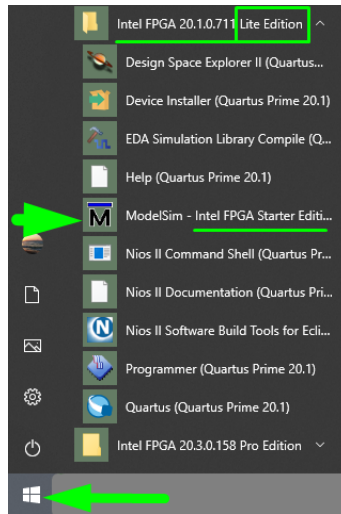


ModelSim Intel FPGA Started Edition Introduction

LAB1_1: BASIC SIMULATION FLOW.....	4
DESIGN FILES.....	5
START MODELSIM	5
CREATE THE WORKING LIBRARY	6
COMPILE THE DESIGN UNITS.....	7
LOAD THE DESIGN	8
RUN THE SIMULATION	10
SET BREAKPOINTS AND STEP THROUGH THE SOURCE	11
ANALYZING WAVEFORMS.....	14
ADDING OBJECTS TO THE WAVE WINDOW	14
ZOOMING THE WAVEFORM DISPLAY	15
RADIX AND FORMAT IN THE WAVE WINDOW	16
USING CURSORS IN THE WAVE WINDOW	17
SAVING AND REUSING THE WINDOW FORMAT	20
CONCLUDING THE LAB	21
LAB1_2: PROJECTS FLOW.....	22
DESIGN FILES.....	23
START MODELSIM	23
CREATE A NEW PROJECT.....	23
ADD OBJECTS TO THE PROJECT.....	24
COMPILE THE DESIGN.....	25
LOAD THE DESIGN	26
ADDING FOLDERS	27
MOVING FILES TO FOLDERS.....	28
USING SIMULATION CONFIGURATIONS	29
CONCLUDING THE LAB.....	31
LAB1_3: WORKING WITH MULTIPLE LIBRARIES	32
DESIGN FILES.....	33
CREATING THE RESOURCE LIBRARY.....	34
CREATING THE PROJECT	35
LOADING WITHOUT LINKING LIBRARIES	37
LINKING TO THE RESOURCE LIBRARY	38
CONCLUDING THE LAB.....	39
LAB1_4: AUTOMATING SIMULATION	40
DESIGN FILES.....	40
CREATING A SIMPLE DO FILE	40
RUNNING IN COMMAND-LINE MODE.....	41
RUNNING TCL SCRIPT	44
CONCLUDING THE LAB	48
LAB1_5: VIEWING AND INITIALIZING MEMORIES	49
DESIGN FILES.....	49
COMPILE AND LOAD THE DESIGN	49
VIEW A MEMORY AND ITS CONTENTS	49
NAVIGATE WITHIN THE MEMORY	51
EXPORT MEMORY DATA TO A FILE	53
INITIALIZE A MEMORY	56
INTERACTIVE DEBUGGING COMMANDS.....	57
CONCLUDING THE LAB	60

Preliminary requirements

Before to start the Lab be sure, you have ModelSim Intel FPGA Started Edition installed.



Overview

ModelSim is a verification and simulation tool for VHDL, Verilog HDL, SystemVerilog.

This lab provides a brief conceptual overview of the ModelSim simulation environment. It is divided into five topics.

The working folder for the lab is **C:\Intel_trn\ModelSim**. The folder contains five subfolders, i.e. there is an individual subfolder for each topic. Each subfolder contains source files for the particular lab.

- Lab1_1: Basic simulation flow.
- Lab1_2: Project flow.
- Lab1_3: Working with Multiple Libraries.
- Lab1_4: Debugging tools.
- Lab1_5: Viewing and Initializing Memories.

Lab1_1: Basic Simulation Flow

In this Lab1_1 you will guide you through the basic simulation flow.

You will learn to:

- Create the Working Design Library
- Compile the Design Units
- Load the Design
- Run the Simulation

The following diagram shows the basic simulation flow for simulating a design in ModelSim.

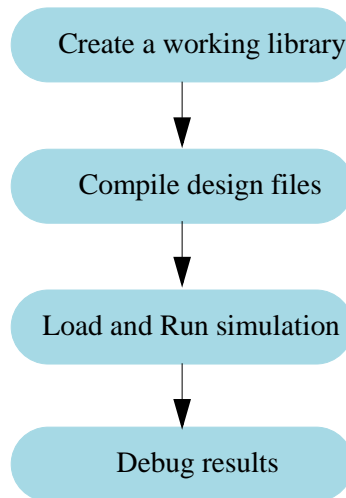


Figure 1

- Creating the Working Library

In ModelSim, all designs are compiled into a library. You typically start a new simulation in ModelSim by creating a working library called "work," which is the default library name used by the compiler as the default destination for compiled design units.

- Compiling Your Design

After creating the working library, you compile your design units into it. The ModelSim library format is compatible across all supported platforms. You can simulate your design on any platform without having to recompile your design.

- Loading the Simulator with Your Design and Running the Simulation

With the design compiled, you load the simulator with your design by invoking the simulator on a top-level module (Verilog) or a configuration or entity/architecture pair (VHDL).

Assuming the design loads successfully, the simulation time is set to zero, and you enter a run command to begin simulation.

- Debugging Your Results

If you don't get the results you expect, you can use ModelSim's robust debugging environment to track down the cause of the problem.

The sample design for this lesson is a simple 8-bit, binary up-counter with an associated test bench.

This lesson uses the Verilog files *counter.v* and *tcounter.v*. These files are located in the **C:\Intel_trn\ModelSim\Lab1_1** folder. Which is a working folder for the Lab1_1.

Start ModelSim

Procedure to start ModelSim :

1. Push Start button in Windows.
2. Click on ModelSim Intel FPGA Started Edition icon.

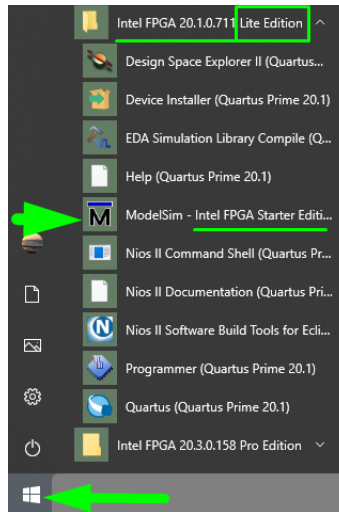


Figure 2

3. Upon opening ModelSim for the first time, you will see the Welcome to ModelSim dialog box.

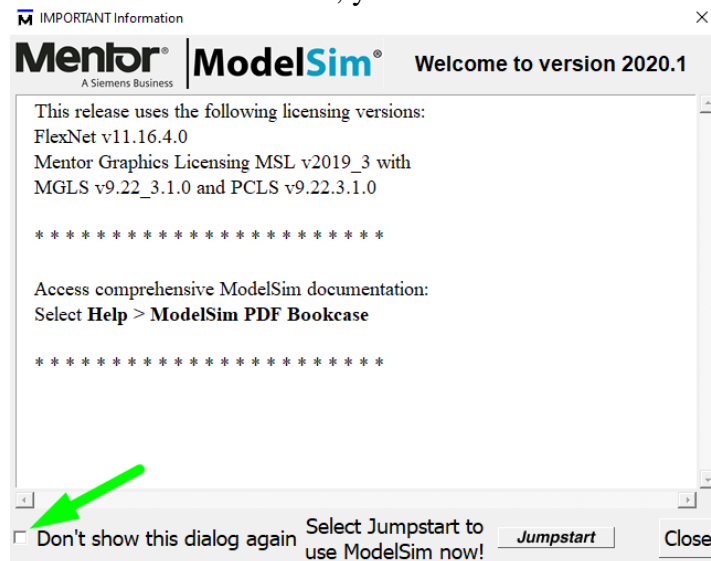


Figure 3

4. Click **Close**.

Create the Working Library

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

1. Change Directory.
 - a. Select File > Change Directory and change to the directory **C:\Intel_trn\ModelSim\Lab1_1**.
2. Create the working library.
 - a. Select File > New > Library.

This opens a dialog box where you specify physical and logical names for the library (Figure 4). You can create a new library or map to an existing library.

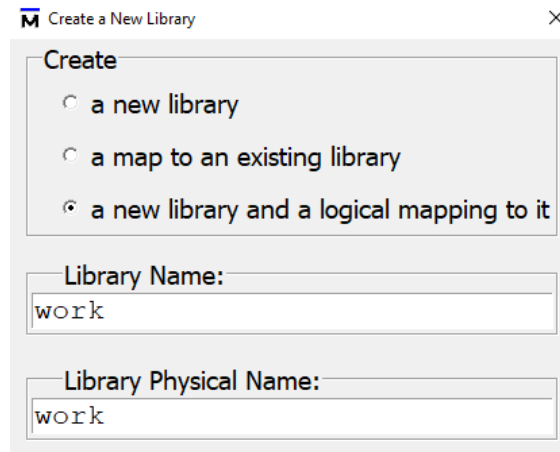


Figure 4

- b. Type **work** in the Library Name field (if it is not already entered automatically).
 - c. Click **OK**.

ModelSim creates a directory called *work* (**C:\Intel_trn\ModelSim\Lab1_1\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 Library window (Figure 5) and records the library mapping for future reference in the ModelSim initialization file (**C:\Intel_trn\ModelSim\Lab1_1\modelsim.ini**).







Library		
Name	Type	Path
 work (empty)	Library	work
 vital2000	Library	\$MODEL_TECH/../../vital2000
 verilog	Library	\$MODEL_TECH/../../verilog
 twentynm_ver	Library	\$MODEL_TECH/../../altera/verik
 twentynm_hssi_ver	Library	\$MODEL_TECH/../../altera/verik
 twentynm_hssi	Library	\$MODEL_TECH/../../altera/vhdl,

Figure 5

Compile the Design Units

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

You can compile your source files using the menus and dialog boxes of the graphic interface, as in the Verilog example below, or by entering a command at the ModelSim> prompt.

1. Compile counter.v and tcounter.v.
 - a. Select **Compile > Compile**. This opens the Compile Source Files dialog box (Figure 6).
 - b. Select both *counter.v* and *tcounter.v* modules from the Compile Source Files dialog box and click **Compile**. The files are compiled into the *work* library.
 - c. When compile is finished, click **Done**.

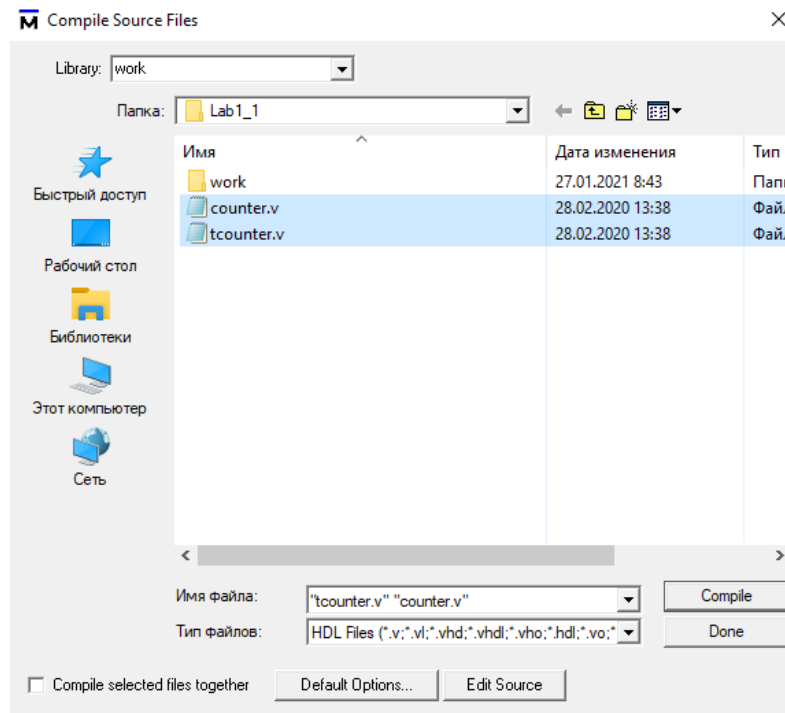


Figure 6

2. View the compiled design units.
 - a. In the Library window, click the '+' icon next to the *work* library and you will see two design units (Figure 7). You can also see their types (Modules, Entities, etc.) and the path to the underlying source files.

Library			Change Column Layout
Name	Type	Path	
work	Library	work	
test_counter	Module	C:/Intel_trn/ModelSim/Lab1_1/tcounter.v	
counter	Module	C:/Intel_trn/ModelSim/Lab1_1/counter.v	
vital2000	Library	\$MODEL_TECH/./vital2000	
verilog	Library	\$MODEL_TECH/./verilog	
twentynm_ver	Library	\$MODEL_TECH/./altera/verilog/twentynm	
twentynm_hssi_ver	Library	\$MODEL_TECH/./altera/verilog/twentynm...	

Figure 7

Load the Design

Now you're ready to load the design into the simulator.

1. Load the *test_counter* module into the simulator.
 - a. In the Library window, click the '+' sign next to the **work** library to show the files contained there.
 - b. Double-click *test_counter* 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 box. 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 8).

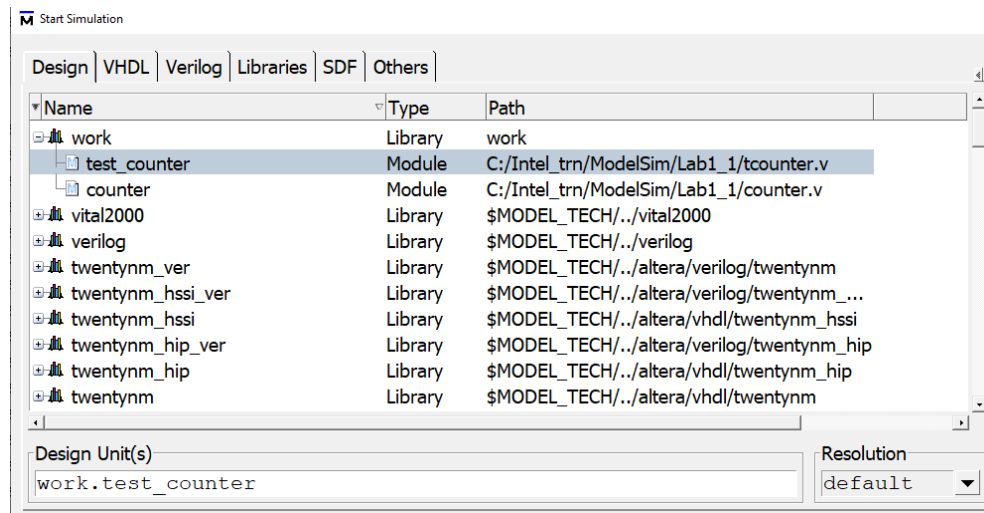


Figure 8

When the design is loaded, you will have Structure window (labeled **sim**), Objects, Processes, Wave windows opened (Figure 9).

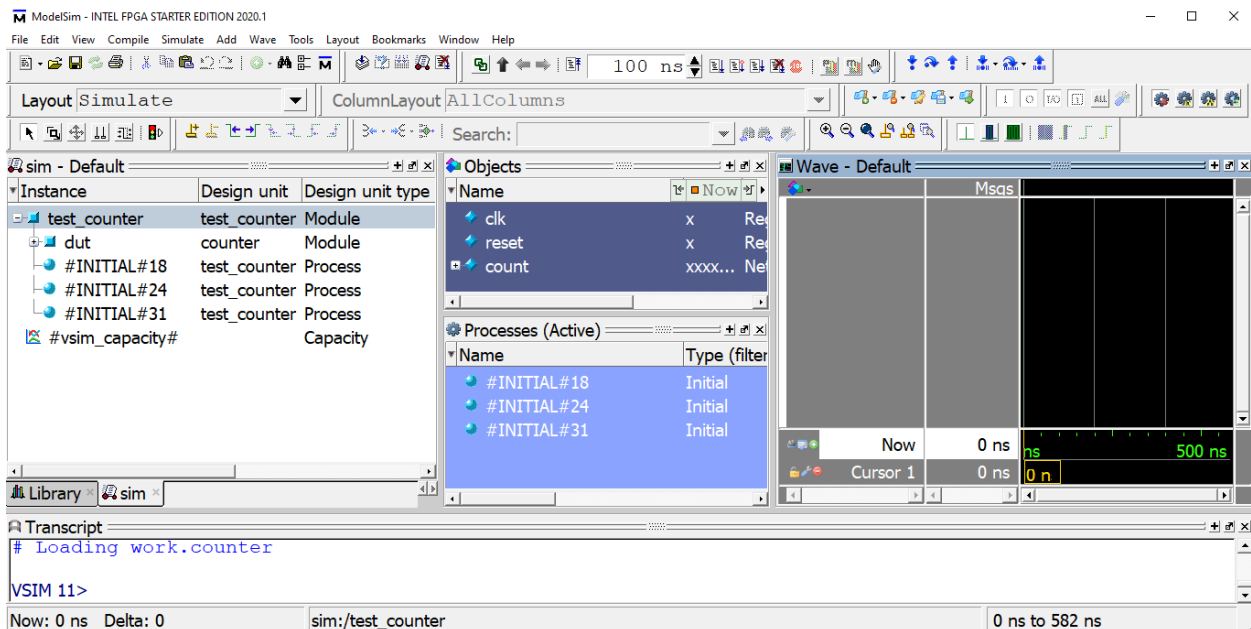


Figure 9

If you don't see some windows pointed above you can open these manually by selecting the windows in a menu bar View (Figure 10).

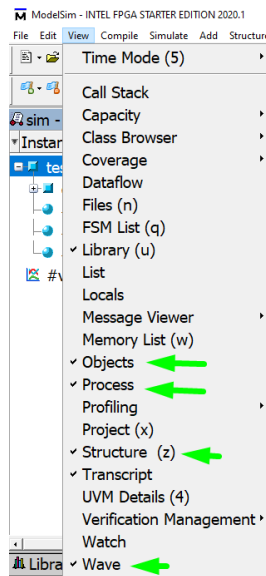


Figure 10

Structure window (labeled **sim**) window displays the hierarchical structure of the design as shown in Figure 11. You can navigate within the design hierarchy in the Structure (**sim**) window by clicking on any line with a '+' (expand) or '-' (contract) icon.

Instance	Design unit	Design unit type	Top Category	Visibility
test_counter	test_counter	Module	DU Instance	+acc=...
dut	counter	Module	DU Instance	+acc=...
#INITIAL#18	test_counter	Process	-	+acc=...
#INITIAL#24	test_counter	Process	-	+acc=...
#INITIAL#31	test_counter	Process	-	+acc=...
#vsim_capacity#		Capacity	Statistics	+acc=...

Figure 11

The Objects window shows the names and current values of data objects in the current region selected in the Structure (sim) window (Figure 12). Data objects include signals, nets, registers, constants and variables not declared in a process, generics, parameters.

The Processes window displays a list of processes in one of four viewing modes: Active, In Region, Design, and Hierarchical. The Design view mode is intended for primary navigation of ESL (Electronic System Level) designs where processes are a foremost consideration. By default, this window displays the active processes in your simulation (Active view mode).

The screenshot shows two windows. The top window is 'Objects' and the bottom is 'Processes (Active)'.

Name	Value	Kind	Mode
clk	1'bx	Register	Internal
reset	1'bx	Register	Internal
count	8'hxx	Net	Internal

Name	Type (filtered)	State	Order
#INITIAL#18	Initial	Ready	4
#INITIAL#24	Initial	Ready	5
#INITIAL#31	Initial	Ready	6

Figure 12

Run the Simulation

You are ready to run the simulation.

But before you do, you will add signals to the Wave window.

1. Add signals to the Wave window.
 - a. In the Structure (sim) window, right-click *test_counter* to open a popup context menu.
 - b. Select **Add Wave** (Figure 13).

All signals in the design are added to the Wave window.

