



ModelSim* - Intel® FPGA Edition Simulation Quick-Start

Intel® Quartus® Prime Pro Edition

Updated for Intel® Quartus® Prime Design Suite: **19.4**



UG-20093 | 2019.12.30

Latest document on the web: [PDF](#) | [HTML](#)



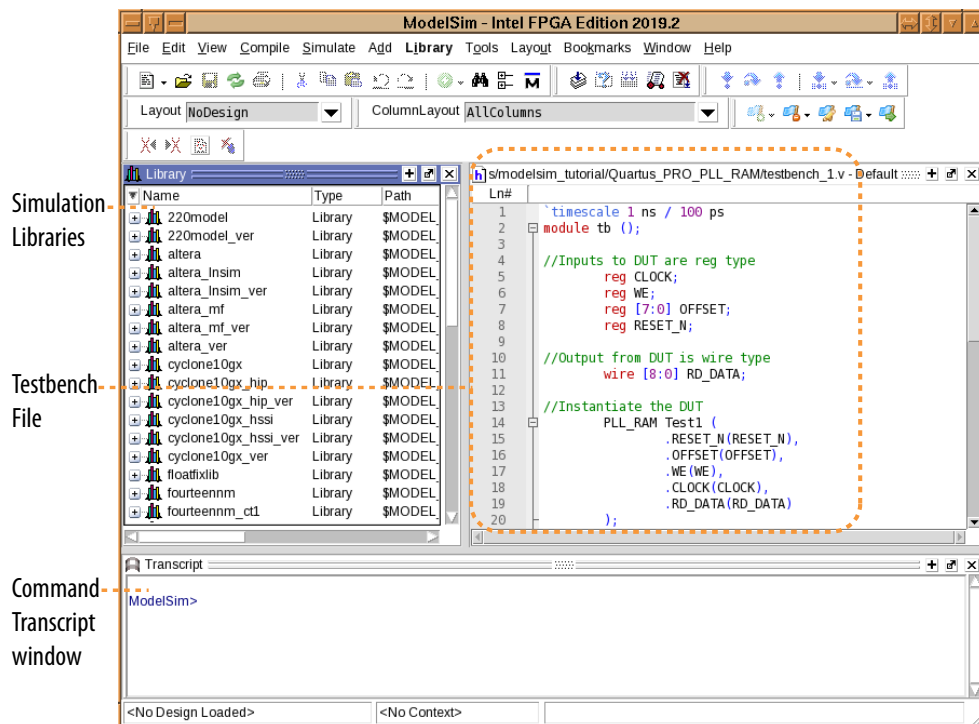
Contents

1. ModelSim* - Intel® FPGA Edition Simulation Quick-Start Intel® Quartus® Prime Pro Edition.....	3
1.1. Open the Example Design.....	4
1.2. Specify EDA Tool Settings.....	4
1.3. Generate a Simulator Setup Script Template.....	5
1.4. Modify the Simulator Setup Script.....	6
1.5. Compile and Simulate the Design.....	8
1.6. View Signal Waveforms.....	9
1.7. Add Signals to the Simulation.....	11
1.8. Rerun Simulation.....	12
1.9. Modify the Simulation Testbench.....	12
2. ModelSim - Intel FPGA Edition Simulation Quick-Start Revision History.....	14

1. ModelSim* - Intel® FPGA Edition Simulation Quick-Start Intel® Quartus® Prime Pro Edition

This document demonstrates how to simulate an Intel® Quartus® Prime Pro Edition design in the ModelSim* - Intel FPGA Edition simulator. Design simulation verifies your design before device programming. The Intel Quartus Prime software generates simulation files for supported EDA simulators during design compilation.

