Simulation Quick-Start for ModelSim* - Intel® FPGA Edition

Intel® Quartus® Prime Pro Edition

UG-20093
2017.07.15

Last updated for Intel® Quartus® Prime Design Suite: 17.0

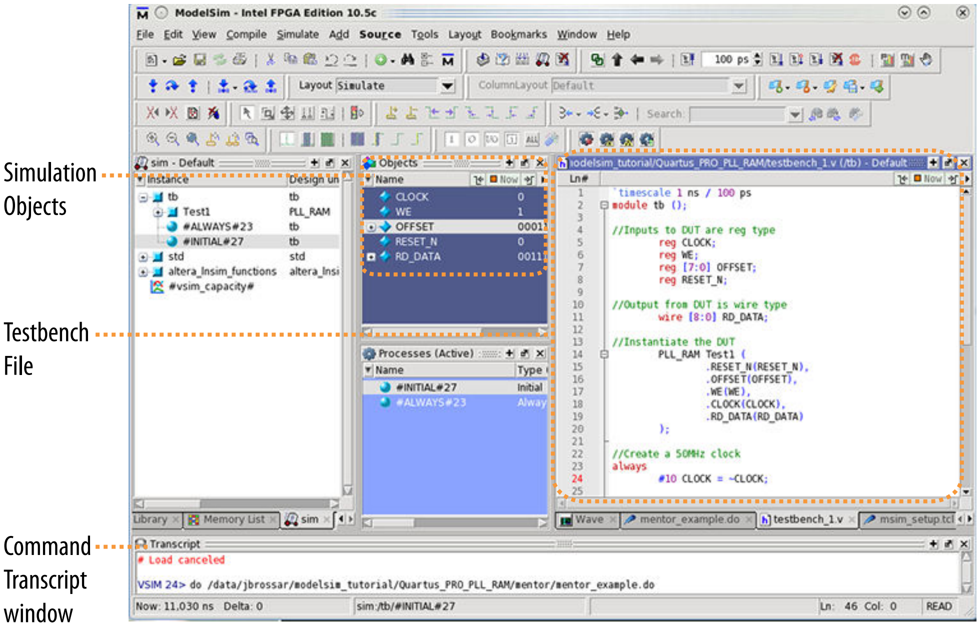
Subscribe
Send Feedback

Contents

1 Simulation Quick-Start for ModelSim* - Intel® FPGA Edition (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



Instance	Design un
tb	tb
Test1	PLL_RAM
#ALWAYS#23	tb
#INITIAL#27	tb
std	std
altera_lnsim_functions	altera_lnsi
#vsim_capacity#	

Name	Now
CLOCK	0
WE	1
OFFSET	0001
RESET_N	0
RD_DATA	0011

Name	Type
#INITIAL#27	Initial
#ALWAYS#23	Alway

odelsim_tutorial/Quartus_PRO_PLL_RAM/testbench_1.v (/tb) - Default

```
`timescale 1 ns / 100 ps
module tb ();

//Inputs to DUT are reg type
        reg CLOCK;
        reg WE;
        reg [7:0] OFFSET;
        reg RESET_N;

//Output from DUT is wire type
        wire [8:0] RD_DATA;

//Instantiate the DUT
        PLL_RAM Test1 (
                .RESET_N(RESET_N),
                .OFFSET(OFFSET),
                .WE(WE),
                .CLOCK(CLOCK),
                .RD_DATA(RD_DATA)
        );

//Create a 50MHz clock
always
        #10 CLOCK = ~CLOCK;
```

```
# Load canceled

VSIM 24> do /data/jbrossar/modelsim_tutorial/Quartus_PRO_PLL_RAM/mentor/mentor_example.do
```

Now: 11,030 ns Delta: 0 sim:/tb/#INITIAL#27 Ln: 46 Col: 0 READ

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 5
5. Compile and Simulate the Design on page 7
6. View Signal Waveforms on page 8
7. Add Signals to the Simulation on page 10
8. Rerun Simulation on page 11
9. Modify the Simulation Testbench on page 11

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.

