Using ModelSim-Altera in a Quartus II Design Flow

December 2002, ver. 1.2 Application Note 204

Introduction

This application note is a getting-started guide to using ModelSim®-Altera software in Altera® programmable logic device (PLD) design flows. Proper functional and timing simulation is important to ensure design functionality and success. Using the ModelSim-Altera software simplifies the verification process for large, complex designs, like those for Stratix™ devices, ensuring proper system functionality and quick time-to-market.

This document provides step-by-step explanations of the basic ModelSim-Altera functional/behavioral hardware description language (HDL) and gate-level timing simulations. It also describes the location of the simulation libraries and how to automate simulations.

Altera software subscriptions include the ModelSim-Altera software for PC or UNIX platforms. The ModelSim-Altera software is Altera specific and supports behavioral and gate-level timing simulations and either VHDL or Verilog HDL simulations and testbenches for Altera PLDs. This document describes ModelSim-Altera software version 5.6a and the ModelSim PE software version 5.6. The "Altera Design Flow with ModelSim-Altera Software" section describes ModelSim-Altera license setup.

This document contains references to features available in the Altera Quartus® II software version 2.2. Please visit the Altera web site available at **http://www.altera.com** for information on the Quartus II software version 2.2.

Software Compatibility

Table 1 shows which specific ModelSim-Altera software version is compatible with the specific Quartus II software version. ModelSim versions provided directly from Model Technology do not correspond to specific Quartus II software versions.

For help on ModelSim-Altera licensing set-up please see "Software Licensing & Licensing Set-Up" on page 33.

Table 1. Compatibility Between Software Versions

ModelSim-Altera Software	Quartus II Software *(1)*
ModelSim-Altera software version 5.5e	Quartus II software version 2.0
ModelSim-Altera software version 5.5b	Quartus II software version 1.1 (SP1 and SP2)
ModelSim-Altera software version 5.4e	Quartus II software version 1.0 (SP1 and SP2)
ModelSim-Altera software version 5.6a	Quartus II software version 2.1 (SP1) and version 2.2

Note to Table 1:

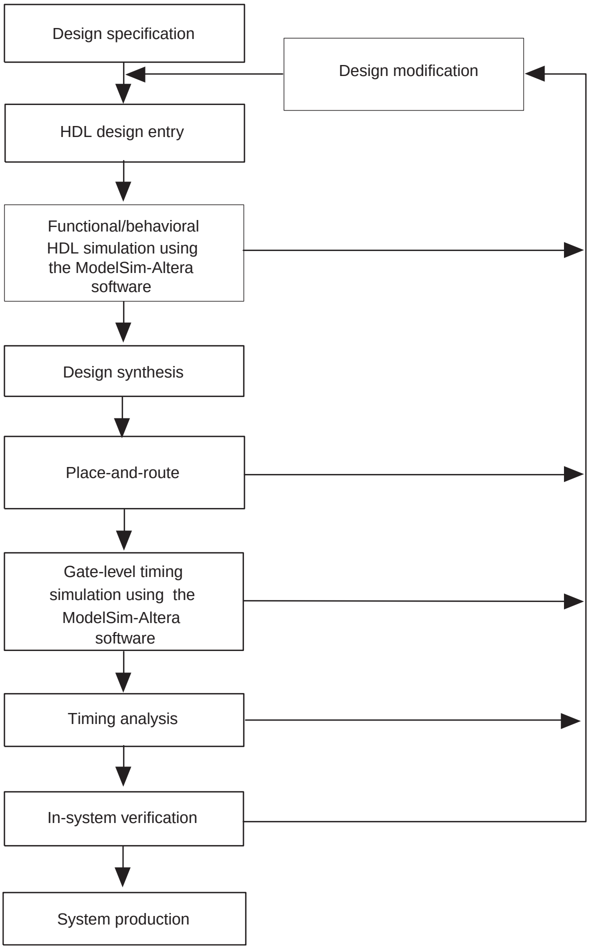
(1) ModelSim-Altera precompiled libraries are updated with Quartus II service packs (SP) and are generally available for download on Altera's web site.

Altera Design Flow with ModelSim-Altera Software

Figure 1 illustrates an Altera design flow using the ModelSim-Altera software, which can perform the following simulation types:

- Functional/behavioral HDL simulations
- Gate-level timing simulations

Figure 1. Altera Design Flow with ModelSim-Altera and Quartus II Software



Functional/Behavioral HDL Simulation

Functional/behavioral HDL simulations verify the functionality of the design. These simulations are independent of any Altera PLD architecture implementation. The ModelSim-Altera software uses either Verilog HDL or VHDL design files, including models for the library of parameterized modules (LPMs) and Altera megafunctions, to generate a functional simulation output of the design based on the set of stimulus applied by the user. Once the HDL designs are verified to be functionally correct, the next step is to synthesize the design and use the Quartus II software for place-and-route.

Gate-Level Timing Simulation

Place-and-route in the Quartus II software produces a design netlist (**.vo** or **.vho** file) and a standard delay format (SDF) output (**.sdo**) file used for gate-level timing simulation in the ModelSim-Altera software. The design netlist output file is a netlist of the design mapped to architecture-specific primitives. The SDF file contains delay information for each architecture primitive and routing element specific to the design. Together, these files provide an accurate simulation of the design for the selected Altera PLD architecture.

Methods for Running ModelSim-Altera

The ModelSim-Altera software runs in the three modes shown in Table 2.

Table 2. ModelSim-Altera Modes

Mode	Description
User interface (UI) mode	Run the ModelSim-Altera software through either the GUI menu inputs or through command-line input.
Interactive command-line mode	Run the ModelSim-Altera software only through the command-line console.
Batch mode	Run the ModelSim-Altera software through batch files executed from a command prompt.

ModelSim-Altera Simulation Flow

The basic simulation flow using the ModelSim-Altera and Quartus II software involves the following steps:

1. Create libraries.
2. Map to libraries.
3. Compile source code and testbenches.
4. Load the design.
5. Add design stimulus.
6. View the simulation results.
7. Advance the simulator.

Creating Libraries

The ModelSim-Altera software libraries are directories that contain compiled VHDL and Verilog design units. Design units can be either VHDL entity-architecture units or Verilog HDL modules. The ModelSim-Altera software must create these directories and place certain files in them. These files are not present if the directories are created through other means (e.g., Windows Explorer or the `mkdir` command in UNIX), but are necessary for successful compilation and simulation in the ModelSim-Altera software.

