


This chapter provides detailed instructions about how to simulate your Altera Quartus® II design in the ModelSim-Altera® software, Mentor Graphics® ModelSim software, and Mentor Graphics QuestaSim software.

Altera provides the ModelSim-Altera software to simplify design simulation with all readily precompiled Altera simulation libraries. The precompiled Altera simulation libraries support the simulation of designs that use Altera devices.

 For more information about ModelSim-Altera, refer to the [Model-Sim Altera Software](#) page of the Altera website.

The Quartus II software subscription includes the ModelSim-Altera Starter Edition, which is a no-cost entry-level version of the ModelSim-Altera Subscription Edition software. The ModelSim-Altera Subscription Edition software offers support for all Altera devices. Both versions are available on Windows and Linux platforms. You can use the ModelSim-Altera software to perform functional, post-synthesis, and gate-level timing simulations for either Verilog HDL or VHDL designs that target an Altera device.

 In this chapter, ModelSim refers to ModelSim SE, PE, and DE, which share the same commands as QuestaSim. ModelSim-Altera refers to ModelSim-Altera Starter Edition and ModelSim-Altera Subscription Edition software.

This chapter includes the following topics:

- [“Software Requirements” on page 2-2](#)
- [“Design Flow with ModelSim-Altera, ModelSim, or QuestaSim Software” on page 2-2](#)
- [“Simulation Libraries” on page 2-3](#)
- [“Simulating with the ModelSim-Altera Software” on page 2-4](#)
- [“Simulating with the ModelSim and QuestaSim Software” on page 2-5](#)
- [“Simulating Designs that Include Transceivers” on page 2-12](#)
- [“Using the NativeLink Feature with ModelSim-Altera, ModelSim, or QuestaSim Software” on page 2-17](#)
- [“Generating a Timing Value Change Dump File \(.vcd\) for the PowerPlay Power Analyzer” on page 2-18](#)
- [“Viewing a Waveform from a .wlf” on page 2-19](#)
- [“Simulating with ModelSim-Altera Waveform Editor” on page 2-20](#)

- “Scripting Support” on page 2–20
- “Software Licensing and Licensing Setup in ModelSim-Altera Subscription Edition” on page 2–22

Software Requirements

To simulate your design, you require the following software:


- ModelSim-Altera, ModelSim, or QuestaSim
- Compiled Altera simulation libraries
- Quartus II simulation netlists—Verilog Design File (.v), Verilog Output File (.vo), VHDL Design File (.vhd), VHDL Output File (.vho), and Synopsys Design Constraints File (.sdc)

 For more information about installing Altera software, refer to the *Altera Software Installation and Licensing* manual.

Design Flow with ModelSim-Altera, ModelSim, or QuestaSim Software

You can perform the following types of simulations with the ModelSim-Altera, ModelSim, or QuestaSim software:

- Functional simulation
- Post-synthesis simulation
- Gate-level timing simulation

 Some versions of ModelSim and QuestaSim support SystemVerilog, PSL assertions, SystemC, and more. For more information about the features supported in the different versions of ModelSim and QuestaSim, refer to Mentor Graphics literature or your Mentor Graphics contact.

You need a version of ModelSim that supports VHDL/Verilog HDL co-simulation to simulate designs that use transceivers in Stratix® V devices.

 For more information about the Quartus II software design flow, refer to the “Design Flow” section in the *Simulating Altera Designs* chapter in volume 3 of the *Quartus II Handbook*.

 For additional documentation about ModelSim-Altera, refer to the ModelSim-Altera Help that ships with the product. Click the **Help** button on the ModelSim-Altera toolbar.

Simulation Libraries

You are required to run functional simulations, post-synthesis simulations, or gate-level timing simulations. Unless you are using ModelSim-Altera, you must compile the necessary library files before you can run the simulation. The ModelSim-Altera software has precompiled Altera libraries. Do not recompile these libraries.

A few exceptions require you to compile gate-level timing simulation library files to perform functional simulation. For example, some Altera megafunctions require gate-level libraries to perform a functional simulation with third-party simulators.

Precompiled Simulation Libraries in the ModelSim-Altera Software

Precompiled libraries for both functional and gate-level simulations are provided for the ModelSim-Altera software. You should not compile these library files before running a simulation.


The precompiled libraries provided in `<ModelSim-Altera path>/altera` must be compatible with the version of the Quartus II software that is used to create the simulation netlist. To check whether the precompiled libraries are compatible with your version of the Quartus II software, refer to the `<ModelSim-Altera path>/altera/version.txt` file. This file shows which version and build of the Quartus II software was used to create the precompiled libraries.

- ❓ For a list of precompiled library names for all functional and gate-level simulation models, refer to *ModelSim-Altera Precompiled Libraries* in Quartus II Help.

Simulation Library Files in the Quartus II Software

In ModelSim and QuestaSim, no precompiled libraries are available. You must compile the necessary libraries to perform functional or gate-level simulation.

- ❓ For a list of all functional simulation library files in the Quartus II directory, refer to *Altera Functional Simulation Libraries* in Quartus II Help. For a list of all post-synthesis and post-fit (gate-level) library files in the Quartus II directory, refer to *Altera Post-Fit Libraries* in Quartus II Help. For a list of logical library names to compile for simulation models, refer to *Libraries For Altera Simulation Models* in Quartus II Help.

 Encrypted Altera simulation model files shipped with the Quartus II software version 10.1 and later can only be read by ModelSim-Altera Edition Software version 6.6c and later. These encrypted simulation model files are located at the `<Quartus II System directory>/quartus/eda/sim_lib/<eda simulation vendor>` directory, where `<eda simulation vendor>` is `aldec`, `cadence`, `mentor`, or `synopsys`.

Disabling Timing Violation on Registers

In certain situations, you can ignore a timing violation and disable timing violations on registers (for example, timing violations that occur in internal synchronization registers used for asynchronous clock domain crossing).

By default, the `x_on_violation_option logic` option is **On**, which means simulation shows “x” whenever a timing violation occurs. To disable showing the timing violation on certain registers, set the `x_on_violation_option logic` option to **Off** on those registers.

