ModelSim®

Advanced Verification and Debugging

Xilinx Tutorial

Version 6.0a

Published: September 24, 2004

Creating the working design library

Before you can simulate a design, you must first create a library and compile the source code into that library.

1 Create a new directory and copy the tutorial files into it.

Start by creating a new directory for this exercise (in case other users will be working with these lessons).

Verilog: Copy *counter.v* and *tcounter.v* files from */<install_dir>/examples* to the new directory.

VHDL: Copy *counter.vhd* and *tcounter.vhd* files from */<install_dir>/examples* to the new directory.

2 Start ModelSim if necessary.

a Type **vsim** at a UNIX shell prompt or use the ModelSim icon in Windows.

Upon opening ModelSim for the first time, you will see the Welcome to ModelSim dialog (Figure 1). Click **Close**.

b Select **File > Change Directory** and change to the directory you created in step 1.

3 Create the working library.

a Select **File > New > Library**.

This opens a dialog where you specify physical and logical names for the library (Figure 2). You can create a new library or map to an existing library. We'll be doing the former.

b Type **work** in the Library Name field if it isn't entered automatically.

Figure 1: The Welcome to ModelSim dialog

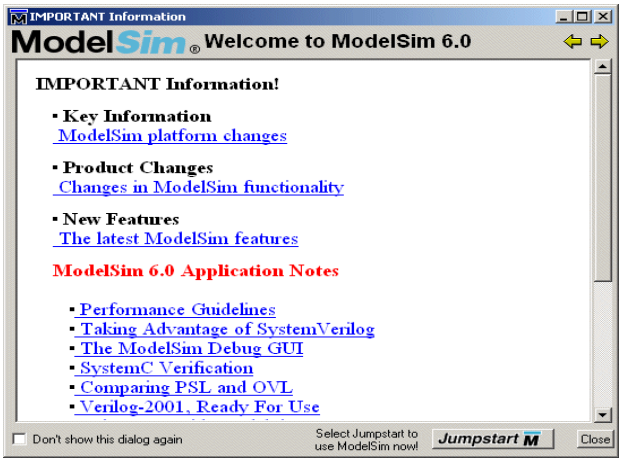
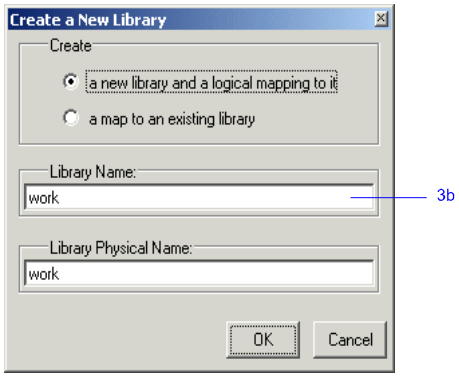


Figure 2: The Create a New Library dialog



c Click **OK**.

ModelSim creates a directory called *work* and writes a specially-formatted file named *_info* into that directory. The *_info* file must remain in the directory to distinguish it as a ModelSim library. Do not edit the folder contents from your operating system; all changes should be made from within ModelSim.

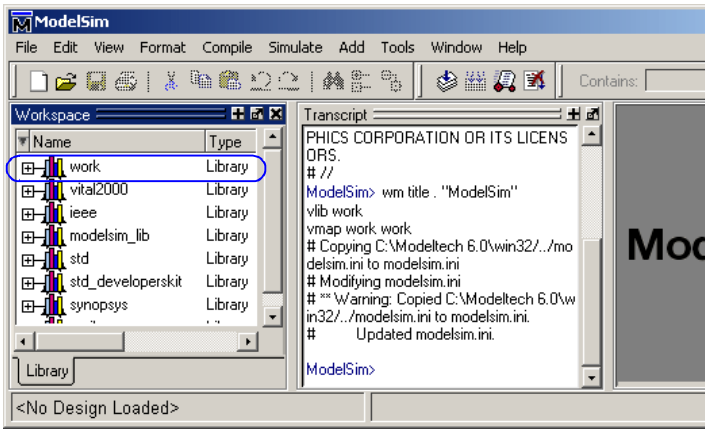
ModelSim also adds the library to the list in the Workspace (Figure 3) and records the library mapping for future reference in the ModelSim initialization file (*modelsim.ini*).

When you pressed OK in step c above, three lines were printed to the Main window Transcript pane:

```
vlib work
vmap work work
# Modifying modelsim.ini
```

The first two lines are the command-line equivalent of the menu commands you invoked. Most menu driven functions will echo their command-line equivalents in this fashion. The third line notifies you that the mapping has been recorded in the ModelSim initialization file.

Figure 3: The newly created work library



Compiling the design

With the working library created, you are ready to compile your source files.

You can compile by using the menus and dialogs of the graphic interface, as in the Verilog example below, or by entering a command at the ModelSim> prompt as in the VHDL example below.

1 **Verilog:** Compile *counter.v* and *tcounter.v*.

a Select **Compile > Compile**.

This opens the Compile Source Files dialog (Figure 4).

If the Compile menu option is not available, you probably have a project open. If so, close the project by selecting **File > Close** when the Workspace pane is selected.

b Select *counter.v*, hold the <Ctrl> key down, and then select *tcounter.v*.

c With the two files selected, click **Compile**.

The files are compiled into the *work* library.

d Click **Done**.

VHDL: Compile *counter.vhd* and *tcounter.vhd*.

a Type **vcom counter.vhd tcounter.vhd** at the ModelSim> prompt and press <Enter> on your keyboard.

2 View the compiled design units.

a On the Library tab, click the '+' icon next to the *work* library and you will see two design units (Figure 5). You can also see their types (Modules, Entities, etc.) and the path to the underlying source files if you scroll to the right.