Table 3 describes the two library types available in the ModelSim-Altera software.

Table 3. The ModelSim-Altera Software Library Types

Library Type	Description
Working Library	Generally contains the current VHDL or Verilog HDL design unit(s) being compiled. This library must be created using the ModelSim-Altera software before compilation. Only one working directory is allowed per compilation.
Resource Library	Contains design units that can be referenced by the current compilation. Multiple resource libraries are allowed during compilation. Examples of the resource library are the **lpm** and **altera_mf** libraries.

To create a library in UI mode:

1. Choose **New > Library** (File menu). The **Create a New Library** dialog box appears.
2. Under **Create**, select **a new library and a logical mapping to it**.
3. In the **Library Name** box, type the library name.
4. Under **Library Physical Name,** specify the library's name or path.
5. Click **OK**.

To create a library in command-line mode, type the following command in the ModelSim-Altera software main window:

`vlib` *<library name>* ↵

Mapping to Libraries

To map to an existing library in UI mode:

1. Choose **New > Library** (File menu). The **Create a New Library** dialog box appears.
2. Under **Create**, select **a map to an existing library**.
3. Under **Library Name**, enter the name of the library.
4. Under **Library Physical Name,** specify the library's name or path.
5. Click **OK**.

To map to libraries in command-line mode, type the following command in the ModelSim-Altera software main window:

`vmap` *<library name> < library name or directory path>* ↵

Compiling Source Code and Testbenches

The method for compiling source code and testbenches is different for VHDL and Verilog HDL files.

VHDL Files

When compiling VHDL source code and testbenches, the files must be compiled in the order of design unit dependencies. The VHDL design unit dependencies are:

- Compile the design entity before the design architecture.
- Compile the package declaration before the package body.
- Compile design units before they are referenced by other design units.
- Compile configurations last.

Changing the VHDL Version

The default standard for VHDL compilation is VHDL ’87. If required, select the option to compile using VHDL ’93 by following these steps in the ModelSim-Altera user interface:

1. Choose **Compile** (Compile menu).

2. In the **Compile HDL Source Files** dialog box, click **Default Options**.

3. Turn on **Use 1993 Language Syntax**.

or

1. Choose **Compile Options** (Compile menu).

2. Turn on **Use 1993 Language Syntax**.

Compiling Source Files

To compile VHDL source files in UI mode:

1. Choose **Compile** (Compile menu).

2. Specify the working library.

3. Specify the VHDL files in the proper order and click **Compile**.

To compile VHDL source files in command line mode, type the following command in the ModelSim-Altera software main window:

`vcom -work` *<library name> <file1>*.**vhd** *<file2>*.**vhd** ↵

Verilog HDL Files

The compilation order of design files and testbenches does not matter when compiling Verilog HDL source code. The ModelSim-Altera software supports incremental compilation for Verilog HDL whereby only design units that have been modified need to be compiled. Perform incremental compilation manually or automatically with the ModelSim-Altera compiler. Manual incremental compilation is more efficient because the compiler does not have to check every design unit for code modifications.

Compiling Source Files

To compile Verilog HDL source files in UI mode:

1. Choose **Compile** (Compile menu).

2. Specify the working library.

3. Specify the Verilog HDL files and click **Compile**.

To compile Verilog HDL source files in command line mode, type the following command in the ModelSim-Altera software main window:

`vlog -work` *<library name> <file1>*.**v** *<file2>*.**v** ↵

Loading the Design

Load the design prior to simulating it. Load only the highest-level design file from the working directory. Extra steps are required to load the SDF file when performing gate-level timing simulations. For more information refer to the "Gate-Level Timing Simulation" section.

To load a design in UI mode:

1. Choose **Simulate** (Simulate menu).

2. Specify the highest-level design unit or testbench and click **Load**.

To load a design in command line mode, type the following command in the ModelSim-Altera software's main window:

`vsim work.`*<top-level design unit>* ↵

Adding Design Stimulus

Add design stimulus to the simulation with VHDL or Verilog HDL testbenches or through the ModelSim-Altera software **force** command. The **force** command provides a simple method for adding simulation stimulus directly from the command line.

The syntax for the **force** command is:

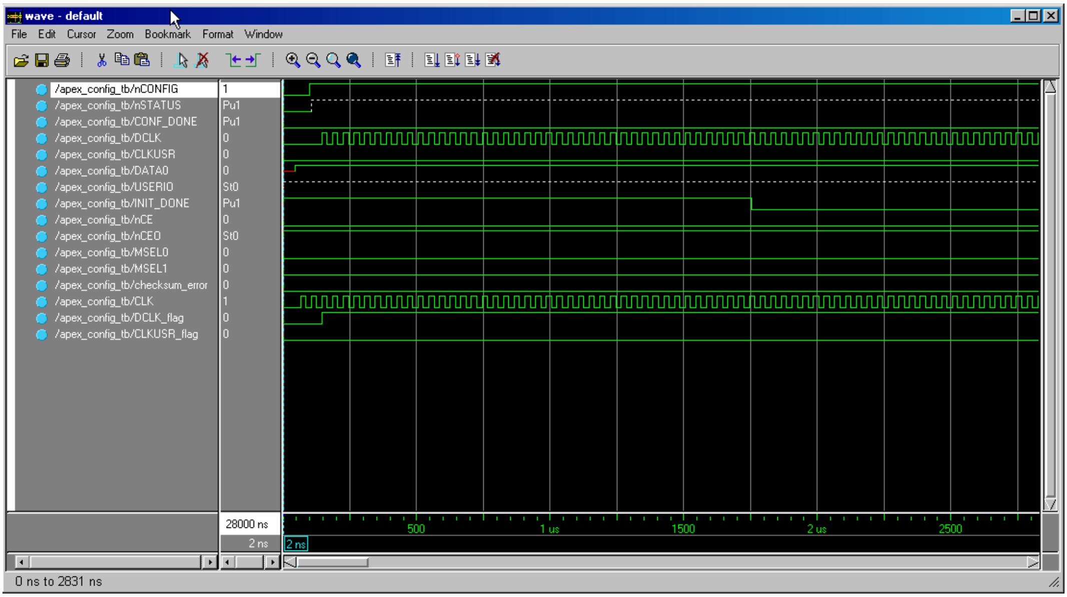
`force` *<signal name to be forced> <value> <time>, <value> <time>* ↵

For more information about using the **force** command, refer to the ModelSim Command Reference available in the **Documentation** section of the ModelSim-Altera software Help menu.

