0 or a Low state
0L -> 1H	the number of times each object has transitioned from a 0 or a Low state to 1 or a High state
0L -> Z	the number of times each object has transitioned from a 0 or a Low state to a high impedance (Z) state
Z -> 0L	the number of times each object has transitioned from a high impedance state to a 0 or a Low state
1H -> Z	the number of times each object has transitioned from a 1 or a High state to a high impedance state
Z -> 1H	the number of times each object has transitioned from a high impedance state to 1 or a High state
# Nodes	the number of scalar bits in each object
# Toggled	the number of nodes that have transitioned at least once
% Toggled	the current ratio of the # Toggled to the # Nodes for each object

Column name	Description
% 01	the percentage of **1H -> 0L** and **0L -> 1H** transitions that have occurred (transitions in the first two columns)
% Full	the percentage of all transitions that have occurred (all six columns)
% Z	the percentage of **0L -> Z**, **Z -> 0L**, **1H -> Z**, and **Z -> 1H** transitions that have occurred (last four columns)

Code coverage toolbar

When you simulate with code coverage enabled, the following toolbar is added to the Main window.

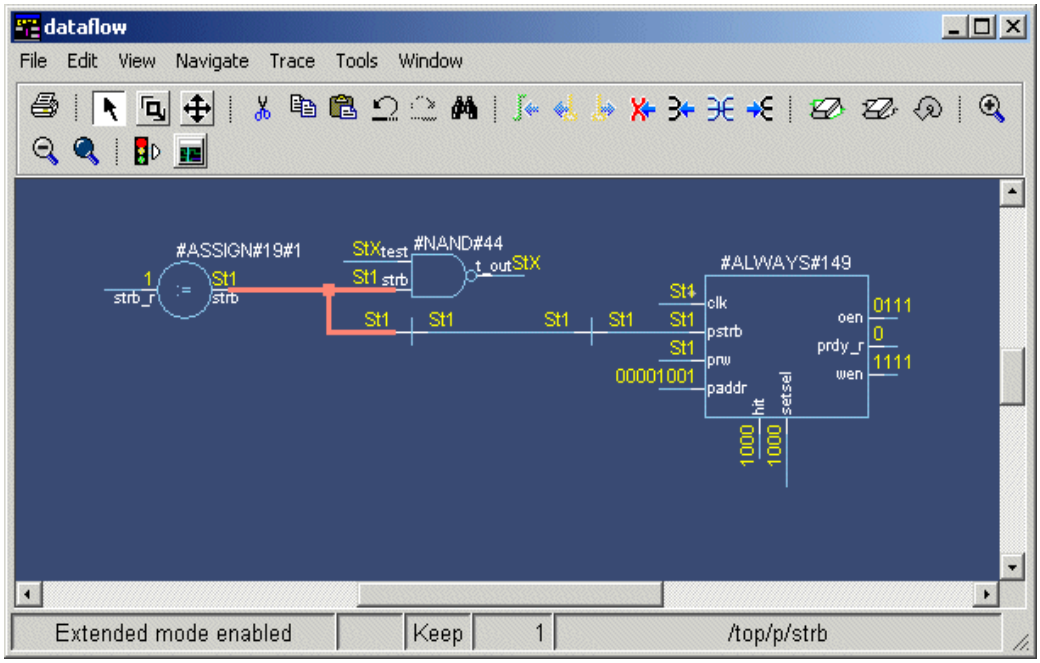
The toolbar has the following buttons:.

Covfilter toolbar buttons	
Button	
	Enable Filtering enables display filtering of coverage statistics in the Workspace and Instance Coverage panes of the Main window
	Threshold above displays all coverage statistics above the Filter Threshold for selected columns
	Threshold below displays all coverage statistics below the Filter Threshold for selected columns
	Filter Threshold specifies the display coverage percentage for the selected coverage columns
	Statement applies the display filter to all Statement coverage columns in the Workspace and Instance Coverage panes of the Main window
	Branch applies the display filter to all Branch coverage columns in the Workspace and Instance Coverage panes of the Main window
	Condition applies the display filter to all Condition coverage columns in the Workspace and Instance Coverage panes of the Main window
	Expression applies the display filter to all Expression coverage columns in the Workspace and Instance Coverage panes of the Main window
	Toggle applies the display filter to all Toggle coverage columns in the Workspace and Instance Coverage panes of the Main window

Dataflow window

The Dataflow window allows you to explore the "physical" connectivity of your design; to trace events that propagate through the design; and to identify the cause of unexpected outputs.

Note: ModelSim versions operating without a dataflow license feature have limited Dataflow functionality. Without the license feature, the window will show only one process and its attached signals or one signal and its attached processes. Contact Model Technology sales for more information if you don't have a dataflow license feature.



Objects you can view

The Dataflow window displays processes; signals, nets, and registers; and interconnect. The window has built-in mappings for all Verilog primitive gates (i.e., AND, OR, etc.). For components other than Verilog primitives, you can define a mapping between processes and built-in symbols. See "Symbol mapping" (UM-313) for details.

Note: You cannot view SystemC objects in the Dataflow window.

Dataflow window menu bar

The following commands are available from the Dataflow window menu bar. Many of the commands are also available from the context menu (click right or 3rd mouse button).

File menu

New Window	create a new Dataflow window
Page setup	configure page formatting for printing
Print	print the current view of the Dataflow window (Windows only)
Print Postscript	print/save the current view of the Dataflow window to a postscript device/file
Close	close the Dataflow window; note that this erases whatever is currently displayed in the window

Edit menu

Undo	undo the last action
Redo	redo the last undone action
Cut	cut the selected object(s)
Copy	copy the selected object(s)
Paste	paste the previously cut or copied object(s) into the display
Erase selected	clear the selected object from the window
Select all	select all objects in the window
Unselect all	deselect all currently selected objects
Erase highlight	remove green highlighting from interconnect lines
Erase all	clear all objects from the window
Regenerate	clear and redraw the display using an optimal layout
Find	search for an instance or signal
Find next	search for the next occurrence of an instance or signal

View menu

Show Wave	open the embedded wave viewer pane
Select	set left mouse button to select mode and middle mouse button to zoom mode
Zoom	set left mouse button to zoom mode and middle mouse button to pan mode
Pan	set left mouse button to pan mode and middle mouse button to zoom mode
Default	set mouse to default mode

Navigate menu

Expand net to drivers	display driver(s) of the selected signal, net, or register
Expand net to readers	display reader(s) of the selected signal, net, or register
Expand net	display driver(s) and reader(s) of the selected signal, net, or register
Expand to design inputs	display the top-level driver of the net, which will most likely be in a testbench or in the top entity or module
Expand to hierarchy inputs	display the primary driver (port) of the net within its level of hierarchy
Hide selected	remove the selected component and all other components from the same region and replace them with a single component representing that region
Show selected	expand the selected component to show all underlying components
View region	clear the window and display all signals from the current region
Add region	display all signals from the current region without first clearing the window
View all nets	clear the window and display all signals from the entire design
Add ports	add port symbols to the port signals in the current region

Trace menu

TraceX™	step back to the last driver of an unknown (X) value
ChaseX™	jump to the source of an unknown (X) value
TraceX Delay	step back in time to the last driver of an unknown (X) value
ChaseX Delay	jump back in time to the point where the output value transitions to X
Trace next event	move the next event cursor to the next input event driving the selected output
Trace event set	jump to the source of the selected input event
Trace event reset	return the next event cursor to the selected output

Tools menu

Load built-in symbol map	load a .bsm file for mapping symbol instances; see "Symbol mapping" (UM-313)
Load symlib library	load a user-defined symbol library
Create symlib index	create an index for a user-defined symbol library
Options	configure Dataflow window preferences

Window menu

The Window menu is identical in all windows. See "Window menu" (GR-32) for a description of the commands.

Dataflow window toolbar

The buttons on the Dataflow window toolbar are described below.

Button	Menu equivalent
Print print the current view of the Dataflow window	File > Print (Windows) File > Print Postscript (UNIX)
Select mode set left mouse button to select mode and middle mouse button to zoom mode	View > Select
Zoom mode set left mouse button to zoom mode and middle mouse button to pan mode	View > Zoom
Pan mode set left mouse button to pan mode and middle mouse button to zoom mode	View > Pan
Cut cut the selected object(s)	Edit > Cut
Copy copy the selected object(s)	Edit > Copy
Paste paste the previously cut or copied object(s)	Edit > Paste
Undo undo the last action	Edit > Undo
Redo redo the last undone action	Edit > Redo
Find search for an instance or signal	Edit > Find

Button	Menu equivalent
Trace input net to event move the next event cursor to the next input event driving the selected output	Trace > Trace next event
Trace Set jump to the source of the selected input event	Trace > Trace event set
Trace Reset return the next event cursor to the selected output	Trace > Trace event reset
Trace net to driver of X step back to the last driver of an unknown value	Trace > TraceX
Expand net to all drivers display driver(s) of the selected signal, net, or register	Navigate > Expand net to drivers
Expand net to all drivers and readers display driver(s) and reader(s) of the selected signal, net, or register	Navigate > Expand net
Expand net to all readers display reader(s) of the selected signal, net, or register	Navigate > Expand net to readers
Erase highlight clear the green highlighting which identifies the path you've traversed through the design	Edit > Erase highlight
Erase all clear the window	Edit > Erase all
Regenerate clear and redraw the display using an optimal layout	Edit > Regenerate

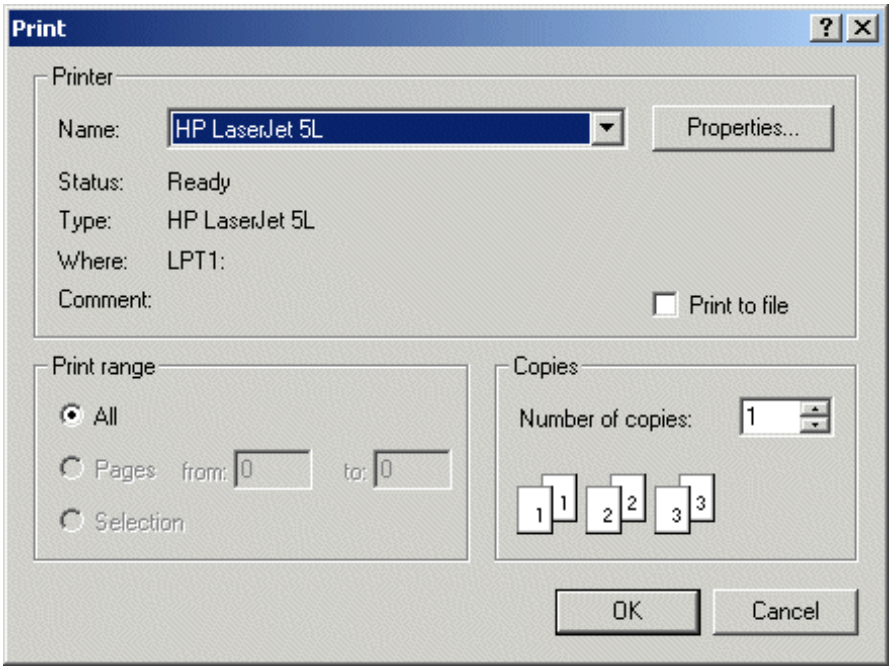
Button	Menu equivalent
Zoom In zoom in by a factor of two from current view	none
Zoom Out zoom out by a factor of two from current view	none
Zoom Full zoom out to show all components in the window	none
Stop Drawing halt any drawing currently happening in the window	none
Show Wave display the embedded wave viewer pane	View > Show Wave

Dataflow window dialogs

This section describes the dialogs that are accessed via the Dataflow window menu bar. The dialogs are listed in the order in which they appear on the menus, top-to-bottom and left-to-right (i.e., starting with the File menu and progressing across the menu bar).

Print dialog

Purpose	Menu command	Additional information
Print the contents of the Dataflow window	**File > Print**	"Printing and saving the display" (UM-310)"Printing on Windows platforms" (UM-311)



The Print dialog includes these options:

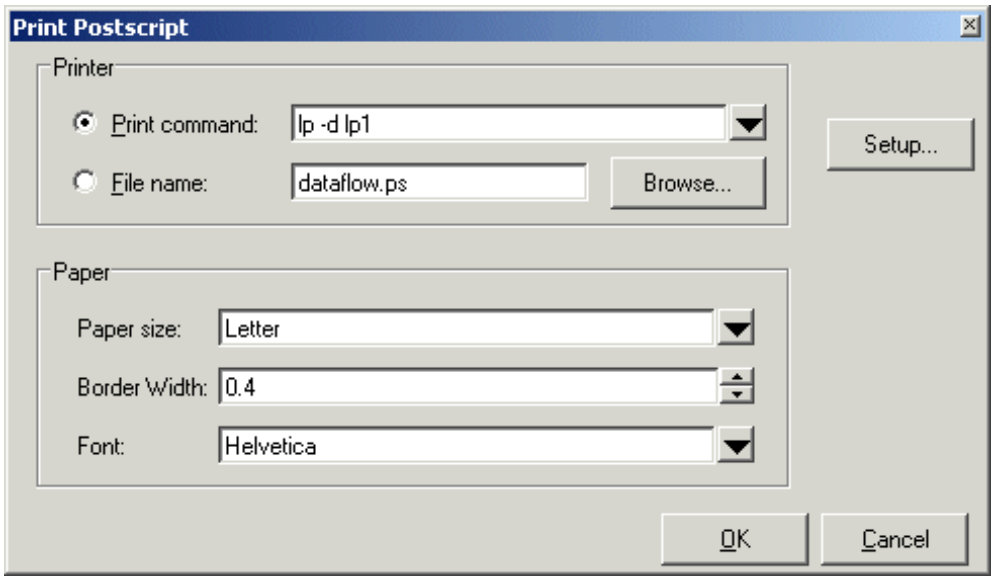
Printer

- **Name**
  Choose the printer from the drop-down menu. Set printer properties with the *Properties* button.
- **Status**
  Indicates the availability of the selected printer.
- **Type**
  Printer driver name for the selected printer. The driver determines what type of file is output if "Print to file" is selected.

- **Where**
  The printer port for the selected printer.
- **Comment**
  The printer comment from the printer properties dialog.
- **Print to file**
  Make this selection to print the display to a file instead of a printer. The printer driver determines what type of file is created. Postscript printers create a Postscript (.ps) file, non-Postscript printers create a .prn or printer control language file. To create an encapsulated Postscript file (.eps) use the **File > Print Postscript** menu selection.

Print Postscript dialog

Purpose	Menu command	Additional information
Print/save a postscript file of the Dataflow window	**File > Print Postscript**	"Printing and saving the display" (UM-310)



The Print Postscript dialog includes these options:

Printer

- **Print command**
  Enter a UNIX print command to print the display in a UNIX environment.
- **File name**
  Enter a filename for the encapsulated Postscript (.eps) file to create; or browse to a previously created .eps file and use that filename.

Paper

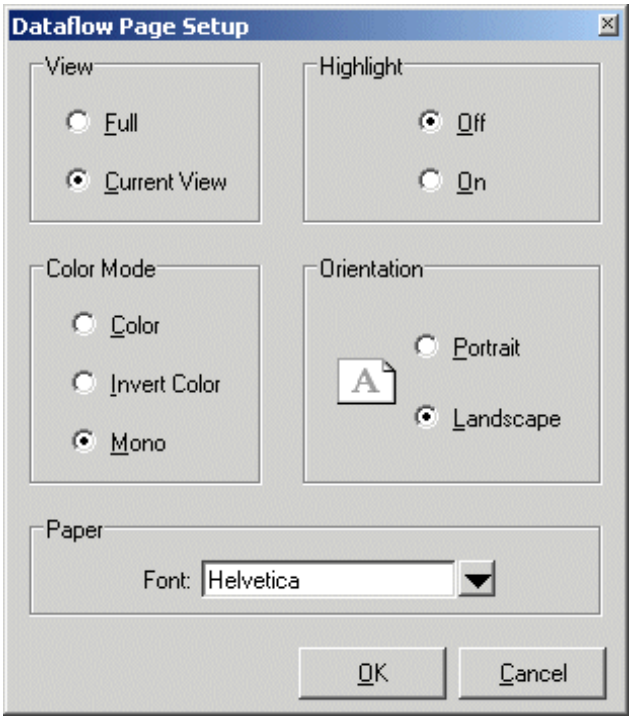
- **Paper size**
  Select the paper size used by the printer.
- **Border width**
  Specify the border in inches.
- **Font**
  Specify the font to use for printing.

Setup button

See "Dataflow Page Setup dialog" (GR-138).

Dataflow Page Setup dialog

Purpose	Menu command	Additional information
Set up page layout for printing	**File** > **Page setup**	"Configuring page setup" (UM-312)

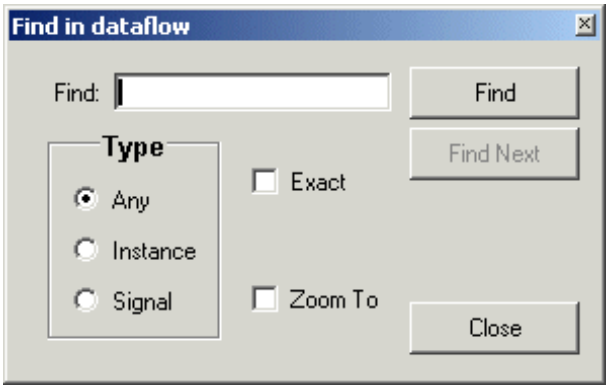


The Dataflow Page Setup dialog includes these options:

- **View**
  Specifies **Full** (everything in the window) or **Current View** (only that which is visible).
- **Highlight**
  Specifies that highlighting (see "Tracking your path through the design" (UM-303)) is **On** or **Off**.
- **Color Mode**
  Specifies **Color** (256 colors), **Invert Color** (gray-scale) or **Mono** (monochrome) color mode.
- **Orientation**
  Specifies **Landscape** (horizontal) or **Portrait** (vertical) orientation.
- **Paper**
  Specifies the font to use for printing (Windows only).

Find in dataflow dialog

Purpose	Menu command	Additional information
Locate a signal or process	**Edit > Find**	"Finding objects by name in the Dataflow window" (UM-309)



The Find in dataflow dialog includes the following options:

- **Find**
  Enter the object name you want to locate.
- **Type**
  Specify whether to search Instance, Signals, or both (Any) for the object name.
- **Exact**
  Find only those objects that match your search exactly. For example, searching for "clk" without **Exact** will find */top/clk* and *clk1*.
- **Zoom To**
  If checked the window zooms in on the located object.

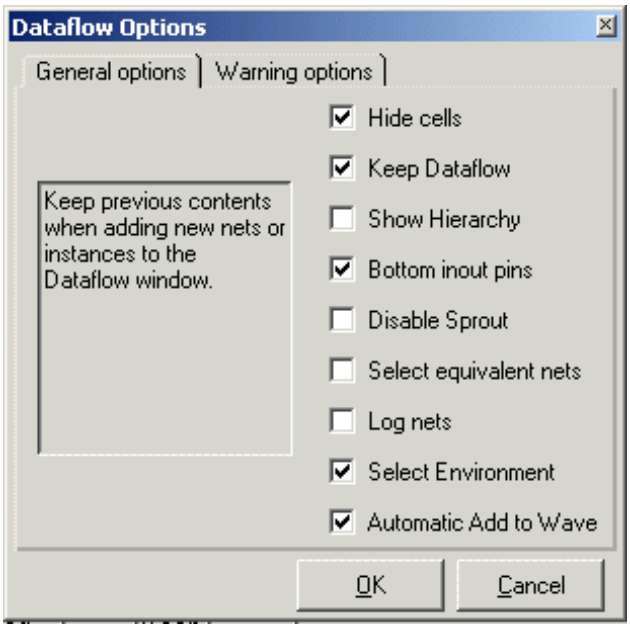
You can continue searching using the Find Next button.

Dataflow Options dialog

Purpose	Menu command	Additional information
Configure Dataflow window options	**Tools > Options**	"Configuring window options" (UM-315)

The settings affect only the current session.

General options tab

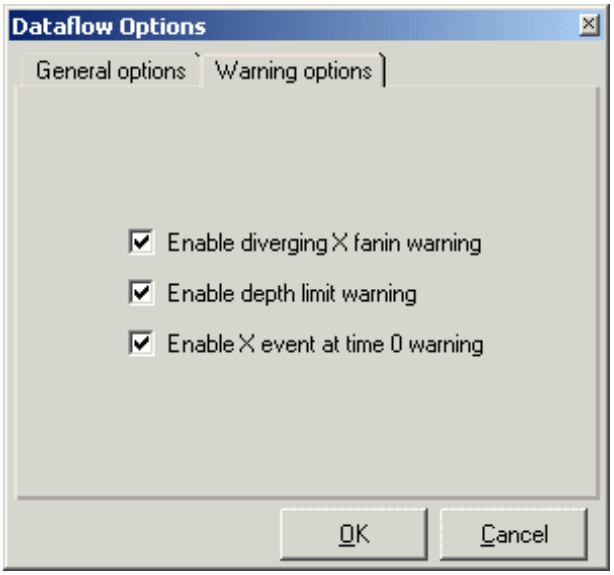


The General options tab includes these options:

- **Hide Cells**
  By default the Dataflow window automatically hides instances that have either 'celldefine, VITAL_LEVEL0, or VITAL_LEVEL1 attributes. Unchecking this disables automatic cell hiding.
- **Keep Dataflow**
  Keeps previous contents when adding new signals or processes to the window.
- **Show Hierarchy**
  Displays connectivity using hierarchical references. Note that selecting this will erase the current contents of the window.
- **Bottom inout pins**
  Places inout pins on the bottom of components rather than on the right with output pins.
- **Disable Sprout**
  Displays only the selected signal or process with its immediate fanin/fanout. Configures window to behave like the Dataflow window of versions prior to 5.6.

- **Select equivalent nets**
  If the object you select traverses hierarchy, then ModelSim selects all connected objects across the hierarchy.
- **Log nets**
  Logs signals when they are added to the window.
- **Select Environment**
  Updates the Structure, Objects, and Source windows to reflect the net selected in the Dataflow window.
- **Automatic Add to Wave**
  Adds signals automatically to the Wave pane or window when executing ChaseX or TraceX.

Warning options tab



The Warning options tab includes these options:

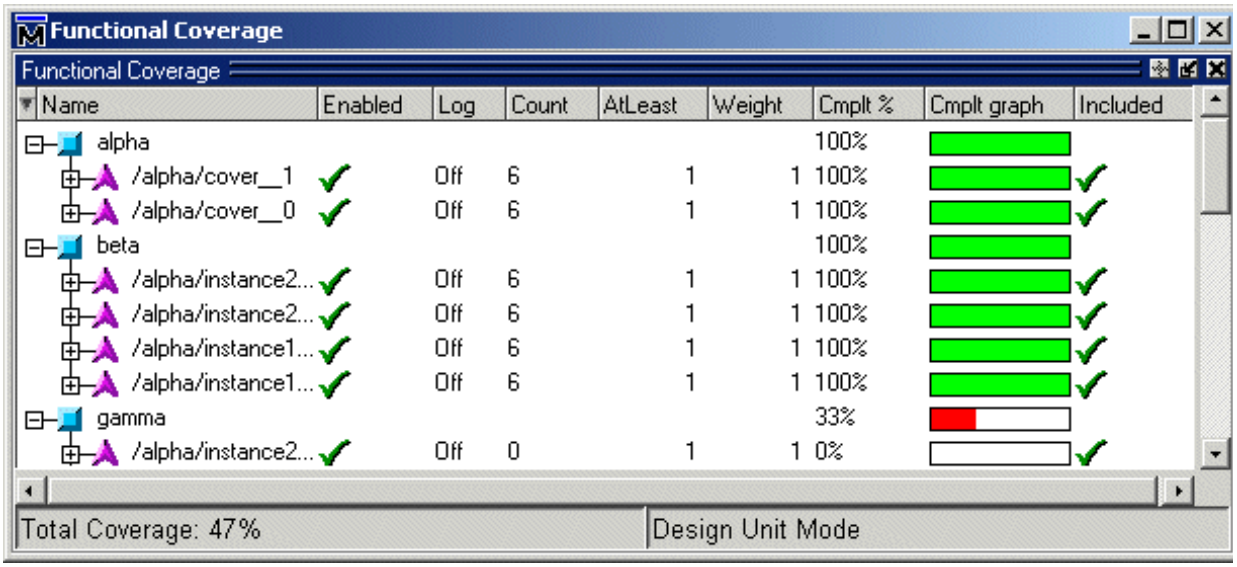
- **Enable diverging X fanin warning**
  Enables the warning message, "ChaseX: diverging X fanin. Reduce the selection list and try again."
- **Enable depth limit warning**
  Enables the warning message, "ChaseX: Stop because depth limit reached! Possible loop?"
- **Enable X event at time 0 warning**
  Enables the warning message, ""Driving X event at time 0."

Customize Toolbar dialog

This dialog is the same for all windows. See "Customize Toolbar dialog" (GR-106) under the Main window for details.

Functional Coverage pane

The Functional Coverage pane displays a list of functional coverage directives in the current region (which is selected via the structure pane in the Main window). See *Chapter 15 - Functional coverage with PSL and ModelSim* in the *ModelSim User's Manual* for more information on functional coverage.



The Functional Coverage pane lists all embedded and external directives that were successfully compiled and simulated during the current session. The plus sign ('+') to the left of the Name column lets you expand the directive hierarchy to show its elements (properties, sequences, clocks, and HDL signals).

The pane can be sorted by design unit or by cover directive. Select **Tools** > **Functional Coverage** > **Design Unit Mode** to change the sorting.

Functional Coverage window columns

The Functional Coverage window includes the columns described below, though only seven are displayed by default. See "Hiding/showing columns in the Assertions pane" (GR-112) for details on how to show or hide particular columns.

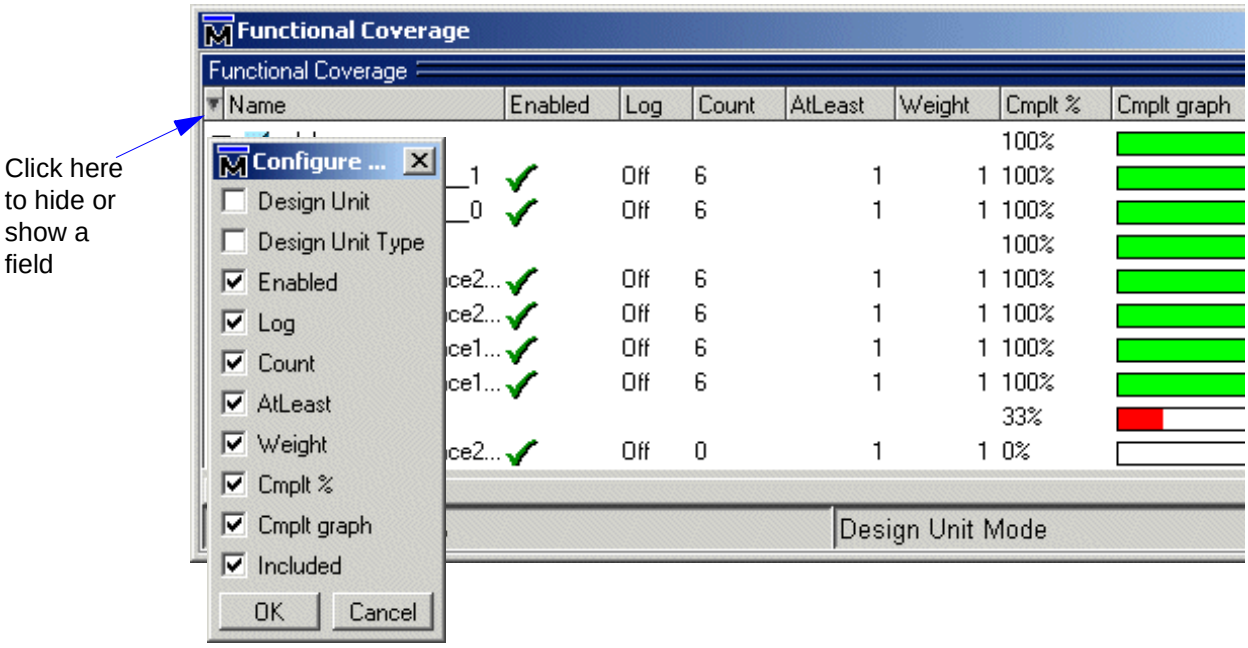
- The **Name** column lists directive names and design units. Also, any signals referenced in a directive are included in the hierarchy. See "Understanding assertion names" (UM-370) for details on how ModelSim names directives.
- The **Design Unit** column identifies the design unit to which the directive is bound. Not displayed by default.
- The **Design Unit Type** column lists the HDL type of the design unit. Not displayed by default.
- The **Enabled** column shows a green checkmark when a directive is enabled and a red X when a directive is disabled.
- The **Log** column indicates whether data for the directive is currently being added to the functional coverage database.

- The **Count** column shows how many times a directive has "fired" during the current simulation.
- The **AtLeast** column shows how many times a directive has to fire to be considered 100% covered. See "Configure cover directives dialog" (GR-149) for more information.
- The **Weight** column shows the weighting factor that has been applied to the directive. See "Configure cover directives dialog" (GR-149) for more information.
- The **Cmplt%** column shows the coverage percentage for a directive. The percentage is the lesser of 100% or Count/AtLeast.
- The **Cmplt graph** column shows a graphical bar chart of the completion percentage. Directives with 100% coverage are displayed in green.
- The **Included** column indicates whether the directive is included in aggregate statistics and reports.

You can also view this same information in textual format using the **fcover report** command (CR-173).

Hiding/showing columns in the Functional Coverage window

You can hide or show any of the columns in the Functional Coverage window. Click the drop-down arrow on the left-hand side of the dialog and select a column name.



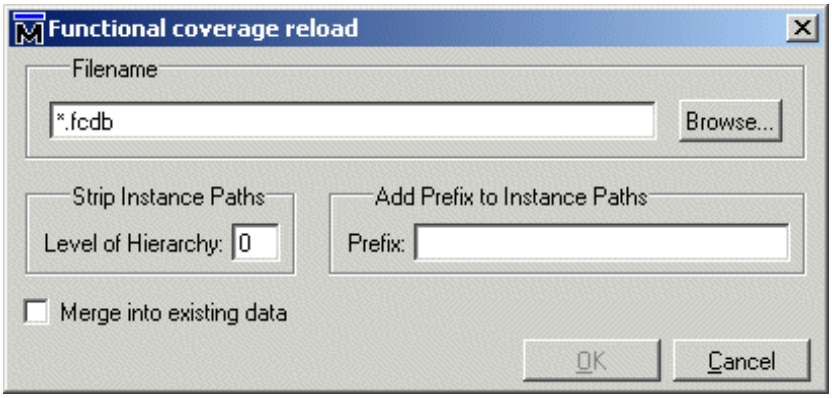
The selection acts as a toggle–select it once to hide a column; select it again to show the column.

Functional Coverage pane dialogs

This section describes the functional coverage-related dialogs that are accessed via the Main window menu bar. Not all dialogs are documented (e.g., Tools > Functional Coverage > Save).

Functional coverage reload dialog

Purpose	Menu command	Additional information
Load a saved coverage database	**Tools > Functional Coverage > Reload**	"Reloading/merging functional coverage data" (UM-396)

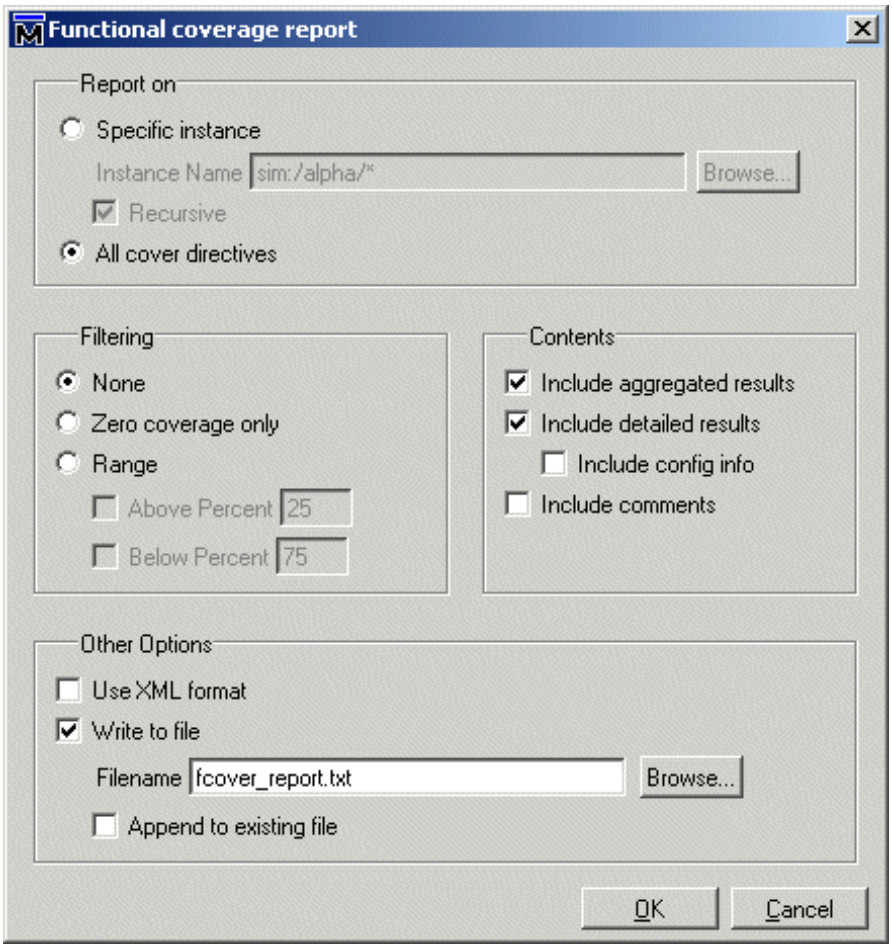


The Functional coverage reload dialog includes these options:

- **Filename**
  The file to reload. Must be a functional coverage database previously saved with the **File > Save** command or the **fcover save** command (CR-175).
- **Strip Instance Paths**
  A number of levels of hierarchy to strip from all instance pathnames. Use this to delete levels of hierarchy when merging statistics from different runs of the same design that were performed in different contexts (e.g., block simulation vs. chip-level simulation vs. system simulation).
- **Add Prefix to Instance Paths**
  A prefix to add to all instance paths. Use this to add levels of hierarchy when merging statistics from different runs of the same design that were performed in different contexts (e.g., block simulation vs. chip-level simulation vs. system simulation).
- **Merge into existing data**
  Merges the file into the current functional coverage database. If left unchecked, the loaded file replaces the current database.

Functional coverage report dialog

Purpose	Menu command	Additional information
Create ASCII report of coverage statistics	**Tools > Functional Coverage > Report**	"Reporting functional coverage statistics" (UM-391)



The Functional coverage filter dialog includes these options:

- **Specific instance**
  An instance for which statistics will be reported.
- **Recursive**
  Searches for directives in subregions of the specified instance.
- **All cover directives**
  Specifies that all directives in the current design will be included in the report.
- **Filtering**
  Filters report data. You can report on all directives, directives with zero coverage, or directives that had a coverage percentage within a specific range.

Contents

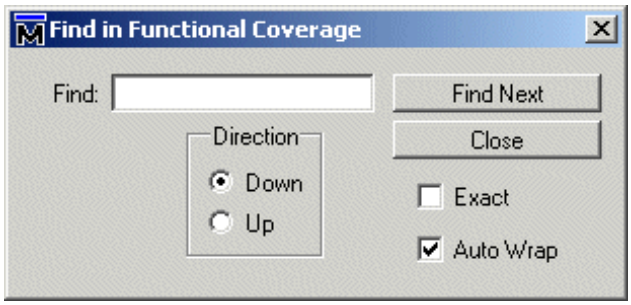
- **Include aggregated results**
  Outputs aggregated statistics when reporting on all directives or directives in a specified instance. See "Understanding aggregated statistics" (UM-393) for more details.
- **Include detailed results**
  Includes details (e.g., "count") associated with each directive.
- **Include config info**
  Includes the current configuration settings for each directive.
- **Include comments**
  Includes any comment meta-data associated with each directive.

Other Options

- **Use XML format**
  Outputs the report in XML format. See "Formatting output in XML" (UM-392).
- **Write to file**
  Writes output to the specified file. If unchecked, the results are output to the Main window transcript.
- **Append to existing file**
  Appends output to the specified file. If unchecked, the output will overwrite the specified file if it already exists.

Find in Functional Coverage dialog

Purpose	Menu command	Additional information
Locate text in the Functional Coverage window	**Edit > Find**	NA



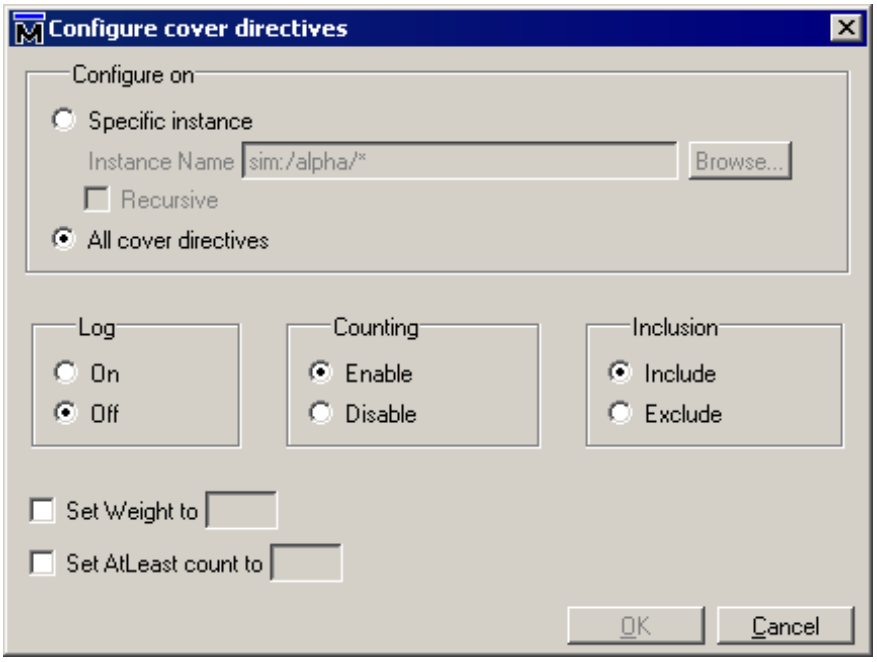
You must activate the Functional Coverage pane by clicking in it before this dialog will be available.

The Find in Functional Coverage dialog includes these options:

- **Find**
  The text you want to locate.
- **Direction**
  Specify the direction to begin searching.
- **Exact**
  Check Exact if you only want to find objects that match your search exactly. For example, searching for "instance" without Exact will find */instance1/cover__1* and */instance2/cover__1*.
- **Auto Wrap**
  Check Auto Wrap to continue the search at the top or bottom of the list.

Configure cover directives dialog

Purpose	Menu command	Additional information
Configure coverage directives	**Tools > Functional Coverage > Configure**	"Configuring functional coverage directives" (UM-386)



The Configure cover directives dialog includes these options:

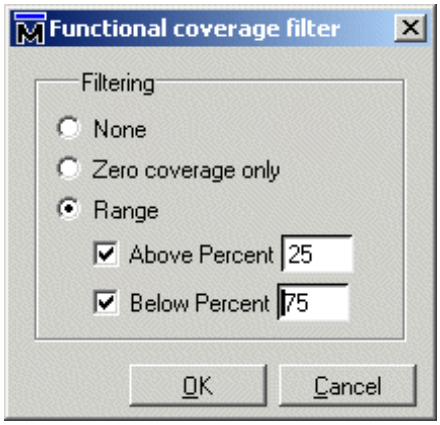
- **Specific instance**
  Specifies a particular instance whose directives will be modified by the configuration settings.
- **Recursive**
  Searches for directives in subregions of the specified instance.
- **All cover directives**
  Specifies that all directives will be modified by the configuration settings.
- **Log**
  Turns on/off logging of directive counts to the Transcript pane. Even with logging off, coverage counts are incremented in the underlying database for the specified coverage directive(s).
- **Counting**
  Enables/disables counting for the specified coverage directive(s). Disabled directives still count toward overall coverage if they had coverage events prior to being disabled.
- **Inclusion**
  Includes/excludes the specified coverage directive(s). Excluded directives will still show

up in the Functional Coverage pane but they do not count toward coverage totals nor do they show up in reports.

- **Set Weight to**
  Applies a weighting to the specified coverage directive(s). Enter a natural integer (>= 0). See "Weighting coverage directives" (UM-387) for more information.
- **Set AtLeast count to**
  Determines the number of times a directive must fire to be considered 100% covered. See "Choosing "AtLeast" counts" (UM-387).

Functional coverage filter dialog

Purpose	Menu command	Additional information
Filter display of directives	**Tools > Functional Coverage > Filter**	"Filtering data in the pane" (UM-388)



The Functional coverage filter dialog includes these options:

- **None**
  All directives are displayed.
- **Zero coverage only**
  Only directives with zero coverage are displayed.
- **Range**
  Specify percentage(s) that define a range from which you want directives displayed.

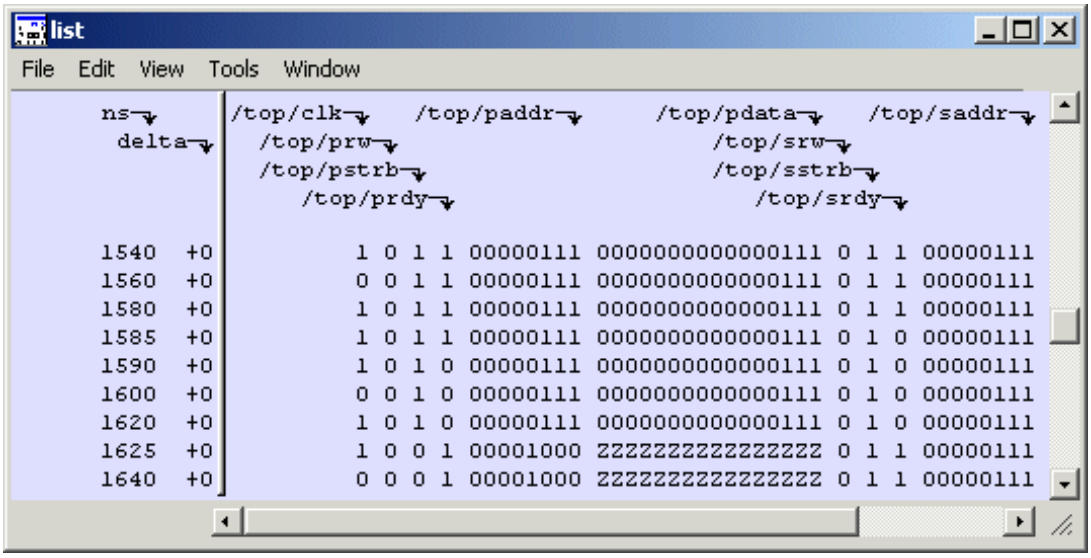
Note that filtering does not affect the gathering of data nor the calculation of aggregated statistics. It merely affects the data display.

Customize Toolbar dialog

This dialog is the same for all windows. See "Customize Toolbar dialog" (GR-106) under the Main window for details.

List window

The List window displays the results of your simulation run in tabular format. The window is divided into two adjustable panes, which allow you to scroll horizontally through the listing on the right, while keeping time and delta visible on the left.



Objects you can view

The following type of objects can be viewed in the List window:

VHDL

signals, aliases, process variables, and shared variables

Verilog

nets, registers, and variables

SystemC

primitive channels and ports

Comparisons

comparison objects; see "Waveform Compare" (UM-270) for more information

Virtuals

Virtual signals and functions

PSL assertions

(indicated by a magenta triangle)
see "Viewing assertions in the Wave window" (UM-382)

List window menu bar

The following menu commands are available from the List window menu bar. Some of the commands are also available on a context menu accessed by clicking the right mouse button on an entry in the right-hand pane.