Figure 4: The Compile HDL Source Files dialog

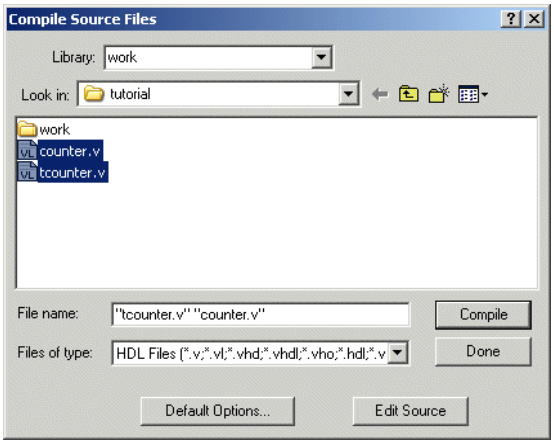
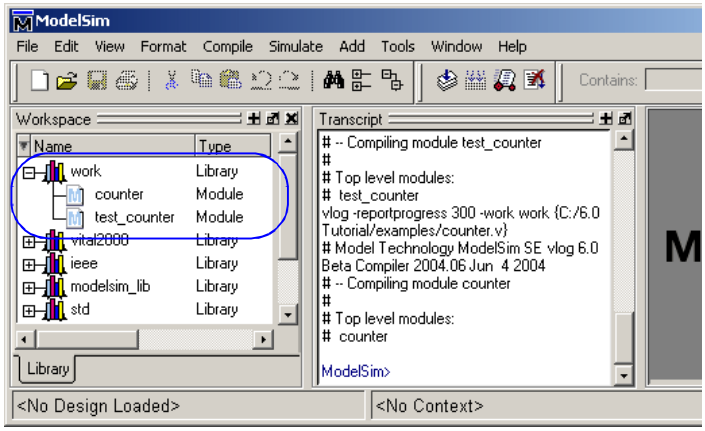


Figure 5: Verilog modules compiled into the work library



Loading the design into the simulator

1 Load the *test_counter* module into the simulator.

a Double-click *test_counter* in the Main window Workspace to load the design.

You can also load the design by selecting **Simulate > Start Simulation** in the menu bar. This opens the Start Simulation dialog. With the Design tab selected, click the '+' sign next to the work library to see the *counter* and *test_counter* modules. Select the *test_counter* module and click OK (Figure 6).

When the design is loaded, you will see a new tab named *sim* that displays the hierarchical structure of the design (Figure 7). You can navigate within the hierarchy by clicking on any line with a '+' (expand) or '-' (contract) icon. You will also see a tab named *Files* that displays all files included in the design.

Figure 6: Loading the design with the Start Simulation dialog

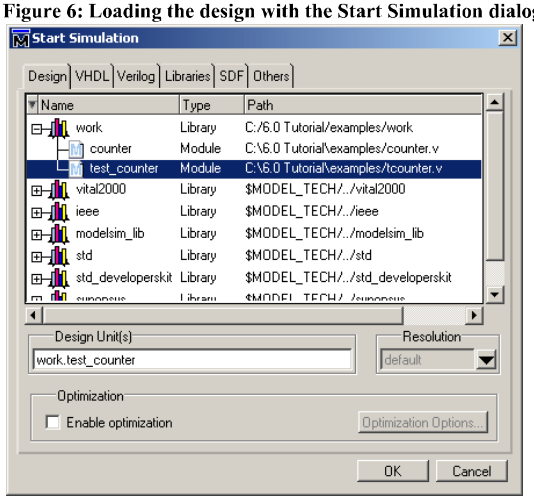
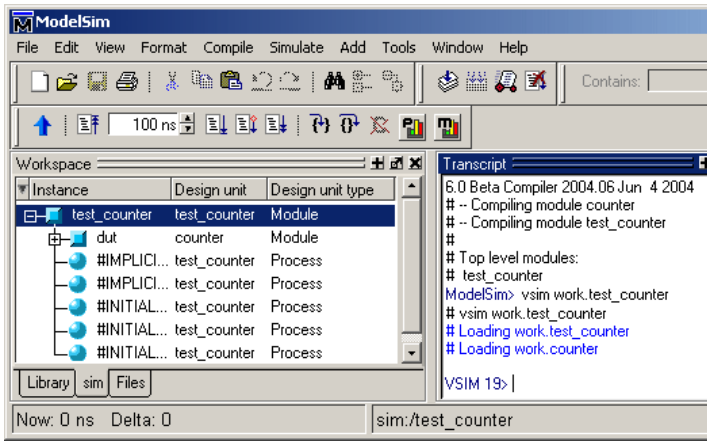


Figure 7: Workspace tab showing a Verilog design



Running the simulation

Now you will run the simulation.

1 Set the graphic user interface to view all debugging windows.

a Select **View > Debug Windows > All Windows**.

This opens all ModelSim windows, giving you different views of your design data and a variety of debugging tools. Most windows will open as panes within the Main window. The Dataflow, List, and Wave windows will open as separate windows. You may need to move or resize the windows to your liking. Panes within the Main window can be undocked to stand alone.

2 Add signals to the Wave window.

a In the Workspace pane, select the **sim** tab.

b Right-click *test_counter* to open a popup context menu.

c Select **Add > Add to Wave** (Figure 8).

Three signals are added to the Wave window.

3 Run the simulation.

a Click the Run icon in the Main or Wave window toolbar.

The simulation runs for 100 ns (the default simulation length) and waves are drawn in the Wave window.

b Type **run 500** at the VSIM> prompt in the Main window.

The simulation advances another 500 ns for a total of 600 ns (Figure 9).

Figure 8: Adding signals to the Wave window

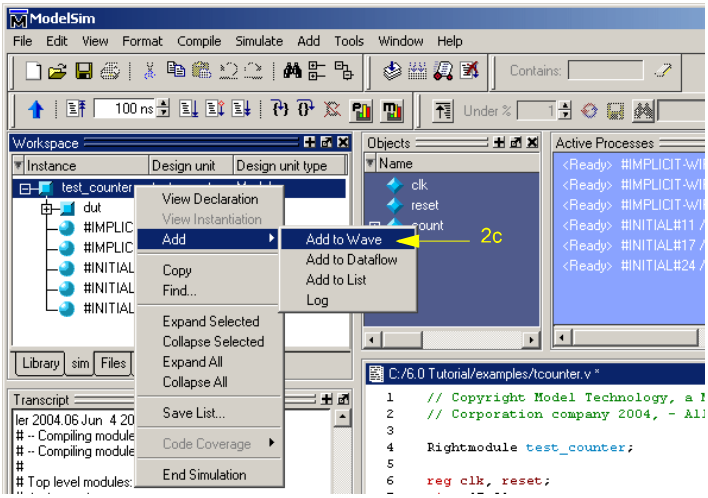
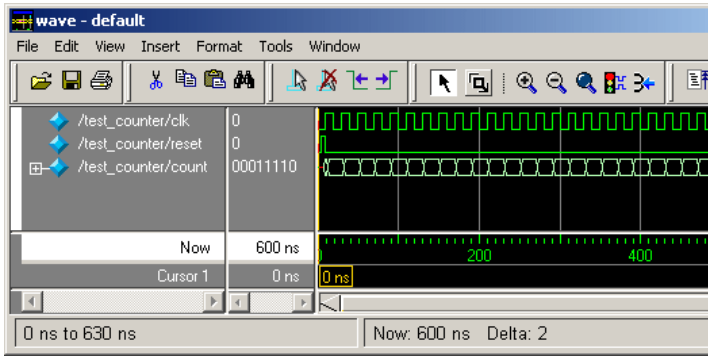