Viewing Simulation Results

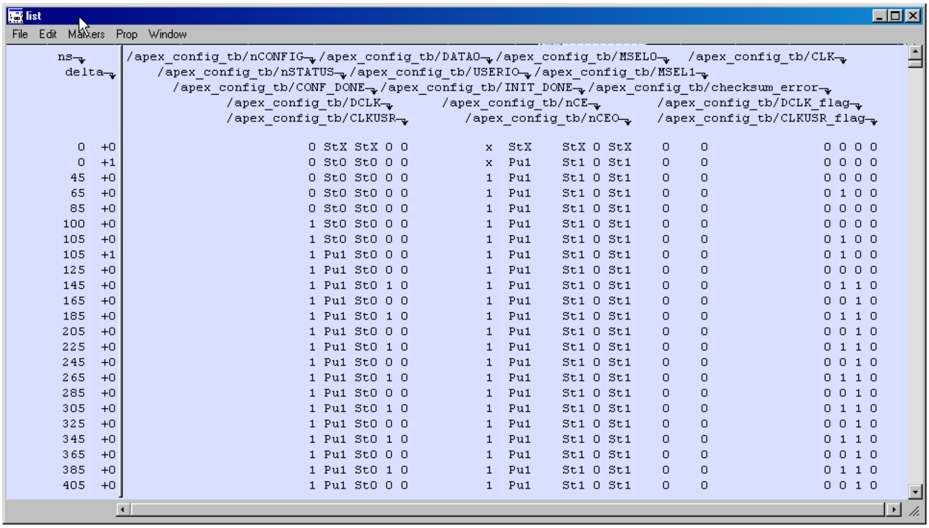
View the simulation stimulus and simulation results in the Wave and List windows. The Wave window displays the simulation results using waveforms as shown in Figure 2. The List window displays simulation results using vectors. The List window is shown in Figure 3.

Figure 2. The ModelSim-Altera Wave Window



```
/apex_config_tb/nCONFIG          1
/apex_config_tb/nSTATUS          Pu1
/apex_config_tb/CONF_DONE        Pu1
/apex_config_tb/DCLK             0
/apex_config_tb/CLKUSR           0
/apex_config_tb/DATA0            0
/apex_config_tb/USERIO           St0
/apex_config_tb/INIT_DONE        Pu1
/apex_config_tb/nCE              0
/apex_config_tb/nCEO             St0
/apex_config_tb/MSEL0            0
/apex_config_tb/MSEL1            0
/apex_config_tb/checksum_error   0
/apex_config_tb/CLK              1
/apex_config_tb/DCLK_flag        0
/apex_config_tb/CLKUSR_flag      0
                                 28000 ns
                                 2 ns      2 ns
0 ns to 2831 ns
```

Figure 3. The ModelSim-Altera List Window



```
     ns    /apex_config_tb/nCONFIG /apex_config_tb/DATA0 /apex_config_tb/MSEL0  /apex_config_tb/CLK
  delta        /apex_config_tb/nSTATUS /apex_config_tb/USERIO /apex_config_tb/MSEL1
                 /apex_config_tb/CONF_DONE /apex_config_tb/INIT_DONE /apex_config_tb/checksum_error
                        /apex_config_tb/DCLK           /apex_config_tb/nCE           /apex_config_tb/DCLK_flag
                        /apex_config_tb/CLKUSR            /apex_config_tb/nCEO       /apex_config_tb/CLKUSR_flag

      0  +0                       0 StX StX 0 0           x  StX    StX 0 StX   0    0            0 0 0 0
      0  +1                       0 St0 St0 0 0           x  Pu1    St1 0 St1   0    0            0 0 0 0
     45  +0                       0 St0 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 0 0
     65  +0                       0 St0 St0 0 0           1  Pu1    St1 0 St1   0    0            0 1 0 0
     85  +0                       0 St0 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 0 0
    100  +0                       1 St0 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 0 0
    105  +0                       1 St0 St0 0 0           1  Pu1    St1 0 St1   0    0            0 1 0 0
    105  +1                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 1 0 0
    125  +0                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 0 0
    145  +0                       1 Pu1 St0 1 0           1  Pu1    St1 0 St1   0    0            0 1 1 0
    165  +0                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 1 0
    185  +0                       1 Pu1 St0 1 0           1  Pu1    St1 0 St1   0    0            0 1 1 0
    205  +0                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 1 0
    225  +0                       1 Pu1 St0 1 0           1  Pu1    St1 0 St1   0    0            0 1 1 0
    245  +0                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 1 0
    265  +0                       1 Pu1 St0 1 0           1  Pu1    St1 0 St1   0    0            0 1 1 0
    285  +0                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 1 0
    305  +0                       1 Pu1 St0 1 0           1  Pu1    St1 0 St1   0    0            0 1 1 0
    325  +0                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 1 0
    345  +0                       1 Pu1 St0 1 0           1  Pu1    St1 0 St1   0    0            0 1 1 0
    365  +0                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 1 0
    385  +0                       1 Pu1 St0 1 0           1  Pu1    St1 0 St1   0    0            0 1 1 0
    405  +0                       1 Pu1 St0 0 0           1  Pu1    St1 0 St1   0    0            0 0 1 0
```

Advancing the Simulator

To advance the simulator, use the Run menu in UI mode or the **run** command in command-line mode. The **run** command has several time step options to advance the simulator. The **restart** option reloads any design elements that have been modified since the last simulation and resets the simulation time to zero.

☞ The **restart** option only restarts the simulation, it does not compile the design files.

For more information about using the **run** command, refer to the ModelSim-Altera Command Reference available in the **Documentation** section of the Help menu.

Functional/ Behavioral HDL Simulation

Functional/behavioral HDL simulation generally performed before place-and-route simulation. It tests for the correct logical function of the design. This section provides detailed instructions about how to perform functional/behavioral HDL simulation in the ModelSim-Altera software and highlights differences in performing similar steps in the Model Technology ModelSim software versions for VHDL and Verilog HDL designs.

LPM and Altera Megafunction Functional Simulation Models

To simulate designs containing LPMs or MegaWizard®-generated functions, use the Altera functional simulation models.

The LPM simulation model files are:

- **220model.v** (for Verilog HDL)
- **220pack.vhd** and **220model.vhd** (for VHDL)

If simulating a design that uses VHDL-1987 with the Quartus II software version 2.2, use **220model_87.vhd**.

These files include simulation models for standard LPM functions.

For more information on LPMs, see the Quartus II Help.

