ALTOCT Megafunction User Guide

2014.07.18

UG-01003 Subscribe Send Feedback

Overview

The Dynamic Calibrated On-Chip Termination (ALTOCT) megafunction IP core is used in double-data rate (DDR) external memory interfaces. This IP core is closely associated with the External DDR Memory PHY Interface (ALTMEMPHY) megafunction. On-chip termination (OCT) improves signal quality over external termination through reduced parasitic, board space, and external component costs.

Device Support

The ALTOCT IP core supports the following Altera® devices:

- Arria® II GX
- Arria II GZ
- Arria V
- Cyclone® V
- Stratix® IV
- Stratix V

ALTOCT IP Core Parameter

Table 1: ALTOCT IP Core Parameter

Parameter	Value	Description
Calibrate OCT on power-up	Turn Off, Turn On	Turn on this option if you want to calibrate OCT on power-up.
How many OCT blocks should be used?	1 to 11	Specify the number of OCT blocks for your design.
Enable parallel termination	Turn Off, Turn On	Turn on this option if you want to enable parallel termination instead of series termination or in addition to series termination for bidirectional pins. Observe the changes in resource usage when this option is enabled.

Parameter	Value	Description
Create 'calibration_wait' input port to prevent calibration	Turn Off, Turn On	The `calibration_wait` input port can be used to halt calibration operation. This option is for advanced users only. Typical users should not enable this option.
Create 'clken' input port	Turn Off, Turn On	The clken input port is used as the clock enable signal. This option is for advanced users only. Typical users should not enable this option.
Enable independent calibration/ shift	Turn Off, Turn On	Turn on this option to enable independent calibration/shift.

ALTOCT IP Core Signals

These tables list the input and output ports for the ALTOCT IP core.

Table 2: ALTOCT IP Core Input Signals

Name	Required	Description	Comments
`calibration_ only_req`	No	User request for calibration only.	input port `[OCT_BLOCK_NUMBER - 1..0]` wide.
`shift_only_req`	No	User request for shift only.	input port `[OCT_BLOCK_NUMBER - 1..0]` wide.
`aclr`	No	Asynchronous clear.	If omitted, value is **GND**. This signal is not applicable for Stratix V devices.
`calibration_ request`	Yes	User request for calibration and shifting operations.	Input port `[OCT_BLOCK_NUMBER - 1..0]` wide.
`calibration_ wait`	No	Clock cycles to wait before starting calibration after calibration request.	Input port `[OCT_BLOCK_NUMBER - 1..0]` wide. If omitted, value is **GND**.
`clock`	Yes	System clock.	Clock should be 20 MHz or less.
`rzqin`	Yes	Reference resistor.	Input port `[OCT_BLOCK_NUMBER - 1..0]` wide.
`s2pload`	Yes	Signal used to enable serial to parallel shifting of calibrated codes to I/O buffers. `s2pload` must be asserted for at least 25 ns for correct transfer.	Input port `[OCT_BLOCK_NUMBER - 1..0]` wide.
`clken`	No	Clock enable.	If omitted, value is **VCC**.

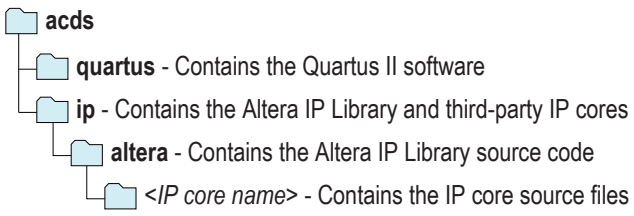
Table 3: ALTOCT IP Core Output Signals

Name	Required	Description	Comments
shift_busy	Yes	Specifies the status of the shifting operation. This signal is asserted until the shifting operation is completed.	Output port [OCT_BLOCK_NUMBER - 1...0] wide
cal_shift_busy	Yes	Specifies the status of calibration or shifting operation. This signal is asserted until the calibration or shifting operation is completed.	Output port [OCT_BLOCK_NUMBER - 1..0] wide
calibration_ busy	Yes	Specifies the status of the calibration operation. This signal is asserted until the calibration operation is completed.	Output port [OCT_BLOCK_NUMBER - 1..0] wide
paralleltermi-nationcontrol	Yes	Specifies parallel termination	Receives the current state of the pull-up and pull-down transmitter control buses from a termination logic block. Output port [OCT_BLOCK_NUMBER * 16..0] wide
seriestermina-tioncontrol	Yes	Specifies the serial termination	Receives the current state of the pull-up and pull-down receiver control buses from a termination logic block. Output port [OCT_BLOCK_NUMBER * 16..0] wide

Installing and Licensing IP Cores

The Quartus II software includes the Altera IP Library. The library provides many useful IP core functions for production use without additional license. You can fully evaluate any licensed Altera IP core in simulation and in hardware until you are satisfied with its functionality and performance. Some Altera IP cores, such as MegaCore® functions, require that you purchase a separate license for production use. After you purchase a license, visit the Self Service Licensing Center to obtain a license number for any Altera product.