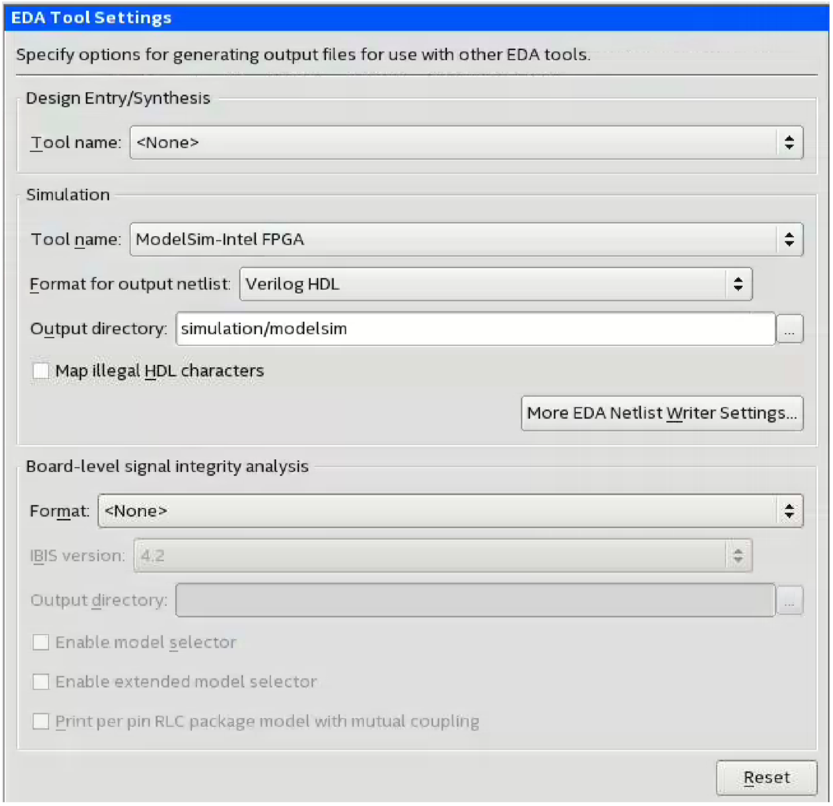
Note: This Quick-Start requires a basic understanding of hardware description language syntax and the Intel Quartus Prime design flow, as the Intel Quartus Prime Pro Edition Foundation Online Training describes.

1. Download and unzip the `Quartus_Pro_PLL_RAM.zip` design example from the Altera wiki.
2. Launch the Intel Quartus Prime Pro Edition software.
3. To open the example design project, click **File ➤ Open Project**, select the **pll_ram.qpf** project file, and then click **OK**.

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-Altera** as the **Tool name**. Retain the default settings for **Format for output netlist** and **Output directory**.



EDA Tool Settings

Specify options for generating output files for use with other EDA tools.

Design Entry/Synthesis

Tool name: <None>

Simulation

Tool name: ModelSim-Intel FPGA

Format for output netlist: Verilog HDL

Output directory: simulation/modelsim

□ Map illegal HDL characters

Board-level signal integrity analysis

Format: <None>

IBIS version: 4.2

Output directory:

□ Enable model selector

□ Enable extended model selector

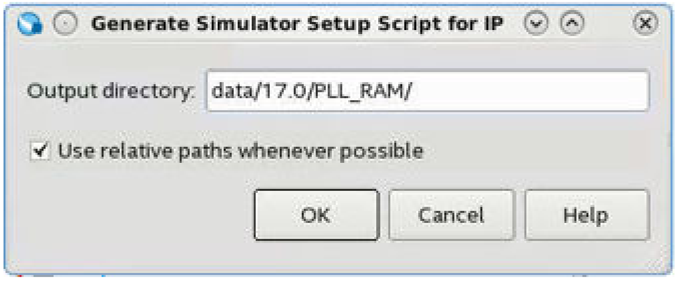
□ Print per pin RLC package model with mutual coupling