Figure 1. ModelSim - Intel FPGA Edition



Design simulation involves generating simulation files, compiling simulation models, running the simulation, and viewing the results. The following steps describe this flow:

1. [Open the Example Design](#) on page 4
2. [Specify EDA Tool Settings](#) on page 4
3. [Generate a Simulator Setup Script Template](#) on page 5
4. [Modify the Simulator Setup Script](#) on page 6
5. [Compile and Simulate the Design](#) on page 8
6. [View Signal Waveforms](#) on page 9



7. [Add Signals to the Simulation](#) on page 11
8. [Rerun Simulation](#) on page 12
9. [Modify the Simulation Testbench](#) on page 12

1.1. Open the Example Design

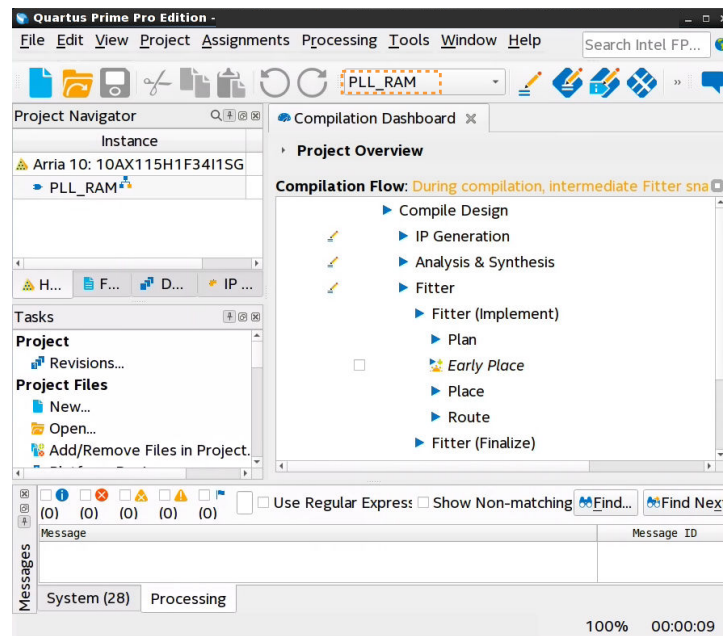
The PLL_RAM example design includes Intel FPGA IP cores to demonstrate the basic simulation flow. Download the example design files and open the project in the Intel Quartus Prime software.

Note:

This Quick-Start requires a basic understanding of hardware description language syntax and the Intel Quartus Prime design flow, as the [Intel Quartus Prime Pro Edition Foundation Online Training](#) describes.

1. Download and unzip the [Quartus_Pro_PLL_RAM.zip](#) design example.
2. Launch the Intel Quartus Prime Pro Edition software version 19.4 or later.
3. To open the example design project, click **File** ► **Open Project**, select the **pll_ram.qpf** project file, and then click **OK**.

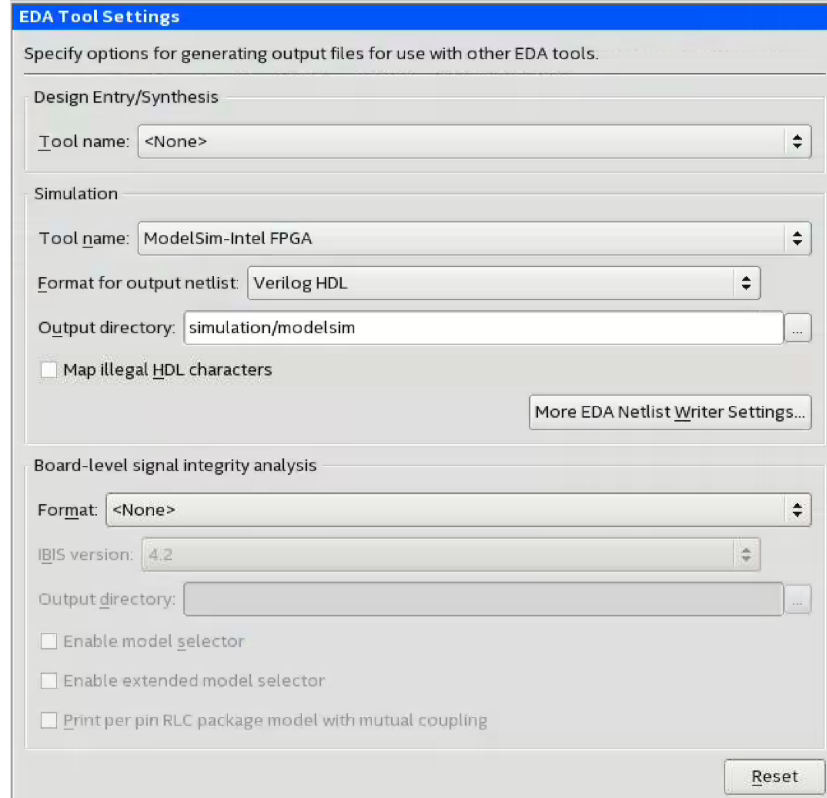
Figure 2. pll_ram Project in the Intel Quartus Prime Pro Edition



