Updating Simulation Models for the POS-PHY Level 4 MegaCore Function

December 2004, ver. 1.1 Application Note 360

Introduction

The Altera® POS-PHY Level 4 MegaCore® function is highly parameterized, and provides a rich set of features to meet the optimization requirements of a number of applications. The MegaWizard® Plug-In (wizard) and IP Toolbench make customizing the MegaCore function during system design easy. Sometimes, however, modifications to generated files or blocks, such as a change to an LVDS block or to a phase-locked loop (PLL) instantiation, are required and cannot be made using the wizard.

This application note describes the register transfer level (RTL) source, the simulation model, and the demonstration testbench files that can be edited for such modifications, and explains how to get the simulation models to converge with the edited wrapper file. Two scripts—to help you update the simulation models after you modify the MegaCore function's top-level wrapper—are provided with this application note. One script is a bash shell for Solaris and Linux users; the other script is for Windows users.

Project Files

All POS-PHY Level 4 MegaCore function variations are generated following the same three-step flow. First, using the wizard you parameterize the MegaCore function. Next, you select whether or not to generate a simulation model in VHDL or Verilog HDL. Finally, you press the **Generate** button to instruct IP Toolbench to generate the MegaCore function that you have parameterized.

IP Toolbench generates the following files: a clear-text top-level file, where the parameter options you selected are set up; a clear-text wrapper file, that connects the various components of the MegaCore function; a clear-text high-speed serial interface (HSSI) block, that instantiates and configures the LVDS and PLL macros; and a number of encrypted Verilog HDL RTL files, that comprise the internals of the MegaCore function. If you have selected to generate a simulation model, the wizard passes the top-level file, the wrapper file, and the Verilog HDL source files to the Quartus® II software to produce a generic Verilog HDL or VHDL model of the MegaCore function as it is parameterized. The generated model is a single file that includes a model of the MegaCore function's top-level, and can be used by any standard simulator. The wizard also generates a Verilog HDL demonstration testbench customized to the MegaCore function's selected parameters.

Table 1 lists, as an example, the files generated for a receiver MegaCore function. For this example, all of the filenames are prefixed by my_rx_core.

Table 1. Receiver Files

Name	Purpose
my_rx_core.v	Clear-text top-level file that sets the parameters to be as you selected them in IP Toolbench.
my_rx_core_aot1169_wrapper_concat.v	Clear-text main wrapper file that ties together the different components of the MegaCore function.
my_rx_core_aot1169_hssi_concat.v	Clear-text high-speed serial interface (HSSI) block, includes instantiations of the high-speed LVDS blocks and PLLs.
my_rx_core_aot1169_atlfifo_concat.v my_rx_core_aot1169_core_concat.v my_rx_core_aot1169_sched_concat.v my_rx_core_aot1169_statproc_concat.v	Encrypted MegaCore function source files that the Quartus II software uses to compile designs, or to generate simulations models.
my_rx_core.vo (my_rx_core.vho)	Verilog HDL (or VHDL) simulation model of the MegaCore function, where the top level is from my_core_rx.v.
my_rx_core_bb.v my_rx_core_bc.v my_rx_core_clk_gen.v my_rx_core_clk_mon.v my_rx_core_delay.v my_rx_core_dip4util.v my_rx_core_gfifo.v my_rx_core_idx.v my_rx_core_inst.v my_rx_core_pbuff_mon.v my_rx_core_pin_mon.v my_rx_core_pkt_checker.v my_rx_core_pl4_proto_chk.v my_rx_core_reset.v my_rx_core_sapmon.v my_rx_core_sapmon_p3.v my_rx_core_tb.v my_rx_core_train_gen.v	Clear-text Verilog HDL files used by the demonstration testbench.

Throughout the design of a project, you may need to modify subtle things about the MegaCore function. For example, you may decide to change the value of parameters set by IP Toolbench without re-running the wizard, and you may also need to change the instantiation of an altlvds macro in the my_core_aot1169_hssi_concat.v file. Once you have made changes to the wrapper file, the top-level file, or the HSSI file, the changed

files are used by subsequent compile sessions in the Quartus II software, and produce compilations that take into account these changes. The Verilog HDL or VHDL simulation models contained in the `my_rx_core.vo` (or `my_rx_core.vho`) file, however, remain unchanged, unless you explicitly update them as described in the following section.

Updating Simulation Models

The simulation models are generated by a feature in the Quartus II software that processes the encrypted RTL Verilog HDL models, synthesizes the design, and writes out a partially mapped netlist file in either VHDL or Verilog HDL.

After you have made changes to the source files generated by IP Toolbench, you can use one of the scripts included with this application note to start the Quartus II software, and regenerate the simulation models. Table 2 summarizes the scripts' command lines.

Table 2. Script Command Lines

Command Line	Description
`pl4_refresh_sim` <*MegaCore name*> <*family*> <*output_file_name*>	Bash shell version of script.
`pl4_refresh_sim_win` <*MegaCore name*> <*family*> <*output_file_name*>	Microsoft Windows version of script. Run from Windows Command window.

The <*MegaCore name*> variable refers to the name used during the MegaCore function generation. This application note uses <*my_rx_core*>. The <*family*> variable refers to the device family used—a Stratix® II, Stratix GX, Stratix, Cyclone™, or Cyclone II device. The variable must be either: `stratixii`, `stratixgx`, `stratix`, `cyclone`, or `cycloneii`. This application note uses <*stratixii*> for the family. The <*output_file_name*> refers to the name of the updated functional simulation model. This application note uses <*my_rx_core2*>.

The following shows an example command line:

```
pl4_refresh_sim my_rx_core stratixii my_rx_core2
```

To function properly, the script must:

1. Be in the same directory as the IP Toolbench-generated MegaCore function.
2. Be able to find the old simulation model, if applicable.

3. Be able to find the Quartus II software in the executable path.

The script determines the presence of an existing simulation model, (e.g., `my_rx_core.vo` or `my_rx_core.vho`), then launches a Quartus II run to generate a new Verilog HDL or VHDL model, depending on the type of file found.

Because the generation of new simulation models requires that the Quartus II software synthesize the design, any changes to the wrapper files, or HSSI blocks, that prevent the MegaCore function from synthesizing into real hardware also prevent new simulation models from being generated.

Conclusion

If your design requires you to make modifications to your POS-PHY Level 4 MegaCore function, outside of the wizard, such as changes to the wrapper file or to the HSSI block, this application note and accompanying scripts provide you with the necessary information and means to update the simulation models to reflect these changes.

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