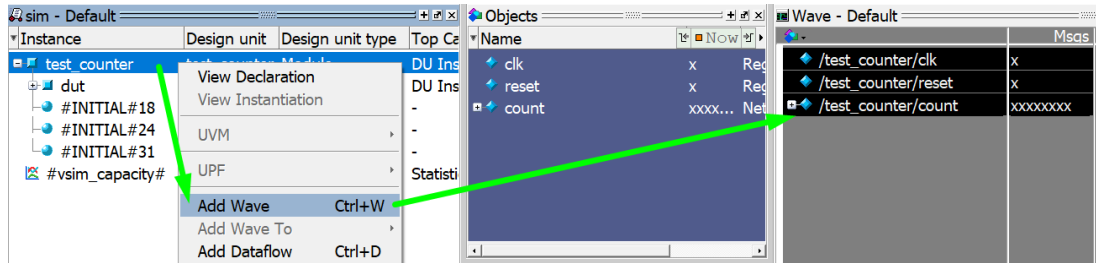


Figure 13

2. Run the simulation.
 - a. Click the Run icon.

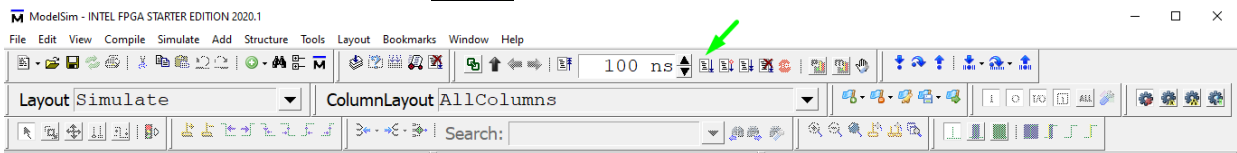


Figure 14

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

- b. Enter **run 500** at the VSIM> prompt in the Transcript window.

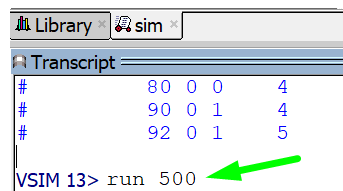


Figure 15

The simulation advances another 500 ns for a total of 600 ns (Figure 16).

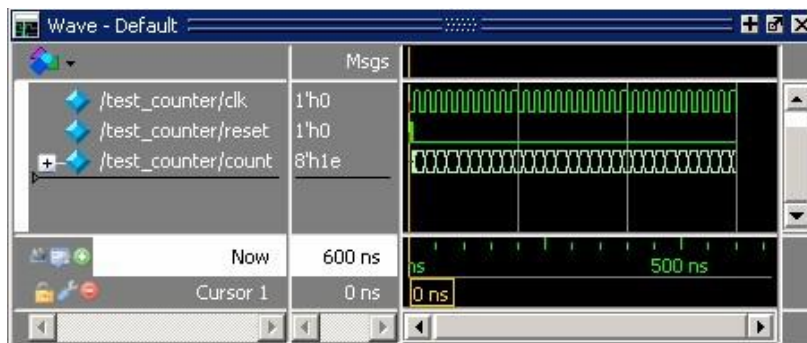


Figure 16

- 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 (ie., a Verilog \$stop statement) that halts the simulation.

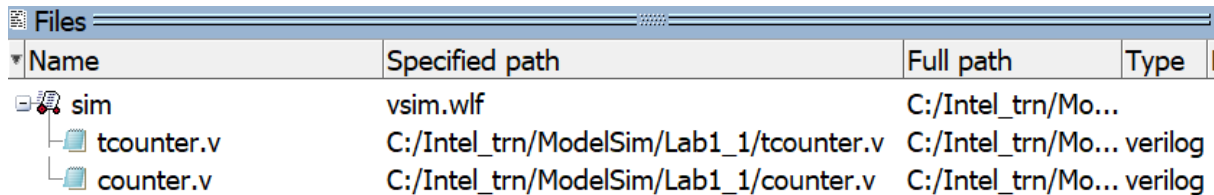


- d. Click the Break icon to stop the simulation.

Set Breakpoints and Step through the Source

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 executable lines, which are indicated with red line numbers.

1. Open *counter.v* in the Source window.
 - a. Select **View > Files** to open the Files window.
 - b. Click the + sign next to the *sim* filename to see the contents of *vsim.wlf* dataset.



Name	Specified path	Full path	Type
sim	vsim.wlf	C:/Intel_trn/Mo...	
tcounter.v	C:/Intel_trn/ModelSim/Lab1_1/tcounter.v	C:/Intel_trn/Mo...	verilog
counter.v	C:/Intel_trn/ModelSim/Lab1_1/counter.v	C:/Intel_trn/Mo...	verilog

Figure 17

- c. Double-click *counter.v* (or *counter.vhd* if you are simulating the VHDL files) to open the file in the Source window.
2. Set a breakpoint on line 36 of *counter.v* (or, line 39 of *counter.vhd* for VHDL).
 - a. Scroll to line 36 and click in the Ln# (line number) column next to the line number.

A red dot appears in the line number column at line number 36 (Figure 18), indicating that a breakpoint has been set.

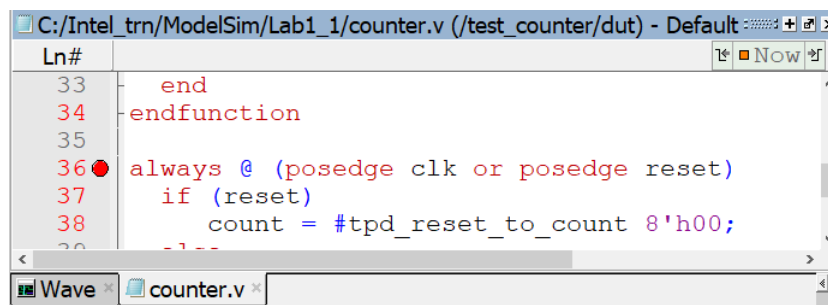



Figure 18

3. Disable, enable, and delete the breakpoint.
 - a. Click the red dot to disable the breakpoint. It will become a gray dot.
 - b. Click the gray dot again to re-enable the breakpoint. It will become a red dot.
 - c. Click the red dot with your right mouse button and select **Remove Breakpoint 36**.
 - d. Click in the line number column next to line number 36 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 box that appears gives you options on what to retain during the restart (Figure 19).



Figure 19

- b. Click the **OK** button in the Restart dialog box.
- 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 20), and issues a Break message in the Transcript window.

Figure 3-12. Blue Arrow Indicates Where Simulation Stopped.

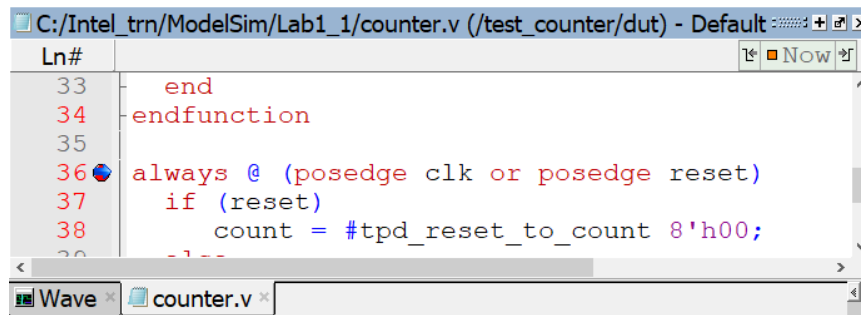


Figure 20

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 21).

Objects				
Name	Value	Kind	Mode	
tpd_reset_to_count	3	Parameter	Internal	
tpd_clk_to_count	2	Parameter	Internal	
count	xxxxxxx	Packed Array	Out	
clk	St0	Net	In	
reset	St1	Net	In	

Figure 21

- Set your mouse pointer over a variable in the Source window and a yellow box will appear with the variable name and the value of that variable at the time of the selected cursor in the Wave window (Figure 22).

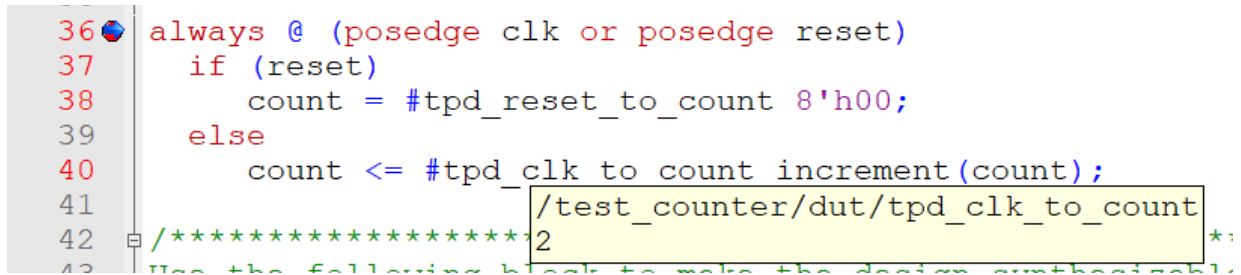


Figure 22

- Highlight a signal, parameter, or variable in the Source window, right-click it, and select **Examine** from the pop-up menu to display the variable and its current value in a Source Examine window (Figure 23).

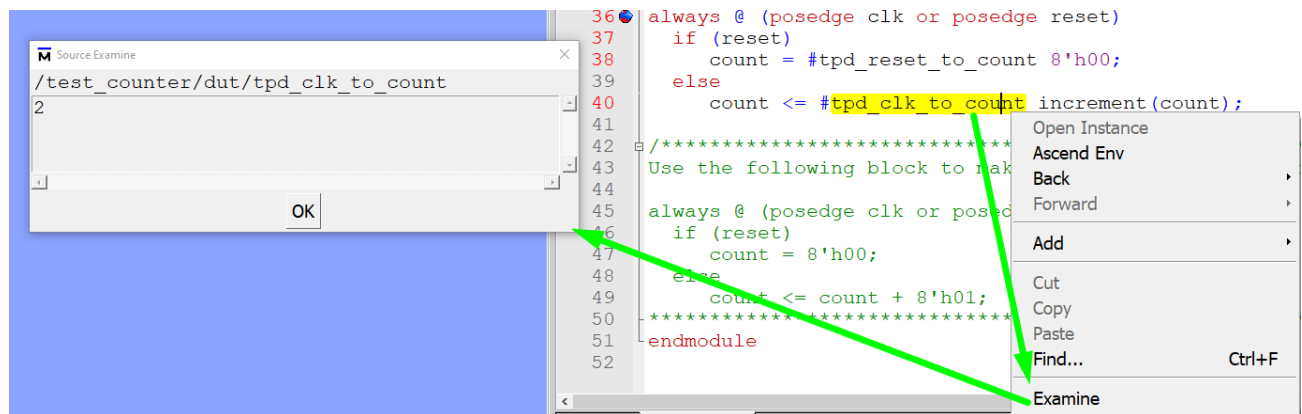


Figure 23

- use the **examine** command at the VSIM> prompt to output a variable value to the Transcript window (i.e., examine count)

5. Try out the step commands.

- Click the Step Into icon on the Step 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.

5. Before continuing, you need to end the current simulation.

- Select Simulate > End Simulation.
- Click **Yes** when prompted to confirm that you wish to quit simulating.

Analyzing Waveforms

The Wave window allows you to view the results of your simulation as HDL waveforms and their values. The Wave window is divided into a number of panes (Figure 24). You can resize the pathnames pane, the values pane, and the waveform pane by clicking and dragging the bar between any two panes.

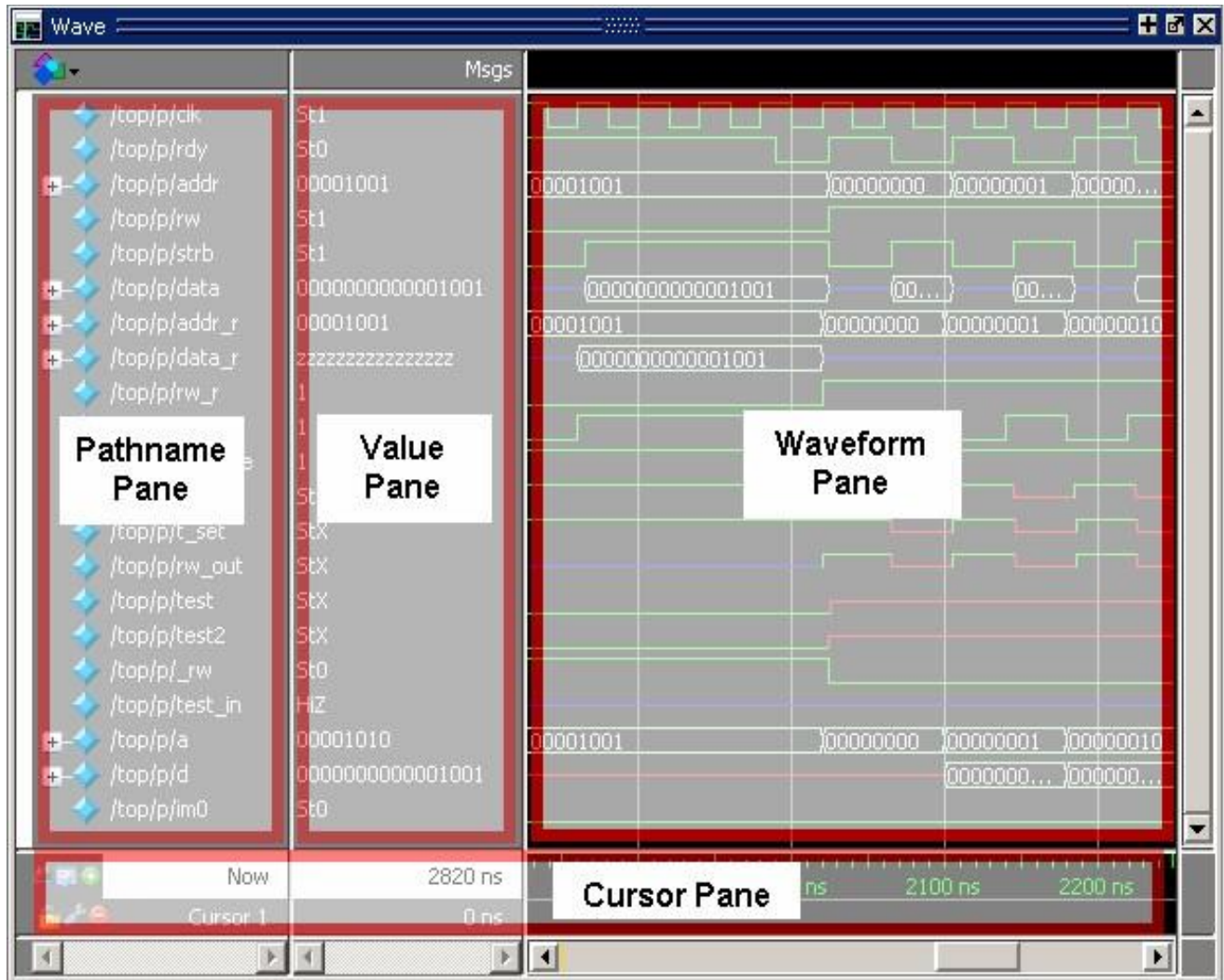


Figure 24

Adding objects to the Wave window

ModelSim offers several methods for adding objects to the Wave window. In this lab, you will try different methods.

1. Click the '+' icon next to the *work* library and double-click *test_counter*.
2. Add objects from the Objects window.
 - a. If it is not already opened, open an Objects window by selecting **View > Objects**.
 - b. Select an item in the Objects window, right-click, and then select **Add to > Wave > Signals in Region**. ModelSim opens a Wave window (if it is not already opened) and displays signals in the region.

Or, place the cursor over an object and click the right mouse button to open a context menu. Then click **Add Wave** to place the object in the Wave window.

Or, select a group of objects then click the right mouse button to open the context menu and click **Add Wave**.

By default ModelSim opens the Wave window in the right side of the Main window. You can change the default via the Preferences dialog box (**Tools > Edit Preferences**).

3. Undock the Wave window.

- a. Click the undock icon on the Wave window.



The Wave window becomes a standalone, un-docked window. Resize the window as needed.

- b. Click the dock icon on the Wave window.



4. Add objects using drag-and-drop.

You can drag an object to the Wave window from many other windows (e.g., Structure, Objects, and Locals).

- a. In the Wave window, select **Edit > Select All** and then **Edit > Delete**.
 - b. Drag an instance (for example: test_counter) from the Structure (sim) window to the Wave window.

ModelSim adds the objects for that instance to the Wave window.

- c. In the Wave window, select **Edit > Select All** and then **Edit > Delete**.
 - d. Drag a signal from the Objects window to the Wave window.
 - e. In the Wave window, select **Edit > Select All** and then **Edit > Delete**.

5. Add objects using the **add wave** command.

- a. Type the following at the VSIM> prompt **add wave ***

ModelSim adds all objects from the current selected region.

- b. Run the simulation for 500 ns so you can see waveforms.

Zooming the Waveform Display

There are numerous methods for zooming the Waveform display. This exercise will show you how to zoom using various techniques.

1. Click the undock icon on the Wave window.



2. Click the Zoom Mode icon on the Wave window toolbar.



- a. In the waveform display, click and drag down and to the right.
 - b. You should see blue vertical lines and numbers defining an area to zoom in (Figure 25).

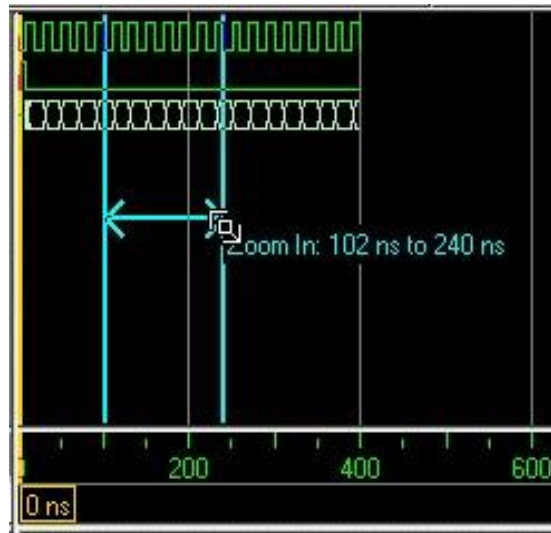



Figure 25

3. Select View > Zoom > Zoom Last.

The waveform display restores the previous display range.

4. Click the Zoom In icon a few times. 
5. In the waveform display, click and drag up and to the right.

You should see a blue line and numbers defining an area to zoom out.

6. Select View > Zoom > Zoom Full.
You should see the waveform zoomed to all available window.

7. Click the Select Mode icon on the Wave window toolbar. 

8. Click the dock icon on the Wave window 

Radix and Format in the Wave window

You can change Radix and Format for buses in the Wave window.

1. Right click bus *count* in the pathname pane.
2. In the pop-up window select Radix=>Unsigned

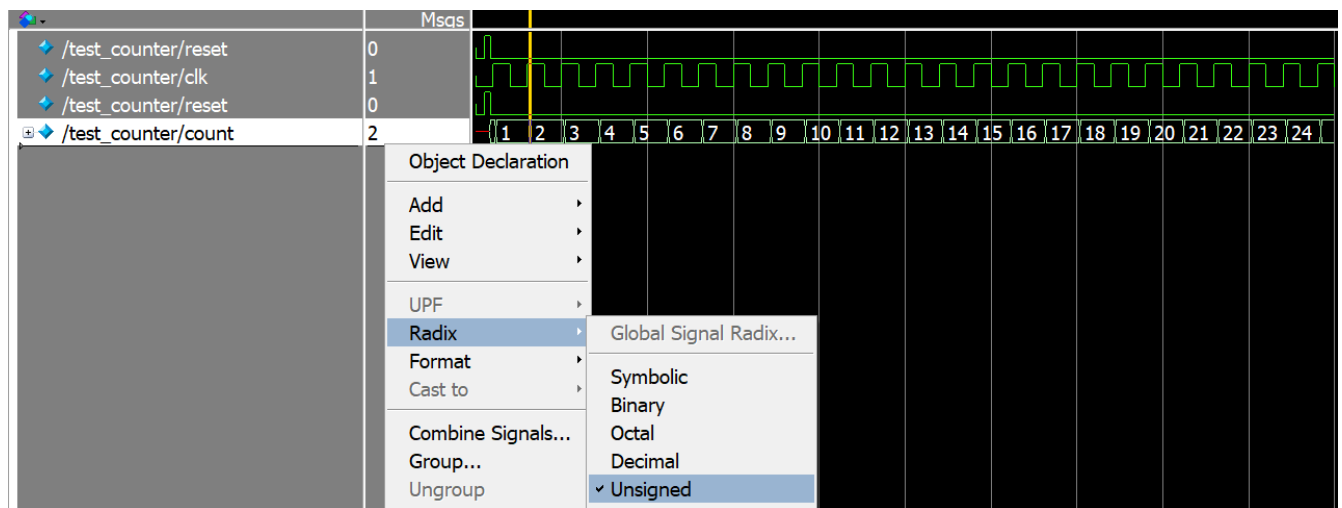


Figure 26

Values of the bus *count* will be displayed as unsigned decimal.