Figure 1: IP Core Installation Path



Note: The default IP installation directory on Windows is **<*drive*>:\altera**<*version number*>; on Linux it is <*home directory*>**/altera/** <*version number*>.

Related Information

- **Altera Licensing Site**
- **Altera Software Installation and Licensing Manual**

Customizing and Generating IP Cores

You can customize IP cores to support a wide variety of applications. The Quartus II IP Catalog displays IP cores available for the current target device. The parameter editor guides you to set parameter values for optional ports, features, and output files.

IP Catalog and Parameter Editor

The Quartus II IP Catalog (**Tools** > **IP Catalog**) and parameter editor help you easily customize and integrate IP cores into your project. You can use the IP Catalog and parameter editor to select, customize, and generate files representing your custom IP variation.

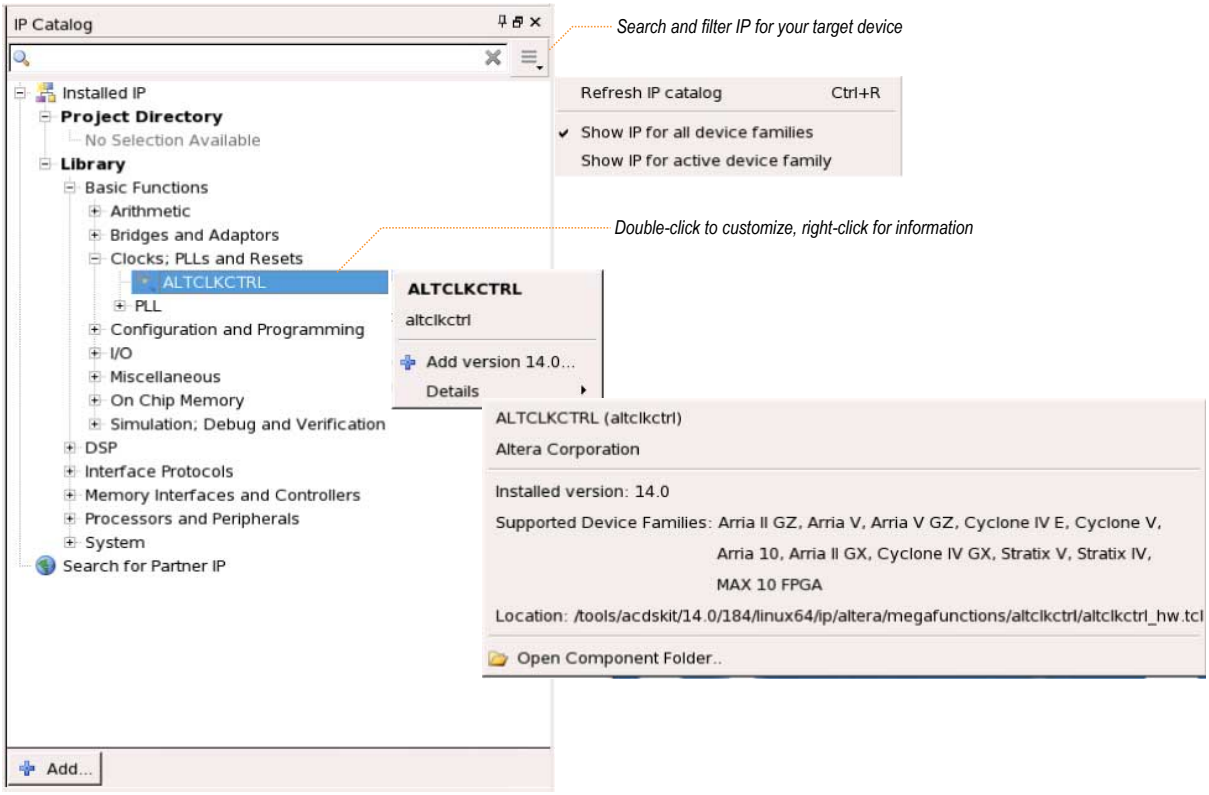
The IP Catalog automatically displays the IP cores available for your target device. Double-click any IP core name to launch the parameter editor and generate files representing your IP variation. The parameter editor prompts you to specify your IP variation name, optional ports, architecture features, and output file generation options. The parameter editor generates a top-level **.qsys** or **.qip** file representing the IP core in your project. Alternatively, you can define an IP variation without an open Quartus II project. When no project is open, select the **Device Family** directly in IP Catalog to filter IP cores by device.