The following Quartus II Tcl command disables timing violation on registers. This Tcl command is also stored in the Quartus II Settings File (`.qsf`).

```
set_instance_assignment -name X_ON_VIOLATION_OPTION OFF -to <register_name>
```

Simulating with the ModelSim-Altera Software

You can perform simulation of Verilog HDL or VHDL designs with the ModelSim-Altera software at three levels: functional, post-synthesis, and gate-level.

For high-speed simulation, you must select a speed of 1 ps or above in the **Resolution** list for your simulator resolutions (**Design** tab of the **Start Simulation** dialog box). If you select a speed slower than 1 ps, the high-speed simulation may fail.

Setting Up a Quartus II Project for the ModelSim-Altera Software

The first steps in performing a simulation are starting the ModelSim-Altera software, changing to your project or simulation directory, and creating libraries for your design.

- ① For more information, refer to *Setting Up a Project with the ModelSim-Altera Software* in Quartus II Help.

Performing Functional Simulation

Functional simulation verifies code syntax and design functionality. The following sections describe how to perform functional simulation in the ModelSim-Altera software for a Verilog HDL or VHDL design.

- ① For information about performing a functional simulation with the ModelSim-Altera software, refer to *Performing a Functional Simulation with the ModelSim-Altera Software* in Quartus II Help.





The ModelSim-Altera software includes precompiled simulation libraries for Altera-provided models. You should not create simulation libraries and compile simulation models for the precompiled Altera libraries.

Performing Post-Synthesis Simulation

Post-synthesis simulation verifies design functionality is preserved after running Analysis & Synthesis flow in Quartus II software. To run post-synthesis simulation, you must generate post-synthesis netlists and simulate the design with the generated netlists.







For more information, refer to the “Generating Post-Synthesis Simulation Netlist Files” section in the *Simulating Altera Designs* chapter in volume 3 of the *Quartus II Handbook*.

-  The ModelSim-Altera software includes precompiled simulation libraries for Altera-provided models. You should not create simulation libraries and compile simulation models for the precompiled Altera libraries.
-  For information about performing a post-synthesis simulation with the ModelSim-Altera software, refer to *Performing a Timing Simulation with the ModelSim-Altera Software* in Quartus II Help.

Performing Gate-Level Timing Simulation

Gate-level timing simulation is an important step to ensure that the FPGA device's functionality is correct and meets all timing requirements after running the Fitter (Place & Route) flow in Quartus II software. To run gate-level simulation, you must generate gate-level timing simulation netlists and simulate your design with the generated netlists.

-  For more information, refer to the "Generating Gate-Level Timing Simulation Netlist Files" section in the *Simulating Altera Designs* chapter in volume 3 of the *Quartus II Handbook*.
-  The ModelSim-Altera software includes precompiled simulation libraries for Altera-provided models. You should not create simulation libraries and compile simulation models for the precompiled Altera libraries.
-  For more information about performing a gate-level simulation with the ModelSim-Altera software, refer to *Performing a Timing Simulation with the ModelSim-Altera Software* in Quartus II Help.
-  For additional documentation about ModelSim-Altera, refer to the ModelSim-Altera Help that ships with the product. Click the **Help** button on the ModelSim-Altera toolbar.

Simulating with the ModelSim and QuestaSim Software

You can simulate Verilog HDL or VHDL designs with the ModelSim and QuestaSim software at three levels: functional, post-synthesis, and gate-level.

You can perform the simulation with the GUI or from the command line. The following sections provide instructions to perform the simulation with the GUI and from the command line. You can proceed to the specific section that meets your needs.

For high-speed simulation, you must select a speed of 1 ps or above in the **Resolution** list for your simulator resolutions (**Design** tab of the **Start Simulation** dialog box). If you select a speed slower than 1 ps, the high-speed simulation may fail.

Simulating VHDL or Verilog HDL Designs with the GUI

This section provides information about performing functional, post-synthesis, and gate-level simulations of VHDL or Verilog HDL designs with the GUI.





Functional Simulation

This section provides information about compiling simulation models and performing a functional simulation.



Compiling Simulation Models into Simulation Libraries

In the Quartus II software, you can use the EDA Simulation Library Compiler tool to help you compile all Altera simulation libraries for Altera devices. The tool helps you compile all or selected Altera device family simulation libraries for ModelSim.


For VHDL, compile the `altera_mf_components.vhd` and `altera_mf.vhd` model files in the `altera_mf` library. Compile the `220pack.vhd` and `220model.vhd` model files in the `lpm` library. For Verilog HDL, compile the `altera_mf.v` model files in the `altera_mf_ver` library. Compile the `220model.v` model files in the `lpm_ver` library.

-  For more information about how to compile simulation models into simulation libraries with the EDA Simulation Library Compiler, refer to the “EDA Simulation Library Compiler” section in the *Simulating Altera Designs* chapter in volume 3 of the *Quartus II Handbook*.
-  For more information about how to compile simulation models into simulation libraries if you are not using the EDA Simulation Library Compiler, refer to *Compiling Libraries with the ModelSim Software* in Quartus II Help.
-  For more information about targeting a Stratix V device, refer to *Guidelines for Compiling Stratix V Libraries* in Quartus II Help.
-  You require the PCIe file only if you are using the IP Compiler for PCI Express® (hard IP implementation).

Performing the Simulation

-  For information about simulating VHDL designs with the GUI, refer to *Performing a Functional Simulation with the ModelSim Software* and *Performing a Functional Simulation with the QuestaSim Software* in Quartus II Help.
-  To see all of the functional simulation library files, refer to *Altera Functional Simulation Libraries* in Quartus II Help.

Post-Synthesis Simulation

-  You cannot perform post-synthesis or post-fit (gate-level) simulation if you are targeting the Stratix V device family.

Performing post-synthesis simulation enables you to verify that the design functionality is preserved after running Analysis & Synthesis in the Quartus II software. To run post-synthesis simulation, you must generate post-synthesis netlists and simulate the design with the generated netlists.

-  For more information, refer to the “Generating Post-Synthesis Simulation Netlist Files” section in the *Simulating Altera Designs* chapter in volume 3 of the *Quartus II Handbook*.