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 2. Generate Simulator Setup Scripts IP Dialog Box



Output directory: data/17.0/PLL_RAM/

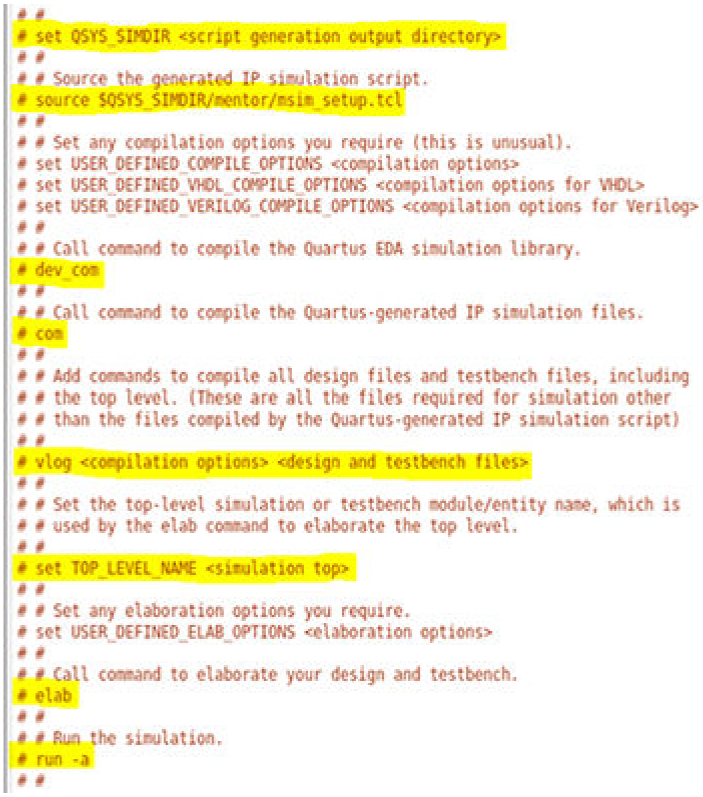
☑ Use relative paths whenever possible

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 3. 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
`set QSYS_SIMDIR <script generation output directory>`	`../`

