LVDS SERDES Transmitter/Receiver IP Cores User Guide

2014.12.15

UG-MF9504

The low-voltage differential signaling serializer or deserializer (LVDS SERDES) megafunction IP cores (ALTLVDS_TX and ALTLVDS_RX) implement the LVDS SERDES interfaces to transmit and receive high-speed differential data. You can configure the features of these IP cores with the IP Catalog and parameter editor.

Features

The following table lists the features of the ALTLVDS_TX and ALTLVDS_RX IP cores.

Table 1: ALTLVDS_TX and ALTLVDS_RX Features

Features	Supported devices
ALTLVDS_TX and ALTLVDS_RX	
Parameterizable data channel widths	All Arria®, Cyclone®, and Stratix® series devices.
Parameterizable serializer/deserializer (SERDES) factors	All Arria, Cyclone, and Stratix series devices.
Registered input and output ports	All Arria, Cyclone, and Stratix series devices.
Support for external phase-locked loops (PLL)	All Arria, Cyclone, and Stratix series devices.
PLLs sharing between transmitters and receivers	All Arria, Cyclone, and Stratix series devices.
PLL control signals	All Arria, Cyclone, and Stratix series devices.
ALTLVDS_RX Only	
Dynamic phase alignment (DPA) mode support	All Arria and Stratix (1)devices.
DPA PLL calibration support	All Stratix(1) series devices.

(1) DPA is available starting from Stratix GX onwards. The first generation Stratix device family does not support DPA.

Features	Supported devices
Soft clock data recovery (CDR) mode support	All Arria and Stratix[2] series devices.

Note: Altera recommends implementing the Bus LVDS (BLVDS) I/O with user logic, instead of the ALTLVDS_TX and ALTLVDS_RX IP cores.

Related Information

- **AN 522: Implementing Bus LVDS Interface in Supported Altera Device Families**

Resource Utilization and Performance

The Quartus II software configures the PLL according to the settings you apply in the ALTLVDS_RX and ALTVDS_TX parameter editor. All supported devices provide the option to use an external PLL, which requires you to enter the appropriate PLL parameters.

When the ALTLVDS_TX and ALTLVDS_RX IP cores are instantiated without the external PLL option, they use one PLL per instance. During compilation, if directed to do so, the compiler tries to merge PLLs whenever possible to minimize resource usage.

The Arria, Cyclone, Hardcopy, and Stratix series support the **Use Shared PLL(s) for Receiver and Transmitter** option to allow both the ALTLVDS_TX and ALTLVDS_RX IP cores to share a PLL. The Quartus II software lets the transmitter and receiver share the same PLL when both use identical input clock sources, identical `pll_areset` sources, identical deserialization factors, and identical output settings. For example, the Quartus II software displays the following message when the PLL merges successfully:

```
Info: Receiver fast PLL <lvds_rx PLL name>
                and transmitter fast PLL <lvds_tx PLL name> are
merged
                together
```

The Quartus II software displays the following message when it cannot merge the PLLs for the LVDS transmitter and receiver pair in the design:

```
Warning: Can't merge transmitter-only fast PLL
                <lvds_tx PLL name> and receiver-only fast PLL
<lvds_rx PLL
                name>
```

Note: One cause for the warning message is that PLLs that are driven by different clocks cannot be merged. For PLL merging to happen, the input clocks and the settings on the outputs must be identical.

Note: To use the LVDS I/O standard in the I/O Bank 1 of Cyclone III and Cyclone IV E devices, ensure that you set the **Configuration device I/O voltage** to 2.5 V, or Auto in the **Device and Pin Options** dialog box of the Quartus II software.

[2] CDR is not available in the first generation Stratix device family and the Stratix II device family. However, soft-CDR is available in all other Stratix series including Stratix GX and Stratix II GX..

For the Stratix series, the side I/O banks contain dedicated SERDES circuitry, which includes the PLLs, serial shift registers, and parallel registers. The transmit and receive functions use varying numbers of LEs depending on the number of channels, serialization, and deserialization factors. For best performance, manually place these LEs in columns as close as possible to the SERDES circuitry and LVDS pins. By default, the Quartus II software places these LEs automatically during placement and routing.

Note: When dedicated SERDES is implemented in LVDS transmitter, the SERDES is directly connected to the LVDS transmitter; therefore, the output of the transmitter cannot be assigned to single-ended I/O standards.

Note: The Quartus II software reports the number of LEs used per ALTLVDS block in the **Fitter Resource Utilization by Entity** section in the **Resource** section of the **Compilation Report**.

The Cyclone series uses DDIO registers as part of the SERDES interface. Because data is clocked on both the rising edge and falling edge, the clock frequency must be half the data rate; therefore, the PLL runs at half the frequency of the data rate. The core clock frequency for the transmitter is data rate divided by serialization factor (J). For the odd serialization factors, depending on the output clock-divide factor (B) and device family, an optional core clock frequency of data rate divided by two times the serialization factor (J) is also available.

Use the following tables to determine the clock and data rate relationships.

Table 2: Cyclone Series ALTLVDS Transmitter Clock Relationships

Clock Type	J = Even	J = Odd
Fast Clock	Data Rate / 2	Data Rate / 2
Slow Clock (outclock)	Data Rate / 2 * B	Data Rate / 2 * B
Core Clock	Data Rate / J	Data Rate / J

Table 3: Cyclone Series ALTLVDS Receiver Clock Relationships

Clock Type	J = Even	J = Odd
Fast Clock	Data Rate / 2	Data Rate / 2
Slow Clock (outclock)	Data Rate / J	Data Rate / J

Related Information

ALTPLL IP Core User Guide

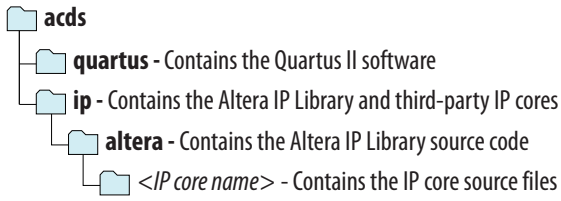
Altera Phase-Locked Loop (Altera PLL) IP Core User Guide

Installing and Licensing IP Cores

The Altera IP Library provides many useful IP core functions for production use without purchasing an additional license. You can evaluate any Altera® IP core in simulation and compilation in the Quartus® II software using the OpenCore® evaluation feature. Some Altera IP cores, such as MegaCore® functions, require that you purchase a separate license for production use. You can use the OpenCore Plus feature to evaluate IP that requires purchase of an additional license until you are satisfied with the functionality and

performance. 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 and parameter editor allow you to quickly select and configure IP core 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.

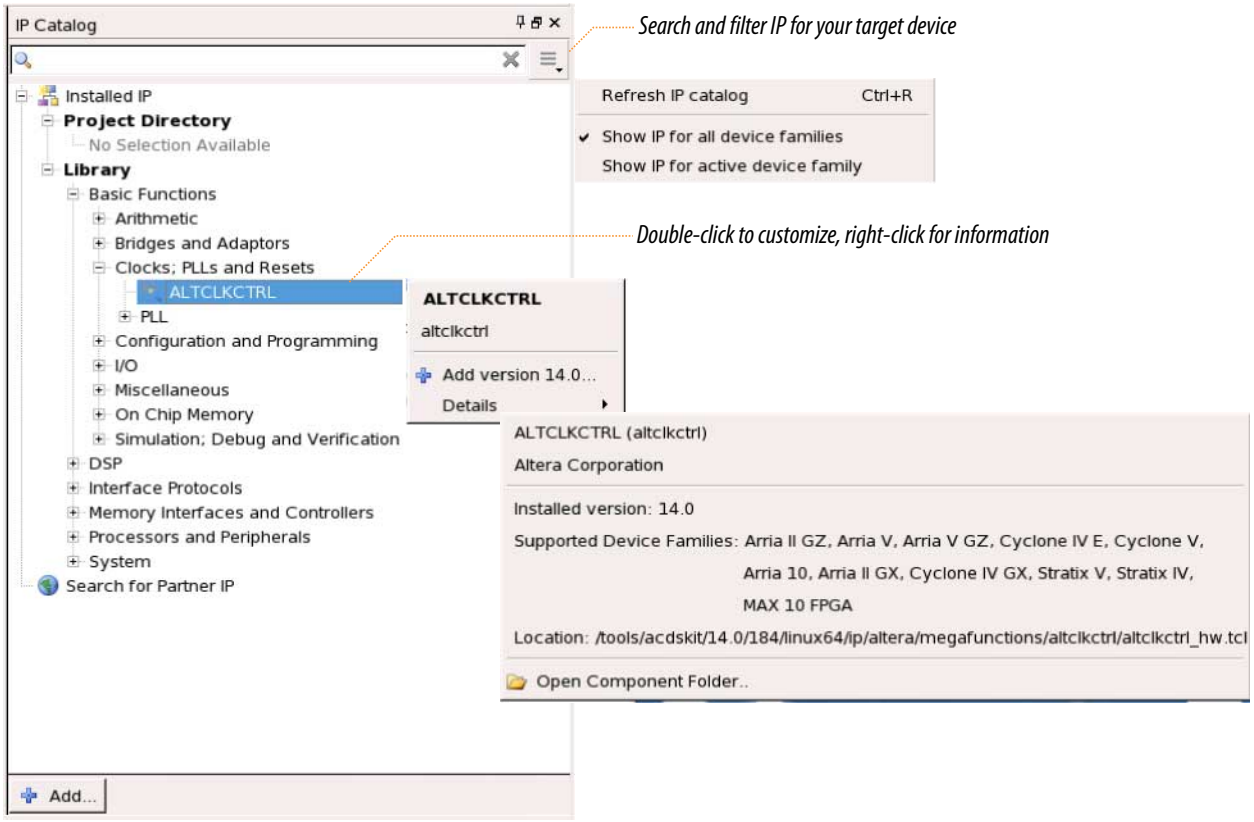
Note: The IP Catalog (**Tools** > **IP Catalog**) and parameter editor replace the MegaWizard™ Plug-In Manager for IP selection and parameterization, beginning in Quartus II software version 14.0. Use the IP Catalog and parameter editor to locate and paramaterize Altera IP cores.

The IP Catalog lists IP cores available for your design. Double-click any IP core to launch the parameter editor and generate files representing your IP variation. The parameter editor prompts you to specify an IP variation name, optional ports, and output file generation options. The parameter editor generates a top-level Qsys system file (**.qsys**) or Quartus II IP file (**.qip**) representing the IP core in your project. You can also parameterize an IP variation without an open project.

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, open the IP core's installation folder, andor view links to documentation.

Figure 2: Quartus II IP Catalog



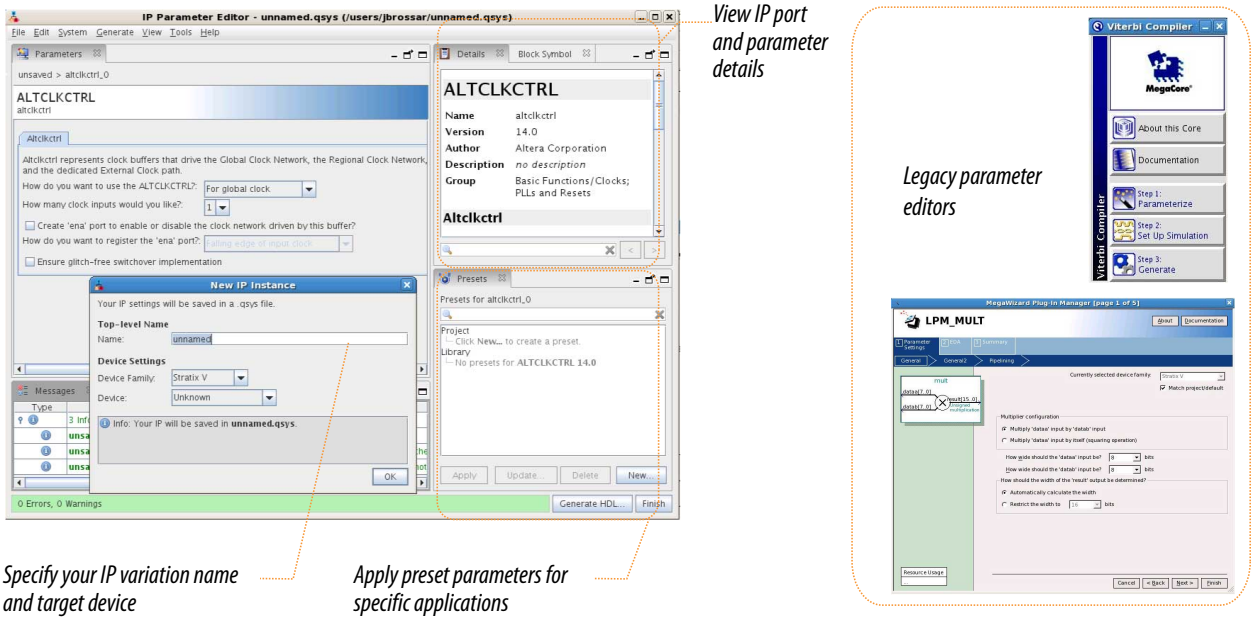
Note: The IP Catalog is also available in Qsys (**View** > **IP Catalog**). The Qsys IP Catalog includes exclusive system interconnect, video and image processing, and other system-level IP that are not available in the Quartus II IP Catalog. For more information about using the Qsys IP Catalog, refer to *Creating a System with Qsys* in the *Quartus II Handbook*.

Using the Parameter Editor

The parameter editor helps you to configure IP core ports, parameters, 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



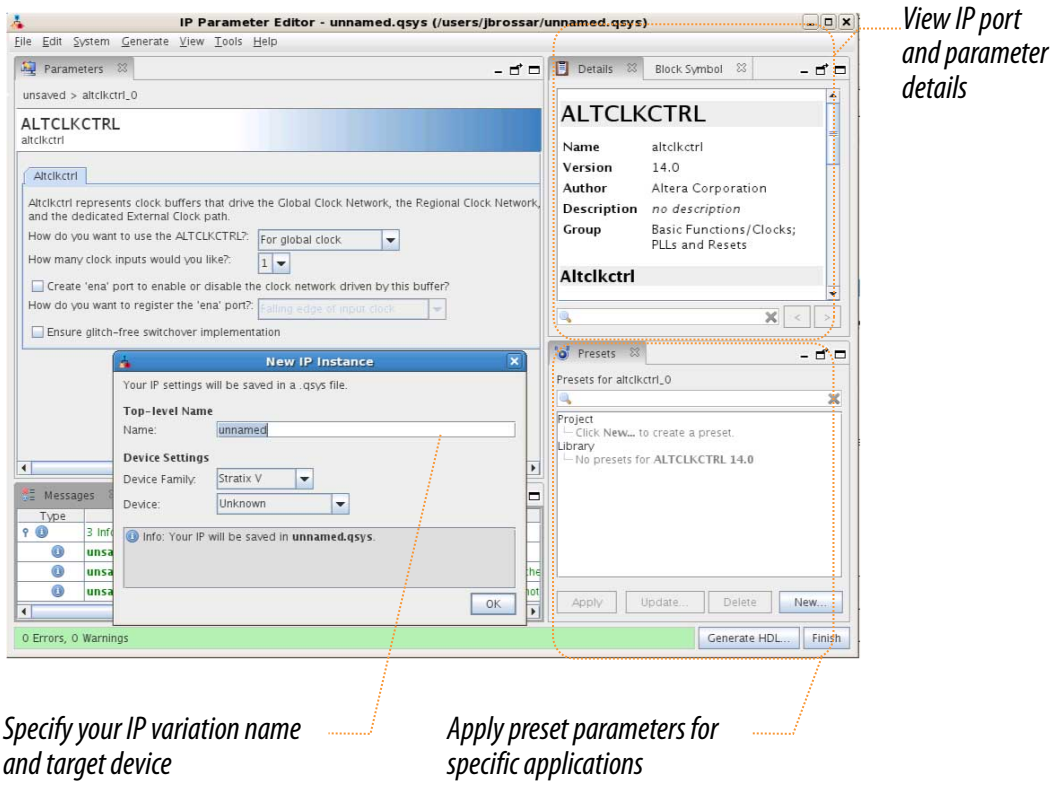
Specifying IP Core Parameters and Options

The parameter editor GUI allows you to quickly configure your custom IP variation. Use the following steps to specify IP core options and parameters in the Quartus II software. Refer to *Specifying IP Core Parameters and Options (Legacy Parameter Editors)* for configuration of IP cores using the legacy parameter editor.