3. Right click bus *count* in the pathname pane.
4. In the pop-up window select Format=>Analog (automatic)

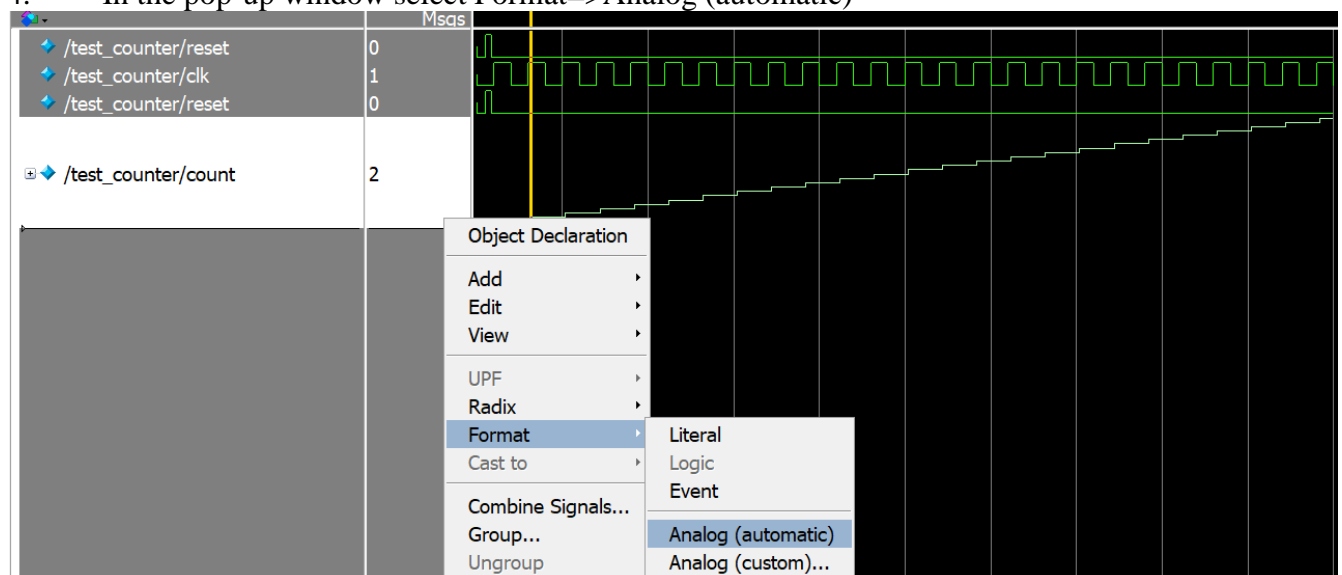


Figure 27

Values of the bus *count* will be displayed in analog manner.

5. Right click bus *count* in the pathname pane.
6. In the pop-up window select Format=>Literal

Using Cursors in the Wave Window

Cursors mark simulation time in the Wave window. When ModelSim first draws the Wave window, it places one cursor at time zero. Clicking in the cursor timeline brings the cursor to the mouse location.

You can also:

- add additional cursors;
- name, lock, and delete cursors;
- use cursors to measure time intervals; and

- use cursors to find transitions.

Let's look at the information provided when using a single cursor.

1. Position the cursor by clicking in the cursor timeline then dragging.
 - a. Click anywhere in the cursor timeline.

The cursor snaps to the time where you clicked (Figure 28).

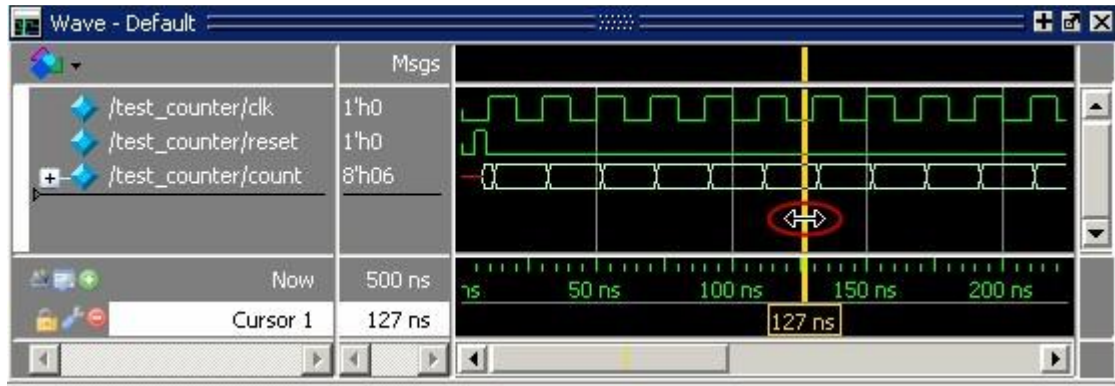


Figure 28

- c. Drag the cursor and observe the value pane.

The signal values change as you move the cursor. This is perhaps the easiest way to examine the value of a signal at a particular time.

You can set the snap distance in the Window Preferences dialog box.

- d. Select **Tools > Window Preferences**
- e. Set **Snap Distance** value as 50 and click OK.

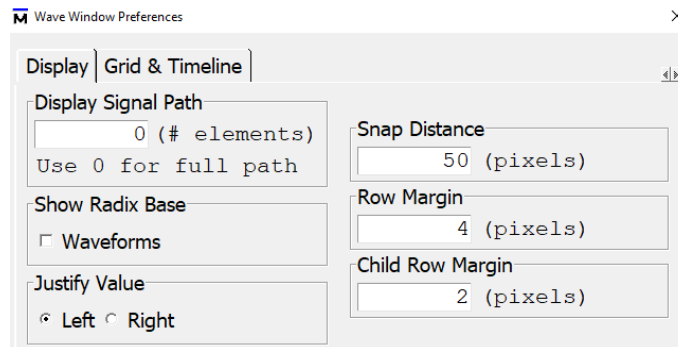


Figure 29

- f. In the waveform pane, position the mouse pointer over the cursor line. When the pointer changes to a two headed arrow (Figure 30), click and hold the left mouse button to select the cursor. Drag the cursor to the right of a transition.

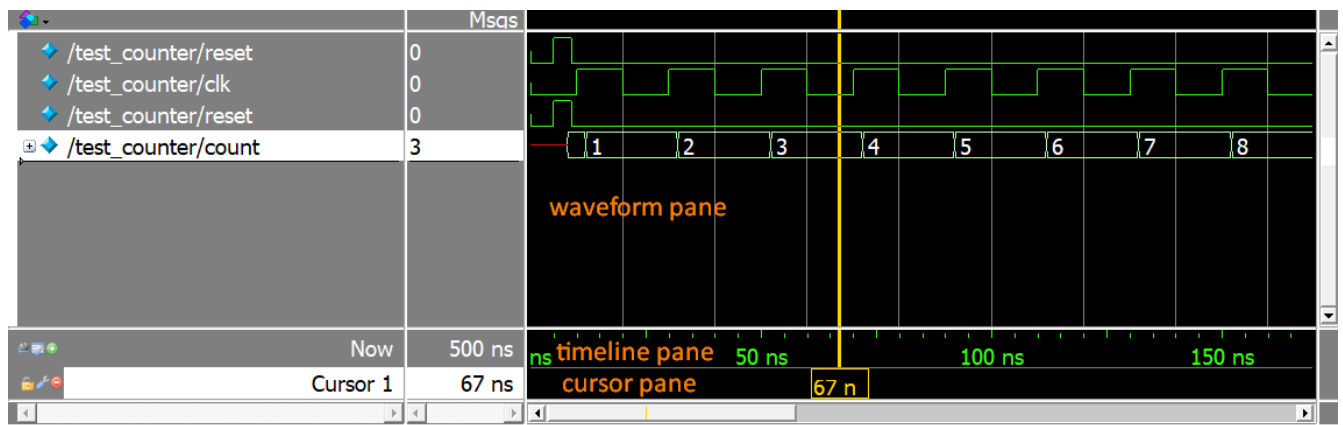


Figure 30

The cursor "snaps" to the nearest transition to the left when you release the mouse button. Cursors "snap" to a waveform edge when you drag a cursor to within 50 pixels of an edge.

g. In the cursor timeline pane, select the yellow timeline indicator box then drag the cursor to the right of a transition (Figure 30).

The cursor does not snap to a transition when you drag in the timeline pane.

2. Rename the cursor.

- Right-click "Cursor 1" in the cursor pane, then select and delete the text.
- Type **A** and press Enter.

The cursor name changes to "A" (Figure 31).

Using Cursors in the Wave Window

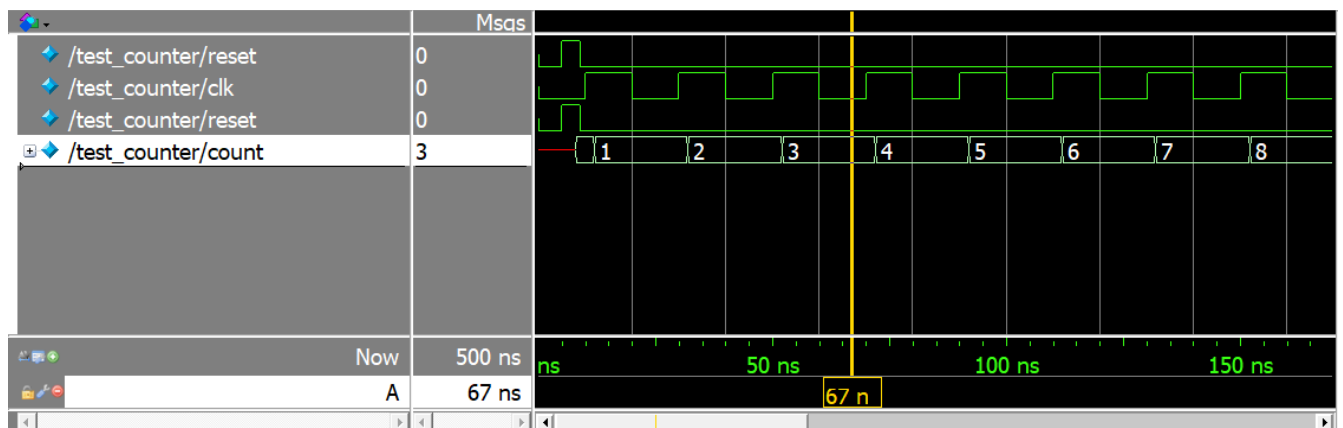


Figure 31

3. Jump the cursor to the next or previous transition.

- Click signal *count* in the pathname pane.

- Click the Find Next Transition icon on the Wave window toolbar.



The cursor jumps to the next transition on the selected signal.

- Click the Find Previous Transition icon on the Wave window toolbar.



The cursor jumps to the previous transition on the selected signal.

Even more information is available when working with multiple cursors.

1. Add a second cursor.
 - a. Click the Insert Cursor icon on the Wave window toolbar.
 - b. Right-click the name of the new cursor and delete the text.
 - c. Type **B** and press Enter.
 - d. Drag cursor *B* and watch the interval measurement change dynamically (Figure 32).

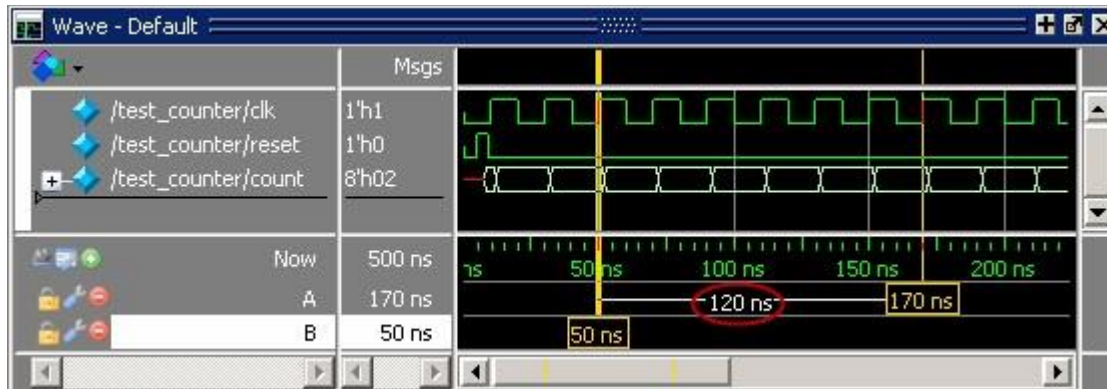


Figure 32

2. Lock cursor *B*.
 - a. Right-click the yellow time indicator box associated with cursor *B* (at 50 ns).
 - b. Select **Lock B** from the popup menu.

The cursor color changes to red and you can no longer drag the cursor (Figure 33).

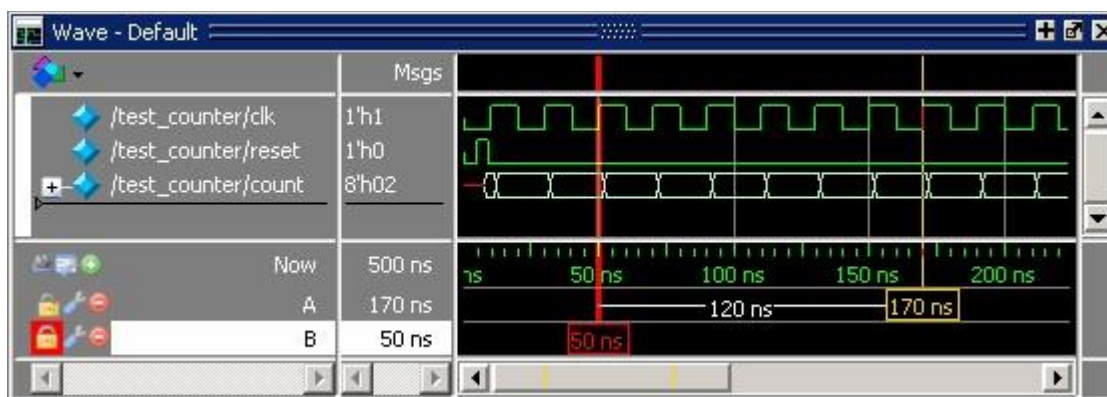


Figure 33

3. Delete cursor *B*.
 - a. Right-click cursor *B* (the red box at 50 ns) and select **Delete B**.

Saving and Reusing the Window Format

If you close the Wave window, any configurations you made to the window (e.g., signals added, cursors set, etc.) are discarded. However, you can use the Save Format command to capture the current Wave window display and signal preferences to a *.do* file. You open the *.do* file later to recreate the Wave window as it appeared when the file was created.

Pay attention: Format files (*.do* files) are design-specific; use them only with the design you were simulating when they were created.

1. Save a format file.
 - a. In the **Wave** window, select **File > Save Format**.
 - b. In the Pathname field of the Save Format dialog box, leave the file name set to *wave.do* and click **OK**.
 - c. Close the Wave window.
2. Load a format file.
 - a. In the Main window, select **View > Wave**.
 - b. Undock the window.

All signals and cursor(s) that you had set are gone.

- c. In the Wave window, select **File > Load**.
- d. In the Open Format dialog box, select *wave.do* and click **Open**.

ModelSim restores the window to its previous state.

- e. Close the Wave window when you are finished by selecting **File > Close Window**.

Concluding the Lab

This concludes this Lab.

1. Select **Simulate > End Simulation**.
2. Click **Yes**.
3. Select **File > Quit** to close ModelSim.

Lab1_2: Projects Flow

In this lab you will practice creating a project.

At a minimum, projects contain a work library and a session state that is stored in an *.mpf* file. A project may also consist of:

- HDL source files or references to source files
- other files such as READMEs or other project documentation
- local libraries
- references to global libraries

A project is a collection mechanism for an HDL design under specification or test. Even though you do not have to use projects in ModelSim, projects may ease interaction with the tool and are useful for organizing files and specifying simulation settings.

The following diagram shows the basic steps for simulating a design within a ModelSim project.

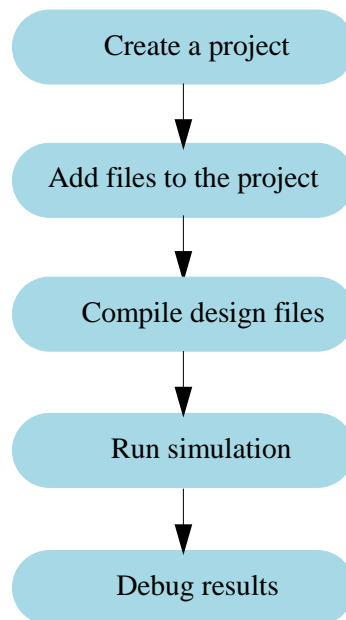


Figure 34

As you can see, the flow is similar to the basic simulation flow. However, there are two important differences:

- You do not have to create a working library in the project flow; it is done for you automatically.
- Projects are persistent. In other words, they will open every time you invoke ModelSim unless you specifically close them.

Design Files

The sample design for this lab is a simple 8-bit, binary up-counter with an associated test bench.

This lesson uses the Verilog files *counter.v* and *tcounter.v*. These files are located in the **C:\Intel_trn\ModelSim\Lab1_2** folder. Which is a working folder for the Lab1_2.

Start ModelSim

Procedure to start ModelSim :

1. Push Start button in Windows.
2. Click on ModelSim Intel FPGA Started Edition icon.

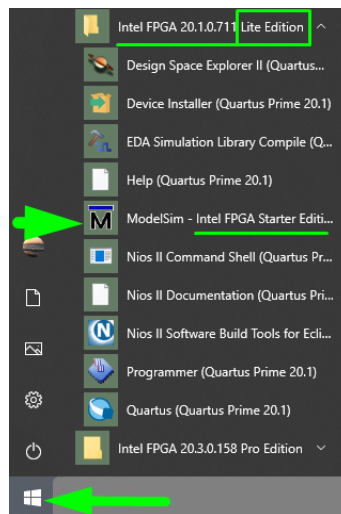


Figure 35

Upon opening ModelSim for the first time, you will see the Welcome to ModelSim dialog box. Click **Close**.

Create a New Project

We will start the process of creating a new project by defining the project settings.

1. Select **File > New > Project** (Main window) from the menu bar.

This opens the Create Project dialog box where you can enter a Project Name, Project Location (i.e., directory), and Default Library Name (Figure 36). You can also reference library settings from a selected .ini file or copy them directly into the project. The default library is where compiled design units will reside.

2. Type **test** in the Project Name field.
3. Click the **Browse** button for the Project Location field to select a directory **C:\Intel_trn\ModelSim\Lab1_2** where the project file will be stored.
4. Leave the Default Library Name set to *work*.

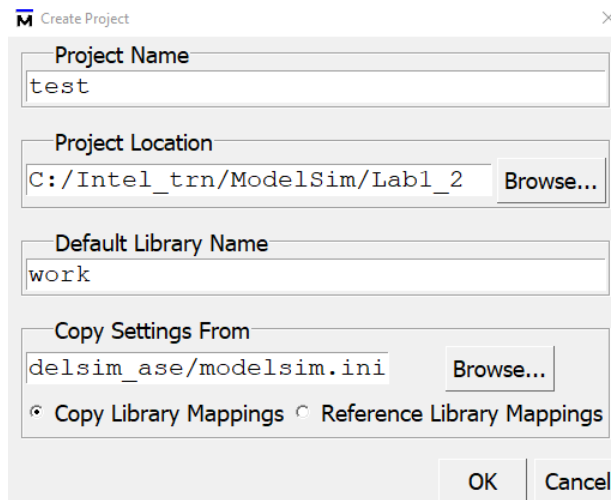


Figure 36

5. Click **OK**.

Add Objects to the Project

Once you click OK to accept the new project settings, a blank Project window and the “Add items to the Project” dialog box will appear.

From the dialog box (Figure 37) you can create a new design file, add an existing file, add a folder for organization purposes, or create a simulation configuration (discussed below).

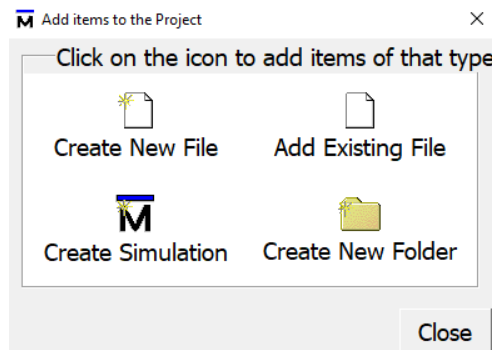


Figure 37

1. Add two existing files.
 - a. Click Add Existing File.

This opens the Add file to Project dialog box (Figure 4-3). This dialog box lets you browse to find files, specify the file type, specify a folder to which the file will be added, and identify whether to leave the file in its current location or to copy it to the project directory.

- b. Click the **Browse** button for the File Name field. This opens the “Select files to add to project” dialog box and displays the contents of the current directory.
- c. Select *counter.v* and *tcounter.v*
- d. Click **Open**.

This closes the “Select files to add to project” dialog box and displays the selected files in the “Add file to Project” dialog box (Figure 38).