Note: The IP Catalog is also available in Qsys (**View** > **IP Catalog**). The Qsys IP Catalog includes exclusive system interconnect, video and image processing, and other system-level IP that are not available in the Quartus II IP Catalog.

Use the following features to help you quickly locate and select an IP core:

- Filter IP Catalog to **Show IP for active device family** or **Show IP for all device families**.
- Search to locate any full or partial IP core name in IP Catalog. Click **Search for Partner IP**, to access partner IP information on the Altera website.
- Right-click an IP core name in IP Catalog to display details about supported devices, installation location, and links to documentation.

Figure 2: Quartus II IP Catalog



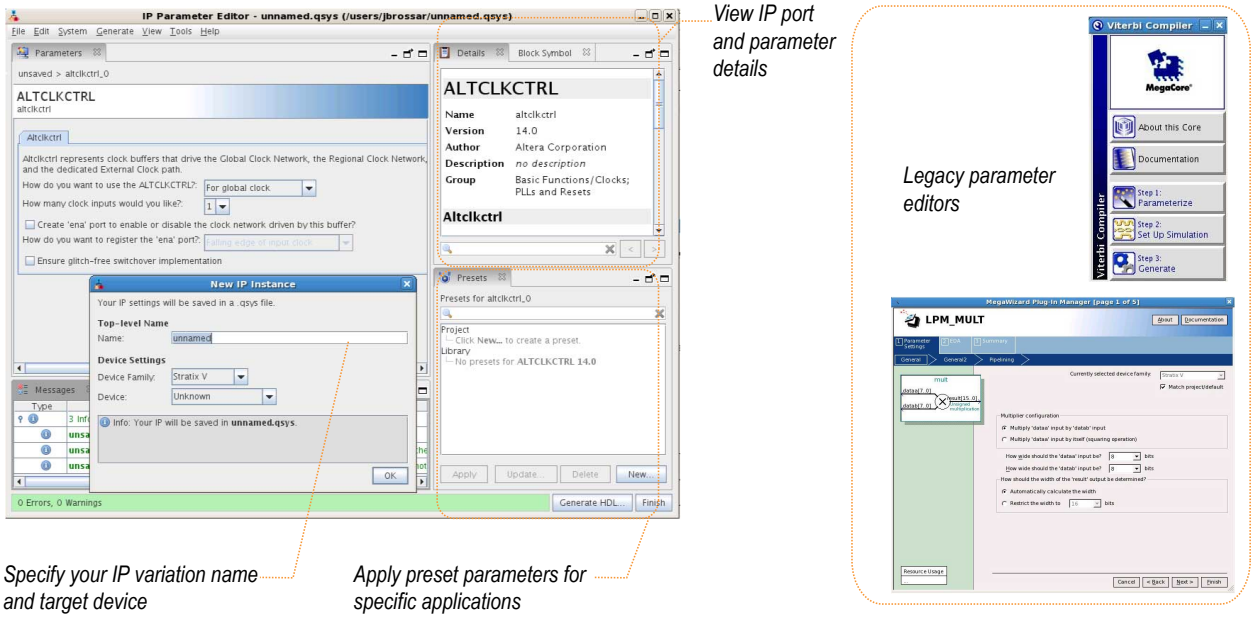
Note: The IP Catalog and parameter editor replace the MegaWizard™ Plug-In Manager in the Quartus II software. The Quartus II software may generate messages that refer to the MegaWizard Plug-In Manager. Substitute "IP Catalog and parameter editor" for "MegaWizard Plug-In Manager" in these messages.

Using the Parameter Editor

The parameter editor helps you to configure your IP variation ports, parameters, architecture features, and output file generation options.