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. The parameter editor saves the IP variation settings in a file named <*your_ip*>**.qsys**. Click **OK**.
3. Specify the parameters and options for your IP variation in the parameter editor, including one or more of the following. Refer to your IP core user guide for information about specific IP core parameters.
   - Optionally select preset parameter values if provided for your IP core. Presets specify initial parameter values for specific applications.
   - Specify parameters defining the IP core functionality, port configurations, and device-specific features.
   - Specify options for processing the IP core files in other EDA tools.
4. Click **Generate HDL**, the **Generation** dialog box appears.
5. Specify output file generation options, and then click **Generate**. The IP variation files generate according to your specifications.
6. To generate a simulation testbench, click **Generate** > **Generate Testbench System**.

7. To generate an HDL instantiation template that you can copy and paste into your text editor, click **Generate** > **HDL Example**.
8. Click **Finish**. The parameter editor adds the top-level **.qsys** file to the current project automatically. If you are prompted to manually add the **.qsys** file to the project, click **Project** > **Add/Remove Files in Project** to add the file.
9. After generating and instantiating your IP variation, make appropriate pin assignments to connect ports.

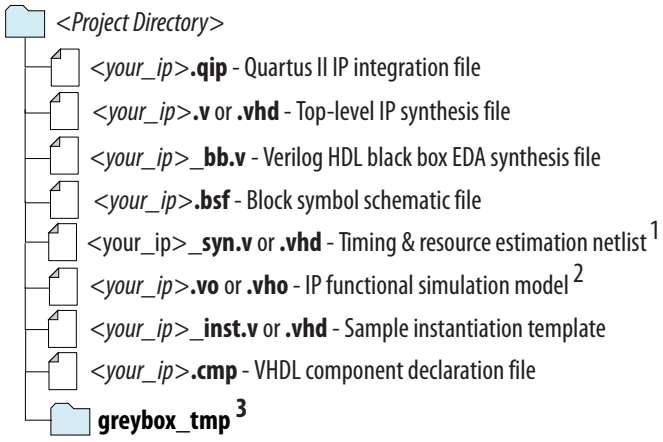
Figure 4: IP Parameter Editor



Files Generated for Altera IP Cores (Legacy Parameter Editor)

The Quartus II generates the following output for IP cores that use the legacy MegaWizard parameter editor.

Figure 5: IP Core Generated Files



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

Upgrading IP Cores

IP core variants generated with a previous version of the Quartus II software may require upgrading before use in the current version of the Quartus II software. Click **Project** > **Upgrade IP Components** to identify and upgrade IP core variants.

The **Upgrade IP Components** dialog box provides instructions when IP upgrade is required, optional, or unsupported for specific IP cores in your design. You must upgrade IP cores that require it before you can compile the IP variation in the current version of the Quartus II software. Many Altera IP cores support automatic upgrade.

The upgrade process renames and preserves the existing variation file (**.v**, **.sv**, or **.vhd**) as <*my_variant*>_**BAK.v**, **.sv**, **.vhd** in the project directory.

Table 4: IP Core Upgrade Status

IP Core Status	Corrective Action
Required Upgrade IP Components	You must upgrade the IP variation before compiling in the current version of the Quartus II software.
Optional Upgrade IP Components	Upgrade is optional for this IP variation in the current version of the Quartus II software. You can upgrade this IP variation to take advantage of the latest development of this IP core. Alternatively you can retain previous IP core characteristics by declining to upgrade.

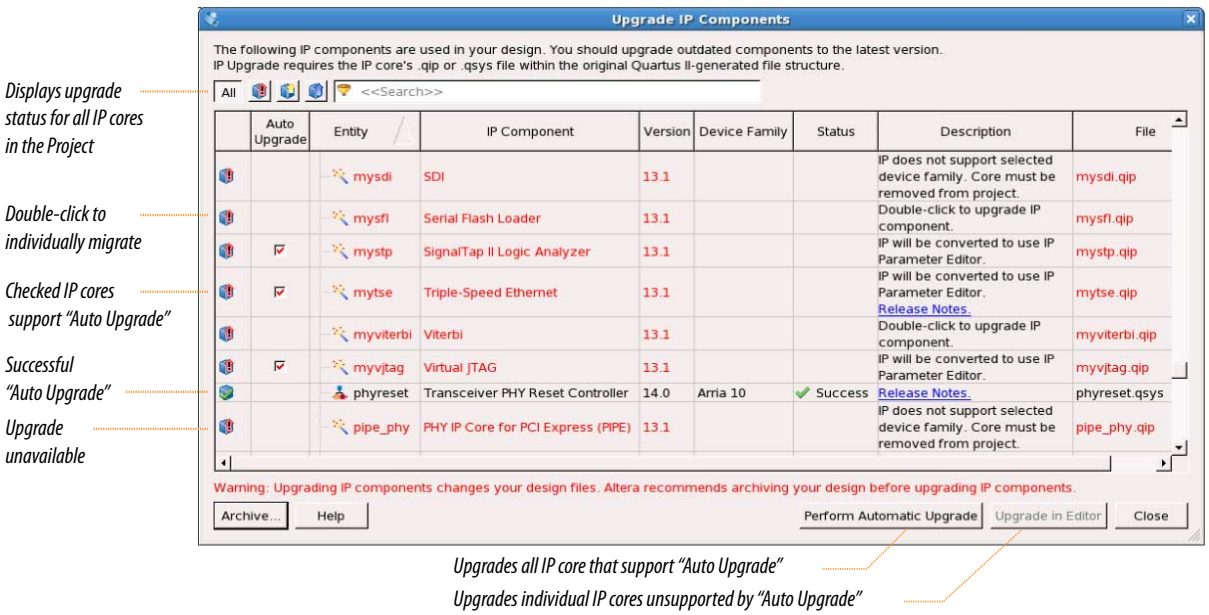
IP Core Status	Corrective Action
Upgrade Unsupported	Upgrade of the IP variation is not supported in the current version of the Quartus II software due to IP core end of life or incompatibility with the current version of the Quartus II software. You are prompted to replace the obsolete IP core with a current equivalent IP core from the IP Catalog.

Before you begin

- Archive the Quartus II project containing outdated IP cores in the original version of the Quartus II software: Click **Project** > **Archive Project** to save the project in your previous version of the Quartus II software. This archive preserves your original design source and project files.
- Restore the archived project in the latest version of the Quartus II software: Click **Project** > **Restore Archived Project**. Click **OK** if prompted to change to a supported device or overwrite the project database. File paths in the archive must be relative to the project directory. File paths in the archive must reference the IP variation **.v** or **.vhd** file or **.qsys** file (not the **.qip** file).

1. In the latest version of the Quartus II software, open the Quartus II project containing an outdated IP core variation. The **Upgrade IP Components** dialog automatically displays the status of IP cores in your project, along with instructions for upgrading each core. Click **Project** > **Upgrade IP Components** to access this dialog box manually.
2. To simultaneously upgrade all IP cores that support automatic upgrade, click **Perform Automatic Upgrade**. The **Status** and **Version** columns update when upgrade is complete. Example designs provided with any Altera IP core regenerate automatically whenever you upgrade the IP core.

Figure 6: Upgrading IP Cores



Example 1: Upgrading IP Cores at the Command Line

You can upgrade IP cores that support auto upgrade at the command line. IP cores that do not support automatic upgrade do not support command line upgrade.

- To upgrade a single IP core that supports auto-upgrade, type the following command:

```
quartus_sh –ip_upgrade –variation_files <my_ip_filepath/my_ip>.<hdl>
<qii_project>

Example:
quartus_sh -ip_upgrade -variation_files mega/pll25.v hps_testx
```

- To simultaneously upgrade multiple IP cores that support auto-upgrade, type the following command:

```
quartus_sh –ip_upgrade –variation_files “<my_ip_filepath/my_ip1>.<hdl>;
<my_ip_filepath/my_ip2>.<hdl>”  <qii_project>

Example:
quartus_sh -ip_upgrade -variation_files "mega/pll_tx2.v;mega/pll3.v"
hps_testx
```

Note: IP cores older than Quartus II software version 12.0 do not support upgrade. Altera verifies that the current version of the Quartus II software compiles the previous version of each IP core. The *Altera IP Release Notes* reports any verification exceptions for Altera IP cores. Altera does not verify compilation for IP cores older than the previous two releases.

Related Information

Altera IP Release Notes

Parameter Settings

You can parameterize IP cores using the IP Catalog and parameter editor.

Related Information

- **Command Line Interface Parameters** on page 26

ALTLVDS_TX Parameter Settings

On the **General** page (page 3) of the parameter editor, depending on the device you selected, you can configure the following options:

- Implement the SERDES circuitry in LEs (logic cells) or dedicated (hard) SERDES block
- Use internal PLL or external PLL

The selections you make on the **General** page determine the features available on the remaining pages of the parameter editor.

The options on pages 1 and 2a of the parameter editor are the same for all supported device families.

The following table lists the parameter settings for the ALTLVDS_TX IP core.

Table 5: ALTLVDS_TX Parameter Settings

Option	Description
General (page 3)	
Implement Deserializer circuitry in logic cells	Turn on this option to implement the SERDES circuitry in logic cells. The transmitter starts its operation on the first fast clock edge after the PLL is locked. This option is intended for slow speeds. The byte alignment might be different from the dedicated SERDES implementation. Turn off this option to use the dedicated SERDES circuitry in the device. When you implement the dedicated SERDES in the LVDS transmitter, the SERDES connects to the LVDS transmitter; therefore, the output of the transmitter cannot be assigned to single-ended I/O standards. This feature is supported in Arria GX, Arria II GX, Arria II GZ, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, and Stratix IV devices. In Cyclone series, except Cyclone V devices, the SERDES is always implemented in logic cells. Cyclone V devices contain dedicated SERDES circuitry. If you turn on this option, there is additional delay for the `tx_outlock` signal to be stable after the `tx_locked` signal is asserted. Perform gate-level simulation to determine the time for the `tx_outclock` signal to stabilize.
What is the number of Channels?	Number of output channels available for the LVDS transmitter. If the required number of channels is not available in the list, type the desired number. For example, if the number of channels is **44**, the port created is `tx_out[43..0]`. The legal values depend on the pins available in the device. For the legal values for your device, refer to the relevant device handbook.
What is the deserialization factor?	Determines the number of parallel bits from the core that the transmitter serializes and sends out. For example, if the deserialization factor is **10** and the number of output channels is **1**, the transmitter serializes every 10 parallel bits into a single output channel. If the deserialization factor is **10** and the number of channels is **44**, the port created is `tx_in[439..0]`. For the valid deserialization factors for your device, refer to the relevant device handbook. When the `divide_by_factor` port shown in the parameter editor is identical to the deserialization factor, the parameter editor disables the 50/50 duty cycle for x5, x7, and x9 modes.

Option	Description
Use External PLL	Turn on this option to use an external PLL to clock the SERDES transmitter. When you turn on this option, the options on the **Frequency/PLL** settings page are disabled. You must use a separate PLL to provide the clocking source and make the necessary connections. You must ensure your circuit has the correct input and functionality to generate an appropriate clock frequency and is correctly connected to the LVDS transmitter. When you have a deserialization factor of two, the IP core bypasses SERDES and implements the SERDES functionality in DDR registers. Your design requires a deserialization factor of at least four to turn on the external PLL option. If you turn off this option, the IP core automatically implements an internal PLL to clock the ALTLVDS_TX block. For Stratix and Stratix GX devices, if you implement SERDES for your LVDS transmitter using a dedicated SERDES block, you do not have the option to use an external PLL.
Use 'tx_data_reset' input port	This option is available when you implement the LVDS in logic cells. When you turn on this option, it adds an input port in the IP core, which when asserted asynchronously resets all the logic in the ALTLVDS_TX IP core excluding the PLL.
Frequency/ PLL Settings (page 4) The options on this page are available only when you are using internal PLL	
What is the output data rate?	Specifies the data rate for the output channel of the transmitter, in Megabits per second (Mbps). For data rate ranges, refer to the Device Data Sheet chapter in the relevant device handbook. This option determines the legal value of the input clock rate.
Specify input clock rate by	Specifies the clock frequency (`tx_inclock` port) or the clock (`inclock_period` parameter) going into the internal PLL. The legal values depend on the output data rate selected.
What is the phase alignment of 'tx_in' with respect to the rising edge of 'tx_inclock'? (in degrees)	Determines the phase alignment of the data transmitted by the core logic array with respect to the `tx_inclock` clock. The available values are **0.00**, **22.50**, **45.00**, **67.50**, **90.00**, **112.50**, **135.00**, **157.50**, **180.00**, **202.50**, **225.00**, **247.50**, **270.00**, **292.50**, **315.00**, and **337.50**. The values for this option are device dependent.

Option	Description
Use 'tx_pll_enable' input port	Turn on to control the enable port of the fast PLL that the IP core uses with this function. If the transmitter shares the PLL with other ALTLVDS blocks, and uses the `tx_pll_enable` port, you must use this port in all the IP core instances and tie the signals together in the design file. If you use a PLL-enabled port in one IP core instance and not another, the PLLs are not shared, and a warning appears during compilation.
Use 'pll_areset' input port	Turn on to control the asynchronous reset port of the PLL that the IP core uses with this function. When the transmitter shares the PLL with other ALTLVDS blocks and uses the `pll_areset` port, you must use this port in all the IP core instances and tie the signals together in the design file. If you use the `pll_areset` port in one IP core instance only, the PLLs are not shared and a warning appears during compilation. The PLL must be reset to set the output clock phase relationships correctly when the PLL loses lock, or if the PLL input reference clock is not stable when the device completes the configuration process.
Align clock to center of data window	Turn on this option to add a phase shift of 90° to the clock, which center-aligns the clock in the data. Turn on this option for PLL merging if you also turn on this option for the receiver. This option is available only for Arria GX, Stratix II, Stratix II GX, and HardCopy II devices when you implement the SERDES in logic cells, and for Cyclone II devices.
Enable self-reset on lost lock in PLL	Turn on this option to reset the PLL automatically whenever the PLL loses lock. This option is available only for Arria II GX, Arria II GZ, HardCopy III, HardCopy IV, Stratix III, and Stratix IV devices when SERDES is implemented in logic cells, and for Cyclone lll and Cyclone lV devices.
Use shared PLL(s) for receivers and transmitters	Turn on this option for your LVDS receivers and transmitters to share the same PLL. Turn on this option if the LVDS receivers and transmitters use the same input clock frequency, deserialization factor, and data rates.

Option	Description
Register 'tx_in' input port using	Turn on this option to specify whether input registers are clocked by the `tx_inclock` signal or `tx_coreclock` signal. When the PLLs are shared, connect the `tx_inclock` signal to the same reference clock as the receiver function. For example, if the `tx_inclock` signal is connected to a 500-MHz input reference clock, and the parallel data rate is not 500 MHz, register the parallel data using the `tx_coreclock` signal that runs at the output serial data rate divided by the deserialization factor. This frequency matches the parallel data rate from the FPGA core. If you turn off this option, a warning message appears that directs you to pre-register the inputs in the logic that feeds the transmitter. When you use the Cyclone series with the ALTLVDS_TX and ALTLVDS_RX IP cores, the interface always sends the most significant bit (MSB) of your parallel data first. When you use the ALTLVDS_TX IP core, you might get setup timing violations when you use the `tx_inclock` signal to register the data that feeds the SERDES blocks. The ALTLVDS_TX IP core gives you the choice to register the `tx_in[]` data with either the `tx_inclock` or `tx_coreclock` signal. The default setting is `tx_coreclock`. Using the `tx_coreclock` signal to register the data before it feeds the SERDES is the better choice, because it has the optimal phase position to register the data with respect to the high-speed clock that drives the SERDES. Your setup timing violations are eliminated when you use the `tx_coreclock` signal instead of the `tx_inclock` signal to register the data in the ALTLVDS_TX IP core. Additionally, you get better timing margins when you use the `tx_coreclock` signal instead of the `tx_inclock` signal, even if you do not have timing violations.
Transmitter Settings (page 5)	
Use 'tx_outclock' output port	The `tx_outclock` signal is associated with the serial transmit data stream. Every `tx_outclock` signal goes through the shift register logic, excluding the following parameter configurations: • When the `outclock_divide_by` signal equals to 1, or • When the `outclock_divide_by` signal equals to `deserialization_factor` signal (for odd factors only) and the `outclock_duty_cycle` signal is **50**.

