12. Upgrading to UniPHY-based Controllers from ALTMEMPHY-based Controllers

EMI_RM_012-2.1

This chapter describes upgrading the following ALTMEMPHY-based controller designs to UniPHY-based controllers:

- DDR2 or DDR3 SDRAM High-Performance Controller II with ALTMEMPHY designs
- DDR2 or DDR3 SDRAM High-Performance Controller with ALTMEMPHY designs

Altera does not support upgrading designs that do not use the AFI.

If your design uses non-AFI IP cores, Altera recommends that you start a new design with the UniPHY IP core . In addition, Altera recommends that any new designs targeting Stratix III, Stratix IV, or Stratix V use the UniPHY datapath.

Upgrading from DDR2 or DDR3 SDRAM High-Performance Controller II with ALTMEMPHY Designs

To upgrade to the DDR2 or DDR3 SDRAM controller with UniPHY IP core from DDR2 or DDR3 SDRAM High-Performance Controller II with ALTMEMPHY designs, follow these steps:

1. Generating Equivalent Design
2. Replacing the ALTMEMPHY Datapath with UniPHY Datapath
3. Resolving Port Name Differences
4. Creating OCT Signals
5. Running Pin Assignments Script
6. Removing Obsolete Files
7. Simulating your Design

The following sections describes these steps.

Generating Equivalent Design

Create a new DDR2 or DDR3 SDRAM controller with UniPHY IP core, by following the steps in volume 2, section 1, chapter 8, *Implementing and Parameterizing Memory IP* and use the following guidelines:

- Specify the same variation name as the ALTMEMPHY variation.

- Specify a directory different than the ALTMEMPHY design directory to prevent files from overwriting each other during generation.

To ease the migration process, ensure the UniPHY-based design you create is as similar as possible to the existing ALTMEMPHY-based design. In particular, you should ensure the following settings are the same in your UniPHY-based design:

- **PHY settings** tab
  - FPGA speed grade
  - PLL reference clock
  - Memory clock frequency
  - There is no need to change the default **Address and command clock phase settings**; however, if you have board skew effects in your ALTMEMPHY design, enter the difference between that clock phase and the default clock phase into the **Address and command clock phase settings**.
- **Memory Parameters** tab—all parameters must match.
- **Memory Timing** tab—all parameters must match.
- **Board settings** tab—all parameters must match.
- **Controller settings** tab—all parameters must match

In ALTMEMPHY-based designs you can turn off dynamic OCT. However, all UniPHY-based designs use dynamic parallel OCT and you cannot turn it off.

Replacing the ALTMEMPHY Datapath with UniPHY Datapath

To replace the ALTMEMPHY datapath with the UniPHY datapath, follow these steps:

1. In the Quartus II software, open the Assignment Editor, on the Assignments menu click **Assignment Editor**.
2. Manually, delete all of the assignments related to the external memory interface pins, except for the location assignments if you are preserving the pinout. By default, these pin names start with the `mem` prefix, though in your design they may have a different name.
3. Remove the old ALTMEMPHY **.qip** file from the project:
   - On the Assignments menu click **Settings**.
   - Specify the old **.qip**, and click **Remove**.

Your design now uses the UniPHY datapath.

Resolving Port Name Differences

Several port names in the ALTMEMPHY datapath are different than in the UniPHY datapath. The different names may cause compilation errors. This section describes the changes you must make in the RTL for the entity that instantiates the memory IP core. Each change applies to a specific port in the ALTMEMPHY datapath. Unconnected ports require no changes.

In some instances, multiple ports in ALTMEMPHY-based designs are mapped to a single port in UniPHY-based designs. If you use both ports in ALTMEMPHY-based designs, assign a temporary signal to the common port and connect it to the original wires. Table 12–1 shows the changes you must make.

Table 12–1. Changes to ALTMEMPHY Port Names

ALTMEMPHY Port	Changes
aux_full_rate_clk	The UniPHY-based design does not generate this signal. You can generate it if you require it.
aux_scan_clk	The UniPHY-based design does not generate this signal. You can generate it if you require it.
aux_scan_clk_reset_n	The UniPHY-based design does not generate this signal. You can generate it if you require it.
dll_reference_clk	The UniPHY-based design does not generate this signal. You can generate it if you require it.
dqs_delay_ctrl_export	This signal is for DLL sharing between ALTMEMPHY instances and is not applicable for UniPHY-based designs.
local_address	Rename to avl_addr.
local_be	Rename to avl_be.
local_burstbegin	Rename to avl_burstbegin.
local_rdata	Rename to avl_rdata.
local_rdata_valid	Rename to avl_rdata_valid.
local_read_req	Rename to avl_read_req.
local_ready	Rename to avl_ready.
local_size	Rename to avl_size.
local_wdata	Rename to avl_wdata.
local_write_req	Rename to avl_write_req.
mem_addr	Rename to mem_a.
mem_clk	Rename to mem_ck.
mem_clk_n	Rename to mem_ck_n.
mem_dqsn	Rename to mem_dqs_n.
oct_ctl_rs_value	Remove from design ("Creating OCT Signals" on page 12–4).
oct_ctl_rt_value	Remove from design ("Creating OCT Signals" on page 12–4).
phy_clk	Rename to afi_clk.
reset_phy_clk_n	Rename to afi_reset_n.
local_refresh_ack local_wdata_req reset_request_n	The controller no longer exposes these signals to the top-level design, so comment out these outputs. If you need it, bring the wire out from the high-performance II controller entity in *<project directory>*/**uniphy/rtl/***<variation name>*_**controller_phy.sv**.