File menu

New Window	create another instance of the List window
Open	provides these options: Dataset – open a dataset Format – run a List window format DO file previously saved with Save Format
Save	provides these options: Dataset – save the current simulation to a WLF file Format – save the current List window display and signal preferences to a DO (macro) file; running the DO file will reformat the List window to match the display as it appeared when the DO file was created
Write List	save the List window data to a text file in one of three formats; see "Saving List window data to a file" (UM-263) for details
Close	close this copy of the List window

Edit menu

Cut	cut the selected object field from the listing; see "Formatting objects in the List window" (UM-259)
Copy	copy the selected object field
Paste	paste the previously cut or copied object to the left of the currently selected object
Delete	delete the selected object field
Select All	select all signals in the List window
Unselect All	deselect all signals in the List window
Add Marker	add a time marker at the currently selected line
Delete Marker	delete the selected marker from the listing
Find	find the specified object label within the List window

Search	search the List window for a specified value, or the next transition for the selected signal

View menu

Signal Properties	set label, radix, trigger on/off, and field width for the selected object
Goto	choose the time marker to go to from a list of current markers

Tools menu

Combine Signals	combine the selected objects into a user-defined bus; keep copies of the original objects rather than moving them; see "Combining objects/creating busses" (UM-264)
Window Preferences	set display properties for all objects in the window: delta settings, trigger on selection, strobe period, label size, and dataset prefix

Window menu

The Window menu is identical in all windows. See "Window menu" (GR-32) for a description of the commands.

The List window context menu

The following commands are available by clicking the right mouse button on an entry in the right-hand pane:

Examine	display the value of the object at the time selected
Annotate Diff	Add a note to explain a comparison difference. See "Waveform Compare" (UM-270) for further information.
Ignore Diff	Disregard the selected comparison difference. See "Waveform Compare" (UM-270) for further information.
Add Marker	add a time marker at the currently selected line
Delete Marker	delete the selected marker from the listing

List window dialogs

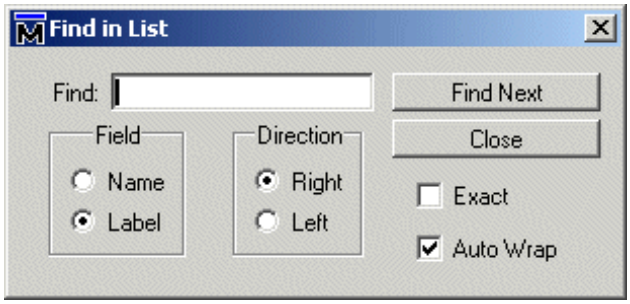
This section describes the dialogs that are accessed via the List window menu bar. The dialogs are listed in the order in which they appear on the menus, top-to-bottom and left-to-right (i.e., starting with the File menu and progressing across the menu bar). Not all dialogs are documented (e.g., File > Write List).

Open Dataset

This is the same dialog as accessed via the Main window File menu. See "Open File dialog" (GR-39).

Find in .list dialog

Purpose	Menu command	Additional information
Locate object names	**Edit > Find**	"Finding signal names" (UM-251)

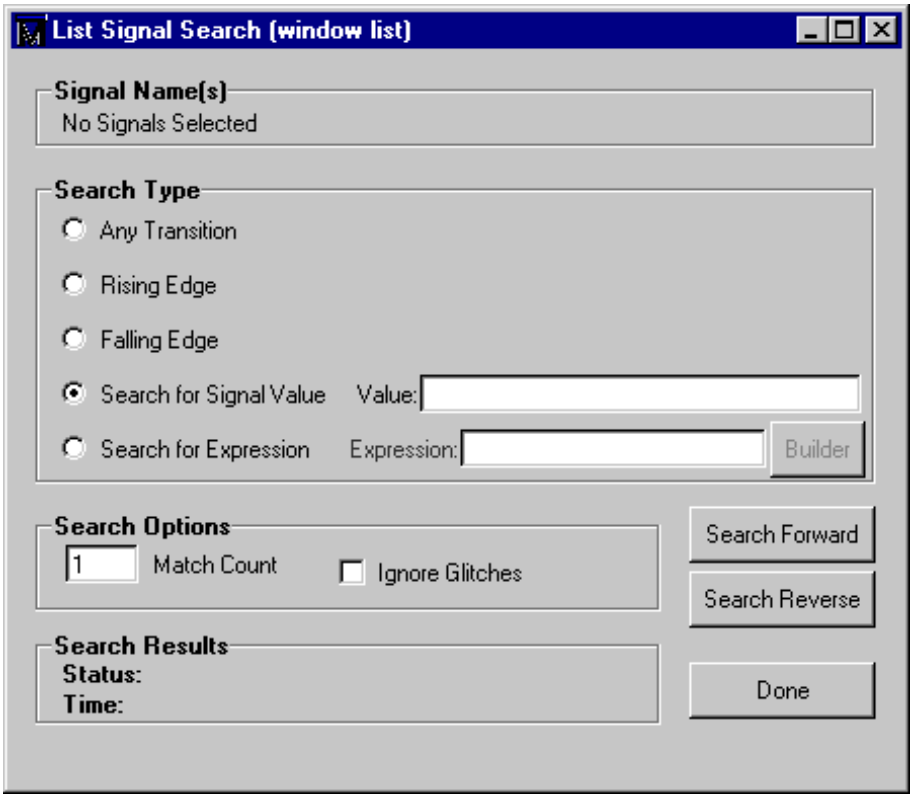


The Find in .list dialog includes these options:

- **Find**
  The text you want to locate.
- **Field**
  Specify **Name** to search the real pathnames of the objects or **Label** to search their assigned names (see "Setting List window display properties" (UM-259)).
- **Direction**
  The direction to search through the List window.
- **Exact**
  Check **Exact** if you only want to find objects that match your search exactly. For example, searching for "clk" without **Exact** will find */top/clk* and *clk1*.
- **Auto Wrap**
  Check **Auto Wrap** to continue the search at the beginning of the window.

List Signal Search dialog

Purpose	Menu command	Additional information
Locate signal values or transitions	**Edit > Search**	"Searching for values or transitions" (UM-252)



The List Signal Search dialog includes these options:

- **Signal Name(s)**
  A list of the objects currently selected in the List window. These objects are the subject of the search.
- **Any Transition**
  Searches for any transition in the selected signal(s).
- **Rising Edge**
  Searches for rising edges in the selected signal(s).
- **Falling Edge**
  Searches for falling edges in the selected signal(s).

- **Search for Signal Value**
  Searches for the value specified in the **Value** field; the value should be formatted using VHDL or Verilog numbering conventions (see "Numbering conventions" (CR-20) for more information).

Note: If your signal values are displayed in binary radix, see "Searching for binary signal values in the GUI" (CR-29) for details on how signal values are mapped between a binary radix and std_logic.

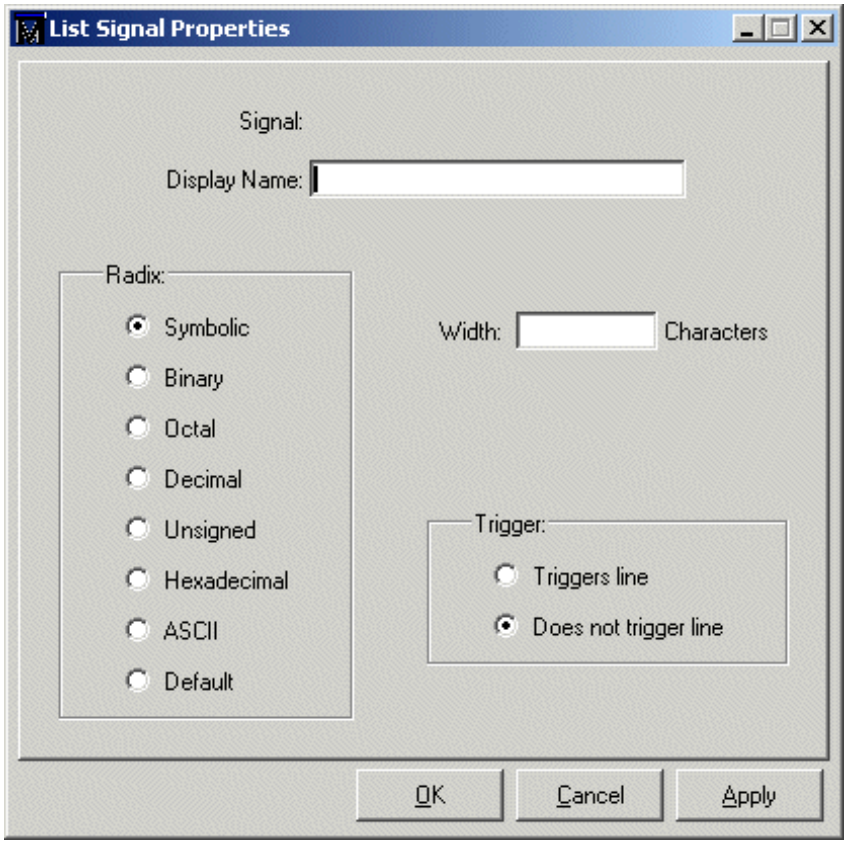
- **Search for Expression**
  Searches for the expression specified in the **Expression** field evaluating to a boolean true. See "Using the Expression Builder for expression searches" (UM-253) for information on the Builder button.

  The expression can involve more than one signal but is limited to signals logged in the List window. Expressions can include constants, variables, and DO files. If no expression is specified, the search will give an error. See "Expression syntax" (CR-23) for more information.
- **Match Count**
  Indicates the number of transitions or matches to search.You can search for the nth transition or the nth match on value.
- **Ignore Glitches**
  Ignores zero width glitches in VHDL signals and Verilog nets.

The Search Results are indicated at the bottom of the dialog.

List Signal Properties dialog

Purpose	Menu command	Additional information
Format signal display properties	**View > Signal Properties**	"Formatting objects in the List window" (UM-259)



The List Signal Properties dialog includes these options:

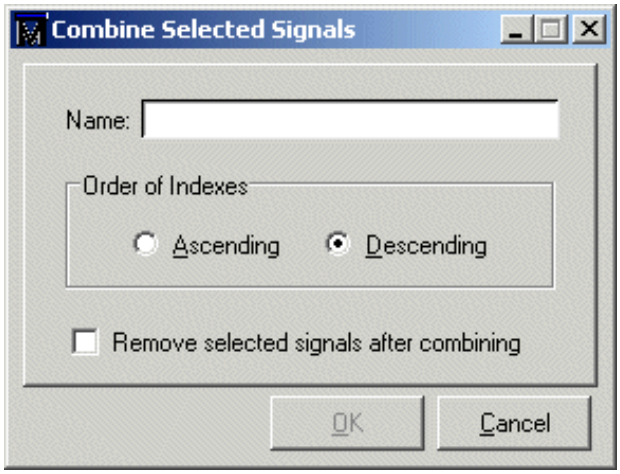
- **Signal**
  Shows the full pathname of the selected signal.
- **Display Name**
  An alias for the signal pathname that will appear at the top of the List window. Labels can make List window data more discernible by un-cluttering the top of the window.
- **Radix**
  Specifies the radix (base) in which the object value is expressed. The default radix is symbolic, which means that for an enumerated type, the List window lists the actual values of the enumerated type of that object. See "Formatting objects in the List window" (UM-259) for more information on setting the radix.

- **Width**
  Allows you to specify the desired width of the column used to list the object value. The default is an approximation of the width of the current value.
- **Trigger: Triggers line**
  Specifies that a change in the value of the selected object causes a new line to be displayed in the List window.
- **Trigger: Does not trigger line**
  Specifies that a change in the value of the selected object does not affect the List window.

  See "Configuring new line triggering in the List window" (UM-265) for more information about triggering.

Combine Selected Signals dialog

Purpose	Menu command	Additional information
Combine signals into a bus	**Tools > Combine Signals**	"Combining objects/creating busses" (UM-264)



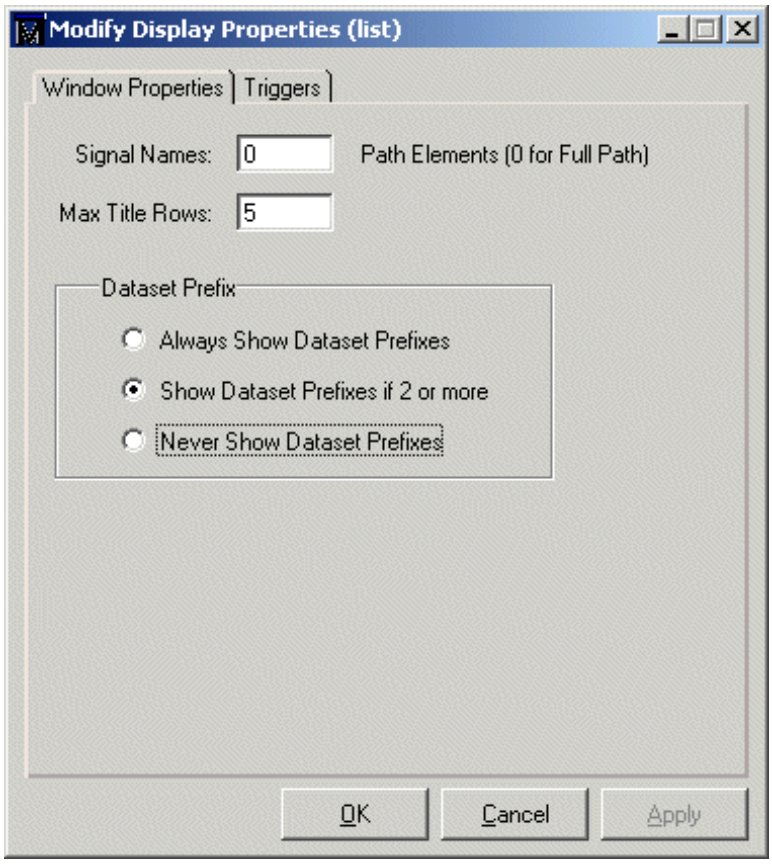
The Combine Selected Signals dialog includes these options:

- **Name**
  Specifies the name of the newly created bus.
- **Order of Indexes**
  Specifies in which order the selected signals are indexed in the bus. If set to **Ascending**, the first signal selected in the List window will be assigned an index of 0. If set to **Descending**, the first signal selected will be assigned the highest index number. Note that the signals are added to the bus in the order that they appear in the window. Ascending and descending affect only the order and direction of the indexes of the bus.
- **Remove selected signals after combining**
  Specifies whether you want to remove the selected signals from the List window once the bus is created.

Modify Display Properties dialog

Purpose	Menu command	Additional information
Configure window display properties	**Tools > Window Preferences**	"Formatting the List window" (UM-259)

Window Properties tab



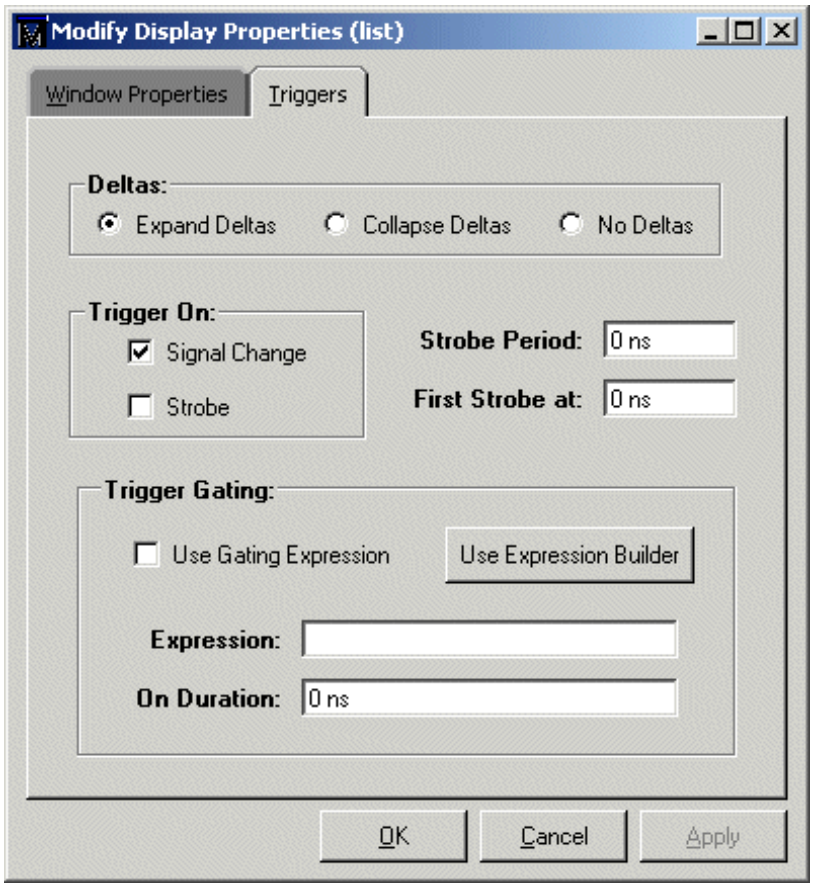
The Window Properties tab includes these options:

- **Signal Names**
  Sets the number of path elements to be shown in the List window. For example, "0" shows the full path. "1" shows only the leaf element.
- **Max Title Rows**
  Sets the maximum number of rows in the name pane.
- **Always Show Dataset Prefixes**
  Displays the dataset prefix associated with each signal pathname. Useful for displaying signals from multiple datasets.
- **Show Dataset Prefix if 2 or more**
  Displays dataset prefixes if there are signals in the window from 2 or more datasets.

- **Never Show Dataset Prefixes**
  Turns off display of dataset prefixes.

Triggers tab

The Triggers tab controls the triggering for the display of new lines in the List window. See "Configuring new line triggering in the List window" (UM-265) for more details.



The Triggers tab includes the following options:

- **Expand Deltas**
  When selected with the **Trigger on: Signal Change** check box, displays a new line for each time step on which objects change, including deltas within a single unit of time resolution.
- **Collapse Deltas**
  Displays only the final value for each time unit.
- **No Deltas**
  Hides the simulation cycle (delta) column.
- **Trigger On Signal Change**
  Triggers on signal changes. Defaults to all signals. Individual signals can be excluded from triggering by using the **View > Signal Properties** dialog or by originally adding them with the **-notrigger** option to the **add list** command (CR-48).

- **Trigger On Strobe**
  Triggers on the Strobe Period you specify; specify the first strobe with First Strobe at:.
- **Use Gating Expression**
  Enables triggers to be gated on (a value of 1) or off (a value of 0) by the specified Expression. See "Using gating expressions to control triggering" (UM-266) for more information.
- **Use Expression Builder** (button)
  Opens the Expression Builder to help you write a gating expression. See "Using gating expressions to control triggering" (UM-266).
- **Expression**
  Enter the expression for trigger gating into this field, or use the Expression Builder (select the Use Expression Builder button). The expression is evaluated when the List window would normally have displayed a row of data (given the trigger on signals and strobe settings above).
- **On Duration**
  Determines for how long triggering stays enabled after the gating expression returns to false (0). The default of 0 duration will enable triggering only while the expression is true (1). The duration is expressed in x number of default timescale units.

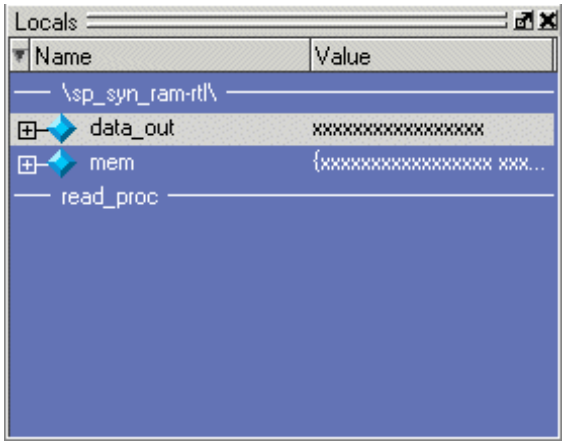
Customize Toolbar dialog

This dialog is the same for all windows. See "Customize Toolbar dialog" (GR-106) under the Main window for details.

Locals pane

The Locals pane displays data objects that are immediately visible from the statement that will be executed next (that statement is denoted by a blue arrow in the Source editor window). The contents of the window change from one statement to the next.

The Locals pane includes two columns. The first column lists the names of the immediately visible data objects. The second column lists the current value(s) associated with each name.

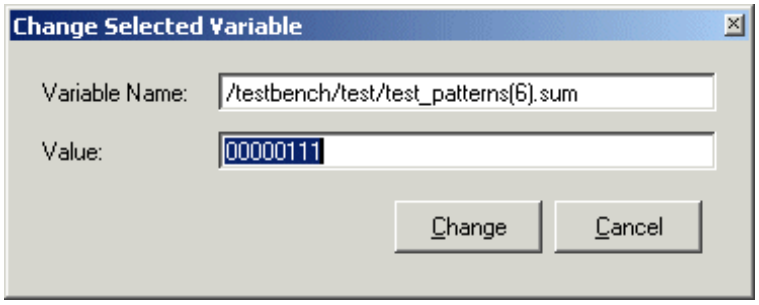


Locals dialogs

This section describes the locals related dialogs that are accessed via the Main window menu bar. Not all dialogs are documented (e.g., File > Save).

Change Selected Variable dialog

Purpose	Menu command	Additional information
Edit a variable value	**Edit** > **Advanced** > **Change**	**change** command (CR-81)

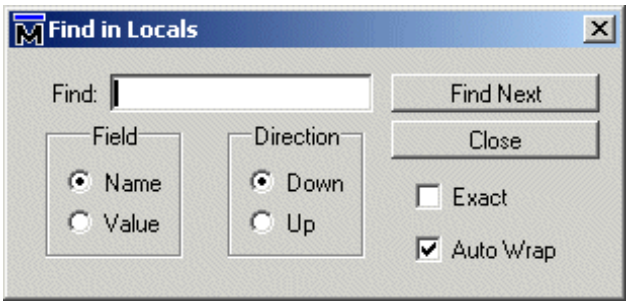


The Change Selected Variable dialog includes these options:

- **Variable Name**
  The variable name whose value you are changing.
- **Value**
  The new value for the specified variable. You can enter any value that is valid for the variable. An array value must be specified as a string (without surrounding quotation marks). To modify the values in a record, you need to change each field separately.

Find in Locals dialog

Purpose	Menu command	Additional information
Locate objects or values	**Edit > Find**	NA



You must activate the Locals pane by clicking in it before this dialog will be available.

The Find in Locals dialog includes these options:

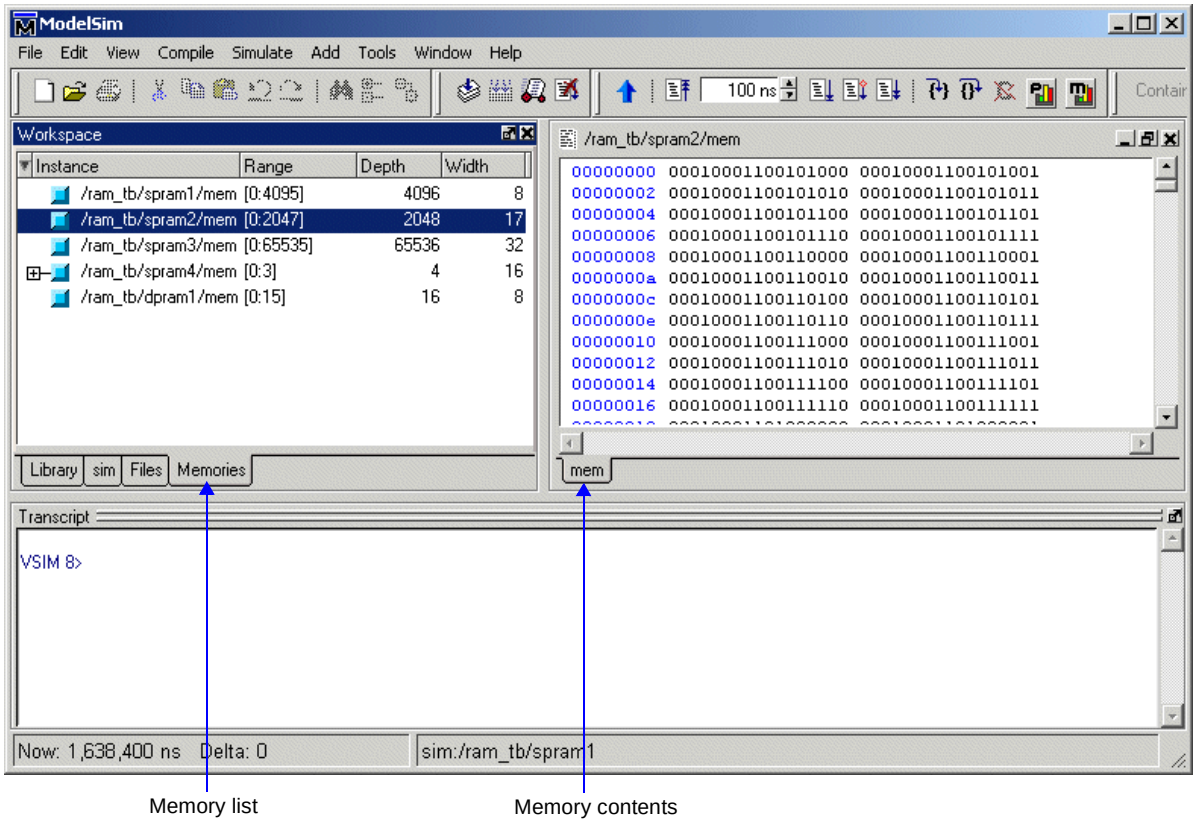
- **Find**
  Specify the text for which you want to search.
- **Field**
  Specify whether to search the name or value column.
- **Direction**
  Specify the direction to begin searching.
- **Exact**
  Check **Exact** if you only want to find objects that match your search exactly. For example searching for "addr" without Exact will find *addr* and *addr_r*.
- **Auto Wrap**
  Check **Auto Wrap** to continue the search at the top or bottom of the pane.

You can also do a quick find from the keyboard. When the Locals pane is active, each time you type a letter the highlight will move to the next object whose name begins with that letter.

Memory windows

The Main window lists and displays the contents of memories in your design. To view memories, select **View > Debug Windows > Memory**.

There are two windows related to memories: the memory list is displayed as a tab in the Main window Workspace; the memory contents windows display in the Main window MDI frame.



The memory list is from the top-level of the design. In other words, it is not sensitive to the context selected in the Structure tab.

Memories you can view

The Memory tab identifies and lists the following types of arrays as memories:

- reg, wire, bit, and std_logic arrays

  Any signal or variable that is an array of two dimensions (including arrays of arrays) are identified as memories and listed if the base type is a Verilog reg or wire type, or a VHDL enumerated type with values in std_ulogic, bit, and all related sub-types.

- Integer arrays

  Single dimensional arrays of integers are interpreted as 2D memory arrays. In these cases, the word width listed in the Memory List pane is equal to the integer size, and the depth is the size of the array itself. The appearance of this type of array in the memory list can be disabled via the View menu or the ShowIntMem (UM-533) variable in the *modelsim.ini* file.

- Single dimensional arrays of VHDL enumerated types other than std_logic or bit

  These enumerated type value sets must have values that are longer than one character. The listed width is the number of entries in the enumerated type definition and the depth is the size of the array itself. The appearance of this type of array in the memory list can be disabled via the View menu or the ShowEnumMem (UM-533) variable in the *modelsim.ini* file.

- 3D or greater arrays

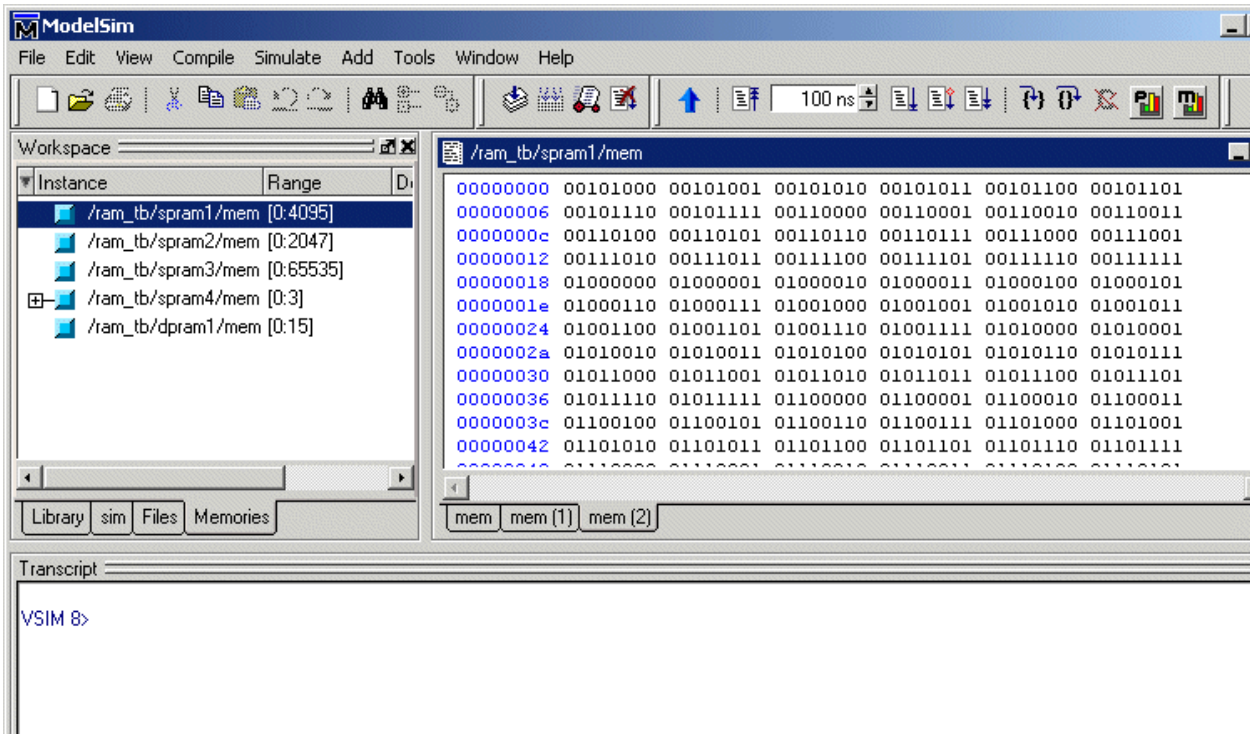
  Memories with three or more dimensions display with a plus sign ’+’ next to their names in the Memory List. Click the ’+’ to show the array indices under that level. When you finally expand down to the 2D level, you can double-click on the index, and the data for the selected 2D slice of the memory will appear in a memory contents window in the MDI frame. The appearance of this type of array in the memory list can be disabled via the View menu or the Show3DMem (UM-533) variable in the *modelsim.ini* file.

Viewing memory contents

When you double-click an instance on the Memory tab, ModelSim automatically displays a memory contents window in the MDI frame (see "Multiple document interface (MDI) frame" (GR-17)). You can also enter the command **add mem <instance>** at the vsim command prompt.

Viewing multiple memory instances

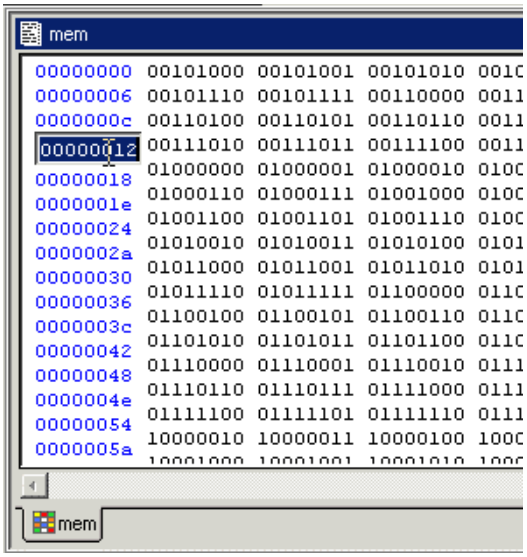
You can view multiple memory instances simultaneously. A window tab appears in the MDI frame for each instance you double-click in the Memory list.



See "Organizing windows with tab groups" (GR-18) for more information on the window tabs.

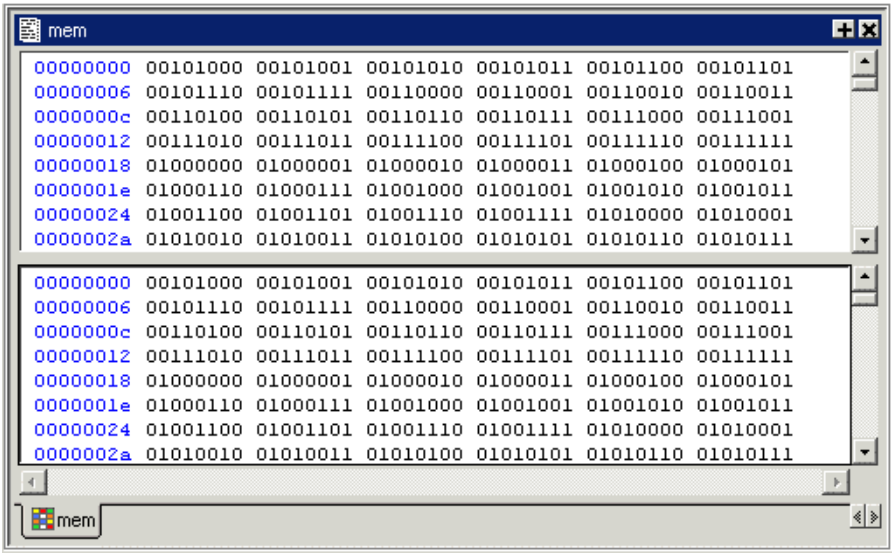
Direct address navigation

You can navigate to any address location directly by editing the address in the address column. Double-click on any address, type in the desired address, and hit **Enter**. The address display scrolls to the specified location.



Splitting the memory contents window

To split a memory contents window into two screens displaying the contents of a single memory instance, select **View > Split Screen** (or right-click in the pane and select **Split Screen** from the pop-up menu). This allows you to view different address locations within the same memory instance simultaneously.



Memory popup menu commands

The following commands are available when you right-click in the Memory tab of the Main window Workspace or the Memory contents pane in the MDI frame.

Memory tab popup menu

View Contents	view memory data for the selected memory in the Memory address pane
Memory Declaration	display the source code declaration for the selected memory in the MDI frame
Compare Contents	compare the contents of the selected memory with a selected reference memory or file; see "Compare Memory dialog" (GR-181)

Memory contents pane popup menus

When you right-click any memory address (first column in the Memory contents pane) the following popup menu commands are available:

Goto	go to specific memory address in currently displayed memory instance
Split Screen	split the memory contents window horizontally; see "Splitting the memory contents window" (GR-172)
Properties	set Address radix, Data radix, and line wrap options (words per line); see "Properties dialog" (GR-183).
Close Instance	close the Memory contents pane display of the currently selected memory instance
Close All	close the Memory contents pane display of all memory instances

When you right-click any memory data field in the Memory contents pane the following popup menu commands are available:

Edit	allows you to manually edit the selected data
Change	change the memory contents for all addresses or a range of addresses in the currently displayed memory instance; see "Change Memory dialog" (GR-179)
Load	load memory data to the currently displayed memory instance from a file; see "Load Memory dialog" (GR-175)
Save	save currently displayed memory data (all or a range) to a file; see "Save Memory dialog" (GR-177)

Find	searches for a specified memory data pattern in the currently displayed memory instance
Split Screen	split the memory contents window horizontally; see "Splitting the memory contents window" (GR-172)
Properties	set Address radix, Data radix, and line wrap options (words per line); see "Properties dialog" (GR-183).
Close Instance	close the Memory contents pane display of the currently selected memory instance
Close All	close the Memory contents pane display of all memory instances

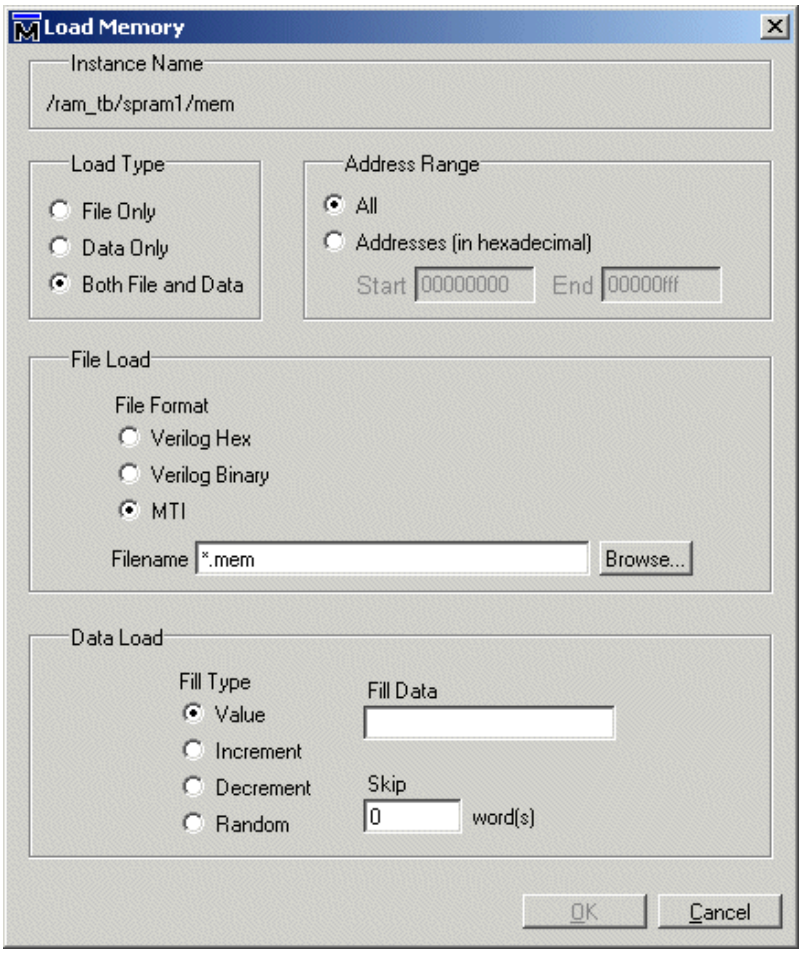
Memory dialogs

This section describes the memory-related dialogs that are accessed via the Main window menu bar. The dialogs are listed in the order in which they appear on the menus, top-to-bottom and left-to-right (i.e., starting with the File menu and progressing across the menu bar). Not all dialogs are documented (e.g., Goto).

Load Memory dialog

Purpose	Menu command	Additional information
Initialize memories	**File** > **Open** (with memory pane active)	**mem load** command (CR-199)

You can initialize memories in your design by either loading the contents from a file or by using an interactive command. An entire memory, a specific range of addresses, or an individual word can be overwritten. Choose the type of Load operation to be performed in the Load Type area. When either File Only or Data Only is selected, the unused section of the dialog is grayed out.

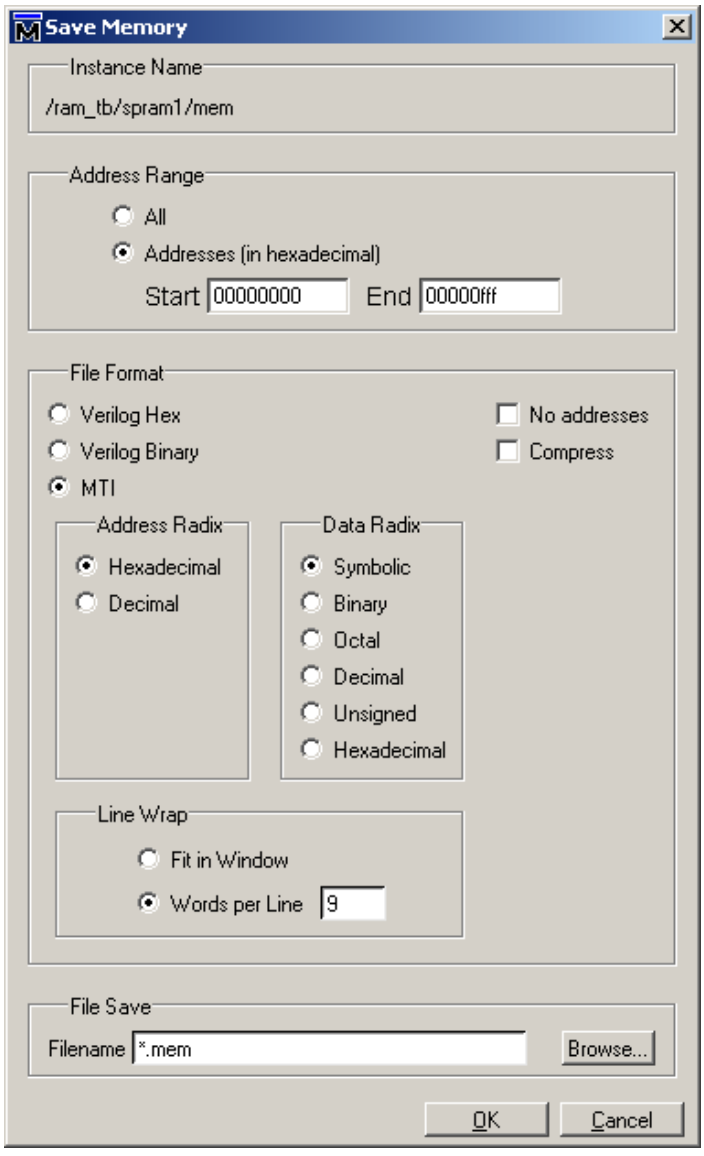


The Load Memory dialog includes these options:

- **Instance Name**
  Displays the name of the memory instance being loaded.
- **Load Type**
  Defines the type of load function you will perform. Your choices for loading data are: File Only, Data Only or Both File and Data.
- **Address Range**
  Specifies all addresses or a range of addresses in the memory that you want to load. The address radix of the displayed memory is shown in parentheses.
- **File Load**
  Contains all inputs related to loading from a file. This area of the dialog is grayed out if Load Type is specified as Data Only.
- **File Format**
  Specifies the format of the file to be loaded. Verilog Hex, Verilog Binary, or MTI format can be explicitly set, or the format can be determined automatically from the file (if the file was created with the **mem save** command).
- **File name**
  The name of the memory file to load. You can manually edit this field or click Browse to select a file.
- **Data Load**
  Contains all inputs related to loading memory data. This area of the dialog is grayed out if Load Type is specified as File Only.
- **Fill Type**
  Specifies how to apply the fill data, either directly as a value, or algorithmically. See the **mem load** command (CR-199) for more information on Fill Type and Fill Data.
- **Fill Data**
  Specifies the fill data for addresses not contained in the load file.
- **Skip**
  Specifies the number of words to skip when applying a fill pattern sequence.

Save Memory dialog

Purpose	Menu command	Additional information
Save memories	**File** > **Save** (with memory pane active)	**mem save** command (CR-202)



The Save Memory dialog includes these options:

- **Instance Name**
  The memory instance being saved.

- **Address Range**
  Specifies all or a range of addresses to be saved into the file.
- **File Format**
  Specifies whether memory is to be saved in Verilog Hex, Verilog Binary, or MTI format. Also, specify the Address and Data radix for MTI format.
- **No addresses**
  Specifies that no addresses are to appear in the saved file. This enables the file to be reloaded anywhere in the memory.
- **Compress**
  Applies a simple ASCII compression to the saved file. The compression algorithm replaces repeating lines with a single asterisk, like is done with the Unix “od” command.
- **Line Wrap**
  Designates that lines of memory data will Fit in Window or allows user to designate Words per Line.
- **Filename**
  Name of file to be saved.