Figure 9: Waves being drawn in the Wave window



c Click the Run -All icon on the Main or Wave window toolbar.

The simulation continues running until you execute a break command or it hits a statement in your code (e.g., a Verilog $stop statement) that halts the simulation.

d Click the Break icon.

The simulation stops running.

Setting breakpoints and stepping in the Source window

Next you will take a brief look at one interactive debugging feature of the ModelSim environment. You will set a breakpoint in the Source window, run the simulation, and then step through the design under test. Breakpoints can be set only on lines with red line numbers.

1 Open *counter.v* in the Source window.

a Select the **Files** tab in the Main window Workspace.

b Double-click *counter.v* to add it to the Source window.

2 Set a breakpoint on line 31 of *counter.v* (if you are simulating the VHDL files, use line 30 instead).

a Scroll to line 31 and click on the line number.

A red ball appears next to the line (Figure 10) indicating that a breakpoint has been set.

3 Disable, enable, and delete the breakpoint.

a Click the red ball to disable the breakpoint. It will become a black circle.

b Click the black circle to re-enable the breakpoint. It will become a red ball.

c Click the red ball with your right mouse button and select **Remove Breakpoint 31**.

d Click on line number 31 again to re-create the breakpoint.

4 Restart the simulation.

a Click the Restart icon to reload the design elements and reset the simulation time to zero.

The Restart dialog that appears gives you options on what to retain during the restart (Figure 11).

b Click **Restart** in the Restart dialog.

Figure 10: A breakpoint in the Source window

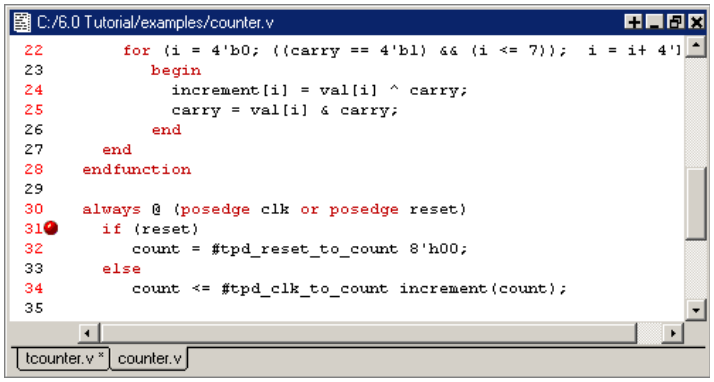
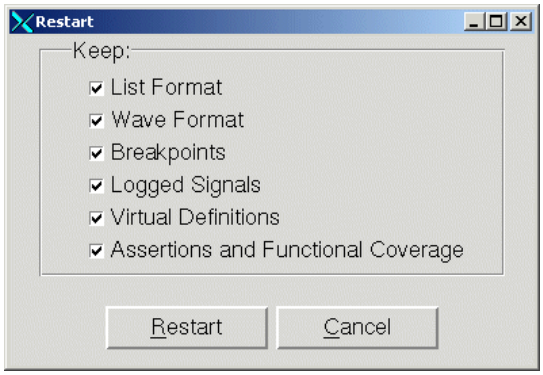


Figure 11: The Restart dialog



c Click the Run -All icon.

The simulation runs until the breakpoint is hit. When the simulation hits the breakpoint, it stops running, highlights the line with a blue arrow in the Source view (Figure 12), and issues a Break message in the Transcript pane.

When a breakpoint is reached, typically you want to know one or more signal values. You have several options for checking values:

- look at the values shown in the Objects window (Figure 13).
- set your mouse pointer over the *count* variable in the Source window, and a "balloon" will pop up with the value (Figure 12)
- highlight the *count* variable in the Source window, right-click it, and select Examine from the pop-up menu
- use the examine command to output the value to the Main window Transcript (i.e., `examine count`)

5 Try out the step commands.

a Click the Step icon on the Main window toolbar.

This single-steps the debugger.

Experiment on your own. Set and clear breakpoints and use the Step, Step Over, and Continue Run commands until you feel comfortable with their operation.

Figure 12: Resting the mouse pointer on a variable in the Source view

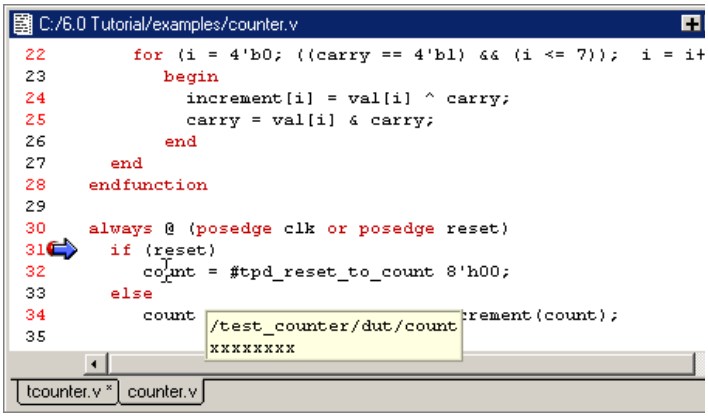
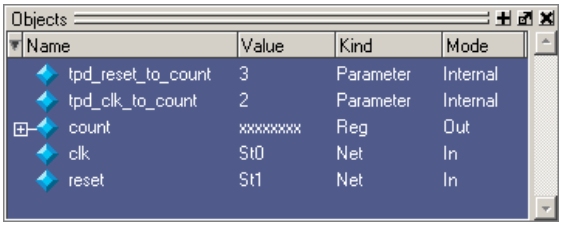


Figure 13: Values shown in the Objects window



Creating a new project

1 If you just finished the previous lesson, ModelSim should already be running. If not, start ModelSim.

   a Type **vsim** at a UNIX shell prompt or use the ModelSim icon in Windows.