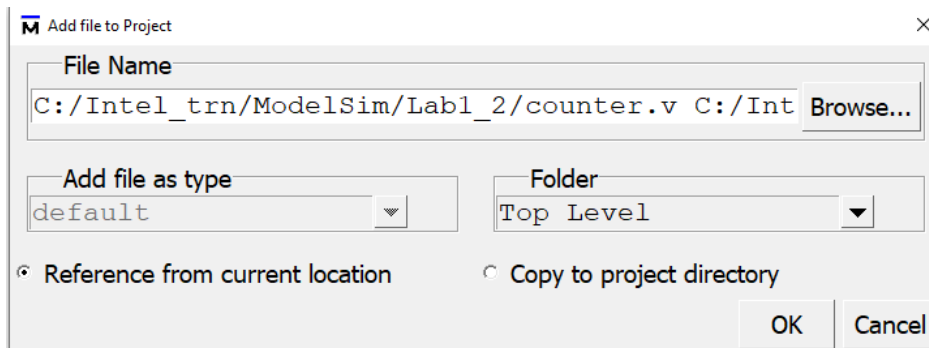


Figure 38

- e. Click **OK** to add the files to the project.
- f. Click **Close** to dismiss the Add items to the Project dialog box.

You should now see two files listed in the Project window (Figure 39).

Project - C:/Intel_trn/ModelSim/Lab1_2/test				
Name	Status	Type	Order	Modified
tcounter.v	?	Verilog	1	02/28/2020 01:38:17 ...
counter.v	?	Verilog	0	02/28/2020 01:38:17 ...

Figure 39

Question mark icons in the Status column indicate that the file has not been compiled or that 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.

Compile the Design

With the Project settings defined and objects added to the project, you are ready to compile the design.

1. Right-click either *counter.v* or *tcounter.v* in the Project window and select **Compile > Compile All** from the pop-up menu.

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

2. View the design units.
 - a. Click the **Library** tab (Figure 40).
 - 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.

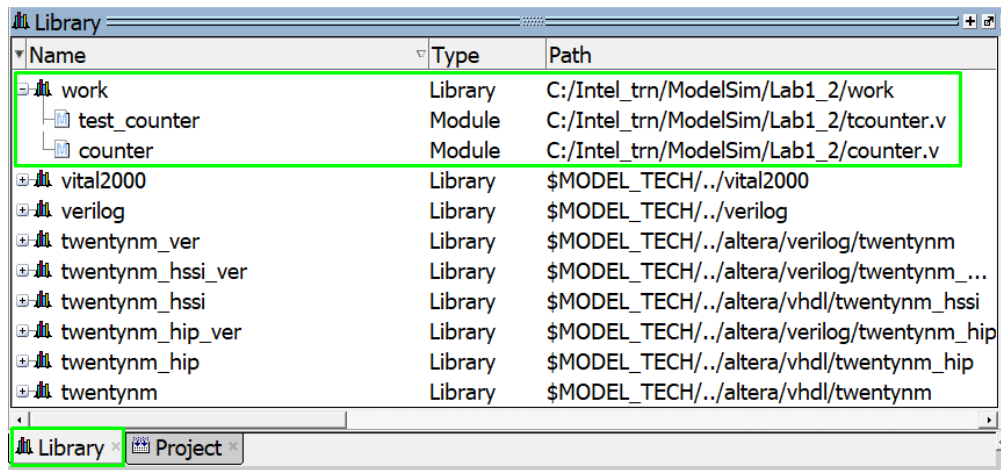


Figure 40

Load the Design

Now you are ready to load the design into the simulator.

1. Load the *test_counter* design unit.
 - a. Double-click the *test_counter* design unit.

The Structure (sim), Objects, Processes and Wave windows appear as part of the tab group **sim** (Figure 41).

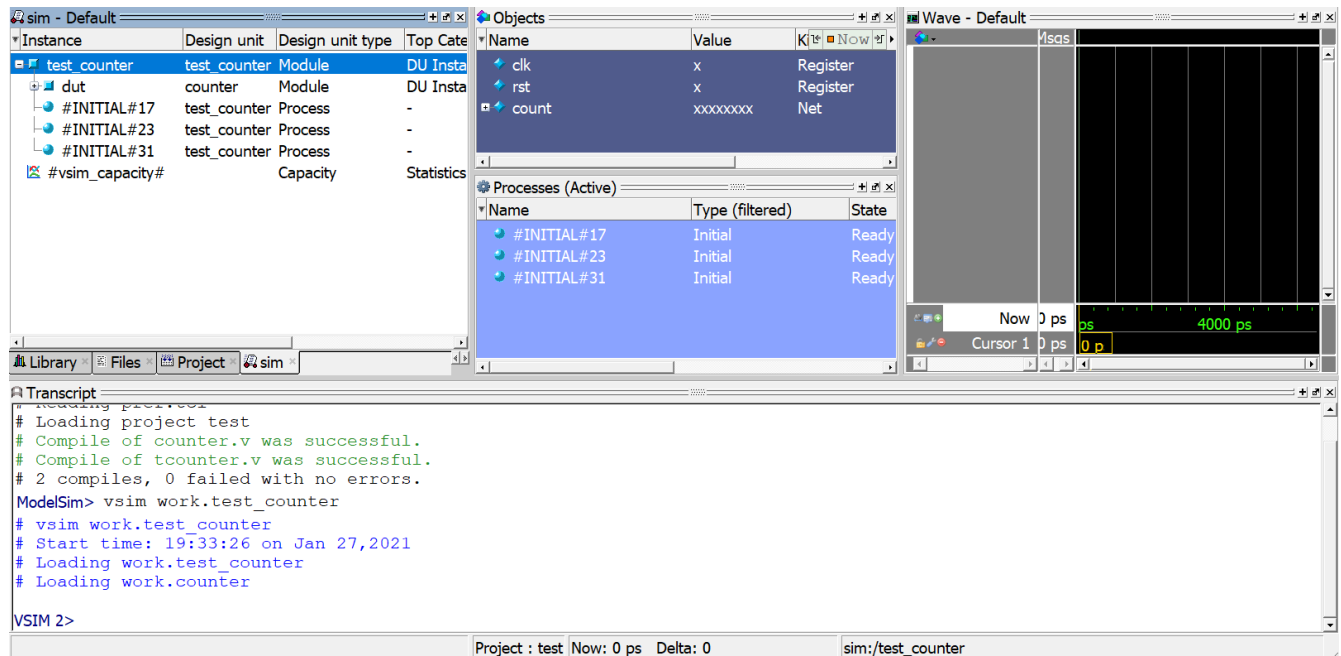


Figure 41

At this point, you would typically run the simulation and analyze or debug your design like you did in the Lab1_1.

For now, you will continue working with the project.

However, first you need to end the simulation that started when you loaded *test_counter*.

2. Select **Simulate > End Simulation**.
3. Click **Yes**.

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 4-3](#)). If you create a folder after adding files, you can edit the file properties to move it to that folder.

Adding Folders

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

1. Add a new folder.
 - a. Be sure you select the **Project** tab.
 - b. Right-click in the Projects window and select **Add to Project > Folder**.

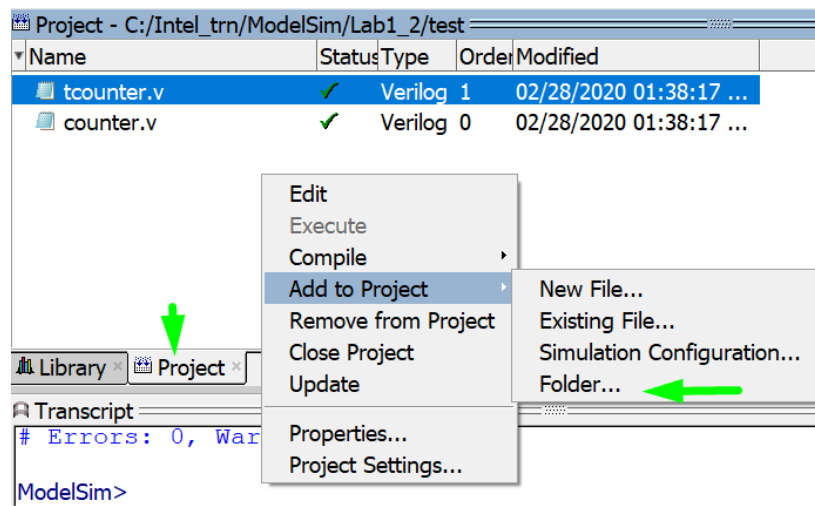


Figure 42

- c. Type **Design Files** in the **Folder Name** field (Figure 43).

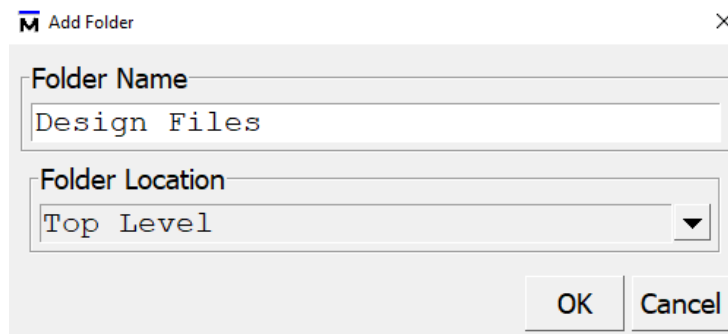


Figure 43

- d. Click **OK**.

The new Design Files folder is displayed in the Project window (Figure 44).

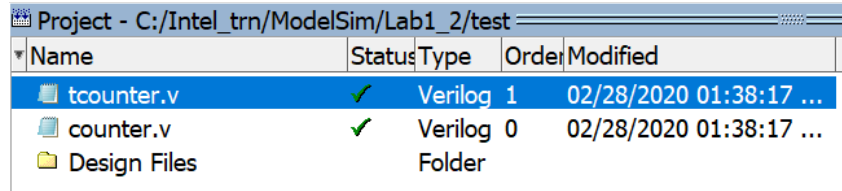


Figure 44

2. Add a sub-folder.
 - a. Right-click anywhere in the Project window and select **Add to Project > Folder**.
 - b. Type **HDL** in the **Folder Name** field (Figure 45).
 - c. Click the **Folder Location** drop-down arrow and select *Design Files*.

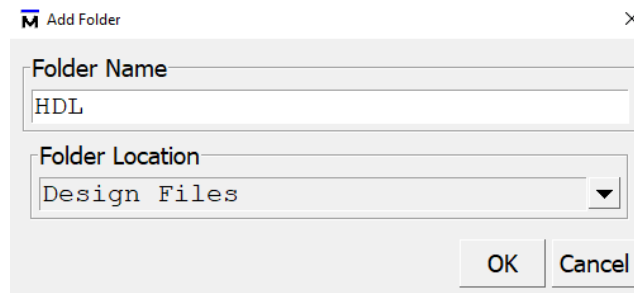


Figure 45

- d. Click **OK**.
- A '+' icon appears next to the *Design Files* folder in the Project window (Figure 46).

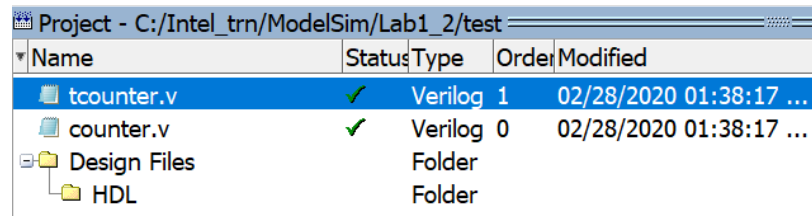


Figure 46

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

Moving Files to Folders

If you do not place files into a folder when you first add the files to the project, you can move them into a folder using the Project Compiler Settings dialog box.

1. Move *tcounter.v* and *counter.v* to the *HDL* folder.
 - a. Select both *counter.v* and *tcounter.v* in the Project window.
 - b. Right-click either file and select **Properties**.

This opens the Project Compiler Settings dialog box (Figure 47), which allows you to set a variety of options on your design files.

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

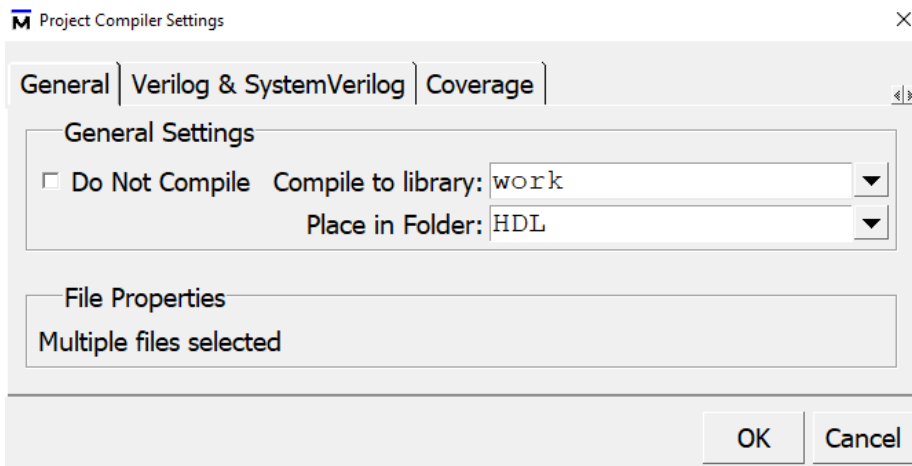


Figure 47

- d. Click **OK**.

The selected files are moved into the HDL folder. Click the '+' icon next to the HDL folder to see the files. The files are now marked with a '?' in the Status column because you moved the files. The project no longer knows if the previous compilation is still valid.

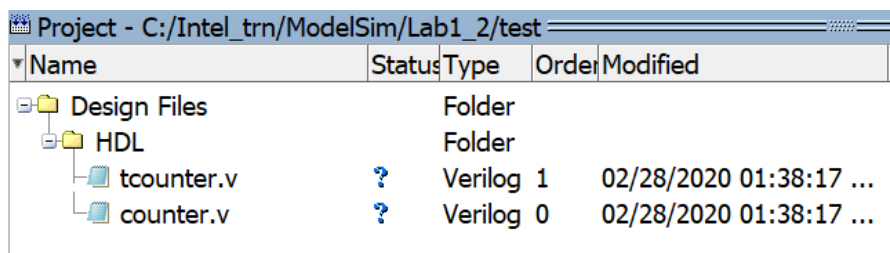


Figure 48

Using Simulation Configurations

A Simulation Configuration associates a design unit(s) and its simulation options. For example, let's say that 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 window and you can double-click it to load *tcounter.v* along with its options.

1. Create a new Simulation Configuration.
 - a. Right-click in the Project window and select **Add to Project > Simulation Configuration** from the popup menu.

This opens the Add Simulation Configuration dialog box (Figure 49). The tabs in this dialog box present several simulation options. You may want to explore the tabs to see what is 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 (-hazards)**.

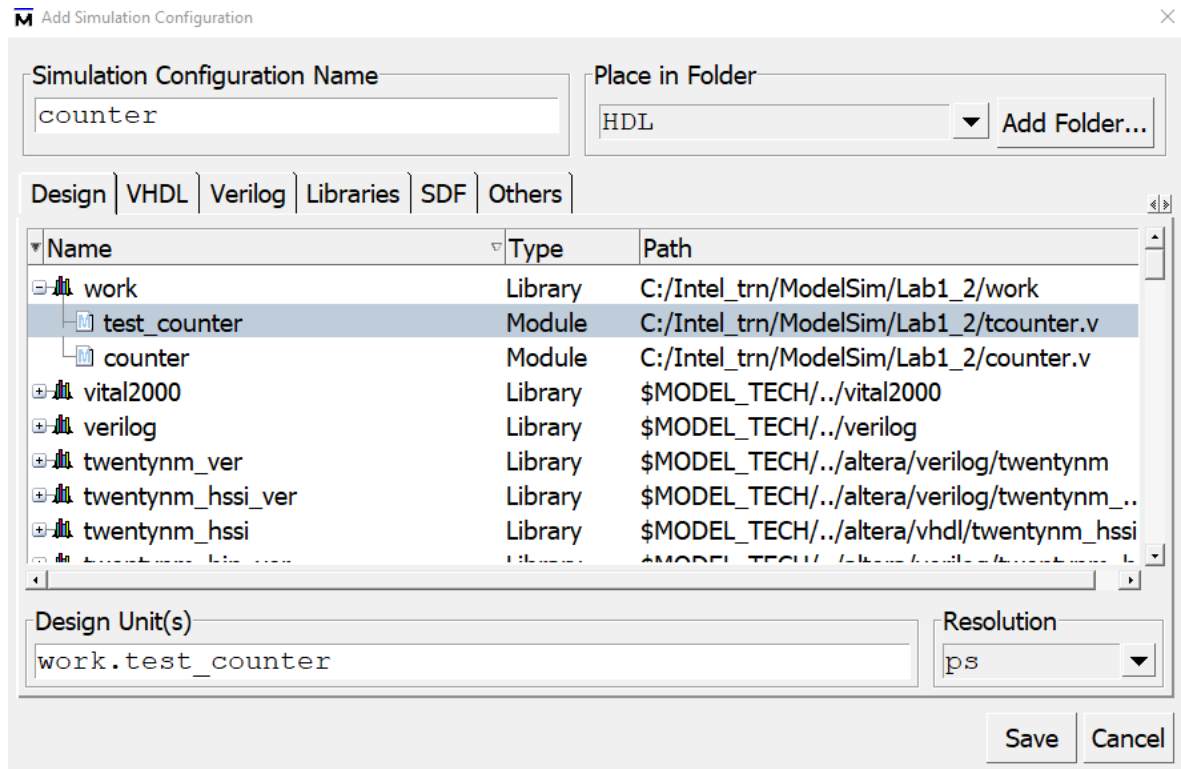


Figure 49

- g. Click **Save**.

The files *tcounter.v* and *counter.v* show question mark icons in the status column because they have changed location since they were last compiled and need to be recompiled.

- h. Select one of the files, *tcounter.v* or *counter.v*.
i. Select **Compile > Compile All**.

The Project window now shows a Simulation Configuration named *counter* in the HDL folder (Figure 50).

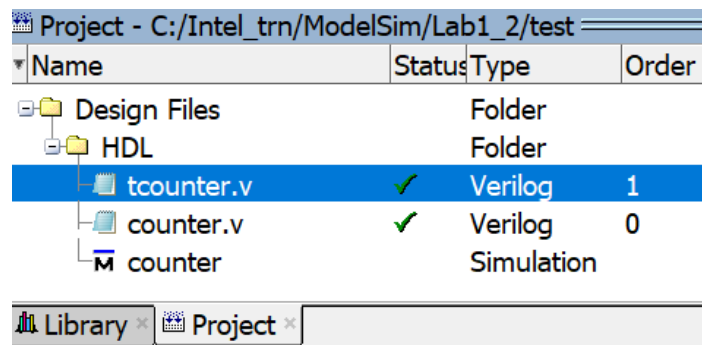


Figure 50

2. Load the Simulation Configuration.
a. Double-click the **counter** Simulation Configuration in the Project window.

The Structure (sim), Objects, Processes and Wave windows appear as part of the tab group **sim** (Figure 51).

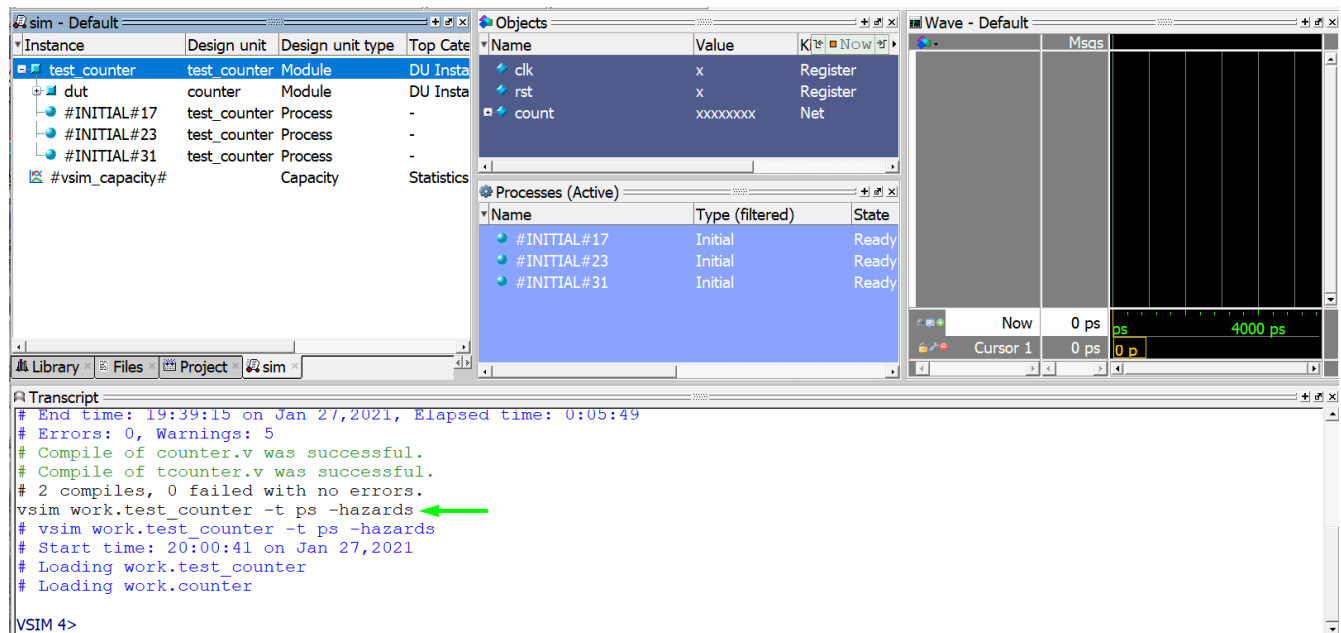


Figure 51

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

Concluding the Lab

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

1. Select **Simulate > End Simulation**.
2. Click **Yes**.
3. In the Project window, right-click and select **Close Project**.
If you do not close the project, it will open automatically the next time you start ModelSim.
4. Select **File > Quit** to close ModelSim.