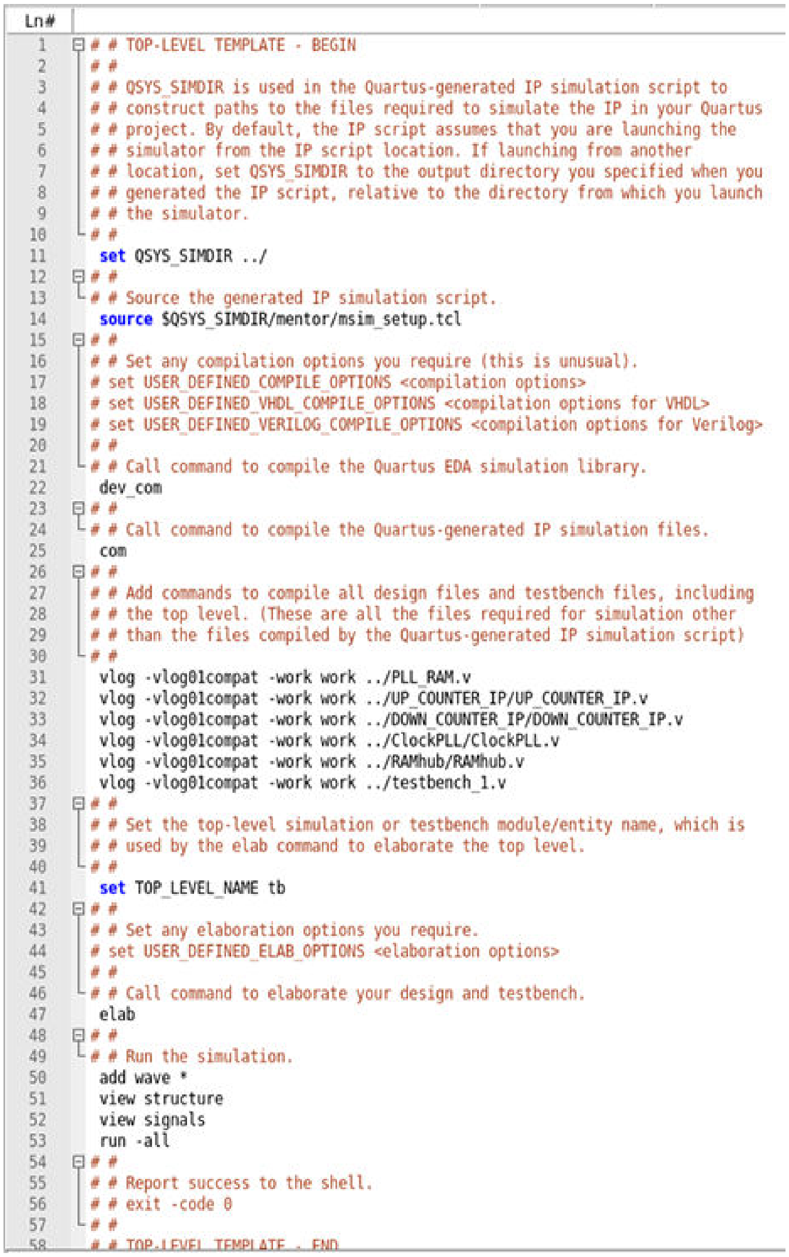
`vlog <compilation options> <design and testbench files>`	`vlog –vlog01compat –work work ../PLL_RAM.v`
	`vlog –vlog01compat –work work ../UP_COUNTER_IP/UP_COUNTER_IP.v`
	`vlog –vlog01compat –work work ../DOWN_COUNTER_IP/DOWN_COUNTER_IP.v`
	`vlog –vlog01compat –work work ../ClockPLL/ClockPLL.v`
	`vlog –vlog01compat –work work ../RAMhub/RAMhub.v`
	`vlog –vlog01compat –work work ../testbench_1.v`

`set Top_Level_Name <simulation top>`	`set Top_Level_Name tb`

`run -a`	`add wave *`
	`view structure`
	`view signals`
	`run –all`

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

Figure 4. Completed Top-Level IP Simulation Setup Script



```
# # TOP-LEVEL TEMPLATE - BEGIN
# #
# # QSYS_SIMDIR is used in the Quartus-generated IP simulation script to
# # construct paths to the files required to simulate the IP in your Quartus
# # project. By default, the IP script assumes that you are launching the
# # simulator from the IP script location. If launching from another
# # location, set QSYS_SIMDIR to the output directory you specified when you
# # generated the IP script, relative to the directory from which you launch
# # the simulator.
# #
 set QSYS_SIMDIR ../
# #
# # 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 -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
# #
# # 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 tb
# #
# # 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.
 add wave *
 view structure
 view signals
 run -all
# #
# # Report success to the shell.
# # exit -code 0
# #
# # TOP-LEVEL TEMPLATE - END
```

