5. Aldec Active-HDL and Riviera-PRO Support

This chapter provides specific guidelines for simulation of Quartus® II designs with the Aldec Active-HDL or Riviera-PRO software. You can also refer to the following for more information about EDA simulation:

- For overview and version support information, Simulating Altera Designs in the Quartus II Handbook and About Using EDA Simulators in Quartus II Help.
- For detailed GUI steps, Preparing for EDA Simulation and Running EDA Simulators in Quartus II Help.

Quick Start Example (Active-HDL VHDL)

You can adapt the following RTL simulation example to get started quickly with Active-HDL:

1. Specify your EDA simulator and executable path in the Quartus II software:
   ```
   set_user_option -name EDA_TOOL_PATH_ACTIVEHDL <active-hdl executable path>
   set_global_assignment -name EDA_SIMULATION_TOOL "Active-HDL (VHDL)"
   ```

2. Compile simulation model libraries using one of the following:
   - Use NativeLink to compile required design files, simulation models, and run your simulator. Verify results in your simulator. Skip steps 3 through 6.
   - Use Simulation Library Compiler to compile all required simulation models.
   - Compile Altera simulation models manually:
     ```
     vlib <library1> <altera_library1>
     vcom -strict93 -dbg -work <library1> <lib1_component/pack.vhd> <lib1.vhd>
     ```

3. Create and open the workspace:
   ```
   createdesign <workspace name> <workspace path>
   opendesign -a <workspace name>.adf
   ```

4. Create the work library and compile the netlist and testbench files:
   ```
   vlib work
   vcom -strict93 -dbg -work work <output netlist> <testbench file>
   ```

5. Load the design:
   ```
   vsim +access+r -t 1ps +transport_int_delays +transport_path_delays \
   -L work -L <lib1> -L <lib2> work.<testbench module name>
   ```

6. Run the simulation in the Active-HDL simulator.
Active-HDL and Riviera-PRO Guidelines

The following guidelines apply to simulating Altera designs in the Active-HDL or Riviera-PRO software.

Compiling SystemVerilog Files

If your design includes multiple SystemVerilog files, you must compile the System Verilog files together with a single alog command.

If you have Verilog files and SystemVerilog files in your design, it is recommended that you compile the Verilog files, and then compile only the SystemVerilog files in the single alog command.

Simulating Transport Delays

By default, the Active-HDL or Riviera-PRO software filters out all pulses that are shorter than the propagation delay between primitives. Turning on the transport delay options in the Active-HDL or Riviera-PRO software prevents the simulation tool from filtering out these pulses.

Table 5–1 describes the transport delay options.

Table 5–1. Transport Delay Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+transport_path_delays</td>
<td>Use when simulation pulses are shorter than the delay in a gate-level primitive. You must include the +pulse_e/number and +pulse_r/number options.</td>
</tr>
<tr>
<td>+transport_int_delays</td>
<td>Use when simulation pulses are shorter than the interconnect delay between gate-level primitives. You must include the +pulse_int_e/number and +pulse_int_r/number options.</td>
</tr>
</tbody>
</table>

The +transport_path_delays and +transport_path_delays options apply by default during NativeLink gate-level timing simulation.

For more information about either of these options, refer to the Active-HDL online documentation installed with the Active-HDL software.

To perform a gate-level timing simulation with the device family library, type the Active-HDL command shown in Example 5–1.

Example 5–1.

```
vsim -t 1ps -L stratixii -sdftyp /il=filtref_vhd.sdo /
work.filtref_vhd_vec_tst +transport_int_delays +transport_path_delays
```

Disabling Timing Violation on Registers

In certain situations, you may want to ignore timing violations on registers and disable the “X” propagation that occurs (for example, timing violations in internal synchronization registers in asynchronous clock-domain crossing).
By default, the `x_on_violation_option` logic option applying to all design registers is `On`, resulting in an output of “X” at timing violation. To disable “X” propagation at timing violations on a specific register, set the `x_on_violation_option` option to `Off` for that register. The following command is an example from the Quartus II Settings File (.qsf):

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

For VHDL designs, the back-annotating process is done by adding the `-sdftyp` option.

**Example**

```
vsim +access +r -t lps +transport_int_delays +transport_path_delays -sdftyp <instance path to design>=<path to SDO file> -L adder -L work 
-L lpm -L altera_mf work.adder_vhd_vec_tst
```

### Using Simulation Setup Scripts

The Quartus II software can generate a `rivierapro_setup.tcl` simulation setup script for IP cores in your design. The use and content of the script file is similar to the `msim_setup.tcl` file described in the Mentor Graphics ModelSim and QuestaSim Support chapter of the Quartus II Handbook.

### Table 5–2. Document Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>November 2012</td>
<td>12.1.0</td>
<td>Relocated general simulation information to <em>Simulating Altera Designs.</em></td>
</tr>
<tr>
<td>June 2012</td>
<td>12.0.0</td>
<td>Removed survey link.</td>
</tr>
<tr>
<td>November 2011</td>
<td>11.0.1</td>
<td>Template update.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Minor editorial updates.</td>
</tr>
<tr>
<td>May 2011</td>
<td>11.0.0</td>
<td>Linked to Help for Stratix V Libraries.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reorganized and reformatted chapter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Other minor changes throughout.</td>
</tr>
<tr>
<td>December 2010</td>
<td>10.0.1</td>
<td>Changed to new document template. No change to content.</td>
</tr>
<tr>
<td>July 2010</td>
<td>10.0.0</td>
<td>Linked to Quartus II Help</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Revised simulation procedures</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added Stratix V simulation information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added Riviera-PRO support</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Minor text edits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Removed Referenced Documents section</td>
</tr>
</tbody>
</table>
### Table 5–2. Document Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| November 2000 | 9.1.0   | ■ Updated Table 6–1  
■ Removed Simulation Library tables and EDA Simulation Library Compiler sections and referenced new *Simulating Designs with EDA Tools* chapter  
■ Added “RTL Functional Simulation for Stratix IV Devices” and “Gate-Level Timing Simulation for Stratix IV Devices” sections  
■ Minor text edits |
| March 2009  | 9.0.0   | ■ Removed “Compile Libraries Using the Altera Simulation Library Compiler”  
■ Added “Compile Libraries Using the EDA Simulation Library Compiler” on page 5–10  
■ Added “Generate Simulation Script from EDA Netlist Writer” on page 5–51  
■ Minor editorial updates |
| November 2008 | 8.1.0   | Added the following sections:  
■ “Compile Libraries Using the Altera Simulation Library Compiler” on page 5–10  
■ Added steps to the procedure “Performing an RTL Simulation Using NativeLink” on page 5–45 for using the Altera Simulation Library Compilation  
■ Added steps to the procedure “Performing a Gate-Level Timing Simulation Using NativeLink” on page 5–47 for using the Altera Simulation Library Compilation  
■ Minor editorial updates  
■ Updated entire chapter using 8½” × 11” chapter template |
| May 2008    | 8.0.0   | Initial release |

For previous versions of the *Quartus II Handbook*, refer to the *Quartus II Handbook Archive*. 