Lab1_3: Working with Multiple Libraries

In this Lab you will practice working with multiple libraries. 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 test bench into it. Finally, you will link to the library containing the counter and then run the simulation.

ModelSim uses libraries in two ways:

- as a local working library that contains the compiled version of your design;
- as a resource library.

The contents of your working library will change as you update your design and recompile. A resource library is typically static and serves as a parts source for your design. You can create your own resource libraries, or they may be supplied by another design team or a third party (e.g., a silicon vendor).

You specify which resource libraries will be used when the design is compiled, and there are rules to specify in which order they are searched. A common example of using both a working library and a resource library is one where your gate-level design and test bench are compiled into the working library, and the design references gate-level models in a separate resource library.

The diagram below shows the basic steps for simulating with multiple libraries.

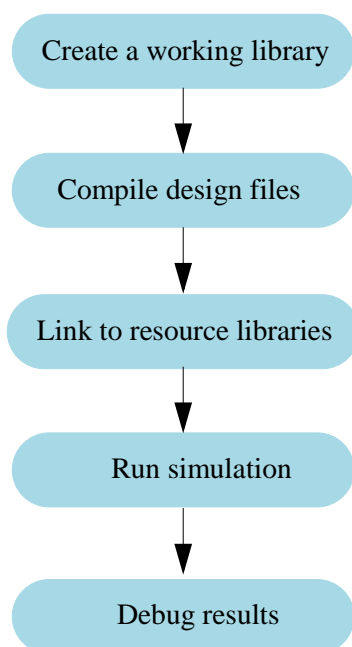


Figure 52

The sample design for this lab is a simple 8-bit, binary up-counter with an associated test bench.

This lesson uses the Verilog files *counter.v* and *tcounter.v*. These files are located in the **C:\Intel_trn\ModelSim\Lab1_3** folder. Which is a working folder for the Lab1_3.

1. Create a directory for the resource library.
 - a. In the folder **C:\Intel_trn\ModelSim\Lab1_3** create a new directory called **resource_library**.
 - b. Copy *counter.v* from **C:\Intel_trn\ModelSim\Lab1_3** to the new directory **C:\Intel_trn\ModelSim\Lab1_3\resource_library**.
2. Create a directory for the test bench.
 - a. In the folder **C:\Intel_trn\ModelSim\Lab1_3** create a new directory called **testbench** that will hold the test bench and project files.
 - b. Copy *tcounter.v* from **C:\Intel_trn\ModelSim\Lab1_3** to the new directory **C:\Intel_trn\ModelSim\Lab1_3\testbench**.

You are creating two directories in this lesson to mimic the situation where you receive a resource library from a third-party.

Before moving forward, make sure the *modelsim.ini* in your install directory is “Read Only.” This will prevent permanent mapping of resource libraries to the master *modelsim.ini* file.

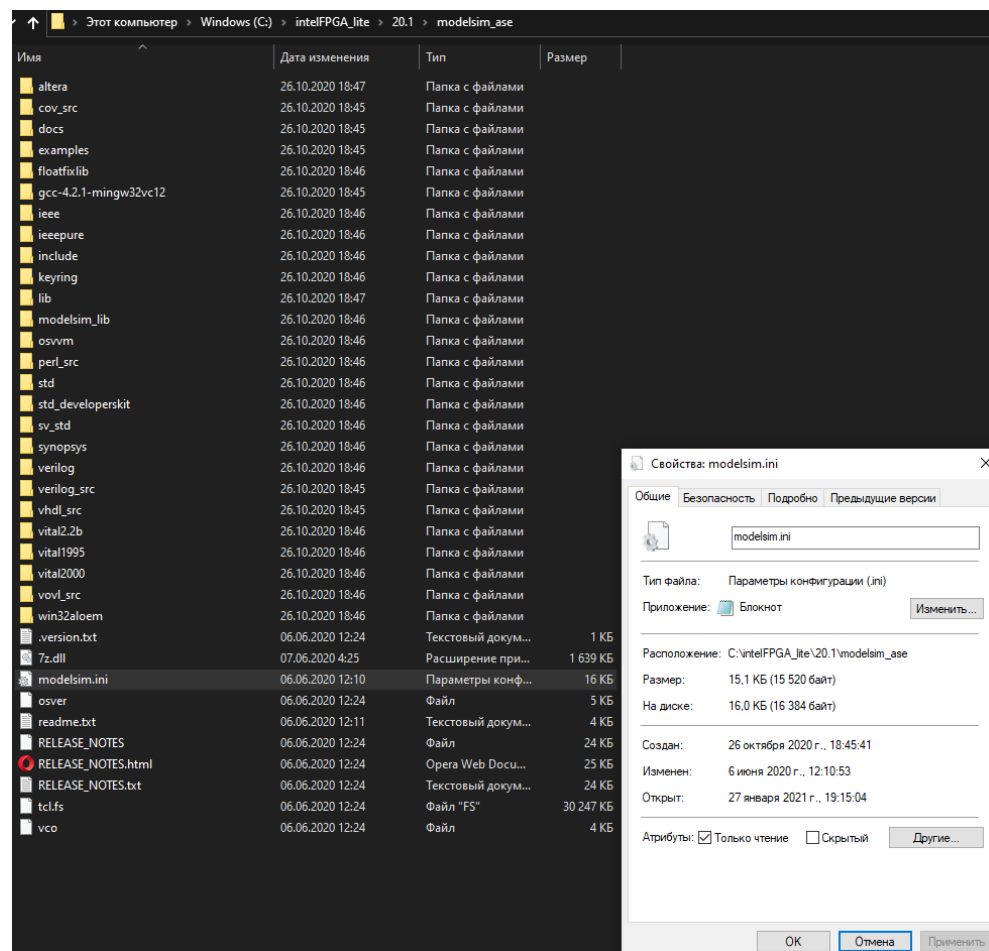


Figure 53

Creating the Resource Library

1. Start ModelSim Intel FPGA Started Edition.
2. Select **File** > **Change Directory** and change to the *C:\Intel_trn\ModelSim\Lab1_3\resource_library* directory you have just created.
3. Create the resource library.
 - a. Select **File** > **New** > **Library**.
 - b. Type **parts_lib** in the Library Name field (Figure 54). The Library Physical Name field is filled out automatically.

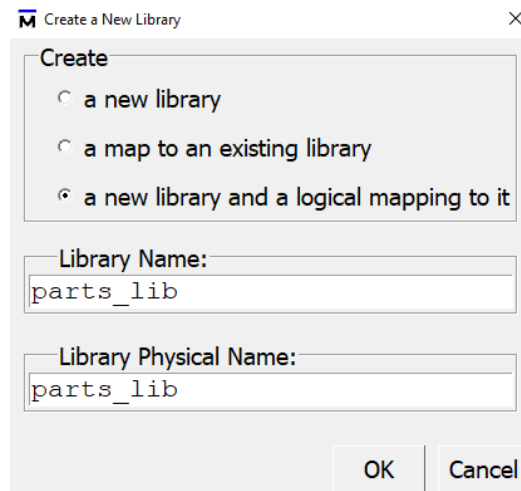


Figure 54

- c. Click **OK**.

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

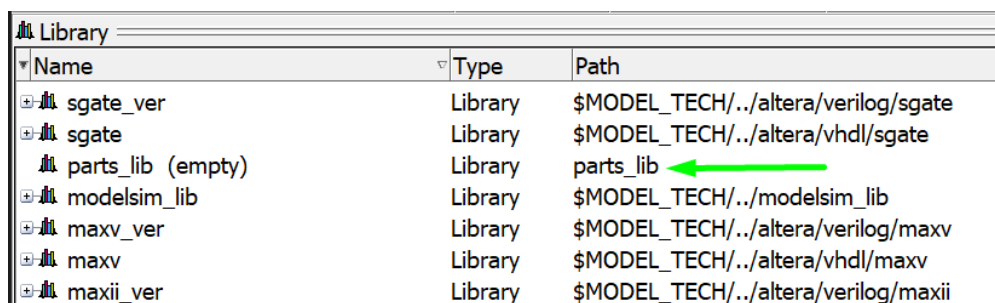


Figure 55

4. 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 56).
- c. Select *counter.v* to compile it.

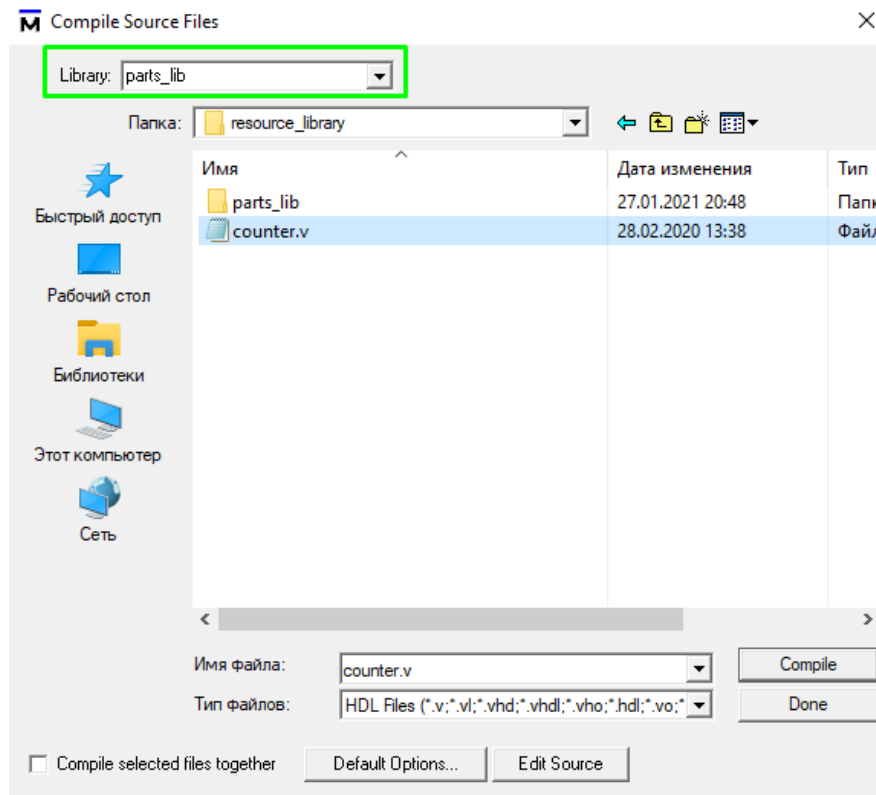


Figure 56

- c. Click **Compile**
- d. Click **Done**.

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

Library		
Name	Type	Path
sgate_ver	Library	\$MODEL_TECH/../../altera/verilog/sgate
sgate	Library	\$MODEL_TECH/../../altera/vhdl/sgate
parts_lib	Library	parts_lib
counter	Module	C:/Intel_trn/ModelSim/Lab1_3/resource_library/counter.v
modelsim_lib	Library	\$MODEL_TECH/../../modelsim_lib
maxv_ver	Library	\$MODEL_TECH/../../altera/verilog/maxv

Figure 57

Creating the Project

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

1. Create the project.
 - a. Select File > New > Project.
 - b. Type **counter** in the Project Name field.
 - c. Select, by using **Browse** button, a value for Project Location field as C:/Intel_trn/ModelSim/Lab1_3/testbench.
 - d. Do not change the Default Library Name field. (The default library name is *work*.)
 - e. Make sure "Copy Library Mappings" is selected. The default *modelsim.ini* file will be used.

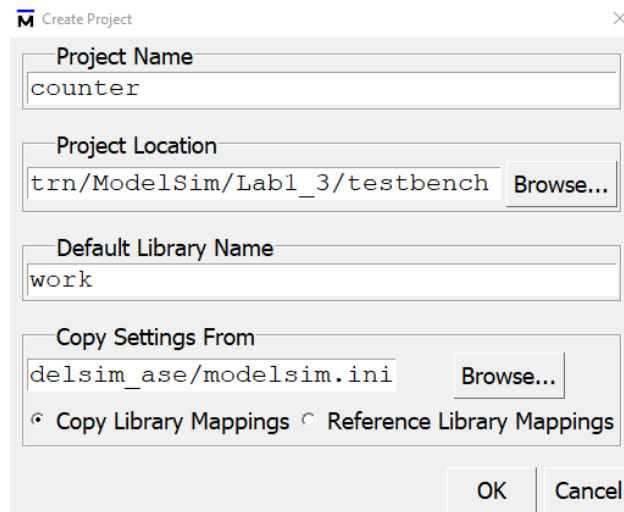


Figure 58

- f. Click **OK**.
2. Add the test bench to the project.
 - a. Click **Add Existing File** in the Add items to the Project dialog box.
 - b. Click the **Browse** button and select *tcounter.v* in the “Select files to add to project” dialog box.
 - c. Click **Open**.

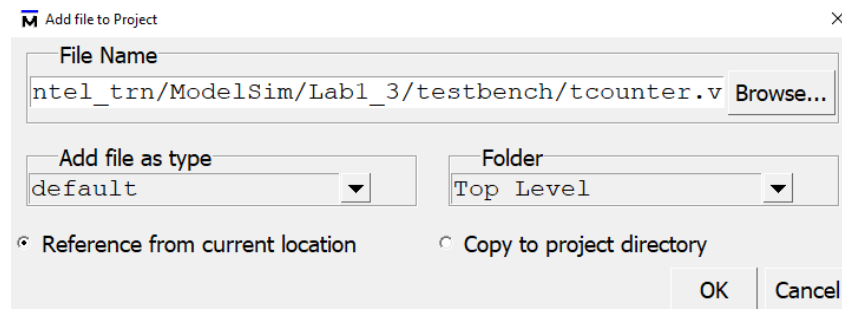


Figure 59

- d. Click **OK**.
 - e. Click **Close** to dismiss the “Add items to the Project” dialog box.
- The *tcounter.v* file is listed in the Project window.
3. Compile the test bench.
 - a. Right-click *tcounter.v* and select **Compile > Compile Selected**.

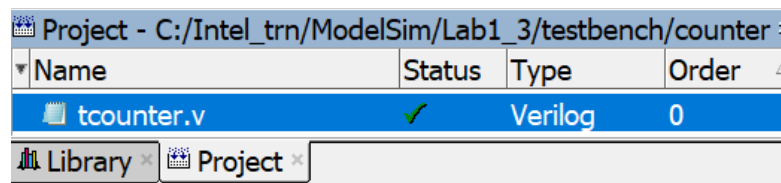


Figure 60

Loading Without Linking Libraries

To wrap up this part of the lab, you will link to the *parts_lib* library you created earlier.

But first, try loading the test bench without the link and see what happens. Now we will load the Verilog test bench into the simulator.

1. Load a Verilog design with a missing resource library.

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

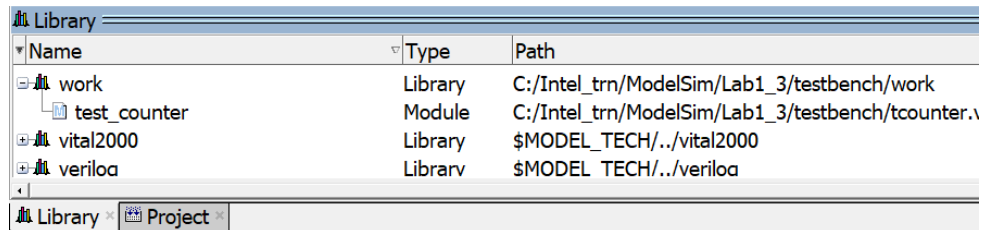


Figure 61

The Transcript reports an error (Figure 62).

When you see a message that contains text like "Error: (vsim-3033)", you can view more detail by using the **verror** command.

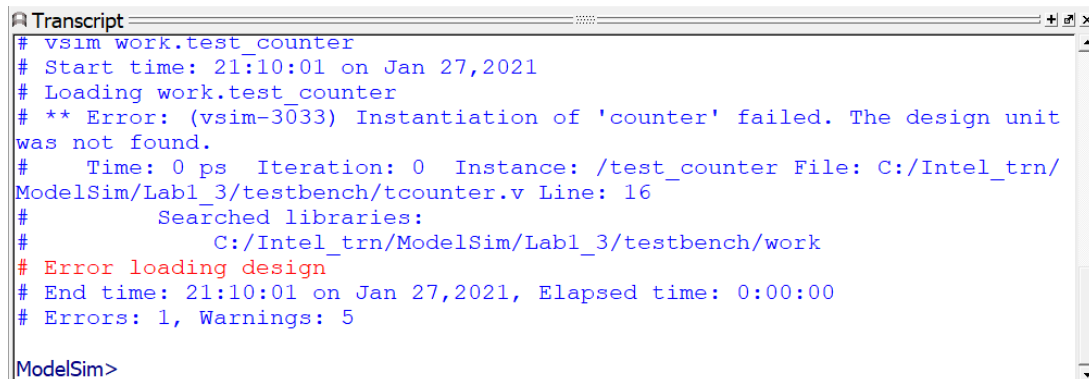
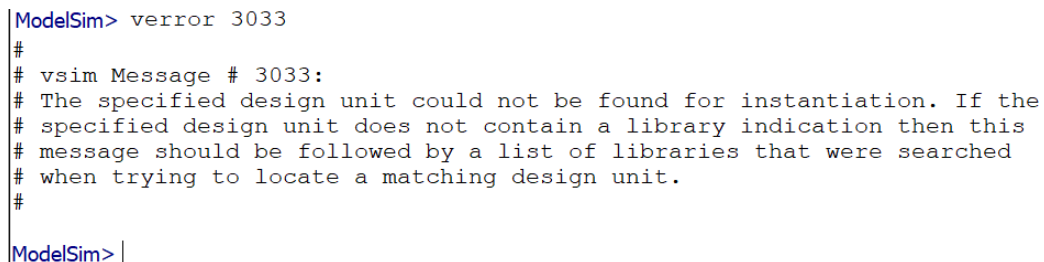


Figure 62

- 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*.

- c. Type **quit -sim** at the ModelSim> prompt to quit the simulation.

Linking to the Resource Library

Linking to a resource library 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 ***C:\Intel_trn\ModelSim\Lab1_3\resource_library\parts_lib*** directory you created earlier in the lab.
- e. Click **OK**.

The dialog box should have *parts_lib* listed in the Search Libraries field (Figure 63).

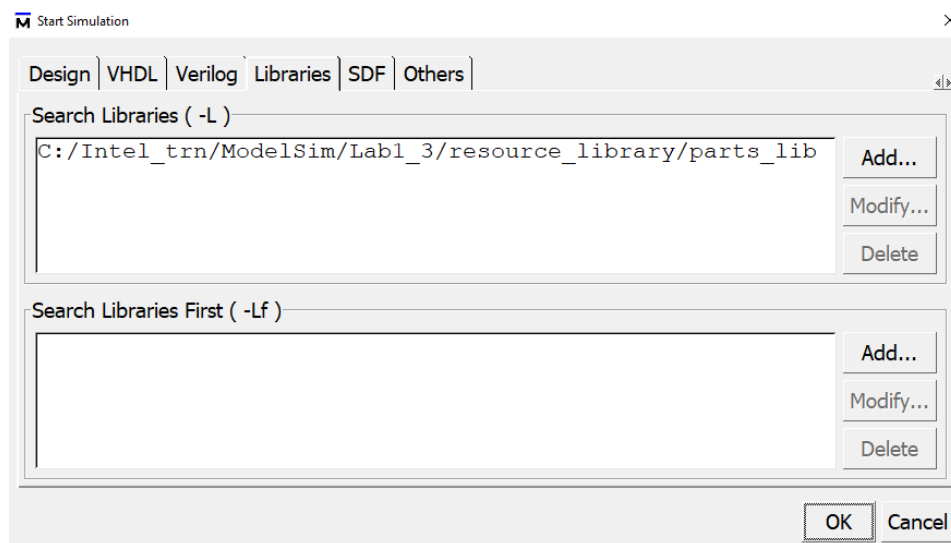


Figure 63

- f. Click OK. The design loads without errors.