2 Create a new project.

   a Select **Create a Project** from the Welcome dialog *or* **File > New > Project** (Main window) from the menu bar.

   This opens a dialog where you enter a Project Name, Project Location (i.e., directory), and Default Library Name (Figure 14). The default library is where compiled design units will reside.

   b Type **test** in the Project Name field.

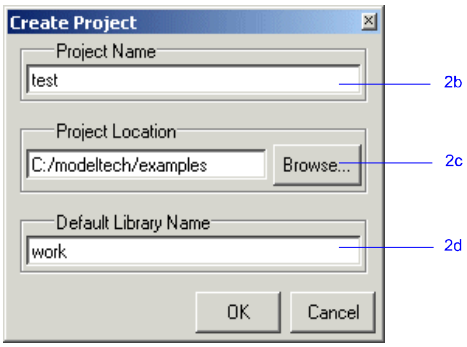
   c Click **Browse** to select a directory where the project file will be stored.

   d Leave the Default Library Name set to *work*.

   e Click **OK**.

   If you see the Select Initial Ini dialog, asking which *modelsim.ini* file you would like the project to be created from, select the **Use Default Ini** button.

Figure 14: The Create Project dialog



Adding objects to the project

Once you click OK to accept the new project settings, you will see a blank Project tab in the workspace area of the Main window and the Add items to the Project dialog will appear (Figure 15). From this dialog you can create a new design file, add an existing file, add a folder for organization purposes, or create a simulation configuration (discussed below).

1 Add two existing files.

a Click **Add Existing File**.

This opens the Add file to Project dialog (Figure 16). This dialog lets you browse to find files, specify the file type, specify which folder to add the file to, and identify whether to leave the file in its current location or to copy it to the project directory.

b Click **Browse**.

c Open the *examples* directory in your ModelSim installation tree.

d **Verilog:** Select *counter.v*, hold the <Ctrl> key down, and then select *tcounter.v*.
VHDL: Select *counter.vhd*, hold the <Ctrl> key down, and then select *tcounter.vhd*.

e Click **Open** and then **OK**.

f Click **Close** to dismiss the Add items to the Project dialog.

Figure 15: Adding new items to a project

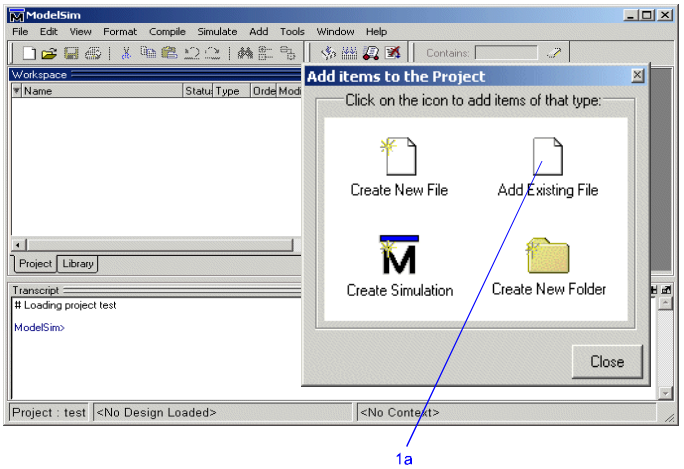
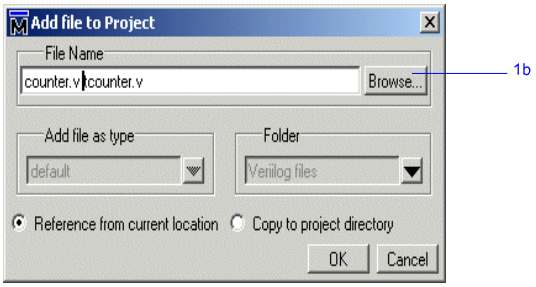


Figure 16: The Add file to Project dialog



You should now see two files listed in the Project tab of the Workspace pane (Figure 17).

Question mark icons (?) in the Status column mean the file hasn't been compiled or the source file has changed since the last successful compile. The other columns identify file type (e.g., Verilog or VHDL), compilation order, and modified date.

Changing compile order (VHDL)

Compilation order is important in VHDL designs. Follow these steps to change compilation order within a project.

1 Change the compile order.

a Select **Compile > Compile Order**.

This opens the Compile Order dialog box (Figure 18).

b Click the **Auto Generate** button.

ModelSim "determines" the compile order by making multiple passes over the files. It starts compiling from the top; if a file fails to compile due to dependencies, it moves that file to the bottom and then recompiles it after compiling the rest of the files. It continues in this manner until all files compile successfully or until a file(s) can't be compiled for reasons other than dependency.

Alternatively, you can select a file and use the Move Up and Move Down buttons to put the files in the correct order.

c Click **OK** to close the Compile Order dialog.