Option	Description
What is the outclock divide factor (B)?	Specifies the frequency of the `tx_outclock` signal as the transmitter output data rate divided by the outclock divide factor (B). For the legal values, refer to the relevant device handbook. For a SERDES factor of **5** and **9**, the outclock divide factors available are **1**, **5**, and **9**. The divide factor of **2** is not available. For Cyclone II devices and later, when the `implement_in_les` parameter is `ON`, the `outclock_duty_cycle` of **50** is not supported in the following parameter configurations: • deserialization_factor signal is **5**, **7**, or **9** • `outclock_divide_by` signal equals to `deserialization_factor` • `outclock_multiply_by` is **2**
Specify phase alignment of 'tx_outclock' with respect to 'tx_out'	Specifies the phase alignment of `tx_outclock` signal with respect to the `tx_out` signal. This option is available only if you use the `tx_outclock` signal.
What is the phase alignment of 'tx_outclock' with respect to 'tx_out'?	The available values are **0.00**, **22.50**, **45.00**, **67.50**, **90.00**, **112.50**, **135.00**, **157.50**, **180.00**, **202.50**, **225.00**, **247.50**, **270.00**, **292.50**, **315.00**, and **337.50**. The values for this option are device dependent. This option is available only when you implement the SERDES in logic cells and uses the `tx_outclock` signal.
What is the outclock duty cycle?	The default value is **50**. The `outclock_duty_cycle` of **50** is not supported when: • `deserialization_factor` signal is **5**, **7**, or **9** • `outclock_divide_by` signal equals to `deserialization_factor` • `outclock_multiply_by` is **2**
Use 'tx_locked' output port	Allows you to monitor the lock status of the PLL. The status of the lock port is identical for the transmitter and receiver when the IP core uses shared PLLs.
Use 'tx_coreclock' output port	Turn on this option to show the core clock frequency during simulation. Enables the transmitter core clock signal to the registers of all the logic that feeds the LVDS transmitter function. If any other clock feeds the transmit function, your design must implement the clock domain transfer circuitry. You must add a false path constraint from the `slow_clock` signal to the `fast_clock` signal in the ALTLVDS_TX IP core whenever the faster `core_clock` signal implementation is used for odd deserialization factors.

Option	Description
What is the clock resource used for 'tx_coreclock'?	Specifies the clock resource type fed to the `tx_coreclock` signal. Allowed values are **Auto selection** (the Compiler determines the type), **Global clock**, and **Regional clock**. The default value is **Auto selection**.
Simulation Model (page 6)	
Simulation Libraries	Specifies the libraries needed for functional simulation by third-party tools.
Generate netlist	Specifies whether to turn on the option to generate synthesis area and timing estimation netlist.
Summary (page 7)	
Summary	Specifies the types of files to be generated. A gray checkmark indicates a file that is automatically generated; a green checkmark indicates an optional file. Choose from the following types of files: • AHDL Include file (<*function name*>**.inc**) • VHDL component declaration file (<*function name*>**.cmp**) • Quartus II symbol file (<*function name*>**.bsf**) • Instantiation template file (<*function name*>**_inst.v** or <*function name*>**_inst.vhd** • Verilog HDL block box file (<*function name*>**_bb.v**) • Pin Planner File (<*function name*>**_.ppf**) If you turn on the **Generate netlist** option, the file for that netlist is also available (<*function name*>**_syn.v**).

Related Information

- **Introduction to Altera IP Cores**

ALTLVDS_RX Parameter Settings

On the **General** page (page 3) of the parameter editor, depending on the device you selected, you can configure the following options:

- Implement the SERDES circuitry in LEs (logic cells) or dedicated SERDES
- Use internal PLL or external PLL
- Use DPA mode or non-DPA mode

The selections you make on the **General** page determine the features available on the remaining pages of the parameter editor.

The following table lists the parameter settings for the LVDS receiver IP core.

Table 6: ALTLVDS_RX Parameter Settings

Option	Description
General (page 3)	
Implement Deserializer circuitry in logic cells	Turn on this option to implement the SERDES circuitry in logic cells. The receiver starts its operation on the first fast clock edge after the PLL is locked. This option is intended for slow speeds. The byte alignment may be different from the hard SERDES implementation. Turn off this option to use the dedicated SERDES circuitry in the device. This option is supported in Arria GX, Arria II GX, Arria II GZ, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, and Stratix IV devices. In Cyclone series, except Cyclone V devices, the SERDES is always implemented in logic cells. Cyclone V devices contain dedicated SERDES circuitry.
Enable Dynamic Phase Alignment mode	Turn on this option to correct the skews created by the different trace lengths on the data channels routed to the device. This mode adds several ports and parameters to the IP core instances. This option is available for Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V devices only. Enabling the DPA mode changes the appearance of the graphic representation of the IP core in the left-hand pane. When you turn on the DPA mode, additional ports and parameters are added to the IP core. Depending on the selected device, the following pages are added to the parameter editor to include the additional DPA mode settings: • **DPA settings 1** • **DPA settings 2** • **DPA settings 3**
What is the number of channels?	The number DPA settings 3of input channels available for the LVDS receiver. If the required number of channels is not available in the list, type the desired number in this box. For example, if the number of channels is **44**, the port created is `tx_out[43..0]`. The legal values depend on the pins available in the device. For the legal values available for your device, refer to the relevant device handbook.

Option	Description
What is the deserialization factor?	Determines the number of serial input data bits that the receiver deserializes and sends to the core on a single cycle. For the valid deserialization factors for your device, refer to the relevant device handbook. For example, if the deserialization factor is **10** and the number of input channels is 1, the receiver deserializes every 10 serial bits into 10 bits of parallel data to send to the core. If the deserialization factor is 10 and the number of channels is **44**, the port created is `rx_out[439..0]`.
Use External PLL	Turn on this option to use an external PLL to clock the SERDES receiver. When you turn on this option, the options on the **Frequency/PLL settings** page are disabled. You must use a separate PLL to provide the clocking source and make the necessary connections. You must ensure your circuit has the correct input and functionality to generate an appropriate clock frequency and is correctly connected to the LVDS receiver. When you have a deserialization factor of two, the IP core bypasses the SERDES and implements the SERDES functionality in DDR registers. A deserialization factor of at least four is required to use the external PLL option. If you turn off this option, the IP core automatically implements an internal PLL to clock the ALTLVDS_RX block. For Stratix and Stratix GX devices, if you implement SERDES for your LVDS transmitter using a dedicated SERDES block, you do not have the option to use an external PLL.
Use 'rx_data_reset' input port	This option is enabled when you implement the LVDS in logic cells. Turn on this option to add an input port to the IP core. When the input port asserts, the IP core asynchronously resets all the logic in the ALTLVDS_RX IP core excluding the PLL.
Is this interface constrained to the left, or right banks?	Turn on this option if the LVDS interface is constrained to the left or right IO banks. This option determines the PLL compensation mode in Cyclone V devices.
Frequency/ PLL Settings (page 4) The options on this page are available only when you are using internal PLL	
What is the input data rate?	Specifies the data rate for the input channel of the receiver, in Mbps. For data rate ranges, refer to the specific Device Data Sheet chapter in the respective device handbook. This value determines the legal input clock rate values.
Specify input clock rate by	Specifies the clock frequency (`rx_inclock`) and the clock period (`inclock_period`) for the internal PLL. The legal values depend on the output data rate selected.

Option	Description
Use shared PLL(s) for receivers and transmitters	When you turn on this option, your LVDS receivers and transmitters can share the same PLL. Turn on this option when the LVDS receivers and transmitters use the same input clock frequency, deserialization factor, and data rates.
Use 'pll_areset' input port	Turn on this option to control the asynchronous reset port of the PLL that the IP core uses with this function. When other ALTLVDS blocks share the PLL with the receiver and use the `pll_areset` port, you must use this port in all IP core instantiations and tie the signals together in the design file. If you use the `pll_areset` port only in one IP core instance, the PLLs are not shared, and a warning appears during compilation. The PLL must be reset to set the output clock phase relationships correctly when the PLL loses lock, or if the PLL input reference clock is not stable when the device completes the configuration process.
Use 'rx_pll_enable' input port	Turn on this option to control the enable port of the fast PLL that the IP core uses with this function. If the receiver shares the PLL with other ALTLVDS blocks, and uses the `rx_pll_enable` port, you must use this port in all IP core instances and tie the signal together in the design file. If you use the `rx_pll_enable` port only in one IP core instance, the PLLs are not shared and a warning appears during compilation.
Use 'rx_locked' output port	Turn on this option to monitor the lock status of the PLL. The status of the lock port is identical for the transmitter and the receiver when the IP cores use shared PLLs. In this case, monitor the lock output from the receiver IP core.
What is the clock resource used for 'rx_outclock'?	Specifies the clock resource type fed from the `rx_outclock` port. Legal values are **Auto selection** (the Compiler determines the type), **Global clock**, and **Regional clock**. The default value is **Auto selection**.
What is the phase alignment of 'rx_in' with respect to 'rx_inclock'?	Determines the phase alignment of the data that the receiver core receives with respect to the rx_inclock signal. Available values are **0.00**, **22.50**, **45.00**, **67.50**, **90.00**, **112.50**, **135.00**, **157.50**, **180.00**, **202.50**, **225.00**, **247.50**, **270.00**, **292.50**, **315.00**, and **337.50**. The values for this option are device dependent. This option is only available if you turn off the DPA mode.

Option	Description
Use source-synchronous mode of the PLL	Turn on this option to ensure that the IP core instance makes the required phase adjustment to guarantee a consistent relationship between the clock and the data, at the capture register and at the pin. Always turn on this option, unless you have performed all of the necessary phase adjustments manually. Altera recommends that you turn on this option when you use non-dedicated SERDES schemes. This option is only available when you implement the SERDES in LEs.
Align clock to center of data window at capture point	Turn on this option to add a phase shift of 90° to the clock, which center-aligns the clock in the data. This option is only available for Arria GX, Cyclone II, Stratix II GX, Stratix II, and HardCopy II devices when you implement the SERDES in logic cells.
Enable self-reset on lost lock in the PLL	Turn on this option to reset the PLL automatically when the PLL loses lock. This option is only available for Arria II GX, Arria II GZ, HardCopy III, HardCopy IV, Stratix III, Stratix IV, Cyclone III and Cyclone IV devices when you implement the SERDES in logic cells.
Enable FIFO for DPA channels	The phase-compensation FIFO buffer synchronizes parallel data to the global clock domain of the core. This option is only available in Stratix GX devices when you turn on the DPA mode.
DPA Settings 1 (page 5) The options on this page are available when you turn on the DPA mode.	
Use 'rx_divfwdclk' output port and bypass the DPA FIFO	Turn on this option to divide the DPA clock by the deserialization factor and then forward the DPA clock to the core. The DPA clock drives the bit-slip and alignment circuitry, bypassing the FIFO. Turn on this option for soft-CDR mode. This option is available in Arria II GX, Arria II GZ, Arria V, Arria V GZ, HardCopy III, HardCopy IV, Stratix III, Stratix IV, and Stratix V devices only.
What is the simulated recovered clock phase drift?	Models a phase drift in the recovered clock. Clock phase drift is expressed as the equivalent number of full clock cycles of drift for every parts per million (PPM) clock cycles. The value for this option can be positive, negative or zero.

Option	Description
Use 'rx_dpll_enable' input port	Enables the path through the DPA circuitry. The option supports dynamic, channel-by-channel control of the DPA circuitry. To enable the DPA circuitry for a channel, set the port for the target channel to `1`. If this port is not used, the Quartus II software enables all of the channels.
Use 'rx_dpll_hold' input port	Prevents the DPA circuitry from switching to a new clock phase on the target channel. Each DPA block monitors the phase of the incoming data stream continuously and selects a new clock phase when needed. When this port is held high, the selected channels hold their current phase setting.
Use 'rx_fifo_reset' input port	Resets the FIFO buffer between the DPA circuit and the data alignment circuit. The FIFO buffer holds the data passing between the DPA and the LVDS clock domains. When this port is held high, the FIFOs in the selected channels are reset. This option is available only if you turn off the **Use 'rx_divfwdclk' output port and bypass the DPA FIFO** option.
DPA Settings 2 (page 6) The options on this page are available when you turn on the DPA mode.	
Use 'rx_reset' input port	Resets all components of the DPA circuit. You must retrain the DPA circuit after this port resets the DPA circuitry.
Automatically reset the bit serial FIFO when 'rx_dpa_locked' rises for the first time	Specifies when the bit-serial FIFO resets for DPA circuit. This option is only available in Stratix II, Arria GX, and HardCopy II devices.
User explicitly resets the bit serial FIFO through 'rx_reset'	When you turn on the `rx_reset` port, the ALTLVDS_RX parameter editor allows you to choose whether or not to automatically reset the bit-serial FIFO when `rx_dpa_locked` signal rises for the first time. This is a useful feature because it keeps the synchronizer FIFO in reset until the DPA locks. This option is only available in Stratix II, Arria GX, and HardCopy II devices.

Option	Description
Use 'rx_dpa_locked' output port	The DPA block samples the data on one of eight phase clocks with a 45° resolution between phases. This port lets you monitor the status of the DPA circuit and determine when it has locked onto the phase closest to the incoming data phase. The `rx_dpa_locked` port behaves differently for various device families. After the IP core asserts the `rx_dpa_locked` signal is upon initial lock, the `rx_dpa_locked` signal does not deassert in Arria V, Arria V GZ, Stratix III, Stratix IV, Stratix V, HardCopy III, HardCopy IV, and Arria II GX unless explicitly reset using `rx_reset` or `rx_dpa_lock_reset`. In Stratix GX, Stratix II, HardCopy II, and Arria GX, the `rx_dpa_locked` signal toggles depending on how the next two settings are selected. After power up or reset, the `rx_dpa_locked` signal is asserted after the DPA circuitry acquires an initial lock to the optimum phase. You must not use the `rx_dpa_locked` signal to validate the integrity of the LVDS link. Use error checkers (for example, CRC or DIP4) to validate the integrity of the LVDS link. The `rx_dpa_locked` signal is not supported when using non-DPA mode or soft-CDR mode.
When phase alignment circuitry switches to a new phase	DPA deasserts when the phase alignment circuitry switches to a new phase. This option is only available in Stratix II, HardCopy II, and Arria GX devices.
When there are two phase changes in the same direction	The `rx_dpa_locked` signal deasserts after the DPA switches two phases in the same direction. This option is only available in Stratix II, HardCopy II, and Arria GX devices.
Use 'rx_dpa_lock_reset' input port	Resets the DPA lock circuitry.
Use a DPA initial phase selection of	Turn on this option to select the initial phase setting. Specifies whether to turn on this option and its value. Simulation honors this phase selection in simulating the forwarded clock. This option is available for Arria II GX, Arria II GZ, Arria V, Arria V GZ, HardCopy III, HardCopy IV, Stratix III, Stratix IV, and Stratix V devices only.
Align DPA to rising edge of data only	Turn on this option to align the DPA to the rising edge of the data only or turn of this option to align the DPA to both the rising and falling edges of the data. This option is available for Arria II GX, Arria II GZ, HardCopy III, HardCopy IV, Stratix III, Stratix IV, and Stratix V devices only.
DPA Settings 3 (page 7) The options on this page are available when you turn on the DPA mode.	