Table 4 shows the location of the model files in the Quartus II software and the ModelSim-Altera software for Verilog HDL designs.

Table 4. Location of LPM Simulation Models for Verilog HDL Designs

Software	Verilog HDL
Quartus II	<*Quartus II installation directory*>`\eda\sim_lib\` *(1)*
ModelSim-Altera (PC)	<*ModelSim-Altera installation directory*>`\altera\verilog\220model\`
ModelSim-Altera (UNIX)	<*ModelSim-Altera installation directory*>`/modeltech/altera/verilog/220model/`

Notes to Table 4:

(1) For Model Technology's ModelSim, use the files provided with the Quartus II software.

Table 5 shows the location of the model files in the Quartus II software and the ModelSim-Altera software for VHDL designs.

Table 5. Location of LPM Simulation Models for VHDL Designs

Software	VHDL
Quartus II	<*Quartus II installation directory*>`\eda\sim_lib\` *(1), (2)*
ModelSim-Altera (PC)	<*ModelSim-Altera installation directory*>`\altera\vhdl\220model\`
ModelSim-Altera (UNIX)	<*ModelSim-Altera installation directory*>`/modeltech/altera/vhdl/220model/`

Notes to Table 5:

(1) Compile 220pack.**vhd** before 220model.**vhd**.

(2) For Model Technology's ModelSim, use the files provided with the Quartus II software.

The Altera megafunction simulation model files are listed below:

- altera_mf.**v** (for Verilog HDL)
- altera_mf.**vhd** and altera_mf_components.**vhd** (for VHDL)
- 220model_87.**vhd** (for VHDL--Version 2.2 of the Quartus II software only.)

These files include simulation models for Altera-specific megafunctions.

☞ If simulating a design that uses VHDL-1987 with the Quartus II software version 2.2, use **220model_87.vhd**.

☞ If simulating a design implementing the 3.125-Gigabit transciever blocks available in Stratix GX devices, use the `altgxb` library found in the directories specified in Table 6 and Table 7. For more information about these libraries, see the readme file at <*path*>**\quartus\eda\sim_lib\modelsim**.

Table 6 shows the location of these files in the Quartus II software and the ModelSim-Altera software for Verilog HDL designs.

Table 6. Location of Altera Megafunction Simulation Models for Verilog HDL Designs

Software	Verilog HDL
Quartus II	<*Quartus II installation directory*>`\eda\sim_lib\` *(1)*
ModelSim-Altera (PC)	<*ModelSim-Altera installation directory*>`\altera\verilog\altera_mf\`
ModelSim-Altera (UNIX)	<*ModelSim-Altera installation directory*>`/modeltech/altera/verilog/altera_mf/`

Notes to Table 6:

(1) For Model Technology's ModelSim, use the files provided with the Quartus II software.

Table 7 shows the location of these files in the Quartus II software and the ModelSim-Altera software for VHDL designs.

Table 7. Location of Altera Megafunction Simulation Models for VHDL Designs

Software	VHDL
Quartus II	<*Quartus II installation directory*>`\eda\sim_lib\` *(1), (2)*
ModelSim-Altera (PC)	<*ModelSim-Altera installation directory*>`\altera\vhdl\altera_mf\`
ModelSim-Altera (UNIX)	<*ModelSim-Altera installation directory*>`/modeltech/altera/vhdl/altera_mf/`

Notes to Table 7:

(1) Compile altera_mf_components.**vhd** before altera_mf.**vhd**.

(2) For Model Technology's ModelSim, use the files provided with the Quartus II software.

Functional/Behavioral HDL Simulation for VHDL Designs

Table 8 provides step-by-step instructions to perform functional/behavioral HDL simulation for VHDL designs. The first column shows the GUI steps, and the second column shows the command-line entries for a simulation.

Table 8. Functional/Behavioral HDL Simulation for VHDL Designs (Part 1 of 4)

GUI	Command Line
Specify the project simulation directory	
1. Choose **Change Directory** (File menu). 2. In the **Choose a Directory** dialog box, specify the project simulation directory. 3. Click **OK**.	`cd` *<full path to project simulation directory>* ↵
Create a new work library	
4. Choose **New > Library** (File Menu). 5. Under **Create**, select **a new library and a logical mapping to it**. 6. In the **Library Name** box, type `work` (default). 7. Click **OK**.	`vlib work` ↵
Map the design libraries to the work library	
8. Choose **New > Library** (File Menu). 9. Under **Create**, select **a map to an existing library**.	■ For LPM functions type: `vmap lpm` *<path to LPM megafunction library>* ↵ ■ For Altera megafunctions type: `vmap altera_mf` *<path to Altera megafunction library>* ↵

Table 8. Functional/Behavioral HDL Simulation for VHDL Designs (Part 2 of 4)

GUI	Command Line
Map the design libraries to the work library (continued)	
10. In the **Library Name** box, type one of the following: a. For LPM functions: **lpm** b. For Altera megafunctions: **altera_mf** 11. In the **Library Physical Name** box, specify the location of the LPM or Altera megafunction simulation model. Refer to Table 5 and Table 7 for more information. ☞ If using Model Technology's ModelSim, map the **lpm** and **altera_mf** libraries to the **work** library. Then compile the LPM or Altera megafunction VHDL files into the **lpm** or **altera_mf** libraries, respectively. Refer to Table 5 and Table 7 for the location of these files. 12. Click **OK**. ☞ If the following error is received: **# Error: Could not write to <ModelSim installation directory>\win32aloem\..\modelsim.ini: Permission denied**, choose **Change Directory** (File menu), specify the directory of the current ModelSim project and click **Open**. This error sometimes occurs even after the project simulation directory has been specified. In this case repeat steps 8 to 12. 13. If necessary, repeat steps 8 to 12 for the next library.	☞ If using Model Technology's ModelSim, map the **lpm** and **altera_mf** libraries to the **work** library. Then compile the LPM or Altera megafunction VHDL files into the **lpm** or **altera_mf** libraries, respectively. Refer to Table 5 and Table 7 for the location of these files.