Figure 17: Newly added project files display a '?' for status

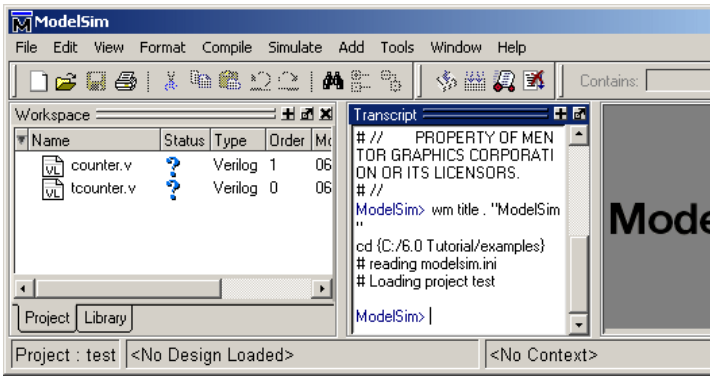
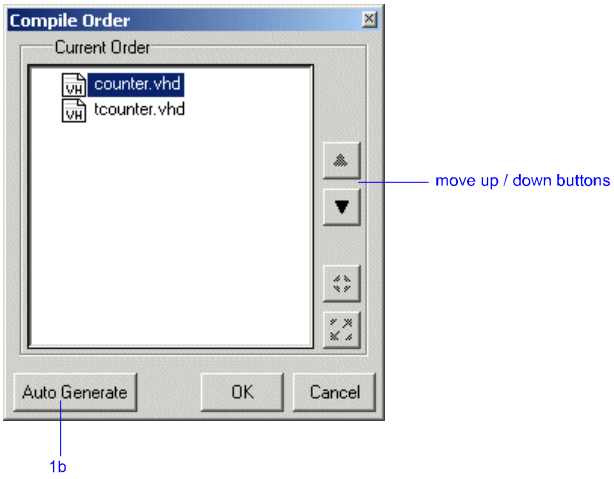


Figure 18: The Compile Order dialog box



Compiling and loading a design

1 Compile the files.

   a Right-click anywhere in the Project tab and select **Compile > Compile All** from the pop-up menu.

   ModelSim compiles both files and changes the symbol in the Status column to a check mark. A check mark means the compile succeeded. If the compile had failed, the symbol would be a red 'X', and you would see an error message in the Transcript pane.

2 View the design units.

   a Click the **Library** tab in the workspace.

   b Click the "+" icon next to the *work* library.

   You should see two compiled design units, their types (modules in this case), and the path to the underlying source files (Figure 19).

3 Load the *test_counter* design unit.

   a Double-click the *test_counter* design unit.

   You should see a new tab named *sim* that displays the structure of the *test_counter* design unit (Figure 20). A fourth tab named *Files* contains information about the underlying source files.

   At this point you would generally run the simulation and analyze or debug your design like you did in the previous lesson. For now, you'll continue working with the project. However, first you need to end the simulation that started when you loaded *test_counter*.

4 End the simulation.

   a Select **Simulate > End Simulation**.

   b Click **Yes**.

Figure 19: The Library tab with an expanded library

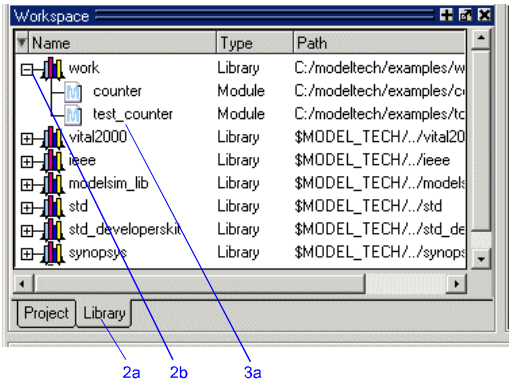
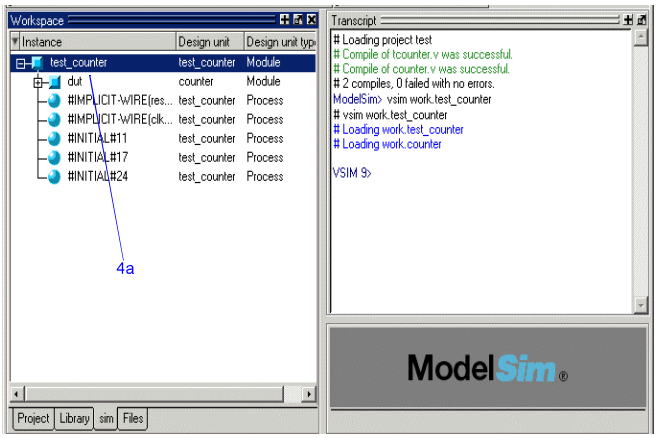


Figure 20: The structure tab for the *counter* design unit



Organizing projects with folders

If you have a lot of files to add to a project, you may want to organize them in folders. You can create folders either before or after adding your files. If you create a folder before adding files, you can specify in which folder you want a file placed at the time you add the file (see Folder field in Figure 16). If you create a folder after adding files, you edit the file properties to move it to that folder.

Adding folders

As shown previously in Figure 15, the Add items to the Project dialog has an option for adding folders. If you have already closed that dialog, you can use a menu command to add a folder.

1 Add a new folder.

a Select **File > Add to Project > Folder**.

b Type **Design Files** in the **Folder Name** field (Figure 21).

c Click **OK**.

You'll now see a folder in the Project tab (Figure 22).

2 Add a sub-folder.

a Right-click anywhere in the Project tab and select **Add to Project > Folder**.

b Type **HDL** in the **Folder Name** field (Figure 23).

c Click the **Folder Location** drop-down arrow and select *Design Files*.

d Click OK.

Figure 21: Adding a new folder to the project

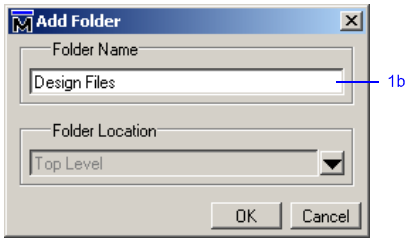


Figure 22: A folder in a project

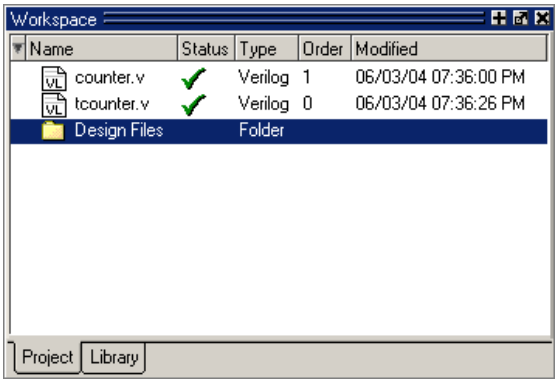
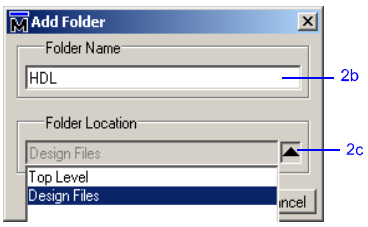


Figure 23: Creating a subfolder



You'll now see a '+' icon next to the *Design Files* folder in the Project tab (Figure 24).

e Click the '+' icon to see the *HDL* sub-folder.