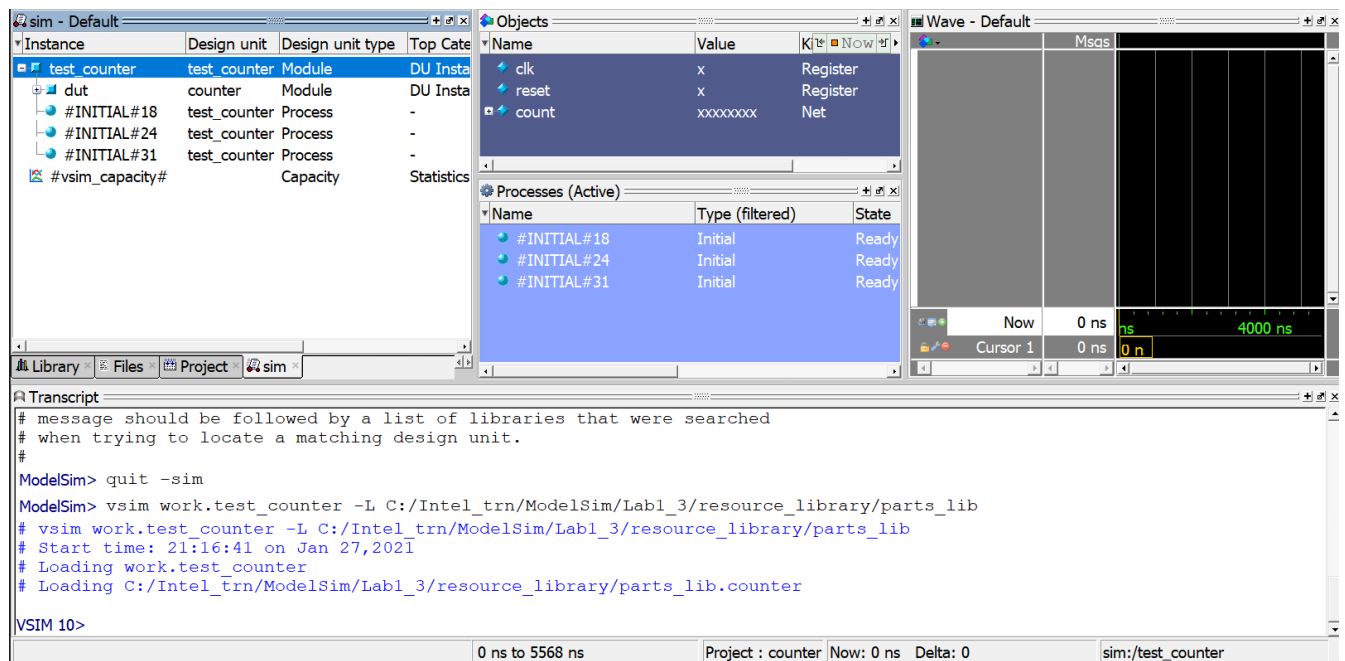


Figure 64

Concluding the Lab

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

1. Select Simulate > End Simulation.
2. Click Yes.
3. Select the Project window to make it active.
4. Select **File > Close Project**.
5. Click Yes.

Lab1_4: Automating Simulation

Aside from executing a couple of pre-existing DO files, the previous lessons focused on using ModelSim in interactive mode: executing single commands, one after another, via the GUI menus or Main window command line. In situations where you have repetitive tasks to complete, you can increase your productivity with DO files.

DO files are scripts that allow you to execute many commands at once. The scripts can be as simple as a series of ModelSim commands with associated arguments, or they can be full-blown Tcl programs with variables, conditional execution, and so forth. You can execute DO files from within the GUI or you can run them from the system command prompt without ever invoking the GUI.

Design Files

Example files are located in the **C:\Intel_trn\ModelSim\Lab1_4** folder. Which is a working folder for the Lab1_4.

Creating a Simple DO File

Creating a DO file is as simple as typing a set of commands in a text file. In this lab, you will create a DO file that loads a design, adds signals to the Wave window, provides stimulus to those signals, and then advances the simulation. You can also create a DO file from a saved transcript file.

1. Start ModelSim Intel FPGA Started Edition.
2. Change to the directory **C:\Intel_trn\ModelSim\Lab1_1** you used in the Lab1_1.
3. Create a DO file that will add signals to the Wave window, force signals, and run the simulation.
 - a. Select **File > New > Source > Do** to create a new DO file.
 - b. Enter the following commands into the Source window:

```
vsim test_counter  
add wave count  
add wave clk  
add wave reset  
force -freeze clk 0 0, 1 {50 ns} -r 100 ns  
force reset 1  
run 100 ns  
force reset 0  
run 300 ns  
force reset 1  
run 400 ns  
force reset 0  
run 200 ns
```



```

C:/Intel_trn/ModelSim/Lab1_1/sim.do - Default
Ln#
1 vsim test_counter
2 add wave count
3 add wave clk
4 add wave reset
5 force -freeze clk 0 0, 1 {50 ns} -r 100 ns
6 force reset 1
7 run 100 ns
8 force reset 0
9 run 300 ns
10 force reset 1
11 run 400 ns
12 force reset 0
13 run 200ns

```

Figure 65

4. Save the file.
 - a. Select File > Save As.
 - b. Type **sim.do** in the File name: field and save it to the current directory.
5. Execute the DO file.
 - a. Enter **do sim.do** at the VSIM> prompt.

ModelSim loads the design, executes the saved commands and draws the waves in the Wave window. (Figure 66)

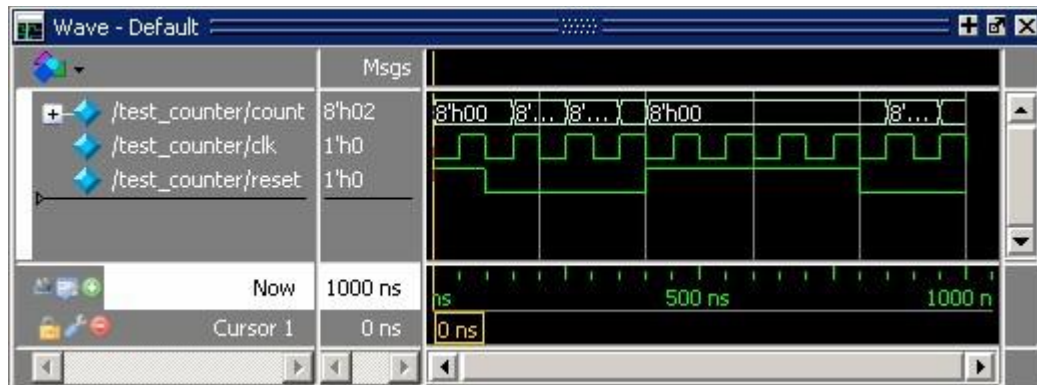


Figure 66

6. When you done, select **File > Quit** to quit ModelSim.

Running in Command-Line Mode

We use the term "command-line mode" to refer to simulations that are run from a system command prompt without invoking the GUI.

Several ModelSim commands (e.g., vsim, vlib, vlog, etc.) are actually stand-alone executables that can be invoked at the system command prompt.

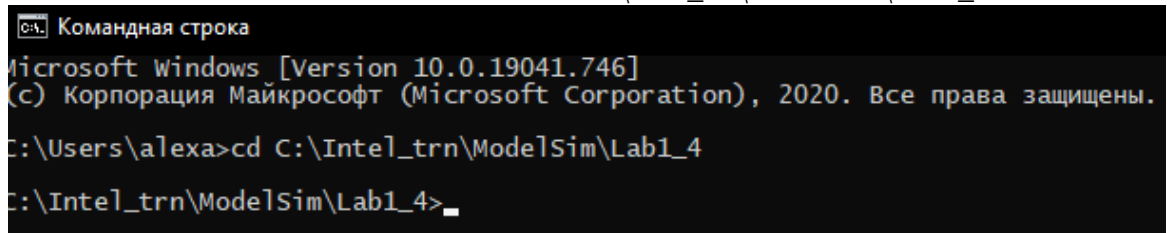
Additionally, you can create a DO file that contains other ModelSim commands and specify a file when you invoke the simulator.

This lab uses the Verilog file *counter.v* and *stim.do* file, which are in folder C:\Intel_trn\ModelSim\Lab1_4.

1. In the Search field, near Start button, enter cmd and start Command line window.



2. In command line window enter command: `cd C:\Intel_trn\ModelSim\Lab1_4`



```
Командная строка
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\alexa>cd C:\Intel_trn\ModelSim\Lab1_4
C:\Intel_trn\ModelSim\Lab1_4>
```

Figure 67

3. Create a new design library and compile the source file.
a. Type **vlib work** at the command line prompt.

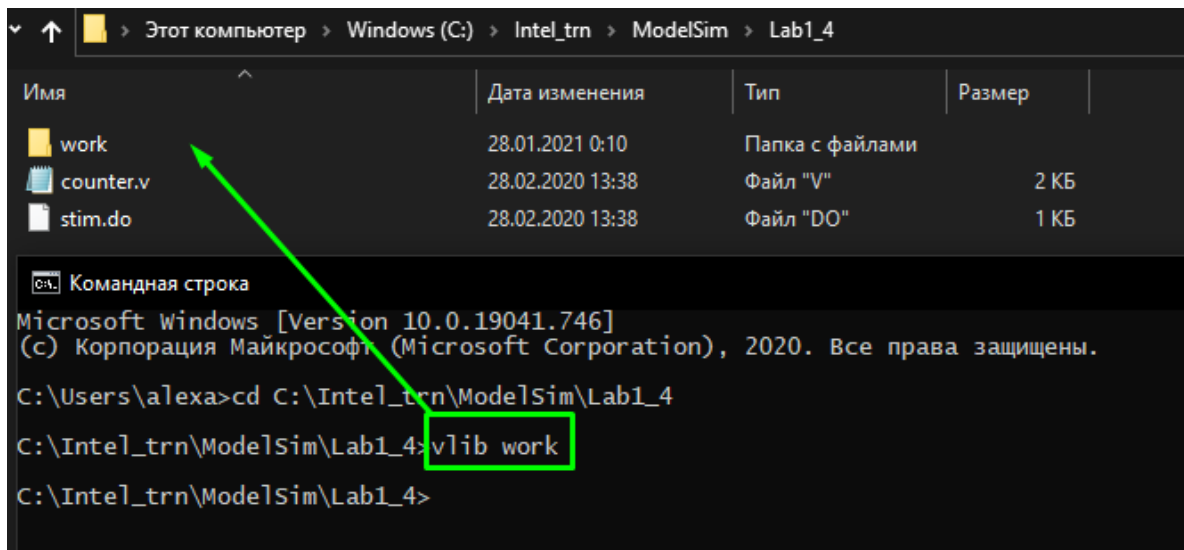
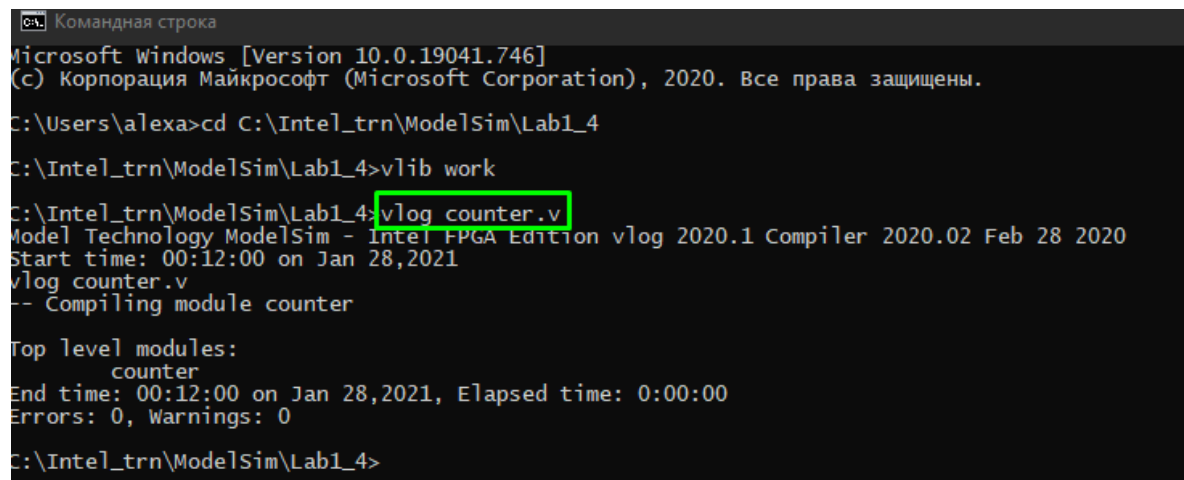


Figure 68

- b. For Verilog, type **vlog counter.v** at the command line prompt.



```
Командная строка
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\alexa>cd C:\Intel_trn\ModelSim\Lab1_4
C:\Intel_trn\ModelSim\Lab1_4>vlib work
C:\Intel_trn\ModelSim\Lab1_4>vlog counter.v
Model Technology ModelSim - Intel FPGA Edition vlog 2020.1 Compiler 2020.02 Feb 28 2020
Start time: 00:12:00 on Jan 28,2021
vlog counter.v
-- Compiling module counter

Top level modules:
    counter
End time: 00:12:00 on Jan 28,2021, Elapsed time: 0:00:00
Errors: 0, Warnings: 0

C:\Intel_trn\ModelSim\Lab1_4>
```

Figure 69

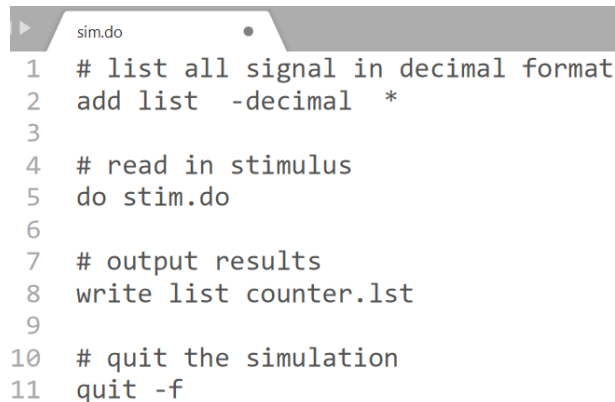
4. Create a DO file.
a. Open a text editor (you can use any text editor of your choice).
b. Type the following lines into a new file:

```

# list all signals in decimal format
add list -decimal *
# read in stimulus
do stim.do
# output results
write list counter.lst
# quit the simulation
quit -f

```

- c. Save the file with the name *sim.do* and place it in the current **C:\Intel_trn\ModelSim\Lab1_4** directory.



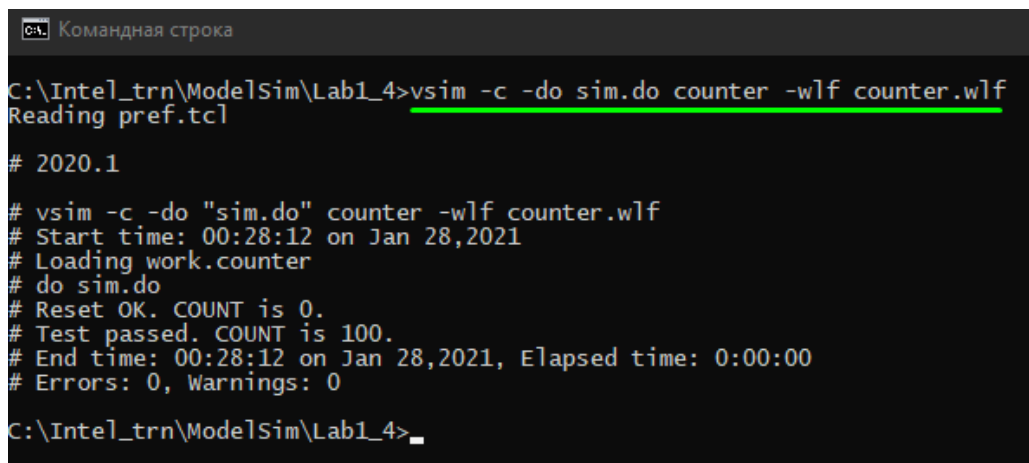
```

sim.do
1  # list all signal in decimal format
2  add list -decimal *
3
4  # read in stimulus
5  do stim.do
6
7  # output results
8  write list counter.lst
9
10 # quit the simulation
11 quit -f

```

Figure 70

5. Run the command line mode simulation.
- a. Enter the following command at the DOS/UNIX prompt:
- ```
vsim -c -do sim.do counter -wlf counter.wlf
```



```

C:\Intel_trn\ModelSim\Lab1_4>vsim -c -do sim.do counter -wlf counter.wlf
Reading pref.tcl
2020.1

vsim -c -do "sim.do" counter -wlf counter.wlf
Start time: 00:28:12 on Jan 28,2021
Loading work.counter
do sim.do
Reset OK. COUNT is 0.
Test passed. COUNT is 100.
End time: 00:28:12 on Jan 28,2021, Elapsed time: 0:00:00
Errors: 0, Warnings: 0

C:\Intel_trn\ModelSim\Lab1_4>

```

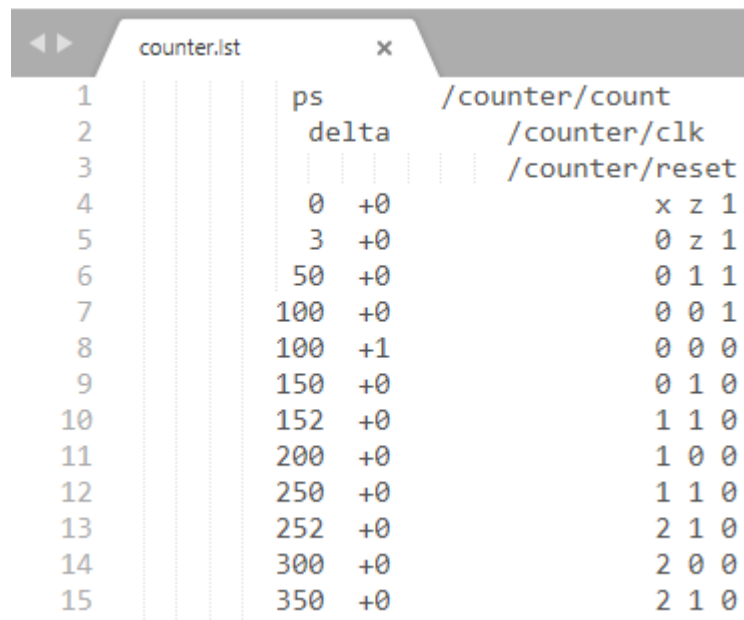
**Figure 71**

The **-c** argument instructs ModelSim not to invoke the GUI.

The **-wlf** argument saves the simulation results in a WLF file. This allows you to view the simulation results in the GUI for debugging purposes.

6. View the list output.

- a. Open *counter.lst* and view the simulation results. Output produced by the Verilog version of the design should look like Figure 72:



|    | ps    | /counter/count | /counter/clk | /counter/reset |
|----|-------|----------------|--------------|----------------|
| 1  |       |                |              |                |
| 2  | delta |                |              |                |
| 3  |       |                |              |                |
| 4  | 0     | +0             | x z 1        |                |
| 5  | 3     | +0             | 0 z 1        |                |
| 6  | 50    | +0             | 0 1 1        |                |
| 7  | 100   | +0             | 0 0 1        |                |
| 8  | 100   | +1             | 0 0 0        |                |
| 9  | 150   | +0             | 0 1 0        |                |
| 10 | 152   | +0             | 1 1 0        |                |
| 11 | 200   | +0             | 1 0 0        |                |
| 12 | 250   | +0             | 1 1 0        |                |
| 13 | 252   | +0             | 2 1 0        |                |
| 14 | 300   | +0             | 2 0 0        |                |
| 15 | 350   | +0             | 2 1 0        |                |

Figure 72

6. View the results in the GUI.

Since you saved the simulation results in *counter.wlf*, you can view them in the GUI by invoking VSIM with the **-view** argument.

- a. Type **vsim -view counter.wlf** at the prompt.

The GUI opens and a dataset tab named "counter" is displayed (Figure 73).

- b. Right-click the *counter* instance and select **Add Wave**.