Option	Description
Enable PLL Calibration	Turn on this option to phase-shift the PLL outputs when the `dpa_pll_cal_busy` signal is high. The default setting is **OFF**. This option is available for Arria II GZ, HardCopy III, HardCopy IV, Stratix III, and Stratix IV devices only. When you enable PLL calibration, you cannot merge the PLL with other PLLs.
Use 'dpa_pll_recal' input port	This port recalibrates the PLL without resetting the DPA. This option is available for Arria II GZ, HardCopy III, HardCopy IV, Stratix III, and Stratix IV devices only.
What is the input data rate?	Specifies the data rate for the input channel of the receiver, in Mbps. For data rate ranges, refer to the specific Device Data Sheet chapter in the respective device handbook. This value determines the legal input clock rate values.
Receiver Settings (page 8)	
Register outputs	Turn on this option to implement soft-CDR receiver modes in standard mode. In standard mode, the outputs of the receiver are registered by the `rx_outclock` signal. Turn off this option if you do not want to register the receiver outputs. In no output register mode, you must register the output registers in the design logic that is fed by the receiver, and then specify a **Source Multiply** assignment from the receiver to the output registers with a value equal to the deserialization factor.
Use 'rx_cda_reset' input port	The port resets the data alignment circuitry, restoring the latency bit counter to zero. This option is available only if you turn on the **Use 'rx_channel_data_align' input port** option. This option is available only if you use dedicated SERDES block.
Use 'rx_cda_max' output port	Indicates when the rollover point is reached in the data alignment circuit. This port is available only if you turn on the **Use 'rx_channel_data_align' input port** option. This option is available only if you use a a dedicated SERDES block.
After how many pulses does the data alignment circuitry restore the serial latency back to 0?	Specifies, in pulses, when the DPA circuitry restores the serial data latency to 0. The value does not have to be the same as the deserialization factor, but set the value to the deserialization factor to make the rollover occur for every deserialization factor. The available values for this option range from 1 to 11. This option is available only if you use a dedicated SERDES block.

Option	Description
Align data to the rising edge of clock	When you turn on this option, the data path is registered on the positive edge of the `diffioclk` signal (also referred to as the LVDS clock). When you turn off this option, the data path is registered on the negative edge of the `diffioclk` signal. This option is available only if you use a dedicated SERDES block, and is available only in non-DPA mode. This option changes the phase that captures the received data by 180°. Use caution when you turn off this option. The phase shift of the capture clock is automatically set according to the setting for the **What is the phase alignment of 'rx_in' with respect to the rising edge of 'rx_inclock'? (in degrees)** option. Changing the phase of the capture clock can lead to data corruption. If you turn off this option, the LVDS data is aligned to the falling edge of the clock. For an example, if you have two receivers interface with identical parameters except for the `rx_in` signal relationship to the `rx_inclock` signal, and you want to merge PLLs, one interface must have a 0° (rising edge) alignment, and the second interface must have a 180° (falling edge) alignment. You can only merge the PLLs when they have the same clock and phase settings; both must be set with the same alignment. You can set both receivers to be 0° aligned, and turn off **Align data to the rising edge of clock** on the 180° aligned interface.
Use 'rx_coreclk' input port	This option is enabled when the LVDS is implemented in logic. When you turn on this option, it adds an input port, which when asserted performs an asynchronous reset of all the logic in the ALTLVDS_RX IP core excluding the PLL.
Use 'rx_channel_data_align' input port	Turn on this option to control bit insertion on a channel-by-channel basis to align the word boundaries of the incoming data. The data slips one bit for every pulse on the `rx_channel_data_align` port. This option is available only if you use a dedicated SERDES block. You can use control characters in the data stream so your logic can have a known pattern to search for. You can compare the data received for each channel, compare to the control character you are looking for, then pulse the `rx_channel_data_align` port as required until you successfully receive the control character. To use this port, you must meet the following requirements: • The minimum pulse width is one period of the parallel clock in the logic array (`rx_outclock`). • The minimum low time between pulses is one period of the parallel clock. • There is no maximum high or low time. • Valid data is available on the third parallel clock cycle after the rising edge of the `rx_channel_data_align` signal.

Option	Description
Enable independent bitslip controls for each channel	Turn on this option to allow an independent `rx_data_align` signal for each channel that independently control the bit slip capability of each channel. This option is available if you implement the SERDES in LEs.
Add extra register for 'rx_data_align' input port	Turn on this option to enable the synchronization register of the receiver. If you turn on this option, you can also add an extra register to register the `rx_data_align` port using the `rx_outclock` port. This option is available if you implement the SERDES in LEs.
Use 'rx_data_align_reset' input port	Turn on this option to create the reset port for the bit-slip circuitry. This option is available if you implement the SERDES in LEs.
Which output synchronization buffer implementation should be used?	Specifies where to implement the buffer. The values are **Use RAM Buffer**, **Use Multiplexer and synchronization register**, and **Use logic element based RAM buffer**. A value of **Use Multiplexer and synchronization register** implements a multiplexer instead of a buffer. A value of **Use RAM Buffer** implements a buffer in RAM blocks. A value of **Use logic element based RAM buffer** implements a buffer in logic elements. The **Use RAM Buffer** and **Use logic element based RAM buffer** values use more logic, but result in the correct word alignment. If omitted, the default value is **Use RAM Buffer**.
Simulation Model (page 9)	
Simulation Libraries	Specifies the libraries needed for functional simulation by third-party tools.
Generate netlist	Turn on this option to generate synthesis area and timing estimation netlist.
Summary (page 10)	
Summary	Specifies the types of files to be generated. A gray checkmark indicates a file that is automatically generated; a green checkmark indicates an optional file. Choose from the following types of files: • AHDL Include file (<*function name*>**.inc**) • VHDL component declaration file (<*function name*>**.cmp**) • Quartus II symbol file (<*function name*>**.bsf**) • Instantiation template file (<*function name*>**_inst.v** or <*function name*>**_inst.vhd** • Verilog HDL block box file (<*function name*>**_bb.v**) • Pin Planner File (<*function name*>**_.ppf**) If you turn on the **Generate netlist** option, the file for that netlist is also available (<*function name*>**_syn.v**).

Related Information

- **Introduction to Altera IP Cores**
- **Stratix IV Device Family Errata Sheet**

Command Line Interface Parameters

Expert users can choose to instantiate and parameterize the IP core through the command-line interface using the clear box generator command. This method requires you to have command-line scripting knowledge.

The following table lists the parameters for the ALTLVDS_TX IP core.

Table 7: ALTLVDS_TX Parameters

Parameter	Type	Description
`common_rx_tx_pll`	String	Specifies whether the compiler uses the same PLL for both the LVDS receiver and the LVDS transmitter, or multiple LVDS receivers, or multiple LVDS transmitters, or both. You can use common PLLs if the same input clock source, same deserialization factor, same `pll_areset` source, and same data rates are used. The values are `ON` and `OFF`. If omitted, the default value is `ON`. Only available for Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone, Cyclone II, Cyclone III, Cyclone IV, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V devices.
`coreclock_divide_by`	Integer	Specifies the core clock output frequency to either be core clock or core clock divided by 2. The value are `1` or `2`. This parameter is only available when using odd SERDES factors. When using a divide-by factor of 1, fewer device resources are used, but you may not be able to achieve timing at higher data rates. Altera recommends using a divide-by factor of two for higher data rates. This parameter is available for the Cyclone series.

Parameter	Type	Description
deserialization_factor	Integer	Specifies the number of bits per channel. The following is the device support and its values with normal mode: • Arria II GX, Arria II GZ, Arria V, Arria V GZ: 1 to 10 • Arria GX: 1, 2, 4, to 10 • Cyclone, Cyclone II, Cyclone III, Cyclone IV, Cyclone V: 1, 2, 4, to 10 • HardCopy II, HardCopy III, and HardCopy IV: 1, 2, 4, to 10 • Stratix and Stratix GX: 1, 2, 4, 7, 8, to 10 • Stratix II and Stratix II GX: 1, 2, 4, to 10 • Stratix III, Stratix IV, and Stratix V: 1 to 10 Arria GX, Arria II GX, Arria II GZ, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, and Stratix IV devices have the values of 1, 2, 4, to 10 with SERDES using logic cells.
enable_clk_latency	String	Specifies whether the PLLs use clock latency. The values are ON and OFF.
implement_in_les	String	Specifies whether to implement SERDES circuitry in logic cells, which allows the circuitry to behave similarly to Stratix LVDS circuitry. You must use the implement_in_les parameter for SERDES functions that require data rates that are lower than the dedicated circuitry. The values are ON and OFF. For Cyclone, Cyclone II, Cyclone III, and Cyclone IV devices, the value is always ON. Available for all devices except the MAX series. The ALTLVDS_TX IP core starts its operation at the first rising edge of the fast clock, after the PLL has locked. This is intended for slow speeds and the bit alignment might be different from a dedicated SERDES implementation.

Parameter	Type	Description
inclock_data_alignment	String	Specifies the phase alignment of the tx_in[] and tx_inclock input ports in terms of the tx_inclock frequency. The clock phase alignment for the inclock_data_alignment parameter specifies the positive phase shift needed for the clock for alignment with the data. The following are the parameter values and its values in degrees (°): • EDGE_ALIGNED: 0° • 45_DEGREES: 45° • 90_DEGREES: 90° • 135_DEGREES: 135° • CENTER_ALIGNED: 180° • 225_DEGREES: 225° • 270_DEGREES: 270° • 315_DEGREES: 315° If omitted, the default value is EDGE_ALIGNED. Available for Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone, Cyclone II, Cyclone III, Cyclone IV, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V devices.
inclock_period	Integer	Specifies the input clock either by frequency (MHz in the parameter editor) or period (ps in HDL code). This parameter is required when the external PLL option is not used.
number_of_channels	Integer	Specifies the number of LVDS channels.

Parameter	Type	Description
outclock_alignment	String	Specifies the alignment of tx_outclock with respect to the VCO of a fast PLL. The clock phase alignment for the outclock_alignment parameter is data leading. This parameter is only used by the RTL simulation model and has no affect on how the Fitter sets the PLL parameters. Values are: • EDGE_ALIGNED: 0° • 45_DEGREES: 45° • 90_DEGREES: 90° • 135_DEGREES: 135° • CENTER_ALIGNED:180° • 225_DEGREES: 225° • 270_DEGREES: 270° • 315_DEGREES: 315° If omitted, the default value is EDGE_ALIGNED. Available for all devices excluding the MAX series.
outclock_divide_by	Integer	Specifies the period of the tx_outclock port as [INCLOCK_PERIOD * OUTCLOCK_DIVIDE_BY] and the frequency of the tx_outclock port as [INCLOCK_PERIOD/OUTCLOCK_DIVIDE_BY]. The default value for this parameter is the value of the deserialization_factor parameter. Only available for Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone, Cyclone II, Cyclone III, Cyclone IV, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V devices. For more information about the DESERIALIZATION_FACTOR and outclock_divide_by values, refer to Table 8.

Parameter	Type	Description
outclock_duty_cycle	Integer	Specifies the external clock timing constraints. A value of 50 is not supported in the outclock_duty_cycle parameter when the following is true: • DESERIALIZATION_FACTOR value is 5, 7, or 9. • OUTCLOCK_DIVIDE_BY value is equal to the value of DESERIALIZATION_FACTOR. • OUTCLOCK_MULTIPLY_BY value is 2. This is always true for Cyclone II, Cyclone III, Cyclone IV devices, and true for Arria V, Arria V GZ, Stratix II, Stratix III, Stratix IV, and Stratix V devices when the implement_in_les parameter value is set to ON.
outclock_multiply_by	Integer	Specifies the multiplication factor. The values are 1 and 2. If omitted, the default value is 1. Only available for Cyclone, Cyclone II, Stratix, Stratix GX, and Stratix II devices.
outclock_phase_shift	Integer	This parameter is used to set the phase shift parameters used by the PLL. Specifies the phase shift of the output clock relative to the input clock. Phase shifts of 0.0, 0.25, 0.5, or 0.75 times the input period (0, 90, or 270°) are implemented precisely. The allowed range for the phase shift is between 0 ps and 1 input clock period. If the phase shift is outside this range, the compiler adjusts it to fall within this range. For other phase shifts, the compiler chooses the closest allowed value. If omitted, the default value is 0.
outclock_resource	String	Specifies the clock resource type to use with the tx_coreclock port. The values are AUTO, REGIONAL CLOCK, and GLOBAL CLOCK. If omitted, the default value is AUTO. Only available for Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone, Cyclone II, Cyclone III, Cyclone IV, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V devices.

Parameter	Type	Description
output_data_rate	Integer	Specifies the data rate out of the PLL. The multiplication value for the PLL is OUTPUT_DATA_RATE/INCLOCK_PERIOD. Only available for Arria GX, Arria II GX, Arria II GZ, Cyclone, Cyclone II, Cyclone III, Cyclone IV, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, and Stratix IV devices.
pll_bandwidth_type	String	Specifies the loop filter bandwidth control setting on the PLL. The values are LOW, MEDIUM, and HIGH. This parameter is only available for the Stratix II device.
pll_self_reset_on_loss_lock	String	The values are ON and OFF. If omitted, the default value is OFF. When this parameter is enabled, the PLL is reset when it loses lock. This parameter is valid for Arria V, Arria V GZ, Cyclone III, Cyclone IV, Stratix, Stratix II GX, Stratix III, and Stratix IV devices when the implement_in_les parameter is set is ON.
registered_input	String	Indicates whether the tx_in[] port is registered. The values are ON, OFF, TX_INCLOCK, and TX_CORECLOCK. If omitted, the default value is ON when using the tx_coreclock port to register the data in logic elements. The TX_INCLOCK and TX_CORECLOCK values are available for Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone, Cyclone II, Cyclone III, Cyclone IV, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V devices. If the registered_input parameter is set to OFF, you must pre-register the tx_in[] port in the logic feeding the transmitter.
use_external_pll	String	Specifies whether the ALTLVDS_TX IP core generates a PLL or connect to a user-specified PLL. Altera recommends instantiating the external PLL with the parameter editor. Only available for Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone, Cyclone II, Cyclone III, Cyclone IV, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX, Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V devices.

Parameter	Type	Description
use_no_phase_shift	String	When set to OFF, a phase shift of 90° is added to the clock to center the clock in the data. Use this parameter when the implement_in_les parameter value is set to ON for Cyclone II, Stratix II, Stratix III, and Stratix IV devices. The values are ON and OFF. If omitted, default value is ON. Altera recommends setting this parameter to OFF unless you have completed a phase adjustment.

The following table lists the DESERIALIZATION_FACTOR and outclock_divide_by values.

Table 8: DESERIALIZATION_FACTOR and OUTCLOCK_DIVIDE_BY Values

Devices	DESERIALIZATION_FACTOR Value	OUTCLOCK_DIVIDE_BY Value
Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, HardCopy II, HardCopy III, HardCopy IV, Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V	4	2
		4
	5	5
	6	2
		6
	7	7
	8	2
		4
		8
	9	9
	10	2
		10
Stratix and Stratix GX	4	2
		4
	7	7
	8	2
		4
		8
	10	2
		10

Devices	DESERIALIZATION_FACTOR Value	OUTCLOCK_DIVIDE_BY Value
Cyclone, Cyclone II, Cyclone III, Cyclone IV, and Cyclone V	4	2
		4
		8
	5	2
		5
		10
	6	2
		6
		12
	7	2
		7
		14
	8	2
		4
		8
		16
	9	2
		9
		18
	10	2
		4
		10
		20

The following table lists the parameters for the ALTLVDS_RX IP core.

Table 9: ALTLVDS_RX Parameters

Parameter	Type	Description
buffer_implementation	String	Specifies where to implement the buffer. The values are MUX, RAM, and LES. A value of MUX implements a multiplexer instead of buffer implementation. A value of RAM implements a buffer in RAM blocks. A value of LES implements a buffer in logic elements. The RAM and LES values use more logic, but result in the correct word alignment. If omitted, the default value is RAM. To use the buffer_implementation parameter, the implement_in_les parameter must be turned ON. You can also use the buffer_implementation parameter with deserialization factors of 5, 7, or 9 only.
common_rx_tx_pll	String	Specifies whether the compiler uses the same PLL for both the LVDS receiver and the LVDS transmitter, or multiple LVDS receivers or multiple LVDS transmitters, or both. You can use common PLLs if the same input clock source, same deserialization factor, same pll_areset source, and same data rates are used. Values are ON and OFF. If omitted, the default value is ON.
data_align_rollover	Integer	Specifies, in pulses, when the DPA circuitry restores the serial data latency to 0. You must enable the rx_dpa_locked port and the enable_dpa_mode parameter if this parameter is specified. The legal integer value ranges from 1 to 11. If omitted, the default value is 4.