MTI memory data file format

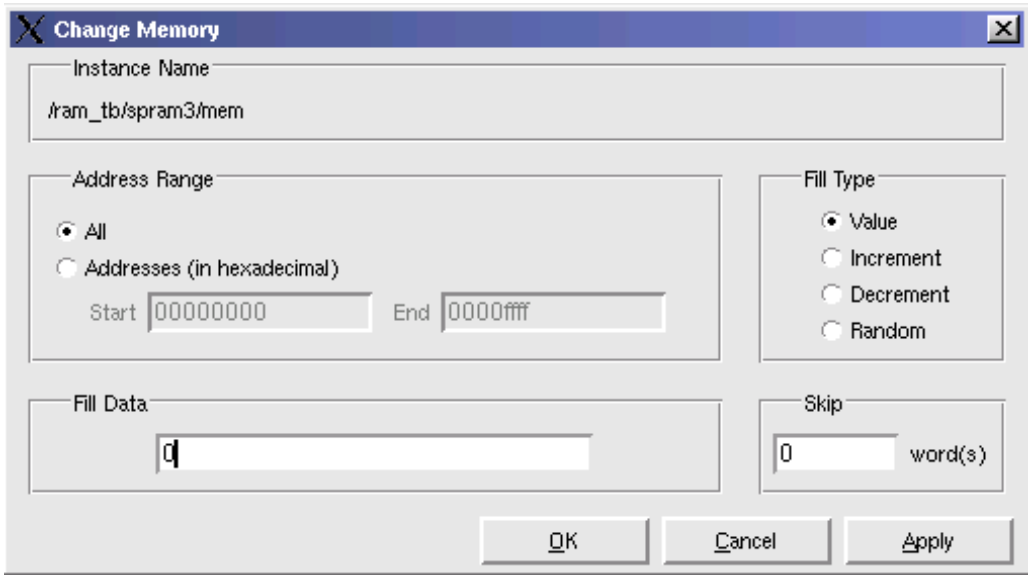
The MTI memory data file format is as illustrated in the following example:

```
// memory data file
// (do not edit the following line - required for mem load use)
// format=mti addressradix=d dataradix=s direction=ascending
 0: 110 110 110 110 110 110
 6: 110 110 110 110 000 000
12: 000 000 000 000 000 000
18: 000 000 000 000 000 000
24: 000 000 000 000 000 000
30: 000 000
```

The possible format, address radix, data radix, and direction settings are as specified by the corresponding options in the **mem save** command (CR-202) and **mem load** command (CR-199).

Change Memory dialog

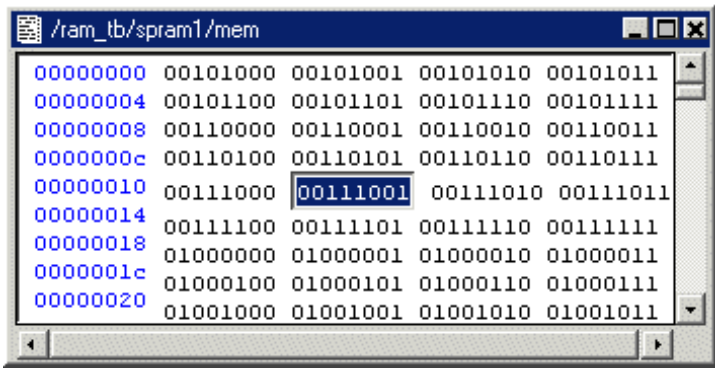
Purpose	Menu command	Additional information
Edit memory data values	**Edit** > **Advanced** > **Change** (with memory pane active)	NA



The Change Memory dialog includes the following:

- **Instance Name**
  Displays the name of the memory instance being loaded.
- **Address Range**
  Specifies all addresses or a starting and ending address to be changed. The address radix of the currently displayed memory is shown in parentheses.
- **Fill Data**
  Specifies the fill data for specified addresses.
- **Fill Type**
  Specifies how to apply the fill data, either directly as a value, or algorithmically. See the **mem load** command (CR-199) for more information on Fill Type and Fill Data.
- **Skip**
  Specifies the number of words to skip after applying a fill pattern sequence.

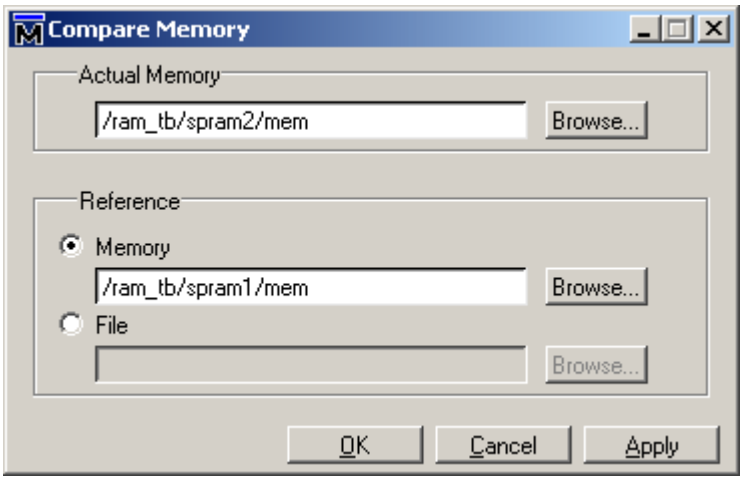
Changing data for individual addresses



To edit memory data in place, double-click (or right-click and select **Edit)** any word in a memory contents window. The data is highlighted. Type in the desired change. Pressing <Enter> commits the change; <Esc> aborts it. <Tab> scrolls down the list of data entries, while <Shift>-<Tab> scrolls up the list.

Compare Memory dialog

Purpose	Menu command	Additional information
compare selected memory to reference memory or file	right click Memory tab of Main window Workspace > select **Compare Contents**	NA

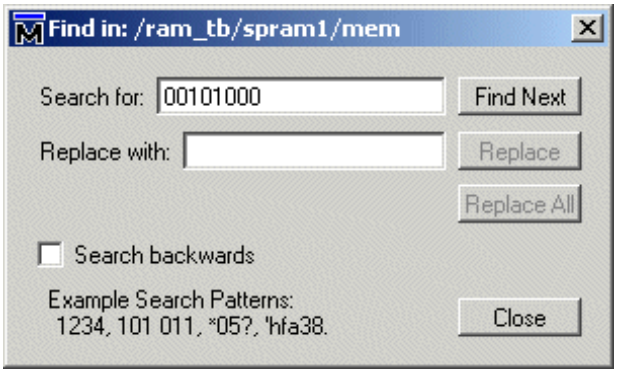


The Compare Memory dialog includes the following fields:

- **Actual Memory**
  Specifies the name of the memory that will be compared to a reference memory or file. You can manually edit this field or click Browse to select a memory.
- **Reference Memory**
  Specifies the name of the reference memory. You can manually edit this field or click Browse to select a memory.
- **Reference File**
  Specifies the name of the reference file. You can manually edit this field or click Browse to select a file.

Find dialog

Purpose	Menu command	Additional information
Locate a value in the memory contents	**Edit > Find**	NA



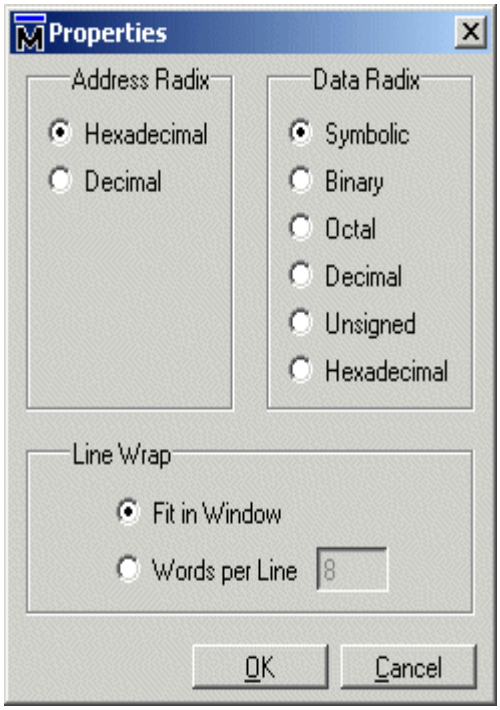
You must activate a memory content window in the MDI frame by clicking in it before this dialog will be available.

The Find dialog includes these options:

- **Search for:**
  The value you want to find.
- **Replace with**
  An optional value to replace the located value.
- **Search backwards**
  Search backwards through the memory.

Properties dialog

Purpose	Menu command	Additional information
Configure window display properties	**View > Properties**	**mem display** command (CR-196)



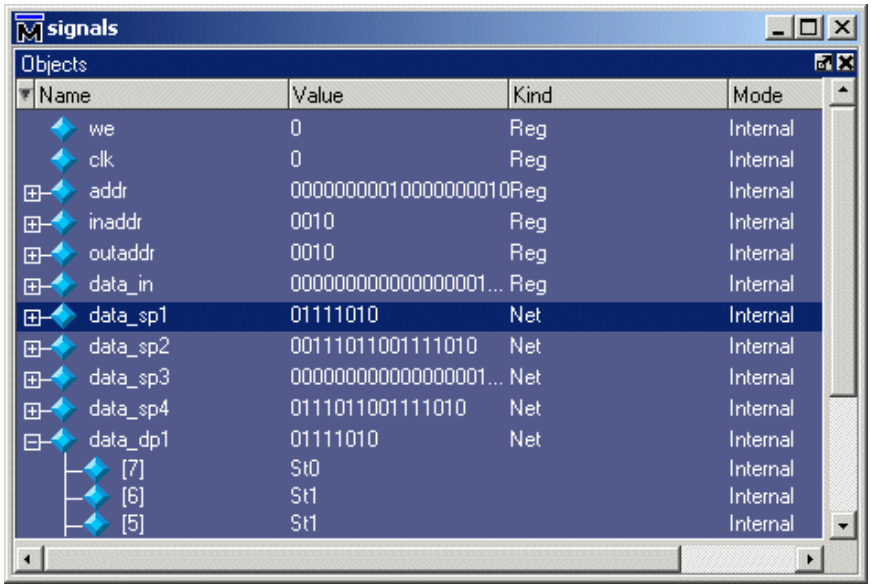
The Properties dialog includes these options:

- **Address Radix**
  The radix for the address. Can be Hexadecimal or Decimal.
- **Data Radix**
  The radix for the data. Non-enumerated type memories can be Symbolic, Binary, Octal, Decimal, Unsigned, and Hexadecimal. Enumerated type memories are only symbolic data types, and all other options are grayed out.
- **Line Wrap**
  The number of words per line can be set, or arbitrarily determined based on the size of the window.

Objects pane

The Objects pane shows the names and current values of declared data objects in the current region (selected in the structure tabs of the Workspace). Data objects include signals, nets, registers, constants and variables not declared in a process, generics, parameters, and member data variables of a SystemC module.

Clicking an entry in the window highlights that object in the Dataflow and Wave windows. Double-clicking an entry highlights that object in a Source editor window (opening a Source editor window if one is not open already). You can also right click an object name and add it to the List or Wave window, or the current log file.



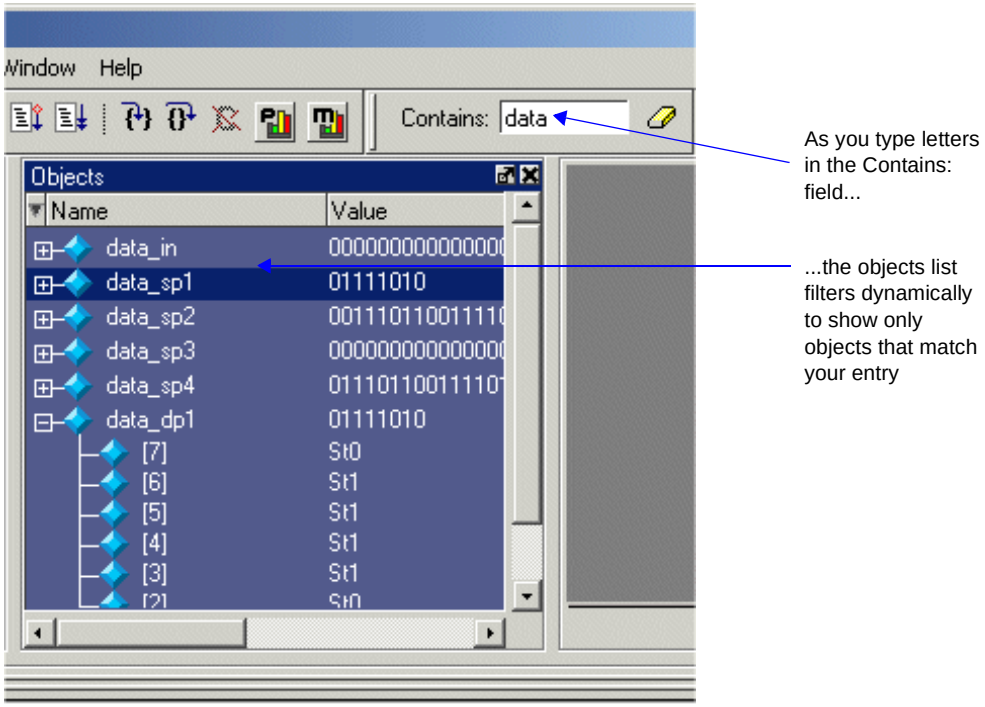
Filtering the objects list

You can filter the objects list by name or by object type.

Filtering by name

To filter by name, start typing letters in the **Contains** field on the Main window toolbar.

As you type letters, the objects list filters to show only those signals that contain those letters.



To display all objects again, click the Eraser icon to clear the entry.

Filters are stored relative to the region selected in the Structure window. If you re-select a region that had a filter applied, that filter is restored. This allows you to apply different filters to different regions.

Filtering by signal type

The **View > Filter** menu selection allows you to specify which signal types to display in the Objects window. Multiple options can be selected.

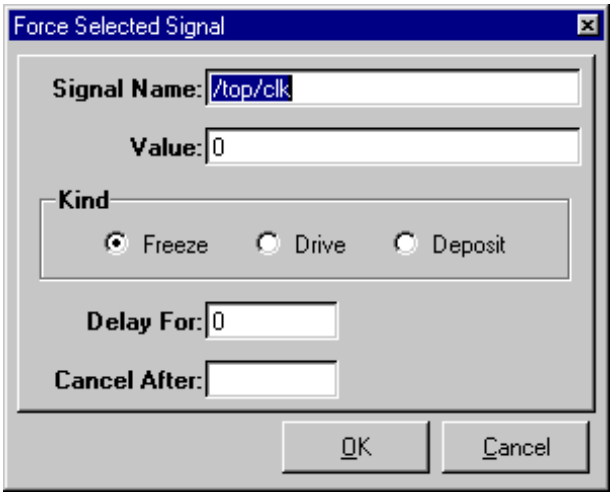
Objects dialogs

This section describes the objects related dialogs that are accessed via the Main window menu bar. Not all dialogs are documented (e.g., File > Save).

Force Selected Signal dialog

Purpose	Menu command	Additional information
Apply stimulus to a signal or net	**Edit > Advanced > Force**	**force** command (CR-180)

The **Force** function (unavailable for SystemC) allows you to apply stimulus to the selected signal or net. Multiple signals can be selected and forced; the force dialog remains open until all of the signals are either forced, skipped, or you close the dialog. To cancel a force command, use the **Edit > Advanced > NoForce** command.



The Force Selected Signal dialog includes these options:

- **Signal Name**
  Specifies the signal or net for the applied stimulus.
- **Value**
  Initially displays the current value, which can be changed by entering a new value into the field. A value can be specified in radixes other than decimal by using the form (for VHDL and Verilog, respectively):

  ```
  base#value -or- b|o|d|h'value
  ```

  16#EE or h'EE, for example, specifies the hexadecimal value EE.

Kind

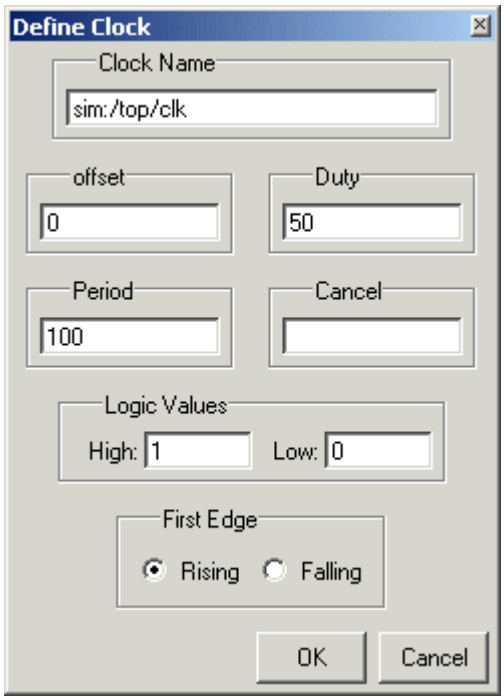
- **Freeze**
  Freezes the signal or net at the specified value until it is forced again or until it is unforced with a **noforce** command (CR-208).

  Freeze is the default for Verilog nets and unresolved VHDL signals and **Drive** is the default for resolved signals.

  If you prefer Freeze as the default for resolved and unresolved signals, you can change the default force kind in the *modelsim.ini* file (see "Preference variables located in INI files" (UM-524)).
- **Drive**
  Attaches a driver to the signal and drives the specified value until the signal or net is forced again or until it is unforced with a **noforce** command (CR-208). This type of force is illegal for unresolved VHDL signals.
- **Deposit**
  Sets the signal or net to the specified value. The value remains until there is a subsequent driver transaction, or until the signal or net is forced again, or until it is unforced with a **noforce** command (CR-208).
- **Delay For**
  Allows you to specify how many time units from the current time the stimulus is to be applied.
- **Cancel After**
  Cancels the force after the specified period of simulation time.

Define Clock dialog

Purpose	Menu command	Additional information
Add clock signals	**Edit** > **Advanced** > **Clock**	**force** command (CR-180)

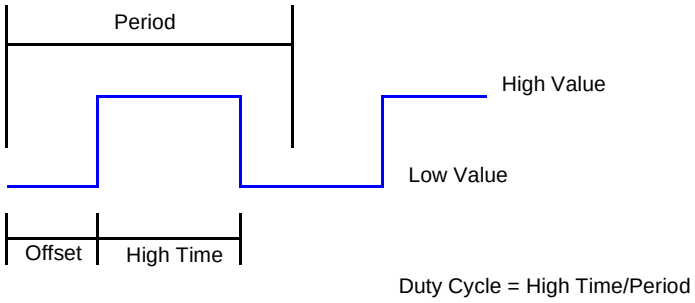


The Define Clock dialog includes the following options:

- **Clock Name**
  The name of the clock signal.
- **Offset**
  The time after the current simulation time that the first transition will occur.
- **Duty**
  The percentage of the period that the clock is high or low.
- **Period**
  The period of the clock signal.
- **Cancel**
  A simulation period after which the clock definition should be cancelled.
- **Logic Values**
  If the signal type is std_logic, std_ulogic, bit, verilog wire, verilog net, or any other logic type where 1 and 0 are valid, then 1 is the default High Value and 0 is the default Low Value. For other signal types, you will need to specify a High Value and a Low Value for the clock.

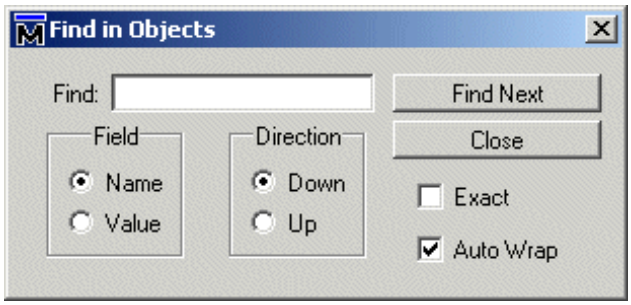
- **First Edge**
  Specify whether the first transition will be rising or falling.

For clock signals starting on the rising edge, a visual depiction of Period, Offset, and Duty Cycle looks like this:



Find in Objects dialog

Purpose	Menu command	Additional information
Locate objects or values	**Edit > Find**	NA



You must activate the Objects pane by clicking in it before this dialog will be available.

The Find in Signals dialog includes these options:

- **Find**
  Specify the text for which you want to search.
- **Field**
  Specify whether to search the name or value column.
- **Direction**
  Specify the direction to begin searching.
- **Exact**
  Check **Exact** if you only want to find objects that match your search exactly. For example, searching for "clk" without Exact will find */top/clk* and */top/clk1*.
- **Auto Wrap**
  Check **Auto Wrap** to continue the search at the top or bottom of the window.

You can also do a quick find from the keyboard. When the Objects window is active, each time you type a letter the signal selector (highlight) will move to the next signal whose name begins with that letter.

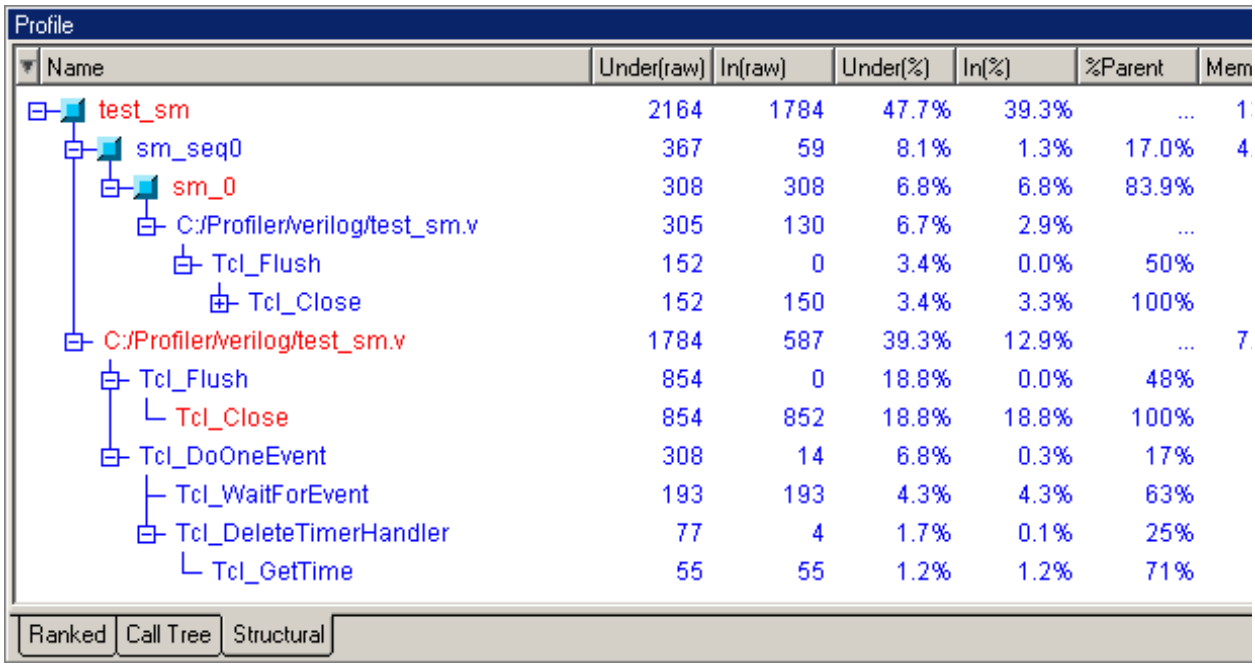
Modify Breakpoints dialog

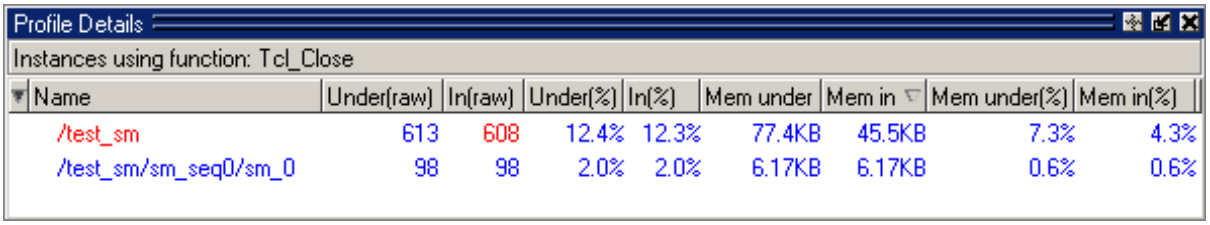
This dialog is the same as the one you access in the Main window. See "Modify Breakpoints dialog" (GR-95) for more information.

Profile panes

The Profile and Profile Details panes display the results of statistical performance and memory allocation profiling. By default, both panes are displayed within the Main window but they can be undocked from the Main window to stand alone. Each pane contains three tabs for displaying profile results: Ranked, Call Tree, and Structural.

For details about using the profiler see *Chapter 12 - Profiling performance and memory use* in the *ModelSim User's Manual*.





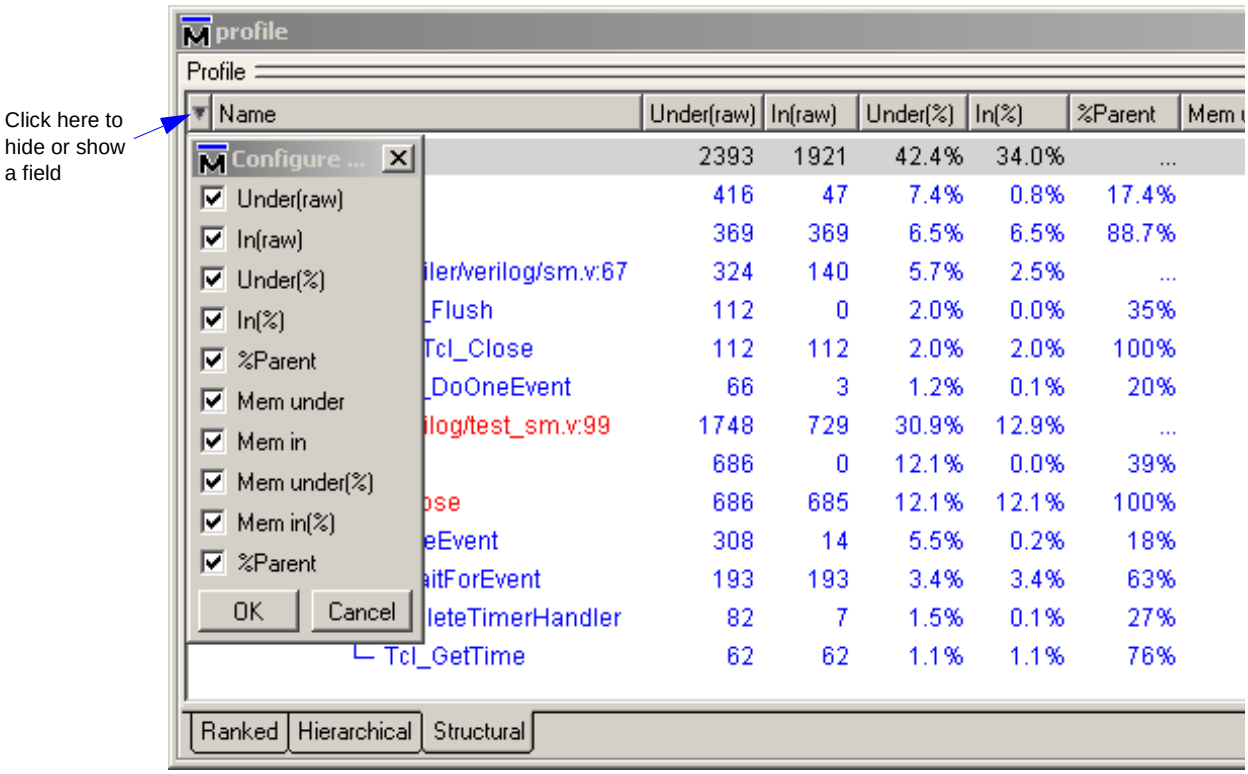
Profile pane columns

The Profile panes include the columns described below. See "Hiding/showing columns in the Profile panes" (GR-194) for details on how to show or hide particular columns.

- The **Name** column lists the filename of an HDL function or instance, and the line number at which it appears. Most useful names consist of a line of VHDL or Verilog source code. If you use a PLI/VPI or FLI routine, then the name of the C function that implements that routine can also appear in the Name column.
- The **Under (raw)** column lists the raw number of Profiler samples collected during the execution of a function, including all support routines under that function; or, the number of samples collected for an instance, including all instances beneath it in the structural hierarchy.
- The **In (raw)** column lists the raw number of Profiler samples collected during a function or instance.
- The **Under%** column lists the ratio (as a percentage) of the samples collected during the execution of a function and all support routines under that function to the total number of samples collected; or, the ratio of the samples collected during an instance, including all instances beneath it in the structural hierarchy, to the total number of samples collected.
- The **In%** column lists the ratio (as a percentage) of the total samples collected during a function or instance.
- The **%Parent** column (not in Ranked view) lists the ratio, as a percentage, of the samples collected during the execution of a function or instance to the samples collected in the parent function or instance.
- The **Mem under** column lists the amount of memory allocated to a function, including all support routines under that function; or, the amount of memory allocated to an instance, including all instances beneath it in the structural hierarchy.
- The **Mem in** column lists the amount of memory allocated to a function or instance.
- The **Mem under (%)** column lists the ratio (as a percentage) of the amount of memory allocated to a function and all of its support routines to the total memory available; or, the ratio of the amount of memory allocated to an instance, including all instances beneath it in the structural hierarchy, to the total memory available.
- The **Mem in (%)** column lists the ratio (as a percentage) of the amount of memory allocated to a function or instance to the total memory available.
- The **%Parent** column lists (not in Ranked view) the ratio, as a percentage, of the memory allocated to a function or instance to the memory allocated to the parent function or instance.

Hiding/showing columns in the Profile panes

You can hide or show any of the columns in the Profile panes. Click the drop-down arrow on the left-hand side of the dialog and select a column name.



The selection acts as a toggle. Select it once to hide a column; select it again to show the column.

Profiler popup menu commands

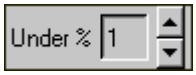
The following menu commands are available when you right click in the Ranked, Call Tree, and Structural views of the Profile window, or in the Profile Details window. (The Ranked view does not contain all of the commands available in the Call Tree and Structural views.)

View Source	displays source code of selected function in Source window
Function Usage	displays all instances using the selected function in the Profile Details window
Instance Usage	(Structural view only) displays all instances with the same definition as the selected instance in the Profile Details window
View Instantiation	(Structural view only) opens Source window at the point in the source code where the selected instance is instantiated
Callers & Callees	displays all callers and callees for the selected function in the Profile Details window. Items above the selected function are callers; items below are callees. The selected function is designated with an arrow.
Display in Call Tree	expands the Call Tree view of the Profile window to display all occurrences of the selected function; puts the selected function into a search buffer so you can easily cycle across all occurrences of that function.
Display in Structural	expands the Structural view of the Profile window to display all occurrences of the selected function; puts the selected function into a search buffer so you can easily cycle across all occurrences of that function.
The following commands are only available in the Call Tree and Structural views	
Set Root	causes the display to be rooted at the currently selected item
Ascend Root	causes the display root to ascend one level
Reset Root	causes the display to be reset to normal
Expand Selected	expand hierarchy of the selected function or instance
Collapse Selected	collapse hierarchy of the selected function or instance
Expand All	expand hierarchy of all functions and instances
Collapse All	collapse hierarchy of all functions and instances
Collapse To Parent	collapse hierarchy to the parent function or instance
Expand One Level	expand hierarchy of selected function or instance one level
Show Calls	toggle the display of call stack entries

Profiler toolbar

The Ranked, Call Tree and Structural views all share a toolbar in the Main window. The table below describes the icons in this toolbar.

Main window Profiler toolbar buttons

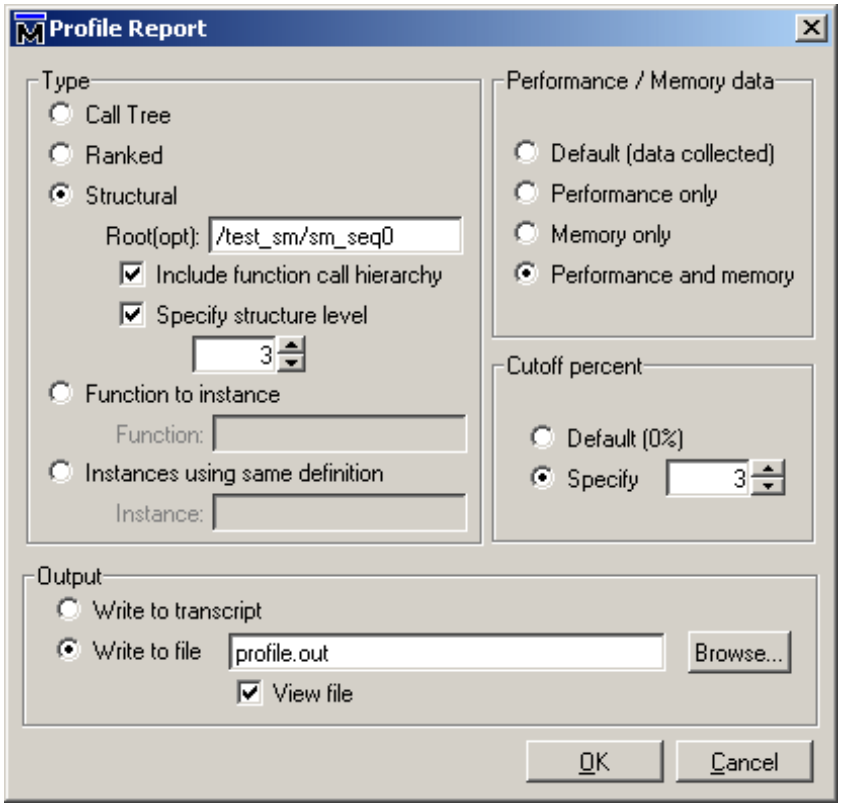
Button	Menu equivalent	Command equivalents
Memory Profiling enable collection of memory usage data	Tools > Profile > Memory	
Performance Profiling enable collection of statistical performance data	Tools > Profile > Performance	
Collapse Sections on/off toggling of reporting for collapsed processes and functions.	Tools > Profile > Collapse Sections	
 **Profile Cutoff** display performance and memory profile data equal to or greater than set percentage		
Refresh profile data refresh profile performance and memory data after changing profile cutoff		
Save profile results save profile data to output file (prompts for file name)	Tools > Profile > Profile Report	profile report (CR-231)
Profile Find search for the named string in the Profile pane		

Profiler dialogs

This section describes the profiler-related dialogs that are accessed via the Main window menu bar. Not all dialogs are documented (e.g., File > Save).

Profile Report dialog

Purpose	Menu command	Additional information
Create textual reports from performance and memory profile results	**Tools** > **Profile** > **Profile Report**	**profile report** command (CR-231)



The Profile Report dialog includes the following options:

- **Type**
  Save a textual report from Call Tree, Ranked, and Structural profile data. The Structural option allows you to designate the root instance for the report, include the function call hierarchy, and specify the structure level. You can also create a Function to instance report for the designated function, and a report of Instances using the same definition as the designated instance.

- **Performance/Memory data**
  Elect to save performance profile data only, memory allocation data only, or both.
- **Cutoff percent**
  Report results including and above the designated or Default percentage.
- **Output**
  Writes the textual report directly to the transcript or to a file. Will also display the file immediately after it is created if "View file" is selected.

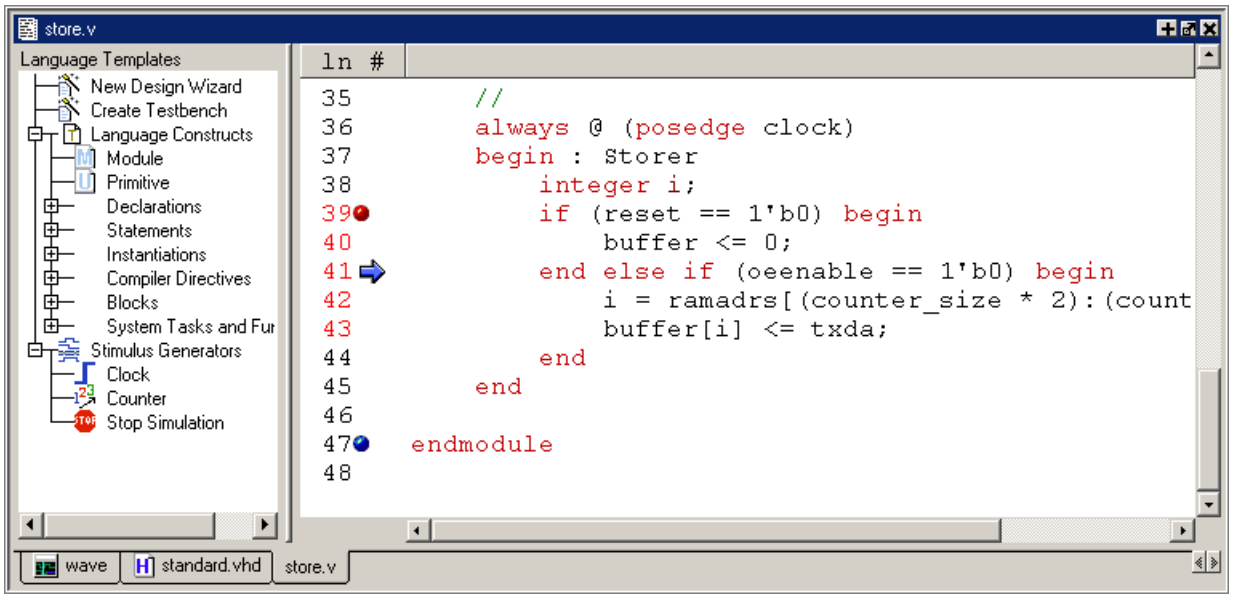
Source window

Source files display by default in the MDI frame of the Main window along with the memory content and Wave windows. The window can be undocked from the Main window by pressing the Undock button in the window header or by using the **view -undock source** command.

You can edit source files as well as set breakpoints, step through design files, and view code coverage statistics.

By default, the Source window displays your source code with line numbers. You may also see the following graphic elements:

- Red line numbers – denote lines on which you can set a breakpoint
- Blue arrow – denotes the currently active line or a process that you have selected in the "Active Processes pane" (GR-108)
- Red circles – denote file-line breakpoints; gray circles denote breakpoints that are currently disabled
- Blue circles – denote line bookmarks
- Language Templates pane – displays Language templates (GR-201)

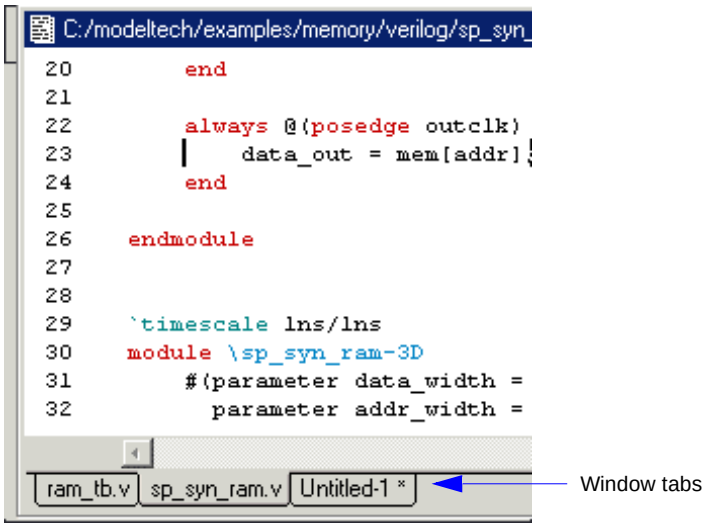


Opening source files

You can open source files using the **File > Open** command. Alternatively, you can open source files by double-clicking objects in other windows. For example, if you double-click an item in the Objects window or in the structure tab of the Workspace, the underlying source file for the object will open, and the cursor will scroll to the line where the object is defined.

Displaying multiple source files

By default each file you open or create is marked by a window tab, as shown in the graphic below.



See "Organizing windows with tab groups" (GR-18) for more information on these tabs.

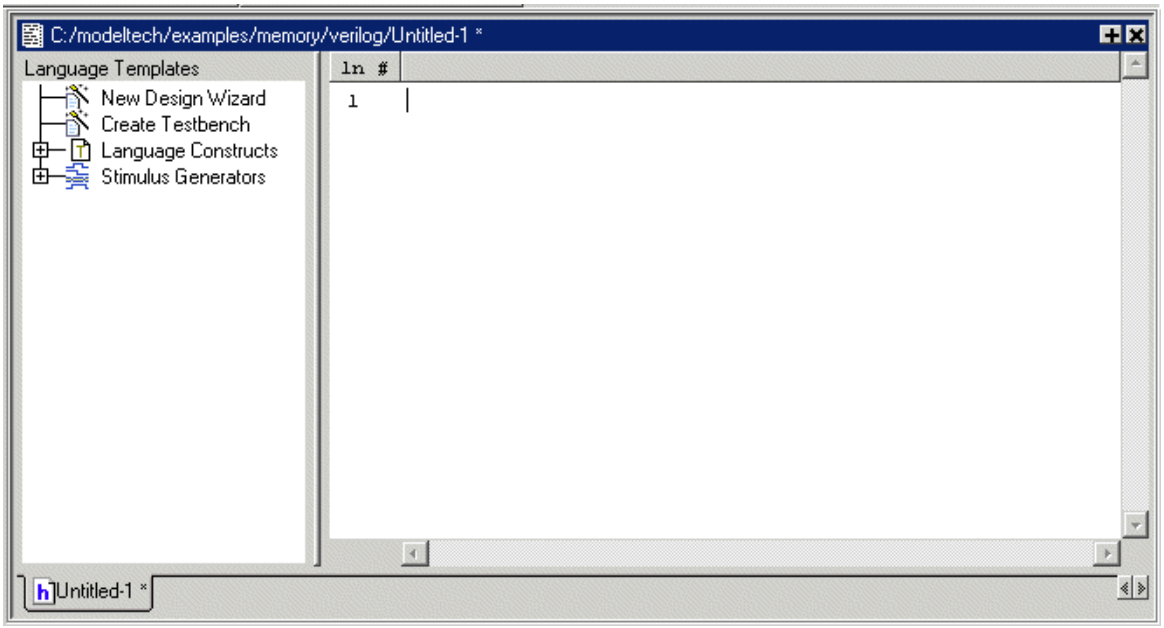
You can also display source files in independent windows within the MDI frame. To switch between the two layouts, select **View > Tabbed MDI**.

Language templates

ModelSim language templates help you write code. They are a collection of wizards, menus, and dialogs that produce code for new designs, testbenches, language constructs, logic blocks, etc.

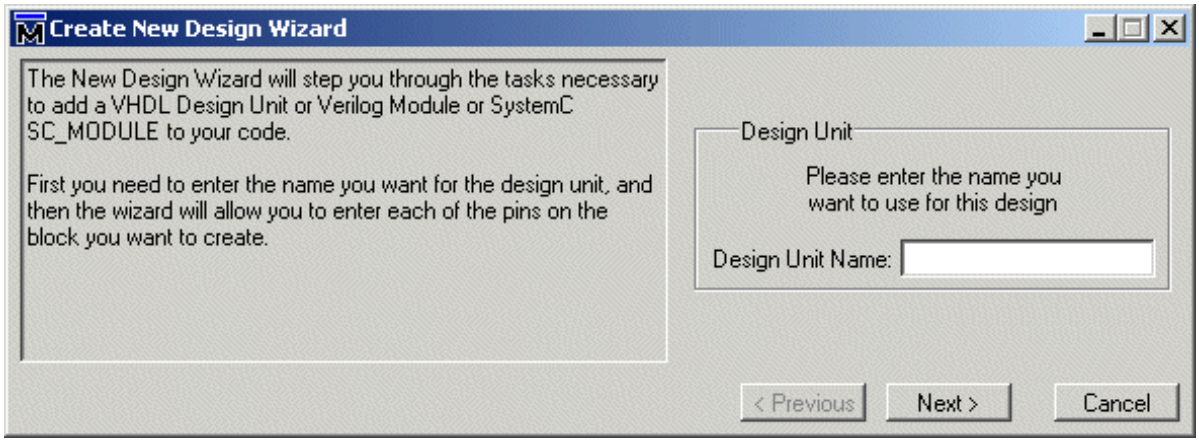
Important: The language templates are not intended to replace thorough knowledge of coding. They are intended as an interactive "reference" for creating small sections of code. If you are unfamiliar with a particular language, you should attend a training class or consult one of the many available books.

To use the templates, either open an existing file, or select **File > New > Source** to create a new file. Once the file is open, select **View > Source > Show language templates**. This displays a pane that shows the available templates.