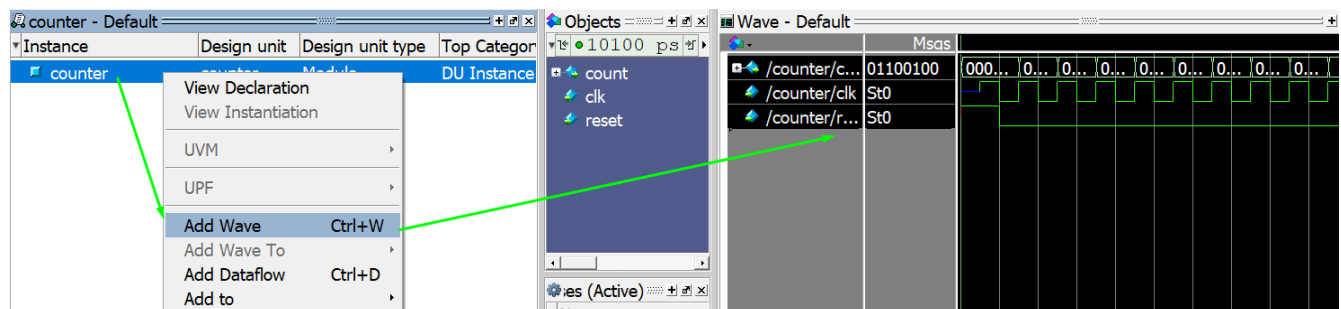


Figure 73

The waveforms display in the Wave window.

7. When you finish viewing the results, select **File > Quit** to close ModelSim.

## Running Tcl script

The DO files used in previous steps contained only ModelSim commands. However, DO files are just Tcl scripts. This means you can include a whole variety of Tcl constructs such as procedures, conditional operators, math and trig functions, regular expressions, and so forth.

In this exercise, you create a simple Tcl script that tests for certain values on a signal and then adds bookmarks that zoom the Wave window when that value exists. Bookmarks allow you to save a particular zoom range and scroll position in the Wave window.

1. Create the script.

- a. In a text editor, create a new file and enter the following lines:

```
proc add_wave_zoom {stime num} {
 echo "Bookmarking wave. Bookmark is bk$num"
 bookmark add wave "bk$num" "[expr $stime - 100] [expr $stime + 50]" 0
}
```

These commands do the following:

- Create a new procedure called "add\_wave\_zoom" that has two arguments, *stime* and *num*.
- Create a bookmark with a zoom range: from the current simulation time minus 100 time units, to the current simulation time plus 50 time units.

- b. Now add these lines to the bottom of the script:

```
add wave -decimal count
add wave clk
add wave reset
```

These commands do the following:

- Add signal count to the Wave window in decimal radix.
- Add clk and reset signals to the Wave window.

- c. Now add these lines to the bottom of the script:

```
when {clk'event and clk="1"} {
 if {[examine -decimal count]== "27"} {
 echo "Count is [exa -decimal count]"
 add_wave_zoom $now 1
 } elseif {[examine -decimal count]== "47"} {
 echo "Count is [exa -decimal count]"
 add_wave_zoom $now 2
 }
}
```

These commands do the following:

- Use a when statement to identify when *clk* transitions to 1.
- Examine the value of *count* at those transitions and add a bookmark if it is a certain value.

- d. Save the script with the name *add\_bkmrk.do* into **C:\Intel\_trn\ModelSim\Lab1\_1** directory.

```

add_bkmrk.do
1 proc add_wave_zoom {stime num} {
2 echo "Bookmarking wave. Bookmark is bk$num"
3 bookmark add wave "bk$num" "[expr $stime - 100] [expr $stime + 50]" 0
4 }
5
6 add wave -decimal count
7 add wave clk
8 add wave reset
9
10 when {clk'event and clk="1"} {
11 if {[examine -decimal count]== "27"} {
12 echo "Count is [exa -decimal count]"
13 add_wave_zoom $now 1
14 } elseif {[examine -decimal count]== "47"} {
15 echo "Count is [exa -decimal count]"
16 add_wave_zoom $now 2
17 }
18 }

```

**Figure 74**

2. Load the *test\_counter* design unit and make sure the radix is set to binary.
  - a. Start ModelSim Intel FPGA Started Edition Run
  - b. Change to the directory **C:\Intel\_trn\ModelSim\Lab1\_1** you used in the Lab1\_1.
  - c. Enter the following command at the ModelSim> prompt: **vsim test\_counter**
3. Execute the DO file and run the design.
  - a. Type **do add\_bkmrk.do** at the VSIM> prompt.
  - b. Type **run 1000 ns** at the VSIM> prompt.

The simulation runs and the DO file creates two bookmarks.

4. Check Transcript window and find the following lines.
 

```

540 0 0 27
Count is 27
Bookmarking wave. Bookmark is bk1
550 0 1 27

...

940 0 0 47
Count is 47
Bookmarking wave. Bookmark is bk2
950 0 1 47

```

**Figure 75**

5. Navigate to the Bookmarks created
  - a. Undock the Wave window.
  - b. Zoom Full the Wave window.
  - c. Be sure you can see count, clk, reset signals.
  - d. Click anywhere in the Wave window
  - e. Select **Bookmarks** menu and then **bk1 450->600**

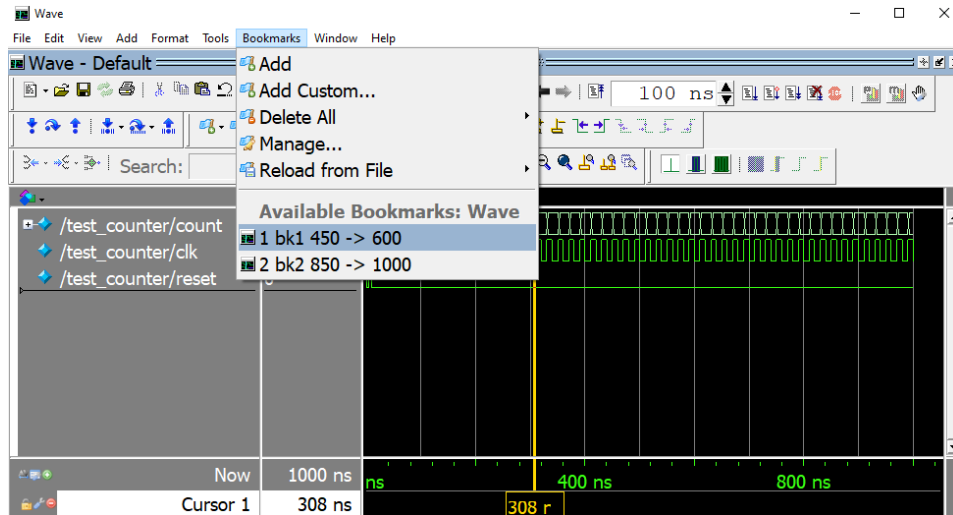


Figure 76

- f. Select **Bookmarks** menu and then **bk1 450->600**
- g. Check that Wave window is Zoomed to **count** value 27 (decimal) and you see waveform of current simulation time minus 100 time units, to the current simulation time plus 50 time units

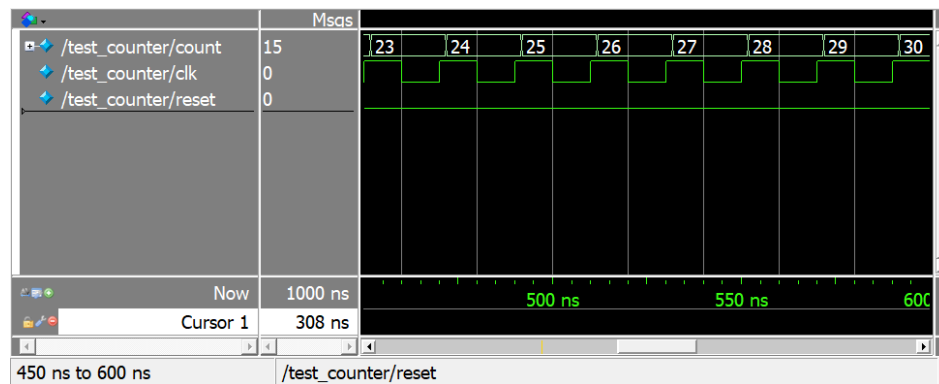


Figure 77

- h. Select **Bookmarks** menu and then **bk2 850->1000**
- i. Check that Wave window is Zoomed to **count** value 47 (decimal) and you see waveform of current simulation time minus 100 time units, to the current simulation time plus 50 time units

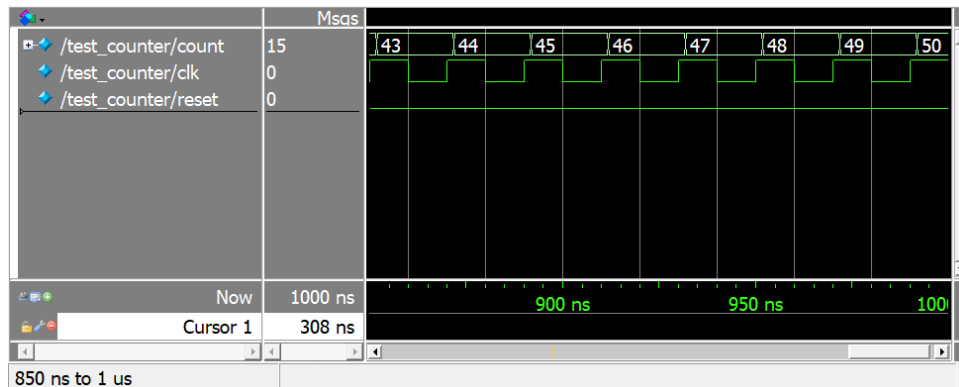


Figure 78

## Concluding the Lab

This concludes this Lab.

1. Select **Simulate > End Simulation**.
2. Click **Yes**.
3. Select **File > Quit** to close ModelSim.
4. In the folder **C:\Intel\_trn\ModelSim\Lab1\_1** find **bookmarks.do** file.  
The file contains bookmarks created during the last tool invocation. The bookmarks will be automatically loaded during the next start of simulation. If you don't like loading the previously created bookmarks, delete the file **bookmarks.do**.



# Lab1\_5: Viewing and Initializing Memories

In this lab, you will learn how to view and initialize memories.

ModelSim defines and lists any of the following as memories:

- reg, wire arrays
- Integer arrays

## Design Files

Example files (*ram\_tb.v*; *sp\_syn\_ram.v*; *dp\_syn\_ram.v*) are located in the **C:\Intel\_trn\ModelSim\Lab1\_5** folder. This is a working folder for the Lab1\_5.

## Compile and Load the Design

Before viewing and initializing memories, we need to compile and load a design.

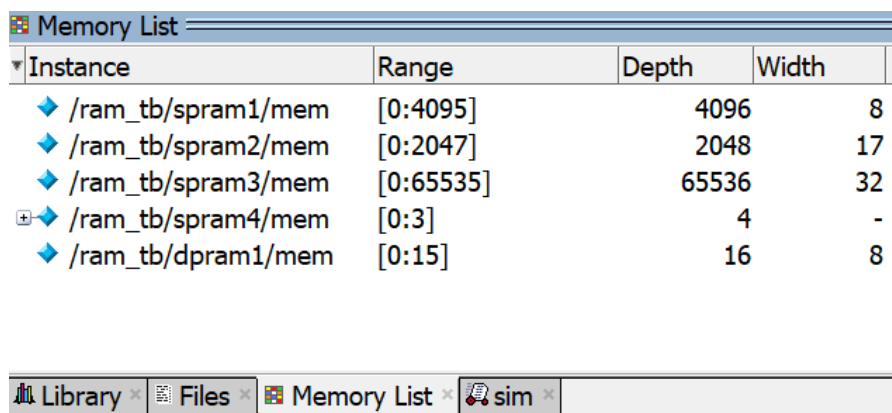
1. Start ModelSim Intel FPGA Started Edition.
2. Change directory (**File > Change Directory**) to **C:\Intel\_trn\ModelSim\Lab1\_5**
3. Create the working library and compile all files in the design.
  - a. Type **vlib work** at the *ModelSim>* prompt.
  - b. Type **vlog \*.v** at the *ModelSim>* prompt to compile all Verilog files in the design.
4. Load the design.
  - a. On the **Library** tab of the **Main** window Workspace, click the "+" icon next to the *work* library.
  - b. Double-click the *ram\_tb* design unit to load the design.

## View a Memory and its Contents

The Memory List window lists all memory instances in the design, showing for each instance the range, depth, and width. Double-clicking an instance opens a window displaying the memory data.

1. Open the Memory List window and view the data of a memory instance
  - a. If the Memory List window is not already open, select **View > Memory List**.

A Memory List window is shown in Figure 79. View a Memory and its Contents



| Instance              | Range     | Depth | Width |
|-----------------------|-----------|-------|-------|
| ♦ /ram_tb/spram1/mem  | [0:4095]  | 4096  | 8     |
| ♦ /ram_tb/spram2/mem  | [0:2047]  | 2048  | 17    |
| ♦ /ram_tb/spram3/mem  | [0:65535] | 65536 | 32    |
| ♦+ /ram_tb/spram4/mem | [0:3]     | 4     | -     |
| ♦ /ram_tb/dpram1/mem  | [0:15]    | 16    | 8     |

Figure 79

- b. Double-click the `/ram_tb/spram1/mem` instance in the memory list to view its contents.

A Memory Data window opens (Figure 80) displaying the contents of `spram1`. The first column (blue hex characters) lists the addresses, and the remaining columns show the data values.

The data is all **X** (Figure 80) because you have not yet simulated the design.

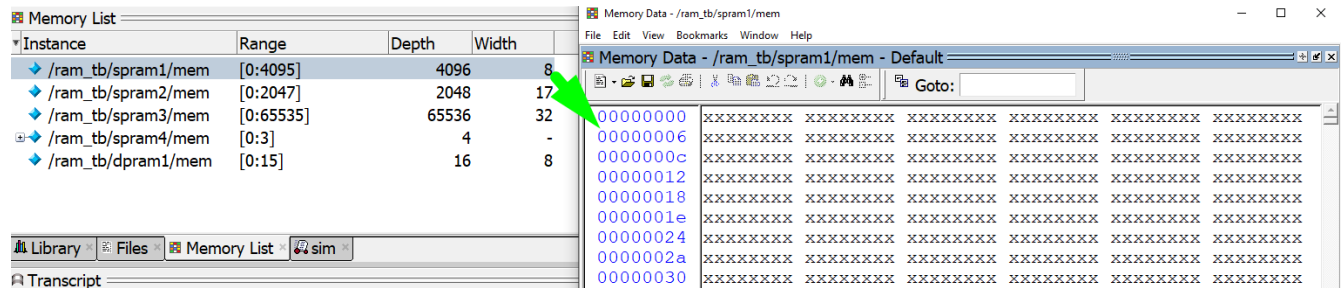


Figure 80

- c. Double-click the instance `/ram_tb/spram2/mem` in the Memory List window. This opens a second Memory Data window that contains the addresses and data for the `spram2` instance. For each memory instance that you click in the Memory List window, a new Memory Data window opens.
2. Simulate the design.

- a. Click the **run -all** icon in the Main window.



A Source window opens showing the source code for the `ram_tb` file at the point where the simulation stopped.

3. Open the Memory List window and view the data of a memory instance after simulation
  - a. Click the **Memory ...spram1/mem** tab to bring that Memory data window to the foreground. The data fields are shown in Figure 81.

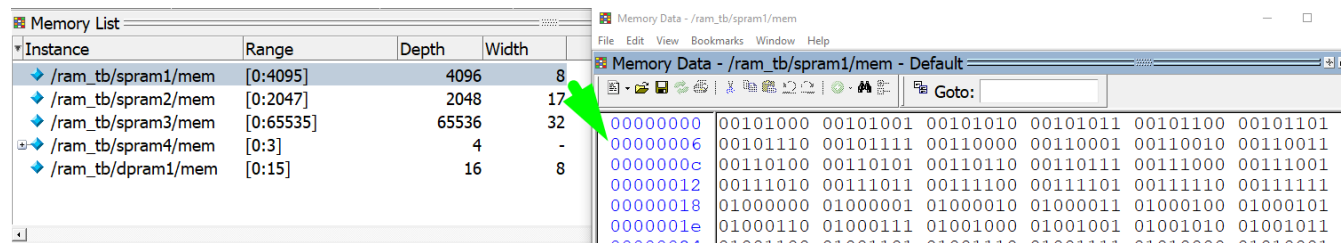
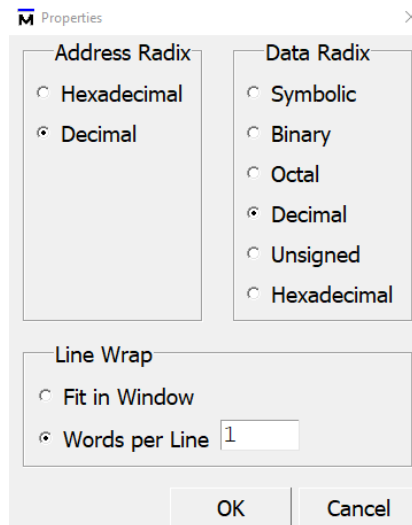


Figure 81

4. Change the address radix and the number of words per line for instance `/ram_tb/spram1/mem`.
  - a. Right-click anywhere in the `spram1` Memory Data window and select **Properties**.
  - b. The Properties dialog box opens (Figure 82).
  - c. For the **Address Radix**, select **Decimal**. This changes the radix for the addresses only.
  - d. Change the Data Radix to **Decimal**.
  - e. Select **Words per line** and type **1** in the field.



**Figure 82**

- c. Click OK.

You can see the results of the settings in Figure 83.

If the figure does not match what you have in your ModelSim session, check to make sure you set the Address Radix and the Data Radix as decimal.

|   |    |
|---|----|
| 0 | 40 |
| 1 | 41 |
| 2 | 42 |
| 3 | 43 |
| 4 | 44 |
| 5 | 45 |
| 6 | 46 |
| 7 | 47 |

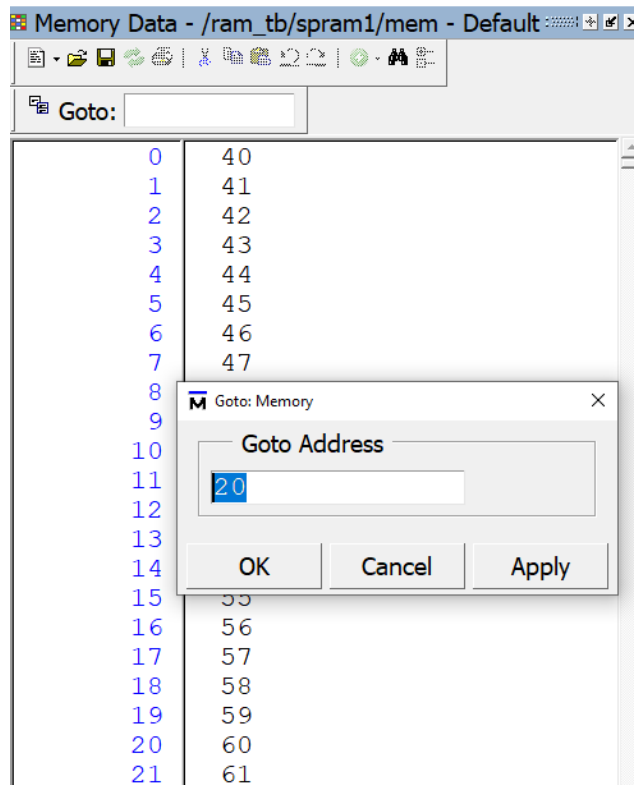
**Figure 83**

## Navigate Within the Memory

You can navigate to specific memory address locations, or to locations containing particular data patterns. First, you will go to a specific address.

1. Use Goto to find a specific address.
  - a. Right-click anywhere in address column and select **Goto** (Figure 84).

The Goto dialog box opens in the data pane.



**Figure 84**

- b. Type **20** in the Goto Address field.
- c. Click **OK**.

The requested address appears in the top line of the window.

2. To quickly move to a particular address edit the address location directly.
  - a. Double click address 20 in the address column.
  - b. Enter address 30 (Figure 85).

|    |    |
|----|----|
| 17 | 57 |
| 18 | 58 |
| 19 | 59 |
| 30 | 60 |
| 21 | 61 |
| 22 | 62 |
| 23 | 63 |
| 24 | 64 |
| 25 | 65 |
| 26 | 66 |
| 27 | 67 |
| 28 | 68 |
| 29 | 69 |
| 30 | 70 |
| 31 | 71 |

**Figure 85**

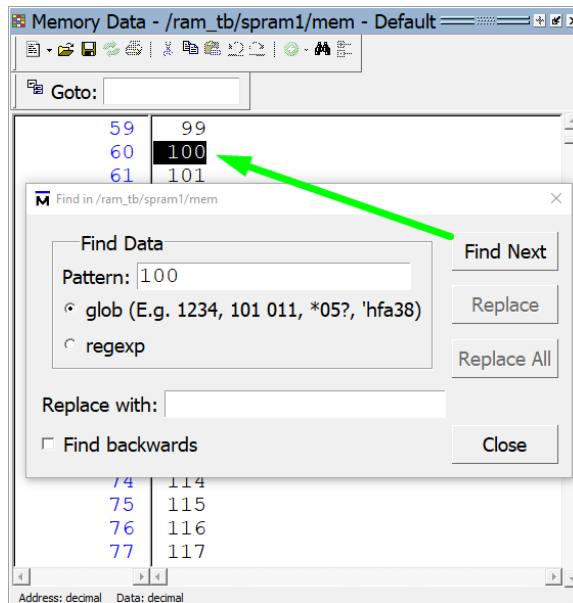
- c. Press the Enter or Return key on your keyboard.