Moving files to folders

Now that you have folders, you can move the files into them. If you are running on a Windows platform, you can simply drag-and-drop the files into the folder. On Unix platforms, you either have to place the files in a folder when you add the files to the project, or you have to move them using the properties dialog.

1 Move *tcounter.v* and *counter.v* to the *HDL* folder.

a Select *counter.v*, hold the <Ctrl> key down, and then select *tcounter.v*.

b Right-click either file and select **Properties**.

This opens the Project Compiler Settings dialog (Figure 25), which lets you set a variety of options on your design files.

c Click the **Place In Folder** drop-down arrow and select *HDL*.

d Click OK.

The two files are moved into the HDL folder. Click the '+' icons on the folders to see the files.

The files are now marked with a '?' icon. Because you moved the files, the project no longer knows if the previous compilation is still valid.

Figure 24: A folder with a sub-folder

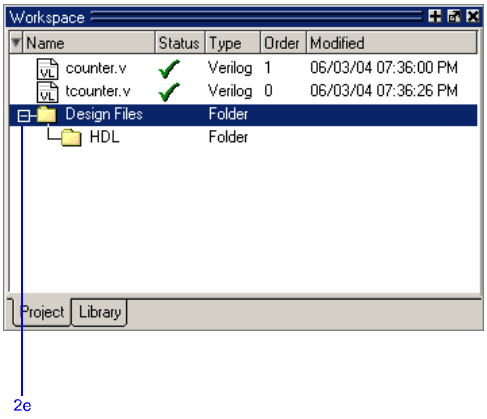
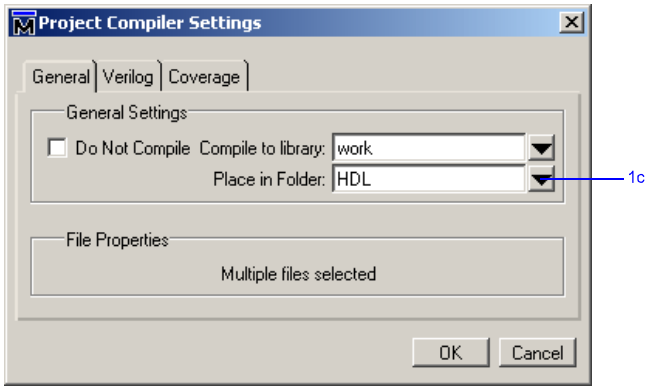


Figure 25: Changing file location via the project settings dialog



Simulation Configurations

A Simulation Configuration associates a design unit(s) and its simulation options. For example, say every time you load *tcounter.v* you want to set the simulator resolution to picoseconds (ps) and enable event order hazard checking. Ordinarily you would have to specify those options each time you load the design. With a Simulation Configuration, you specify options for a design and then save a "configuration" that associates the design and its options. The configuration is then listed in the Project tab and you can double-click it to load *counter.v* along with its options.

1 Create a new Simulation Configuration.

a Select **File > Add to Project > Simulation Configuration**.

This opens the Simulate dialog (Figure 26). The tabs in this dialog present a myriad of simulation options. You may want to explore the tabs to see what's available. You can consult the ModelSim User's Manual to get a description of each option.

b Type **counter** in the **Simulation Configuration Name** field.

c Select *HDL* from the **Place in Folder** drop-down.

d Click the '+' icon next to the *work* library and select *test_counter*.

e Click the **Resolution** drop-down and select *ps*.

f For Verilog, click the Verilog tab and check **Enable Hazard Checking**.

g Click **OK**.

The Project tab now shows a Simulation Configuration named *counter* (Figure 27).

2 Load the Simulation Configuration.

a Double-click the *counter* Simulation Configuration in the Project tab.

In the Transcript pane of the Main window, the **vsim** (the ModelSim simulator) invocation shows the **-hazards** and **-t ps** switches (Figure 28). These are the command-line equivalents of the options you specified in the Simulate dialog.

Figure 26: The Simulation Configuration dialog

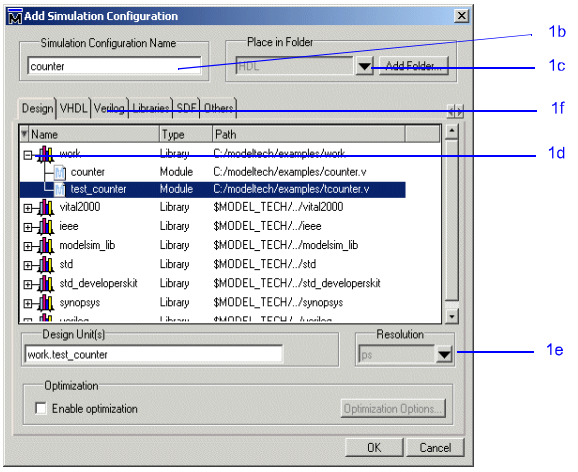
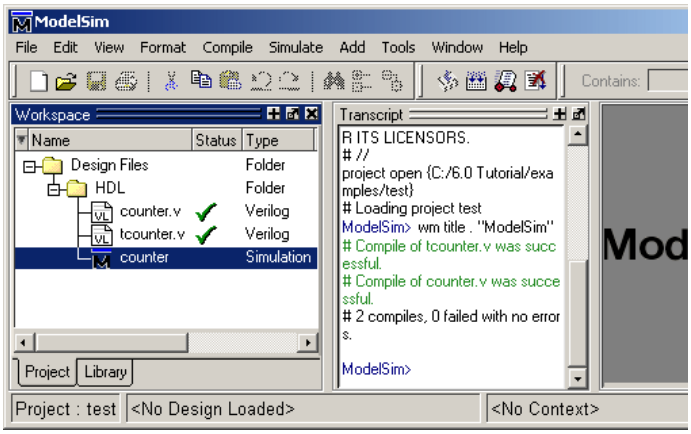


Figure 27: A Simulation Configuration in the Project tab



Lesson wrap-up

This concludes this lesson. Before continuing you need to end the current simulation and close the current project.