Table 8. Functional/Behavioral HDL Simulation for VHDL Designs (Part 3 of 4)

GUI	Command Line
Compile source files and design files with LPM and Altera megafunctions into the work library	
14. Choose **Compile** (Compile menu) or click the **Compile** toolbar icon. 15. In the **Library** list, select the **work** library. 16. In the **File Name** box, specify the design files to be compiled and click **Compile**. ☞ VHDL source files must be compiled according to the hierarchy (i.e., top-level file last). Testbenches are compiled at this time also. 17. Click **Done**.	`vcom -work work` *<file1>*.**vhd** *<file2>*.**vhd** ↵ ☞ *<file1>*.**vhd** will be compiled before *<file2>*.**vhd**
Load designs	
18. Choose **Simulate** (Simulate menu) or click the **Simulate** toolbar icon. 19. In the **Library** list (Design tab), select the **work** library. 20. Specify the top-level file. If using a testbench as the top-level file, specify the testbench. ☞ To obtain the most accurate simulation results, select **ps** in the **Simulation Resolution** list. The increased simulation accuracy increases simulation time. 21. Click **Add**. 22. Click **Load**.	`vsim work.`*<top-level design unit>* ↵

Table 8. Functional/Behavioral HDL Simulation for VHDL Designs (Part 4 of 4)

GUI	Command Line
Run a simulation	
23. Choose **Signals** and **Wave** (View menu).	`view signals` ↵
24. Drag signals to monitor from the Signals window and drop them into the Wave window. If not using a testbench, use the **force** command to add stimulus as shown in the command line column.	`view wave` ↵ `add wave /`<*signal name*> ↵ `force` <*signal name*> <*value*> <*time*>`,` <*value*> <*time*> ↵
25. Choose **Run** (Simulate menu) by selecting the time until which to run the simulation. ☞ To simulate or view signals not in the top-level file, view the Structure window.	`run` <*time period*> ↵

Functional/Behavioral HDL Simulation for Verilog HDL Designs

Table 9 provides step-by-step instructions for performing functional/behavioral HDL simulation for Verilog HDL designs. The first column shows the GUI steps, and the second column shows the command line entries for a simulation.

Table 9. Functional/Behavioral HDL Simulation for Verilog HDL Designs (Part 1 of 4)

GUI	Command Line
Specify the project simulation directory	
1. Choose **Change Directory** (File menu). 2. In the **Choose a Directory** dialog box, specify the project simulation directory. 3. Click **OK**.	`cd` <*full path to project simulation directory*> ↵

Table 9. Functional/Behavioral HDL Simulation for Verilog HDL Designs (Part 2 of 4)

GUI	Command Line
Create a work library	
4. Choose **New > Library** (File Menu). 5. Under **Create**, select **a new library and a logical mapping to it**. 6. In the **Library Name** box, type `work` (default). 7. Click **OK**	`vlib work` ↵
Compile source files and design files with LPM and Altera megafunctions into work library	
8. Choose **Compile** (Compile menu) or click the **Compile** toolbar icon. 9. In the **Library** list, select the **work** library. 10. In the **File Name** box, specify the design files to be compiled and click **Compile**. 11. Click **Done**	`vlog -work work` *<file1>*`.v` *<file2>*`.v` ↵

Table 9. Functional/Behavioral HDL Simulation for Verilog HDL Designs (Part 3 of 4)

GUI	Command Line
Load the designs and map to the Altera precompiled libraries	
1. Choose **Simulate** (Simulate menu) or click the **Simulate** toolbar icon. 2. In the **Load Design** dialog box, click the **Libraries** tab. 3. In the **Search Libraries (-L)** box, click **Add**. 4. Specify the location to the LPM or Altera megafunction simulation model. Refer to Table 4 and Table 6 for more information. 5. In the **Load Design** dialog box, click the **Design** tab. 6. In the **Library** list, select the **work** library. 7. Specify the top-level file. If using a testbench as the top-level file, specify the testbench. ☞ To obtain the most accurate simulation results, select **ps** in the **Simulation Resolution** list. The increased simulation accuracy increases simulation time. 8. Click **Add** and then click **Load**. ☞ If using Model Technology's ModelSim, first create the **lpm** and **altera_mf** libraries, then compile the LPM or Altera megafunction Verilog HDL files into the **lpm** or **altera_mf** libraries, respectively. See Table 4 and Table 6 for the location of these files in the Quartus II installation directory. When loading the design, specify the **lpm** and/or **altera_mf** library just created.	`vsim -L` *<path to library 1>* `-L` *<path to library 2>* `work.`*<top-level design unit>* ↵ ☞ If using Model Technology's ModelSim, first create the **lpm** and **altera_mf** libraries, then compile the LPM or Altera megafunction Verilog HDL files into the **lpm** or **altera_mf** libraries, respectively. See Table 4 and Table 6 for the location of these files in the Quartus II installation directory. When loading the design, specify the **lpm** and/or **altera_mf** library just created.

Table 9. Functional/Behavioral HDL Simulation for Verilog HDL Designs (Part 4 of 4)

GUI	Command Line
Run a simulation	
9. Choose **Signals** and **Wave** (View menu).	`view signals` ↵
10. Drag signals to be monitored from the Signals window and drop them into the Wave window. If not using a testbench, use the **force** command to add stimulus as shown in the command line column.	`view wave` ↵ `add wave /`*<signal name>* ↵ `force` *<signal name> <value> <time>*`,` *<value> <time >* ↵
11. Run the simulation using the Run menu by selecting the time unit until which to run the simulation. ☞ To simulate or view signals not in the top-level file, view the Structure window.	`run` *<time period>* ↵

Gate-Level Timing Simulation

Gate-level timing simulation is a post place-and-route simulation to verify the operation of the design after the worst-case timing delays have been calculated. This section provides detailed instructions on how to perform gate-level timing simulation in the ModelSim-Altera software and highlights differences in performing similar steps in the Model Technology ModelSim software versions for VHDL and Verilog HDL designs.

Quartus II Output Files for use in the ModelSim-Altera Software

To perform gate-level timing simulation, the ModelSim-Altera software requires information on how the design was placed into device-specific architectural blocks. The Quartus II software provides this information in the form of **.vo** for Verilog HDL and **.vho** for VHDL output files. The accompanying timing information is stored in a SDF file that annotates the delay for the elements found in the **.vo** or **.vho** output file.

Table 10 shows the different Quartus II output files for use in the ModelSim-Altera software for Verilog HDL and VHDL.

Table 10. Quartus II Output Files for Use in the ModelSim-Altera Software

HDL Type	Quartus II Output Netlist File	Standard Delay Format (SDF) Output File
Verilog HDL	<*filename*>.**vo**	<*filename*>_v.**sdo**
VHDL	<*filename*>.**vho**	<*filename*>_vhd.**sdo**

To obtain a ModelSim-Altera simulation output file in the Quartus II software, perform the following steps:

1. Choose **Open Project** (File menu).

2. Select the project and click **OK**.

3. Choose **EDA Tool Settings** (Project menu).

4. In the **Simulation Tool** box:

   - If using ModelSim-Altera, select **ModelSim OEM (VHDL/Verilog HDL output from Quartus II).**
   - If using Model Technology's ModelSim, select **ModelSim (VHDL/Verilog HDL output from Quartus II)**.