1.5 Compile and Simulate the Design

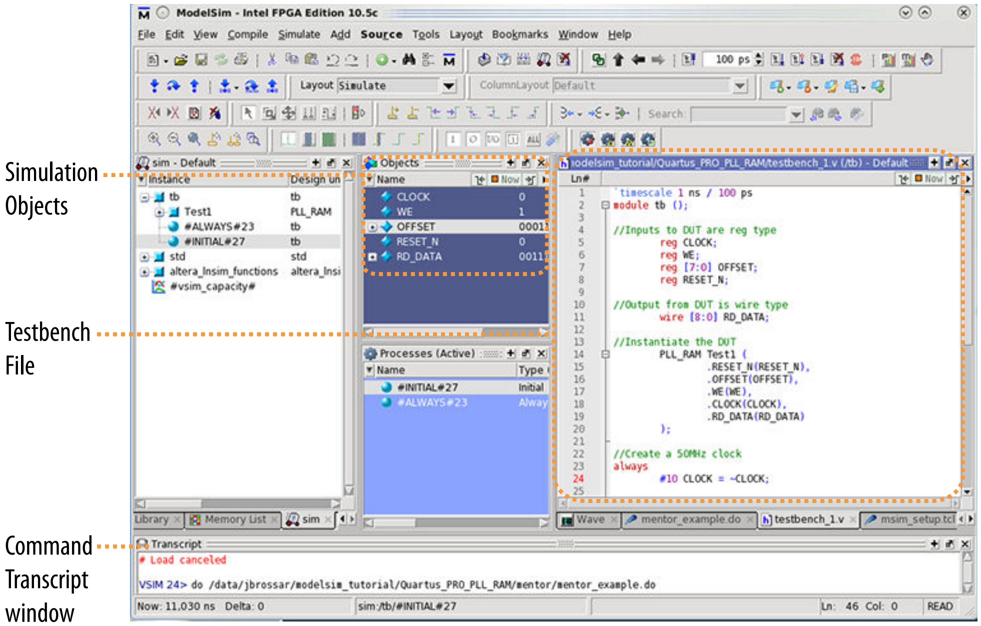
Run the top-level mentor_example.do script in the ModelSim - Intel FPGA Edition software to compile and simulate your design.

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. In PLL_RAM project directory, right-click the `mentor_example.do` file, select **Open with**, and specify the path to the ModelSim - Intel FPGA Edition executable. The file opens in ModelSim - Intel FPGA Edition. Repeat this step to open the `testbench_1.v` 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 5. ModelSim - Intel FPGA Edition GUI



```
`timescale 1 ns / 100 ps
module tb ();

//Inputs to DUT are reg type
        reg CLOCK;
        reg WE;
        reg [7:0] OFFSET;
        reg RESET_N;

//Output from DUT is wire type
        wire [8:0] RD_DATA;

//Instantiate the DUT
        PLL_RAM Test1 (
                .RESET_N(RESET_N),
                .OFFSET(OFFSET),
                .WE(WE),
                .CLOCK(CLOCK),
                .RD_DATA(RD_DATA)
        );

//Create a 50MHz clock
always
        #10 CLOCK = ~CLOCK;
```

```
# Load canceled

VSIM 24> do /data/jbrossar/modelsim_tutorial/Quartus_PRO_PLL_RAM/mentor/mentor_example.do
```

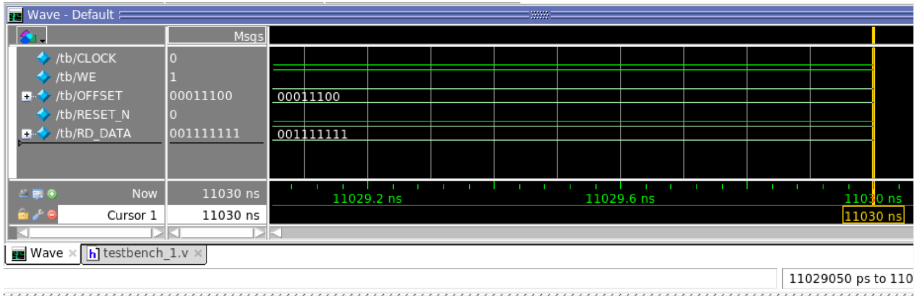
```
Now: 11,030 ns   Delta: 0   sim:/tb/#INITIAL#27   Ln: 46 Col: 0   READ
```

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 6. ModelSim - Intel FPGA Edition Wave Window