Parameter	Type	Description
deserialization_factor	Integer	Specifies the number of bits per channel. The values of this parameter for each supported device in normal mode are as follows: • Arria II GX, Arria II GZ, Arria V, Arria V GZ: 1 to 10. • Arria GX: 1, 2, 4, to 10. • Cyclone series: 1, 2, 4, to 10. • HardCopy II, HardCopy III, and HardCopy IV: 1, 2, 4, to 10. • Stratix and Stratix GX: 1, 2, 4, 7, 8, to 10. • Stratix II and Stratix II GX: 1, 2, 4, to 10. • Stratix III, Stratix IV, and Stratix V: 1, 2, 3, 4, to 10. Arria GX, Arria II GX, Arria II GZ, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix II, Stratix II GX, Stratix III, and Stratix IV have the values of 1, 2, 4, to 10 with SERDES using logic cells. The values of this parameter for each supported device in DPA mode are as follows: • Arria II GX, Arria II GZ, Arria V, Arria V GZ: 1 to 10. • Arria GX: 1, 2, 4, to 10. • HardCopy II, HardCopy III, and HardCopy IV: 1, 2, 4, to 10. • Stratix GX: 8 and 10. • Stratix II and Stratix II GX: 1, 2, 4, to 10. • Stratix III, Stratix IV, and Stratix V: 1 to 10.
dpa_initial_phase_value	Integer	Specifies the initial phase value. The values are 0 through 7. If the parameter value is set to OFF, the dpa_initial_phase_value parameter is set to 0.
enable_dpa_calibration	String	The values are ON and OFF. The default value is ON. Set this parameter to ON to phase shift the PLL outputs when the dpa_pll_cal_busy signal is high.
enable_dpa_align_to_rising_edge_only	String	Specifies that the DPA aligns to the rising edge of data only. Values are ON and OFF. If omitted, the default value is OFF. A value of OFF specifies that the DPA aligns to both the rising and falling edge of data.

Parameter	Type	Description
enable_dpa_fifo	String	Indicates whether the DPA FIFO buffer is enabled for this channel.You must enable the rx_dpa_locked port and enable_dpa_mode parameter if this parameter is specified. The values are ON and OFF. If omitted, the default value is ON. This parameter is available for Stratix GX devices in DPA mode only.
enable_dpa_initial_phase_selection	String	Specifies whether the dpa_initial_phase_value parameter is enabled. The values are ON and OFF. If omitted, the default value is OFF. When set to OFF, the dpa_initial_phase_value parameter value is set to 0.
enable_dpa_mode	String	Turns on DPA mode. The values are ON and OFF. If omitted, the default value is OFF.
enable_dpa_pll_calibration	String	The values are ON and OFF. The default value is OFF. Set this parameter to ON or OFF if you are instantiating the ALTLVDS_RX IP core in DPA mode with PLL calibration.
enable_soft_cdr_mode	String	Specifies whether the rx_divfwdclk port is used. When set to ON, the rx_divfwdclk port is driven by the DPA clock, and then it is divided down by the deserialization factor. When set to ON, the DPA FIFO is bypassed and rx_fifo_reset and reset_fifo_on_first_lock are ignored. The values are ON and OFF. If omitted, the default is OFF.
implement_in_les	String	Specifies whether to implement SERDES circuitry in logic cells, which allows the circuitry to behave similar to Stratix LVDS circuitry. Use the implement_in_les parameter for SERDES functions that require data rates that are lower than the dedicated circuitry. Values are ON and OFF. Note that the receiver IP core starts capturing the LVDS stream at the first rising edge of the fast clock, after the PLL has locked. This is intended for slow speeds and the bit alignment may be different from a hard SERDES implementation.

Parameter	Type	Description
inclock_data_alignment	String	Specifies the phase alignment of the rx_in and rx_inclock input ports in terms of the rx_inclock frequency. The clock phase alignment for the inclock_data_alignment parameter specifies the positive phase shift needed for the clock for alignment with the data. This parameter is only used by the RTL simulation model and has no affect on how the Fitter sets the PLL parameters. The following are the parameter values and the corresponding phase shifts in degrees (°): • EDGE_ALIGNED: 0° • 45_DEGREES: 45° • 90_DEGREES: 90° • 135_DEGREES: 135° • CENTER_ALIGNED: 180° • 225_DEGREES: 225° • 270_DEGREES: 270° • 315_DEGREES: 315° If omitted, the default value is EDGE_ALIGNED.
inclock_period	Integer	Specifies the period or frequency of the rx_inclock port. The default time unit is an integer in picoseconds (ps). In AHDL designs only, strings, such as 50.5 MHz, are acceptable.
inclock_phase_shift	Integer	This parameter is used to set the phase shift parameters used by the PLL. Specifies a phase shift in 15° increments.
input_data_rate	Integer	Specifies the data rate into the PLL. The multiplication value for the PLL is INPUT_DATA_RATE/INCLOCK_PERIOD.
lose_lock_on_one_change	String	Specifies when the DPA circuitry should lose lock. You must enable the rx_dpa_locked port and the enable_dpa_mode parameter if this parameter is specified. Values are ON and OFF. If omitted, the default value is ON.
number_of_channels	Integer	Specifies the number of LVDS channels.
outclock_resource	String	Specifies the clock resource type to use with the rx_outclock port. The values are AUTO, Regional Clock, and Global Clock. If omitted, the default value is AUTO.
pll_operation_mode	String	Specifies the source synchronous mode for Cyclone II and Stratix II device LE PLLs. The values are NORMAL and SOURCE_SYNCHRONOUS. If omitted, the default value is NORMAL.

Parameter	Type	Description
pll_self_reset_on_loss_lock	String	The values are ON and OFF. If omitted, the default value is OFF. When this parameter is enabled, the PLL is reset when it loses lock. This parameter is valid for Cyclone III, Cyclone IV, Stratix III, and Stratix IV devices when the implement_in_les parameter is set to ON.
port_rx_channel_data_align	String	Edge-sensitive bit-slip control signal. Each rising edge on this signal causes the data re-alignment circuitry to shift the word boundary by one bit. The minimum pulse width requirement is one parallel clock cycle. There is no maximum pulse width requirement. Determines if the rx_channel_data_align port is used or unused. The values are PORT_USED, PORT_UNUSED, and PORT_CONNECTIVITY. When set to PORT_USED, the rx_channel_data_align port is used. When set to PORT_UNUSED, the rx_channel_data_align port is unused. When set to PORT_CONNECTIVITY, the Quartus II software checks the connectivity of the rx_channel_data_align port to determine port usage. If omitted, the default value is PORT_CONNECTIVITY.
port_rx_data_align	String	Determines if the rx_align_data_reg port is used or unused. The values are PORT_USED, PORT_UNUSED, and PORT_CONNECTIVITY. When set to PORT_USED, the rx_align_data_reg port is used. When set to PORT_UNUSED, the rx_align_data_reg port is unused. When set to PORT_CONNECTIVITY, the Quartus II software checks the connectivity of the rx_align_data_reg port to determine port usage. If omitted, the default value is PORT_CONNECTIVITY.
registered_data_align_input	String	Specifies whether the rx_align_data_reg port is registered. The values are ON and OFF. If omitted, the default is ON. Only available for Stratix and Stratix GX devices.
registered_output	String	Indicates whether the rx_out[] port should be registered. The values are ON and OFF. If omitted, the default is ON. If the registered_output parameter is set to OFF, you should pre-register the rx_out[] port in the logic feeding the receiver.

Parameter	Type	Description
reset_fifo_at_first_lock	String	Specifies when the bit-serial FIFO resets. Normally, the bit-serial FIFO is reset when the DPA circuitry is locked or reset through the rx_reset port. The rx_dpa_locked port and the enable_dpa_mode parameter must be enabled if this parameter is specified. The values are ON and OFF. If omitted, the default value is ON. Only available for Arria GX, Arria II GX, Arria II GZ, Stratix II and Stratix II GX devices.
rx_align_data_reg	String	Controls byte alignment circuitry. If omitted, the default value is RISING_EDGE. This port is available for Stratix III devices only.
use_coreclock_input	String	Indicates whether the rx_coreclk port or the clock from PLL is used as the non-peripheral clock. You must connect the rx_coreclk port if you turn on this parameter. The values are ON and OFF. If omitted, the default value is OFF. This parameter is only available for Stratix GX devices. This parameter is available in DPA mode only.
use_external_pll	String	Specifies whether the ALTVDS_RX IP core generates a PLL or connect to a user-specified PLL. Altera recommends instantiating the external PLL with the parameter editor. Only available for Arria GX, Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone, Cyclone II, Cyclone III, Cyclone IV, HardCopy II, HardCopy III, HardCopy IV, Stratix, Stratix GX. Stratix II, Stratix II GX, Stratix III, Stratix IV, and Stratix V devices. This option is not available when using deserialization factor of 1 and 2 in the Cyclone series.
use_no_phase_shift	String	The values are ON and OFF. If omitted, default value is ON. Altera recommends setting this parameter to OFF unless you have done a phase adjustment. When set to OFF, a phase shift of 90° is added to the clock to center the clock in the data. Use this parameter when the pll_operation_mode parameter value is set to SOURCE_SYNCHRONOUS for Cyclone II and Stratix II devices.

Related Information

- on page 67

Ports

This section describes the ports for the ALTLVDS_TX and ALTLVDS_RX IP cores.

ALTLVDS_TX Ports

The following table lists the input and output ports for the ALTLVDS_TX IP core.

n is the number of channels. *m* is the `deserialization_factor` × `number_of_channels`.

Note: If you use dedicated SERDES, regardless of device family, you do not need to make additional constraints on the data port.

Table 10: ALTLVDS_TX Input and Output Ports

For Stratix IV, Arria II, and Cyclone IV devices, use the ALTPLL IP core. For Stratix V, Arria V, and Cyclone V devices use the Altera PLL IP core.

Port Name	Direction	Width (Bit)	Description
`pll_areset`	Input	1	Asynchronously resets all counters to the initial values.
`sync_inclock`	Input	1	Optional clock for the input registers.
`tx_data_reset`	Input	*n*	Asynchronous reset for the shift registers, capture registers, and synchronization registers for all channels. This port is available only when `implement_in_les` parameter is set to `ON`. This port does not affect the data realignment block or the PLL.
`tx_enable`	Input	1	Enables external PLL usage. When the `tx_enable` port is specified, connect the port to the `enable0` or `enable1` port of a PLL IP core instance. However, the `enable0`, `enable1` ports and the **Set up PLL in LVDS mode** option are available for Stratix II devices only.
`tx_in[]`	Input	*m*	This is parallel data which needs to be serially transmitted by the IP core. Input data must be synchronous to the `tx_coreclock` signal. The data bus width per channel is the same as the serialization factor (SF)

Port Name	Direction	Width (Bit)	Description
tx_inclock	Input	1	Reference clock input for the transmitter PLL. The parameter editor automatically selects the appropriate PLL multiplication factor based on the data rate and reference clock frequency selection. When using Stratix II devices in external PLL mode, connect the tx_inclock port to the sclkout0 or sclkout1 port. When using Cyclone and Cyclone II devices in external PLL mode, connect the tx_inclock port to other clocks. Refer to the respective device handbook for supported input clock frequency ranges.
tx_pll_enable	Input	1	Enables control for the LVDS PLL.
tx_syncclock	Input	1	Slow clock input port. In the Quartus II software version 8.0 or later, the tx_syncclock port is necessary for even deserialization factors in external PLL mode.
tx_coreclock	Output	1	Output clock used to feed non-peripheral logic. FPGA fabric-transmitter interface clock. The parallel transmitter data generated in the FPGA fabric must be clocked with this clock.
tx_locked	Output	1	Provides the LVDS PLL status. This port stays high when the PLL is locked to the input reference clock, and stays low when the PLL fails to lock.
tx_out[]	Output	*n*	Serialized LVDS data output port of n channels wide. tx_out[(n-1)..0] drives parallel data from tx_in[(J * n)-1 ..0] where *J* is the serialization factor and *n* is the number of channels. tx_out[0] drives data from tx_in[(J-1)..0]. tx_out[1] drives data from the next *J* number of bits on tx_in.
tx_outclock	Output	1	External reference clock. The frequency of this clock is programmable to be the same as the data rate (up to 717 MHz), half the data rate, or one-fourth the data rate. The phase offset of this clock, with respect to the serial data, is programmable in increments of 45°.

Related Information

- **Introduction to Altera IP Cores**
- **PLL Clock Signals for LVDS Interface in External PLL Mode** on page 62

ALTLVDS_RX Ports

The following table lists the input and output ports for the ALTLVDS_RX IP core.

Note: *n* is the number of channels. *m* is the `deserialization_factor` × `number_of_channels`.

Table 11: ALTLVDS_RX Input and Output Ports

For Stratix IV, Arria II, and Cyclone IV devices, use the ALTPLL IP core. For Stratix V, Arria V, and Cyclone V devices use the Altera PLL IP core.

Port Name	Direction	Width (Bit)	Description
`dpa_pll_recal`	Input	1	Enables dynamic recalibration without resetting the DPA circuitry or the PLL. Only available in DPA mode when PLL calibration is enabled.
`pll_areset`	Input	1	Asynchronously resets all counters to initial values. The minimum pulse width requirement for this signal is 10 ns.
`pll_phasedone`	Input	1	Specifies whether dynamic phase reconfiguration is complete. Only available when using an external PLL when PLL calibration is enabled.
`rx_cda_reset`	Input	*n*	Asynchronous reset to the data realignment circuitry. The minimum pulse width requirement for this reset is one parallel clock cycle. This signal resets the data realignment block. This port is not available for Arria V and Cyclone V devices. You can reset the CDA or bitslip in Arria V and Cyclone V devices by asserting the `rx_channel_data_align` signal until the bitslip counter rolls over.
`rx_channel_data_align`	Input	*n*	Controls byte alignment circuitry.
`rx_coreclk`	Input	*n*	LVDS reference input clock. Replaces the non-peripheral clock from the PLL. One clock for each channel.
`rx_data_align`	Input	1	Controls byte alignment circuitry. You can register this port using the `rx_outclock` port. This port is available when `implement_in_les` parameter is set to `ON` and can be implemented using flexible LVDS.

Port Name	Direction	Width (Bit)	Description
rx_data_align_reset	Input	1	Resets the byte alignment circuitry. Use the rx_data_align_reset input port when you need to reset the PLL during device operation and when you need to re-establish the word alignment. This port is available when implement_in_les parameter is set to ON.
rx_data_reset	Input	*n*	Asynchronous reset for all channels, excluding the PLL.
rx_deskew	Input	1	Specifies whether to activate calibration mode.
rx_dpa_lock_reset	Input	n	Forces the rx_dpa_locked port to low and forces the lock counter to start counting again.
rx_dpll_enable	Input	*n*	Enables the data path that flows through the DPA circuit. This port is available only when DPA mode is enabled. This port is supported in Arria GX, HardCopy II, Stratix II, and Stratix II GX devices only.
rx_dpll_hold	Input	*n*	Prevents the DPA circuitry from switching to a new phase. When low, the DPA tracks any dynamic phase variations between the clock and data. When high, the DPA holds the last locked phase and does not track any dynamic phase variations between the clock and data. This port is not available in non-DPA mode.
rx_dpll_reset	Input	*n*	Asynchronous reset for all channels.
rx_enable	Input	1	Enables external PLL usage. When the rx_enable port is specified, it must connect to the enable0 or enable1 port of a PLL IP core instance configured in LVDS mode. However, the enable0, enable1 ports and the **Set up PLL in LVDS mode** option are available for Stratix II devices only.
rx_fifo_reset	Input	*n*	Asynchronous reset to the FIFO between the DPA and the data realignment circuits. The synchronizer block must be reset after a DPA loses lock condition and the data checker shows corrupted received data. The minimum pulse width requirement for this reset is one parallel clock cycle. This signal resets the FIFO block. Only available when DPA mode is enabled.