- ② For information about performing a post-synthesis simulation with the GUI, refer to *Performing a Timing Simulation with the ModelSim Software* and *Performing a Timing Simulation with the QuestaSim Software* in Quartus II Help.

Gate-Level Timing Simulation

- 👉 You cannot perform post-synthesis or post-fit (gate-level) simulation if you are targeting the Stratix V device family.

Gate-level simulation is a very important step to ensure that the functionality of your FPGA device is still correct and meets all required timing requirements after running the Fitter (Place & Route) flow in Quartus II software. To run gate-level simulation, you must generate gate-level timing simulation netlists and simulate your design with the generated netlists.

- 👉 For more information, refer to the “Generating Gate-Level Timing Simulation Netlist Files” section in the *Simulating Altera Designs* chapter in volume 3 of the *Quartus II Handbook*.
- ② For information about performing a gate-level simulation with the GUI, refer to *Performing a Timing Simulation with the ModelSim Software* and *Performing a Timing Simulation with the QuestaSim Software* in Quartus II Help.

Simulating VHDL or Verilog HDL Designs with the Command Line

This section provides information about performing functional, post-synthesis, and gate-level simulations of VHDL or Verilog HDL designs from the command line.

Simulating designs from the ModelSim and QuestaSim command line gives you more flexibility and control in compiling the libraries and loading and simulating the design files. All simulation commands are Tcl commands that can be coded into a `<filename>.do`, which allows you to run a simulation in batch mode. You have to run only `<filename>.do`, and the ModelSim and QuestaSim tool automatically runs all commands that are coded in the `<filename>.do` script macro file.

Functional Simulation

Functional simulation verifies code syntax and design functionality.

Below are examples of the commands used in a `<filename>.do` to perform functional simulation for VHDL or Verilog HDL designs. Use `lib1` to represent an Altera-provided library.

To create and compile an Altera library, type the following commands:

- For VHDL designs:

```
vlib <lib1> ←  
vmap <lib1> <lib1> ←  
vcom -work <lib1> <lib1>.vhd ←  
vlib <lib2> ←  
vmap <lib2> < lib2> ←  
vcom -work < lib2> < lib2>.vhd ←
```

- For Verilog HDL designs:

```
vlib <lib1>_ver ←
vmap <lib1>_ver <lib1>_ver ←
vcom -work <lib1> <lib1>.v ←
vlib <lib2>_ver ←
vmap <lib2>_ver <lib2>_ver ←
vcom -work <lib2>_ver <lib2>.v ←
```

To create the work library and compile the design and testbench files, type the following commands:

- For VHDL designs:

```
vlib work ←
vmap work work ←
vcom -work work <design_file1>.vhd <design_file2>.vhd <testbench_file>.vhd ←
```

- For Verilog HDL designs:

```
vlib work ←
vmap work work ←
vlog -work work <design_file1>.v <design_file2>.v <testbench_file>.v ←
```

To load the design, type the following command:

- For VHDL Designs

```
vsim -L work -L <lib1> -L <lib2> work.<testbench module name> ←
```

- For Verilog HDL Designs

```
vsim -L work -L <lib1>_ver -L <lib2>_ver work.<testbench module name> ←
```

To add signals to the waveform viewer and run the simulation, type the following commands:

```
add wave * ←
run ←
```

Examples:

Example 2-1. For VHDL Designs

```
# Create and compile Altera libraries

vlib altera_mf
vmap altera_mf altera_mf
vcom -work altera_mf altera_mf_components.vhd altera_mf.vhd
vlib lpm
vmap lpm lpm
vcom -work lpm 220pack.vhd 220model.vhd

# Create work library and compile design files and testbench file

vlib work
vmap work work
vcom -work work top_level.vhd adder.vhd testbench.vhd

# Load design

vsim -L work -L altera_mf -L lpm work.testbench

# add signals to the waveform viewer and run simulation

add wave *
run
```

Example 2-2. For Verilog HDL Designs

```
# Create and compile Altera libraries

vlib altera_mf_ver
vmap altera_mf_ver altera_mf_ver
vlog -work altera_mf_ver altera_mf.v
vlib lpm_ver
vmap lpm_ver lpm_ver
vlog -work lpm_ver 220model.v

# Create work library and compile design files and testbench file

vlib work
vmap work work
vlog -work work top_level.v adder.v testbench.v

# Load design


vsim -L work -L altera_mf_ver -L lpm_ver work.testbench

# add signals to the waveform viewer and run simulation


add wave *
run
```

- ① For more information on functional simulation libraries provided by Altera, refer to *Altera Functional Simulation Libraries* in Quartus II Help.
- ① For more information about targeting a Stratix V device, refer to *Guidelines for Compiling Stratix V Libraries* in Quartus II Help.

Post-Synthesis Simulation

-  You cannot perform post-synthesis or post-fit (gate-level) simulation if you are targeting the Stratix V device family.

Perform post-synthesis simulation to verify that design functionality is preserved after synthesis. Create the post-synthesis netlist in the Quartus II software and use the netlist to perform post-synthesis simulation with the ModelSim and QuestaSim software. Before running post-synthesis simulation, generate post-synthesis simulation netlist files.

-  For more information about how to generate the netlist, refer to the “Generating Post-Synthesis Simulation Netlist Files” section in the *Simulating Altera Designs* chapter in volume 3 of the *Quartus II Handbook*.
- ① For information about performing a post-synthesis simulation with the GUI, refer to *Performing a Timing Simulation with the ModelSim Software* and *Performing a Timing Simulation with the QuestaSim Software* in Quartus II Help.

Type the following commands to perform a post-synthesis simulation for VHDL or Verilog HDL designs. Use *lib1* to represent any Altera-provided libraries.