- Use preset settings in the parameter editor (where provided) to instantly apply preset parameter values for specific applications.
- View port and parameter descriptions, and links to documentation.
- Generate testbench systems or example designs (where provided).

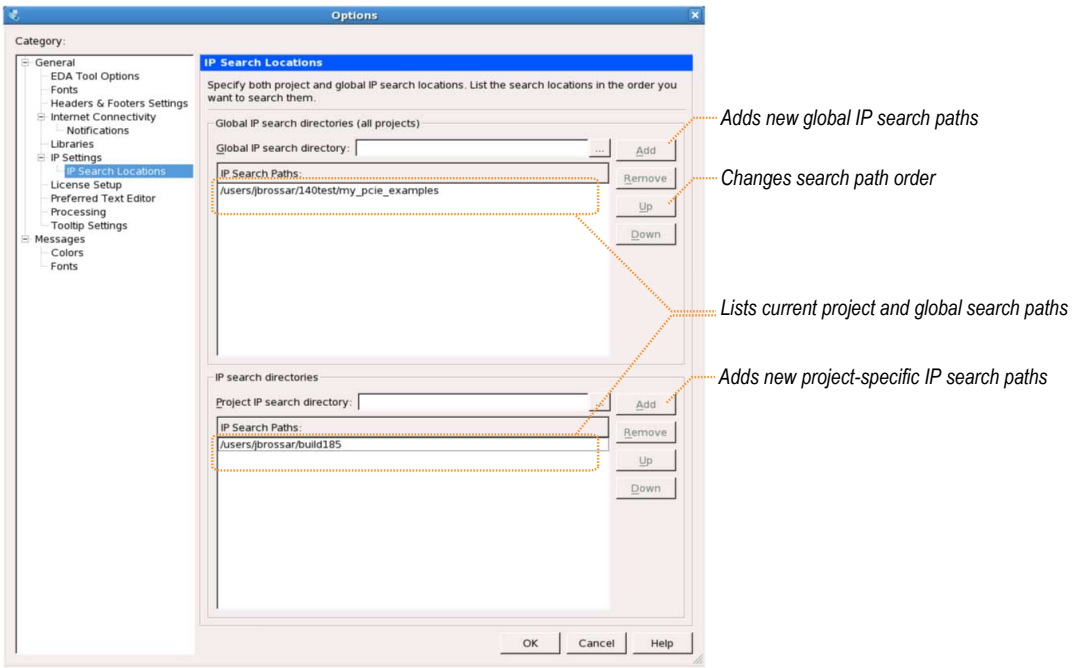
Figure 3: IP Parameter Editors



Adding IP Cores to IP Catalog

The IP Catalog automatically displays Altera IP cores found in the project directory, in the Altera installation directory, and in the defined IP search path. You can include IP libraries and files from other locations by adding search path locations. To add global or project-specific search path locations, click **Tools** > **Options** > **IP Search Locations** and specify the path to your IP cores.

Figure 4: Specifying IP Search Locations



If your project includes two IP core files with the same name, the following search path precedence rules determine the resolution of files:

1. Project directory.
2. Project database directory.
3. Project IP search path specified in **IP Search Locations**, or with the SEARCH_PATH assignment in the revision **.qsf**.
4. Global IP search path specified in **IP Search Locations**, or with the SEARCH_PATH assignment in the **quartus2.ini** file.
5. Quartus II software libraries directory, such as ***<Quartus II Installation>*\libraries**.

Specifying IP Core Parameters and Options

Follow these steps to specify IP core parameters and options.