5. Click **OK**.

6. Compile the project.

7. The Quartus II output files will be located in the <*full path to project*>`\simulation\modelsim\` directory.

Location of Simulation Libraries

In the Quartus II software, the timing information for specific device architecture entities is located in the ATOM simulation files. The timing simulation library files differ based on device family and whether using Verilog HDL or VHDL. Table 11 provides a description of the various ModelSim-Altera precompiled device libraries.

Table 11. Various ModelSim-Altera Precompiled Device Libraries

Library	Description
cyclone	Precompiled library for Cyclone™ designs
stratixgx	Precompiled library for Stratix™ GX designs
stratixgx_gxb	Precompiled library for Stratix™ GX designs using the Gigabit Transceiver Block (altgxb Megafunction)
stratix	Precompiled library for Stratix designs
apexii	Precompiled library for APEX™ II designs
apex20ke	Precompiled library for APEX 20KC, APEX 20KE and ARM®-based Excalibur™ designs
apex20k	Precompiled library for APEX™ 20K designs
mercury	Precompiled library for Mercury™ designs
flex10ke	Precompiled library for FLEX 10KE and ACEX™ 1K designs
flex6000	Precompiled library for FLEX® 6000 designs
max	Precompiled library for MAX® 7000 and MAX 3000 designs

Table 12 shows the location of the timing simulation libraries in the ModelSim-Altera software for Verilog HDL for PCs.

Table 12. Location of Timing Simulation Libraries for ModelSim-Altera for Verilog HDL on a PC

Library	Verilog HDL
cyclone	<*ModelSim-Altera installation directory*>`\altera\verilog\cyclone\`
stratixgx	<*ModelSim-Altera installation directory*>`\altera\verilog\stratixgx\`
stratixgx_gxb	<*ModelSim-Altera installation directory*>`\altera\verilog\stratixgx_gxb\`
stratix	<*ModelSim-Altera installation directory*>`\altera\verilog\stratix\`
apexii	<*ModelSim-Altera installation directory*>`\altera\verilog\apexii\`
apex20ke	<*ModelSim-Altera installation directory*>`\altera\verilog\apex20ke\`
apex20k	<*ModelSim-Altera installation directory*>`\altera\verilog\apex20k\`
mercury	<*ModelSim-Altera installation directory*>`\altera\verilog\mercury\`
flex10ke	<*ModelSim-Altera installation directory*>`\altera\verilog\flex10ke\`
flex6000	<*ModelSim-Altera installation directory*>`\altera\verilog\flex6000\`
max	<*ModelSim-Altera installation directory*>`\altera\verilog\max\`

Table 13 shows the location of the timing simulation libraries in the ModelSim-Altera software for VHDL for PCs.

Table 13. Location of Timing Simulation Libraries for ModelSim-Altera for VHDL on a PC

Library	VHDL
cyclone	*<ModelSim-Altera installation directory>*`\altera\vhdl\cyclone\`
stratixgx	*<ModelSim-Altera installation directory>*`\altera\vhdl\stratixgx\`
stratixgx_gxb	*<ModelSim-Altera installation directory>*`\altera\vhdl\stratixgx_gxb\`
stratix	*<ModelSim-Altera installation directory>*`\altera\vhdl\stratix\`
apexii	*<ModelSim-Altera installation directory>*`\altera\vhdl\apexii\`
apex20ke	*<ModelSim-Altera installation directory>*`\altera\vhdl\apex20ke\`
apex20k	*<ModelSim-Altera installation directory>*`\altera\vhdl\apex20k\`
flex10ke	*<ModelSim-Altera installation directory>*`\altera\vhdl\flex10ke\`
flex6000	*<ModelSim-Altera installation directory>*`\altera\vhdl\flex6000\`
mercury	*<ModelSim-Altera installation directory>*`\altera\vhdl\mercury\`
max	*<ModelSim-Altera installation directory>*`\altera\vhdl\max\`

Table 14 shows the location of the timing simulation libraries in the ModelSim-Altera software for Verilog HDL for UNIX.

Table 14. Location of Timing Simulation Libraries for ModelSim-Altera for Verilog HDL with UNIX

Library	Verilog HDL
cyclone	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/cyclone/`
stratixgx	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/stratixgx/`
stratixgx_gxb	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/stratixgx_gxb/`
stratix	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/stratix/`
apexii	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/apexii/`
apex20ke	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/apex20ke/`
apex20k	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/apex20k/`
mercury	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/mercury/`
flex10ke	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/flex10ke/`
flex6000	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/flex6000/`
max	*<ModelSim-Altera installation directory>*`/modeltech/altera/verilog/max/`

Table 15 shows the location of the timing simulation libraries in the ModelSim-Altera software for VHDL for UNIX.

Table 15. Location of Timing Simulation Libraries for ModelSim-Altera for VHDL with UNIX

Library	VHDL
cyclone	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/cyclone/`
stratixgx	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/stratixgx/`
stratixgx_gxb	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/stratixgx_gxb/`
stratix	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/stratix/`
apexii	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/apexii/`
apex20ke	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/apex20ke/`
apex20k	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/apex20k/`
mercury	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/mercury/`
flex10ke	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/flex10ke/`
flex6000	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/flex6000/`
max	*<ModelSim-Altera installation directory>*`/modeltech/altera/vhdl/max/`

The timing simulation libraries are also available in the Quartus II software at the following location: <*Quartus II installation directory*>`\eda\sim_lib\`. Model Technology ModelSim software users must use the files provided with the Quartus II software. If using VHDL, compile <*device family*>`_atoms.`**vhd** before compiling <*device family*>`_components.`**vhd**.

☞ The precompiled timing simulation library for designs using the 3.125 Gigabit transciever blocks in Stratix GX devices can be found in the Quartus II installation directory at <*Quartus II installation directory*>`\eda\sim_lib\modelsim\vhdl\stratixgx_gxb` (for VHDL designs), or <*Quartus II installation directory*>`\eda\sim_lib\modelsim\verilog\stratixgx_gxb` (for Verilog HDL designs).