To create and compile Altera libraries, type the following commands:

- For VHDL designs:

```
vlib work ↵
vmap work work ↵
vcom -work work <output_netlist>.vho <testbench file>.vhd ↵
```

- For Verilog HDL designs:

```
vlib work ↵
vmap work work ↵
vlog -work work <output_netlist>.vo <testbench file>.v ↵
```

To load the design, type the following command:

- For VHDL designs:

```
vsim +transport_int_delays +transport_path_delays -L work -L \
<lib1>-L <lib2> work.<testbench module name> ↵
```

- For Verilog HDL designs:

```
vsim -t ps +transport_int_delays +transport_path_delays -L work -L \
<lib1>_ver -L <lib2>_ver work.<testbench module name> ↵
```

To add signals to the waveform viewer and to run simulation, type the following commands:

```
add wave * ↵
run ↵
```

Examples:

Example 2-3. For VHDL Designs

```
# Create and compile Altera libraries

vlib altera
vmap altera altera
vcom -work altera altera_primitives_components.vhd \
altera_primitives.vhd
vlib stratixiii
vmap stratixiii stratixiii
vcom -work stratixiii stratixiii.atoms.vhd stratixiii_components.vhd

# Create work library and compile design files and testbench file

vlib work
vmap work work
vcom -work work top_level.vho testbench.vhd

# Load design

vsim +transport_int_delays +transport_path_delays -L work -L \
altera -L stratixiii work.testbench

# add signals to the waveform viewer and run simulation

add wave *
run
```

Example 2-4. For Verilog HDL Designs

```
# Create and compile Altera libraries

vlib altera_ver
vmap altera_ver altera_ver
vlog -work altera_ver altera_primitives.v
vlib stratixiii_ver
vmap stratixiii_ver stratixiii_ver
vlog -work stratixiii_ver stratixiii_atoms.v

# Create work library and compile design files and testbench file


vlib work
vmap work work
vlog -work work top_level.vo testbench.v

# Load design


vsim +transport_int_delays +transport_path_delays -L work -L
altera_ver -L stratixiii_ver work.testbench

#add signals to the waveform viwer and run simulation

add wave *
run
```

-  For more information about Altera-provided post-fit libraries, refer to [Altera Post-Fit Libraries](#) in Quartus II Help.

Gate-Level Timing Simulation

-  You cannot perform post-synthesis or post-fit (gate-level) simulation if you are targeting the Stratix V device family.

The steps for gate-level timing simulation are similar with the steps for post-synthesis simulation, except for the following differences:

- For gate-level timing simulation, you must back-annotate the Standard Delay Format Output File (**.sdo**)
- For VHDL designs, you must add the **-sdftyp** option for back-annotating

Example 2-5.

```
vsim +transport_int_delays +transport_path_delays -sdftyp \  
<instance path to design> = <path to SDO file> -L work \  
-L stratixiii -L altera work.testbench
```

You do not have to set the value (minimum, average, maximum) for the ***.sdo**, because the Quartus II EDA Netlist Writer generates the ***.sdo** with the same value for the minimum, average, and maximum timing values.

If you instantiate your design in the testbench file under the **i1** label, the *<design instance>* should be "i1" (for example, /i1=<my design>.sdo).

For Verilog HDL designs, the back-annotating process is already set within the **output_netlist.vo** script. You are not required to back-annotate the **.sdo** again.

Passing Parameter Information from Verilog HDL to VHDL

You must use in-line parameters to pass values from Verilog HDL to VHDL. Using the defparam construct causes an error in simulation. In the example below:

```
lpm_add_sub_component (
    .dataa (dataa),
    .datab (datab),
    .result (sub_wire0)
);
defparam
lpm_add_sub_component.lpm_direction = "ADD",
lpm_add_sub_component.lpm_hint =
"ONE_INPUT_IS_CONSTANT=NO,CIN_USED=NO",
lpm_add_sub_component.lpm_type = "LPM_ADD_SUB",
lpm_add_sub_component.lpm_width = 12;
```

You will see the following error message:

```
# ** Error: (vsim-3043)
/apps2/home/users/bhlee/SPR_ADOQS/ADOQS10000935_IN_LINE_PARAMETER/lpm_
add_sub1.v(67): Unresolved reference to 'lpm_add_sub_component' in
lpm_add_sub_component.lpm_direction.

# Region: /IN_LINE_PARAMETER_vlg_vec_tst/i1/b2v_inst
```

This megafunction instantiation has been modified to use in-line parameters:

```
lpm_add_sub#(12, "SIGNED", "ADD", 0, "LPM_ADD_SUB", "ONE_INPUT_IS_CONSTANT=
NO,CIN_USED=NO")
lpm_add_sub_component (
    .dataa (dataa),
    .datab (datab),
    .result (sub_wire0)
);
```



The sequence of the parameters depends on the sequence of the GENERIC in the VHDL component declaration.

Speeding Up Simulation

By default, the ModelSim and QuestaSim software runs in a debug-optimized mode. To run the ModelSim and QuestaSim software in speed-optimized mode, add the following two vlog command-line switches:

```
vlog -fast -05
```

In this mode, module boundaries are flattened and loops are optimized, which eliminates levels of debugging hierarchy and may result in faster simulation. This switch is not supported in the ModelSim-Altera simulator.

Simulating Designs that Include Transceivers

If your design includes an Arria® GX, Arria II GX, Cyclone® IV, HardCopy® IV, Stratix GX, Stratix II GX, Stratix IV, or Stratix V transceiver, you must compile additional library files to perform functional or gate-level timing simulations.



You cannot perform post-synthesis or post-fit (gate-level) simulation if you are targeting the Stratix V device family.


For high-speed simulation, you must select a speed of 1 ps and above in the **Resolution** list for your simulator resolutions (**Design** tab of the **Start Simulation** dialog box). If you select a speed slower than 1 ps, the high-speed simulation may fail.

-  If you are using the IP Compiler for PCI Express (hard IP implementation) in your design, refer to the “Simulating the Design” section in the *IP Compiler for PCI Express User Guide*.

The following sections show you how to perform functional and gate-level timing simulation on transceiver devices. Command-line templates are provided. In these templates, cross-reference the values of **Library Name** and **Library File** with [Table 2-1 on page 2-14](#) and [Table 2-2 on page 2-16](#) according to a given transceiver device.

Functional Simulation

The following sections list the commands you need to type to perform a functional simulation for ModelSim-Altera and ModelSim or QuestaSim.