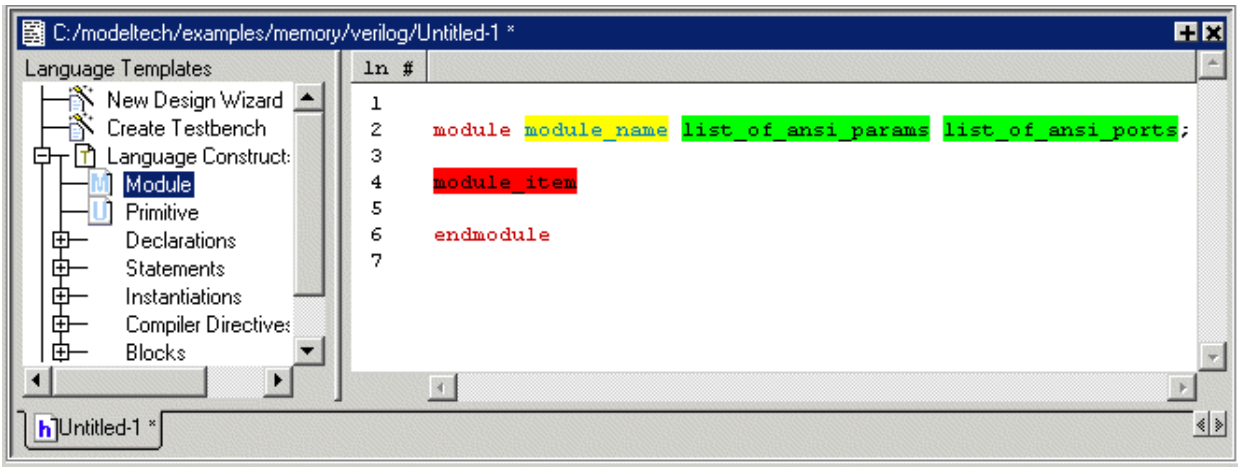


The templates that appear depend on the type of file you create. For example Module and Primitive templates are available for Verilog files, and Entity and Architecture templates are available for VHDL files.

Double-click an object in the list to open a wizard or to begin creating code. Some of the objects bring up wizards while others insert code into your source file. The dialog below is part of the wizard for creating a new design. Simply follow the directions in the wizards.

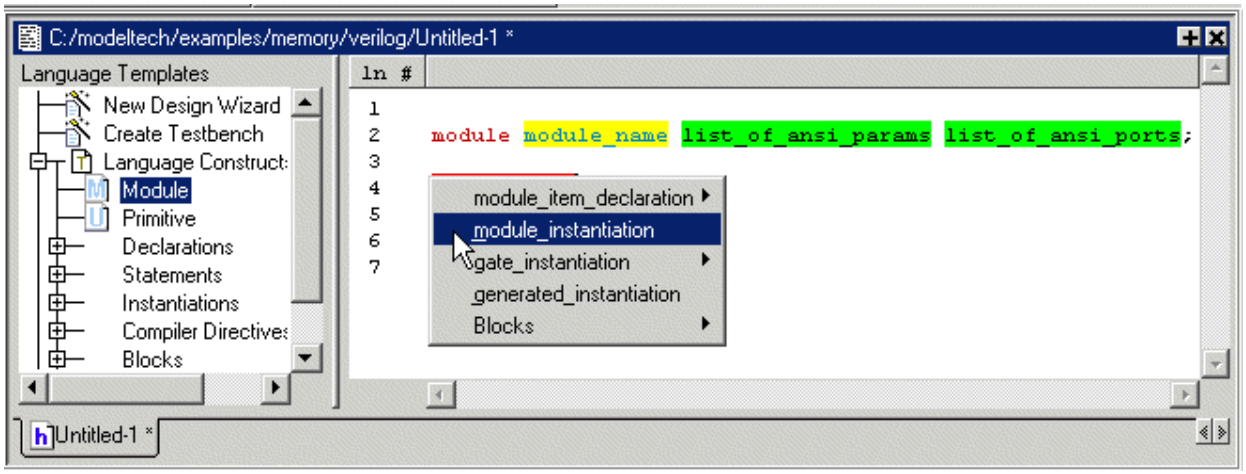


Code inserted into your source contains a variety of highlighted fields. The example below shows a module statement inserted from the Verilog template.



Some of the fields, such as *module_name* in the example above, are to be replaced with names you type. Other fields can be expanded by double-clicking and still others offer a

context menu of options when double-clicked. The example below shows the menu that appears when you double-click *module_item*.



Setting file-line breakpoints

You can easily set "File-line breakpoints" (GR-264) in a Source window using your mouse. Click on a red line number at the left side of the Source window, and a red circle denoting a breakpoint will appear. The breakpoints are toggles – click once to create the breakpoint; click again to disable or enable the breakpoint.

To delete the breakpoint completely, right click the red circle, and select **Remove Breakpoint**. Other options on the context menu include:

- **Disable/Enable Breakpoint**
  Deactivate or activate the selected breakpoint.
- **Edit Breakpoint**
  Open the "File Breakpoint dialog" (GR-98) to change breakpoint arguments.
- **Edit All Breakpoints**
  Open the "Modify Breakpoints dialog" (GR-95)

Checking object values and descriptions

There are two quick methods to determine the value and description of an object displayed in the Source window:

- select an object, then right-click and select **Examine** or **Describe** from the context menu
- pause over an object with your mouse pointer to see an examine pop-up

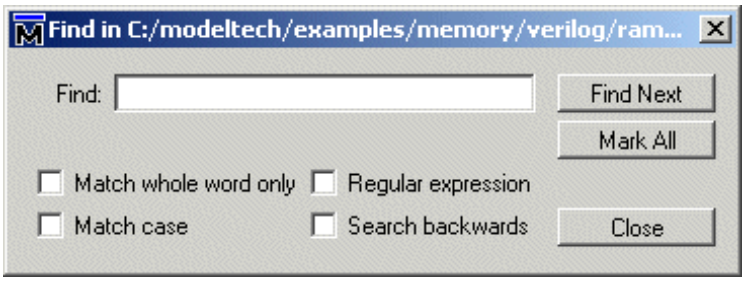
Select **Tools > Options > Examine Now** or **Tools > Options > Examine Current Cursor** to choose at what simulation time the object is examined or described.

You can also invoke the **examine** (CR-162) and/or **describe** (CR-147) command on the command line or in a macro.

Finding and replacing in a Source window

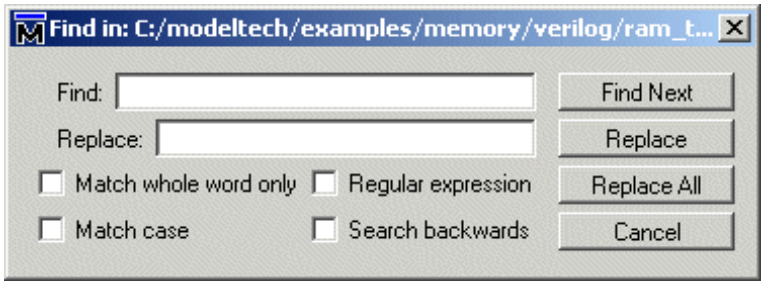
Two dialogs with slightly different options allow you to find, mark, and replace text strings or regular expressions in a Source window.

If you select **Edit > Find**, you will see the following dialog:



Enter the value to search for in the **Find** field. Optionally specify whether the entries are case sensitive and whether to search backwards from the current cursor location. Check **Match whole word only** to prevent implicit wildcards. Check the **Regular expression** checkbox if you are using regular expressions. The Mark All button places bookmarks on all lines that contain the text for which you are searching (see "Marking lines with bookmarks" (GR-204)).

If you select **Edit > Replace**, you will see this dialog:



Most of the options are the same as those in the Find dialog, but you would also enter a value in the **Replace** field.

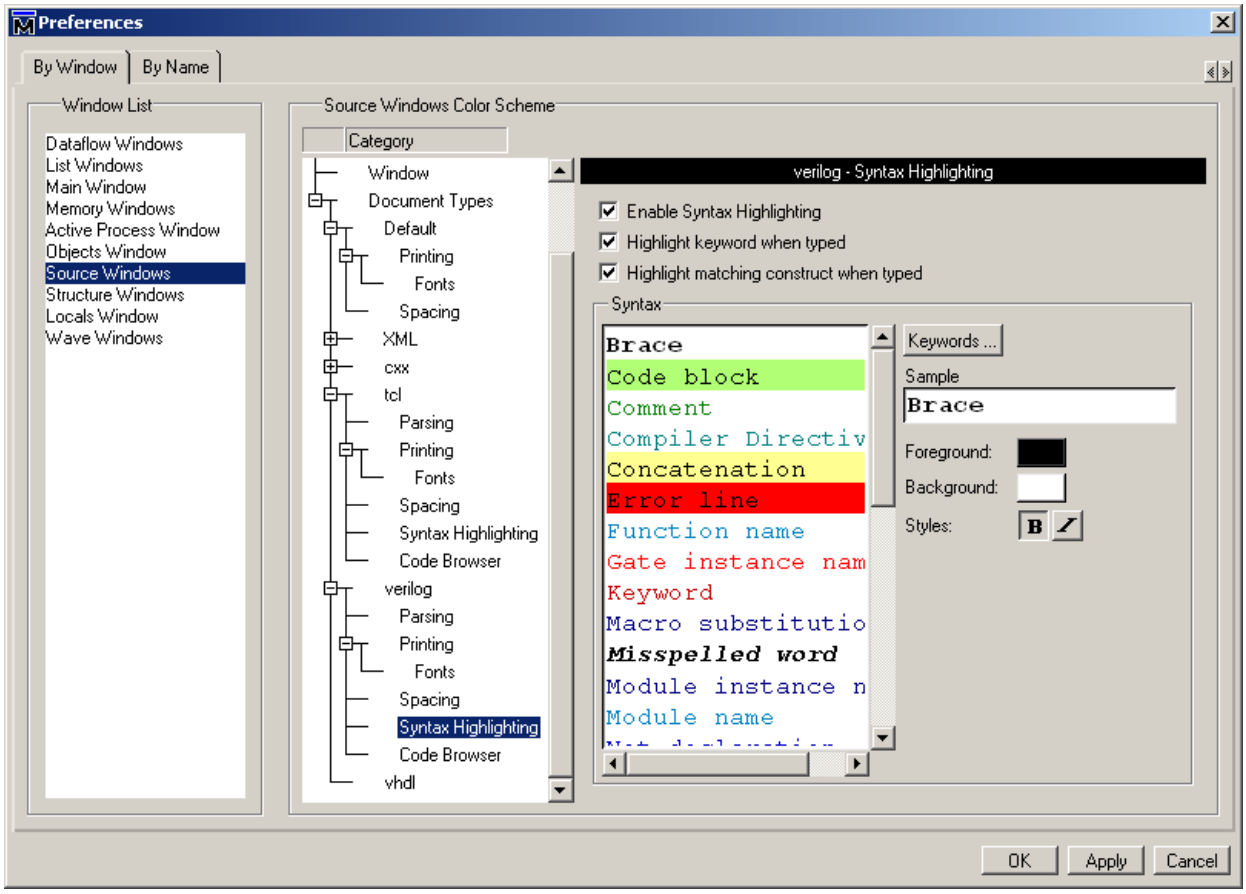
Marking lines with bookmarks

Source window bookmarks are blue circles that mark lines in a source file. These graphical icons may ease navigation through a large source file by "highlighting" certain lines.

As noted above in the discussion about finding text in the Source window, you can insert bookmarks on any line containing the text for which you are searching. The other method for inserting bookmarks is to right-click a line number and select **Add/Remove Bookmark**. To remove a bookmark, right-click the line number and select Add/Remove Bookmark again.

Customizing the Source window

You can customize a variety of settings for Source windows. For example, you can change fonts, spacing, colors, syntax highlighting, and so forth. To customize Source window settings, select **Tools > Edit Preferences**. This opens the Preferences dialog. Select **Source Windows** from the Window List.



Select an item from the Category list and then edit the available properties on the right. Click OK or Apply to accept the changes.

The changes will be active for the next Source window you open. The changes are saved automatically when you quit ModelSim.

Source window menus

The following menu commands are available in the Source window when it is undocked from the Main window. When the Source window is docked in the Main window, these functions can be found in the Main window menu bar (see Main window menu bar (GR-20)). Several commands are also available in a context menu by right-clicking in a Source window.

File menu

New	create a new VHDL, Verilog, SystemC, DO or other source file
Open	open a source file
Use Source	specify an alternative file to use for the current source file; this alternative source mapping exists for the current simulation only
Source Directory	add to a list of directories to search for source files; you can set this permanently using the **SourceDir** variable in the *modelsim.tcl* file
Save/Save As	save the active source file
Print	print the active source file
Close	close the active source file

Edit menu

Undo	undo previous action
Cut	cut selected source code
Copy	copy selected source code
Paste	paste code at selected location in the source code
Select All	select all source code in the active source file
Unselect All	unselect selected source code
Comment/ Uncomment Selected	add or remove comment characters from the selected lines
Find	find specific code in the active source file; allows you to match whole word only, match case, search for regular expressions, and search backwards in the active source file; you can also mark all occurrences
Replace	find and replace specified source code in the active source file; allows you to match whole word only, match case, search for and replace regular expressions, and search backwards in the active source file; you can also replace all occurrences

Previous Coverage Miss	jump to previous line of code not executed
Next Coverage Miss	jump to next line of code not executed
Goto	jump to a line number in the active source file
Read Only	toggle read only status of active source file

View menu

Show line numbers	toggle display of line numbers Show language templates
Show language templates	toggle display of the Language templates (GR-201) pane
Properties	list a variety of information about the source file; for example, file type, file size, file modification date

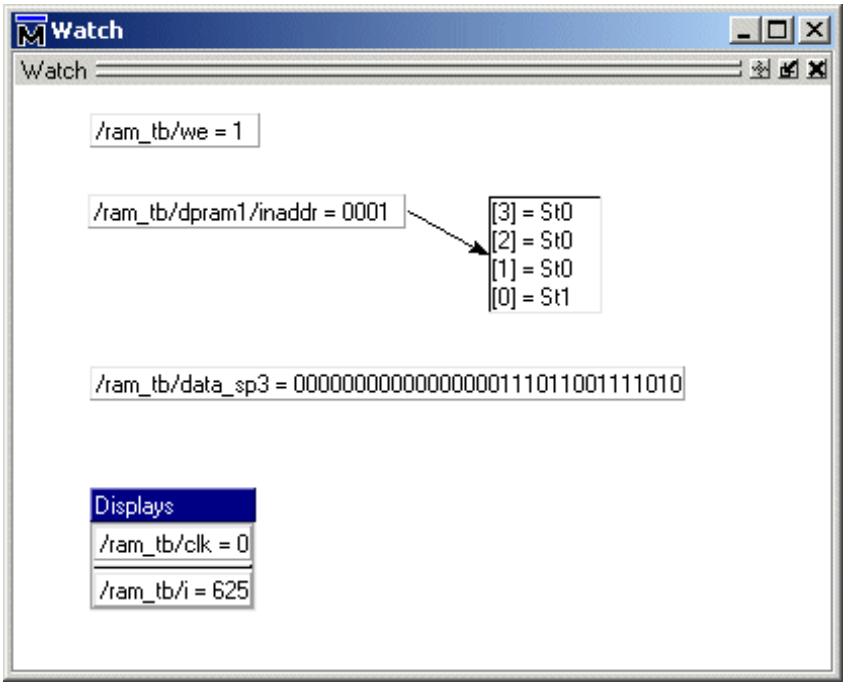
Tools menu

Examine	display the current value of the object selected in the active Source window; same as the **examine** (CR-162) command
Describe	display information about the object selected in the active Source window; same as the **describe** command (CR-147)
Drivers	list the names of all drivers of the object selected in the active Source window
Readers	list the names of all readers of the object selected in the active Source window
Compile	compile the active source file
C Debug	see "C Debug sub-menu" (GR-31)
Breakpoints	add, edit, or delete file-line and signal breakpoints; see "Creating and managing breakpoints" (GR-264)
Options	the Options menu includes these Source related options: Examine Now–examine selected object at the current simulation time; this option affects the behavior of the Examine and Describe commands as well as the examine popup; see "Checking object values and descriptions" (GR-203) Examine Current Cursor–examine selected object at the time of the current cursor in the Wave window; this option affects the behavior of the Examine and Describe commands as well as the examine popup; see "Checking object values and descriptions" (GR-203)

Watch pane

The Watch pane shows values for signals and variables at the current simulation time. Unlike the Objects or Locals pane, the Watch pane allows you to view any signal or variable in the design regardless of the current context.

.



Objects you can view

VHDL objects

signals, aliases, generics, constants, and variables

Verilog objects

nets, registers, variables, named events, and module parameters

SystemC objects

primitive channels and ports

Virtual objects

virtual signals and virtual functions

Adding objects to the pane

To add objects to the Watch pane, drag-and-drop objects from the Structure tab, Objects pane, or Locals pane. Alternatively, use the **add watch** command (CR-51).

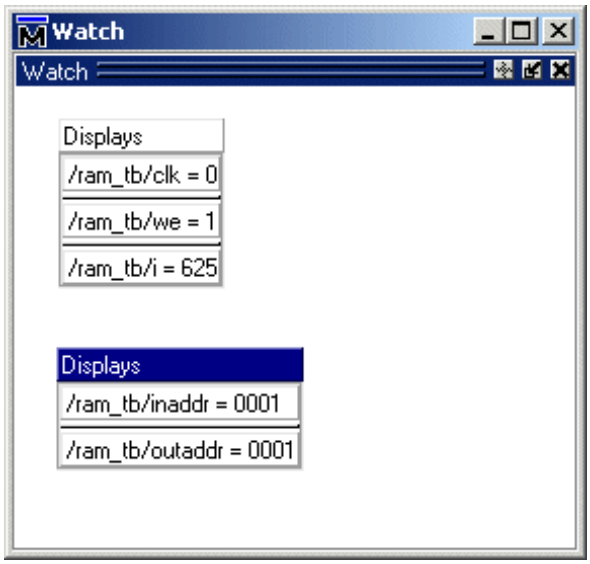
Expanding objects to show individual bits

If you add an array or record to the Watch pane, you can view individual bit values by double-clicking the array or record. As shown in the graphic above, */ram_tb/dpram1/inaddr* has been expanded to show all the individual bit values. Notice the arrow that "ties" the array to the individual bit display.

Grouping and ungrouping objects

You can group objects in the Watch pane so they display and move together. Select the objects, then right click one of the objects and choose Group.

In the graphic below, two different sets of objects have been grouped together.



To ungroup them, right-click the group and select Ungroup.

Saving and reloading format files

You can save a format file (a DO file, actually) that will redraw the contents of the Watch window. Right-click anywhere in the window and select **Save Format**.

Once you have saved the file, you can reload it by right-clicking and selecting **Load Format**.

Other Watch pane commands

The table below summarizes some other Watch pane commands, all of which are available via a context menu by clicking in the pane with the right mouse button.

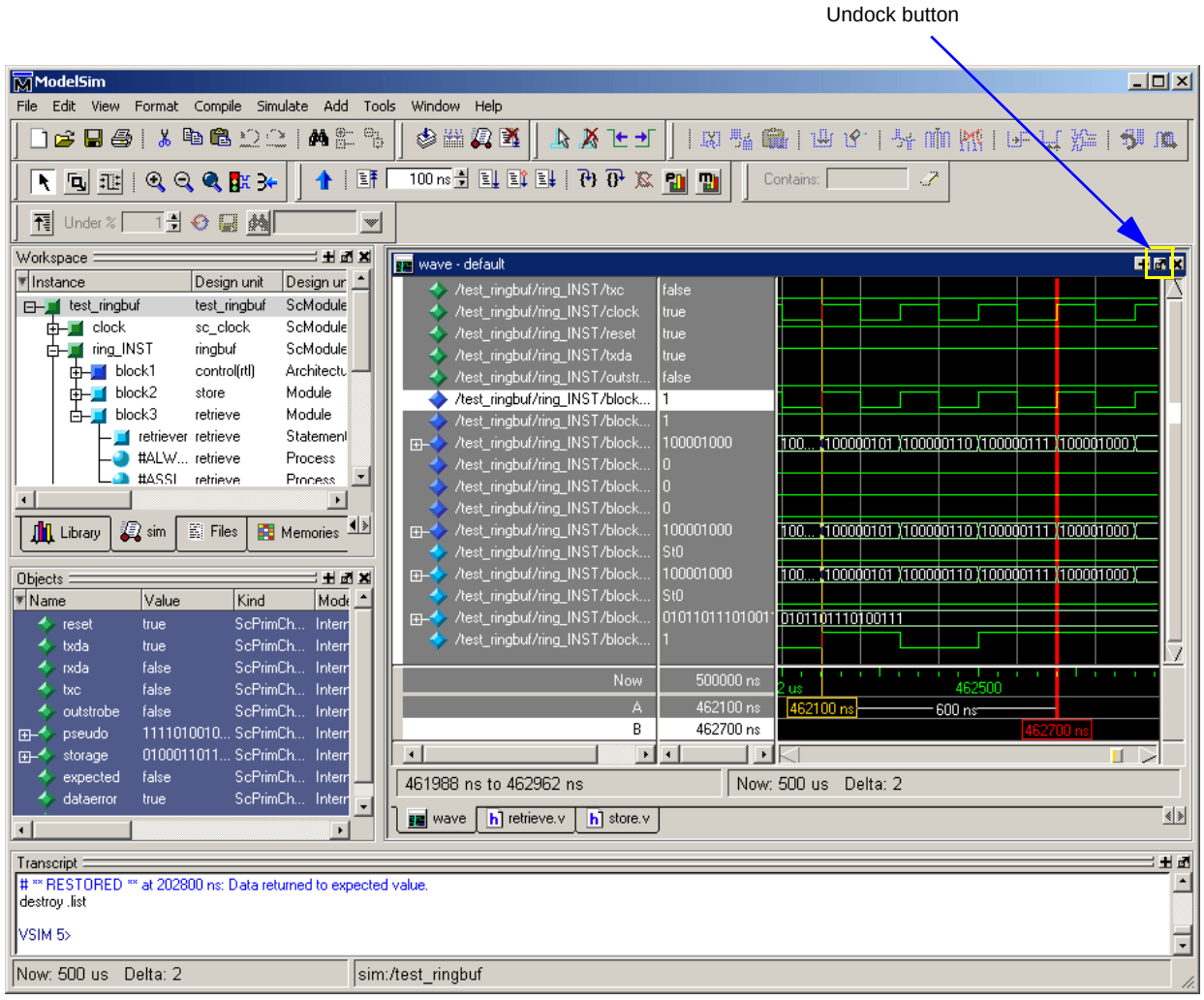
Command	Description
Add Wave	Add the selected items to the Wave window
Add List	Add the selected items to the List window
Log Signal	Add selected signals to the active log file

Command	Description
Force	Apply stimulus to a net or signal; see "Force Selected Signal dialog" (GR-186)
Noforce	Cancel a previous force command
Clock	Create a clock signal; see "Define Clock dialog" (GR-188)
Change	Modify the value of a variable
Follow Selection Context	Identifies location of object in design hierarchy in structure tab of Workspace
Save Format	Save the current Watch window display and signal preferences to a DO (macro) file; running the DO file will reformat the Watch window to match the display as it appeared when the DO file was created
Load Format	Executes or loads a previously saved DO file in the Watch window
Clear	Clear all objects from the Watch pane
Tile	Organizes objects in the pane
Group/Ungroup	Group or ungroup selected objects
Properties	Set the displayed radix of the selected signal
Clear	Clear the Watch window

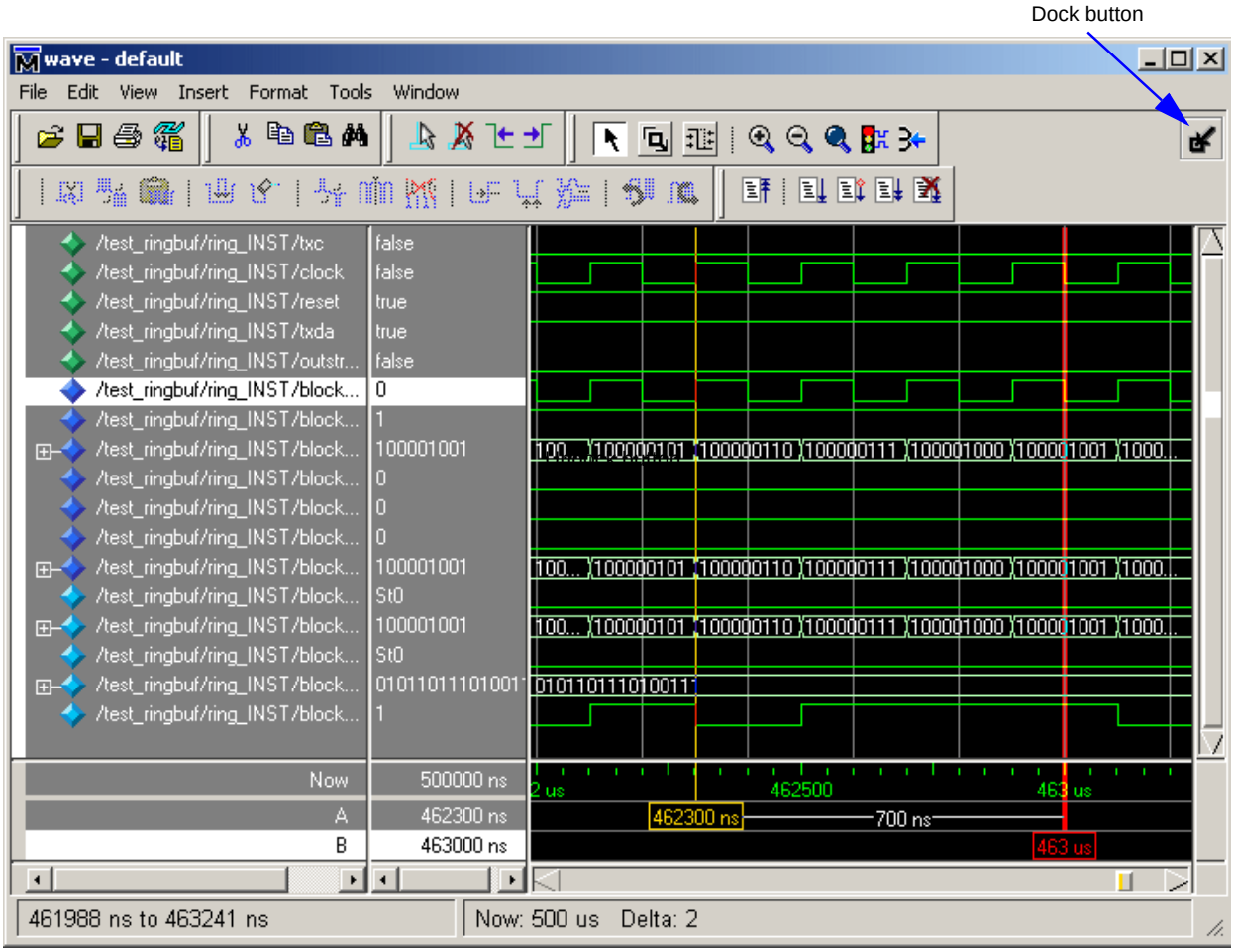
Wave window

The Wave window, like the List window, allows you to view the results of your simulation. In the Wave window, however, you can see the results as waveforms and their values.

The Wave window opens by default in the MDI frame of the Main window as shown below. The window can be undocked from the main window by pressing the Undock button in the window header or by using the **view -undock wave** command. The preference variable PrefWave(ViewUnDocked) can be used to control this default behavior. By setting the value of this variable to 1, the Wave Window will open undocked.

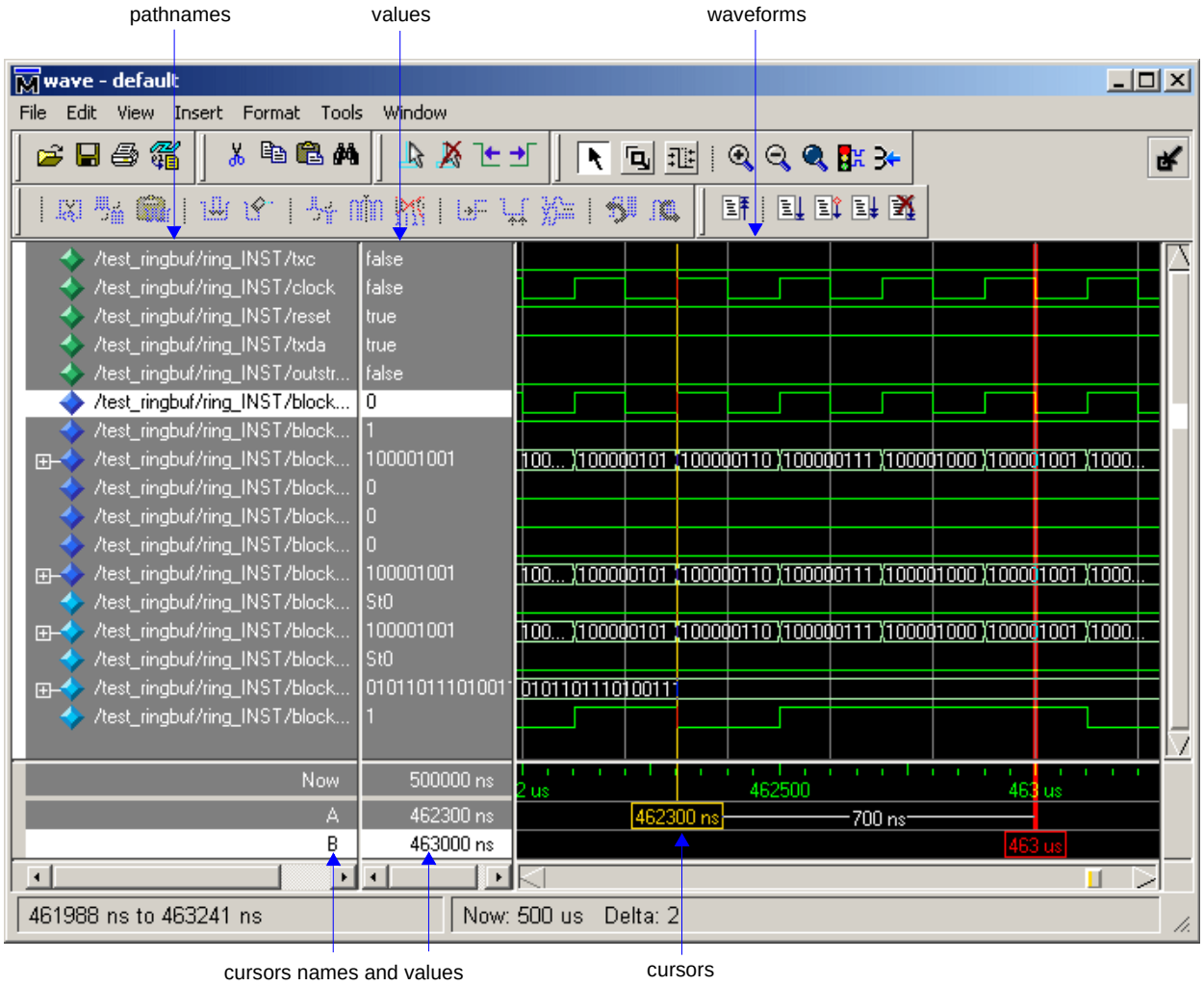


Here is an example of a Wave window that is undocked from the MDI frame. All menus and icons associated with Wave window functions now appear in the menu and toolbar areas of the Wave window.



If the Wave window is docked into the Main window MDI frame, all menus and icons that were in the standalone version of the Wave window move into the Main window menu bar and toolbar. See "Main window menu bar" (GR-20) for more information.

The Wave window is divided into a number of window panes. All window panes in the Wave window can be resized by clicking and dragging the bar between any two panes.



Wave window panes

The sections below describe the various Wave window panes.

Pathname pane

The pathname pane displays signal pathnames. Signals can be displayed with full pathnames, as shown here, or with only the leaf element displayed. You can increase the size of the pane by clicking and dragging on the right border. The selected signal is highlighted.

The white bar along the left margin indicates the selected dataset (see "Splitting Wave window panes" (UM-257)).

Value pane

The value pane displays the values of the displayed signals.

The radix for each signal can be symbolic, binary, octal, decimal, unsigned, hexadecimal, ASCII, or default. The default radix can be set by selecting **Simulate > Runtime Options**.

The data in this pane is similar to that shown in the Objects pane (GR-184), except that the values change dynamically whenever a cursor in the waveform pane is moved.

Waveform pane

The waveform pane displays the waveforms that correspond to the displayed signal pathnames. It also displays up to 20 cursors. Signal values can be displayed in analog step, analog interpolated, analog backstep, literal, logic, and event formats. Each signal can be formatted individually. The default format is logic.

If you rest your mouse pointer on a signal in the waveform pane, a popup displays with information about the signal. You can toggle this popup on and off in the **Wave Window Properties** dialog (see "Grid & Timeline tab" (GR-257)).

Cursor panes

There are three cursor panes–the left pane shows the cursor names; the middle pane shows the current simulation time and the value for each cursor; and the right pane shows the absolute time value for each cursor and relative time between cursors. Up to 20 cursors can be displayed. See "Measuring time with cursors in the Wave window" (UM-245) for more information.

Objects you can view

The following types of objects can be viewed in the Wave window

VHDL objects

(indicated by a dark blue diamond)
signals, aliases, process variables, and shared variables

Verilog objects

(indicated by a light blue diamond)
nets, registers, variables, and named events

SystemC objects

(indicated by a green diamond)
primitive channels and ports

Virtual objects

(indicated by an orange diamond)
virtual signals, buses, and functions, see; "Virtual Objects (User-defined buses, and more)" (UM-233) for more information

Comparison objects

(indicated by a yellow triangle)
comparison region and comparison signals; see "Waveform Compare" (UM-270) for more information

PSL assertions

(indicated by a magenta triangle)
see "Viewing assertions in the Wave window" (UM-382)

Coverage directives

(indicated by a magenta arrow)
see "Viewing coverage directives in the Wave window" (UM-389)

Created waveforms

(indicated by a red dot on a diamond)
see *Chapter 10 - Generating stimulus with Waveform Editor*

The data in the object values pane is very similar to the Objects window, except that the values change dynamically whenever a cursor in the waveform pane is moved.

At the bottom of the waveform pane you can see a time line, tick marks, and the time value of each cursor's position. As you click and drag to move a cursor, the time value at the cursor location is updated at the bottom of the cursor.

You can resize the window panes by clicking on the bar between them and dragging the bar to a new location.

Waveform and signal-name formatting are easily changed via the Format menu (GR-218). You can reuse any formatting changes you make by saving a Wave window format file (see "Saving the window format" (UM-261)).

Wave window menu bar

The following commands are available from the Wave window menu bar when the Wave window is undocked. When the Wave window is docked in the MDI frame (see "Wave window" (GR-211)), the commands are distributed amongst the Main window menus.

Many of these commands are also available via a context menu by clicking your right mouse button within the Wave window itself.

File menu

New Window	create a new instance of the Wave window
Open	open a dataset (WLF file)
Load	run a Wave window format (DO) file previously saved with Save Format
Save	provides these options: Dataset – save the current simulation to a WLF file Format – save the current Wave window display and signal preferences to a DO (macro) file; running the DO file will reformat the Wave window to match the display as it appeared when the DO file was created
Export	provides these options: Waveform – export a created waveform; see "Exporting waveforms to a stimulus file" (GR-294); Image – saves a bitmap image of the Wave window.
Import EVCD	import a previously exported EVCD file for waveform editor; see "Driving simulation with the saved stimulus file" (GR-295)
Page Setup	configure page setup including paper size, margins, label width, cursors, grid, color, scaling and orientation
Print (Windows only)	send the contents of the Wave window to a selected printer; see "Printing and saving waveforms in the Wave window" (UM-262) for details
Print Postscript	save or print the waveform display as a Postscript file; see "Saving a .eps file and printing under UNIX" (UM-262) for details
Close	close this copy of the Wave window
Quit	exit ModelSim

Edit menu

Cut	cut the selected object and waveform from the Wave window
Copy	copy the selected object and waveform
Paste	paste the previously cut or copied object above the currently selected object
Delete	delete the selected object and its waveform
Edit Wave	select from numerous options for editing waveforms; see "Editing waveforms" (GR-290)
Edit Cursor	open a dialog to specify the location of the selected cursor
Delete Cursor	delete the selected cursor from the window
Delete Window Pane	delete the selected window pane
Remove All (Panes and Signals)	removes all signals and additional window panes, leaving the window in its original state
Select All Unselect All	select, or unselect, all object names in the pathname pane
Find	find the specified object label within the pathname pane or the specified value within the value pane
Search	search the waveform display for a specified value, or the next transition for the selected object; see "Searching for values or transitions" (UM-252)
Force	force a value on the selected object; see "Force Selected Signal dialog" (GR-186)
Noforce	cancel a previous force command
Clock	define a clock; see "Define Clock dialog" (GR-188)

View menu

Zoom <selection>	selection: Full, In, Out, Last, or Range to change the waveform display range
Mouse Mode	toggle mouse pointer between Select Mode (click left mouse button to select, drag with middle mouse button to zoom) and Zoom Mode (drag with left mouse button to zoom, click middle mouse button to select)

Object Declaration	open the source file in the Source window and highlight the declaration for the currently selected object
Cursors	choose a cursor to go to from a list of available cursors
Bookmarks	choose a bookmark to go to from a list of available bookmarks
Goto Time	scroll the Wave window so the specified time is in view; "g" hotkey produces the same result
Sort	sort the top-level objects in the pathname pane; sort with full path or viewed name; use ascending or descending order
Justify Values	justify values to the left or right margins of the window pane
Refresh Display	clear the Wave window, empty the file cache, and rebuild the window from scratch
Properties	set properties for the selected object (use the Format menu to change individual properties)

Insert menu

Divider	insert a divider at the current location
Breakpoint	add a breakpoint on the selected signal; see "Signal breakpoints" (GR-264)
Bookmark	add a bookmark with the current zoom range and scroll location; see "Saving zoom range and scroll position with bookmarks" (UM-250)
Cursor	add a cursor to the waveform pane
Window Pane	split the pathname, values and waveform window panes to provide room for a new waveset

Format menu

Radix	set the selected objects' radix
Format	set the waveform format for the selected objects – Literal, Logic, Event, Analog
Color	set the color for the selected objects from a color palette
Height	set the waveform height in pixels for the selected objects

Tools menu

Waveform Compare	see "Waveform Compare sub-menu" (GR-29)
Breakpoints	add, edit, and delete signal breakpoints; see "Creating and managing breakpoints" (GR-264)
Bookmarks	add, edit, delete, and goto bookmarks; see "Saving zoom range and scroll position with bookmarks" (UM-250)
Dataset Snapshot	enable periodic saving of simulation data to a WLF file; see "Saving at intervals with Dataset Snapshot" (UM-231)
Combine Signals	combine the selected objects into a user-defined bus; see "Combining objects/creating busses" (UM-264)
Window Preferences	set various display properties such as signal path length, cursor snap distance, row margin, dataset prefixes, waveform popup, etc.

Window menu

The Window menu is identical in all windows. See "Window menu" (GR-32) for a description of the commands.

Wave window toolbar

The Wave window toolbar gives you quick access to these ModelSim commands and functions.

Wave window toolbar buttons			
Button		**Menu equivalent**	**Other options**
	Load Wave Format run a Wave window format (DO) file previously saved with Save Format	File > Open > Format	**do** wave.do see **do** command (CR-151)
	Save Wave Format save the current Wave window display and signal preferences to a do (macro) file	File > Save > Format	none
	Print print a user-selected range of the current Wave window display to a printer or a file	File > Print File > Print Postscript	none
	Export Waveform export a created waveform	File > Export > Waveform	none
	Cut cut the selected signal from the Wave window	Edit > Cut	right mouse in pathname pane > Cut
	Copy copy the signal selected in the pathname pane	Edit > Copy	right mouse in pathname pane > Copy
	Paste paste the copied signal above another selected signal	Edit > Paste	right mouse in pathname pane > Paste
	Find find a name or value in the Wave window	Edit > Find	<control-f> Windows <control-s> UNIX

Wave window toolbar buttons			
Button		**Menu equivalent**	**Other options**
	Insert Cursor add a cursor to the waveform pane	Insert > Cursor	right click in cursor pane and select New Cursor
	Delete Cursor delete the selected cursor from the window	Edit > Delete Cursor	right mouse in cursor pane > Delete Cursor n
	Find Previous Transition locate the previous signal value change for the selected signal	Edit > Search (Search Reverse)	keyboard: Shift + Tab **left** <arguments> see **left** command (CR-189)
	Find Next Transition locate the next signal value change for the selected signal	Edit > Search (Search Forward)	keyboard: Tab **right** <arguments> see **right** command (CR-250)
	Select Mode set mouse to Select Mode – click left mouse button to select, drag middle mouse button to zoom	View > Mouse Mode > Select Mode	none
	Zoom Mode set mouse to Zoom Mode – drag left mouse button to zoom, click middle mouse button to select	View > Mouse Mode > Zoom Mode	none
	Zoom In 2x zoom in by a factor of two from the current view	View > Zoom > Zoom In	keyboard: i I or + right mouse in wave pane > Zoom In
	Zoom Out 2x zoom out by a factor of two from current view	View > Zoom > Zoom Out	keyboard: o O or - right mouse in wave pane > Zoom Out
	Zoom Full zoom out to view the full range of the simulation from time 0 to the current time	View > Zoom > Zoom Full	keyboard: f or F right mouse in wave pane > Zoom Full
	Stop Wave Drawing halts any waves currently being drawn in the Wave window	none	.wave.tree interrupt

Wave window toolbar buttons			
Button		**Menu equivalent**	**Other options**
	Show Drivers display driver(s) of the selected signal, net, or register in the Dataflow window	[Dataflow window] Navigate > Expand net to drivers	[Dataflow window] Expand net to all drivers right mouse in wave pane > Show Drivers
	Restart reloads the design elements and resets the simulation time to zero, with the option of keeping the current formatting, breakpoints, and WLF file	Main menu: Simulate > Run > Restart	**restart** <arguments> see: **restart** (CR-246)
	Run run the current simulation for the default time length	Main menu: Simulate > Run > Run <default_length>	use the **run** command at the VSIM prompt see: **run** (CR-252)
	Continue Run continue the current simulation run	Main menu: Simulate > Run > Continue	use the **run -continue** command at the VSIM prompt see: **run** (CR-252)
	Run -All run the current simulation forever, or until it hits a breakpoint or specified break event	Main menu: Simulate > Run > Run -All	use the **run -all** command at the VSIM prompt see: **run** (CR-252), also see "Assertions tab" (GR-86)
	Break stop the current simulation run	none	none

Waveform editor toolbar

ModelSim's waveform editor has its own toolbar. The toolbar becomes active once you add an editable wave to the Wave window. See *Chapter 10 - Generating stimulus with Waveform Editor* in the *ModelSim User's Manual* for more details.

Waveform editor toolbar buttons			
Button		**Menu equivalent**	**Other options**
	Cut Wave cut the selected section of the waveform to the clipboard	Edit > Edit Wave > Cut	**wave edit cut** see **wave edit** command (CR-400)

Waveform editor toolbar buttons			
Button		**Menu equivalent**	**Other options**
	Copy Wave copy the selected section of the waveform to the clipboard	Edit > Edit Wave > Copy	**wave edit copy** see **wave edit** command (CR-400)
	Paste Wave paste the wave from the clipboard	Edit > Edit Wave > Paste	**wave edit paste** see **wave edit** command (CR-400)
	Insert Pulse Insert a transition at the selected time	Edit > Edit Wave > Insert Pulse	**wave edit insert_pulse** see **wave edit** command (CR-400)
	Delete Edge Delete the selected transition	Edit > Edit Wave > Delete Edge	**wave edit delete** see **wave edit** command (CR-400)
	Invert Invert the selected section of the waveform	Edit > Edit Wave > Invert	**wave edit invert** see **wave edit** command (CR-400)
	Mirror Mirror the selected section of the waveform	Edit > Edit Wave > Mirror	**wave edit mirror** see **wave edit** command (CR-400)
	Change Value Change the value of the selected section of the waveform	Edit > Edit Wave > Value	**wave edit change_value** see **wave edit** command (CR-400)
	Stretch Edge Move the selected edge by increasing/decreasing waveform duration	Edit > Edit Wave > Stretch Edge	**wave edit stretch** see **wave edit** command (CR-400)
	Move Edge Move the selected edge without increasing/decreasing waveform duration	Edit > Edit Wave > Move Edge	**wave edit move** see **wave edit** command (CR-400)
	Extend All Waves Increase the duration of all editable waves	Edit > Edit Wave > Extend All Waves	**wave edit extend** see **wave edit** command (CR-400)

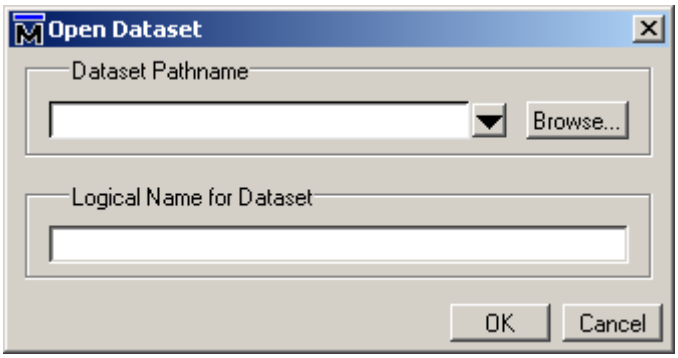
Waveform editor toolbar buttons			
Button		**Menu equivalent**	**Other options**
	Wave Undo Undo a previous waveform edit	Edit > Edit Wave > Undo	**wave edit undo** see **wave edit** command (CR-400)
	Wave Redo Redo a previously undone waveform edit	Edit > Edit Wave > Redo	**wave edit redo** see **wave edit** command (CR-400)

Wave window dialogs

This section describes the dialogs that are accessed via the Main window menu bar when the Wave window is docked, and via the Wave window menu bar when it is undocked. The dialogs are listed in the order in which they appear on the menus, top-to-bottom and left-to-right (i.e., starting with the File menu and progressing across the menu bar). Not all dialogs are documented (e.g., Open Format dialog).