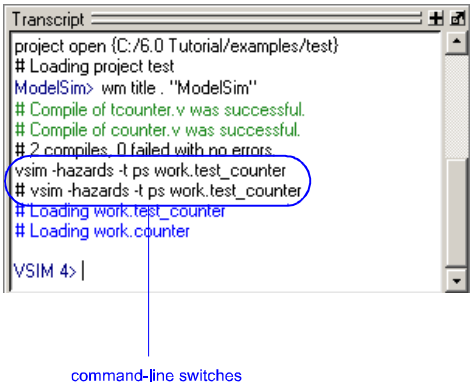
1 Select **Simulate > End Simulation**. Click Yes.

2 Select the Project tab in the Main window Workspace.

3 Right-click the *test* project to open a context popup menu and select **Close Project**.

If you do not close the project, it will open automatically the next time you start ModelSim.

Figure 28: Transcript shows options used for Simulation Configuration



Lesson 4 - Working with multiple libraries

Topics

The following topics are covered in this lesson:

Introduction

In this lesson you will practice working with multiple libraries. As discussed in *Lesson 1 - ModelSim conceptual overview*, you might have multiple libraries to organize your design, to access IP from a third-party source, or to share common parts between simulations.

You will start the lesson by creating a resource library that contains the *counter* design unit. Next, you will create a project and compile the testbench into it. Finally, you will link to the library containing the counter and then run the simulation.

Design files for this lesson

The sample design for this lesson is a simple 8-bit, binary up-counter with an associated testbench. The pathnames are as follows:

Verilog – *<install_dir>/modeltech/examples/counter.v* and *tcounter.v*

VHDL – *<install_dir>/modeltech/examples/counter.vhd* and *tcounter.vhd*

This lesson uses the Verilog files *tcounter.v* and *counter.v* in the examples. If you have a VHDL license, use *tcounter.vhd* and *counter.vhd* instead.

Related reading

ModelSim User's Manual, *3 - Design libraries* (UM-45)

Creating the resource library

1 Create a directory for the resource library.

Create a new directory called *resource_library*. Copy *counter.v* from *<install_dir>/modeltech/examples* to the new directory.

2 Create a directory for the testbench.

Create a new directory called *testbench* that will hold the testbench and project files. Copy *tcounter.v* from *<install_dir>/modeltech/examples* to the new directory.

You are creating two directories in this lesson to mimic the situation where you receive a resource library from a third-party. As noted earlier, we will link to the resource library in the first directory later in the lesson.

3 Start ModelSim and change to the exercise directory.

If you just finished the previous lesson, ModelSim should already be running. If not, start ModelSim.

a Type **vsim** at a UNIX shell prompt or use the ModelSim icon in Windows.

If the Welcome to ModelSim dialog appears, click **Close**.

b Select **File > Change Directory** and change to the *resource_library* directory you created in step 1.

4 Create the resource library.

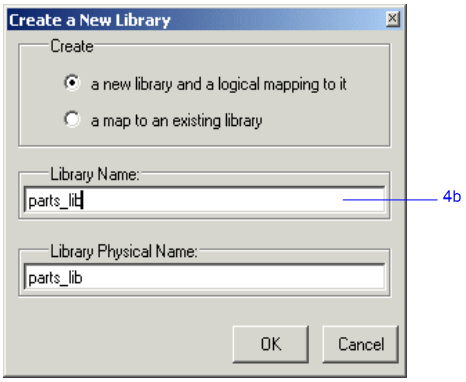
a Select **File > New > Library**.

b Type **parts_lib** in the Library Name field (Figure 29).

The Library Physical Name field is filled out automatically.

Once you click OK, ModelSim creates a directory for the library, lists it in the Library tab of the Workspace, and modifies the *modelsim.ini* file to record this new library for the future.

Figure 29: Creating the new resource library



5 Compile the counter into the resource library.

a Click the Compile icon on the Main window toolbar.

b Select the *parts_lib* library from the Library list (Figure 30).

c Double-click *counter.v* to compile it.

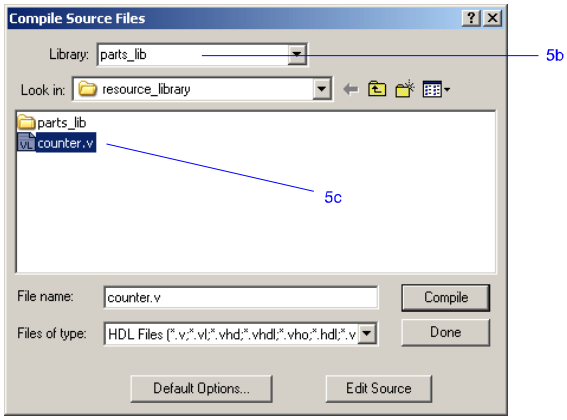
d Click Done.

You now have a resource library containing a compiled version of the *counter* design unit.

6 Change to the *testbench* directory.

a Select **File > Change Directory** and change to the *testbench* directory you created in step 2.

Figure 30: Compiling into the resource library



Creating the project

Now you will create a project that contains *tcounter.v*, the counter's testbench.

1 Create the project.
   a Select **File > New > Project**.
   b Type **counter** in the Project Name field.
   c Click **OK**.
   d If a dialog appears asking about which *modelsim.ini* file to use, click **Use Default Ini**.

2 Add the testbench to the project.
   a Click **Add Existing File** in the Add items to the Project dialog.
   b Click the Browse button and select *tcounter.v*.
   c Click Open and then OK.
   d Click Close to dismiss the Add items to the Project dialog.
   The *tcounter.v* file is listed in the Project tab of the Main window.

3 Compile the testbench.
   a Right-click *tcounter.v* and select **Compile > Compile Selected**.

Linking to the resource library

To wrap up this part of the lesson, you will link to the *parts_lib* library you created earlier. But first, try simulating the testbench without the link and see what happens.

ModelSim responds differently for Verilog and VHDL in this situation.

Verilog

1 Simulate a Verilog design with a missing resource library.

a In the Library tab, click the '+' icon next to the *work* library and double-click *test_counter*.