-  For Stratix V, you must compile the libraries listed in *Guidelines for Compiling Stratix V Libraries* in Quartus II Help.

ModelSim-Altera

The following examples show the commands you need to type to perform a functional simulation for ModelSim-Altera.

Example 2-6. For VHDL Designs

```
vcom -work <my_design>.vhd <my_testbench>.vhd  
vsim -L lpm -L altera_mf -L sgate \  
-L <Library Name> work.<my_testbench>
```

Example 2-7. For Verilog HDL Designs

```
vcom -work <my_design>.vhd <my_testbench>.vhd  
vlog -L lpm_ver -L altera_mf_ver -L sgate_ver \  
-L <Library Name>_ver work.<my_testbench>
```

ModelSim or QuestaSim

The following examples show the commands you need to type to perform a functional simulation for ModelSim or QuestaSim.

Example 2–8. For VHDL Designs

```
vcom -work altera_mf altera_mf_components.vhd altera_mf.vhd
vcom -work lpm 220pack.vhd 220model.vhd
vcom -work sgate sgate_pack.vhd sgate.vhd
vcom -work <Library Name> <Library File 1>.vhd \
<Library File 2>.vhd
vcom -work <my_design>.vhd <my_testbench>.vhd
vsim -L lpm -L altera_mf -L sgate -L <Library Name> work.<my_testbench>
```

Example 2–9. For Verilog HDL Designs

```
vcom -work altera_mf altera_mf_components.vhd altera_mf.vhd
vcom -work lpm 220pack.vhd 220model.vhd
vcom -work sgate sgate_pack.vhd sgate.vhd
vcom -work <Library Name> <Library File 1>.vhd \
<Library File 2>.vhd
vcom -work <my_design>.vhd <my_testbench>.vhd
vsim -L lpm -L altera_mf -L sgate -L <Library Name> work.<my_testbench>
```

Table 2–1. Functional Simulation Libraries for Transceiver Devices

Devices	Transceiver Libraries		Common Libraries
	Library Files to Compile	Logical Library Name	
Arria GX	arriagx_hssi_components.vhd arriagx_hssi_atoms (.vhd or .v)	arriagx_hssi	<ul style="list-style-type: none"> ■ altera_mf ■ lpm ■ sgate
Arria II GX	arriaii_hssi_components.vhd arriaii_hssi_atoms (.vhd or .v)	arriaii_hssi	
Cyclone IV	cycloneiv_hssi_components.vhd cycloneiv_hssi_atoms (.vhd or .v)	cycloneiv_hssi	
HardCopy IV	hardcopyiv_hssi_components.vhd hardcopyiv_hssi_atoms (.vhd or .v)	hardcopyiv_hssi	
Stratix GX	stratixgx_mf_components.vhd stratixgx_mf (.vhd or .v)	altgxb	
Stratix II GX	stratixiigx_hssi_components.vhd stratixiigx_hssi (.vhd or .v)	stratixiigx_hssi	
Stratix IV GX	stratixiv_hssi_components.vhd stratixiv_hssi (.vhd or .v)	stratixiv_hssi	
Stratix V GX	stratixv_hssi_atoms_ncrypt.v stratixv_hssi_components.vhd stratixv_hssi_atoms (.vhd or .v)	stratixv_hssi	

For a list of simulation library files to compile for the common libraries, refer to [Table 2–3 on page 2–17](#).

Gate-Level Timing Simulation

The following sections list the commands you need to type to perform a gate-level timing simulation for ModelSim-Altera and ModelSim or QuestaSim.

- ② For Stratix V, you must compile the libraries listed in *Guidelines for Compiling Stratix V Libraries* in Quartus II Help.

ModelSim-Altera

The following examples show the commands you need to type to perform a gate-level timing simulation for ModelSim-Altera.

Example 2-10. For VHDL Designs

```
vcom -work <my design>.vho <my testbench>.vhd
vsim -L lpm -L altera_mf -L sgate -L <Library Name 1> -L <Library Name 2> \
-sdftyp <design instance>=<path to .sdo file>.sdo work.<my testbench> \
-t ps - +transport_int_delays+transport_path_delays
```

Example 2-11. For Verilog HDL Designs

```
vlog -work <my design>.vo <my testbench>.v
vsim -L lpm_ver -L altera_mf_ver -L sgate_ver -L <Library Name 1>_ver -L \
<Library Name 2>_ver work.<my testbench> -t ps +transport_int_delays \
+transport_path_delays
```

ModelSim or QuestaSim

The following examples show the commands you need to type to perform a gate-level timing simulation for ModelSim or QuestaSim.

Example 2-12. For VHDL Designs

```
vcom -work lpm 220pack.vhd 220model.vhd
vcom -work altera_mf altera_mf_components.vhd altera_mf.vhd
vcom -work sgate sgate_pack.vhd sgate.vhd
vcom -work <Library Name 1> <Library File 1>.vhd <Library File 2>.vhd
vcom -work <Library Name 2> <Library File 1>.vhd <Library File 2>.vhd
vcom -work <my design>.vho <my testbench>.vhd
vsim -L lpm -L altera_mf -L sgate -L <Library Name 1> -L <Library Name 2> \
-sdftyp <design instance>=<path to .sdo file>.sdo work.<my testbench> \
-t ps +transport_int_delays +transport_path_delays
```

Example 2-13. For Verilog HDL Designs

```
vlog -work lpm_ver 220model.v
vlog -work altera_mf_ver altera_mf.v
vlog -work sgate_ver sgate.v
vlog -work <Library Name 1>_ver <Library File 1>.v
vlog -work <Library Name 2>_ver <Library File 2>.v
vlog -work <my design>.vo <my testbench>.v
vsim -L lpm_ver -L altera_mf_ver -L sgate_ver -L <Library Name 1>_ver \
-L <Library Name 2>_ver work.<my testbench> -t ps +transport_int_delays \
+transport_path_delays
```

Table 2-2 lists the gate-level timing simulation libraries for transceiver devices.