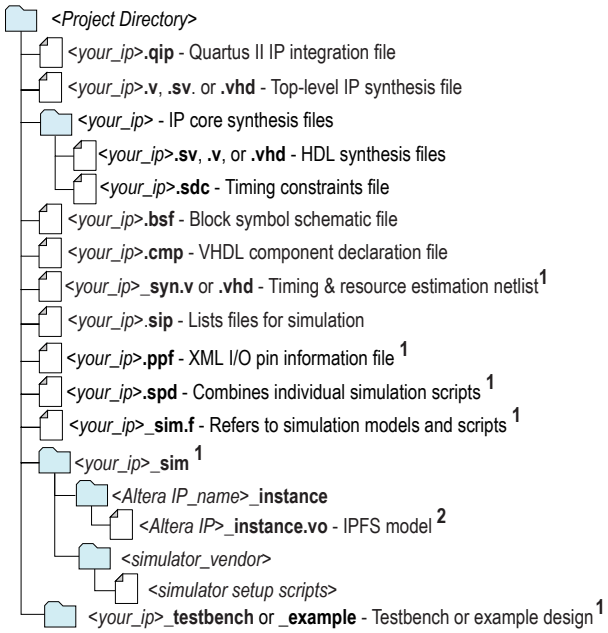
1. In the IP Catalog (**Tools** > **IP Catalog**), locate and double-click the name of the IP core to customize. The parameter editor appears.
2. Specify a top-level name for your custom IP variation. This name identifies the IP core variation files in your project. If prompted, also specify the target Altera device family and output file HDL preference. Click **OK**.
3. Specify parameters and options for your IP variation:
   - Optionally select preset parameter values. Presets specify all initial parameter values for specific applications (where provided).
   - Specify parameters defining the IP core functionality, port configurations, and device-specific features.
   - Specify options for generation of a timing netlist, simulation model, testbench, or example design (where applicable).
   - Specify options for processing the IP core files in other EDA tools.
4. Click **Finish** or **Generate** to generate synthesis and other optional files matching your IP variation specifications. The parameter editor generates the top-level **.qip** or **.qsys** IP variation file and HDL files for synthesis and simulation. Some IP cores also simultaneously generate a testbench or example design for hardware testing.

The top-level IP variation is added to the current Quartus II project. Click **Project** > **Add/Remove Files in Project** to manually add a **.qip** or **.qsys** file to a project. Make appropriate pin assignments to connect ports.

Files Generated for Altera IP Cores

The Quartus II software generates the following output for your IP core.

Figure 5: IP Core Generated Files



Notes:
1. If supported and enabled for your IP variation
2. If functional simulation models are generated

Features

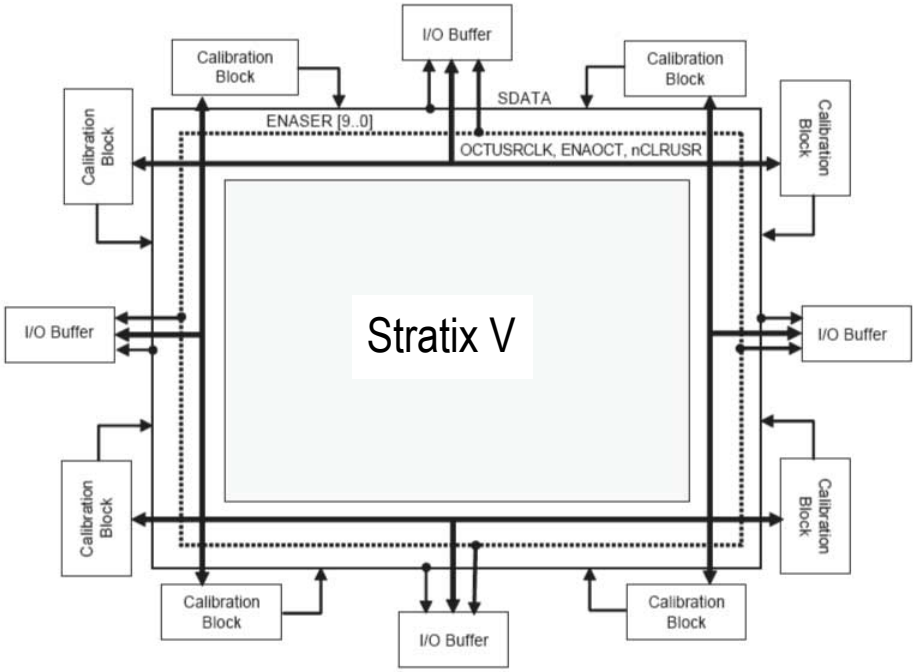
The ALTOCT IP core provides the following features:

- Support for up to 10 OCT blocks
- Support for calibrated on-chip series termination (R_S) and calibrated on-chip parallel termination (R_T) on all I/O pins
- Calibrated termination values of 25 and 50 ohm
- Support OCT calibration in user mode

Architecture

The OCT calibration architecture provides dynamic series and parallel on-chip termination to improve I/O impedance matching and termination capabilities. OCT improves signal quality over external termination through reduction of parasitic, board space, and external component costs. The OCT architecture support R_S with and without calibration, R_T with calibration, dynamic series and parallel termination for single-ended I/O standards, and on-chip differential termination (R_D) for differential LVDS I/O standards. OCT is supported in all I/O banks by selecting one of the OCT I/O standards.