Gate-Level Timing Simulation for VHDL Designs

Table 16 provides step-by-step instructions for performing gate-level timing simulation for VHDL designs. The first column shows the GUI steps, and the second column shows the command-line entries for a simulation.

Table 16. Gate-Level Timing Simulation for VHDL Designs (Part 1 of 4)

GUI	Command Line
Specify the project simulation directory	
1. Choose **Change Directory** (File menu). 2. In the **Choose a Directory** dialog box, specify the project simulation directory. 3. Click **OK**.	`cd` *<full path to project simulation directory>* ↵
Create a work library	
4. Choose **New > Library** (File Menu). 5. Under **Create**, select **a new library and a logical mapping to it**. 6. Under **Library Name**, type `work`. 7. Click **OK**.	`vlib work` ↵

Table 16. Gate-Level Timing Simulation for VHDL Designs (Part 2 of 4)

GUI	Command Line
Map to the precompiled library	
8. Choose **New > Library** (File Menu). 9. Under **Create**, select **a map to an existing library**. 10. In the **Library Name** box, type in the device family name. Refer to Table 11 for more information on the device family name. 11. In the **Library Maps to** box, specify the full path to the device family precompiled library. Refer to Table 13 and Table 15 for more information on the location of the precompiled library. ☞ If using Model Technology's ModelSim, first create the device family library and map it to the **work** library. Next, the compile the device family ATOM library files into the device family library. Refer to the section titled "Location of Simulation Libraries" for the location of these files in the Quartus II software.	PC: `vmap` <device family> <*ModelSim installation directory*>`\altera\vhdl\`<*device family*> ↵ UNIX: `vmap` <*device family*> <*ModelSim installation directory*>`/modeltech/altera/vhdl/`<*device family*> ↵ ☞ If using Model Technology's ModelSim, first create the <*device family*> library and map it to the **work** library. Next, the compile the device family ATOM library files into the <*device family*> library. Refer to the section titled "Location of Simulation Libraries" for the location of these files in the Quartus II software

Table 16. Gate-Level Timing Simulation for VHDL Designs (Part 3 of 4)

GUI	Command Line
Map to the precompiled library (continued)	
12. Click **OK**. ☞ If the following error is received: **# Error: Could not write to <ModelSim installation directory>/win32aloem/../modelsim.ini: Permission denied**, choose **Change Directory** (File menu), specify the directory of the current ModelSim project and click **Open**. This error sometimes occurs even after the proper project simulation directory has been specified. In this case, repeat steps 8 to 12. 13. If necessary, repeat steps 8 to 12 for next library.	
Compile .vho file into work library	
14. Choose **Compile** (Compile menu). 15. In the **Library** list, select the **work** library. 16. Specify the **.vho** file and click **Compile**. Testbenches are compiled at this stage as well. If the testbench instantiates the **.vho** file, compile the testbench last. 17. Click **Done**.	`vcom -work work` *<design>*`.vho` *<testbench>*`.vhd` ↵ ☞ *<design>*.**vho** is compiled before *<testbench>*.**vhd** assuming the use of a testbench.

Table 16. Gate-Level Timing Simulation for VHDL Designs (Part 4 of 4)

GUI	Command Line
Load designs	
18. Choose **Simulate** (Simulate menu) or click the **Load Design** toolbar icon. 19. Click the **SDF** tab and click **Add**. 20. Specify the location of the SDF file and click **OK**. 21. In the **Library** list (**Design** tab), select the **work** library. 22. Specify the top-level file. If using a testbench as the top-level file, specify the testbench. 23. Click **Add**. 24. Click **Load**. ☞ To obtain the most accurate simulation results, select **ps** in the **Simulation Resolution** list. The increased simulation accuracy increases simulation time.	`vsim -sdftyp /=`*<design>*`.sdo work.`*<top-level design unit>* ↵
Run simulation	
25. Choose **Signals** and **Wave** (View menu). 26. Drag signals to monitor from the Signals window and drop them into the Wave window. If not using a testbench, use the **force** command to add stimulus as shown in the Command Line section below. 27. Run the simulation using the Run menu by selecting the time until which to run the simulation. ☞ To simulate or view signals not in the top-level file, view the Structure window.	`view signals` ↵ `view wave` ↵ `add wave /`*<signal name>* ↵ `force` *<signal name> <value> <time>*`,` *<value> <time>* ↵ `run` *<time period>* ↵

Gate-Level Timing Simulation for Verilog HDL Designs

This section provides step-by-step instructions on performing gate-level timing simulation for Verilog HDL designs. The first column shows the GUI steps, and the second column shows the command-line entries for a simulation.

Table 17. Gate-Level Timing Simulation for Verilog HDL Designs (Part 1 of 3)

GUI	Command Line
Specify the project simulation directory	
1. Choose **Change Directory** (File menu). 2. In the **Choose a Directory** dialog box, specify the project simulation directory. 3. Click **OK**.	`cd` *<full path to project simulation directory>* ↵
Create a new work library	
4. Choose **New > Library** (File Menu). 5. Under **Create**, select **a new library and a logical mapping to it**. 6. Under **Library Name**, type `work`. 7. Click **OK**.	`vlib work` ↵
Compile .vo file into the work library	
8. Choose **Compile** (Compile menu). 9. In the **Library** list, select the **work** library. 10. Specify the **.vo** file and click **Compile**. Testbenches are compiled at this stage as well. 11. Click **Done**.	`vcom -work work` *<design>*`.vo` *<testbench>*`.v` ↵

Table 17. Gate-Level Timing Simulation for Verilog HDL Designs (Part 2 of 3)

GUI	Command Line
Load designs and map to the Altera precompiled libraries	
12. Choose **Simulate** (Simulate menu) or click the **Simulate** toolbar icon. 13. In the **Simulate** dialog box, click the **Libraries** tab. 14. In the **Search Libraries (-L)** box, click **Add**. 15. Specify the full path to the device family's precompiled library. Refer to Table 12 and Table 14 for more information on the location to the precompiled library. ☞ If using Model Technology's ModelSim, first create the device family library, then compile the device family ATOM library files into the device family library. Refer to the section titled "Location of Simulation Libraries" for the location of these files in the Quartus II software. When loading the design, specify the device family library just created. 16. Click the **SDF** tab. 17. Click **Add**. 18. Specify the location of the SDF file and click **OK**. 19. In the **Library** list (Design tab), select the **work** library. 20. Specify the top-level file. If using a testbench as the top-level file, select and load the testbench.	`vsim -L` *<ModelSim installation directory>*`\altera\verilog\`*<device family>* `-sdftyp /=`*<design>*`.sdo work.`*<top-level design unit>* ↵ ☞ If using Model Technology's ModelSim, first create the *<device family>* library, then compile the device family ATOM library files into the *<device family>* library. Refer to the section titled "Location of Simulation Libraries" for the location of these files in the Quartus II software. When loading the design, specify the *<device family>* library just created.