Table 2-2. Gate-Level Timing Simulation Libraries for Transceiver Devices

Devices	Transceiver Libraries		Common Libraries
	Library Files to Compile	Logical Library Name	
Arria GX	arriagx_components.vhd arriagx_atoms (.vhd or .v)	arriagx	<ul style="list-style-type: none"> ■ altera_mf ■ lpm ■ sgate
	arriagx_hssi_components.vhd arriagx_hssi_atoms (.vhd or .v)	arriagx_hssi	
Arria II GX	arriaii_components.vhd arriaii_atoms (.vhd or .v)	arriaii	
	arriaii_hssi_components.vhd arriaii_hssi_atoms (.vhd or .v)	arriaii_hssi	
Cyclone IV	cycloneiv_components.vhd cycloneiv_atoms (.vhd or .v)	cycloneiv	
	cycloneiv_hssi_components.vhd cycloneiv_hssi_atoms (.vhd or .v)	cycloneiv_hssi	
HardCopy IV	hardcopyiv_components.vhd hardcopyiv_atoms (.vhd or .v)	hardcopyiv	
	hardcopyiv_hssi_components.vhd hardcopyiv_hssi_atoms (.vhd or .v)	hardcopyiv_hssi	
Stratix GX	stratixgx_components.vhd stratixgx_atoms (.vhd or .v)	stratixgx	
	stratixgx_hssi_components.vhd stratixgx_hssi_atoms (.vhd or .v)	stratixgx_gxb	
Stratix II GX	stratixiigx_components.vhd stratixiigx_atoms (.vhd or .v)	stratixiigx	
	stratixiigx_hssi_components.vhd stratixiigx_hssi_atoms (.vhd or .v)	stratixiigx_hssi	
Stratix IV GX	stratixiv_components.vhd stratixiv_atoms (.vhd or .v)	stratixiv	
	stratixiv_hssi_components.vhd stratixiv_hssi_atoms (.vhd or .v)	stratixiv_hssi	
Stratix V GX	stratixiv_hssi_components.vhd stratixiv_atoms (.vhd or .v)	stratixiv	
	stratixv_hssi_atoms_ncrypt.v stratixv_hssi_components.vhd stratixv_hssi_atoms (.vhd or .v)	stratixv_hssi	

Table 2-3 lists the simulation library files to compile for the common libraries needed for all Altera transceiver designs.

Table 2-3. Common Libraries

Library Name	Library Files to Compile
altera_mf	altera_mf_components (.vhd or .v)
	altera_mf (.vhd or .v)
lpm	220pack (.vhd or .v)
	220model (.vhd or .v)
sgate	sgate_pack (.vhd or .v)
	sgate (.vhd or .v)

Transport Delays

By default, the ModelSim and QuestaSim software filters out all pulses that are shorter than the propagation delay between primitives. Turning on the **transport delay** options in the ModelSim and QuestaSim software prevents the simulation tool from filtering out these pulses.

Table 2-4 describes the transport delay options.

Table 2-4. Transport Delay Options

Option	Description
+transport_path_delays	Use this option when the pulses in your simulation are shorter than the delay within a gate-level primitive. You must include the +pulse_e/number and +pulse_r/number options.
+transport_int_delays	Use this option when the pulses in your simulation are shorter than the interconnect delay between gate-level primitives. You must include the +pulse_int_e/number and +pulse_int_r/number options.



The **+transport_path_delays** and **+transport_int_delays** options are also used by default in the NativeLink feature for gate-level timing simulation.




For more information about either of these options, refer to the ModelSim-Altera Command Reference installed with the ModelSim and QuestaSim software.

The following ModelSim and QuestaSim software command shows the command line syntax to perform a gate-level timing simulation with the device family library:

```
vsim -t lps -L stratixii -sdftyp /il=filtref_vhd.sdo work.filtref_vhd_vec_tst \
+transport_int_delays +transport_path_delays
```

Using the NativeLink Feature with ModelSim-Altera, ModelSim, or QuestaSim Software

The NativeLink feature in the Quartus II software facilitates the seamless transfer of information between the Quartus II software and EDA tools and allows you to run ModelSim and QuestaSim within the Quartus II software.

-  For more information, refer to the “Launching the EDA Simulator with the NativeLink Feature” section in the *Simulating Altera Designs* chapter in volume 3 of the *Quartus II Handbook*.

ModelSim and QuestaSim Error Message Information

ModelSim and QuestaSim error and warning messages are tagged with a `vsim` or `vcom` code. To determine the cause and resolution for a `vsim` or `vcom` error or warning, use the `verror` command.

For example, ModelSim and QuestaSim may display the following error message:

```
# ** Error:
C:/altera_trn/DUALPORT_TRY/simulation/modelsim/DUALPORT_TRY.vho(31):
(vcom-1136) Unknown identifier "stratixiii".
```

In this case, type the following command:

```
verror 1136 ←
```

At that point, the error message appears as follows:

```
# vcom Message # 1136:
# The specified name was referenced but was not found. This indicates
# that either the name specified does not exist or is not visible at
# this point in the code.
```

Generating a Timing Value Change Dump File (.vcd) for the PowerPlay Power Analyzer

To generate a timing Value Change Dump File (`.vcd`) for the PowerPlay Power Analyzer, you must first generate a `<filename>_dump_all_vcd_nodes.tcl` script file in the Quartus II software and run the `<filename>_dump_all_vcd_nodes.tcl` script file from the ModelSim, QuestaSim, or ModelSim-Altera software to generate a timing `<filename>.vcd`. The PowerPlay Power Analyzer can then use this timing `<filename>.vcd` for power analysis.

The generation of the `<filename>_dump_all_vcd_nodes.tcl` script file from the Quartus II software and the `<filename>.vcd` file from ModelSim using the script file can be integrated together so that the generation of the `<filename>.vcd` file can be achieved with a one-button push-automated process.

To set up the automated process flow to generate the `<filename>.vcd` file, perform the following steps:

1. Set up the EDA simulator execution path. For more information, refer to the “Setting Up the EDA Simulator Execution Path” section in the *Simulating Altera Designs* chapter in the Quartus II Handbook.
2. Configure the NativeLink settings for simulation. For more information, refer to the “Configuring NativeLink Settings” section in the *Simulating Altera Designs* chapter in the Quartus II Handbook.
3. When you are configuring the NativeLink settings, turn on the **Generate Value Change Dump (VCD) file script** option.
4. Set the top-level design instance name that you want to use in your testbench.
5. Once all settings are finalized, run a full compilation in the Quartus II software.