Figure 6: Signals for Shift-Out Codes from the OCT Calibration Block to I/O Buffers



This figure shows the OCT calibration architecture in Stratix V devices. The SDATA, OCTUSRCLK, and ENASER signals are used to serially transfer calibrated codes from each OCT calibration block to any I/O. To serially shift the 16-bit OCT R_S calibration code and the 16-bit OCT R_T calibration code into the registers located in the I/O buffer, 32 clock cycles using OCTUSRCLK are required. When calibration is complete, the 32-bit OCT calibration code (16-bit OCT R_S code and 16-bit OCT R_T) must be serially shifted out from each OCT calibration block to the corresponding I/O buffer.

After calibrated codes are shifted in serially to each I/O bank, the calibrated codes must be converted from serial format to parallel format before the codes are used in the I/O buffers. Use the S2PENA signals to complete the serial-to-parallel shifting.

Power-Up Mode Calibration and User Mode Calibration in Quartus II Software

Stratix V devices have two termination related Quartus Settings File (**.qsf**) assignments:

- INPUT_TERMINATION
- OUTPUT_TERMINATION

Termination can exist on input and output buffers, and sometimes simultaneously.

When calibrated termination uses only the **.qsf** assignments , the power-up mode of the calibration scheme is used, and user mode calibration updates are unavailable.

To use user mode calibration, the ALTOCT IP core must be instantiated into the design. If more than one group of pins needs to be calibrated on-demand, more than one calibration block must be instantiated.

There are two methods to associate pin groups with a calibration block:

- Use a QSF assignment to indicate which pin (bus) is associated with which calibration block.
- Instantiate the I/O buffer primitives at the top level and connect them to the appropriate calibration blocks.

Note: All I/O banks with the same VCCIO can share a calibration block, even if that particular I/O bank has its own calibration block. You can connect any number of I/O pins that support calibrated termination to a calibration block, unless the specific I/O pins on the device are not usable with a particular calibration block. The Quartus II software produces warning messages if there is no pin connected to the block.

For more information about the OCT architecture and the calibration modes for other device families, refer to the respective device handbooks.

Design Example: Series Calibration for Four Calibration Blocks Using Stratix V Device

This design example uses the ALTOCT IP core to calibrate four calibration blocks using series termination for the Stratix V device.

In this example, you perform the following tasks:

- Set the Quartus II software
- Simulate the design in the ModelSim®-Altera software

Setting the Quartus II Software

1. In the Quartus II software, on the Tools menu, click **Options**.
2. In the General category, click **EDA Tool Options**.
3. In the ModelSim-Altera field, browse to **modelsim_ae\win32aloem** in your working directory.
4. Click **OK**.
5. On the Assignments menu, click **Settings**.
6. Click **EDA Tool Settings**.
7. In the Simulation field, select **ModelSim-Altera**.
8. Under the EDA Tool Settings category, click **Simulation**.
9. In the Tool name field, select **ModelSim-Altera**.
10. In the Format for output netlist, select **Verilog HDL**.
11. In the Compile test bench field, select **oct_test_vlg_vec_tst**
12. Click **Test Benches**. The Test Benches window appears.
13. Click **Edit** to update the **oct_testbench.vt** location.
14. In the Test bench name field, ensure that the entry is **oct_test_vlg_vec_tst**
15. In the Top level module in test bench field, ensure that the entry is **oct_test_vlg_vec_tst**
16. Ensure **oct_testbench.vt** location is accurate

Generating and Simulating the Design Example

Perform the following steps to generate and simulate the dynamic on-chip termination calibration blocks:

1. Open **alt_oct_SV.zip** and extract **altoct_ex_SV.qar**.
2. In the Quartus II software, open **altoct_ex_SV.qar** and restore the archive file into your working directory
3. Start **Analysis and Synthesis**
4. On the Tools menu, click **Run Simulation Tool**.
5. Click **RTL Simulation**. The ModelSim-Altera software launches.
6. Click on the zooming lenses to view the simulation results.

Verify Termination or Calibration Assignments of the Design Example

Figure 7: Assignments Used in the altoct_ex Design

This figure shows the necessary assignments in the design example.