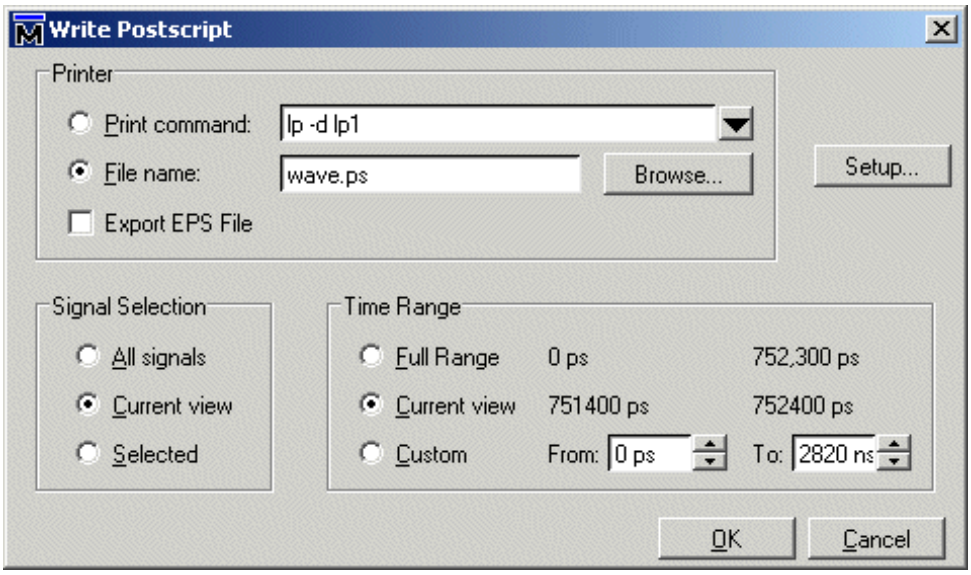
Open Dataset dialog

Purpose	Menu command	Additional information
Open a new dataset	**File** > **Open**	Opening datasets (UM-227)



Write Postscript dialog

Purpose	Menu command	Additional information
Print/save a postscript file	**File > Print Postscript**	"Printing and saving waveforms in the Wave window" (UM-262)



The Write Postscript dialog includes these options:

Printer

- **Print command**
  Enter a UNIX print command to print the waveform in a UNIX environment.
- **File name**
  Enter a filename to be created or browse to a previously created file and use that filename.
- **Export EPS File**
  Save the file in encapsulated postscript (EPS) format.

Signal Selection

- **All signals**
  Print all signals.
- **Current View**
  Print signals in the current view.
- **Selected**
  Print all selected signals.

Time Range

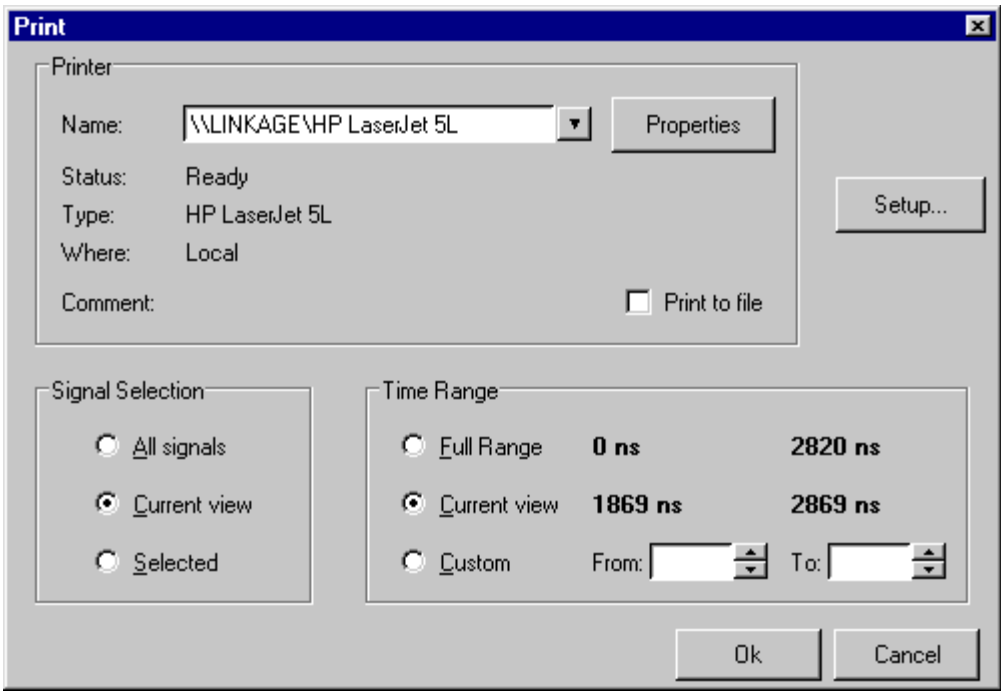
- **Full Range**
  Print all specified signals in the full simulation range.
- **Current view**
  Print the specified signals for the viewable time range.
- **Custom**
  Print the specified signals for a user-designated **From** and **To** time.

Setup button

See "Page Setup dialog" (GR-230)

Print dialog

Purpose	Menu command	Additional information
Print the Wave window contents	**File > Print**	"Printing and saving waveforms in the Wave window" (UM-262)



The Print dialog includes these options:

Printer

- **Name**
  Choose the printer from the drop-down menu. Set printer properties with the **Properties** button.
- **Status**
  Indicates the availability of the selected printer.
- **Type**
  Printer driver name for the selected printer. The driver determines what type of file is output if "Print to file" is selected.
- **Where**
  The printer port for the selected printer.
- **Comment**
  The printer comment from the printer properties dialog.

- **Print to file**
  Make this selection to print the waveform to a file instead of a printer. The printer driver determines what type of file is created. Postscript printers create a Postscript (.ps) file, non-Postscript printers create a .prn or printer control language file. To create an encapsulated Postscript file (.eps) use the **File** > **Print Postscript** menu selection.

Signal Selection

- **All signals**
  Print all signals.
- **Current View**
  Print signals in current view.
- **Selected**
  Print all selected signals.

Time Range

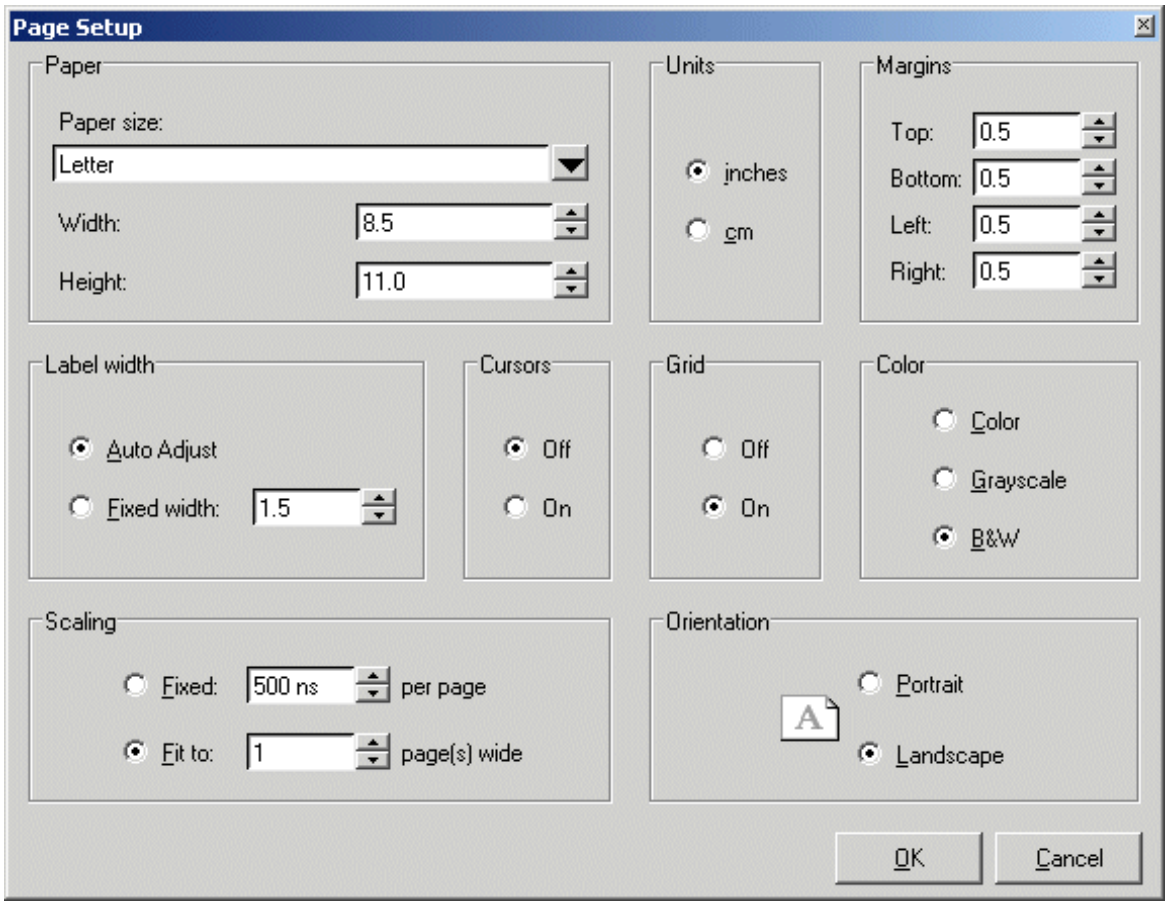
- **Full Range**
  Print all specified signals in the full simulation range.
- **Current view**
  Print the specified signals for the viewable time range.
- **Custom**
  Print the specified signals for a user-designated **From** and **To** time.

Setup button

See "Page Setup dialog" (GR-230)

Page Setup dialog

Purpose	Menu command	Additional information
Set up page layout for printing	**File > Page Setup**	"Printing and saving waveforms in the Wave window" (UM-262)



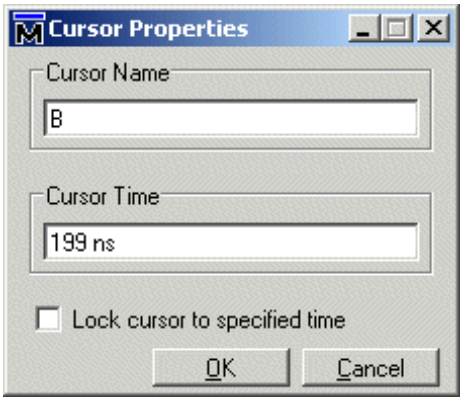
The Page Setup dialog includes these options:

- **Paper Size**
  Select your output page size from a number of options; also choose the paper width and height.
- **Units**
  Specify whether measurements are in inches or centimeters.
- **Margins**
  Specify the page margins; changing the **Margin** will change the **Scale** and **Page** specifications.
- **Label width**
  Specify Auto Adjust to accommodate any length label, or set a fixed label width.

- **Cursors**
  Turn printing of cursors on or off.
- **Grid**
  Turn printing of grid lines on or off.
- **Color**
  Select full color printing, grayscale, or black and white.
- **Scaling**
  Specify a **Fixed** output time width in nanoseconds per page – the number of pages output is automatically computed; or, select **Fit to** define the number of pages to be output based on the paper size and time settings; if set, the time-width per page is automatically computed.
- **Orientation**
  Select the output page orientation, **Portrait** or **Landscape**.

Cursor Properties dialog

Purpose	Menu command	Additional information
Name, move, or lock a cursor	**Edit** > **Edit Cursor**	"Measuring time with cursors in the Wave window" (UM-245)

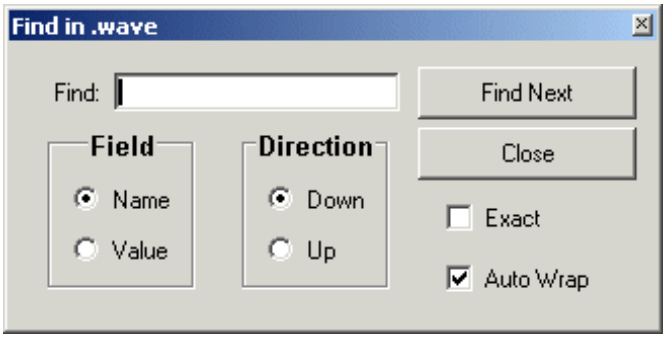


The Cursor Properties dialog includes these options:

- **Cursor Name**
  The name of the selected cursor.
- **Cursor Time**
  The position of the selected cursor.
- **Lock cursor to specified time**
  Freezes the selected cursor so it will not move. A cursor will turn red in the Wave window when it is locked. See "Working with cursors" (UM-245) for more information.

Find in .wave dialog

Purpose	Menu command	Additional information
Locate objects	**Edit > Find**	"Searching in the Wave and List windows" (UM-251)



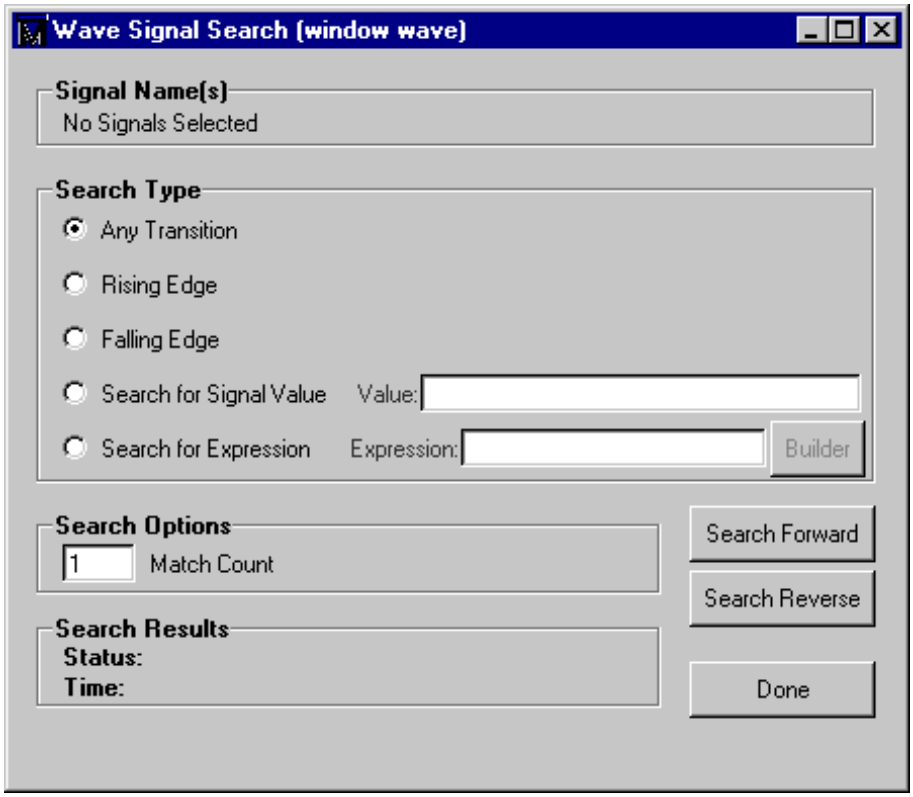
The Find in .wave dialog includes these options:

- **Find**
  Specify the text for which you want to search.
- **Field**
  Specify whether to search the name or value column.
- **Direction**
  Specify the direction to begin searching.
- **Exact**
  Check **Exact** if you only want to find objects that match your search exactly. For example searching for "addr" without Exact will find *addr* and *addr_r*.
- **Auto Wrap**
  Check **Auto Wrap** to continue the search at the top or bottom of the window.

The find operation works only within the active pane.

Wave Signal Search dialog

Purpose	Menu command	Additional information
Locate values and transitions	**Edit > Search**	"Searching in the Wave and List windows" (UM-251)



The Wave Signal Search dialog includes these options:

- **Signal Name(s)**
  A list of the objects currently selected in the Wave window. These objects are the subject of the search.
- **Any Transition**
  Searches for any transition in the selected signal(s).
- **Rising Edge**
  Searches for rising edges in the selected signal(s).
- **Falling Edge**
  Searches for falling edges in the selected signal(s).

- **Search for Signal Value**
  Searches for the value specified in the **Value** field; the value should be formatted using VHDL or Verilog numbering conventions (see "Numbering conventions" (CR-20) for more information).

  Note: If your signal values are displayed in binary radix, see "Searching for binary signal values in the GUI" (CR-29) for details on how signal values are mapped between a binary radix and std_logic.

- **Search for Expression**
  Searches for the expression specified in the **Expression** field evaluating to a boolean true. See "Using the Expression Builder for expression searches" (UM-253) for information on the Builder button.

  The expression can involve more than one signal but is limited to signals logged in the Wave window. Expressions can include constants, variables, and DO files. If no expression is specified, the search will give an error. See "Expression syntax" (CR-23) for more information.

- **Match Count**
  Indicates the number of transitions or matches to search.You can search for the nth transition or the nth match on value.

The Search Results are indicated at the bottom of the dialog.

Force Selected Signal dialog

This is the same dialog that is accessible via the Objects window. See "Force Selected Signal dialog" (GR-186) for more information.

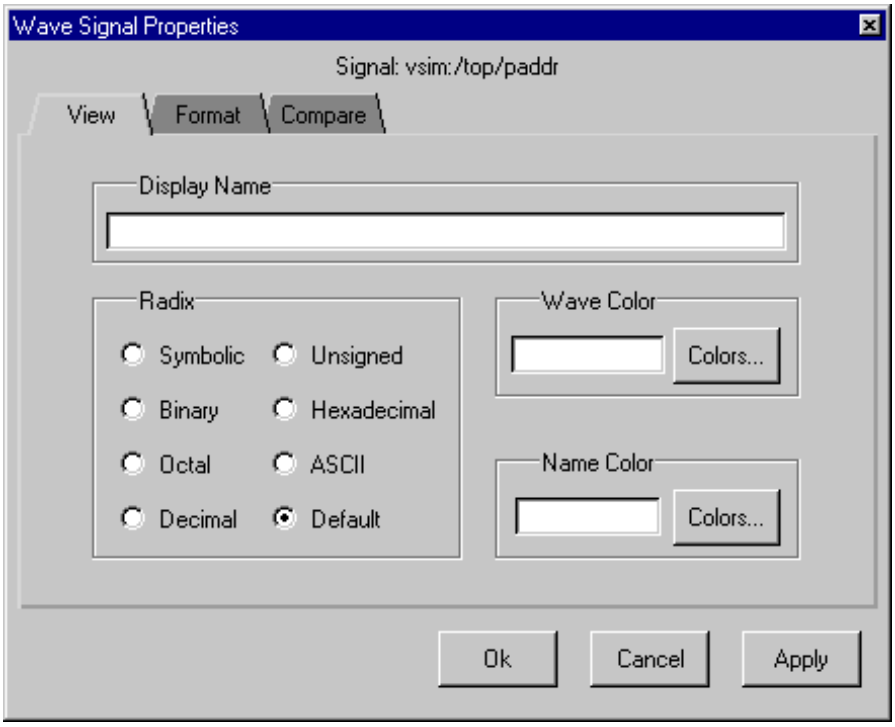
Define Clock dialog

This is the same dialog that is accessible via the Objects window. See "Define Clock dialog" (GR-188) for more information.

Wave Signal Properties dialog

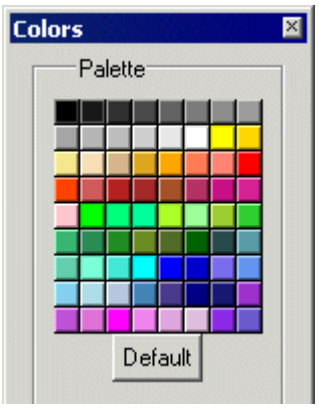
Purpose	Menu command	Additional information
Format object display properties	**View > Properties**	"Formatting the Wave window" (UM-255)

View tab



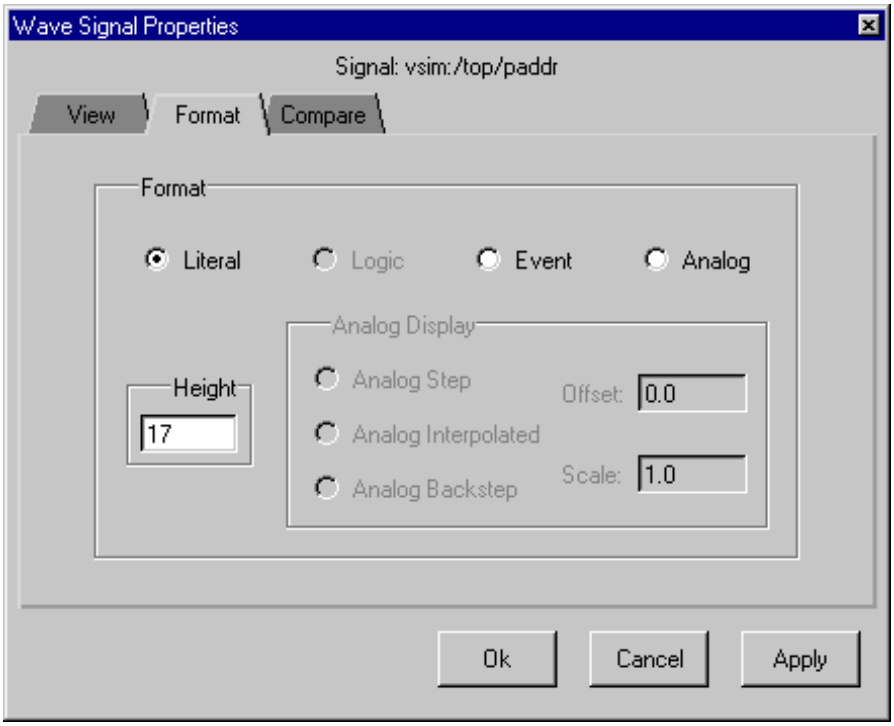
The View tab includes these options:

- **Display Name**
  Specifies a new name (in the pathname pane) for the selected signal.
- **Radix**
  Specifies the Radix of the selected signal(s). See "Changing radix (base)" (UM-255).



- **Wave Color**
  Specifies the waveform color. Select a new color from the color palette, or enter a color name. The Default button in the Colors palette allows you to return the selected object's color back to its default value.
- **Name Color**
  Specifies the signal name's color. Select a new color from the color palette, or enter a color name. The Default button in the Colors palette allows you to return the selected object's color back to its default value.

Format tab



The Format tab includes these options:

Format

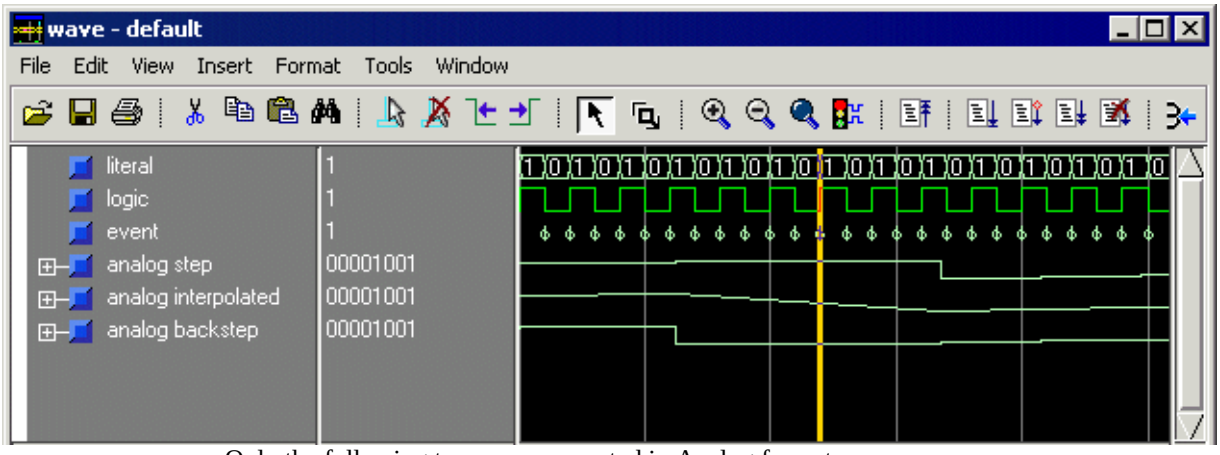
- **Literal**
  Displays the waveform as a box containing the object value (if the value fits the space available). This is the only format that can be used to list a record.
- **Logic**
  Displays values as U, X, 0, 1, Z, W, L, H, or -.
- **Event**
  Marks each transition during the simulation run.

- **Analog**
  Displays the values in one of three analog styles, as detailed below.
- **Height**
  Allows you to specify the height (in pixels) of the waveform.

Analog

- **Analog Step**
  Displays the waveform in step style.
- **Analog Interpolated**
  Displays the waveform in interpolated style.
- **Analog Backstep**
  Displays the waveform in backstep style. Often used for power calculations.
- **Offset**
  Allows you to adjust the scale of the object as it is seen on the display. Offset is the number of pixels offset from zero.
- **Scale**
  Reduces (if less than 1) or increases (if greater than 1) the number of pixels displayed.

The signals in the following illustration demonstrate the various signal formats.



Only the following types are supported in Analog format:

- **VHDL types:**
  All vectors - std logic vectors, bit vectors, and vectors derived from these types
  Scalar integers
  Scalar reals
  Scalar times
- **Verilog types:**
  All vectors
  Scalar reals
  Scalar integers
- **SystemC types:**
  Vector types (sc_int<>, sc_bigint<>, etc.)
  Scalar integers (char, short, int, long, etc.)
  float, double

Compare tab

The **Compare** tab includes the same options as those in the "Add Signal Options dialog" (GR-244).

Wave Divider Properties dialog

Purpose	Menu command	Additional information
Divide the Wave window	**Insert > Divider**	"Dividing the Wave window" (UM-256)

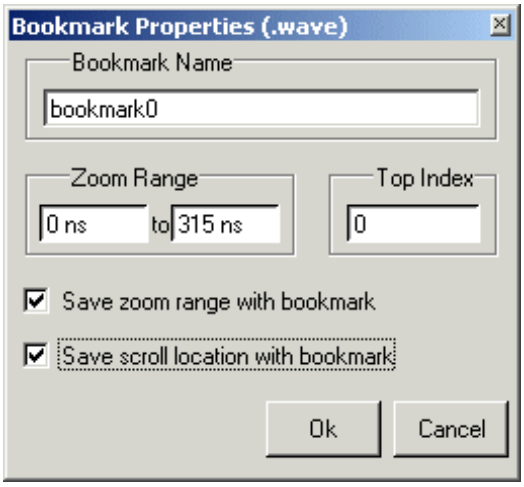
When the Wave window is docked, this selection is available through the right-click menu.

The Wave Divider Properties dialog includes these options:

- **Divider Name**
  The name of the divider. Default is "New Divider". A dash "-" cannot be used as the first character of the Divider Name. A space may be used as the first character, and it may be followed by a dash. For example, "-Test Signals-" is not permitted as a Divider Name but " -Test Signals- " is permitted.
- **Divider Height**
  The height of the divider in pixels.

Bookmark Properties dialog

Purpose	Menu command	Additional information
Save zoom and scroll position	(window docked) **Add** > **Bookmark** (window undocked) **Insert** > **Bookmark**	"Saving zoom range and scroll position with bookmarks" (UM-250)

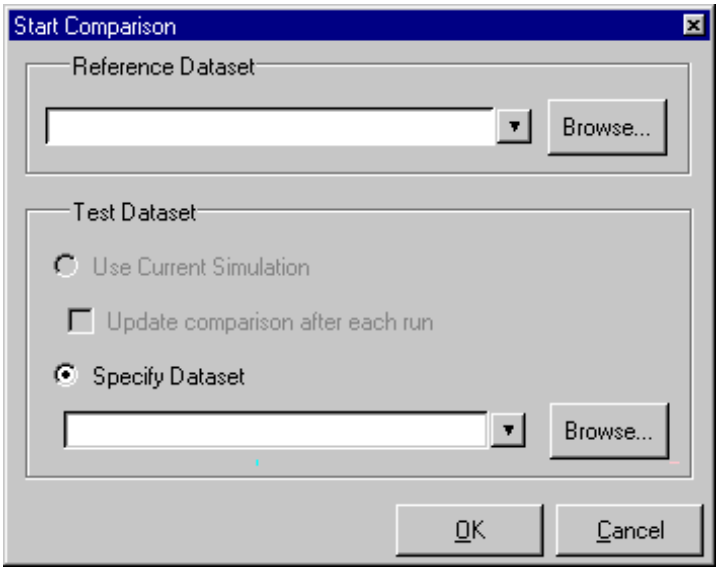


The Bookmark Properties dialog includes the following options.

- **Bookmark Name**
  A text label to assign to the bookmark. The name will identify the bookmark on the **View** > **Bookmarks** menu.
- **Zoom Range**
  A starting value and ending value that define the zoom range.
- **Top Index**
  The object that will display at the top of the Wave window. For instance, if you specify 15, the Wave window will be scrolled down to show the 15th object in the window.
- **Save zoom range with bookmark**
  When checked the zoom range will be saved in the bookmark.
- **Save scroll location with bookmark**
  When checked the scroll location will be saved in the bookmark.

Start Comparison dialog

Purpose	Menu command	Additional information
Start a waveform comparison	**Tools > Waveform Compare > Start Comparison**	"Setting up a comparison with the GUI" (UM-271)



The Start Comparison dialog includes the following options.

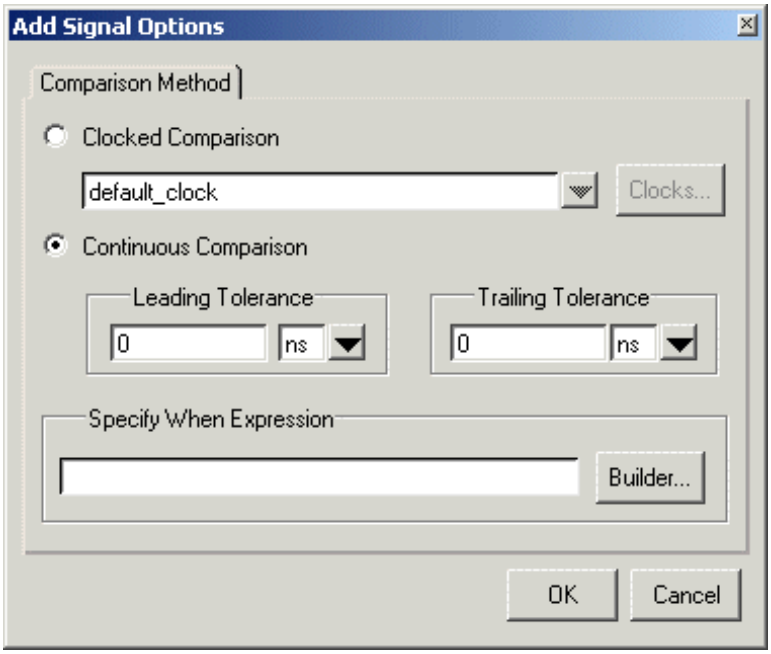
- **Reference Dataset**
  The dataset to which the test dataset will be compared. It can be a saved dataset, the current simulation dataset, or any part of the current simulation dataset.

Test Dataset

- **Use Current Simulation**
  Uses the current active simulation as the test dataset.
- **Update comparison after each run**
  Tracks the current simulation, updating differences after every run command. If you wish to see differences soon after they occur, use many relatively short run commands.
- **Specify Dataset**
  Enter the name of an existing dataset to compare against the reference dataset.

Add Signal Options dialog

Purpose	Menu command	Additional information
Specify comparison method	**Tools > Waveform Compare > Add > Compare by Signal**	"Specifying the comparison method" (UM-276)

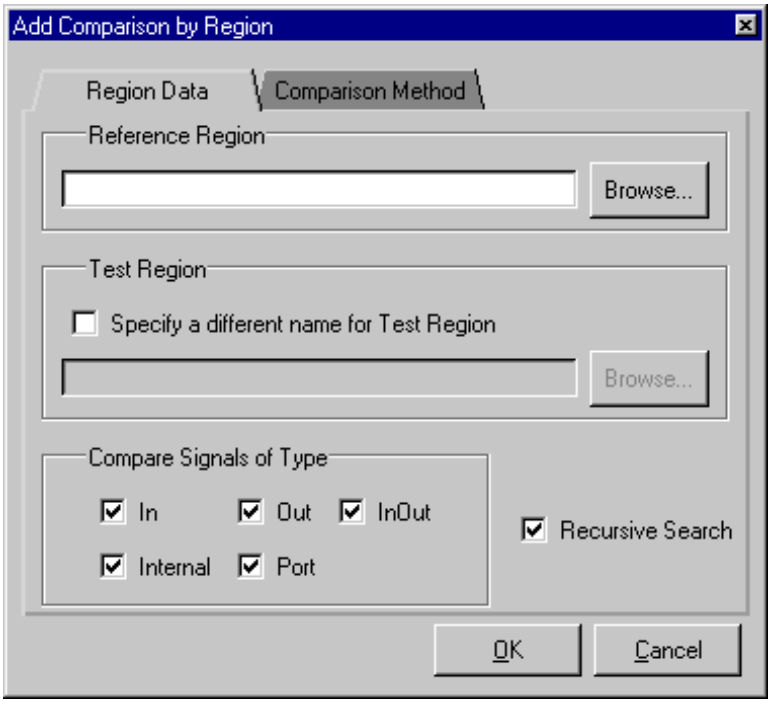


The Add Signal Options dialog includes these options:

- **Clocked Comparison**
  Compare the datasets only at or just after an edge on some signal. In this mode, you define one or more clocks. The test signal is compared to a reference signal and both are sampled relative to the defined clock.
- **Continuous Comparison**
  Compare the test signals to the reference signals at each transition of the reference.
- **Leading Tolerance**
  How much earlier the test signal edge may occur before the reference signal edge. May be specified differently for each signal compared.
- **Trailing Tolerance**
  How much later the test signal edge may occur after the reference signal edge. May be specified differently for each signal compared.
- **Specify When Expression**
  A conditional expression that must evaluate to "true" or 1 at the signal edge for the comparison to become effective. See "GUI_expression_format" (CR-22) for legal expression syntax.

Add Comparison by Region dialog

Purpose	Menu command	Additional information
Add a region to a waveform comparison	**Tools > Waveform Compare > Add > Compare by Region**	"Adding signals, regions, and clocks" (UM-274)



The Add Comparison by Region dialog includes these options:

Region Data tab

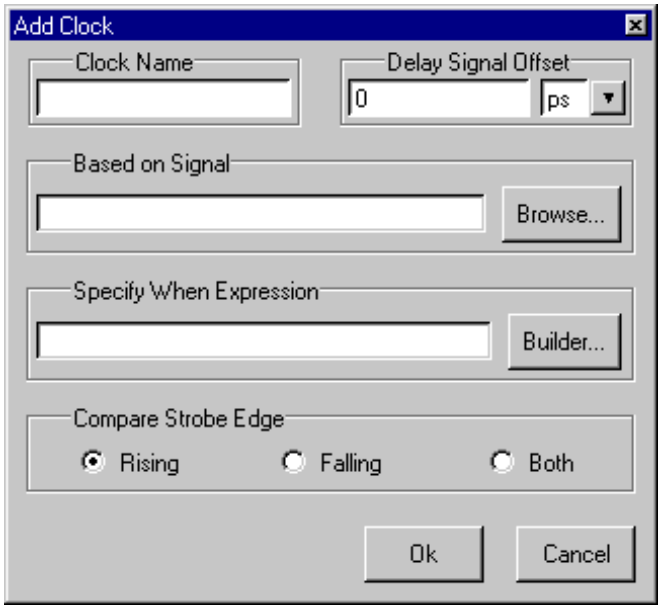
- **Reference Region**
  The reference region that will be used in the comparison.
- **Test Region**
  Use this if you have a test region that has a different name than the reference region.
- **Compare Signals of Type**
  The types of signals to compare.
- **Recursive Search**
  Check if you wish to search for signals in the hierarchy below the specified region.

Comparison Method tab

The Comparison Method tab includes the same options as those in the "Add Signal Options dialog" (GR-244).

Add Clocks dialog

Purpose	Menu command	Additional information
Add a clock for a clocked comparison	**Tools** > **Waveform Compare** > **Add** > **Clocks**	"Adding signals, regions, and clocks" (UM-274)



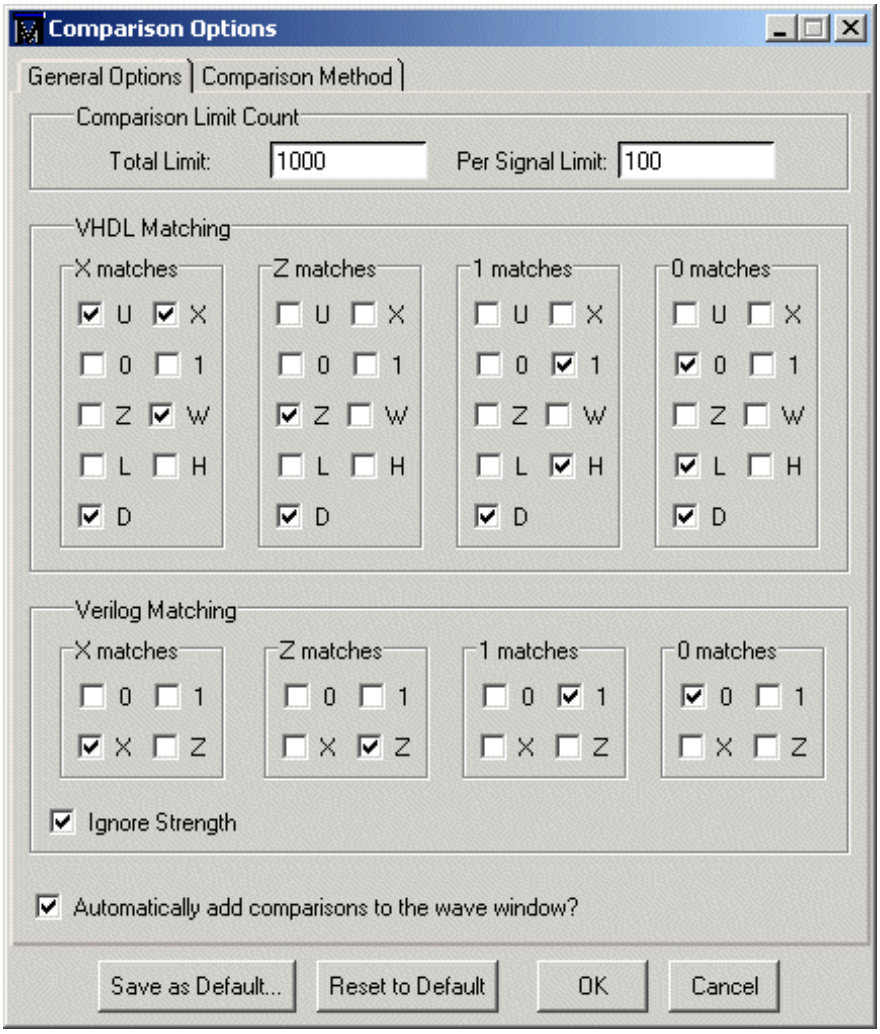
The Add Clock dialog includes these options:

- **Clock Name**
  A name for the clock.
- **Delay Signal Offset**
  A time value for delaying the sample time beyond the specified signal edge.
- **Based on Signal**
  The signal whose edge(s) is to be used as the strobe trigger.
- **Specify When Expression**
  A conditional expression that must evaluate to "true" or "1" for the clock edge to be used as a strobe. Optional. The expression is evaluated at the time of the clock edge rather than after the delay has been applied. See "GUI_expression_format" (CR-22) for legal expression syntax.
- **Compare Strobe Edge**
  The edge(s) of the specified signal that should be used for the strobe trigger.

Comparison Options dialog

Purpose	Menu command	Additional information
Configure waveform comparison option	**Tools > Waveform Compare > Options**	"Setting compare options" (UM-278)

General Options tab



The General Options tab includes these options:

- **Comparison Limit Count**
  Limit the comparison to a specific number of total differences and/or a specific number of differences per signal.
- **VHDL Matching**
  Designate which VHDL signal values will match X, Z, 1, and 0 values.

- **Verilog Matching**
  Designate which Verilog signal values will match X, Z, 1, and 0 values.
- **Ignore Strength**
  Ignore the strength of the Verilog signal and consider only logic values.
- **Automatically add comparisons to the Wave window?**
  If checked new signal comparison objects are added automatically to the Wave window.
- **Save as Default**
  Saves the settings to a *modelsim.tcl* file so they become the defaults for future comparisons. See "Preference variables located in Tcl files" (UM-540) for more information.
- **Reset to Default**
  Reset the dialog to the default options with which ModelSim ships.

Comparison Method tab

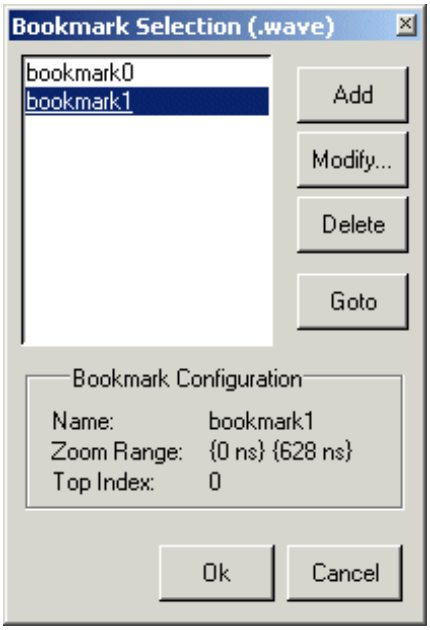
The Comparison Method tab includes the same options as those in the "Add Signal Options dialog" (GR-244).

Modify Breakpoints dialog

This is the same dialog that is accessible via the Main window. See "Modify Breakpoints dialog" (GR-95) for more information.

Bookmark Selection dialog

Purpose	Menu command	Additional information
Add/edit/delete bookmarks	**Tools** > **Bookmarks**	"Saving zoom range and scroll position with bookmarks" (UM-250)

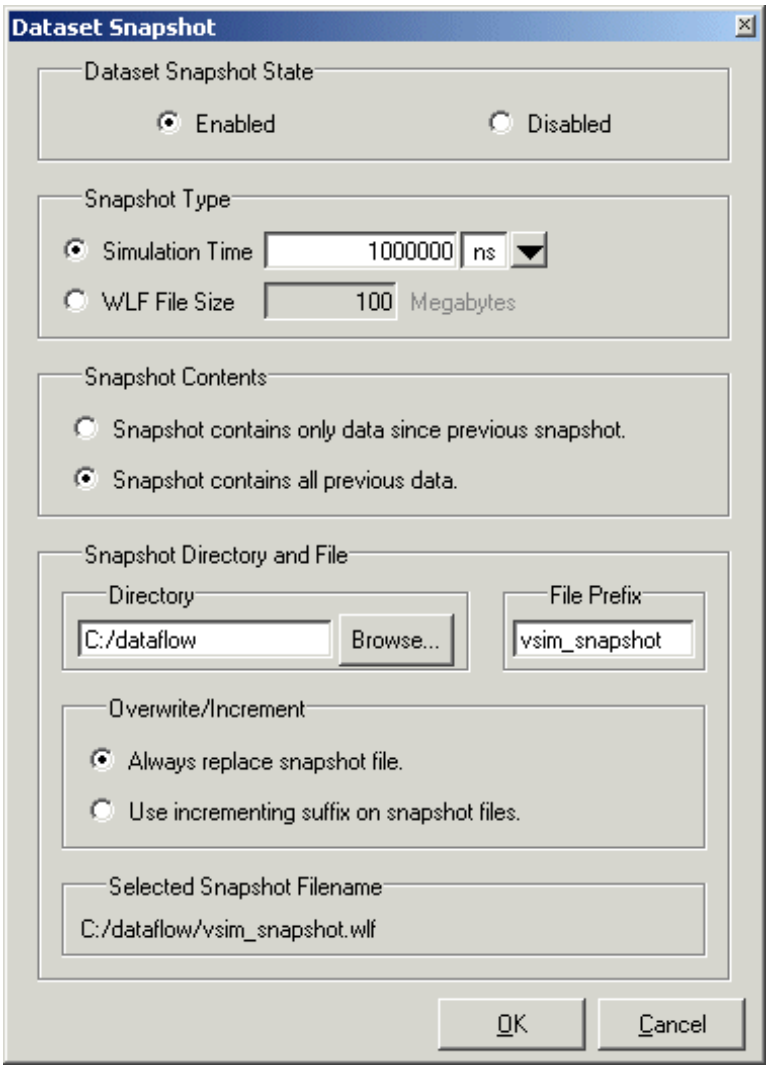


The Bookmark Selection dialog includes the following options.