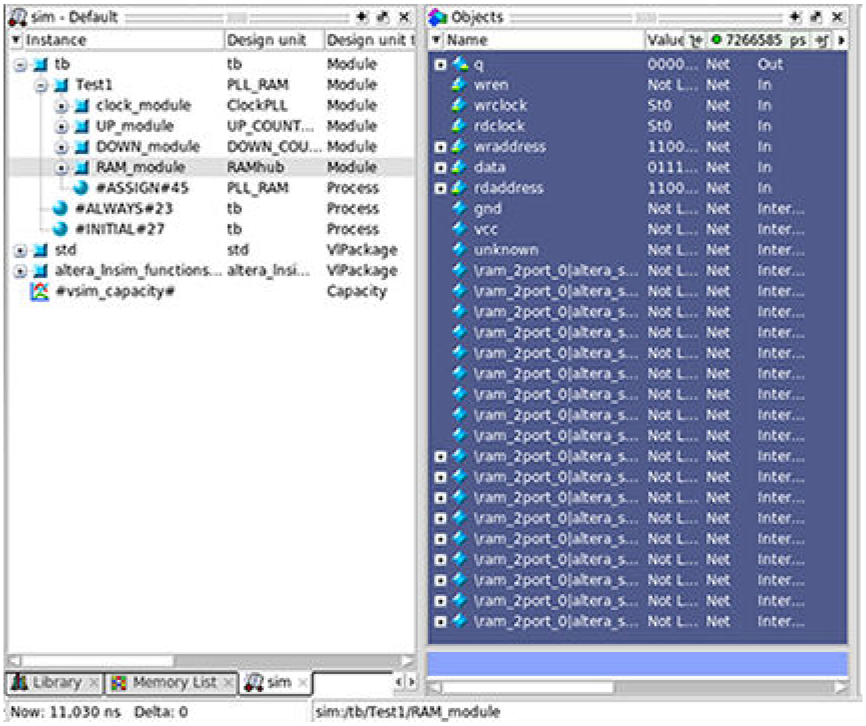


Wave - Default

	Msgs	
/tb/CLOCK	0	
/tb/WE	1	
/tb/OFFSET	00011100	00011100
/tb/RESET_N	0	
/tb/RD_DATA	001111111	001111111
Now	11030 ns	
Cursor 1	11030 ns	11030 ns

2. 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 7. ModelSim - Intel FPGA Edition Sim and Objects Windows



sim - Default

Instance	Design unit	Design unit t
tb	tb	Module
Test1	PLL_RAM	Module
clock_module	ClockPLL	Module
UP_module	UP_COUNT...	Module
DOWN_module	DOWN_COU...	Module
RAM_module	RAMhub	Module
#ASSIGN#45	PLL_RAM	Process
#ALWAYS#23	tb	Process
#INITIAL#27	tb	Process
std	std	VlPackage
altera_lnsim_functions...	altera_lnsi...	VlPackage
#vsim_capacity#		Capacity

Objects

Name	Value		
q	0000...	Net	Out
wren	Not L...	Net	In
wrclock	St0	Net	In
rdclock	St0	Net	In
wraddress	1100...	Net	In
data	0111...	Net	In
rdaddress	1100...	Net	In
gnd	Not L...	Net	Inter...
vcc	Not L...	Net	Inter...
unknown	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...
\ram_2port_0|altera_s...	Not L...	Net	Inter...