Creating OCT Signals

In ALTMEMPHY-based designs, the Quartus II Fitter creates the `alt_oct` block outside the IP core and connects it to the `oct_ctl_rs_value` and `oct_ctl_rt_value` signals. In UniPHY-based designs, the OCT block is part of the IP core, so the design no longer requires these two ports. Instead, the UniPHY-based design requires two additional ports, `oct_rup` and `oct_rdn`. You must create these ports in the instantiating entity as input pins and connect to the UniPHY instance. Then route these pins to the top-level design and connect to the OCT R_{UP} and R_{DOWN} resistors on the board.

For information on OCT control block sharing, refer to "The OCT Sharing Interface" in this volume.

Running Pin Assignments Script

Remap your design by running analysis and synthesis. When analysis and synthesis completes, run the pin assignments Tcl script and then verify the new pin assignments in the Assignment Editor.

Removing Obsolete Files

After you upgrade the design, you may remove the unnecessary ALTMEMPHY design files from your project. To identify these files, examine the original ALTMEMPHY-generated **.qip** file in any text editor.

Simulating your Design

You must use the UniPHY memory model to simulate your new design. To use the UniPHY memory model, follow these steps:

1. Edit your instantiation of the UniPHY datapath to ensure the `local_init_done`, `local_cal_success`, `local_cal_fail`, `soft_reset_n`, `oct_rdn`, `oct_rup`, `reset_phy_clk_n`, and `phy_clk` signals are at the top-level entity so that an instantiating testbench can refer to those signals.
2. To use the UniPHY testbench and memory model, generate the example design when generating your IP instantiation.

3. Specify that your third-party simulator should use the UniPHY testbench and memory model instead of the ALTMEMPHY memory model:
   a. On the Assignments menu, point to **Settings** and click the **Project Settings** window.
   b. Select the **Simulation** tab, click **Test Benches**, click **Edit**, and replace the ALTMEMPHY testbench files with the following files:

- **\<*project directory*>\<*variation name*>_example_design\simulation\verilog\submodules\altera_avalon_clock_source.sv**
  or
  \<*project directory*>\<*variation name*>_example_design\simulation\vhdl\submodules\altera_avalon_clock_source.vhd
- **\<*project directory*>\<*variation name*>_example_design\simulation\verilog\submodules\altera_avalon_reset_source.sv**
  or
  \<*project directory*>\<*variation name*>_example_design\simulation\vhdl\submodules\altera_avalon_reset_source.vhd
- **\<*project directory*>\<*variation name*>_example_design\simulation\verilog\<*variation name*>_example_sim.v**
  or
  \uniphy\<*variation name*>_example_design\simulation\vhdl\<*variation name*>_example_sim.vhd
- **\<*project directory*>\<*variation name*>_example_design\simulation\verilog\submodules\verbosity_pkg.sv**
- **\<*project directory*>\<*variation name*>_example_design\simulation\verilog\submodules\status_checker_no_ifdef_params.sv**
  or
  \<*project directory*>\<*variation name*>_example_design\simulation\vhdl\submodules\status_checker_no_ifdef_params.sv
- **\<*project directory*>\<*variation name*>_example_design\simulation\verilog\submodules\alt_mem_if_common_ddr_mem_model_ddr3_mem_if_dm_pins_en_mem_if_dqsn_en.sv**
  or
  \<*project directory*>\<*variation name*>_example_design\simulation\vhdl\submodules\alt_mem_if_common_ddr_mem_model_ddr3_mem_if_dm_pins_en_mem_if_dqsn_en.sv
- **\<*project directory*>\<*variation name*>_example_design\simulation\verilog\submodules\alt_mem_if_ddr3_mem_model_top_ddr3_mem_if_dm_pins_en_mem_if_dqsn_en**
  or
  \<*project directory*>\<*variation name*>_example_design\simulation\vhdl\submodules\alt_mem_if_ddr3_mem_model_top_ddr3_mem_if_dm_pins_en_mem_if_dqsn_en

4. Open the <*variation name*>**_example_sim.v** file and find the UniPHY-generated simulation example design module name below: <*variation name*>**_example_sim_e0**.
5. Change the module name above to the name of your top-level design module.

6. Update the following port names of the example design in the UniPHY-generated <*variation name*>**_example_sim.v** file. (Table 12–2).

Table 12–2. Example Design Port Names

Example Design Name	New Name
pll_ref_clk	Rename to clock_source.
mem_a	Rename to mem_addr.
mem_ck	Rename to mem_clk.
mem_ck_n	Rename to mem_clk_n.
mem_dqs_n	Rename to mem_dqsn.
drv_status_pass	Rename to pnf.
afi_clk	Rename to phy_clk.
afi_reset_n	Rename to reset_phy_clk_n.
drv_status_fail	This signal is not available, so comment out this output.
afi_half_clk	This signal is not exposed to the top-level design, so comment out this output.

For more information about generating example simulation files, refer to Simulating Memory IP, in volume 2 of the *External Memory Interface Handbook*.

Document Revision History

Table 12–3 lists the revision history for this document.

Table 12–3. Document Revision History

Date	Version	Changes
November 2011	2.1	Revised Simulating your Design section.