	Status	From	To	Assignment Name	Value	Enabled
1	Ok		test_output[0]	I/O Standard	1.2 V	Yes
2	Ok		test_output[1]	I/O Standard	1.2 V	Yes
3	Ok		test_output[2]	I/O Standard	1.2 V	Yes
4	Ok		test_output[3]	I/O Standard	1.2 V	Yes
5	Ok		test_output[0]	Output Termination	Series 50 Ohm with Calibration	Yes
6	Ok		test_output[1]	Output Termination	Series 50 Ohm with Calibration	Yes
7	Ok		test_output[2]	Output Termination	Series 50 Ohm with Calibration	Yes
8	Ok		test_output[3]	Output Termination	Series 50 Ohm with Calibration	Yes
9	Ok		test_output[0]	Termination Control Block	cal_out:inst|cal_out_alt_oct_57p:cal_out_alt_oct_57p_component|sd1a_0	Yes
10	Ok		test_output[1]	Termination Control Block	cal_out:inst|cal_out_alt_oct_57p:cal_out_alt_oct_57p_component|sd1a_1	Yes
11	Ok		test_output[2]	Termination Control Block	cal_out:inst|cal_out_alt_oct_57p:cal_out_alt_oct_57p_component|sd1a_2	Yes
12	Ok		test_output[3]	Termination Control Block	cal_out:inst|cal_out_alt_oct_57p:cal_out_alt_oct_57p_component|sd1a_3	Yes

Perform the following steps to verify the assignments.

1. In the Quartus II software, on the Assignments menu, click **Assignment Editor**. The Assignment Editor appears.

   Perform the following steps to verify the assignments.

   The first set of assignments are I/O Standard assignments that are used on the pins of the design. This selection depends on which I/O standards are valid for a particular application. In this design example, the functional output pins are `test_output[3..0]`, which are all assigned the 1.2 V standard.

   The second set of assignments are Output Termination assignments that are assigned to pins that have termination. You can specify the assignments as non-calibrated termination or calibrated termination. In this design example, the functional output pins are `test_output[3..0]`, which are all assigned the Output Termination assignments with the value of **Series 50 Ohm with Calibration**. In this assignment, these pins function as outputs for the FPGA chip only.

   The third set of assignments are Termination Control Block assignments that are used to indicate which calibration block is used to calibrate a particular I/O pin or group. In this design example, the functional output pins are `test_output[3..0]`, which are assigned with the value of `cal_out:inst|cal_out_alt_oct_57p:cal_out_alt_oct_57p_component|sd1a_x`, where `x` indicates the termination calibration block that is used. Because the output pins, `test_output[3..0]`, are four bits, each bit of the pin is assigned to one calibration block.

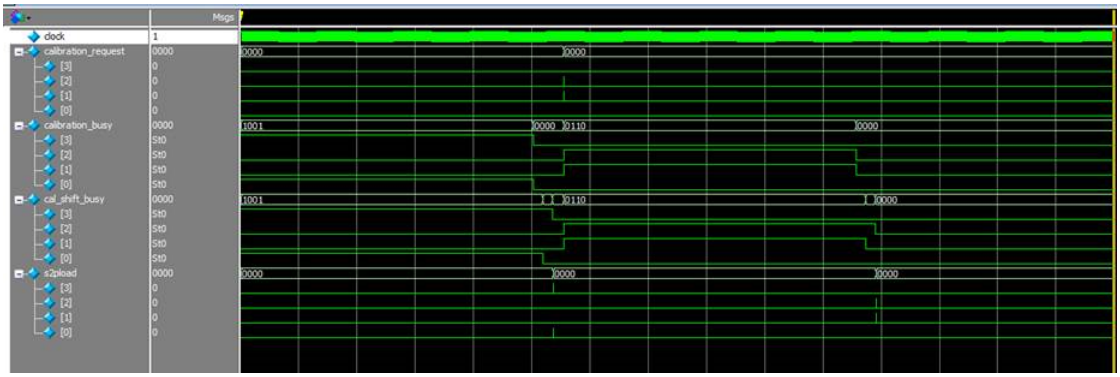
2. Verify that all the assignments in the Assignment Editor match what is shown in **Figure 7**.

Functional Simulation in the ModelSim-Altera Software

The design is simulated in the ModelSim-Altera software to generate a waveform display of the device behavior. You should be familiar with the ModelSim-Altera software before using the design examples.

Figure 8: ModelSim-Altera Simulation Waveforms

This figure shows the expected simulation results in the ModelSim-Altera software.