Now: 11,030 ns Delta: 0

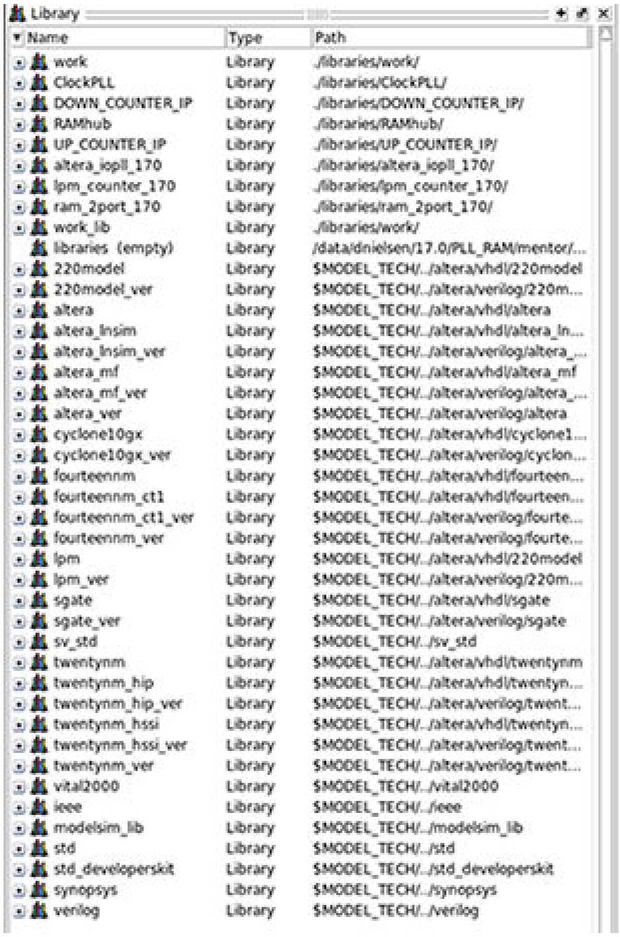
sim:/tb/Test1/RAM_module

3. 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.