- **Add**
  Add a new bookmark.
- **Modify**
  Edit the selected bookmark.
- **Delete**
  Delete the selected bookmark.
- **Goto**
  Zoom and scroll the Wave window using the selected bookmark.

Dataset Snapshot dialog

Purpose	Menu command	Additional information
Save simulation results at regular intervals	**Tools > Dataset Snapshot**	"Saving at intervals with Dataset Snapshot" (UM-231)



The Dataset Snapshot dialog includes these options:

Dataset Snapshot State

- **Enabled/Disabled**
  Enable or disable Dataset Snapshot. All other dialog options are unavailable if Disabled is selected.

Snapshot Type

- **Simulation Time**
  Specifies that data is copied to the specified snapshot file every <x> time units. Default is 1000000 time units.
- **WLF File Size**
  Specifies that data is copied to the specified snapshot file whenever the current simulation WLF file reaches <*x*> megabytes. Default is 100 MB.

Snapshot Contents

- **Snapshot contains only data since previous snapshot**
  Specifies that each snapshot contains only data since the last snapshot. This option causes ModelSim to clear the current simulation WLF file each time a snapshot is taken.
- **Snapshot contains all previous data**
  Specifies that each snapshot contains all data from the time signals were first logged. The entire contents of the current simulation WLF file are saved each time a snapshot is taken.

Snapshot Directory and File

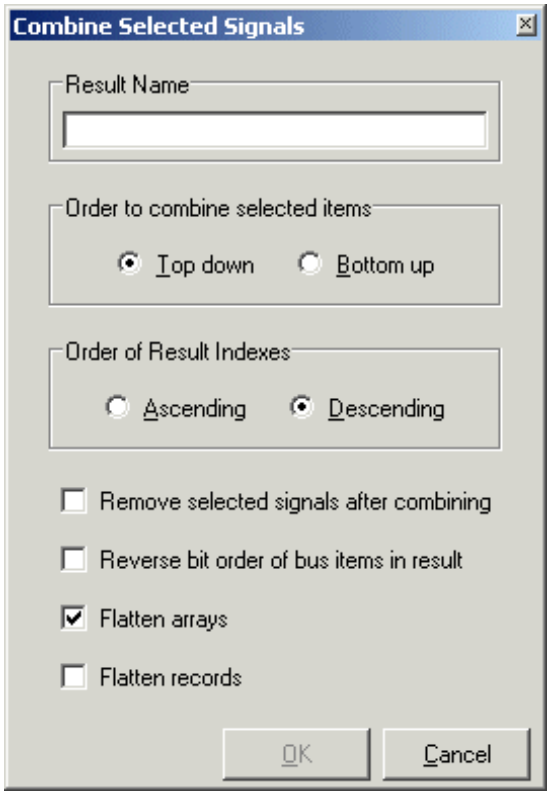
- **Directory**
  The directory in which ModelSim saves the snapshot files.
- **File Prefix**
  The name of the snapshot files. ModelSim adds *.wlf* to the snapshot files.

Overwrite / Increment

- **Always replace snapshot file**
  Specifies that a single file is created for all snapshots. Each new snapshot overwrites the previous.
- **Use incrementing suffix on snapshot files**
  Specifies that a new file is created for each snapshot. Each new snapshot creates a separate file (e.g., *vsim_snapshot_1.wlf*, *vsim_snapshot_2.wlf*, etc.).

Combine Selected Signals dialog

Purpose	Menu command	Additional information
Create busses from individual signals	**Tools > Combine Signals**	"Combining objects/creating busses" (UM-264)



The Combine Selected Signals dialog includes these options:

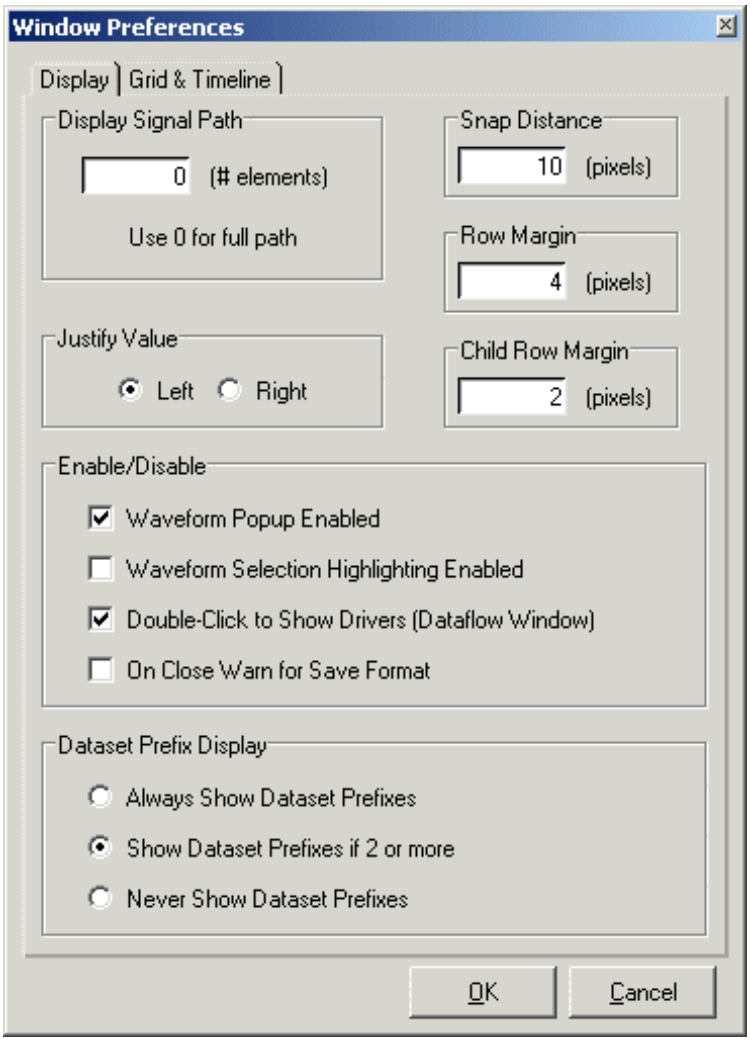
- **Result Name**
  Specifies the name of the newly created bus.
- **Order to combine selected items**
  Specifies the order in which to combine the selected signals. "Top down" specifies that the selected signals are ordered as they appear top-to-bottom in the Wave window. "Bottom up" reverses the order.
- **Order of Result Indexes**
  Specifies in which order the selected signals are indexed in the bus. If set to Ascending, the first signal selected in the Wave window will be assigned an index of 0. If set to Descending, the first signal selected will be assigned the highest index number.
- **Remove selected signals after combining**
  Specifies whether you want to remove the selected signals from the Wave window once the bus is created.

- **Reverse bit order of bus items in the result**
  If checked, the bits of each selected signal are reversed in the newly created bus. The order of the signals in the bus is not affected.
- **Flatten arrays**
  If checked, ModelSim combines the signals into one big array. If unchecked, ModelSim combines signals together without merging them into one array. The signals become elements of a record and retain their original names. When expanded, the new signal looks just like a group of signals.
- **Flatten records**
  If checked, causes elements of a record type signal to be pulled up to the top level.

Window Preferences dialog

Purpose	Menu command	Additional information
Configure window preferences	(window docked) **Tools > Options > Wave Preferences** (window undocked) **Tools > Window Preferences**	"Formatting the Wave window" (UM-255)

Display tab



The Display tab includes the following options:

- **Display Signal Path**
  Sets the display to show anything from the full pathname of each signal (e.g., *sim:/top/*

clk) to only its leaf element (e.g., *sim:clk*). A non-zero number indicates the number of path elements to be displayed. The default is Full Path.

- **Justify Value**
  Specifies whether the signal values will be justified to the left margin or the right margin in the values window pane.
- **Snap Distance**
  Specifies the distance the cursor needs to be placed from an object edge to jump to that edge (a 0 specification turns off the snap).
- **Row Margin**
  Specifies the distance in pixels between top-level signals.
- **Child Row Margin**
  Specifies the distance in pixels between child signals.

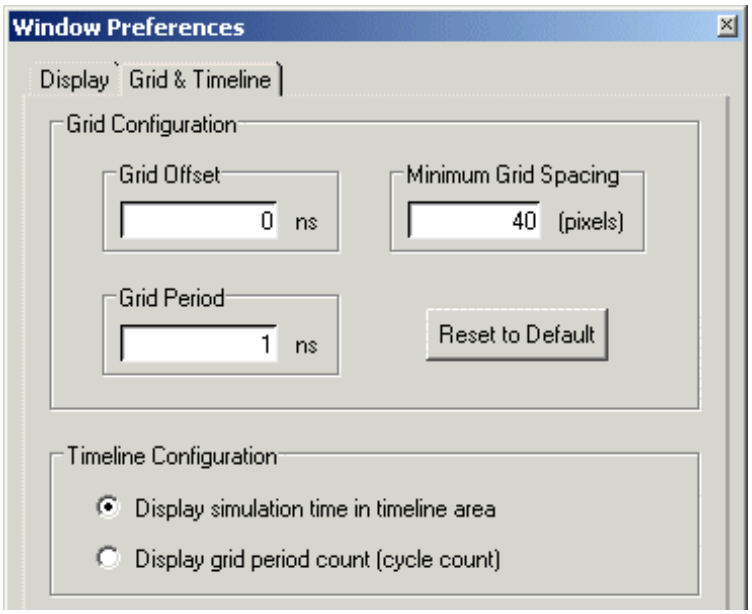
Enable/Disable

- **Waveform Popup Enable**
  Toggles on/off the popup that displays when you rest your mouse pointer on a signal or comparison object.
- **Waveform Selection Highlighting Enabled**
  Toggles on/off waveform highlighting. When enabled the waveform is highlighted if you select the waveform or its value.
- **Double-Click to Show Drivers (Dataflow Window)**
  Toggles on/off double-clicking to show the drivers of the selected waveform. See "Displaying drivers of the selected waveform" (UM-269) for more details.
- **On Close Warn for Save Format**
  Toggles on/off a message that prompts you to save the Wave window format when you close the window. See "Saving the window format" (UM-261) for more details.

Dataset Prefix Display

- **Always Show Dataset Prefixes**
  All dataset prefixes will be displayed along with the dataset prefix of the current simulation ("sim").
- **Show Dataset Prefixes if 2 or more**
  Displays all dataset prefixes if 2 or more datasets are displayed. "sim" is the default prefix for the current simulation.
- **Never Show Dataset Prefixes**
  No dataset prefixes will be displayed. This selection is useful if you are running only a single simulation.

Grid & Timeline tab



The Grid & Timeline tab includes the following options:

- **Grid Offset**
  Specifies the time (in user time units) of the first grid line. Default is 0.
- **Grid Period**
  Specifies the time (in user time units) between subsequent grid lines. Default is 1.
- **Minimum Grid Spacing**
  Specifies the closest (in pixels) two grid lines can be drawn before intermediate lines will be removed. Default is 40.
- **Timeline Configuration**
  Specifies whether to display simulation time or grid period count on the horizontal axis. Default is to display simulation time.

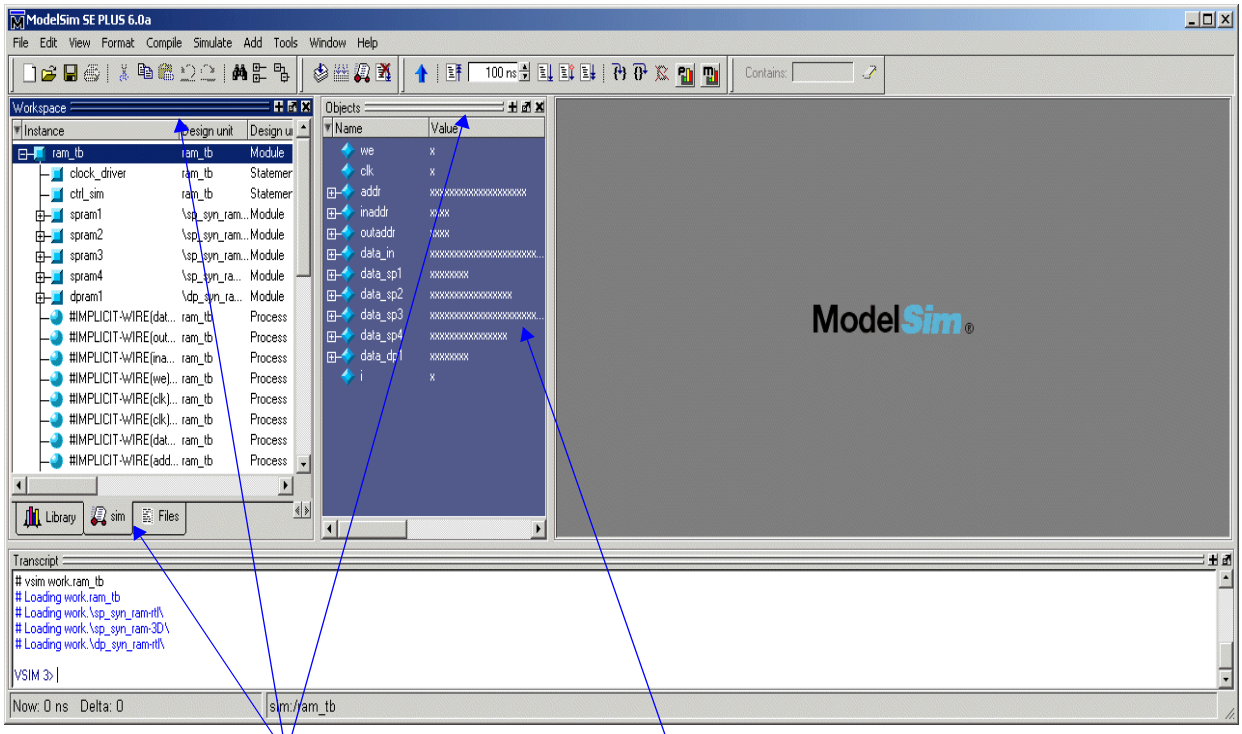
You can also access this tab by right-clicking in the cursor tracks at the bottom of the Wave window and selecting Grid & Timeline Properties.

Customizing the GUI layout

You can customize the layout of panes, windows, toolbars, etc., and ModelSim will remember your settings the next time you start the tool. This section discusses the various options for customizing the GUI layout.

Moving, docking, and undocking panes

Window panes (e.g., Transcript) can be positioned at various places within the parent window or they can be dragged out ("undocked") of the parent window altogether. When you see a double bar at the top edge of a pane, it means you can modify the pane position.



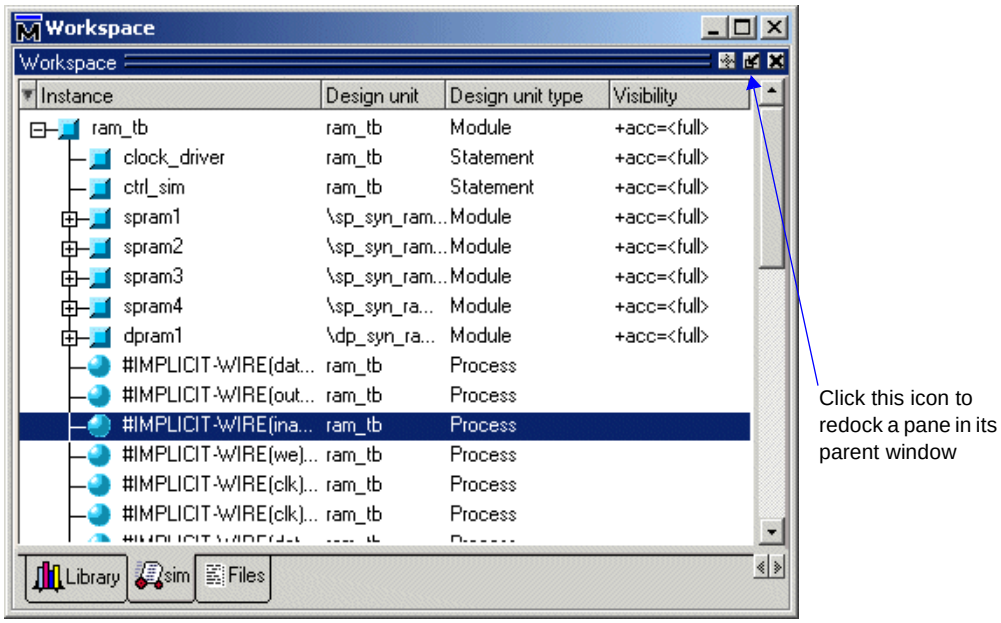
Click-and-drag on the double bar to move, undock, or dock a pane

Click this icon to undock a pane; click it again to redock

Click-and drag on the double bar to reposition the pane inside the parent window. As you move the mouse to various parts of the main window, a gray outline will show you valid locations to drop the pane.

Or, drag the pane outside of the parent window, and when you let go of the mouse button, the pane becomes a free-floating window.

To redock a floating pane, click on the double bar at the top of the window and drag it back into the parent window or click the undock/dock icon as shown in the graphic below:

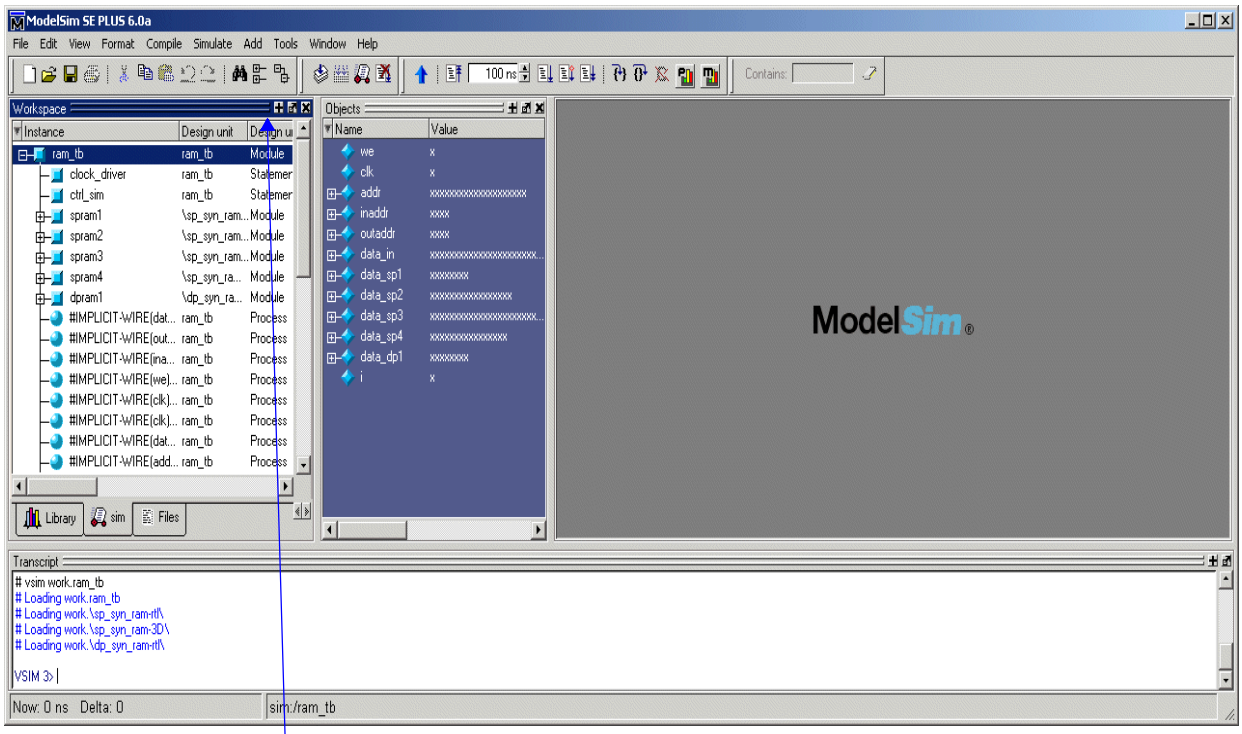


You can also undock a pane by clicking the undock/dock icon, as noted in the picture above.

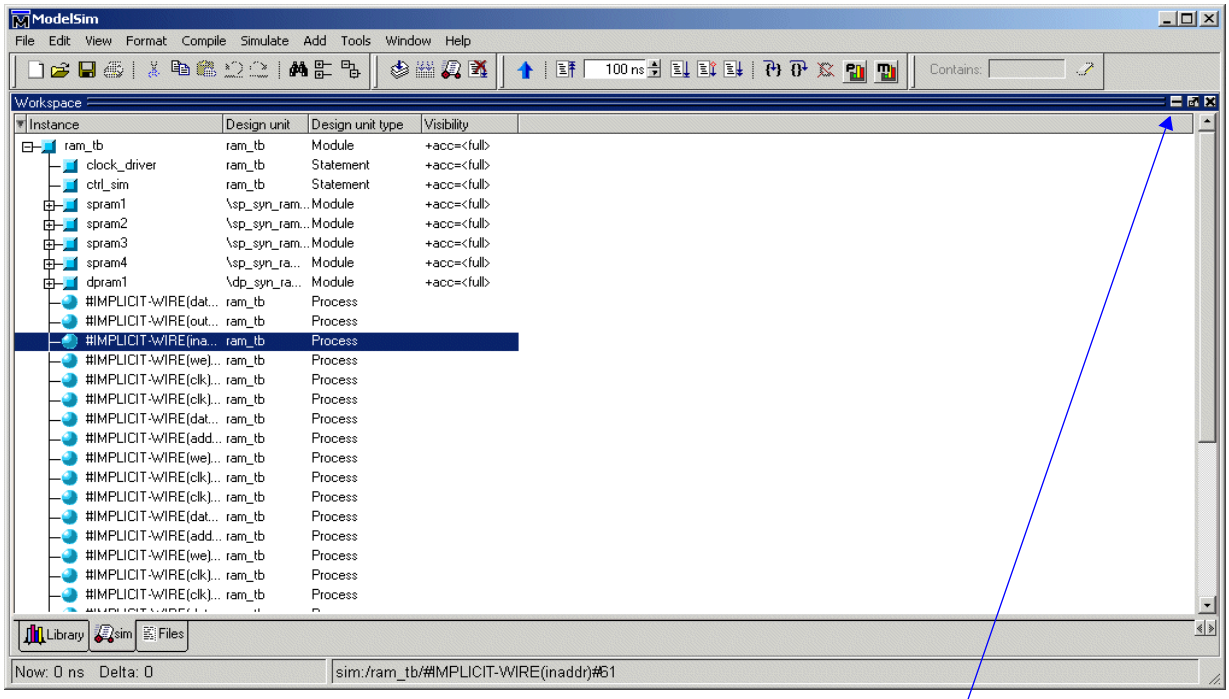
Note: If you want to return to the original default layout, select **Window** > **Initial Layout**.

Zooming panes

You can expand panes to fill the entire Main window by clicking the zoom icon. For example, in the graphic below, clicking the zoom icon on the Workspace pane makes it fill the entire Main window, as shown on the following page.



Click the zoom icon to expand a pane to fill the entire window

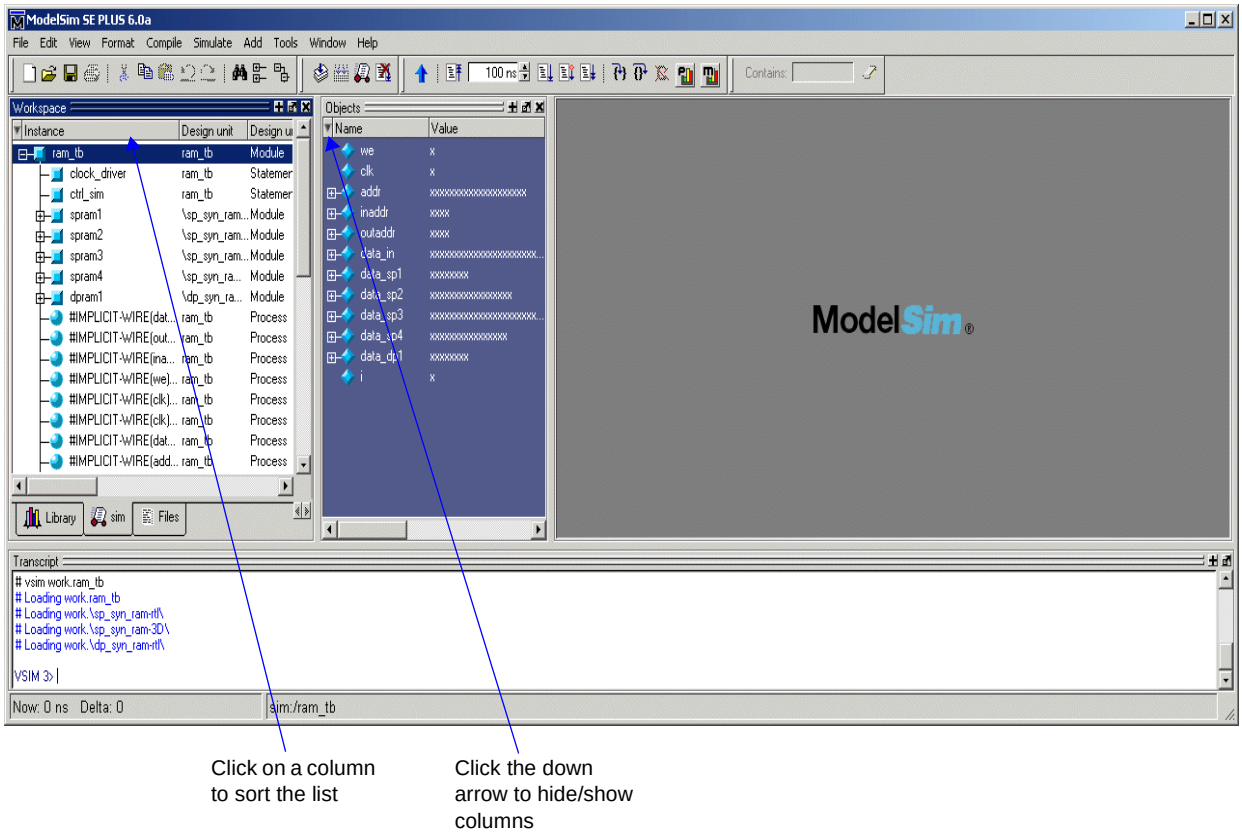


Click the unzoom icon to restore the pane to its original size and position

Columnar information display

Many panes (e.g., Objects, Workspace, etc.) display information in a columnar format. You can perform a number of operations on columnar formats:

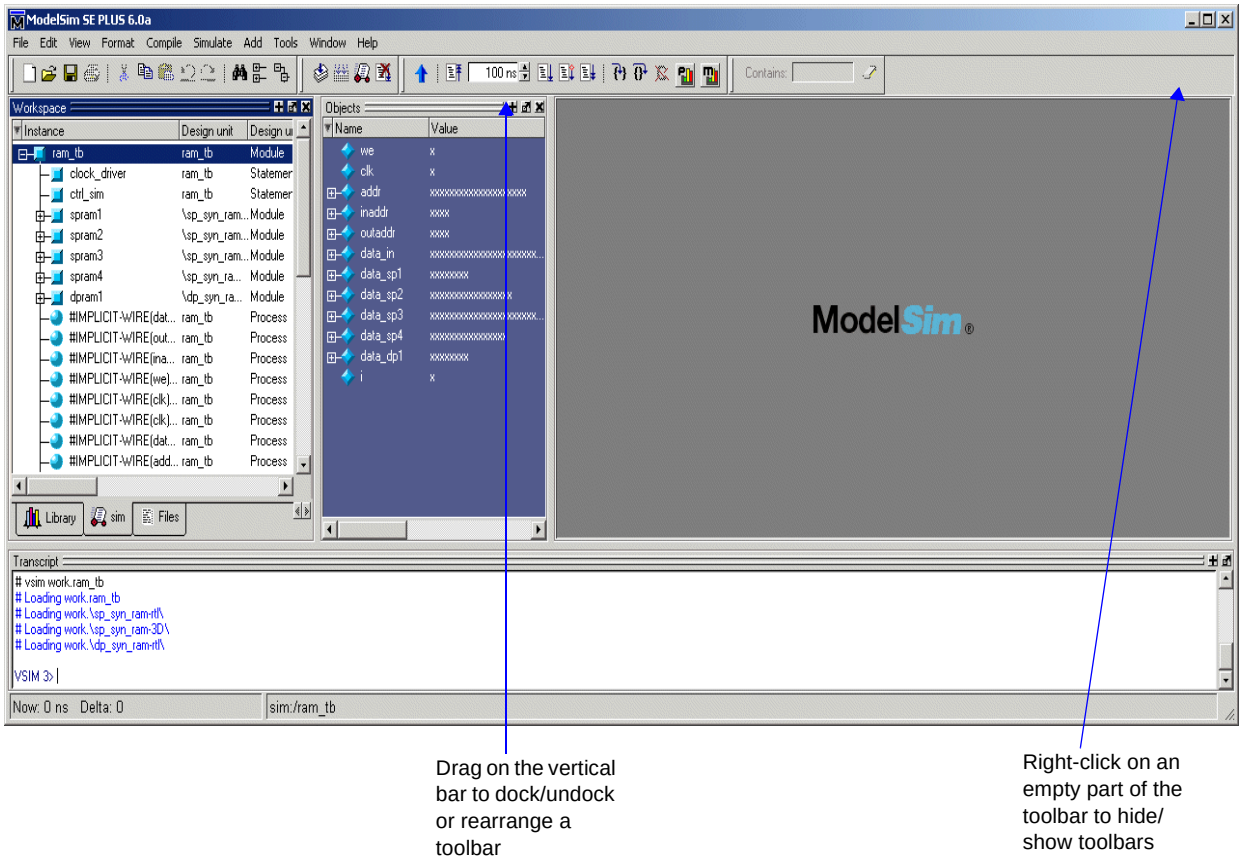
- Click and drag on a column heading to rearrange columns
- Click and drag between columns to increase/decrease column size
- Sort columns by clicking once on the column heading to sort in ascending order; clicking twice to sort in descending order; and clicking three times to sort in default order.
- Hide or show columns by either right-clicking a column heading and selecting an object from the context menu or by clicking the column-list drop down arrow and selecting an object.



Quick access toolbars

Toolbar buttons provide access to commonly used commands and functions. Toolbars can be docked and undocked (moved to or from the main toolbar area) by clicking and dragging on the vertical bar at the left-edge of a toolbar.

You can also hide/show the various toolbars. To hide or show a toolbar, right-click on a blank spot of the main toolbar area and select a toolbar from the list.



To reset toolbars to their original state, right-click on a blank spot of the main toolbar area and select **Reset**.

Creating and managing breakpoints

ModelSim supports both signal (i.e., when conditions) and file-line breakpoints. Breakpoints can be set from multiple locations in the GUI or from the command line. Breakpoints within SystemC portions of the design can only be set using File-line breakpoints (GR-264).

Signal breakpoints

Signal breakpoints (when conditions) instruct ModelSim to perform actions when the specified conditions are met. For example, you can break on a signal value or at a specific simulator time (see the **when** command (CR-407) for additional details). When a breakpoint is hit, a message in the Main window transcript identifies the signal that caused the breakpoint.

Setting signal breakpoints from the command line

You use the **when** command (CR-407) to set a signal breakpoint from the VSIM> prompt. See the *Command Reference* for further details.

Setting signal breakpoints from the GUI

Signal breakpoints are most easily set in the Objects pane (GR-184) and the Wave window (GR-211). Right-click a signal and select **Insert Breakpoint** from the context menu. A breakpoint is set on that signal and will be listed in the **Breakpoints** dialog.

Alternatively you can set signal breakpoints from the "Modify Breakpoints dialog" (GR-95).

File-line breakpoints

File-line breakpoints are set on executable lines in your source files. When the line is hit, the simulator stops.

Since C Debug is invoked when you set a breakpoint within a SystemC module, your C Debug settings must be in place prior to setting a breakpoint. See Setting up C Debug (UM-402) for more information. Once invoked, C Debug can be exited using the C Debug menu.

Setting file-line breakpoints from the command line

You use the **bp** command (CR-75) to set a file-line breakpoint from the VSIM> prompt. See the *Command Reference* for further details.

Setting file-line breakpoints from the GUI

File-line breakpoints are most easily set using your mouse in the Source window (GR-199). Click on a blue line number at the left side of the Source window, and a red diamond denoting a breakpoint will appear. The breakpoints are toggles – click once to create the colored diamond; click again to disable or enable the breakpoint. To delete the breakpoint completely, click the red diamond with your right mouse button, and select **Remove Breakpoint**.

Alternatively you can set file-line breakpoints from the "Modify Breakpoints dialog" (GR-95).

2 - Setting GUI preferences

Chapter contents

ModelSim GUI preferences GR-266
Setting variables from the GUI GR-267
Setting variables from the command line GR-267

This chapter describes how to set preferences for the ModelSim GUI.

ModelSim GUI preferences

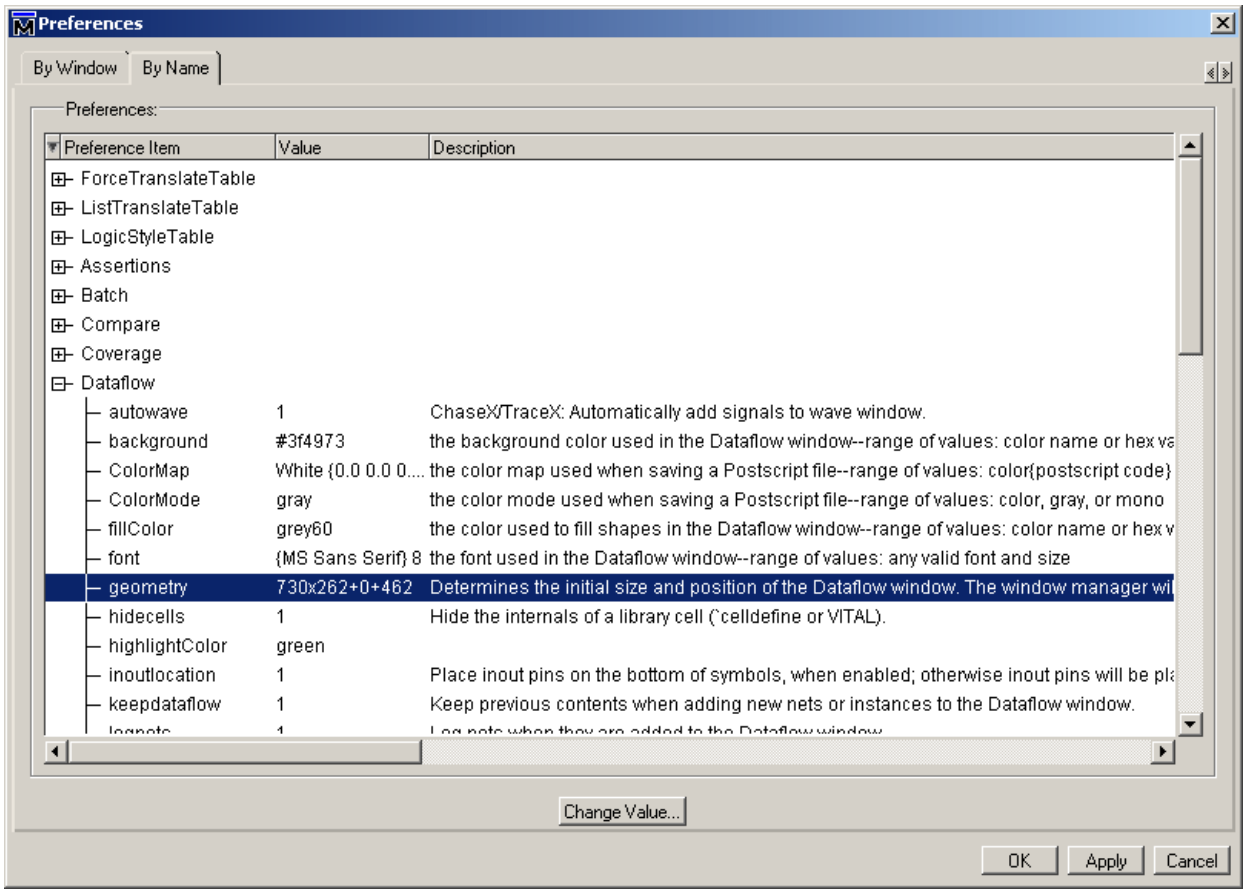
ModelSim Tcl preference variables give you control over fonts, colors, prompts, and other GUI characteristics. When you invoke ModelSim the first time, it loads default preferences from the *pref.tcl* file. You can customize the preference variables and save a file called *modelsim.tcl* file that ModelSim reads in lieu of *pref.tcl*. Once you have created a *modelsim.tcl* file, ModelSim attempts to load the file each time it starts up. ModelSim searches for the file as follows:

- use MODELSIM_TCL (UM-521) environment variable if it exists (if MODELSIM_TCL is a list of files, each file is loaded in the order that it appears in the list); else
- use *./modelsim.tcl*; else
- use $(HOME)/modelsim.tcl if it exists

Important: If your preference file is not named *modelsim.tcl*, or if the file is not located in the directories mentioned above, you must refer to it with the MODELSIM_TCL environment variable.

Setting variables from the GUI

Select **Tools** > **Edit Preferences** in the Main window to open the Preferences dialog box.



You can change settings on either the By Window tab or the By Name tab. The By Window tab allows you to change colors and fonts. The By Name tab lists every Tcl variable in ModelSim. Clicking the OK button saves all changes to a user preference file which is invoked every time ModelSim is invoked. See "Preferences dialog" (GR-104) for more details on the dialog.

Setting variables from the command line

Use the Tcl **set** command (UM-477) to customize preference variables from the Main window command line:

```
set <variable name> <variable value>
```

This command establishes variable values for the current session only. To save the current preference settings to a *modelsim.tcl* file, use the **write preferences** command:

```
write preferences modelsim.tcl
```

End-User License Agreement

IMPORTANT - USE OF THIS SOFTWARE IS SUBJECT TO LICENSE RESTRICTIONS.
CAREFULLY READ THIS LICENSE AGREEMENT BEFORE USING THE SOFTWARE.

This license is a legal "Agreement" concerning the use of Software between you, the end user, either individually or as an authorized representative of the company acquiring the license, and Mentor Graphics Corporation and Mentor Graphics (Ireland) Limited acting directly or through their subsidiaries or authorized distributors (collectively "Mentor Graphics"). USE OF SOFTWARE INDICATES YOUR COMPLETE AND UNCONDITIONAL ACCEPTANCE OF THE TERMS AND CONDITIONS SET FORTH IN THIS AGREEMENT. If you do not agree to these terms and conditions, promptly return, or, if received electronically, certify destruction of Software and all accompanying items within five days after receipt of Software and receive a full refund of any license fee paid.

END-USER LICENSE AGREEMENT

1. **GRANT OF LICENSE.** The software programs you are installing, downloading, or have acquired with this Agreement, including any updates, modifications, revisions, copies, documentation and design data ("Software") are copyrighted, trade secret and confidential information of Mentor Graphics or its licensors who maintain exclusive title to all Software and retain all rights not expressly granted by this Agreement. Mentor Graphics grants to you, subject to payment of appropriate license fees, a nontransferable, nonexclusive license to use Software solely: (a) in machine-readable, object-code form; (b) for your internal business purposes; and (c) on the computer hardware or at the site for which an applicable license fee is paid, or as authorized by Mentor Graphics. A site is restricted to a one-half mile (800 meter) radius. Mentor Graphics' standard policies and programs, which vary depending on Software, license fees paid or service plan purchased, apply to the following and are subject to change: (a) relocation of Software; (b) use of Software, which may be limited, for example, to execution of a single session by a single user on the authorized hardware or for a restricted period of time (such limitations may be communicated and technically implemented through the use of authorization codes or similar devices); (c) support services provided, including eligibility to receive telephone support, updates, modifications, and revisions. Current standard policies and programs are available upon request.

2. **ESD SOFTWARE.** If you purchased a license to use embedded software development ("ESD") Software, Mentor Graphics grants to you a nontransferable, nonexclusive license to reproduce and distribute executable files created using ESD compilers, including the ESD run-time libraries distributed with ESD C and C++ compiler Software that are linked into a composite program as an integral part of your compiled computer program, provided that you distribute these files only in conjunction with your compiled computer program. Mentor Graphics does NOT grant you any right to duplicate or incorporate copies of Mentor Graphics' real-time operating systems or other ESD Software, except those explicitly granted in this section, into your products without first signing a separate agreement with Mentor Graphics for such purpose.

3. **BETA CODE.** Portions or all of certain Software may contain code for experimental testing and evaluation ("Beta Code"), which may not be used without Mentor Graphics' explicit authorization. Upon Mentor Graphics' authorization, Mentor Graphics grants to you a temporary, nontransferable, nonexclusive license for experimental use to test and evaluate the Beta Code without charge for a limited period of time specified by Mentor Graphics. This grant and your use of the Beta Code shall not be construed as marketing or

offering to sell a license to the Beta Code, which Mentor Graphics may choose not to release commercially in any form. If Mentor Graphics authorizes you to use the Beta Code, you agree to evaluate and test the Beta Code under normal conditions as directed by Mentor Graphics. You will contact Mentor Graphics periodically during your use of the Beta Code to discuss any malfunctions or suggested improvements. Upon completion of your evaluation and testing, you will send to Mentor Graphics a written evaluation of the Beta Code, including its strengths, weaknesses and recommended improvements. You agree that any written evaluations and all inventions, product improvements, modifications or developments that Mentor Graphics conceived or made during or subsequent to this Agreement, including those based partly or wholly on your feedback, will be the exclusive property of Mentor Graphics. Mentor Graphics will have exclusive rights, title and interest in all such property. The provisions of this subsection shall survive termination or expiration of this Agreement.

4. **RESTRICTIONS ON USE.** You may copy Software only as reasonably necessary to support the authorized use. Each copy must include all notices and legends embedded in Software and affixed to its medium and container as received from Mentor Graphics. All copies shall remain the property of Mentor Graphics or its licensors. You shall maintain a record of the number and primary location of all copies of Software, including copies merged with other software, and shall make those records available to Mentor Graphics upon request. You shall not make Software available in any form to any person other than employees and contractors, excluding Mentor Graphics' competitors, whose job performance requires access. You shall take appropriate action to protect the confidentiality of Software and ensure that any person permitted access to Software does not disclose it or use it except as permitted by this Agreement. Except as otherwise permitted for purposes of interoperability as specified by applicable and mandatory local law, you shall not reverse-assemble, reverse-compile, reverse-engineer or in any way derive from Software any source code. You may not sublicense, assign or otherwise transfer Software, this Agreement or the rights under it, whether by operation of law or otherwise (“attempted transfer”), without Mentor Graphics’ prior written consent and payment of Mentor Graphics’ then-current applicable transfer charges. Any attempted transfer without Mentor Graphics' prior written consent shall be a material breach of this Agreement and may, at Mentor Graphics' option, result in the immediate termination of the Agreement and licenses granted under this Agreement.

   The terms of this Agreement, including without limitation, the licensing and assignment provisions shall be binding upon your heirs, successors in interest and assigns. The provisions of this section 4 shall survive the termination or expiration of this Agreement.