1.2. Specify EDA Tool Settings

Specify EDA tool settings to generate simulation files for supported simulators.

1. In the Intel Quartus Prime software, click **Assignments** ► **Settings** ► **EDA Tool Settings**.
2. Under **Simulation**, select **ModelSim-Intel FPGA** as the **Tool name**. Retain the default settings for **Format for output netlist** and **Output directory**.

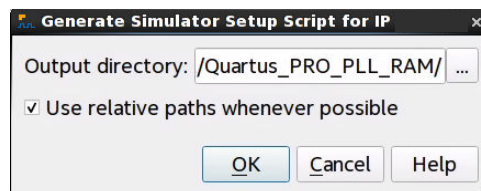


1.3. Generate a Simulator Setup Script Template

Simulator setup scripts help you to simulate the IP cores in your design. Follow these steps to generate the vendor-specific simulator setup script template for the IP modules in the example design. You can then customize this template for your specific simulation goals.

1. To compile the design, click **Processing > Start Compilation**. The Messages window indicates when compilation is complete.
2. Click **Tools > Generate Simulator Setup Script for IP**. Retain the default **Output directory** and **Use relative paths whenever possible** setting for the setup script file. The setup script template generates in the directory that you specify.

Figure 3. Generate Simulator Setup Scripts IP Dialog Box



1.4. Modify the Simulator Setup Script

Modify the generated simulator setup script to enable specific commands that simulate the IP cores in the project.

1. In a text editor, open the `/PLL_RAM/mentor/msim_setup.tcl` file.
2. Create a new text file with the name `mentor_example.do` and save it in the `/PLL_RAM/mentor/` directory.
3. In the `msim_setup.tcl` file, copy the section of code enclosed within the `TOP-LEVEL TEMPLATE - BEGIN` and `TOP-LEVEL TEMPLATE - END` comments, and then paste this code into the new `mentor_example.do` file.
4. In the `mentor_example.do` file, delete the single pound (`#`) characters preceding the following highlighted lines to enable compilation commands:

Figure 4. Uncomment Highlighted Simulation Commands in the Script

```
##
## set QSYS_SIMDIR <script generation output directory>
##
## Source the generated IP simulation script.
## source $QSYS_SIMDIR/mentor/msim_setup.tcl
##
## Set any compilation options you require (this is unusual).
## set USER_DEFINED_COMPILE_OPTIONS <compilation options>
## set USER_DEFINED_VHDL_COMPILE_OPTIONS <compilation options for VHDL>
## set USER_DEFINED_VERILOG_COMPILE_OPTIONS <compilation options for Verilog>
##
## Call command to compile the Quartus EDA simulation library.
## dev_com
##
## Call command to compile the Quartus-generated IP simulation files.
## com
##
## Add commands to compile all design files and testbench files, including
## the top level. (These are all the files required for simulation other
## than the files compiled by the Quartus-generated IP simulation script)
##
## vlog <compilation options> <design and testbench files>
##
## Set the top-level simulation or testbench module/entity name, which is
## used by the elab command to elaborate the top level.
##
## set TOP_LEVEL_NAME <simulation top>
##
## Set any elaboration options you require.
## set USER_DEFINED_ELAB_OPTIONS <elaboration options>
##
## Call command to elaborate your design and testbench.
## elab
##
## Run the simulation.
## run -a
##
```