4. In the **Sim** window, click a module under **Test1** to display the module's signals in the **Objects** window.

5. View the simulation library files in the **Library** window.

Figure 8. ModelSim - Intel FPGA Edition Library Window



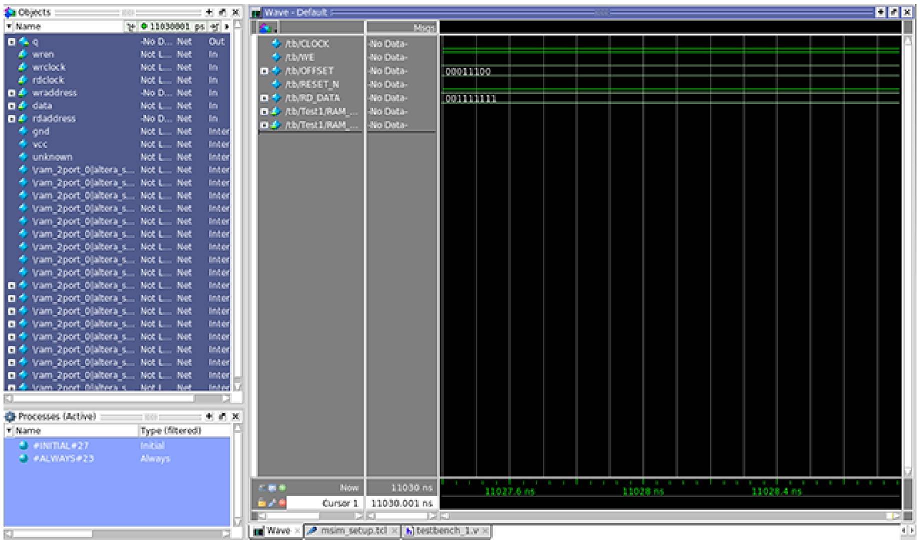
Name	Type	Path
work	Library	./libraries/work/
ClockPLL	Library	./libraries/ClockPLL/
DOWN_COUNTER_IP	Library	./libraries/DOWN_COUNTER_IP/
RAMhub	Library	./libraries/RAMhub/
UP_COUNTER_IP	Library	./libraries/UP_COUNTER_IP/
altera_iopll_170	Library	./libraries/altera_iopll_170/
lpm_counter_170	Library	./libraries/lpm_counter_170/
ram_2port_170	Library	./libraries/ram_2port_170/
work_lib	Library	./libraries/work/
libraries (empty)	Library	/data/dnielsen/17.0/PLL_RAM/mentor/...
220model	Library	$MODEL_TECH/../altera/vhdl/220model
220model_ver	Library	$MODEL_TECH/../altera/verilog/220m...
altera	Library	$MODEL_TECH/../altera/vhdl/altera
altera_lnsim	Library	$MODEL_TECH/../altera/vhdl/altera_ln...
altera_lnsim_ver	Library	$MODEL_TECH/../altera/verilog/altera_...
altera_mf	Library	$MODEL_TECH/../altera/vhdl/altera_mf
altera_mf_ver	Library	$MODEL_TECH/../altera/verilog/altera_...
altera_ver	Library	$MODEL_TECH/../altera/verilog/altera
cyclone10gx	Library	$MODEL_TECH/../altera/vhdl/cyclone1...
cyclone10gx_ver	Library	$MODEL_TECH/../altera/verilog/cyclon...
fourteennm	Library	$MODEL_TECH/../altera/vhdl/fourteen...
fourteennm_ct1	Library	$MODEL_TECH/../altera/vhdl/fourteen...
fourteennm_ct1_ver	Library	$MODEL_TECH/../altera/verilog/fourte...
fourteennm_ver	Library	$MODEL_TECH/../altera/verilog/fourte...
lpm	Library	$MODEL_TECH/../altera/vhdl/220model
lpm_ver	Library	$MODEL_TECH/../altera/verilog/220m...
sgate	Library	$MODEL_TECH/../altera/vhdl/sgate
sgate_ver	Library	$MODEL_TECH/../altera/verilog/sgate
sv_std	Library	$MODEL_TECH/../sv_std
twentynm	Library	$MODEL_TECH/../altera/vhdl/twentynm
twentynm_hip	Library	$MODEL_TECH/../altera/vhdl/twentyn...
twentynm_hip_ver	Library	$MODEL_TECH/../altera/verilog/twent...
twentynm_hssi	Library	$MODEL_TECH/../altera/vhdl/twentyn...
twentynm_hssi_ver	Library	$MODEL_TECH/../altera/verilog/twent...
twentynm_ver	Library	$MODEL_TECH/../altera/verilog/twent...
vital2000	Library	$MODEL_TECH/../vital2000
ieee	Library	$MODEL_TECH/../ieee
modelsim_lib	Library	$MODEL_TECH/../modelsim_lib
std	Library	$MODEL_TECH/../std
std_developerskit	Library	$MODEL_TECH/../std_developerskit
synopsys	Library	$MODEL_TECH/../synopsys
verilog	Library	$MODEL_TECH/../verilog

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 9. Add Signals To Wave Window



Objects

Name	● 11030001 ps		
q	-No D...	Net	Out
wren	Not L...	Net	In
wrclock	Not L...	Net	In
rdclock	Not L...	Net	In
wraddress	-No D...	Net	In
data	Not L...	Net	In
rdaddress	-No D...	Net	In
gnd	Not L...	Net	Inter
vcc	Not L...	Net	Inter
unknown	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter
\ram_2port_0|altera_s...	Not L...	Net	Inter

Processes (Active)

Name	Type (filtered)
#INITIAL#27	Initial
#ALWAYS#23	Always

Wave - Default

	Msgs	
/tb/CLOCK	-No Data-	
/tb/WE	-No Data-	
/tb/OFFSET	-No Data-	00011100
/tb/RESET_N	-No Data-	
/tb/RD_DATA	-No Data-	001111111
/tb/Test1/RAM_...	-No Data-	
/tb/Test1/RAM_...	-No Data-	
Now	11030 ns	11027.6 ns 11028 ns 11028.4 ns
Cursor 1	11030.001 ns	

3. To add the internal signals between the down-counter and dual-port RAM module, right-click **rdaddress** and then click **Add Wave**.

4. 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.

5. 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:

1. 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.

2. 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 Document Revision History

This document has the following revision history.

Table 2. Document Revision History

Date	Changes
2017.07.15	Initial release.