6. Start a gate-level timing simulation; in the Tools menu, select **Run EDA Simulation**, and then click **EDA Gate Level Simulation**.

Performing gate-level simulation generates the

`<filename>_dump_all_vcd_nodes.tcl` file, the ModelSim simulation

`<filename>_run_msim_gate_vhdl/verilog.do` file (which includes the `.vcd` and `.tcl` execution lines), and all the other necessary files to perform the simulation.

ModelSim then automatically launches and runs the generated `.do` to start the simulation.

7. Break the simulation if your testbench does not have a break point. End the simulation to have ModelSim generate the `.vcd`. You can only generate the `.vcd` after simulation ends with the **End Simulation** function.



For more information about using the timing `<filename>.vcd` for power estimation, refer to the *PowerPlay Power Analysis* chapter in volume 3 of the *Quartus II Handbook*.

Viewing a Waveform from a .wlf

A Wave Log Format File (`.wlf`) is automatically generated when your simulation is run. The `.wlf` is used for generating the waveform view through ModelSim-Altera, ModelSim, or QuestaSim.

To view a waveform from a `.wlf` through ModelSim-Altera, ModelSim, or QuestaSim, perform the following steps:

1. Type `vsim` at the command line. The **ModelSim/QuartaSim** or **ModelSim-Altera** dialog box appears.
2. On the File menu, click **Datasets**. The **Datasets Browser** dialog box appears.
3. Click **Open** and browse to the directory that contains your `.wlf`.
4. Select the `.wlf` file and click **Open**, then click **OK**.
5. Click **Done**.
6. In the Object browser, select the signals that you want to observe.
7. On the Add menu, click **Wave** and then click **Selected Signals**.

You cannot view a waveform from a `.vcd` in ModelSim-Altera, ModelSim, or QuestaSim directly. The `.vcd` must first be converted to a `.wlf`.

1. Use the `vcd2wlf` command to convert the file. For example, type the following at the command-line:


```
vcd2wlf <example>.vcd <example>.wlf ↵
```

2. After you convert the `.vcd` to a `.wlf`, follow the procedures for viewing a waveform from a `.wlf` through ModelSim and QuestaSim.

You can also convert your `.wlf` to a `.vcd` by using the `wlf2vcd` command.



Simulating with ModelSim-Altera Waveform Editor

You can use the ModelSim-Altera Waveform Editor as a simple method to create a design stimulus for simulation. You can create this design stimulus via interactive manipulation of waveforms from the wave window in ModelSim-Altera. With the ModelSim-Altera waveform editor, you can create and edit waveforms, drive simulation directly from created waveforms, and save created waveforms into a stimulus file.

-  For more information, refer to the *Generating Stimulus with Waveform Editor* chapter in the *ModelSim SE User's Manual* available on the ModelSim website (www.model.com).

Scripting Support

You can run procedures and create settings described in this chapter in a Tcl script. You can also run some procedures at the command line prompt.

-  For more information about Tcl scripting, refer to the *Tcl Scripting* chapter in volume 2 of the *Quartus II Handbook*.
-  For more information about command line scripting, refer to the *Command-Line Scripting* chapter in volume 2 of the *Quartus II Handbook*.

For detailed information about scripting command options, refer to the Quartus II Help command line and Tcl API help browser. To access this information, type the following command to start a help browser:

```
quartus_sh --qhelp ↵
```

Generating a Post-Synthesis Simulation Netlist for ModelSim and QuestaSim

You can use the Quartus II software to generate a post-synthesis simulation netlist with Tcl commands or with a command at the command-line prompt. The following example assumes that you are selecting ModelSim and QuestaSim (Verilog HDL output from the Quartus II software).

Tcl Commands

Use the following Tcl commands to set the output format to Verilog HDL, to set the simulation tool to ModelSim and QuestaSim for Verilog HDL, and to generate a functional netlist:

```
set_global_assignment-name EDA_SIMULATION_TOOL "ModelSim (Verilog)" ↵  
set_global_assignment-name EDA_GENERATE_FUNCTIONAL_NETLIST ON ↵
```

or

```
set_global_assignment-name EDA_SIMULATION_TOOL "QuestaSim (Verilog)" ↵  
set_global_assignment-name EDA_GENERATE_FUNCTIONAL_NETLIST ON ↵
```

Command Prompt

Use the following command to generate a simulation output file for the ModelSim and QuestaSim simulator. Specify VHDL or Verilog HDL for the format:

```
quartus_eda <project name> --simulation=on --format=<format> \  
--tool=ModelSim --functional ←
```

or

```
quartus_eda <project name> --simulation=on --format=<format> \  
--tool=QuestaSim --functional ←
```

Generating a Gate-Level Timing Simulation Netlist for ModelSim and QuestaSim

Use the Quartus II software to generate a gate-level timing simulation netlist with Tcl commands or with a command at the command prompt.

Tcl Commands

Use one of the following Tcl commands:

- `set_global_assignment -name EDA_SIMULATION_TOOL \
"ModelSim-Altera (Verilog)" ←`
or
`set_global_assignment -name EDA_SIMULATION_TOOL \
"QuestaSim-Altera (Verilog)" ←`
- `set_global_assignment -name EDA_SIMULATION_TOOL \
"ModelSim-Altera (VHDL)" ←`
or
`set_global_assignment -name EDA_SIMULATION_TOOL \
"QuestaSim-Altera (VHDL)" ←`
- `set_global_assignment -name EDA_SIMULATION_TOOL "ModelSim (Verilog)" ←`
or
`set_global_assignment -name EDA_SIMULATION_TOOL "QuestaSim (Verilog)" ←`
- `set_global_assignment -name EDA_SIMULATION_TOOL "ModelSim (VHDL)" ←`
or
`set_global_assignment -name EDA_SIMULATION_TOOL "QuestaSim (VHDL)" ←`