Port Name	Direction	Width (Bit)	Description
rx_in[]	Input	*n*	LVDS serial data input port of n channels wide. rx_in[(n-1)..0] is deserialized and driven on rx_out[(J * n)-1 ..0] where *J* is the deserialization factor and *n* is the number of channels. rx_in[0] drives data to rx_out[(J-1)..0]. rx_in[1] drives data to the next *J* number of bits on rx_out.
rx_inclock	Input	1	LVDS reference input clock. The parameter editor automatically selects the appropriate PLL multiplication factor based on the data rate and reference clock frequency selection. When using Stratix II devices in external PLL mode, connect the rx_inclock port to the sclkout0 or sclkout1 port. When using Cyclone and Cyclone II devices in external PLL mode, connect the rx_inclock port to other clocks. Refer to the respective device handbook for supported input clock frequency ranges.
rx_pll_enable	Input	1	Enables control for the LVDS PLL.
rx_readclock	Input	1	Clock input port for reading operation.
rx_reset	Input	*n*	Asynchronous reset to the DPA circuitry and FIFO. The minimum pulse width requirement for this reset is one parallel clock cycle. This signal resets DPA and FIFO blocks. You can connect this port if the enable_dpa_mode parameter is turned on.
rx_syncclock	Input	1	Slow clock input port.
dpa_pll_cal_busy	Output	1	Busy signal that is asserted high when PLL calibration occurs. PLL clock signals are phase adjusted for two fast clock cycles ahead. Available only when DPA mode with PLL calibration is enabled.
pll_phasecounterselect	Output	1	Specifies the PLL counter select. Available only when DPA mode with PLL calibration is enabled.
pll_phasestep	Output	1	Specifies dynamic phase shifting. Available only when DPA mode with PLL calibration is enabled.
pll_phaseupdown	Output	1	Specifies dynamic phase adjustment. Available only when DPA mode with PLL calibration is enabled.
pll_scanclk	Output	1	Clock signal for the serial scan chain. Available only when DPA mode with PLL calibration is enabled.

Port Name	Direction	Width (Bit)	Description
rx_cda_max	Output	*n*	Data re-alignment (bit slip) roll-over signal. When high for one parallel clock cycle, this signal indicates that the user-programmed number of bits for the word boundary to roll-over have been slipped. Indicates when the next rx_channel_data_align pulse restores the serial data latency back to 0.
rx_divfwdclk	Output	*n*	Parallel DPA clock to the FPGA fabric logic array. The parallel receiver output data to the FPGA fabric logic array is synchronous to this clock in soft-CDR mode. This signal is not available in non-DPA and DPA modes. Divides and forwards the clock to the source from the DPA block of the clock channel. When the enable_soft_cdr_mode parameter is set to ON, the rx_divfwdclk port is used. When set to ON, the rx_divfwdclk port clocks the synchronization registers.
rx_dpa_locked	Output	*n*	Indicates whether the channel is locked to DPA mode. This signal only indicates an initial DPA lock condition to the optimum phase after power up or reset. This signal is not deasserted if the DPA selects a new phase out of the eight clock phases to sample the received data. You must not use the rx_dpa_locked signal to determine a DPA loss-of-lock condition.
rx_locked	Output	1	Provides the LVDS PLL status. Stays high when the PLL is locked to rx_inclock, and stays low when the PLL fails to lock.
rx_out	Output	*m*	Receiver parallel data output. The data bus width per channel is the same as the deserialization factor (DF). The output data is synchronous to the rx_outclock signal in non-DPA and DPA modes. It is synchronous to the rx_divfwdclk signal in soft-CDR mode.
rx_outclock	Output	1	Parallel output clock from the receiver PLL. The parallel data output from the receiver is synchronous to this clock in non-DPA and DPA modes. This port is not available when you turn on the Use External PLL option in the parameter editor. The FPGA fabric-receiver interface clock must be driven by the PLL instantiated through the PLL IP core parameter editor.

Related Information

- **Introduction to Altera IP Cores**
- **PLL Clock Signals for LVDS Interface in External PLL Mode** on page 62

Prototypes and Component Declarations

This section describes the prototypes and component declarations of the ALTLVDS_TX and ALTLVDS_RX IP cores.

Verilog HDL Prototype

You can locate the Verilog HDL prototype in the Verilog Design File (**.v**) **altera_mf.v** in the <*Quartus II installation directory*>**\eda\synthesis** directory.

VHDL Component Declaration

You can locate VHDL component declaration in the VHDL Design File (**.vhd**) **altera_mf_components.vhd** in the <*Quartus II installation directory*>**\libraries\vhdl\altera_mf** directory.

VHDL LIBRARY-USE Declaration

The VHDL LIBRARY-USE declaration is not required if you use the VHDL Component Declaration.

```
LIBRARY altera_mf;
USE altera_mf.altera_mf_components.all;
```

Functional Description

This section describes the various receiver modes and features, the functionality of the ports and the timing analysis of the IP cores.

Receiver Modes

The physical medium connecting the transmitter and receiver LVDS channels may introduce a skew between the serial data and the source-synchronous clock. The instantaneous skew between each LVDS channel and the clock also varies with the jitter on the data and clock signals as seen by the receiver.

The three receiver modes provide different options to overcome skew between the source-synchronous clock (non-DPA, DPA) /reference clock (soft-CDR) and the serial data.

The ALTLVDS_RX IP core supports the following receiver modes:

- **DPA Mode**
- **Non-DPA Mode**
- **Soft-CDR Mode**

DPA Mode

In DPA mode, the DPA circuitry automatically chooses the best phase to compensate for the skew between the source-synchronous clock and the received serial data.

Non-DPA Mode

Non-DPA mode allows you to statically select the optimal phase between the source synchronous clock and the received serial data to compensate for the skew.

Soft-CDR Mode

The soft-CDR mode removes the clock from the clock-embedded data, a capability required for the serial gigabit media independent interface (SGMII) protocol. The PLL requires a reference clock, but the reference clock need not be source-synchronous with the data.

Clock Forwarding

In soft-CDR mode, the ALTLVDS_RX IP core divides the DPA clock and the data by the deserialization factor. The newly divided clock signal, `rx_divfwdclk`,is then placed on the PCLK network, which carries the clock signal to the core. In supported devices, each LVDS channel can be in soft-CDR mode and can drive the core using the PCLK network. The clock forwarding feature is supported in Arria II GX, Arria II GZ, Arria V, Arria V GZ, HardCopy III, HardCopy IV, Stratix III, Stratix IV, and Stratix V devices.

Note: For more information about periphery clock networks for specific devices, refer to the *Clock Networks and PLLs* chapter in volume 1 of the respective device handbook.

When using soft-CDR mode, the `rx_reset` port must not be asserted after the DPA training is asserted because the DPA continuously chooses new phase taps from the PLL to track parts per million (ppm) differences between the reference clock and incoming data. The parallel clock `rx_outclock`, generated by the left and right PLL, is also forwarded to the FPGA fabric.

Note:

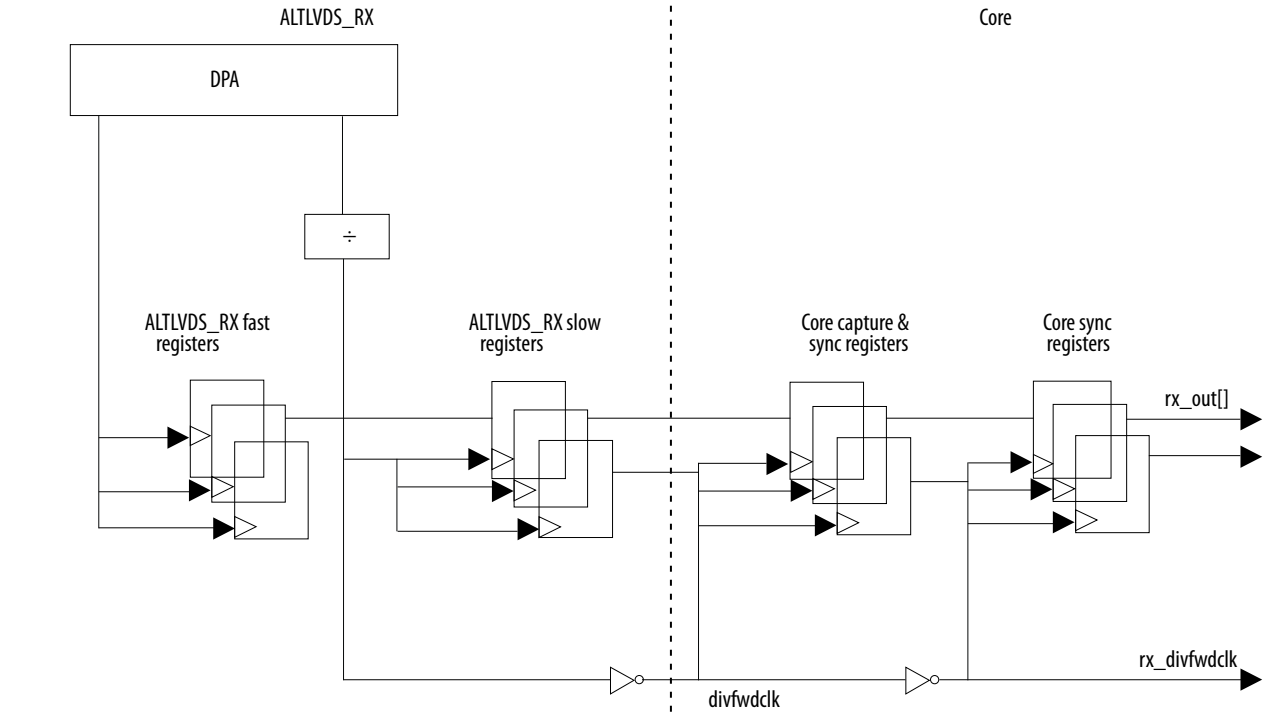
- For ppm tolerance specifications between the source clock and received data, refer to the appropriate device data sheet or device handbook for each device.
- For more information about receiver modes, refer to the *High-Speed Differential I/O Interfaces* chapter in the respective device handbook.

The **Standard Mode** on page 47 and **No Output Register Mode** on page 48 sections describe the implementation of soft -CDR mode in the ALTLVDS_RX block.

Standard Mode

The following figure shows the implementation of soft-CDR mode in standard mode. In standard mode, the first two stages of core-capture registers are created automatically by the ALTLVDS_RX parameter editor. You must clock any additional user registers from the positive edge of the `rx_divfwdclk` clock; using the negative edge makes it harder to meet timing, and the duty cycle is not guaranteed.

Figure 7: ALTLVDS_RX Block in Standard Mode

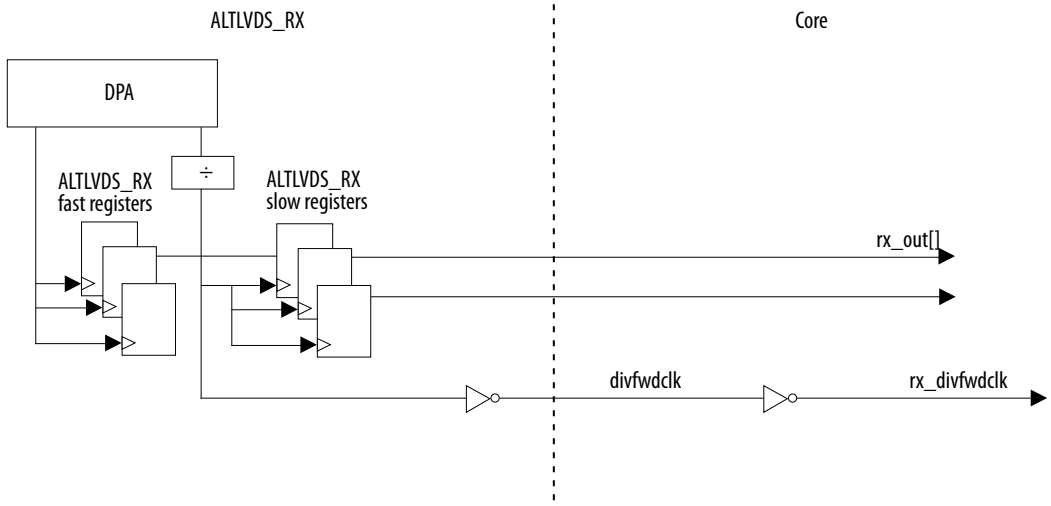


Note: For LVDS RX channel operating in soft-CDR mode, Altera recommends you to use `rx_divfwdclk` (instead of any static clock) as the SignalTap capturing clock. Using static clock as the SignalTap capturing clock leads to bit error during the SignalTap sampling.

No Output Register Mode

The following figure shows the implementation of soft-CDR mode in no-output register mode. In this mode, you must create the capture registers by the user logic. To ensure even slack for both setup and hold, you must clock the first capture register stage by the falling edge of the `rx_divfwdclk` clock and clock the second stage of the registers by the rising edge of the rx_divwdclk clock. The register clocking method gives the equivalent implementation as the standard mode implementation.

Figure 8: ALTLVDS_RX Block in No Output Register Mode



DPA PLL Calibration

The following sections describe DPA PLL calibration and its effects in Stratix III, Stratix IV, Stratix IV Engineering Sample (ES), and Arria II devices.

- **DPA PLL Calibration in Stratix IV ES Devices** on page 49
- **DPA PLL Calibration in Arria II and Stratix IV Devices and Later** on page 50
- **Effects of DPA PLL Calibration** on page 51

Related Information

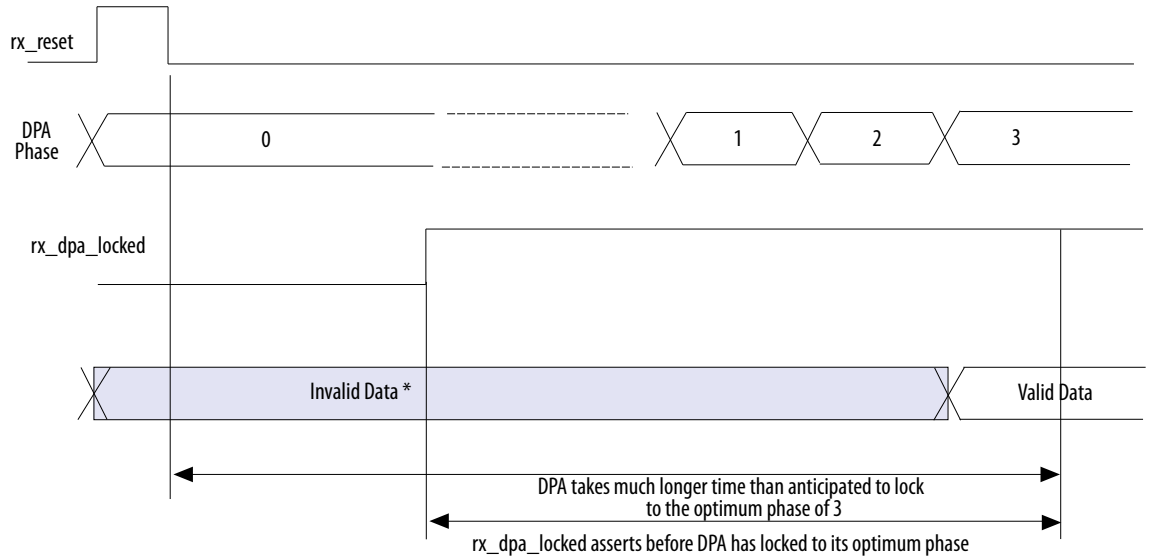
High-Speed Differential I/O Interfaces and DPA in Arria V Devices, Volume 1: Device Interfaces and Intergration, Arria V Device Handbook

DPA PLL Calibration in Stratix IV ES Devices

Applications using a fixed, cyclical training pattern with sparse data transitions can cause the PLL phase to remain unchanged, which results in DPA misalignment. When DPA misaligns the DPA circuitry remains at the initial configured phase or takes a significantly longer time to lock onto the optimum phase. A non-ideal phase might result in data bit errors, even after the DPA lock signal goes high. Resetting the DPA circuit may not solve the problem.

The following figure shows that the DPA takes longer time to lock onto the optimum phase even after the `rx_reset` and `rx_dpa_locked` signals are asserted, resulting in data errors.

Figure 9: DPA Misalignment Issue



In the Quartus II software versions 9.0 and later, the DPA PLL calibration feature is added to the ALTLVDS_RX IP core to overcome the DPA misalignment issue found in Stratix IV ES devices; the Stratix IV production devices are not affected. The DPA PLL calibration feature is available when the LVDS receiver is configured in DPA or soft-CDR mode. DPA PLL calibration phase-shifts the PLL outputs to induce progress in the PLL's phase-detect up and down counter and to facilitate a new phase selection.

The following events occur during the DPA PLL calibration process:

1. The ALTLVDS_RX IP core counts 256 data transitions; the PLL calibrates the phase forward by two clocks.
2. The ALTLVDS_RX IP core counts 256 transitions; the PLL calibrates the phase backward by two clocks so that the PLL timing returns to normal.
3. The ALTLVDS_RX IP core counts 256 data transitions, and then asserts the `rx_dpa_locked` signal.