The pane jumps to address 30.

3. Now, let us find a particular data entry.
  - a. Right-click anywhere in the data column and select **Find**.

The **Find in** dialog box opens (Figure 86).

- b. Type 100 in the **Find data field** and click **Find Next**.



**Figure 86**

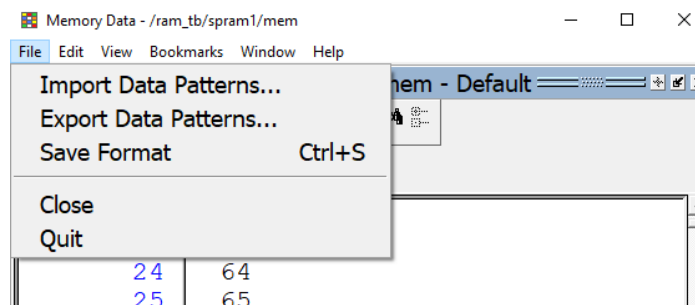
The data scrolls to an address of the first occurrence of that data value. Click **Find Next** a few more times to search through the list and check address field for each occurrence of that data value.

- c. Click **Close** to close the dialog box.

## Export Memory Data to a File

You can save memory data to a file that can be loaded at some later point in the simulation.

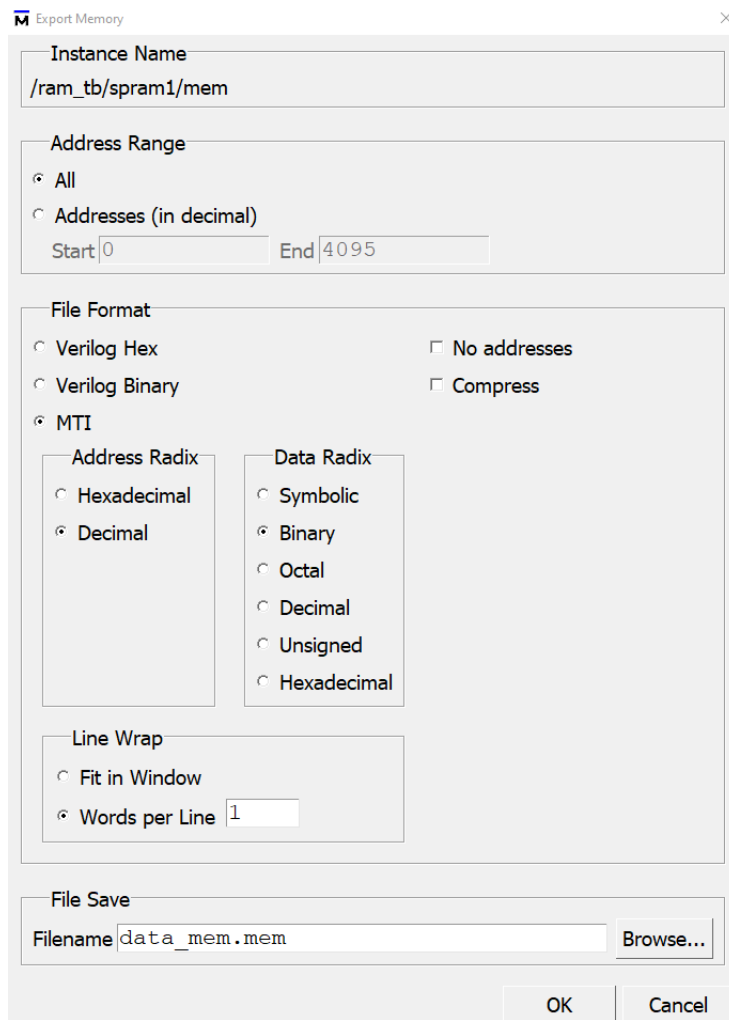
1. Export a memory pattern from the `/ram_tb/spram1/mem` instance to a file.
  - a. Make sure `/ram_tb/spram1/mem` is open and selected.
  - b. Select **File > Export Data Patterns**



**Figure 87**

This brings up the Export Memory dialog box (Figure 88).

- c. For the Address Radix, select **Decimal**.
- d. For the Data Radix, select **Binary**.
- e. For the **Words per Line**, set to 1.
- f. Type `data_mem.mem` into the **Filename** field.



The 'Export Memory' dialog box is shown with the following settings:

- Instance Name:** /ram\_tb/spram1/mem
- Address Range:**
  - ☒ All
  - ☐ Addresses (in decimal)
    - Start: 0
    - End: 4095
- File Format:**
  - ☐ Verilog Hex ☐ No addresses
  - ☐ Verilog Binary ☐ Compress
  - ☒ MTI
 

**Address Radix**
    - ☐ Hexadecimal
    - ☒ Decimal

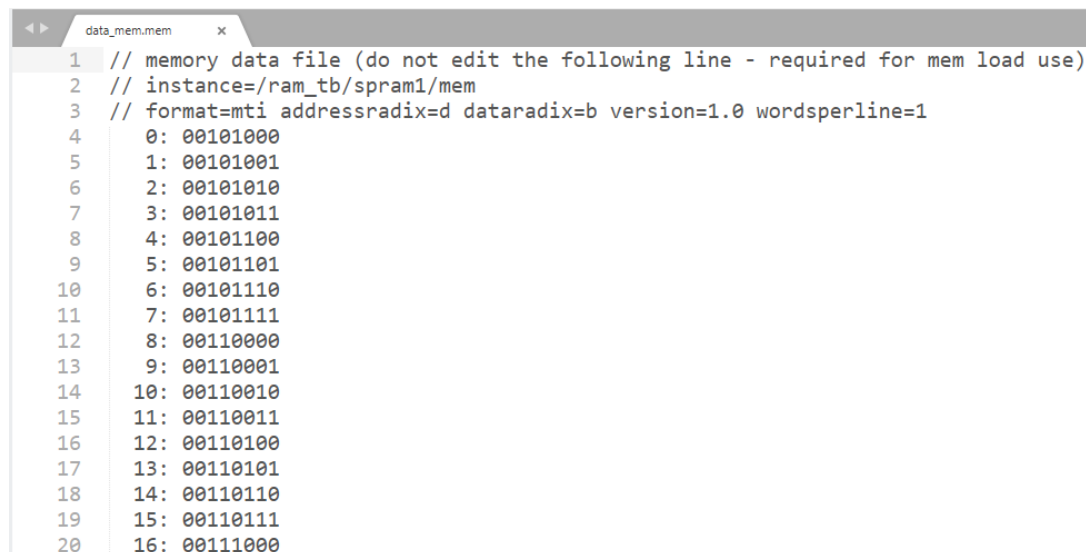
**Data Radix**
    - ☐ Symbolic
    - ☒ Binary
    - ☐ Octal
    - ☐ Decimal
    - ☐ Unsigned
    - ☐ Hexadecimal
- Line Wrap:**
  - ☐ Fit in Window
  - ☒ Words per Line: 1
- File Save:**
  - Filename: data\_mem.mem
  - Browse...

Buttons: OK, Cancel

**Figure 88**

g. Click OK.

You can view the exported file in any editor.



```

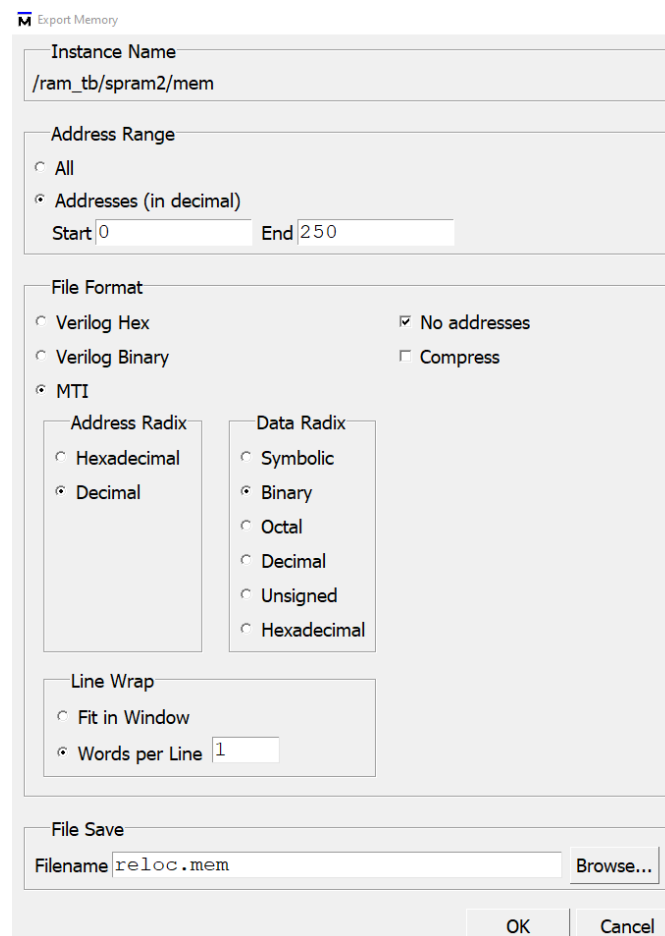
1 // memory data file (do not edit the following line - required for mem load use)
2 // instance=/ram_tb/spram1/mem
3 // format=mti addressradix=d dataradix=b version=1.0 wordsperline=1
4 0: 00101000
5 1: 00101001
6 2: 00101010
7 3: 00101011
8 4: 00101100
9 5: 00101101
10 6: 00101110
11 7: 00101111
12 8: 00110000
13 9: 00110001
14 10: 00110010
15 11: 00110011
16 12: 00110100
17 13: 00110101
18 14: 00110110
19 15: 00110111
20 16: 00111000

```

**Figure 89**

Memory pattern files can be exported as relocatable files, simply by leaving out the address information (No addresses checkpoint in [Figure 7-12](#)). Relocatable memory files can be loaded anywhere in a memory because no addresses are specified.

2. Export a relocatable memory pattern file from the `/ram_tb/spram2/mem` instance.
  - a. Select the Memory Data window for the `/ram_tb/spram2/mem` instance.
  - b. Right-click on the memory contents to open a popup menu and select **Properties**.
  - c. In the Properties dialog box, set the Address Radix to **Decimal**; the Data Radix to **Binary**; and the **Line Wrap** to **1 Words per Line**. Click **OK** to accept the changes and close the dialog box.
  - d. Select **File > Export Data Pattern** to bring up the Export Memory dialog box.
  - e. For the Address Range, specify a Start address of **0** and End address of **250**.
  - f. For the File Format, select **MTI** and **No addresses** to create a memory pattern that you can use to relocate somewhere else in the memory, or in another memory.
  - g. For Address Radix select **Decimal**, and for Data Radix select **Binary**.
  - h. For the **Words per Line**, set to 1.
  - i. Enter the file name as **reloc.mem**,



**Figure 90**

- j. Click **OK** to save the memory contents and close the dialog box. You will use this file for initialization in the next section.

## Initialize a Memory

In ModelSim, it is possible to initialize a memory using one of three methods: from an exported memory file, from a fill pattern, or from both.

First, let us initialize a memory from a file only.

You will use the one you exported previously: *data\_mem.mem*.

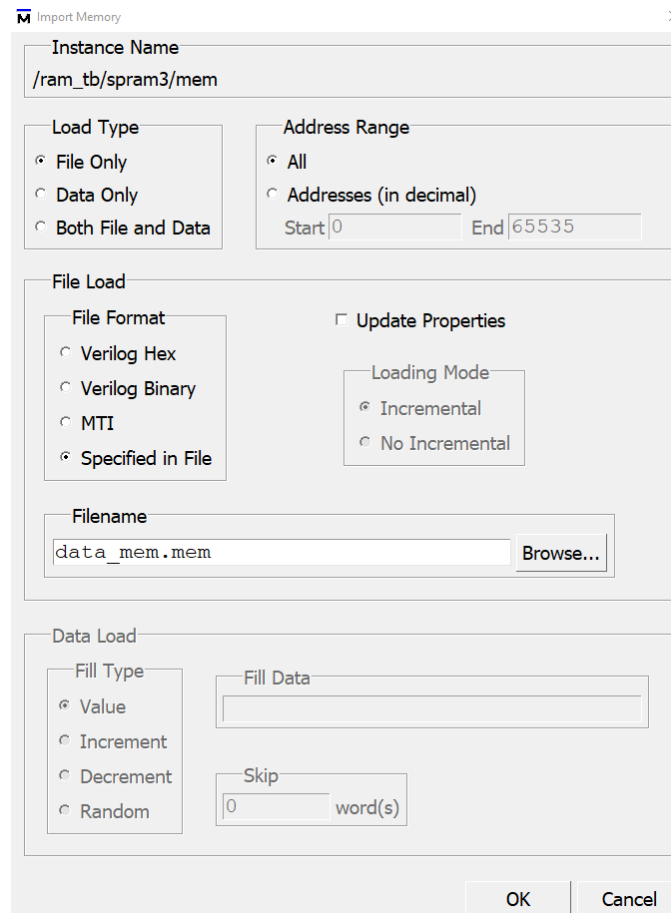
1. View instance */ram\_tb/spram3/mem*.
  - a. Double-click the */ram\_tb/spram3/mem* instance in the Memory List window.

This will open a new Memory Data window to display the contents of */ram\_tb/spram3/mem*.

- b. Right-click and select **Properties** to bring up the Properties dialog box.
  - c. Change the Address Radix to **Decimal**, Data Radix to **Binary**, **Words per Line to 1**, and click OK.

Familiarize yourself with the contents so you can identify changes once the initialization is complete.

2. Initialize *spram3* from a file.
  - a. Right-click anywhere in the data column and select **Import Data Patterns** to bring up the Import Memory dialog box (Figure 91).
  - b. For Load Type, select **File Only**.
  - c. Type *data\_mem.mem* in the Filename field.



**Figure 91**

- d. Click **OK**.



The addresses (from 0 up to 4095) in instance `/ram_tb/spram3/mem` are updated with the data from `data_mem.mem`, which has only 4096 address entries.

3. Import the `/ram_tb/spram3/mem` instance with a relocatable memory pattern (`reloc.mem`) and a fill pattern.
  - a. Right-click in the data column of `spram3` and select **Import Data Patterns** to bring up the Import Memory dialog box.
  - b. For Load Type, select **Both File and Data**.
  - c. For Address Range, select **Addresses** and enter **0** as the Start address and **300** as the End address.

- d. For File Load, select the MTI File Format and enter **reloc.mem** in the Filename field.
- e. For Data Load, select a Fill Type of **Increment**.
- f. In the Fill Data field, set the seed value of **0** for the incrementing data.
- g. Click **OK**.
- h. View the data near address 250 by double-clicking on any address in the Address column and entering **250**.

[illegible]

Now, before you leave this section, go ahead and close the memory instances already being viewed.

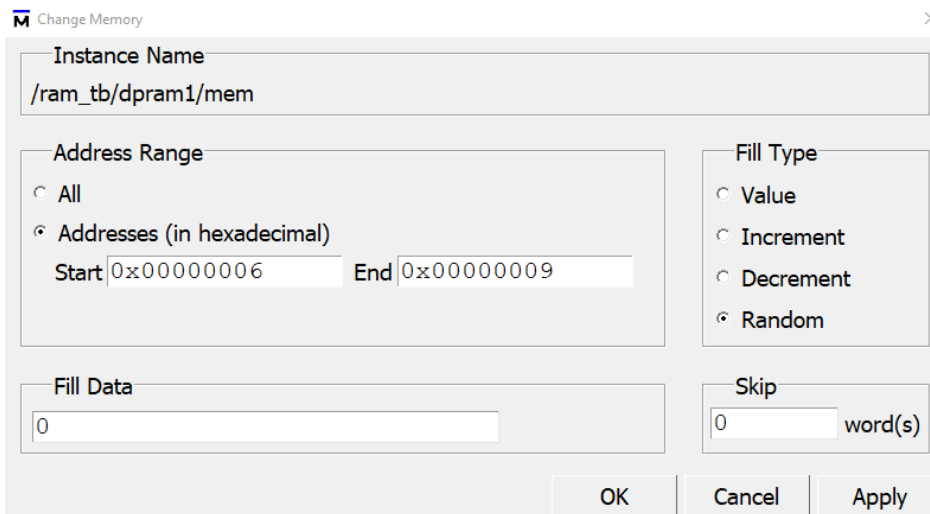
## Interactive Debugging Commands

1. Open a memory instance and change its display characteristics.
  - a. Double-click instance */ram\_tb/dpram1/mem* in the Memory List window.
  - b. Right-click in the *dpram1* Memory Data window and select **Properties**.
  - c. Change the Address and Data Radix to **Hexadecimal**.
  - d. Select **Words per line** and enter **2**.
  - e. Click **OK**. The result should be as in Figure 93.

|          |       |
|----------|-------|
| 00000000 | 06 03 |
| 00000002 | 7a 1b |
| 00000004 | 1c 1d |
| 00000006 | 1e 1f |
| 00000008 | 20 21 |
| 0000000a | 22 23 |
| 0000000c | 24 25 |
| 0000000e | 26 27 |

**Figure 93**

2. Initialize a range of memory addresses from a fill pattern.
  - a. Right-click in the data column of `/ram_tb/dpram1/mem` and select **Change** to open the Change Memory dialog box (Figure 94).
  - b. Select **Addresses** and enter the start address as **0x00000006** and the end address as **0x00000009**. The "0x" hex notation is optional.
  - c. Select Random as the Fill Type.
  - d. Enter **0** as the **Fill Data**, setting the seed for the Random pattern.



The dialog box titled "Change Memory" contains the following fields and options:

- Instance Name:** `/ram_tb/dpram1/mem`
- Address Range:**
  - ☐ All
  - ☒ Addresses (in hexadecimal)
    - Start: `0x00000006`
    - End: `0x00000009`
- Fill Type:**
  - ☐ Value
  - ☐ Increment
  - ☐ Decrement
  - ☒ Random
- Fill Data:** `0`
- Skip:** `0` word(s)
- Buttons: OK, Cancel, Apply

**Figure 94**

- e. Click **OK**.

The data in the specified range are replaced with a generated random fill pattern (Figure 95).

|          |       |
|----------|-------|
| 00000000 | 06 03 |
| 00000002 | 7a 1b |
| 00000004 | 1c 1d |
| 00000006 | 92 40 |
| 00000008 | 04 31 |
| 0000000a | 22 23 |
| 0000000c | 24 25 |
| 0000000e | 26 27 |

**Figure 95**

3. Change contents by highlighting.
 

You can also change data by highlighting them in the Address Data pane.

  - a. Highlight the data for the addresses **0x0000000c:0x0000000e**, as shown in Figure 96.

|          |       |
|----------|-------|
| 00000000 | 06 03 |
| 00000002 | 7a 1b |
| 00000004 | 1c 1d |
| 00000006 | 92 40 |
| 00000008 | 04 31 |
| 0000000a | 22 23 |
| 0000000c | 24 25 |
| 0000000e | 26 27 |

**Figure 96**

- b. Right-click the highlighted data and select **Change**.

This brings up the Change memory dialog box. Note that the Addresses field is already populated with the range you highlighted.

- c. Select **Value** as the Fill Type. (Refer to Figure 97)
- d. Enter the data values into the Fill Data field as follows: **31 32 33 34**.

**Figure 97**

- e. Click **OK**.

The data in the address locations change to the values you entered (Figure 98).

|          |       |
|----------|-------|
| 00000000 | 06 03 |
| 00000002 | 7a 1b |
| 00000004 | 1c 1d |
| 00000006 | 92 40 |
| 00000008 | 04 31 |
| 0000000a | 22 23 |
| 0000000c | 31 32 |
| 0000000e | 33 34 |

**Figure 98**

4. Edit data in place.

To edit only one value at a time, do the following:

- a. Double click any value in the Data column.
- b. Enter the desired value and press the Enter or Return key on your keyboard.

If you needed to cancel the edit function, press the Esc key on your keyboard.

## Concluding the Lab

This concludes this Lab.

5. Select **Simulate > End Simulation**.
6. Click **Yes**.
7. Select **File > Quit** to close ModelSim.