5. Replace the following lines in the `mentor_example.do` script:



Table 1. Specify Values in the mentor_example.do Script

Replace this Line	With this Line
<pre>set QSYS_SIMDIR <script generation output directory></pre>	<pre>../</pre>
<pre>vlog <compilation options> <design and testbench files></pre>	<pre>vlog -vlog01compat -work work ../PLL_RAM.v vlog -vlog01compat -work work ../UP_COUNTER_IP/UP_COUNTER_IP.v vlog -vlog01compat -work work ../DOWN_COUNTER_IP/DOWN_COUNTER_IP.v vlog -vlog01compat -work work ../ClockPLL/ClockPLL.v vlog -vlog01compat -work work ../RAMhub/RAMhub.v vlog -vlog01compat -work work ../testbench_1.v</pre>
<pre>set TOP_LEVEL_NAME <simulation top></pre>	<pre>set TOP_LEVEL_NAME tb</pre>
<pre>run -a</pre>	<pre>add wave * view structure view signals run -all</pre>

6. Save the /PLL_RAM/mentor/mentor_example.do file. The following figure shows the mentor_example.do file after revisions are complete:



Figure 5. Completed Top-Level IP Simulation Setup Script

```
Ln# |
1  | ## TOP-LEVEL TEMPLATE - BEGIN
2  | ##
3  | ## QSYS_SIMDIR is used in the Quartus-generated IP simulation script to
4  | ## construct paths to the files required to simulate the IP in your Quartus
5  | ## project. By default, the IP script assumes that you are launching the
6  | ## simulator from the IP script location. If launching from another
7  | ## location, set QSYS_SIMDIR to the output directory you specified when you
8  | ## generated the IP script, relative to the directory from which you launch
9  | ## the simulator.
10 | ##
11 | set QSYS_SIMDIR ../
12 | ##
13 | ## Source the generated IP simulation script.
14 | source $QSYS_SIMDIR/mentor/msim_setup.tcl
15 | ##
16 | ## Set any compilation options you require (this is unusual).
17 | ## set USER_DEFINED_COMPILE_OPTIONS <compilation options>
18 | ## set USER_DEFINED_VHDL_COMPILE_OPTIONS <compilation options for VHDL>
19 | ## set USER_DEFINED_VERILOG_COMPILE_OPTIONS <compilation options for Verilog>
20 | ##
21 | ## Call command to compile the Quartus EDA simulation library.
22 | dev_com
23 | ##
24 | ## Call command to compile the Quartus-generated IP simulation files.
25 | com
26 | ##
27 | ## Add commands to compile all design files and testbench files, including
28 | ## the top level. (These are all the files required for simulation other
29 | ## than the files compiled by the Quartus-generated IP simulation script)
30 | ##
31 | vlog -vlog01compat -work work ../PLL_RAM.v
32 | vlog -vlog01compat -work work ../UP_COUNTER_IP/UP_COUNTER_IP.v
33 | vlog -vlog01compat -work work ../DOWN_COUNTER_IP/DOWN_COUNTER_IP.v
34 | vlog -vlog01compat -work work ../ClockPLL/ClockPLL.v
35 | vlog -vlog01compat -work work ../RAMhub/RAMhub.v
36 | vlog -vlog01compat -work work ../testbench_1.v
37 | ##
38 | ## Set the top-level simulation or testbench module/entity name, which is
39 | ## used by the elab command to elaborate the top level.
40 | ##
41 | set TOP_LEVEL_NAME tb
42 | ##
43 | ## Set any elaboration options you require.
44 | ## set USER_DEFINED_ELAB_OPTIONS <elaboration options>
45 | ##
46 | ## Call command to elaborate your design and testbench.
47 | elab
48 | ##
49 | ## Run the simulation.
50 | add wave *
51 | view structure
52 | view signals
53 | run -all
54 | ##
55 | ## Report success to the shell.
56 | ## exit -code 0
57 | ##
58 | ## TOP-LEVEL TEMPLATE - END
```