Note: For more information about DPA lock time specification, refer to the Device Data Sheet chapter in the respective device handbook.

Related Information

Stratix IV DPA Misalignment

DPA PLL Calibration in Arria II and Stratix IV Devices and Later

Starting with the Arria II device and the production versions of Stratix IV devices, DPA PLL calibration is implemented for each receiver channel independently using delay elements in the LVDS receiver path. Anytime the `rx_reset` port is deasserted for a receiver channel, the DPA circuitry is reset, and the calibration and locking process begins. The DPA circuitry in an LVDS receiver can reset at any time without impacting other LVDS receivers sharing the same PLL.

1. The following events occur during the DPA calibration process:
2. The ALTLVDS_RX IP core counts 256 data transitions, then inserts delay elements on the LVDS receiver data path to skew the clock and data relationship.
3. The ALTLVDS_RX IP core counts 256 data transitions, then removes the delay elements on the LVDS receiver data path, restoring the original clock to data relationship.
4. The ALTLVDS_RX IP core counts 256 data transitions, and then asserts the `rx_dpa_locked` signal.

With the Stratix IV production devices, you can choose to use the DPA PLL calibration method to be backward compatible with Stratix III and Stratix IV ES devices by turning on **Enable PLL calibration** in the ALTLVDS_RX parameter editor. If you turn off **Enable PLL calibration** in the ALTLVDS_RX parameter editor, the receiver IP core uses delay elements in the receiver data path.

Arria II devices always use the DPA calibration method using delay elements in the receiver data path.

Effects of DPA PLL Calibration

There are two notable effects when DPA PLL calibration is enabled: effect on the timing of the logic clocked by the PLL, and effect related to the merging PLLs.

During PLL phase calibration, the I/O timing is pulled in by quarter of the voltage-controlled oscillator (VCO) period. All outputs of the PLL, including the slow clock, are affected. All HSIO TX data from interfaces, clocked by the affected PLL, clocks out quarter of the VCO period earlier. Likewise, all HSIO RX data clocks quarter cycle out of phase with the VCO but has less time to be sampled. For the slow clock that drives the core and the system, there is a loss of quarter of the VCO period on internal timing, across clock domain transfers in the core. The quarter period-pull greatly affects a design that has cross-clock transfer without using a FIFO, and the two clocks are not from the same PLL.

If DPA PLL calibration is enabled, PLLs, between receiver and transmitter instances or multiple receiver instances, do not merge even if the **Share PLLs for receivers and transmitters** setting is enabled. To force merging of such PLLs, use `FORCE_MERGE_PLLS=ON` setting in the Quartus II Settings File (**.qsf**).

Related Information

Quartus II Settings File Manual

Initialization and Reset

This section describes the initialization and reset aspects, using control characters. This section also provides a recommended initialization and reset flow for the ALTLVDS_TX and ALTLVDS_RX IP cores.

Initializing ALTLVDS_TX and ALTLVDS_RX

With the ALTLVDS_TX and ALTLVDS_RX IP cores, the PLL is locked to the reference clock prior to implementing the SERDES blocks for data transfer. The PLL starts to lock to the reference clock during device initialization. The PLL is operational when the PLL achieves lock during user mode. If the clock reference is not stable during device initialization, the PLL output clock phase shifts becomes corrupted.

When the PLL output clock phase shifts are not set correctly, the data transfer between the high-speed LVDS domain and the low-speed parallel domain might not be successful, which leads to data corruption. Assert the `pll_areset` port for at least 10 ns, and then deassert the `pll_areset` port and wait until the PLL lock becomes stable. After the PLL lock port asserts and is stable, the SERDES blocks are ready for operation.

When using DPA, further steps are required for initialization and reset recovery. The DPA circuit samples the incoming data and finds the optimal phase tap from the PLL to capture the data on a receiver channel-

by-channel basis. If the PLL has not locked to a stable clock source, the DPA circuit might lock prematurely to a non-ideal phase tap. Use the `rx_reset` port to keep the DPA in reset until the PLL lock signal is asserted and stable.

In Stratix GX, Stratix II, Stratix II GX, HardCopy II, and Arria GX devices, when using the `rx_reset` port, the ALTLVDS_RX parameter editor allows you to choose whether or not to automatically reset the bit serial FIFO when the `rx_dpa_locked` signal asserts for the first time. This is a useful feature because it keeps the synchronizer FIFO in reset until the DPA locks. To provide optimal timing between the DPA domain, it is important to keep the FIFO in reset until the DPA locks.

With Stratix III, HardCopy III, Arria II GX, Arria II GZ devices and later generations of these devices, the `rx_dpa_lock` signal asserts only after a specific number of transitions are detected in the parallel data stream. You must not assert `rx_fifo_reset` port until the `rx_dpa_lock` signal asserts, otherwise, there will be no data transitions in the parallel data, and the `rx_dpa_lock` signal will never assert.

Note: Altera recommends asserting the `rx_fifo_reset` port after the `rx_dpa_locked` signal asserts, and then deassert the `rx_fifo_reset` port to begin receiving data.

Each time the DPA shifts the phase taps during normal operation to track variations between the relationship of the reference clock source and the data, the timing margin for the data transfer between clock domains is reduced.

For Stratix GX, Stratix II, Stratix II GX, HardCopy II, and Arria GX devices, when the ALTLVDS_RX IP core deasserts the `rx_dpa_locked` port to indicate that the DPA has selected a new phase tap to capture the data. You can choose the options in the ALTLVDS_RX parameter editor if you want the DPA lock signal to deassert after one phase step, or after two phase steps in the same direction (check device family availability for this option).

With Stratix III, HardCopy III, Arria II GX, Arria II GZ devices and later generations of these devices, the ALTLVDS_RX asserts the `rx_dpa_locked` port upon initial DPA lock. This port remains asserted throughout the operation until the ALTLVDS_RX IP core asserts the `rx_reset` or `rx_dpa_lock_reset` ports. The `rx_dpa_locked` port does not indicate if the DPA has selected a new phase.

Note: Altera recommends using the data checkers to ensure data accuracy.

Resetting the DPA

When the data becomes corrupted, you must reset the DPA circuitry using the `rx_reset` port and `rx_fifo_reset` port.

Assert the `rx_reset` port to reset the entire DPA block. This requires the DPA to be trained before it is ready for data capture.

Note: Altera recommends using the option to automatically reset the bit serial FIFO when the `rx_dpa_locked` signal rises for the first time, if available for your device family; otherwise, toggle the `rx_fifo_reset` port after `rx_dpa_locked` is asserted. This option ensures the synchronization FIFO is set with the optimal timing to transfer data between the DPA and high-speed LVDS clock domains.

Assert the `rx_fifo_reset` port to reset only the synchronization FIFO. This allows you to continue system operation without having to re-train the DPA. Using this port can fix data corruption because it resets the FIFO; however, it does not reset the DPA circuit. In Stratix GX, Stratix II, Stratix II GX, HardCopy II, and Arria GX devices, the `rx_dpa_locked` port remains in its previous state; if it was deasserted, it remains deasserted and you are not be able to use it to know when the DPA is using the ideal phase tap for data capture.

When the DPA is locked, the ALTLVDS block is ready to capture data. The DPA finds the optimal sample location to capture each bit. The next step is to set up the word boundary using custom logic to control the `rx_channel_data_align` port on a channel-by-channel basis.

The word aligner or the bit-slip circuit can be reset using the `rx_cda_reset` port. This circuit can be reset anytime and is not dependent on the PLL or DPA circuit operation.

Aligning the Word Boundaries

To align the word boundaries, it is useful to have control characters in the data stream so that your logic can have a known pattern to search for. You can compare the data received for each channel, compare to the control character you are looking for, then pulse the `rx_channel_data_align` port as required until you successfully receive the control character.

Note: Altera recommends setting the `rx_cda_max[]` port to the deserialization factor or higher, which allows enough depth in the bit slip circuit to roll through an entire word if required.

If you do not have control characters in the received data, you need a deterministic relationship between the reference clock and data to predict the word boundary using timing simulation or laboratory measurements. The only way to ensure a deterministic relationship on the default word position in the SERDES when the device powers up, or anytime the PLL is reset, is to have a reference clock equal to the data rate divided by the deserialization factor. For example, if the data rate is 800 Mbps, and the deserialization factor is 8, the PLL requires a 100-MHz reference clock. This is important because the PLL locks to the rising edge of the reference clock. If you have one rising edge on the reference clock per serial word received, the deserializer always starts at the same position. Using timing simulation, or lab measurements, monitor the parallel words received and determine how many pulses are required on the `rx_channel_data_align` port to set your word boundaries. You can create a simple state machine to apply the required number of pulses when you enter user mode, or anytime you reset the PLL and DPA blocks.

Recommended Initialization and Reset Flow

Altera recommends that you follow these steps to initialize and reset the ALTLVDS IP cores:

1. During entry into user mode, or anytime in user mode operation when the interface requires a reset, assert the `pll_areset` and `rx_reset` ports.
2. Deassert the `pll_areset` port and monitor the `rx_locked` port (`rx_locked` is the PLL lock indicator).
3. Deassert the `rx_reset` port after the `rx_locked` port becomes asserted and stable.
4. Apply the DPA training pattern and allow the DPA circuit to lock. (If a training pattern is not available, any data with transitions is required to allow the DPA to lock.) Refer to the respective device data sheet for DPA lock time specifications.
5. Wait for the `rx_dpa_locked` port to assert.
6. Beginning with Stratix III, HardCopy III, Arria II GX, and Arria II GZ devices, assert `rx_fifo_reset` for at least one parallel clock cycle, and then de-assert `rx_fifo_reset`.
7. Assert the `rx_cda_reset` port for at least one parallel clock cycle, and then deassert the `rx_cda_reset` port.
8. Begin word alignment by applying pulses as required to the `rx_channel_data_align` port.
9. When the word boundaries are established on each channel, the interface is ready for operation.

Source-Synchronous Timing Analysis and Timing Constraints

This section defines the source-synchronous differential data orientation timing parameters, the timing budget definitions, and how to use these timing parameters to determine a design's maximum performance.

Different modes of LVDS receivers use different specifications in deciding the ability to sample the received serial data correctly.

Dedicated SERDES

The ALTLVDS_TX and ALTLVDS_RX IP cores implemented in a dedicated SERDES and using the DPA mode are characterized and guaranteed to function correctly within the LVDS system. Refer to the respective device handbook for details about whether dedicated SERDES and DPA are supported for the device family. The Quartus II compiler automatically ensures the associated delay chain settings are set correctly for the data path at the LVDS transmitter/receiver that uses the source-synchronous compensation mode of PLL operation.

You can optionally add false path constraints to the asynchronous input and output ports to avoid unconstrained path warnings. For non-DPA mode, you can optionally constrain the synchronous input ports to improve the accuracy of the receiver skew margin analysis.

Note: The TimeQuest Timing Analyzer automatically adds the required multicycle path, false path, and clock uncertainty constraints to analyze timing for the dedicated SERDES if you add `derive_pll_clocks` to your Synopsys Design Constraints (**.sdc**) file.

SERDES in LEs

For receiver designs that are using the SERDES in LEs, you must ensure proper timing constraints for the TimeQuest timing analyzer tool in the Quartus II software to indicate whether the SERDES captures the data as expected or otherwise.

For dedicated SERDES and SERDES in LEs, you can set the timing constraints using the following methods:

- Setting timing constraints using the TimeQuest Timing Analyzer GUI
- Setting timing constraints manually in the **.sdc**.

Receiver Skew Margin and Transmitter Channel-to-Channel Skew

Changes in system environment, such as temperature, media (cable, connector, or PCB), and loading, affect the receiver's setup and hold times; internal skew affects the sampling ability of the receiver.

In non-DPA mode, use receiver skew margin (RKSM), receiver channel-to-channel skew (RCCS), and sampling window (SW) specifications to analyze the timing for high-speed source-synchronous differential signals in the receiver data path. The following equation shows the relationship between RSKM, RCCS, and SW.

Figure 10: RSKM

$$RSKM = \frac{TUI - SW - RCCS}{2}$$

Where:

- RSKM—is the timing margin between the receiver's clock input and the data input SW.
- Time unit interval (TUI)—is the time period of the serial data ($1/f_{MAX}$). Also known as the LVDS period in the TimeQuest Timing Analyzer section in the Quartus II Compilation Report.
- SW—is the period of time that the input data must be stable to ensure that data is successfully sampled by the LVDS receiver. The SW is a device property and varies with device speed grade.
- RCCS— is the timing difference between the fastest and slowest input transitions, including t_{CO} variations and clock skew. Specify RCCS by applying minimum and maximum `set_input_delay` constraints to the receiver inputs, where RCCS is the difference between the maximum and minimum value.

To obtain accurate RSKM results in the TimeQuest analyzer, specify your RCCS figure using `set_input_delay` constraints.

The difference between your `set_input_delay -min` and `set_input_delay -max` must match your RCCS figure.

For example, to specify an RCCS figure of 0.3 ns:

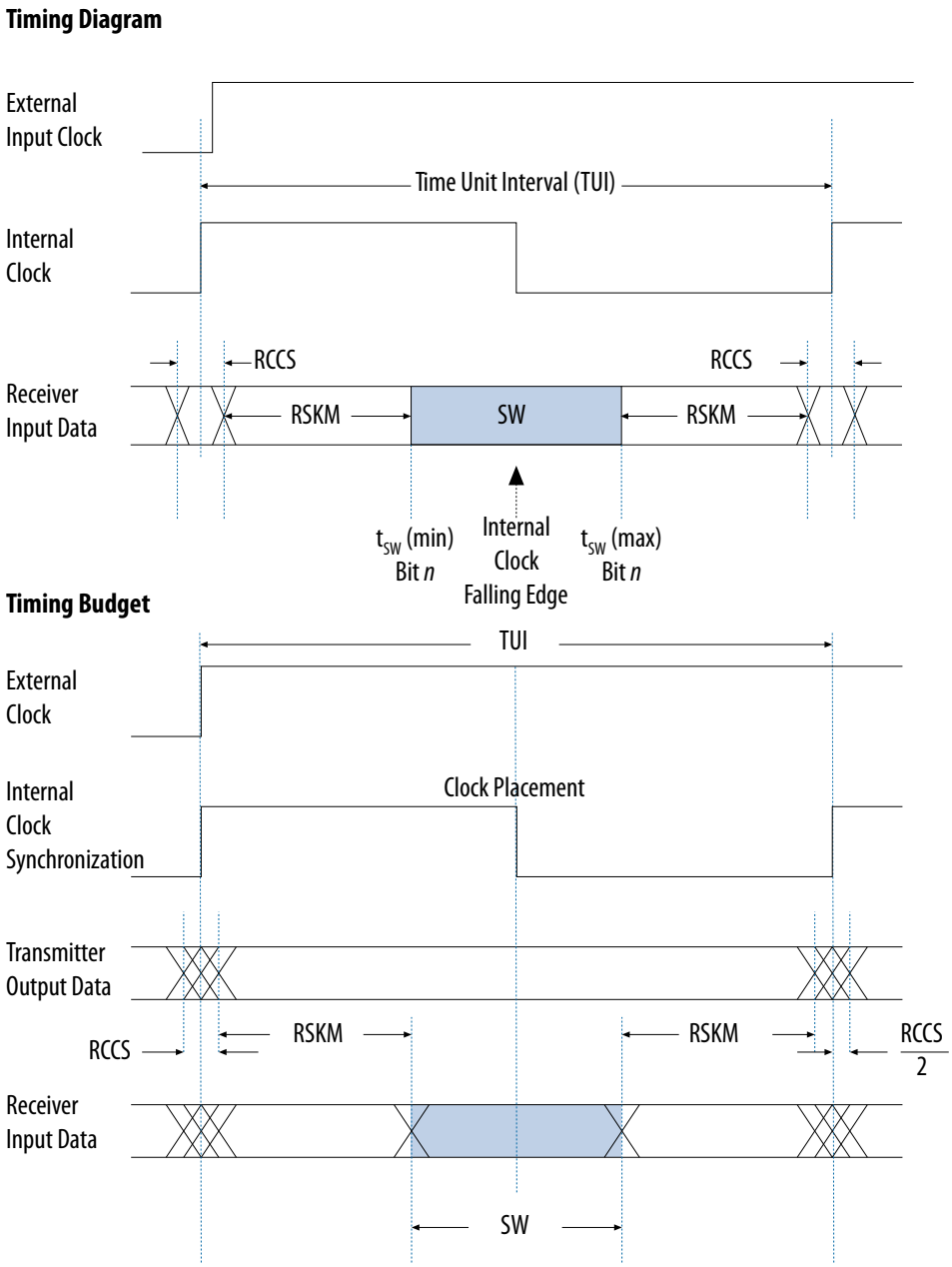
```
set_input_delay -clock rx_inclock -min 0 [get_ports {rx_in*}]

set_input_delay -clock rx_inclock -max 0.3 [get_ports {rx_in*}]
```

The TimeQuest analyzer takes the 0.3 ns RCCS figure into account during RSKM analysis.