5. **LIMITED WARRANTY.**

   5.1. Mentor Graphics warrants that during the warranty period Software, when properly installed, will substantially conform to the functional specifications set forth in the applicable user manual. Mentor Graphics does not warrant that Software will meet your requirements or that operation of Software will be uninterrupted or error free. The warranty period is 90 days starting on the 15th day after delivery or upon installation, whichever first occurs. You must notify Mentor Graphics in writing of any nonconformity within the warranty period. This warranty shall not be valid if Software has been subject to misuse, unauthorized modification or installation. MENTOR GRAPHICS' ENTIRE LIABILITY AND YOUR EXCLUSIVE REMEDY SHALL BE, AT MENTOR GRAPHICS' OPTION, EITHER (A) REFUND OF THE PRICE PAID UPON RETURN OF SOFTWARE TO MENTOR GRAPHICS OR (B) MODIFICATION OR REPLACEMENT OF SOFTWARE THAT DOES NOT MEET THIS LIMITED WARRANTY, PROVIDED YOU HAVE OTHERWISE COMPLIED WITH THIS AGREEMENT. MENTOR GRAPHICS MAKES NO

WARRANTIES WITH RESPECT TO: (A) SERVICES; (B) SOFTWARE WHICH IS LICENSED TO YOU FOR A LIMITED TERM OR LICENSED AT NO COST; OR (C) EXPERIMENTAL BETA CODE; ALL OF WHICH ARE PROVIDED "AS IS."

5.2. THE WARRANTIES SET FORTH IN THIS SECTION 5 ARE EXCLUSIVE. NEITHER MENTOR GRAPHICS NOR ITS LICENSORS MAKE ANY OTHER WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, WITH RESPECT TO SOFTWARE OR OTHER MATERIAL PROVIDED UNDER THIS AGREEMENT. MENTOR GRAPHICS AND ITS LICENSORS SPECIFICALLY DISCLAIM ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF INTELLECTUAL PROPERTY.

6. **LIMITATION OF LIABILITY.** EXCEPT WHERE THIS EXCLUSION OR RESTRICTION OF LIABILITY WOULD BE VOID OR INEFFECTIVE UNDER APPLICABLE LAW, IN NO EVENT SHALL MENTOR GRAPHICS OR ITS LICENSORS BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES (INCLUDING LOST PROFITS OR SAVINGS) WHETHER BASED ON CONTRACT, TORT OR ANY OTHER LEGAL THEORY, EVEN IF MENTOR GRAPHICS OR ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL MENTOR GRAPHICS' OR ITS LICENSORS' LIABILITY UNDER THIS AGREEMENT EXCEED THE AMOUNT PAID BY YOU FOR THE SOFTWARE OR SERVICE GIVING RISE TO THE CLAIM. IN THE CASE WHERE NO AMOUNT WAS PAID, MENTOR GRAPHICS AND ITS LICENSORS SHALL HAVE NO LIABILITY FOR ANY DAMAGES WHATSOEVER.

7. **LIFE ENDANGERING ACTIVITIES.** NEITHER MENTOR GRAPHICS NOR ITS LICENSORS SHALL BE LIABLE FOR ANY DAMAGES RESULTING FROM OR IN CONNECTION WITH THE USE OF SOFTWARE IN ANY APPLICATION WHERE THE FAILURE OR INACCURACY OF THE SOFTWARE MIGHT RESULT IN DEATH OR PERSONAL INJURY.

8. **INDEMNIFICATION.** YOU AGREE TO INDEMNIFY AND HOLD HARMLESS MENTOR GRAPHICS AND ITS LICENSORS FROM ANY CLAIMS, LOSS, COST, DAMAGE, EXPENSE, OR LIABILITY, INCLUDING ATTORNEYS' FEES, ARISING OUT OF OR IN CONNECTION WITH YOUR USE OF SOFTWARE AS DESCRIBED IN SECTION 7.

9. INFRINGEMENT.

9.1. Mentor Graphics will defend or settle, at its option and expense, any action brought against you alleging that Software infringes a patent or copyright or misappropriates a trade secret in the United States, Canada, Japan, or member state of the European Patent Office. Mentor Graphics will pay any costs and damages finally awarded against you that are attributable to the infringement action. You understand and agree that as conditions to Mentor Graphics' obligations under this section you must: (a) notify Mentor Graphics promptly in writing of the action; (b) provide Mentor Graphics all reasonable information and assistance to defend or settle the action; and (c) grant Mentor Graphics sole authority and control of the defense or settlement of the action.

9.2. If an infringement claim is made, Mentor Graphics may, at its option and expense: (a) replace or modify Software so that it becomes noninfringing; (b) procure for you

the right to continue using Software; or (c) require the return of Software and refund to you any license fee paid, less a reasonable allowance for use.

9.3. Mentor Graphics has no liability to you if infringement is based upon: (a) the combination of Software with any product not furnished by Mentor Graphics; (b) the modification of Software other than by Mentor Graphics; (c) the use of other than a current unaltered release of Software; (d) the use of Software as part of an infringing process; (e) a product that you make, use or sell; (f) any Beta Code contained in Software; (g) any Software provided by Mentor Graphics' licensors who do not provide such indemnification to Mentor Graphics' customers; or (h) infringement by you that is deemed willful. In the case of (h) you shall reimburse Mentor Graphics for its attorney fees and other costs related to the action upon a final judgment.

9.4. THIS SECTION 9 STATES THE ENTIRE LIABILITY OF MENTOR GRAPHICS AND ITS LICENSORS AND YOUR SOLE AND EXCLUSIVE REMEDY WITH RESPECT TO ANY ALLEGED PATENT OR COPYRIGHT INFRINGEMENT OR TRADE SECRET MISAPPROPRIATION BY ANY SOFTWARE LICENSED UNDER THIS AGREEMENT.

10. **TERM.** This Agreement remains effective until expiration or termination. This Agreement will immediately terminate upon notice if you exceed the scope of license granted or otherwise fail to comply with the provisions of Sections 1, 2, or 4. For any other material breach under this Agreement, Mentor Graphics may terminate this Agreement upon 30 days written notice if you are in material breach and fail to cure such breach within the 30-day notice period. If Software was provided for limited term use, this Agreement will automatically expire at the end of the authorized term. Upon any termination or expiration, you agree to cease all use of Software and return it to Mentor Graphics or certify deletion and destruction of Software, including all copies, to Mentor Graphics' reasonable satisfaction.

11. **EXPORT.** Software is subject to regulation by local laws and United States government agencies, which prohibit export or diversion of certain products, information about the products, and direct products of the products to certain countries and certain persons. You agree that you will not export any Software or direct product of Software in any manner without first obtaining all necessary approval from appropriate local and United States government agencies.

12. **RESTRICTED RIGHTS NOTICE.** Software was developed entirely at private expense and is commercial computer software provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the U.S. Government or a U.S. Government subcontractor is subject to the restrictions set forth in the license agreement under which Software was obtained pursuant to DFARS 227.7202-3(a) or as set forth in subparagraphs (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clause at FAR 52.227-19, as applicable. Contractor/manufacturer is Mentor Graphics Corporation, 8005 SW Boeckman Road, Wilsonville, Oregon 97070-7777 USA.

13. **THIRD PARTY BENEFICIARY.** For any Software under this Agreement licensed by Mentor Graphics from Microsoft or other licensors, Microsoft or the applicable licensor is a third party beneficiary of this Agreement with the right to enforce the obligations set forth herein.

14. **AUDIT RIGHTS.** With reasonable prior notice, Mentor Graphics shall have the right to audit during your normal business hours all records and accounts as may contain information regarding your compliance with the terms of this Agreement. Mentor Graphics shall keep in confidence all information gained as a result of any audit. Mentor

Graphics shall only use or disclose such information as necessary to enforce its rights under this Agreement.

15. **CONTROLLING LAW AND JURISDICTION.** THIS AGREEMENT SHALL BE GOVERNED BY AND CONSTRUED UNDER THE LAWS OF THE STATE OF OREGON, USA, IF YOU ARE LOCATED IN NORTH OR SOUTH AMERICA, AND THE LAWS OF IRELAND IF YOU ARE LOCATED OUTSIDE OF NORTH AND SOUTH AMERICA. All disputes arising out of or in relation to this Agreement shall be submitted to the exclusive jurisdiction of Dublin, Ireland when the laws of Ireland apply, or Wilsonville, Oregon when the laws of Oregon apply. This section shall not restrict Mentor Graphics' right to bring an action against you in the jurisdiction where your place of business is located. The United Nations Convention on Contracts for the International Sale of Goods does not apply to this Agreement.

16. **SEVERABILITY.** If any provision of this Agreement is held by a court of competent jurisdiction to be void, invalid, unenforceable or illegal, such provision shall be severed from this Agreement and the remaining provisions will remain in full force and effect.

17. **PAYMENT TERMS AND MISCELLANEOUS.** You will pay amounts invoiced, in the currency specified on the applicable invoice, within 30 days from the date of such invoice. This Agreement contains the parties' entire understanding relating to its subject matter and supersedes all prior or contemporaneous agreements, including but not limited to any purchase order terms and conditions, except valid license agreements related to the subject matter of this Agreement (which are physically signed by you and an authorized agent of Mentor Graphics) either referenced in the purchase order or otherwise governing this subject matter. This Agreement may only be modified in writing by authorized representatives of the parties. Waiver of terms or excuse of breach must be in writing and shall not constitute subsequent consent, waiver or excuse. The prevailing party in any legal action regarding the subject matter of this Agreement shall be entitled to recover, in addition to other relief, reasonable attorneys' fees and expenses.

Rev. 040401, Part Number 221417

Index

CR = Command Reference, UM = User's Manual, GR = GUI Reference

Symbols

#, comment character UM-476
+acc option, design object visibility UM-126
+typdelays CR-365
-, in a coverage report UM-353
.so, shared object file
    loading PLI/VPI C applications UM-568
    loading PLI/VPI C++ applications UM-574
{} CR-15
'hasX, hasX CR-24

Numerics

1076, IEEE Std UM-30
    differences between versions UM-75
1364, IEEE Std UM-30, UM-113
2001, keywords, disabling CR-366
64-bit libraries UM-66
64-bit ModelSim, using with 32-bit FLI apps UM-598
64-bit time
    now variable UM-543
    Tcl time commands UM-481

A

+acc option, design object visibility UM-126
abort command CR-44
absolute time, using @ CR-18
ACC routines UM-591
accelerated packages UM-65
access
    hierarchical objects UM-417
    limitations in mixed designs UM-190
Active Processes pane GR-108
    see also windows, Active Processes pane
add button command CR-45
Add file to Project dialog GR-44
Add Folder dialog GR-47
add list command CR-48
add monitor command CR-51
add PSL files UM-53, GR-53, GR-56
add wave command CR-52
add_menu command CR-56
add_menucb command CR-58
add_menuitem simulator command CR-59
add_separator command CR-60
add_submenu command CR-61
aggregates, SystemC UM-180
alias command CR-62
analog
    signal formatting CR-53, GR-239
    supported signal types GR-239
annotating interconnect delays, v2k_int_delays CR-389
architecture simulator state variable UM-542
archives
    described UM-59
archives, library CR-356
argc simulator state variable UM-542
arguments
    passing to a DO file UM-487
arguments, accessing commandl-line UM-183
arithmetic package warnings, disabling UM-538
array of sc_signal<T> UM-180
arrays
    indexes CR-12
    slices CR-12, CR-15
AssertFile .ini file variable UM-529
assertion fail command CR-63
assertion pass command CR-65
assertion report command CR-67
AssertionFailEnable .ini variable UM-529
AssertionFailLimit .ini variable UM-529
AssertionFailLog .ini variable UM-529
AssertionFormat .ini file variable UM-529
AssertionFormatBreak .ini file variable UM-529
AssertionFormatError .ini file variable UM-529
AssertionFormatFail .ini file variable UM-530
AssertionFormatFatal .ini file variable UM-530
AssertionFormatNote .ini file variable UM-530
AssertionFormatWarning .ini file variable UM-530
AssertionPassEnable .ini variable UM-530
AssertionPassLimit .ini variable UM-530
AssertionPassLog .ini variable UM-530
assertions
    configuring from the GUI GR-86
    enabling CR-63, CR-65
    failure behavior CR-63
    file and line number UM-529
    flow UM-362
    library and use clauses UM-367
    limitations UM-362
    messages
        alternate output file UM-381
        turning off UM-538

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

multiclocked properties UM-369
pass behavior CR-65
reporting on CR-67, UM-381
selecting severity that stops simulation GR-86
setting format of messages UM-529
testing for with onbreak command CR-214
viewing in Wave window UM-382
warnings, locating UM-529
Assertions pane
described GR-110
hiding/showing columns GR-112
assume directives
disabling UM-363
SimulateAssumeDirectives .ini variable UM-533
AtLeast counts, functional coverage UM-387
attributes, of signals, using in expressions CR-24
auto find bp command UM-406
auto step mode, C Debug UM-407

B

bad magic number error message UM-227
balloon dialog, toggling on/off GR-256
balloon popup
C Debug GR-99
base (radix)
List window UM-259
Memory window GR-183
Wave window UM-255
batch_mode command CR-69
batch-mode simulations UM-28
halting CR-410
bd (breakpoint delete) command CR-70
binary radix, mapping to std_logic values CR-29
BindAtCompile .ini file variable UM-527
binding, VHDL, default UM-79
bitwise format UM-279
blocking assignments UM-134
bookmark add wave command CR-71
bookmark delete wave command CR-72
bookmark goto wave command CR-73
bookmark list wave command CR-74
bookmarks
Source window GR-204
Wave window UM-250
bp (breakpoint) command CR-75
brackets, escaping CR-15
break
on assertion GR-86
on signal value CR-407
stop simulation run GR-35
BreakOnAssertion .ini file variable UM-530
breakpoints
C code UM-403
conditional CR-407
continuing simulation after CR-252
deleting CR-70, GR-203, GR-264
listing CR-75
setting CR-75, GR-203
setting automatically in C code UM-407
signal breakpoints (when statements) CR-407
Source window, viewing in GR-199
time-based
in when statements CR-411
.bsm file UM-313
buffered/unbuffered output UM-534
bus contention checking CR-84
configuring CR-86
disabling CR-87
bus float checking
configuring CR-89
disabling CR-90
enabling CR-88
busses
escape characters in CR-15
RTL-level, reconstructing UM-234
user-defined CR-53, UM-264
buswise format UM-279
button
adding to windows GR-106
buttons, adding to the Main window toolbar CR-45

C

C applications
compiling and linking UM-568
debugging UM-399
C callstack
moving down CR-237
moving up CR-219
C Debug UM-399
auto find bp UM-406
auto step mode UM-407
debugging functions during elaboration UM-410
debugging functions when exiting UM-414
function entry points, finding UM-406
initialization mode UM-410
menu reference GR-31
registered function calls, identifying UM-407
running from a DO file UM-402

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Stop on quit mode UM-414
C Debug setup dialog GR-99
C debugging CR-79
C++ applications
compiling and linking UM-574
cancelling scheduled events, performance UM-108
case choice, must be locally static CR-314
case sensitivity
named port associations UM-207
VHDL vs. Verilog CR-15
causality, tracing in Dataflow window UM-306
cd (change directory) command CR-78
cdbg command CR-79
cdbg_wait_for_starting command UM-402
cell libraries UM-144
cells
hiding in Dataflow window GR-140, GR-141
change command CR-81
change directory, disabled GR-21
Change Memory dialog GR-179
Change Selected Variable dialog GR-167
change_menu_cmd command CR-83
chasing X UM-307
check contention add command CR-84
check contention config command CR-86
check contention off command CR-87
check float add command CR-88
check float config command CR-89
check float off command CR-90
check stable off command CR-91
check stable on command CR-92
-check_synthesis argument CR-312
warning message UM-552
CheckPlusargs .ini file variable (VLOG) UM-530
checkpoint command CR-93
checkpoint/restore UM-86, UM-142
CheckpointCompressMode .ini file variable UM-531
CheckSynthesis .ini file variable UM-527
class member selection, syntax CR-13
class of sc_signal<T> UM-180
cleanup
SystemC state-based code UM-175
clean-up of SystemC state-based code UM-175
clock change, sampling signals at UM-268
clocked comparison UM-276
Code Coverage
$coverage_save system function UM-152
by instance UM-334
columns in workspace GR-116
condition coverage UM-334, UM-355
coverage clear command CR-128
coverage exclude command CR-129
coverage reload command CR-131
coverage report command CR-132
coverage save command CR-135
Current Exclusions pane GR-121
data types supported UM-335
Details pane GR-123
display filter toolbar GR-127
enabling with vcom or vlog UM-337
enabling with vsim UM-337
excluding lines/files UM-347
exclusion filter files UM-348
expression coverage UM-334, UM-356
important notes UM-336
Instance Coverage pane GR-122
Main window coverage data UM-340
merge utility UM-354
merging report files CR-131
merging reports CR-320
missed branches GR-120
missed coverage GR-120
pragma exclusions UM-347
reports UM-350
Source window data UM-341
source window details GR-124
statistics in Main window UM-340
toggle coverage UM-334
excluding signals CR-281
toggle details GR-123
vcover report command CR-322
Workspace pane GR-116
collapsing ports, and coverage reporting UM-345
collapsing time and delta steps UM-232
colorization, in Source window GR-205
columns
hide/showing in GUI GR-262
moving GR-262
sorting by GR-262
Combine Selected Signals dialog GR-161
combining signals, busses CR-53, UM-264
command history GR-28
command line args, accessing
vsim sc_arg command CR-389
CommandHistory .ini file variable UM-531
command-line arguments, accessing UM-183
command-line mode UM-27
commands
.main clear CR-43
abort CR-44
add button CR-45
add list CR-48

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

add monitor CR-51
add wave CR-52
add_menu CR-56
add_menucb CR-58
add_menuitem CR-59
add_separator CR-60
add_submenu CR-61
alias CR-62
assertion fail command CR-63
assertion pass CR-65
assertion report CR-67
batch_mode CR-69
bd (breakpoint delete) CR-70
bookmark add wave CR-71
bookmark delete wave CR-72
bookmark goto wave CR-73
bookmark list wave CR-74
bp (breakpoint) CR-75
cd (change directory) CR-78
cdbg CR-79
change CR-81
change_menu_cmd CR-83
check contention add CR-84
check contention config CR-86
check contention off CR-87
check float add CR-88
check float config CR-89
check float off CR-90
check stable off CR-91
check stable on CR-92
checkpoint CR-93
compare add CR-94
compare annotate CR-98, CR-101
compare clock CR-99
compare close CR-105
compare delete CR-104
compare info CR-106
compare list CR-107
compare open CR-119
compare options CR-108
compare reload CR-112
compare savediffs CR-115
compare saverules CR-116
compare see CR-117
compare start CR-114
configure CR-123
coverage clear CR-128
coverage exclude CR-129
coverage reload CR-131
coverage report CR-132
coverage save CR-135
dataset alias CR-136
dataset clear CR-137
dataset close CR-138
dataset info CR-139
dataset list CR-140
dataset open CR-141
dataset rename CR-142, CR-143
dataset snapshot CR-144
delete CR-146
describe CR-147
disable_menu CR-149
disable_menuitem CR-150
disablebp CR-148
do CR-151
down CR-152
drivers CR-154
dumplog64 CR-155
echo CR-156
edit CR-157
enable_menu CR-159
enable_menuitem CR-160
enablebp CR-158
environment CR-161
event watching in DO file UM-487
examine CR-162
exit CR-166
fcover clear
    functional coverage
        clearing database CR-167
fcover comment CR-168
fcover configue CR-169
fcover reload CR-171
fcover report CR-173
fcover save CR-175
find CR-176
force CR-180
gdb dir CR-183
getactivecursortime CR-184
getactivemarkertime CR-185
help CR-186
history CR-187
lecho CR-188
left CR-189
log CR-191
lshift CR-193
lsublist CR-194
macro_option CR-195
mem display CR-196
mem list CR-198
mem load CR-199
mem save CR-202

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

mem search CR-204
modelsim CR-206
next CR-207
noforce CR-208
nolog CR-209
notation conventions CR-10
notepad CR-211
noview CR-212
nowhen CR-213
onbreak CR-214
onElabError CR-215
onerror CR-216
pause CR-217
play CR-218
pop CR-219
power add CR-220
power report CR-221
power reset CR-222
printenv CR-223, CR-224
profile clear CR-225
profile interval CR-226
profile off CR-227
profile on CR-228
profile option CR-229
profile reload CR-230
profile report CR-231
property list CR-234
property wave CR-235
push CR-237
pwd CR-238
quietly CR-239
quit CR-240
radix CR-241
readers CR-242
record CR-243
report CR-244
restart CR-246
restore CR-248
resume CR-249
right CR-250
run CR-252
sccom CR-254
scgenmod CR-258
search CR-260
searchlog CR-262
seetime CR-264
setenv CR-265
shift CR-266
show CR-267
splitio CR-270
status CR-271
step CR-272
stop CR-273
system UM-479
tb (traceback) CR-274
tcheck_set CR-275
tcheck_status CR-277
toggle add CR-279
toggle disable CR-281
toggle enable CR-282
toggle report CR-283
toggle reset CR-284
transcribe CR-285
transcript CR-286
transcript file CR-287
TreeUpdate CR-423
tssi2mti CR-288
unsetenv CR-289
up CR-290
variables referenced in CR-17
vcd add CR-292
vcd checkpoint CR-293
vcd comment CR-294
vcd dumpports CR-295
vcd dumpportsall CR-297
vcd dumpportsflush CR-298
vcd dumpportslimit CR-299
vcd dumpportsoff CR-300
vcd dumpportson CR-301
vcd file CR-302
vcd files CR-304
vcd flush CR-306
vcd limit CR-307
vcd off CR-308
vcd on CR-309
vcom CR-311
vcover convert CR-319
vcover merge CR-320
vcover report CR-322
vdel CR-327
vdir CR-328
verror CR-329
vgencomp CR-330
view CR-332
virtual count CR-334
virtual define CR-335
virtual delete CR-336
virtual describe CR-337
virtual expand CR-338
virtual function CR-339
virtual hide CR-342
virtual log CR-343

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

virtual nohide CR-345
virtual nolog CR-346
virtual region CR-348
virtual save CR-349
virtual show CR-350
virtual signal CR-351
virtual type CR-354
vlib CR-356
vlog CR-358
vmake CR-369
vmap CR-370
vopt CR-371
vsim CR-373
VSIM Tcl commands UM-480
vsimDate CR-392
vsimId CR-392
vsimVersion CR-392
wave CR-394
wave create CR-397
wave edit CR-400
wave export CR-403
wave import CR-404
wave modify CR-405
WaveActivateNextPane CR-423
WaveRestoreCursors CR-423
WaveRestoreZoom CR-423
when CR-407
where CR-412
wlf2log CR-413
wlf2vcd CR-415
wlfman CR-416
wlfrecover CR-420
write cell_report CR-421
write format CR-422
write list CR-424
write preferences CR-425
write report CR-426
write timing CR-427
write transcript CR-428
write tssi CR-429
write wave CR-431
comment character
Tcl and DO files UM-476
comment characters in VSIM commands CR-10
compare
add region UM-275
add signals UM-274
by signal UM-274
clocked UM-276
difference markers UM-279
displayed in List window UM-281
icons UM-281
method UM-276
options UM-278
pathnames UM-279
reference dataset UM-272
reference region UM-275
tab UM-273
test dataset UM-273
timing differences UM-279
tolerance UM-276
values UM-280
wave window display UM-279
compare add command CR-94
compare annotate command CR-98, CR-101
compare by region UM-275
compare clock command CR-99
compare close command CR-105
compare delete command CR-104
compare info command CR-106
compare list command CR-107
Compare Memory dialog GR-181
compare open command CR-119
compare options command CR-108
compare reload command CR-112
compare savediffs command CR-115
compare saverules command CR-116
compare see command CR-117
compare simulations UM-225
compare start command CR-114
compatibility, of vendor libraries CR-328
compile
gensrc errors during UM-185
projects
add PSL files UM-53, GR-53, GR-56
compile order
auto generate UM-46
changing UM-46
Compile Order dialog GR-69
Compile Source Files dialog
dialogs
Compile Source Files GR-59
compiler directives UM-153
IEEE Std 1364-2000 UM-153
XL compatible compiler directives UM-154
Compiler Options dialog GR-60
compiling
changing order in the GUI UM-46
graphic interface to GR-59
grouping files UM-47
order, changing in projects UM-46
properties, in projects UM-52

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

range checking in VHDL CR-316, UM-74
source errors, locating GR-264
SystemC CR-254, CR-258, UM-164
converting sc_main() UM-164
exporting top level module UM-165
for source level debug UM-167
invoking sccom UM-167
linking the compiled source UM-172
modifying source code UM-164
replacing sc_start() UM-164
using sccom vs. raw C++ compiler UM-170
Verilog CR-358, UM-114
incremental compilation UM-115
library components, including CR-361
optimizing performance CR-360
XL 'uselib compiler directive UM-120
XL compatible options UM-119
VHDL CR-311, UM-73
at a specified line number CR-313
selected design units (-just eapbc) CR-313
standard package (-s) CR-316
VITAL packages UM-95
compiling the design
overview UM-25
component declaration
generating SystemC from Verilog or VHDL UM-223
generating VHDL from Verilog UM-204
vgencomp for SystemC UM-223
vgencomp for VHDL UM-204
component, default binding rules UM-79
Compressing files
VCD tasks UM-462
compressing files
VCD files CR-295, CR-304
concatenation
directives CR-28
of signals CR-27, CR-351
ConcurrentFileLimit .ini file variable UM-531
conditional breakpoints CR-407
configuration simulator state variable UM-542
configurations
instantiation in mixed designs UM-203
Verilog UM-122
configurations, simulating CR-373
configure command CR-123
Configure cover directives dialog GR-149
connectivity, exploring UM-303
constants
in case statements CR-314
values of, displaying CR-147, CR-162
contention checking CR-84
context menu
List window GR-155
context menus
Library tab UM-61
context sensitivity UM-501
control function, SystemC UM-192
control_foreign_signal() function UM-183
conversion, radix CR-241
convert real to time UM-99
convert time to real UM-98
coverage
merging data UM-354
saving raw data UM-354
see also Code Coverage
see also functional coverage
coverage clear command CR-128
coverage exclude command CR-129
coverage reload command CR-131
coverage report command CR-132
Coverage Report dialog GR-90
coverage reports UM-350
reporting all signals UM-345
sample reports UM-352
xml format UM-351
coverage save command CR-135
$coverage_save system function UM-152
CoverAtLeast .ini file variable UM-531
CoverEnable .ini file variable UM-531
CoverLimit .ini file variable UM-531
CoverLog .ini file variable UM-531
CoverWeight .ini file variable UM-531
covreport.xsl UM-351
CppOptions .ini file variable (sccom) UM-528
CppPath .ini file variable (sccom) UM-528
Create a New Library dialog GR-38
Create Project dialog GR-37
Create Project File dialog GR-43
current exclusions
pragmas UM-347
Current Exclusions pane GR-121
cursors
adding, deleting, locking, naming UM-245
link to Dataflow window UM-302
measuring time with UM-245
trace events with UM-306
Wave window UM-245
Customize Toolbar dialog GR-106
customizing
adding buttons CR-45
via preference variables GR-266

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

D

deltas
    explained UM-80
data types
    Code Coverage UM-335
database, functional coverage, saving UM-395
Dataflow Options dialog GR-140
Dataflow Page Setup dialog GR-138
Dataflow window UM-300, GR-128
    automatic cell hiding GR-140, GR-141
    menu bar GR-129
    options GR-140, GR-141
    pan UM-305
    zoom UM-305
    see also windows, Dataflow window
dataflow.bsm file UM-313
dataset alias command CR-136
Dataset Browser UM-229, GR-49
    dialog GR-49
dataset clear command CR-137
dataset close command CR-138
dataset info command CR-139
dataset list command CR-140
dataset open command CR-141
dataset rename command CR-142, CR-143
Dataset Snapshot UM-231
dataset snapshot command CR-144
datasets UM-225
    environment command, specifying with CR-161
    managing UM-229
    openingdialogs
        Open File GR-39
    reference UM-272
    restrict dataset prefix display UM-230
    test UM-273
DatasetSeparator .ini file variable UM-531
debuggable SystemC objects UM-176
debugging
    C code UM-399
debugging the design, overview UM-26
declarations, hiding implicit with explicit CR-318
default binding
    BindAtCompile .ini file variable UM-527
    disabling UM-79
default binding rules UM-79
default clock UM-368
Default editor, changing UM-521
DefaultForceKind .ini file variable UM-531
DefaultRadix .ini file variable UM-531
DefaultRestartOptions variable UM-531, UM-539
defaults
    restoring UM-520
+define+ CR-359
Define Clock dialog GR-188
definition (ID) of memory GR-170
delay
    delta delays UM-80
    interconnect CR-378
    modes for Verilog models UM-144
    SDF files UM-439
    stimulus delay, specifying GR-187
+delay_mode_distributed CR-359
+delay_mode_path CR-359
+delay_mode_unit CR-359
+delay_mode_zero CR-360
'delayed CR-24
DelayFileOpen .ini file variable UM-532
delaying test signal, Waveform Comparison GR-244
delete command CR-146
deleting library contents UM-61
delta collapsing UM-232
delta simulator state variable UM-542
deltas
    collapsing in the List window GR-163
    collapsing in WLF files CR-382
    hiding in the List window CR-124, GR-163
    in List window UM-265
    referencing simulator iteration
        as a simulator state variable UM-542
dependencies, checking CR-328
dependent design units UM-73
describe command CR-147
descriptions of HDL items GR-203
design library
    creating UM-60
    logical name, assigning UM-62
    mapping search rules UM-63
    resource type UM-58
    VHDL design units UM-73
    working type UM-58
design object icons, described GR-12
Design Optimization dialog GR-70
design portability and SystemC UM-168
design units UM-58
    report of units simulated CR-426
    Verilog
        adding to a library CR-358
details
    code coverage GR-123
dialogs GR-49
    Add file to Project GR-44

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Add Folder GR-47
C Debug setup GR-99
Change Memory GR-179
Change Selected Variable GR-167
Combine Selected Signals GR-161
Compare Memory GR-181
Compile Order GR-69
Compiler Options GR-60
Configure cover directives GR-149
Coverage Report GR-90
Create a New Library GR-38
Create Project GR-37
Create Project File GR-43
Customize Toolbar GR-106
Dataflow Options GR-140
Dataflow Page Setup GR-138
Define Clock GR-188
Design Optimization GR-70
File Breakpoint GR-98
Filter instance list GR-92
Find in Assertions GR-113
Find in dataflow GR-139
Find in FCovers GR-148
Find in List GR-156
Find in Locals GR-168
Find in memory GR-182
Find in Process GR-109
Force Selected Signal GR-186
Functional coverage filter GR-151
Functional coverage reload GR-145
Functional coverage report GR-146
List Signal Properties GR-159
List Signal Search GR-157
Load Coverage Data GR-89
Macro GR-102
Modify Breakpoints GR-95
Modify Display Properties GR-162
Optimization Configuration GR-45
Preferences GR-104
Print GR-135
Print Postscript GR-137
Profile Report GR-93, GR-197
Project Compiler Settings GR-50
Project Settings GR-57
Properties (memory) GR-183
Restart GR-88
Runtime Options GR-85
Save Memory GR-177
Signal Breakpoints GR-97
Simulation Configuration GR-46
Start Simulation GR-76
SystemC Link dialog GR-68
directories
mapping libraries CR-370
moving libraries UM-63
directory, changing, disabled GR-21
disable_menu command CR-149
disable_menuitem command CR-150
disablebp command CR-148
distributed delay mode UM-145
dividers
adding from command line CR-52
Wave window UM-256
DLL files, loading UM-568, UM-574
do command CR-151
DO files (macros) CR-151
error handling UM-490
executing at startup UM-521, UM-534
parameters, passing to UM-487
Tcl source command UM-491
docking
window panes GR-258
documentation UM-35
DOPATH environment variable UM-521
down command CR-152
DPI
export TFs UM-551
DPI export TFs UM-551
DPI use flow UM-566
drag & drop preferences GR-103
drivers
Dataflow Window UM-303
show in Dataflow window UM-269
Wave window UM-269
drivers command CR-154
drivers, multiple on unresolved signal GR-53, GR-62
dump files, viewing in ModelSim CR-310
dumplog64 command CR-155
dumpports tasks, VCD files UM-461

E

echo command CR-156
edges, finding CR-189, CR-250
edit command CR-157
Editing
in notepad windows UM-605
in the Main window UM-605
in the Source window UM-605
EDITOR environment variable UM-521
editor, default, changing UM-521

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

elab_defer_fli argument UM-84, UM-140
elaboration file
  creating UM-83, UM-139
  loading UM-83, UM-139
  modifying stimulus UM-83, UM-139
  resimulating the same design UM-82, UM-138
  simulating with PLI or FLI models UM-84, UM-140
elaboration, interrupting CR-373
embedded wave viewer UM-304
empty port name warning UM-551
enable_menu command CR-159
enable_menuitem command CR-160
enablebp command CR-158
encryption
  +protect argument CR-365
  ‘protect compiler directive UM-155
  -nodebug argument (vcom) CR-314
  -nodebug argument (vlog) CR-363
  securing pre-compiled libraries UM-70
end_of_construction() function UM-183
end_of_simulation() function UM-183
ENDFILE function UM-91
ENDLINE function UM-91
endpoint directives
  clocking and UM-371
  restrictions on UM-371
endpoints, PSL directive UM-398
‘endprotect compiler directive UM-155
entities
  default binding rules UM-79
entities, specifying for simulation CR-390
entity simulator state variable UM-542
enumerated types
  user defined CR-354
environment command CR-161
environment variables UM-521
  accessed during startup UM-613
  reading into Verilog code CR-359
  referencing from ModelSim command line UM-523
  referencing with VHDL FILE variable UM-523
  setting in Windows UM-522
  specifying library locations in modelsim.ini file UM-525
  specifying UNIX editor CR-157
  state of CR-224
  TranscriptFile, specifying location of UM-534
  used in Solaris linking for FLI UM-568, UM-574
  using in pathnames CR-15
  using with location mapping UM-67
  variable substitution using Tcl UM-479
environment, displaying or changing pathname CR-161
error
  can’t locate C compiler UM-551
Error .ini file variable UM-536
errors
  bad magic number UM-227
  during compilation, locating GR-264
  getting details about messages CR-329
  getting more information UM-546
  libswift entry not found UM-555
  multiple definition UM-186
  onerror command CR-216
  out-of-line function UM-186
  SDF, disabling CR-380
  SystemC loading UM-184
  Tcl_init error UM-552
  void function UM-186
  VSIM license lost UM-555
errors, changing severity of UM-546
escape character CR-15
event order
  changing in Verilog CR-358
  in optimized designs UM-128
  in Verilog simulation UM-132
event queues UM-132
event watching commands, placement of UM-487
events, tracing UM-306
examine command CR-162
examine tooltip
  toggling on/off GR-256
exclusion filter files UM-348
  excluding udp truth table rows UM-349
exclusions
  lines and files UM-347
exit codes UM-549
exit command CR-166
expand net UM-303
Explicit .ini file variable UM-527
export TFs, in DPI UM-551
Exporting SystemC modules
  to Verilog UM-214
exporting SystemC modules
  to VHDL UM-223
exporting top SystemC module UM-165
Expression Builder UM-253
  configuring a List trigger with UM-266
  saving expressions to Tcl variable UM-253
extended identifiers CR-16
  in mixed designs UM-203, UM-222

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

F

-f CR-360
F8 function key UM-607
-fast CR-360
fcover clear command CR-167
fcover comment command CR-168
fcover configue command CR-169
fcover reload command CR-171
fcover report command CR-173
fcover save command CR-175
features, new UM-499
field descriptions
  coverage reports UM-352
FIFOs, viewing SystemC UM-181
File Breakpoint dialog GR-98
File compression
  VCD tasks UM-462
file compression
  SDF files UM-439
  VCD files CR-295, CR-304
file format
  MTI memory data GR-178
file I/O
  splitio command CR-270
  TextIO package UM-88
  VCD files UM-455
file-line breakpoints GR-203
files
  opening in GUI GR-39
files, grouping for compile UM-47
filter
  processes GR-108
Filter instance list dialog GR-92
filtering signals in Objects window GR-185
filters
  for Code Coverage UM-348
find command CR-176
Find in Assertions dialog GR-113
Find in dataflow dialog GR-139
Find in FCovers dialog GR-148
Find in List dialog GR-156
Find in Locals dialog GR-168
Find in memory dialog GR-182
Find in Process dialog GR-109
Find in Transcript dialog
  dialogs
    Find in Transcript GR-48
fixed point types UM-182
FLI UM-100
  debugging UM-399
folders, in projects UM-50
font scaling
  for dual monitors GR-28
fonts
  controlling in X-sessions GR-13
  scaling GR-13
force command CR-180
  defaults UM-539
Force Selected Signal dialog GR-186
foreign language interface UM-100
foreign model loading
  SmartModels UM-618
foreign module declaration
  Verilog example CR-259, UM-210
  VHDL example UM-218
foreign module declaration, SystemC UM-209
format file UM-261
  List window CR-422
  Wave window CR-422, UM-261
FPGA libraries, importing UM-69
function calls, identifying with C Debug UM-407
Functional coverage
  merging databases offline CR-320
functional coverage
  AtLeast counts UM-387
  comments in the database CR-168
  compiling and simulating UM-385
  configuring directives CR-169
  described UM-360
  merging statistics CR-171, UM-396
  reloading CR-171, UM-396
  reporting CR-173, UM-391
  saving database CR-175, UM-395
  weighting directives UM-387
Functional coverage filter dialog GR-151
Functional coverage reload dialog GR-145
Functional coverage report dialog GR-146
functions
  SystemC
    control UM-192
    observe UM-192
    unsupported UM-182

G

-g C++ compiler option UM-178
g++, alternate installations UM-168
gate-level designs
  optimizing UM-127
gdb

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

setting source directory CR-183
gdb debugger UM-400
gdb dir command CR-183
generate statements, Veilog UM-123
GenerateFormat .ini file variable UM-532
generic support
SystemC instantiating VHDL UM-218
generics
assigning or overriding values with -g and -G CR-375
examining generic values CR-162
limitation on assigning composite types CR-376
VHDL UM-195
get_resolution() VHDL function UM-96
getactivecursortime command CR-184
getactivemarkertime command CR-185
glitches
disabling generation
from command line CR-384
from GUI GR-78
global visibility
PLI/FLI shared objects CR-376, UM-581
GlobalSharedObjectsList .ini file variable UM-532
graphic interface UM-237, UM-299, GR-9
UNIX support UM-29
grayed-out menu options UM-501
grouping files for compile UM-47
grouping objects, Monitor window GR-209
GUI preferences, saving GR-266
GUI_expression_format CR-22
GUI expression builder UM-253
syntax CR-23

H

hardware model interface UM-628
'hasX CR-24
Hazard .ini file variable (VLOG) UM-525
hazards
-hazards argument to vlog CR-361
-hazards argument to vsim CR-385
limitations on detection UM-135
help command CR-186
hierarchical reference support, SystemC UM-183
hierarchical references
SystemC/HDL designs UM-192
hierarchical references, mixed-language UM-190
hierarchy
driving signals in UM-419, UM-429
forcing signals in UM-97, UM-425, UM-434
referencing signals in UM-97, UM-422, UM-432
releasing signals in UM-97, UM-427, UM-436
viewing signal names without GR-255
highlighting, in Source window GR-205
history
of commands
shortcuts for reuse CR-19, UM-603
history command CR-187
hm_entity UM-629
HOME environment variable UM-521
HP aCC, restrictions on compiling with UM-169

I

I/O
splitio command CR-270
TextIO package UM-88
VCD files UM-455
icons
shapes and meanings GR-12
ieee .ini file variable UM-525
IEEE libraries UM-65
IEEE Std 1076 UM-30
differences between versions UM-75
IEEE Std 1364 UM-30, UM-113
IgnoreError .ini file variable UM-532
IgnoreFailure .ini file variable UM-532
IgnoreNote .ini file variable UM-532
IgnoreVitalErrors .ini file variable UM-527
IgnoreWarning .ini file variable UM-532
implicit operator, hiding with vcom -explicit CR-318
importing EVCD files, waveform editor GR-295
importing FPGA libraries UM-69
+incdir+ CR-361
incremental compilation
automatic UM-116
manual UM-116
with Verilog UM-115
index checking UM-74
indexed arrays, escaping square brackets CR-15
INF, in a coverage report UM-353
$init_signal_driver UM-429
init_signal_driver UM-419
$init_signal_spy UM-432
init_signal_spy UM-97, UM-422
init_usertfs function UM-412, UM-561
Initial dialog box, turning on/off UM-520
initialization of SystemC state-based code UM-175
initialization sequence UM-615
inlining

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Verilog modules UM-125
VHDL subprograms UM-74
instance
code coverage UM-334
instantiation in mixed-language design
Verilog from VHDL UM-203
VHDL from Verilog UM-207
instantiation in SystemC-Verilog design
SystemC from Verilog UM-214
Verilog from SystemC UM-209
instantiation in SystemC-VHDL design
VHDL from SystemC UM-217
instantiation in VHDL-SystemC design
SystemC from VHDL UM-222
interconnect delays CR-378, UM-451
annotating per Verilog 2001 CR-389
internal signals, adding to a VCD file CR-292
IOPATH
matching to specify path delays UM-445
iteration_limit, infinite zero-delay loops UM-81
IterationLimit .ini file variable UM-532

K

keyboard shortcuts
List window UM-608
Main window UM-605
Source window UM-605
Wave window UM-609
keywords
disabling 2001 keywords CR-366
enabling System Verilog keywords CR-365

L

-L work UM-118
language templates GR-201
language versions, VHDL UM-75
lecho command CR-188
left command CR-189
libraries
64-bit and 32-bit in same library UM-66
archives CR-356
creating UM-60
dependencies, checking CR-328
design libraries, creating CR-356, UM-60
design library types UM-58
design units UM-58
group use, setting up UM-63
IEEE UM-65
importing FPGA libraries UM-69
including precompiled modules GR-71, GR-80
listing contents CR-328
mapping
from the command line UM-62
from the GUI UM-62
hierarchically UM-537
search rules UM-63
modelsim_lib UM-96
moving UM-63
multiple libraries with common modules UM-118
naming UM-62
precompiled modules, including CR-361
predefined UM-64
refreshing library images CR-316, CR-365, UM-66
resource libraries UM-58
std library UM-64
Synopsys UM-65
vendor supplied, compatibility of CR-328
Verilog CR-386, UM-117, UM-194
VHDL library clause UM-64
working libraries UM-58
working vs resource UM-24
working with contents of UM-61
library map file, Verilog configurations UM-122
library mapping, overview UM-25
library maps, Verilog 2001 UM-122
library simulator state variable UM-542
library, definition in ModelSim UM-24
libsm UM-618
libswift UM-618
entry not found error UM-555
License .ini file variable UM-532
licensing
License variable in .ini file UM-532
linking SystemC source UM-172
lint-style checks CR-362
List Signal Properties dialog GR-159
List Signal Search dialog GR-157
List window UM-243, GR-153
adding items to CR-48
context menu GR-155
GUI changes UM-509
setting triggers UM-266
waveform comparison UM-281
see also windows, List window
LM_LICENSE_FILE environment variable UM-521
Load Coverage Data dialog GR-89
loading the design, overview UM-26
Locals window GR-166
see also windows, Locals window

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

location maps, referencing source files UM-67
locations maps
  specifying source files with UM-67
lock message UM-551
locking cursors UM-245
log command CR-191
log file
  log command CR-191
  nolog command CR-209
  overview UM-225
  QuickSim II format CR-413
  redirecting with -l CR-377
  virtual log command CR-343
  virtual nolog command CR-346
  see also WLF files
Logic Modeling
  SmartModel
    command channel UM-622
  SmartModel Windows
    lmcwin commands UM-623
    memory arrays UM-624
long simulations
  saving at intervals UM-231
lshift command CR-193
lsublist command CR-194