1.5. Compile and Simulate the Design

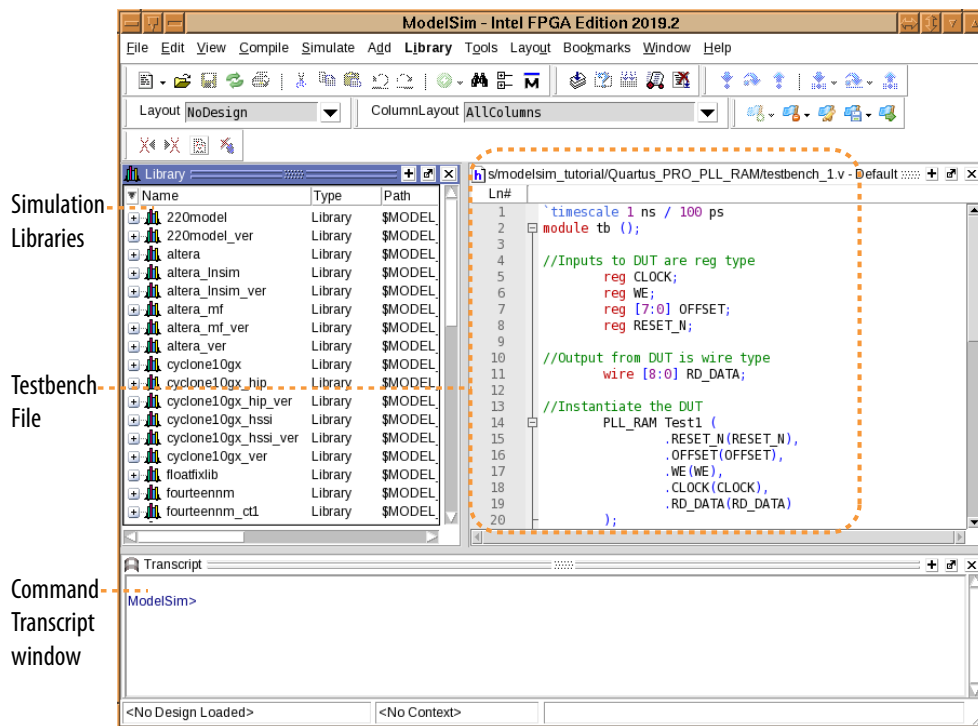
Run the top-level `mentor_example.do` script in the ModelSim - Intel FPGA Edition software to compile and simulate your design.



1. Launch the ModelSim - Intel FPGA Edition software. The ModelSim - Intel FPGA Edition GUI organizes the elements of your simulation onto separate windows and tabs.
2. From PLL_RAM project directory, open the testbench_1.v file. Similarly, open the mentor/mentor_example.do file.
3. To display the **Transcript** window, click **View > Transcript**. You can enter commands for ModelSim - Intel FPGA Edition directly in the **Transcript** window.
4. Type the following command in the **Transcript** window and then press Enter:
do mentor_example.do

The design compiles and simulates, according to your specifications in the mentor_example.do script. The following figure shows the ModelSim - Intel FPGA Edition simulator:

Figure 6. ModelSim - Intel FPGA Edition GUI

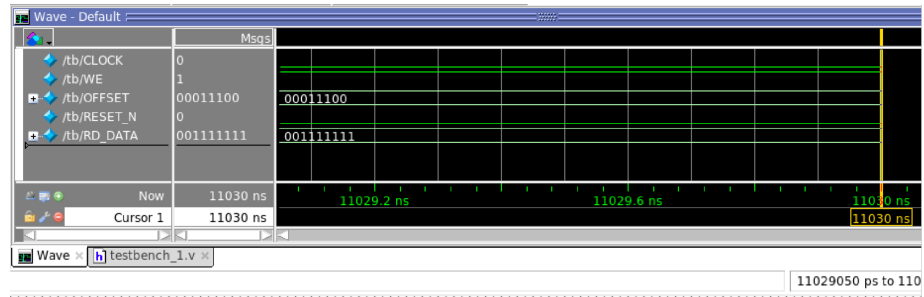


1.6. View Signal Waveforms

Follow these steps to view signals in the testbench_1.v simulation waveform:

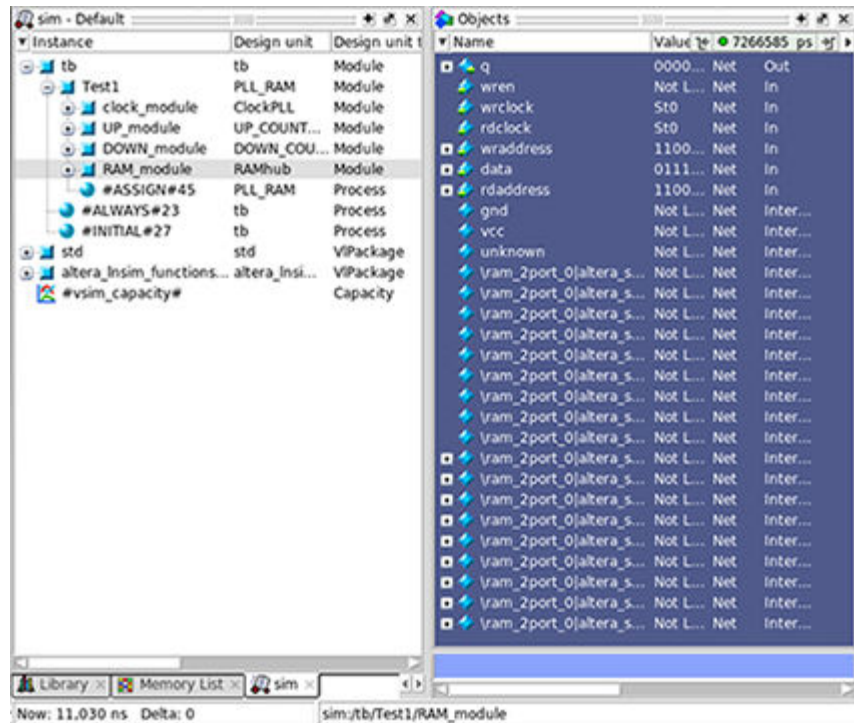
1. Click the **Wave** window. The simulation waveform ends at 11030 ns, as the testbench specifies. The **Wave** window lists the CLOCK, WE, OFFSET, RESET_N, and RD_DATA signals.

Figure 7. ModelSim - Intel FPGA Edition Wave Window



- To view the signals in the top-level `pll_ram.v` design, click the **Sim** tab. The **Sim** window synchronizes with the **Objects** window.

Figure 8. ModelSim - Intel FPGA Edition Sim and Objects Windows



- To view the top-level module signals, expand the **tb** folder in the **Objects** tab. Similarly, expand the **Test1** folder. The **Objects** window displays the `UP_module`, `DOWN_module`, `PLL_module`, and `RAM_module` signals.
- In the **Sim** window, click a module under **Test1** to display the module's signals in the **Objects** window.
- View the simulation library files in the **Library** window.