Command Line






Generate a simulation output file for the ModelSim and QuestaSim simulator by specifying VHDL or Verilog HDL for the format by typing the following command at the command prompt:

```
quartus_eda <project name> --simulation=on --format=<format> \  
--tool=ModelSim ←
```

or

```
quartus_eda <project name> --simulation=on --format=<format> \  
--tool=QuestaSim ←
```

Software Licensing and Licensing Setup in ModelSim-Altera Subscription Edition

-  For more information about the ModelSim-Altera Subscription Edition software, including pricing, refer to the [ModelSim-Altera Software](#) page of the Altera website.
-  For more information about obtaining and setting up the license for the ModelSim-Altera Subscription Edition software, refer to the “Licensing Altera Software” section in the [Altera Software Installation and Licensing Manual](#).
-  Currently, Altera does not support companion licensing for ModelSim AE.
-  The USB software guard is not supported by versions earlier than Mentor Graphics ModelSim software version 5.8d.
-  For ModelSim-Altera software versions prior to 5.5b, use the **PCLS** utility included with the software to set up the license.

Conclusion

Using the ModelSim, QuestaSim, and ModelSim-Altera simulation software within the Altera FPGA design flow enables you to easily and accurately perform functional simulations, post-synthesis simulations, and gate-level simulations on their designs. Proper verification of designs at the functional, post-synthesis, and post place-and-route stages with the ModelSim, QuestaSim, and ModelSim-Altera software helps ensure design functionality and success and, ultimately, a quick time-to-market.

Document Revision History


Table 2-5 shows the revision history for this chapter.

Table 2-5. Document Revision History (Part 1 of 2)

Date	Version	Changes
November 2011	11.1.0	<ul style="list-style-type: none"> ■ Added information about encrypted Altera simulation model files in “Simulation Library Files in the Quartus II Software” on page 2-3 ■ Updated Table 2-1 on page 2-14 and Table 2-2 on page 2-16 ■ Updated “Generating a Timing Value Change Dump File (.vcd) for the PowerPlay Power Analyzer” on page 2-18 ■ Added information in “Software Licensing and Licensing Setup in ModelSim-Altera Subscription Edition” on page 2-22
May 2011	11.0.0	<ul style="list-style-type: none"> ■ Updated “Software Requirements” on page 2-2 ■ Updated “Design Flow with ModelSim-Altera, ModelSim, or QuestaSim Software” on page 2-2 ■ Restructured “Simulating with the ModelSim-Altera Software” on page 2-4 ■ Restructured “Simulating with the ModelSim and QuestaSim Software” on page 2-5 ■ Restructured “Simulating Designs that Include Transceivers” on page 2-12 ■ Changed section name from “ModelSim and QuestaSim Error Message Verification” to “ModelSim and QuestaSim Error Message Information” on page 2-18 ■ Changed section name from “Simulating with ModelSim-Altera Waveform” to “Simulating with ModelSim-Altera Waveform Editor” on page 2-20
December 2010	10.1.0	<ul style="list-style-type: none"> ■ Changed to new document template ■ Referenced Simulating Altera Designs chapter ■ Added new section, “Simulating with ModelSim-Altera Waveform Editor” on page 2-20 ■ Removed Stratix V compilation information and linked to Quartus II Help
July 2010	10.0.0	<ul style="list-style-type: none"> ■ Removed simulation library tables and linked to Quartus II Help ■ Added other links to Quartus II Help and ModelSim-Altera Help where appropriate and removed redundant information ■ Added QuestaSim support ■ Added Stratix V simulation information ■ Minor editorial changes throughout ■ Removed Referenced Documents section
November 2009	9.1.0	<ul style="list-style-type: none"> ■ Removed NativeLink information and referenced new <i>Simulating Designs with EDA Tools</i> chapter ■ Added Stratix IV transceiver simulation section ■ Reformatted transceiver simulation sections ■ Text edits throughout chapter

Table 2-5. Document Revision History (Part 2 of 2)

Date	Version	Changes
March 2009	9.0.0	Added the following sections: <ul style="list-style-type: none"> ■ “Compile Libraries Using the EDA Simulation Library Compiler” on page 2-17 ■ “Generate Simulation Script from EDA Netlist Writer” on page 2-77 ■ “Viewing a Waveform from a .wlf File” on page 2-78 Updated the following: <ul style="list-style-type: none"> ■ Table 2-1, Table 2-2, Table 2-5, Table 2-6, Table 2-7, Table 2-8, Table 2-9, Table 2-10 ■ Figure 2-4 on page 2-81 ■ All sections titled “Loading the Design”
November 2008	8.1.0	Updated the following: <ul style="list-style-type: none"> ■ Table 2-2, Table 2-3, Table 2-4, Table 2-5, Table 2-6 ■ Removed <code>--zero_ic_delays</code> from <code>quartus_sta</code> option in “Generate Post-Synthesis Simulation Netlist Files” on page 2-11 ■ Removed steps to include the library when the simulation is run in VHDL mode from all procedures; this is no longer necessary ■ Added information about the Altera Simulation Library Compiler throughout the chapter ■ Added “Compile Libraries Using the Altera Simulation Library Compiler” on page 2-15 ■ Added “Disabling Simulation” on page 2-72 ■ Minor editorial updates ■ Updated entire chapter using 8½” × 11” chapter template
May 2008	8.0.0	Updated the following: <ul style="list-style-type: none"> ■ “Altera Design Flow with ModelSim-Altera or ModelSim Software” on page 2-3 ■ “Simulation Libraries” on page 2-4 ■ “Simulation Netlist Files” on page 2-11 ■ “Perform Simulation Using ModelSim-Altera Software” on page 2-15 ■ “Perform Simulation Using ModelSim Software” on page 2-33 ■ “Simulating Designs that Include Transceivers” on page 2-57 ■ “Using the NativeLink Feature with ModelSim-Altera or ModelSim Software” on page 2-63 ■ “Generating a Timing VCD File for PowerPlay” on page 2-68

 For previous versions of the *Quartus II Handbook*, refer to the [Quartus II Handbook Archive](#).

 Take an [online survey](#) to provide feedback about this handbook chapter.