M

Macro dialog GR-102
macro_option command CR-195
MacroNestingLevel simulator state variable UM-542
macros (DO files) UM-487
  breakpoints, executing at CR-76
  creating from a saved transcript GR-17
  depth of nesting, simulator state variable UM-542
  error handling UM-490
  executing CR-151
  forcing signals, nets, or registers CR-180
  parameters
    as a simulator state variable (n) UM-542
    passing CR-151, UM-487
    total number passed UM-542
  relative directories CR-151
  shifting parameter values CR-266
  Startup macros UM-538
.main clear command CR-43
Main window GR-14
  code coverage UM-340
  GUI changes UM-500
  see also windows, Main window
manuals UM-35
mapping
  data types UM-193
  libraries
    from the command line UM-62
    hierarchically UM-537
  symbols
    Dataflow window UM-313
  SystemC in mixed designs UM-202
  SystemC to Verilog UM-199
  SystemC to VHDL UM-202
  Verilog states in mixed designs UM-194
  Verilog states in SystemC designs UM-198
  Verilog to SytemC, port and data types UM-198
  Verilog to VHDL data types UM-193
  VHDL to SystemC UM-196
  VHDL to Verilog data types UM-195
mapping libraries, library mapping UM-62
mapping signals, waveform editor GR-295
master slave library (SystemC), including CR-256
math_complex package UM-65
math_real package UM-65
+maxdelays CR-362
mc_scan_plusargs()
  using with an elaboration file UM-84, UM-140
mc_scan_plusargs, PLI routine CR-388
MDI frame UM-501, GR-17
MDI pane
  tab groups GR-18
mem display command CR-196
mem list command CR-198
mem load command CR-199
mem save command CR-202
mem search command CR-204
memories
  displaying the contents of GR-169
  initializing GR-175
  loading memory patterns GR-175
  MTI memory data file GR-178
  MTI's definition of GR-170
  navigating to memory locations GR-182
  saving memory data to a file GR-177
  selecting memory instances GR-171
  sparse memory modeling UM-156
  viewing contents GR-171
  viewing multiple instances GR-171
memory
  modeling in VHDL UM-101
memory allocation profiler UM-318
Memory Declaration, View menu UM-513
memory leak, cancelling scheduled events UM-108

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Memory window GR-169
GUI changes UM-510
modifying display GR-183
see also windows, Memory window
window
Memory window
see also Memory window
memory, displaying contents CR-196
memory, listing CR-198
memory, loading contents CR-199
memory, saving contents CR-202
memory, searching for patterns CR-204
menu options grayed-out UM-501
menus
Dataflow window GR-129
List window GR-154
Main window GR-20
Profiler windows GR-195
Source window GR-206
Wave window GR-216
merging coverage data UM-354, UM-396
merging coverage reports CR-320
messages UM-545
bad magic number UM-227
echoing CR-156
empty port name warning UM-551
exit codes UM-549
getting more information CR-329, UM-546
loading, disbling with -quiet CR-316, CR-365
lock message UM-551
long description UM-546
message system variables UM-536
metavalue detected UM-552
ModelSim message system UM-546
redirecting UM-534
sensitivity list warning UM-552
suppressing warnings from arithmetic packages
UM-538
Tcl_init error UM-552
too few port connections UM-554
turning off assertion messages UM-538
VSIM license lost UM-555
warning, suppressing UM-548
metavalue detected warning UM-552
MGC_LOCATION_MAP env variable UM-67
MGC_LOCATION_MAP variable UM-521
+mindelays CR-362
MinGW gcc UM-569, UM-575
missed coverage
branches GR-120
Missed Coverage pane GR-120
mixed-language simulation UM-188
access limitations UM-190
mnemonics, assigning to signal values CR-354
MODEL_TECH environment variable UM-521
MODEL_TECH_TCL environment variable UM-521
modeling memory in VHDL UM-101
ModelSim
commands CR-31–CR-432
modes of operation UM-27
simulation task overview UM-23
tool structure UM-22
verification flow UM-22
modelsim command CR-206
MODELSIM environment variable UM-521
modelsim.ini
found by ModelSim UM-615
default to VHDL93 UM-539
delay file opening with UM-539
environment variables in UM-537
force command default, setting UM-539
hierarchical library mapping UM-537
opening VHDL files UM-539
restart command defaults, setting UM-539
startup file, specifying with UM-538
transcript file created from UM-537
turning off arithmetic package warnings UM-538
turning off assertion messages UM-538
modelsim.tcl file GR-266
modelsim_lib UM-96
path to UM-525
MODELSIM_TCL environment variable UM-521
modes of operation, ModelSim UM-27
Modified field, Project tab UM-45
Modify Breakpoints dialog GR-95
Modify Display Properties dialog GR-162
modules
handling multiple, common names UM-118
with unnamed ports UM-206
Monitor window
adding items to CR-51
grouping/ungrouping objects GR-209
monitor window GR-208
add monitor command CR-51
monitors, dual, font scaling GR-28
mouse shortcuts
Main window UM-605
Source window UM-605
Wave window UM-609
.mpf file UM-38
loading from the command line UM-55
order of access during startup UM-612

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

MTI memory data file GR-178
mti_cosim_trace environment variable UM-521
mti_inhibit_inline attribute UM-74
MTI_SYSTEMC macro UM-168
MTI_TF_LIMIT environment variable UM-522
multiclocked assertions UM-369
multiple document interface UM-501, GR-17
multiple drivers on unresolved signal GR-53, GR-62
Multiple simulations UM-225
multi-source interconnect delays CR-378

N

n simulator state variable UM-542
name case sensitivity, VHDL vs. Verilog CR-15
Name field
    Project tab UM-45
name visibility in Verilog generates UM-123
names, modules with the same UM-118
negative pulses
    driving an error state CR-388
Negative timing
    $setuphold/$recovery UM-150
negative timing
    algorithm for calculating delays UM-136
    check limits UM-136
    extending check limits CR-385
nets
    Dataflow window, displaying in UM-300, GR-128
    drivers of, displaying CR-154
    readers of, displaying CR-242
    stimulus CR-180
    values of
        displaying in Objects window GR-184
        examining CR-162
        saving as binary log file UM-226
    waveforms, viewing GR-211
new features UM-499
next and previous edges, finding UM-610
next command CR-207
Nlview widget Symlib format UM-313
no space in time literal GR-53, GR-62
-no_risefall_delaynets CR-387
NoCaseStaticError .ini file variable UM-527
NoDebug .ini file variable (VCOM) UM-527
NoDebug .ini file variable (VLOG) UM-526
-nodebug argument (vcom) CR-314
-nodebug argument (vlog) CR-363
noforce command CR-208
NoIndexCheck .ini file variable UM-527
+nolibcell CR-363
nolog command CR-209
NOMMAP environment variable UM-522
non-blocking assignments UM-134
NoOthersStaticError .ini file variable UM-527
NoRangeCheck .ini file variable UM-527
Note .ini file variable UM-536
notepad command CR-211
Notepad windows, text editing UM-605
-notrigger argument UM-268
noview command CR-212
NoVital .ini file variable UM-527
NoVitalCheck .ini file variable UM-527
Now simulator state variable UM-542
now simulator state variable UM-542
+nowarn<CODE> CR-364
nowhen command CR-213
numeric_bit package UM-65
numeric_std package UM-65
    disabling warning messages UM-538
NumericStdNoWarnings .ini file variable UM-533

O

object
    defined UM-34
object_list_file, WLF files CR-416
Objects window GR-184
    see also windows, Objects window
observe function, SystemC UM-192
observe_foreign_signal() function UM-183
onbreak command CR-214
onElabError command CR-215
onerror command CR-216
Open File dialog GR-39
opening files GR-39
operating systems supported, *See Installation Guide*
Optimization Configuration dialog GR-45
Optimization Configurations UM-49
optimizations
    disabling for Verilog designs CR-364
    disabling for VHDL designs CR-315
    disabling process merging CR-311
    gate-level designs UM-127
    Verilog designs UM-124
    VHDL subprogram inlining UM-74
    via the gui GR-70
    vopt command CR-371
optimize for std_logic_1164 GR-53, GR-62
Optimize_1164 .ini file variable UM-527

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

optimizing Verilog designs
  design object visibility UM-126
  event order issues UM-128
  timing checks UM-128
OptionFile entry in project files GR-56, GR-65
order of events
  changing in Verilog CR-358
  in optimized designs UM-128
ordering files for compile UM-46
organizing projects with folders UM-50
organizing windows, MDI pane GR-18
OSCI 2.1 features supported UM-183
OSCI simulator, differences from ModelSim UM-182
OSCI simulator, differences with vsim UM-182
others .ini file variable UM-525
overriding the simulator resolution UM-174
overview, simulation tasks in ModelSim UM-23

P

packages
  standard UM-64
  textio UM-64
  util UM-96
  VITAL 1995 UM-93
  VITAL 2000 UM-93
page setup
  Dataflow window UM-312
  Wave window UM-262, GR-230
pan, Dataflow window UM-305
panes
  docking and undocking GR-258
parameter support
  SystemC instantiating Verilog UM-211
  Verilog instantiating SystemC UM-214
parameters
  making optional UM-488
  using with macros CR-151, UM-487
path delay mode UM-145
path delays,matching to IOPATH statements UM-445
pathnames
  comparisons UM-279
  hiding in Wave window UM-255
  in VSIM commands CR-12
  spaces in CR-11
PathSeparator .ini file variable UM-533
pause command CR-217
PedanticErrors .ini file variable UM-527
performance
  cancelling scheduled events UM-108
  improving for Verilog simulations UM-124
  vopt command CR-371
platforms supported, *See Installation Guide*
play command CR-218
PLI
  loading shared objects with global symbol visibility CR-376, UM-581
  specifying which apps to load UM-562
  Veriuser entry UM-562
PLI/VPI UM-158, UM-560
  debugging UM-399
  tracing UM-599
PLIOBJS environment variable UM-522, UM-562
pop command CR-219
popup
  toggling waveform popup on/off UM-280, GR-256
Port driver data, capturing UM-467
ports, unnamed, in mixed designs UM-206
ports, VHDL and Verilog UM-193
Postscript
  saving a waveform in UM-262
  saving the Dataflow display in UM-310
power add command CR-220
power report command CR-221
power reset command CR-222
pragmas UM-347
precedence of variables UM-541
precision, simulator resolution UM-129, UM-191
pref.tcl file GR-266
Preference dialog GR-104
preference variables
  .ini files, located in UM-524
  editing GR-266
  saving GR-266
  Tcl files, located in GR-266
Preferences
  drag and drop GR-103
preferences, saving GR-266
primitives, symbols in Dataflow window UM-313
Print dialog GR-135
Print Postscript dialog GR-137
printenv command CR-223, CR-224
printing
  Dataflow window display UM-310
  waveforms in the Wave window UM-262
Process window GR-143
  see also windows, Process window
processes
  optimizations, disabling merging CR-311
  without wait statements GR-53, GR-62
profile clear command CR-225

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

profile interval command CR-226
profile off command CR-227
profile on command CR-228
profile option command CR-229
profile reload command CR-230
profile report command CR-231, UM-331
Profile Report dialog GR-93, GR-197
Profiler UM-317
  %parent fields UM-325
  clear profile data UM-321
  enabling memory profiling UM-319
  enabling statistical sampling UM-321
  getting started UM-319
  handling large files UM-320
  Hierarchical View UM-325
  interpreting data UM-323
  memory allocation UM-318
  memory allocation profiling UM-321
  profile report command UM-331
  Profile Report dialog UM-332, GR-93
  Ranked View UM-324
  report option UM-331
  reporting GR-93
  results, viewing UM-324
  statistical sampling UM-318
  Structural View UM-326
  unsupported on Opteron UM-317
  view_profile command UM-324
  viewing profile details UM-327
Programming Language Interface UM-158, UM-560
Project Compiler Settings dialog GR-50
Project Settings dialog GR-57
project tab
  information in UM-45
  sorting UM-45
Projects
  MODELSIM environment variable UM-521
projects UM-37
  accessing from the command line UM-55
  adding files to UM-41
  benefits UM-38
  code coverage settings UM-338
  compile order UM-46
    changing UM-46
  compiler properties in UM-52
  compiling files UM-43
  creating UM-40
  creating simulation configurations UM-48
  folders in UM-50
  grouping files in UM-47
  loading a design UM-44
  override mapping for work directory with vcom CR-256, CR-317
  override mapping for work directory with vlog CR-366
  overview UM-38
propagation, preventing X propagation CR-378
Properties (memory) dialog GR-183
property list command CR-234
property wave command CR-235
Protect .ini file variable (VLOG) UM-526
‘protect compiler directive UM-155
protected types UM-101
PSL
  assume directives UM-363
  endpoint directives UM-398
  standard supported UM-30
PSL assertions UM-359
  see also assertions
pulse error state CR-388
push command CR-237
pwd command CR-238

Q

quick reference
  table of ModelSim tasks UM-23
QuickSim II logfile format CR-413
Quiet .ini file variable
  VCOM UM-527
Quiet .ini file variable (VLOG) UM-526
quietly command CR-239
quit command CR-240

R

race condition, problems with event order UM-132
radix
  changing in Objects, Locals, Dataflow, List, and Wave windows CR-241
  character strings, displaying CR-354
  default, DefaultRadix variable UM-531
  List window UM-259
  of signals being examined CR-163
  of signals in Wave window CR-54
  specifying in Memory window GR-183
  Wave window UM-255
radix command CR-241
range checking UM-74
  disabling CR-315
  enabling CR-316

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

readers and drivers UM-303
readers command CR-242
real type, converting to time UM-99
reative testbenches, PSL endpoints UM-398
rebuilding supplied libraries UM-65
reconstruct RTL-level design busses UM-234
record command CR-243
record field selection, syntax CR-13
records, values of, changing GR-167
$recovery UM-150
redirecting messages, TranscriptFile UM-534
reference region UM-275
refreshing library images CR-316, CR-365, UM-66
registered function calls UM-407
registers
  values of
    displaying in Objects window GR-184
    saving as binary log file UM-226
  waveforms, viewing GR-211
report
  simulator control UM-520
  simulator state UM-520
report command CR-244
reporting
  code coverage UM-350
  variable settings CR-17
RequireConfigForAllDefaultBinding variable UM-527
resolution
  in SystemC simulation UM-174
  mixed designs UM-191
  overriding in SystemC UM-174
  returning as a real UM-96
  specifying with -t argument CR-380
  verilog simulation UM-129
  VHDL simulation UM-78
Resolution .ini file variable UM-533
resolution simulator state variable UM-542
resource libraries UM-64
restart command CR-246
  defaults UM-539
  in GUI GR-26
  toolbar button GR-35, GR-127, GR-222
Restart dialog GR-88
restore command CR-248
restoring defaults UM-520
results, saving simulations UM-225
resume command CR-249
right command CR-250
RTL-level design busses
  reconstructing UM-234
run command CR-252
RunLength .ini file variable UM-533
Runtime Options dialog GR-85

S

Save Memory dialog GR-177
saving
  simulation options in a project UM-48
  waveforms UM-225
saving simulations UM-86, UM-142
sc_argc() function UM-183
sc_argv() function UM-183
sc_clock() functions, moving UM-164
sc_cycle() function UM-182
sc_fifo UM-181
sc_foreign_module UM-217
  and parameters UM-211
sc_initialize(), removing calls UM-182
sc_main() function UM-182
sc_main() function, converting UM-164
SC_MODULE_EXPORT macro UM-165
sc_set_time_resolution() function UM-182
sc_start() function UM-182
sc_start() function, replacing in SystemC UM-182
sc_start(), replacing for ModelSim UM-164
ScalarOpts .ini file variable UM-526, UM-527
scaling fonts GR-13
sccom
  using sccom vs. raw C++ compiler UM-170
sccom command CR-254
sccom -link command UM-172, UM-223
sccomLogfile .ini file variable (sccom) UM-528
sccomVerbose .ini file variable (sccom) UM-528
scgenmod command CR-258
scgenmod, using UM-209, UM-217
-sclib command CR-389
scope, setting region environment CR-161
SCV library, including CR-255
SDF
  controlling missing instance messages CR-380
  disabling individual checks CR-275
  disabling timing checks UM-451
  errors and warnings UM-441
  errors on loading, disabling CR-380
  instance specification UM-440
  interconnect delays UM-451
  mixed VHDL and Verilog designs UM-450
  specification with the GUI UM-441
  troubleshooting UM-452
  Verilog

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

$sdf_annotate system task UM-444
optional conditions UM-449
optional edge specifications UM-448
rounded timing values UM-449
SDF to Verilog construct matching UM-445
VHDL
resolving errors UM-443
SDF to VHDL generic matching UM-442
warning messages, disabling CR-380
$sdf_done UM-152
search command CR-260
search libraries CR-386, GR-71, GR-80
searching
binary signal values in the GUI CR-29
Expression Builder UM-253
in the source window GR-204
List window
signal values, transitions, and names CR-22, CR-152, CR-290
next and previous edge in Wave window CR-189, CR-250
Verilog libraries UM-117, UM-207
Wave window
signal values, edges and names CR-189, CR-250, GR-233
searchlog command CR-262
seetime command CR-264
sensitivity list warning UM-552
setenv command CR-265
$setuphold UM-150
severity, changing level for errors UM-546
shared library
building in SystemC UM-172, GR-25
shared objects
loading FLI applications
see ModelSim FLI Reference manual
loading PLI/VPI C applications UM-568
loading PLI/VPI C++ applications UM-574
loading with global symbol visibility CR-376, UM-581
shift command CR-266
Shortcuts
text editing UM-605
shortcuts
command history CR-19, UM-603
command line caveat CR-18, UM-603
List window UM-608
Main window UM-605
Source window UM-605
Wave window UM-609
show command CR-267
show drivers
Dataflow window UM-303
Wave window UM-269
show source lines with errors GR-52, GR-61
Show_BadOptionWarning .ini file variable UM-526
Show_Lint .ini file variable (VLOG) UM-526, UM-527
Show_source .ini file variable
VCOM UM-528
Show_source .ini file variable (VLOG) UM-526
Show_VitalChecksWarning .ini file variable UM-528
Show_Warning1 .ini file variable UM-528
Show_Warning2 .ini file variable UM-528
Show_Warning3 .ini file variable UM-528
Show_Warning4 .ini file variable UM-528
Show_Warning5 .ini file variable UM-528
Show3DMem .ini file variable UM-533
ShowEnumMem .ini file variable UM-533
ShowIntMem .ini file variable UM-533
Signal Breakpoints dialog GR-97
signal interaction
Verilog and SystemC UM-196
Signal Spy UM-97, UM-422
overview UM-418
using in PSL assertions UM-367
$signal_force UM-434
signal_force UM-97, UM-425
$signal_release UM-436
signal_release UM-97, UM-427
signals
alternative names in the List window (-label) CR-48
alternative names in the Wave window (-label) CR-53
applying stimulus to GR-186
attributes of, using in expressions CR-24
breakpoints CR-407
combining into a user-defined bus CR-53, UM-264
Dataflow window, displaying in UM-300, GR-128
drivers of, displaying CR-154
driving in the hierarchy UM-419
environment of, displaying CR-161
filtering in the Objects window GR-185
finding CR-176
force time, specifying CR-181
hierarchy
driving in UM-419, UM-429
referencing in UM-97, UM-422, UM-432
releasing anywhere in UM-427
releasing in UM-97, UM-436
log file, creating CR-191
names of, viewing without hierarchy GR-255
pathnames in VSIM commands CR-12

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

radix
specifying for examine CR-163
specifying in List window CR-49
specifying in Wave window CR-54
readers of, displaying CR-242
sampling at a clock change UM-268
states of, displaying as mnemonics CR-354
stimulus CR-180
transitions, searching for UM-249
types, selecting which to view GR-185
unresolved, multiple drivers on GR-53, GR-62
values of
displaying in Objects window GR-184
examining CR-162
forcing anywhere in the hierarchy UM-97, UM-425, UM-434
replacing with text CR-354
saving as binary log file UM-226
waveforms, viewing GR-211
Signals (Objects) window UM-514
SimulateAssumeDirectives .ini file variable UM-533
Simulating
Comparing simulations UM-225
simulating
batch mode UM-27
command-line mode UM-27
default run length GR-86
delays, specifying time units for CR-18
design unit, specifying CR-373
elaboration file UM-82, UM-138
graphic interface to GR-76
iteration limit GR-86
mixed language designs
compilers UM-190
libraries UM-190
resolution limit in UM-191
mixed Verilog and SystemC designs
channel and port type mapping UM-196
SystemC sc_signal data type mapping UM-197
Verilog port direction UM-198
Verilog state mapping UM-198
mixed Verilog and VHDL designs
Verilog parameters UM-193
Verilog state mapping UM-194
VHDL and Verilog ports UM-193
VHDL generics UM-195
mixed VHDL and SystemC designs
SystemC state mapping UM-202
VHDL port direction UM-201
VHDL port type mapping UM-200
VHDL sc_signal data type mapping UM-200
optimizing Verilog performance CR-360
saving dataflow display as a Postscript file UM-310
saving options in a project UM-48
saving simulations CR-191, CR-382, UM-225
saving waveform as a Postscript file UM-262
speeding-up with the Profiler UM-317
stepping through a simulation CR-272
stimulus, applying to signals and nets GR-186
stopping simulation in batch mode CR-410
SystemC UM-159, UM-173
usage flow for SystemC only UM-163
time resolution GR-77
Verilog UM-129
delay modes UM-144
hazard detection UM-135
optimizing performance UM-124
resolution limit UM-129
XL compatible simulator options UM-136
VHDL UM-78
viewing results in List window UM-243, GR-153
VITAL packages UM-95
simulating the design, overview UM-26
simulation
basic steps for UM-24
Simulation Configuration
creating UM-48
dialog GR-46
simulations
event order in UM-132
saving results CR-143, CR-144, UM-225
saving results at intervals UM-231
saving with checkpoint UM-86, UM-142
simulator resolution
mixed designs UM-191
returning as a real UM-96
SystemC UM-174
Verilog UM-129
VHDL UM-78
vsim -t argument CR-380
simulator state variables UM-542
simulator version CR-381, CR-392
simulator, ModelSim and OSCI differences UM-182
simultaneous events in Verilog
changing order CR-358
sizetf callback function UM-587
sm_entity UM-619
SmartModels
creating foreign architectures with sm_entity UM-619
invoking SmartModel specific commands UM-622
linking to UM-618

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

lmcwin commands UM-623
memory arrays UM-624
Verilog interface UM-625
VHDL interface UM-618
so, shared object file
loading PLI/VPI C applications UM-568
loading PLI/VPI C++ applications UM-574
software version GR-33
source balloon
C Debug GR-99
source code pragmas UM-347
source code, security UM-70, UM-155
source directory, setting from source window GR-21, GR-206
source errors, locating during compilation GR-264
source files, referencing with location maps UM-67
source files, specifying with location maps UM-67
source highlighting, customizing GR-205
source libraries
arguments supporting UM-119
source lines with errors
showing GR-52, GR-61
Source window GR-199
code coverage data UM-341
colorization GR-205
tab stops in GR-205
see also windows, Source window
source-level debug
SystemC, enabling UM-178
spaces in pathnames CR-11
sparse memories
listing with write report CR-426
sparse memory modeling UM-156
SparseMemThreshhold .ini file variable UM-526
specify path delays CR-388
matching to IOPATH statements UM-445
speeding-up the simulation UM-317
splitio command CR-270
square brackets, escaping CR-15
stability checking
disabling CR-91
enabling CR-92
Standard Developer's Kit User Manual UM-35
standards supported UM-30
Start Simulation dialog GR-76
start_of_simulation() function UM-183
Startup
macros UM-538
startup
alternate to startup.do (vsim -do) CR-374
environment variables access during UM-613
files accessed during UM-612
macro in the modelsim.ini file UM-534
startup macro in command-line mode UM-27
using a startup file UM-538
Startup .ini file variable UM-534
state variables UM-542
statistical sampling profiler UM-318
status bar
Main window GR-19
status command CR-271
Status field
Project tab UM-45
std .ini file variable UM-525
std_arith package
disabling warning messages UM-538
std_developerskit .ini file variable UM-525
Std_logic
mapping to binary radix CR-29
std_logic_arith package UM-65
std_logic_signed package UM-65
std_logic_textio UM-65
std_logic_unsigned package UM-65
StdArithNoWarnings .ini file variable UM-534
STDOUT environment variable UM-522
step command CR-272
steps for simulation, overview UM-24
stimulus
applying to signals and nets GR-186
modifying for elaboration file UM-83, UM-139
stop command CR-273
struct of sc_signal<T> UM-180
subprogram inlining UM-74
subprogram write is ambiguous error, fixing UM-90
Support UM-36
Suppress .ini file variable UM-536
symbol mapping
Dataflow window UM-313
symbolic constants, displaying CR-354
symbolic link to design libraries (UNIX) UM-63
symbolic names, assigning to signal values CR-354
Synopsis hardware modeler UM-628
synopsys .ini file variable UM-525
Synopsys libraries UM-65
syntax highlighting GR-205
synthesis
rule compliance checking CR-312, UM-527, GR-52, GR-61
system calls
VCD UM-461
Verilog UM-146
system commands UM-479

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

system tasks
  VCD UM-461
  Verilog UM-146
  Verilog-XL compatible UM-150
system tasks and functions
  ModelSim Verilog UM-152
System Verilog
  enabling with -sv argument CR-365
SystemC
  aggregates of signals/ports UM-180
  class and structure member naming syntax CR-13
  compiling for source level debug UM-167
  compiling optimized code UM-167
  component declaration for instantiation UM-223
  control function UM-192
  converting sc_main() UM-164
  exporting sc_main, example UM-165
  exporting top level module UM-165
  foreign module declaration UM-209
  generic support, instantiating VHDL UM-218
  hierarchical reference support UM-183
  hierarchical references in mixed designs UM-192
  instantiation criteria in Verilog design UM-214
  instantiation criteria in VHDL design UM-222
  Link dialog GR-68
  linking the compiled source UM-172
  maintaining design portability UM-168
  mapping states in mixed designs UM-202
    VHDL UM-202
  master slave library, including CR-256
  mixed designs with Verilog UM-188
  mixed designs with VHDL UM-188
  observe function UM-192
  parameter support, Verilog instances UM-211
  prim channel aggregates UM-180
  replacing sc_start() UM-164
  sc_clock(), moving to SC_CTOR UM-164
  sc_fifo UM-181
  simulating UM-173
  source code, modifying for ModelSim UM-164
  specifying shared library path, command CR-389
  stack space for threads UM-184
  state-based code, initializing and cleanup UM-175
  troubleshooting UM-184
  unsupported functions UM-182
  verification library, including CR-255
  viewable/debuggable objects UM-176
  viewing FIFOs UM-181
  virtual functions UM-175
SystemC modules
  exporting for use in Verilog UM-214
  exporting for use in VHDL UM-223
SystemVerilog UM-30
SystemVerilog DPI
  registering DPIapplications UM-565
  specifying the DPI file to load UM-580

T

tab groups GR-18
tab stops
  Source window GR-205
tb command CR-274
tcheck_set command CR-275
tcheck_status command CR-277
Tcl UM-472–UM-482
  command separator UM-478
  command substitution UM-477
  command syntax UM-474
  evaluation order UM-478
  history shortcuts CR-19, UM-603
  Man Pages in Help menu GR-33
  preference variables GR-266
  relational expression evaluation UM-478
  time commands UM-481
  variable
    in when commands CR-408
    substitution UM-479
  VSIM Tcl commands UM-480
Tcl_init error message UM-552
Technical support and updates UM-36
temp files, VSOUT UM-523
test signal
  delaying GR-244
testbench, accessing internal objectsfrom UM-417
testbenches
  PSL endpoint reactivity UM-398
text and command syntax UM-34
Text editing UM-605
TEXTIO
  buffer, flushing UM-92
TextIO package
  alternative I/O files UM-92
  containing hexadecimal numbers UM-91
  dangling pointers UM-91
  ENDFILE function UM-91
  ENDLINE function UM-91
  file declaration UM-88
  implementation issues UM-90
  providing stimulus UM-92
  standard input UM-89

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

standard output UM-89
WRITE procedure UM-90
WRITE_STRING procedure UM-90
TF routines UM-593, UM-595
TFMPC
disabling warning CR-387
explanation UM-554
time
absolute, using @ CR-18
measuring in Wave window UM-245
resolution in SystemC UM-174
simulation time units CR-18
time resolution as a simulator state variable UM-542
time collapsing CR-382, UM-232
time literal, missing space GR-53, GR-62
time resolution
in mixed designs UM-191
in Verilog UM-129
in VHDL UM-78
setting
with the GUI GR-77
with vsim command CR-380
time type
converting to real UM-98
time, time units, simulation time CR-18
timescale directive warning
disabling CR-387
investigating UM-130
timing
$setuphold/$recovery UM-150
annotation UM-439
differences shown by comparison UM-279
disabling checks CR-364, UM-451
disabling checks for entire design CR-379
disabling individual checks CR-275
in optimized designs UM-128
negative check limits
described UM-136
extending CR-385
status of individual checks CR-277
title, Main window, changing CR-381
TMPDIR environment variable UM-522
to_real VHDL function UM-98
to_time VHDL function UM-99
toggle add command CR-279
toggle coverage
excluding signals CR-281
toggle disable command CR-281
toggle enable command CR-282
toggle report command CR-283
toggle reset command CR-284
toggle statistics
enabling CR-279
reporting CR-283
resetting CR-284
toggling waveform popup on/off UM-280, GR-256
tolerance
leading edge UM-276
trailing edge UM-276
too few port connections, explanation UM-554
toolbar
Dataflow window GR-132
Main window GR-34
Wave window GR-220
waveform editor GR-222
tooltip, toggling waveform popup GR-256
tracing
events UM-306
source of unknown UM-307
transcribe command CR-285
transcript
clearing CR-43
disable file creation UM-537, GR-17
file name, specifed in modelsim.ini UM-537
redirecting with -l CR-377
reducing file size CR-287
saving GR-16
using as a DO file GR-17
transcript command CR-286
transcript file command CR-287
TranscriptFile .ini file variable UM-534
transitions, signal, finding CR-189, CR-250
TreeUpdate command CR-423
triggers, in the List window UM-266
triggers, in the List window, setting UM-265, GR-163
troubleshooting
SystemC UM-184
unexplained behaviors, SystemC UM-184
TSCALE, disabling warning CR-387
TSSI CR-429
in VCD files UM-467
tssi2mti command CR-288
type
converting real to time UM-99
converting time to real UM-98
Type field, Project tab UM-45
types, fixed point in SystemC UM-182

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

U

-u CR-365
unbound component GR-53, GR-62
UnbufferedOutput .ini file variable UM-534
undeclared nets, reporting an error CR-362
undefined symbol, error UM-184
unexplained behavior during simulation UM-184
unexplained simulation behavior UM-184
ungrouping objects, Monitor window GR-209
unit delay mode UM-145
unknowns, tracing UM-307
unnamed ports, in mixed designs UM-206
unresolved signals, multiple drivers on GR-53, GR-62
unsetenv command CR-289
unsupported functions in SystemC UM-182
up command CR-290
UpCase .ini file variable UM-526
use 1076-1993 language standard GR-51, GR-60
use clause, specifying a library UM-64
use explicit declarations only GR-52, GR-61
use flow
  Code Coverage UM-334
  SystemC-only designs UM-163
UseCsupV2 .ini file variable UM-534
user hook Tcl variable GR-107
user-defined bus CR-53, UM-233, UM-264
UserTimeUnit .ini file variable UM-534
UseScv .ini file variable (sccom) UM-528
util package UM-96

V

-v CR-366
v2k_int_delays CR-389
values
  describe HDL items CR-147
  examine HDL item values CR-162
  of HDL items GR-203
  replacing signal values with strings CR-354
variable settings report CR-17
variables
  describing CR-147
  environment variables UM-521
  LM_LICENSE_FILE UM-521
  personal preferences UM-520
  precedence between .ini and .tcl UM-541
  reading from the .ini file UM-536
  referencing in commands CR-17
  setting environment variables UM-521
  simulator state variables
    current settings report UM-520
    iteration number UM-542
    name of entity or module as a variable UM-542
    resolution UM-542
    simulation time UM-542
  value of
    changing from command line CR-81
    changing with the GUI GR-167
    examining CR-162
  values of
    displaying in Objects window GR-184
    saving as binary log file UM-226
Variables (Locals) window UM-518
variables, Tcl, user hook GR-107
vcd add command CR-292
vcd checkpoint command CR-293
vcd comment command CR-294
vcd dumpports command CR-295
vcd dumpportsall command CR-297
vcd dumpportsflush command CR-298
vcd dumpportslimit command CR-299
vcd dumpportsoff command CR-300
vcd dumpportson command CR-301
vcd file command CR-302
VCD files UM-455
  adding items to the file CR-292
  capturing port driver data CR-295, UM-467
  case sensitivity UM-456
  converting to WLF files CR-310
  creating CR-292, UM-456
  dumping variable values CR-293
  dumpports tasks UM-461
  flushing the buffer contents CR-306
  from VHDL source to VCD output UM-463
  generating from WLF files CR-415
  inserting comments CR-294
  internal signals, adding CR-292
  specifying maximum file size CR-307
  specifying name of CR-304
  specifying the file name CR-302
  state mapping CR-302, CR-304
  stimulus, using as UM-458
  supported TSSI states UM-467
  turn off VCD dumping CR-308
  turn on VCD dumping CR-309
  VCD system tasks UM-461
  viewing files from another tool CR-310
vcd files command CR-304
vcd flush command CR-306
vcd limit command CR-307

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

vcd off command CR-308
vcd on command CR-309
vcd2wlf command CR-310
vcom
  enabling code coverage UM-337
vcom command CR-311
vcover command UM-354
vcover convert command CR-319
vcover merge command CR-320
vcover report command CR-322
vdel command CR-327
vdir command CR-328
vector elements, initializing CR-81
vendor libraries, compatibility of CR-328
Vera, see Vera documentation
Verilog
  ACC routines UM-591
  capturing port driver data with -dumpports CR-302, UM-467
  cell libraries UM-144
  compiler directives UM-153
  compiling and linking PLI C applications UM-568
  compiling and linking PLI C++ applications UM-574
  compiling design units UM-114
  compiling with XL 'uselib compiler directive UM-120
  component declaration UM-204
  configurations UM-122
  event order in simulation UM-132
  generate statements UM-123
  instantiation criteria in mixed-language design UM-203
  instantiation criteria in SystemC design UM-209
  instantiation of VHDL design units UM-207
  language templates GR-201
  library usage UM-117
  mapping states in mixed designs UM-194
  mapping states in SystemC designs UM-198
  mixed designs with SystemC UM-188
  mixed designs with VHDL UM-188
  parameter support, instantiating SystemC UM-214
  parameters UM-193
  port direction UM-198
  sc_signal data type mapping UM-197
  SDF annotation UM-444
  sdf_annotate system task UM-444
  simulating UM-129
    delay modes UM-144
    XL compatible options UM-136
  simulation hazard detection UM-135
  simulation resolution limit UM-129
  SmartModel interface UM-625
  source code viewing GR-199
  standards UM-30
  system tasks UM-146
  TF routines UM-593, UM-595
  to SystemC, channel and port type mapping UM-196
  XL compatible compiler options UM-119
  XL compatible routines UM-597
  XL compatible system tasks UM-150
verilog .ini file variable UM-525
Verilog 2001
  disabling support CR-366, UM-526
Verilog PLI/VPI
  64-bit support in the PLI UM-598
  compiling and linking PLI/VPI C applications UM-568
  compiling and linking PLI/VPI C++ applications UM-574
  debugging PLI/VPI code UM-599
  PLI callback reason argument UM-585
  PLI support for VHDL objects UM-590
  registering PLI applications UM-561
  registering VPI applications UM-563
  specifying the PLI/VPI file to load UM-580
Verilog-XL
  compatibility with UM-111, UM-559
Veriuser .ini file variable UM-534, UM-562
Veriuser, specifying PLI applications UM-562
veriuser.c file UM-589
verror command CR-329
version
  obtaining via Help menu GR-33
  obtaining with vsim command CR-381
  obtaining with vsim<info> commands CR-392
vgencomp command CR-330
VHDL
  compiling design units UM-73
  creating a design library UM-73
  delay file opening UM-539
  dependency checking UM-73
  field naming syntax CR-13
  file opening delay UM-539
  foreign language interface UM-100
  hardware model interface UM-628
  instantiation criteria in SystemC design UM-217
  instantiation from Verilog UM-207
  instantiation of Verilog UM-193
  language templates GR-201
  language versions UM-75

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

library clause UM-64
mixed designs with SystemC UM-188
mixed designs with Verilog UM-188
object support in PLI UM-590
optimizations
inlining UM-74
port direction UM-201
port type mapping UM-200
sc_signal data type mapping UM-200
simulating UM-78
SmartModel interface UM-618
source code viewing GR-199
standards UM-30
timing check disabling UM-78
VITAL package UM-65
VHDL utilities UM-96, UM-97, UM-422, UM-432
get_resolution() UM-96
to_real() UM-98
to_time() UM-99
VHDL-1987, compilation problems UM-75
VHDL-1993, enabling support for CR-311, UM-528
VHDL-2002, enabling support for CR-311, UM-528
VHDL93 .ini file variable UM-528
view command CR-332
view_profile command UM-324
viewing
library contents UM-61
waveforms CR-382, UM-225
viewing FIFOs UM-181
virtual count commands CR-334
virtual define command CR-335
virtual delete command CR-336
virtual describe command CR-337
virtual expand commands CR-338
virtual function command CR-339
virtual functions in SystemC UM-175
virtual hide command CR-342, UM-234
virtual log command CR-343
virtual nohide command CR-345
virtual nolog command CR-346
virtual objects UM-233
virtual functions UM-234
virtual regions UM-235
virtual signals UM-233
virtual types UM-235
virtual region command CR-348, UM-235
virtual regions
reconstruct the RTL hierarchy in gate-level design UM-235
virtual save command CR-349, UM-234
virtual show command CR-350
virtual signal command CR-351, UM-233
virtual signals
reconstruct RTL-level design busses UM-234
reconstruct the original RTL hierarchy UM-234
virtual hide command UM-234
virtual type command CR-354
visibility
column in structure tab UM-228
VITAL
compiling and simulating with accelerated VITAL packages UM-95
compliance warnings UM-94
disabling optimizations for debugging UM-95
specification and source code UM-93
VITAL packages UM-93
vital95 .ini file variable UM-525
vlib command CR-356
vlog
enabling code coverage UM-337
vlog command CR-358
vlog.opt file GR-56, GR-65
vlog95compat .ini file variable UM-526
vmake command CR-369
vmap command CR-370
vopt
gui access GR-70
vopt command CR-371, UM-124
VoptFlow .ini file variable UM-534
VPI, registering applications UM-563
VPI/PLI UM-158, UM-560
compiling and linking C applications UM-568
compiling and linking C++ applications UM-574
vsim build date and version CR-392
vsim command CR-373
VSIM license lost UM-555
vsim, differences with OSCI simulator UM-182
VSOUT temp file UM-523

W

Warning .ini file variable UM-536
WARNING[8], -lint argument to vlog CR-362
warnings
changing severity of UM-546
disabling at time 0 UM-538
empty port name UM-551
exit codes UM-549
getting more information UM-546
messages, long description UM-546
metavalue detected UM-552

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

SDF, disabling CR-380
suppressing VCOM warning messages CR-315, UM-548
suppressing VLOG warning messages CR-364, UM-548
suppressing VSIM warning messages CR-387, UM-548
Tcl initialization error 2 UM-552
too few port connections UM-554
turning off warnings from arithmetic packages UM-538
waiting for lock UM-551
watching a signal value GR-208
watching signal values CR-51
wave commands CR-394
wave create command CR-397
wave edit command CR-400
wave export command CR-403
wave import command CR-404
Wave Log Format (WLF) file UM-225
wave log format (WLF) file CR-382
of binary signal values CR-191
see also WLF files
wave modify command CR-405
wave viewer, Dataflow window UM-304
Wave window UM-240, GR-211
adding items to CR-52
compare waveforms UM-279
docking and undocking UM-241, GR-212
in the Dataflow window UM-304
saving layout UM-261
toggling waveform popup on/off UM-280, GR-256
values column UM-280
see also windows, Wave window
WaveActivateNextPane command CR-423
Waveform Compare
created waveforms, using with GR-296
Waveform Comparison CR-94
add region UM-275
adding signals UM-274
clocked comparison UM-276
compare by region UM-275
compare by signal UM-274
compare options UM-278
compare tab UM-273
comparison method UM-276
comparison method tab UM-276
delaying the test signal GR-244
difference markers UM-279
flattened designs UM-283
hierarchical designs UM-283
icons UM-281
introduction UM-270
leading edge tolerance UM-276
List window display UM-281
pathnames UM-279
reference dataset UM-272
reference region UM-275
test dataset UM-273
timing differences UM-279
trailing edge tolerance UM-276
values column UM-280
Wave window display UM-279
Waveform Editor
Waveform Compare, using with GR-296
waveform editor
creating waveforms GR-289
creating waves CR-397
editing commands CR-400
editing waveforms GR-290
importing vcd stimulus file CR-404
mapping signals GR-295
modifying existing waves CR-405
saving stimulus files GR-294
saving waves CR-403
simulating GR-293
toolbar buttons GR-222
waveform logfile
log command CR-191
overview UM-225
see also WLF files
waveform popup UM-280, GR-256
waveforms UM-225
optimize viewing of UM-535
optimizing viewing of CR-382
saving and viewing CR-191, UM-226
viewing GR-211
WaveRestoreCursors command CR-423
WaveRestoreZoom command CR-423
WaveSignalNameWidth .ini file variable UM-534
weighting, coverage directives UM-387
Welcome dialog, turning on/off UM-520
when command CR-407
when statement
time-based breakpoints CR-411
where command CR-412
wildcard characters
for pattern matching in simulator commands CR-17
Windows
Main window
text editing UM-605
Source window

text editing UM-605
windows
Active Processes pane GR-108
buttons, adding to GR-106
code coverage statistics UM-340
Dataflow window UM-300, GR-128
toolbar GR-132
zooming UM-305
Functional coverage browser GR-143
List window UM-243, GR-153
display properties of UM-259
formatting HDL items UM-259
output file CR-424
saving data to a file UM-263
saving the format of CR-422
setting triggers UM-265, UM-266, GR-163
Locals window GR-166
Main window GR-14
adding user-defined buttons CR-45
status bar GR-19
time and delta display GR-19
toolbar GR-34
Memory window GR-169
monitor GR-208
Objects window GR-184
opening
from command line CR-332
with the GUI GR-23
Process window GR-143
specifying next process to be executed GR-143
viewing processing in the region GR-143
Signals window
VHDL and Verilog items viewed in GR-184
Source window GR-199
viewing HDL source code GR-199
Variables window
VHDL and Verilog items viewed in GR-166
Wave window UM-240, GR-211
adding HDL items to UM-244
cursor measurements UM-245
display properties UM-255
display range (zoom), changing UM-249
format file, saving UM-261
path elements, changing CR-125, UM-534
time cursors UM-245
zooming UM-249
WLF file
collapsing deltas CR-382
collapsing time steps CR-382
WLF files
collapsing events UM-232
converting to VCD CR-415
creating from VCD CR-310
filtering, combining CR-416
limiting size CR-382
log command CR-191
optimizing waveform viewing CR-382, UM-535
overview UM-226
repairing CR-420
saving CR-143, CR-144, UM-227
saving at intervals UM-231
specifying name CR-382
wlf2log command CR-413
wlf2vcd command CR-415
WLFCollapseMode .ini file variable UM-534
WLFFilename UM-535
wlfman command CR-416
wlfrecover command CR-420
work library UM-58
creating UM-60
workspace GR-15
code coverage GR-116
Files tab GR-116
write cell_report command CR-421
write format command CR-422
write list command CR-424
write preferences command CR-425
WRITE procedure, problems with UM-90
write report command CR-426
write timing command CR-427
write transcript command CR-428
write tssi command CR-429
write wave command CR-431

X

X
tracing unknowns UM-307
.Xdefaults file, controlling fonts GR-13
X propagation
disabling for entire design CR-378
disabling X generation on specific instances CR-275
xml format
coverage reports UM-351
X-session
controlling fonts GR-13

Y

-y CR-366

Z

zero delay elements UM-80
zero delay mode UM-145
zero-delay loop, infinite UM-81
zero-delay oscillation UM-81
zero-delay race condition UM-132
zoom
    Dataflow window UM-305
    from Wave toolbar buttons UM-249
    saving range with bookmarks UM-250
    with the mouse UM-249
zooming window panes GR-260