For these functional results, the primary objective is to highlight the dynamic calibration process. The design example uses four calibration blocks named calibration block [x], with x=0, 1, 2, 3.

Each calibration block has its own set of control and data signals. For example, `calibration block[3]` has `calibration_request[3]`, `calibration_busy[3]`, and `cal_shift_busy[3]` signals.

To request calibration on `calibration block[x]`, the `calibration_request[x]` signal must be asserted for at least 1 clock cycle to initiate the calibration. Multiple calibration block requests are allowed.

The `calibration_busy[x]` signal indicates the calibration process that is being performed in `calibration block[x]`. Calibration typically takes 1000 clock cycles. Calibration is complete when the `calibration_busy[x]` signal is deasserted.

The `cal_shift_busy[x]` signal is asserted at the same time as the `calibration_busy[x]` signal. This signal actually indicates both the duration of the calibration and the serial shifting of termination codes from calibration `block[x]` to the particular I/O buffers. Serial shifting of series termination codes typically takes 32 clock cycles.

After calibrated codes are shifted in serially to the particular I/O bank, the calibrated codes must be converted from serial format to parallel format before being used in the I/O buffers. To perform the conversion, the `s2pload[x]` signal is asserted at any time for 1 clock cycle after the shifting process.

In the simulation waveform shown in **Figure 8**, all of the calibration blocks have their `calibration_request` signals asserted at a particular time. Initially, requests for calibration are done simultaneously for `calibration block[0]` and `calibration_request[3]`. This can be observed in the behavior of the `calibration_request[3]` and `calibration_request[0]` signals. Both signals are asserted and deasserted at the same time.

Observe the behavior of the `calibration_busy[3]` and `calibration_busy[0]` signals. Notice that both are also asserted and deasserted at the same time. This is because multiple OCT calibration blocks can be calibrated at the same time. When these signals get deasserted, calibration is complete.

Next, observe the `cal_shift_busy[3]` and `cal_shift_busy[0]` signals. They are both asserted at the same time as the `calibration_busy[3]` and `calibration_busy[0]` signals, and both signals get deasserted after

a number of clock cycles. These signals indicate the status of calibration and serial shifting of the termination codes.

Notice that the `cal_shift_busy[3]` and `cal_shift_busy[0]` signals get deasserted at different time—`cal_shift_busy[0]` gets deasserted first and followed by `cal_shift_busy[3]`. This is because the shifting process is serial, and only one calibration block can be active at a time. In the event of multiple calibration block requests, the priority `calibration block[x]` is based on this order. x=0, 1, 2, 3, 4, 5, 6, 7, 8, 9, which means `calibration block[0]` shift codes first and then followed by the other blocks in the order. Observe **Figure 8**, the `cal_shift_busy[0]` gets deasserted first and then followed by `cal_shift_busy[3]`. This is because `calibration block[0]` has higher priority than calibration `block[3]`.

Finally, the calibrated codes must be converted from serial format to parallel format before being used in the I/O buffers. This is done by asserting and deasserting the `s2pload[0]` and `s2pload[3]` signals for 1 clock cycle only after `cal_shift_busy[3]` and `cal_shift_busy[0]` signals have been deasserted. See **Figure 8**.

Document Revision History

Table 4: Document Revision History

Date	Version	Changes
July 2014	2014.07.18	• Updated to include Arria V, Cyclone V, and Stratix V devices information, including the design example and user mode calibration. • Removed legacy devices. • Updated template. • Replaced MegaWizard Plug-In Manager information with IP Catalog. • Added standard information about upgrading IP cores. • Added standard installation and licensing information.
February 2012	3.0	Updated the following sections: • "Device Support" section • "MegaWizard Parameter Settings" section.

Date	Version	Changes
November 2008	2.0	▪ Updated the following sections: • "Device Support" section • "Features" section • "Functional Description" chapter • "Design Example: Series Calibration for Four Calibration Blocks Using Stratix III Device" section • "Functional Simulation in the ModelSim-Altera Software" section • "How to Contact Altera" section Removed the following sections: • "Resource Utilization & Performance" section • "Software and System Requirements" section • "Instantiating Megafunctions in HDL Code" section • "Identifying a Megafunction after Compilation" section • "SignalTap II Embedded Logic Analyzer" section • Removed all screenshots.
December 2006	1.0	Initial release.