Figure 9. ModelSim - Intel FPGA Edition Library Window

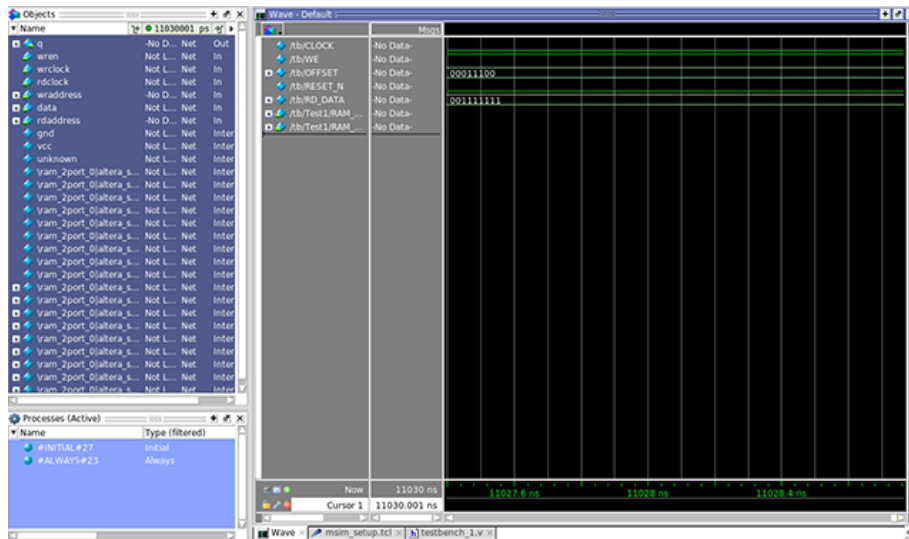


1.7. Add Signals to the Simulation

The CLOCK, WE, OFFSET, RESET_N, and RD_DATA signals automatically appear in the **Wave** window because the top-level design defines these I/O. In addition, you can optionally add internal signals to the simulation.

1. In the **Objects** window, locate the UP_module, DOWN_module, PLL_module, and RAM_module modules.
2. In the **Objects** window, select **RAM_module**. The module's inputs and outputs display.

Figure 10. Add Signals To Wave Window



- To add the internal signals between the down-counter and dual-port RAM module, right-click **rdaddress** and then click **Add Wave**.
- To add the internal signals between the up-counter and dual-port RAM module, right-click **wraddress** and then click **Add Wave**. Alternatively, you can drag and drop these signals from the **Objects** window to the **Wave** window.
- To generate the waveforms for the new signals you add, click **Simulate > Run > Continue**.

1.8. Rerun Simulation

You must rerun the simulation if you make changes to the simulation setup, such as adding signals to the **Wave** window, or modifying the `testbench_1.v` file. Follow these steps to rerun simulation:

- In the ModelSim - Intel FPGA Edition simulator, click **Simulate > Restart**. Retain the default options and click **OK**. These options clear the waveforms and restart the simulation time, while retaining the necessary signals and settings.

Note: Alternatively, you can re-run the `/PLL_RAM/mentor/mentor_example.do` script to re-run simulation at the command line.

- Click **Simulate > Run > Run -all**. The `testbench_1.v` file simulates according to the testbench specifications. To continue simulation, click **Simulate > Run > Continue**. This command continues the simulation until you click the **Stop** button.

1.9. Modify the Simulation Testbench

The `testbench_1.v` example testbench tests only a specific set of conditions and test cases. You can manually edit the `testbench_1.v` file in the ModelSim - Intel FPGA Edition simulator to test other cases and conditions:



1. Open the `testbench_1.v` file in the ModelSim - Intel FPGA Edition simulator.
2. Right-click in the `testbench_1.v` file to confirm that the file is not set to **Read Only**.
3. Enter and save any additional testbench parameters in the `testbench_1.v` file.
4. To generate the waveforms for a testbench that you modify, click **Simulate > Restart**.
5. Click **Simulate > Run > Run -all**.



2. ModelSim - Intel FPGA Edition Simulation Quick-Start Revision History

Document Version	Intel Quartus Prime Version	Changes
2019.12.30	19.4	<ul style="list-style-type: none"> Updated steps and screenshots for Intel Quartus Prime Pro Edition version 19.4. Updated design example file link and content.
2018.09.25	18.0	Corrected syntax errors in mentor_example.do Script.
2018.05.07	18.0	Removed unnecessary step from <i>Run Simulation at Command Line</i> procedure.
2017.07.15	17.1	Initial release.

Intel Corporation. All rights reserved. Agilix, Altera, Arria, Cyclone, Enpirion, Intel, the Intel logo, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

*Other names and brands may be claimed as the property of others.

ISO
9001:2015
Registered