The Main window Transcript reports an error (Figure 31). When you see a message that contains text like "Error: (vsim-3033)", you can view more detail by using the **verror** command.

b Type **verror 3033** at the ModelSim> prompt.

The expanded error message tells you that a design unit could not be found for instantiation. It also tells you that the original error message should list which libraries ModelSim searched. In this case, the original message says ModelSim searched only *work*.

VHDL

1 Simulate a VHDL design with a missing resource library.

a In the Library tab, click the '+' icon next to the *work* library and double-click *test_counter*.

The Main window Transcript reports a warning(Figure 32). When you see a message that contains text like "Warning: (vsim-3473)", you can view more detail by using the **verror** command.

b Type **verror 3473** at the ModelSim> prompt.

The expanded error message tells you that a component ('dut' in this case) has not been explicitly bound and no default binding can be found.

c Type **quit -sim** to quit the simulation.

Figure 31: Verilog simulation error reported in the Main window

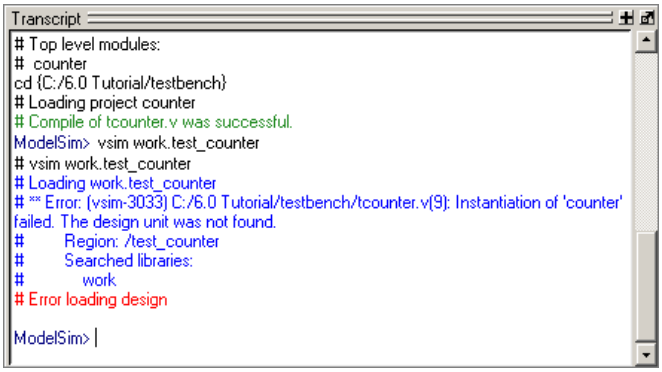
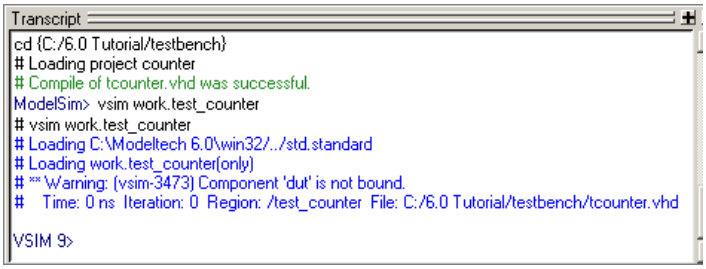


Figure 32: VHDL simulation warning reported in Main window



The process for linking to a resource library differs between Verilog and VHDL. If you are using Verilog, follow the steps in "Linking in Verilog" (T-45). If you are using VHDL, follow the steps in "Linking in VHDL" (T-46) one page later.

Linking in Verilog

Linking in Verilog requires that you specify a "search library" when you invoke the simulator.

1 Specify a search library during simulation.

a Click the Simulate icon on the Main window toolbar.

b Click the '+' icon next to the *work* library and select *test_counter*.

c Click the Libraries tab.

d Click the Add button next to the Search Libraries field and browse to *parts_lib* in the first directory you created earlier in the lesson.

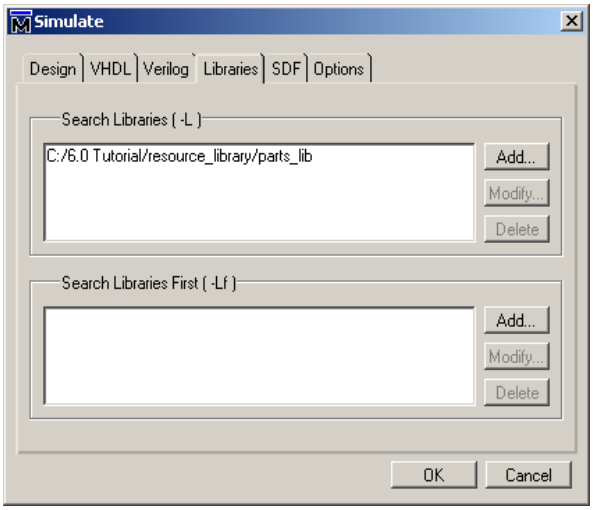
e Click OK.

The dialog should have *parts_lib* listed in the Search Libraries field (Figure 33).

f Click OK.

The design loads without errors.

Figure 33: Specifying a search library in the Simulate dialog



Linking in VHDL

To link to a resource library in VHDL, you have to create a logical mapping to the physical library and then add LIBRARY and USE statements to the source file.

1 Create a logical mapping to *parts_lib*.

a Select **File > New > Library**.

b In the Create a New Library dialog, select **a map to an existing library**.

c Type **parts_lib** in the Library Name field.

d Click Browse to open the Select Library dialog and browse to *parts_lib* in the *resource_library* directory you created earlier in the lesson. Click OK to select the library and close the Select Library dialog.

e The Create a New Library dialog should look similar to the one shown in Figure 34. Click OK to close the dialog.

2 Add LIBRARY and USE statements to *tcounter.vhd*.

a In the Library tab of the Main window, click the '+' icon next to the *work* library.

b Right-click *test_counter* in the work library and select **Edit**.

This opens the file in the Source window.

c Add these two lines to the top of the file:

```
LIBRARY parts_lib;
USE parts_lib.ALL;
```

The testbench source code should now look similar to that shown in Figure 33.

d Select **File > Save**.

3 Recompile and simulate.

a In the Project tab of the Main window, right-click *tcounter. vhd* and select **Compile > Compile Selected**.

b In the Library tab, double-click *test_counter* to load the design.

The design loads without errors.

Figure 34: Mapping to the parts_lib library

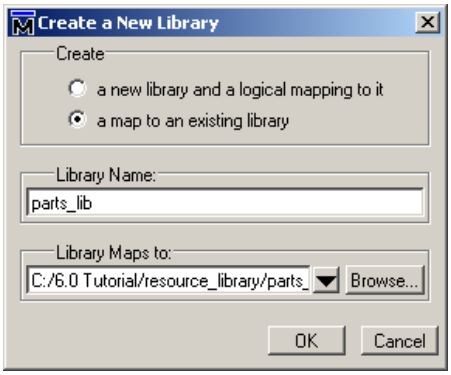


Figure 35: Adding LIBRARY and USE statements to the testbench