Table 17. Gate-Level Timing Simulation for Verilog HDL Designs (Part 3 of 3)

GUI	Command Line
Load designs and map to the Altera precompiled libraries (continued)	
21. Click **Add**. 22. Click **Load**. ☞ To obtain the most accurate simulation results, select **ps** in the **Simulation Resolution** list. The increased simulation accuracy increases simulation time.	
Run a simulation	
23. Choose **Signals** and **Wave** (View menu). 24. Drag signals to monitor from the Signals window and drop them into the Wave window. If not using a testbench, use the **force** command to add stimulus. 25. Run the simulation using the Run menu by selecting the time unit until which to run the simulation. ☞ To simulate or view signals not in the top-level file, view the Structure window.	`view signals` ↵ `view wave` ↵ `add wave /`*<signal name>* ↵ `force` *<signal name> <value> <time>*`,` *<value> <time>* ↵ `run` *<time period>* ↵

Automating Simulation Procedures Within the ModelSim-Altera Software

ModelSim-Altera simulation procedures may be automated using macro files (.**do)** or through tool command language (Tcl) scripts. This section of the document describes how to automate the ModelSim-Altera simulation flow using macro files.

Creating Macro Files (.do)

With the use of macro files, one command entry in the command line interface can invoke the entire simulation processes. Simulation steps may include the following:

- Setting up libraries
- Compiling design files
- Loading and simulating designs
- Forcing simulator stimulus

The syntax of the macro file is the same as the corresponding command-line entry for the various ModelSim simulation commands. A macro file can call another macro file.

For more information on macro files, refer to the ModelSim User's Manual available in the **Documentation** section in the ModelSim-Altera Help menu.

Example Macro Files

In the following two example macro files, the text following the `#s` is a comment to explain the command in the example files. Macro files can contain comments in this format.

The following example is a typical macro file.

```
cd c:\mydir                           # Change project directory to c:\mydir
vlib work                             # Create a work library
vcom -work work counter.vhd           # Compile counter.vhd into the work library
vsim counter                          # Load the design called counter
view *                                # View all ModelSim windows
do stimulus.do                        # Call the stimulus.do macro file
```

The following example is the **stimulus.do** file, which the **my_sim.do** file calls in the previous example.

```
add wave /clk                      # Add the ‘clk,’ ‘clr,’ and ‘load’ input
add wave /clr                         signals to the waveform/list window
add wave /load
add wave -hex /data                # Add the input signal ‘data’ to the
                                      waveform/list window in hex format
add wave /q                        # Add the signal ‘q’ to the waveform/list window
                                      output
force /clk 0 0, 1 50 -repeat 100   # At 0 time units, clk = ‘0’, at 50 time units,
                                      clk = ‘1’, repeat every 100 # time units
force /clr 0 0, 1 100              # At 0 time units, clr = ‘0’, at 100 time units,
                                      clr = ‘1’ and stays at ‘1’
force /load 1 0, 0 100’0’          # At 0 time units, load = ‘1’, at 100 time units,
                                      load = ‘0’ and stays at ‘0
force /data 16#A5 0                # At 0 time units, data = 16-bit hex number ‘A5’
run 1000                           # Run simulation for 1000 time units
```

Invoking a Macro File

Invoke the macro file through the GUI as follows:

1. Choose **Execute Macro** (Macro menu).

2. Specify **.do** macro filename.

3. Click **Open**.

or through the command line as follows:

```
do <filename>.do ↵
```

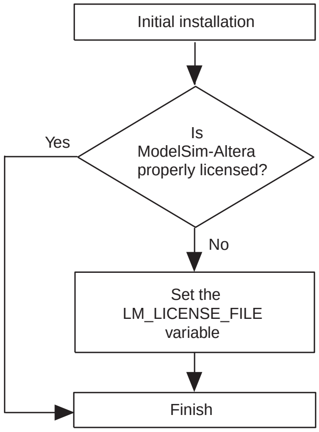
Software Licensing & Licensing Set-Up

License the ModelSim-Altera software through a software guard (T-guard) FIXEDPC license or a network FLOATNET or FLOATPC license. Each Altera software subscription includes a license to either VHDL or Verilog HDL. Network licenses with multiple users may have their licenses split between VHDL and Verilog HDL in any ratio.

Obtain licenses for ModelSim-Altera software from the Altera web site at **http://www.altera.com**. Get licensing information for Model Technology’s ModelSim directly from Model Technology. See Figure 4 for the set-up process.

☞ For ModelSim-Altera versions prior to 5.5b, use the PCLS utility, included with the software, to set up the license.

Figure 4. ModelSim-Altera Licensing Set-up Process


Initial installation
Is ModelSim-Altera properly licensed?
Yes
No
Set the LM_LICENSE_FILE variable
Finish

LM_LICENSE_FILE Variable

Altera recommends setting the LM_LICENSE_FILE environment variable to the location of the license file.

For details on licensing ModelSim-Altera, setting the LM_LICENSE_FILE variable, and using the PCLS Utility for ModelSim-Altera versions prior to 5.5b, see the ***Quartus II Installation and Licensing Manual*** or *AN 205: Understanding Altera Software Licensing*.

Conclusion

Using the ModelSim-Altera simulation software within the Altera PLD design flow enables Altera software users to easily and accurately perform functional and timing simulation on their designs. Proper verification of designs at the functional and post place-and-route stages using the ModelSim-Altera software helps ensure design functionality and success and ultimately a quick time to market.

Revision History

The information contained in *AN 204: Using ModelSim in a Quartus II Design Flow* version 1.2 supersedes information published in previous versions.

Version 1.2

The following changes were made to *AN 204: Using ModelSim in a Quartus II Design Flow* version 1.2:

- Updated procedures to reflect the ModelSim software version 5.6.
- Updated Tables 11, 12, 13, 14, and 15.

Version 1.1

The following changes were made to *AN 204: Using ModelSim in a Quartus II Design Flow* version 1.1:

- Updated Tables 11, 12, 13, 14, and 15.

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
http://www.altera.com
Applications Hotline:
(800) 800-EPLD
Literature Services:
lit_req@altera.com