The following figure shows the relationship between the RSKM, RCCS, and SW.

Figure 11: Differential High-Speed Timing Diagram and Timing Budget for Non-DPA Mode



You must calculate the RSKM value to decide whether you can properly sample the data by the LVDS receiver with the given data rate and device. A positive RSKM value indicates the LVDS receiver can properly sample the data; a negative RSKM value indicates the receiver cannot properly sample the data.

The following example shows the RSKM calculation.

Data Rate: 1 Gbps, Board channel-to-channel skew = **200 ps**

For Stratix IV devices:

RCCS = **100 ps** (pending characterization)

SW = **300 ps** (pending characterization)

TUI = **1000 ps**

Total RCCS = RCCS + Board channel-to-channel skew= 100 ps + 200 ps

= **300 ps**

RSKM= **TUI** - **SW** - **RCCS**

= **1000 ps** - **300 ps** - **300 ps**

= **400 ps** > **0**

Because the RSKM > 0 ps, receiver non-DPA mode must work correctly.

Obtaining the RSKM Report

For LVDS receivers, the Quartus II software provides the RSKM report showing SW, TUI or LVDS period, and RSKM values for non-DPA mode. You can generate the RSKM report by executing the report_rskm command in the TimeQuest Timing Analyzer.

To obtain the RSKM report, follow these steps:

1. In the Quartus II software, under the Tools menu, click **TimeQuest Timing Analyzer**.
2. From the TimeQuest Timing Analyzer, under **Reports**, select **Device Specific** and click **Report RSKM**.

Note: In the TimeQuest timing analyzer tool, the `report_TCCS` and `report_rskm` commands are not available when you are using SERDES in LEs. The commands are only available for transmitter and receiver with dedicated SERDES.

The Quartus II software automatically places the SERDES logic at the best location to meet timing requirements. Therefore, you are not required to perform placement constraints on the ALTLVDS IP core logic. However, you are recommended to perform timing budget evaluation for the overall LVDS interface in your system to ensure the sampling window specifications are met.

The LVDS transmitter and receiver functions with the ALTLVDS IP core are characterized and guaranteed to function correctly within the LVDS system specification (meeting TCCS and SW parameters). Therefore, timing constraints are not required for the SERDES logic using the ALTLVDS IP core. However, if the timing result does not fulfill the requirement or the design needs to be fine-tuned to improve the margin, timing constraints may be necessary.

The setup time (T_{SU}) and hold time (TH) for the LVDS channels as reported in the Quartus II timing report are based on the compiled design and served as a timing reference. You must not use these parameters in the timing report for the sampling window estimation. For sampling window specification, refer to the device datasheet for more information.

Related Information

The Quartus II TimeQuest Timing Analyzer, Volume 3: Verification, Quartus II Handbook

Obtaining the TCCS Report

For LVDS transmitters, the TimeQuest Timing Analyzer provides a TCCS report, which shows TCCS values for serial output ports.

To obtain the TCCS report (`report_TCCS`), follow these steps:

1. In the Quartus II software, under the Tools menu, click **TimeQuest Timing Analyzer**.
2. From the TimeQuest Timing Analyzer, under **Reports**, select **Device Specific** and click **Report TCCS**.

Setting Timing Constraints Using the TimeQuest Timing Analyzer GUI

Timing constraints for the LVDS receiver are needed only for the input clock ports and the synchronous input ports. The synchronous output ports and the asynchronous input and output ports are set to false path.

Constraining the Input Clock Signal

To constrain the input clock signal in the TimeQuest Timing Analyzer, follow these steps:

1. Run full compilation for the LVDS design. Ensure that the timing analysis tool is set to **TimeQuest Timing Analyzer**.
2. After full compilation completes, on the Tools menu, select **TimeQuest Timing Analyzer** to launch the TimeQuest analyzer window.
3. In the **Tasks** list, under **Diagnostic**, click **Report Unconstrained Paths** to view the list of unconstrained paths and ports of the LVDS design.
4. In the **Report** list, under **Unconstrained Paths**, click **Clock Status Summary** to view the clock that requires constraints. The default setting for all unconstrained clocks is 1 GHz. To constrain the clock signal, right-click the clock name and select **Edit Clock Constraint**.
5. In the **Create Clock** dialog box, set the period and the clock rising and falling edge (duty cycle of the clock) constraint. Refer to **Table 12** for timing constraints options and descriptions.
6. Click **Run**.

Constraining the Synchronous Input Ports

Constrain the synchronous input signals for non-DPA mode SERDES to allow the TimeQuest Timing Analyzer to consider your board channel-to-channel skew in the RSKM report. Without these constraints, you need to subtract the board channel-to-channel skew from the RSKM value reported by the TimeQuest Timing Analyzer.

To constrain the synchronous input signals in the TimeQuest Timing Analyzer, follow these steps:

1. Run full compilation for the LVDS design. Ensure that the timing analysis tool is set to **TimeQuest Timing Analyzer**.
2. After full compilation completes, on the Tools menu, select **TimeQuest Timing Analyzer** to launch the TimeQuest analyzer window.
3. In the **Tasks** list, under **Diagnostic**, double-click **Report Unconstrained Paths** to view the list of unconstrained paths and ports of the LVDS design.
4. In the **Report** list, under **Unconstrained Paths** category, expand the **Setup Analysis** folder, and then click **Unconstrained Input Ports**.
5. Set constraints for all the receiver synchronous input ports in the **From** list. To set input delay, perform the following steps:

a. Right-click on the synchronous input port and select **Set Input Delay**.
b. The **Set Input Delay** dialog box appears.
c. Select the desired clock using the pull down menu. The clock name must reference the source synchronous clock that feeds the LVDS receiver.
d. Set the appropriate values for **Input Delay** and **Delay**. Refer to **Table 12** for timing constraints options and descriptions.
e. Click **Run** to incorporate these values in the TimeQuest Timing Analyzer.

If no input delay is set in the TimeQuest Timing Analyzer, the receiver channel-to-channel skew (RCCS) defaults to zero.

Setting False Path for the Asynchronous Input and Output Ports

All asynchronous input and output ports are excluded from the timing analysis of the LVDS core because the signals on these ports are not synchronous to a IP core clock source. The internal structure of the LVDS IP core handles the metastability of these asynchronous signals. Therefore these asynchronous signals are set to false path.

To exclude asynchronous input and output ports from the timing analysis, perform the following steps:

1. Run full compilation for the LVDS design. Ensure that the timing analysis tool is set to **TimeQuest Timing Analyzer**.
2. After full compilation completes, on the Tools menu, select **TimeQuest Timing Analyzer** to launch the TimeQuest analyzer window.
3. In the **Tasks** list, under **Diagnostic**, double-click **Report Unconstrained Paths** to view the list of unconstrained paths and ports of the LVDS design.
4. In the **Report** list, under **Unconstrained Paths** category, expand the **Setup Analysis** folder.
5. Click **Unconstrained Input Port Paths** to view the unconstrained input ports or click **Unconstrained Output Port Paths** to view the unconstrained output ports.
6. Right-click on an ansynchronous input or output port, and select **Set False Path**.

After you specify all timing constraint settings for the clock signal, on the Constraints menu, click **Write SDC File** to write all the constraints to a specific **.sdc**. Then, run full compilation for the LVDS design again.

Setting Timing Constraints Manually in the Synopsys Design Constraint File

You can also set timing constraints manually using SDC commands in an **.sdc**, and include the **.sdc** into your Quartus II design file.

The following example shows a simple source-synchronous interface coding, where the data is aligned with respect to the falling edge of the clock.

```
#**************************************************************
# Create Clock
#**************************************************************
create_clock -name virtual_clock_lvds -period 25
create_clock -name {rx_inclock} -period 25.000 -waveform { 0.000 12.500
} [get_ports {rx_inclock}] -add
#**************************************************************
# Create Generated Clock
#**************************************************************
derive_pll_clocks
**************************************************************
# Set Input Delay
#**************************************************************
```

```
set_input_delay -clock [get_clocks virtual_clock_lvds] -clock_fall -max
0.200 [get_ports rx_in*] -add_delay
set_input_delay -clock [get_clocks virtual_clock_lvds] -clock_fall -min
-0.200 [get_ports rx_in*] -add_delay
```

To add the **.sdc** into your Quartus II design file, follow these steps:

1. In the Quartus II software, click on the Assignments menu, and select **Settings**.
2. On the **Settings** page, under **Category**, select **TimeQuest Timing Analyzer**.
3. On the **TimeQuest Timing Analyzer** subwindow, browse to the **.sdc**, and click **Add**.
4. Click **OK**.

The following table lists the LVDS timing constraints options and descriptions.

Table 12: LVDS Timing Constraints Options and Descriptions

Port Name	Constraint Type	Option		Description
		GUI Setting	SDC command	
Input Clock Constraints				
`rx_inclock`	`create_clock`	Clock name	`-name`	Specifies the name of the LVDS input clock.
		Period	`-period`	Specifies the clock period ($1/f_{max}$).
		Rising, Falling	`-waveform`	Specifies the clock's rising and falling edges or the duty cycle of the clock. For example, a 10 ns period where the first rising edge occurs at 0 ns and the first falling edge occurs at 5 ns would be written as waveform {0 5}. The difference must be within one period unit, and the rise edge must come before the fall edge. The default edge list is {0 <period>/2}, or a 50 percent duty cycle.
		Target	`[get_ports` {<*port name*>}`]`	Specifies the clock input port name connected to `rx_inclock`.
Synchronous Input Port Constraints				
		Minimum, Maximum	`-max` `-min`	Specifies the maximum and minimum delay for the data input to the FPGA.

Port Name	Constraint Type	Option		Description
		GUI Setting	SDC command	
		Rise, Fall, Both	`-clock fall` `-clock rises`	Specifies the clock's rising and falling edges or the duty cycle of the clock.
`rx_in`	`set_input_delay`	Delay	-<*delay value*>	Specifies the data to clock skew in ns.
		Target	`[get_ports` {<*port name*>}`]`	Specifies the data input port name connected to `rx_in`.

Related Information

The Quartus II TimeQuest Timing Analyzer, Volume 3: Verification, Quartus II Handbook

Arria II GX, Arria V, Arria V GZ, Cyclone V, and Stratix V LVDS Package Skew Compensation Report Panel

This section describes the LVDS package skew compensation report panel for the transmitter and non-DPA receiver of the Arria II GX, Arria V, Arria V GZ, Cyclone V, and Stratix V device families.

The report panel contains details about the package trace delay compensation needed between the LVDS pins on the device to meet your timing budget. You can find the report panel in the Quartus II Fitter report under **Resource Section**. The report panel is called **LVDS Receiver Package Skew Compensation**, and **LVDS Transmitter Package Skew Compensation** for the LVDS receiver and LVDS transmitter respectively. The report panel is triggered in the Quartus II software when your design uses a non-DPA receiver, and with an input data rate higher than 840 Mbps.

The following figure shows the LVDS Transmitter Package Skew Compensation report panel.

Figure 12: LVDS Transmitter Package Skew Compensation

LVDS Transmitter Package Skew Compensation

	Name	Pin	Recommended Trace Delay Addition	Estimated TCCS Reduction
1	lvds1:inst2|altlvds_tx:altlvds_tx_component|lvds1_lvds_tx1:auto_generated|wire_tx_dataout[0]	pin_name7[0]	67ps	
2	lvds1:inst2|altlvds_tx:altlvds_tx_component|lvds1_lvds_tx1:auto_generated|wire_tx_dataout[1]	pin_name7[1]	35ps	
3	lvds1:inst2|altlvds_tx:altlvds_tx_component|lvds1_lvds_tx1:auto_generated|wire_tx_dataout[2]	pin_name7[2]	54ps	
4	lvds1:inst2|altlvds_tx:altlvds_tx_component|lvds1_lvds_tx1:auto_generated|wire_tx_dataout[3]	pin_name7[3]	34ps	
5				33ps

The following figure shows the LVDS Receiver Package Skew Compensation report panel.

Figure 13: LVDS Receiver Package Skew Compensation

LVDS Receiver Package Skew Compensation

	Name	Pin	Recommended Trace Delay Addition	Estimated Sampling Window Reduction
1	lvds1:inst2|altlvds_rx:altlvds_rx_component|lvds1_lvds_rx:auto_generated|wire_rx_dataout[12]	pin_name3[3]	63ps	
2	lvds1:inst2|altlvds_rx:altlvds_rx_component|lvds1_lvds_rx:auto_generated|wire_rx_dataout[8]	pin_name3[2]	79ps	
3	lvds1:inst2|altlvds_rx:altlvds_rx_component|lvds1_lvds_rx:auto_generated|wire_rx_dataout[4]	pin_name3[1]	52ps	
4	lvds1:inst2|altlvds_rx:altlvds_rx_component|lvds1_lvds_rx:auto_generated|wire_rx_dataout[0]	pin_name3[0]	65ps	
5	lvds1:inst2|altlvds_rx:altlvds_rx_component|lvds1_lvds_rx:auto_generated|wire_pll_clk[0]	pin_name4	57ps	
6				16ps

The **Recommended Trace Delay Addition** column in the report panel displays the recommended amount of trace delay that you must add to each trace of the corresponding LVDS pins, which reduces the channel-to-channel skew between the LVDS channels. For example, in **Figure 12**, the recommended trace delay addition for `pin_name7[0]` is 67 ps. This means you must manually adjust the PCB trace for `pin_name7[0]` to have a delay addition of 67 ps. The corresponding pin is listed in the **Pin** column, in the report panel.

The report panel also shows the total estimated TCCS and SW reductions when the recommended trace delay values are added to the PCB trace.

ALTLVDS IP Core in External PLL Mode

PLL Clock Signals for LVDS Interface in External PLL Mode

The parameter editor provides the **Use External PLL** option. This option allows you to control PLL settings to support different data rates, dynamic phase shift, and other settings. In external PLL mode, you must instantiate a PLL IP core to generate the various clock and load enable signals.

Note: For Stratix IV, Arria II, and Cyclone IV devices, use the ALTPLL IP core. For Stratix V, Arria V, and Cyclone V devices use the Altera PLL IP core.

If you enable the **Use External PLL** option, you require the following signals from the PLL IP core:

- Serial clock input to the SERDES of the ALTLVDS transmitter and receiver.
- Load enable to the SERDES of the ALTLVDS transmitter and receiver.
- Parallel clock to clock the transmitter FPGA fabric logic.
- Parallel clock for the receiver `rx_syncclock` port and receiver FPGA fabric logic.
- Asynchronous PLL reset port of the ALTLVDS receiver.

Generate the serial clock output, load enable output, and the parallel clock output on ports `c0`, `c1`, and `c2`, along with the locked signal of the PLL IP core instance. You can choose any of the PLL output clock ports to generate the interface clocks.

Note: The high-speed clock generated from the PLL is for clocking the LVDS SERDES circuitry only. Do not use the high-speed clock to drive other logic because the allowed frequency to drive the core logic is restricted by the PLL FOUT specification.

Table 13: Signal Interface Between PLL IP Core and ALTLVDS IP Core

This table lists the signal interface between the output ports of the PLL IP core and the input ports of the ALTLVDS transmitter and receiver.

From the PLL	To the ALTLVDS Transmitter	To the ALTLVDS Receiver
Serial clock output (c0) **Note:** The serial clock output (c0) can only drive `tx_inclock` on the ALTLVDS transmitter and `rx_inclock` on the ALTLVDS receiver. This clock cannot drive the core logic.	`tx_inclock` (serial clock input to the transmitter)	`rx_inclock` (serial clock input)

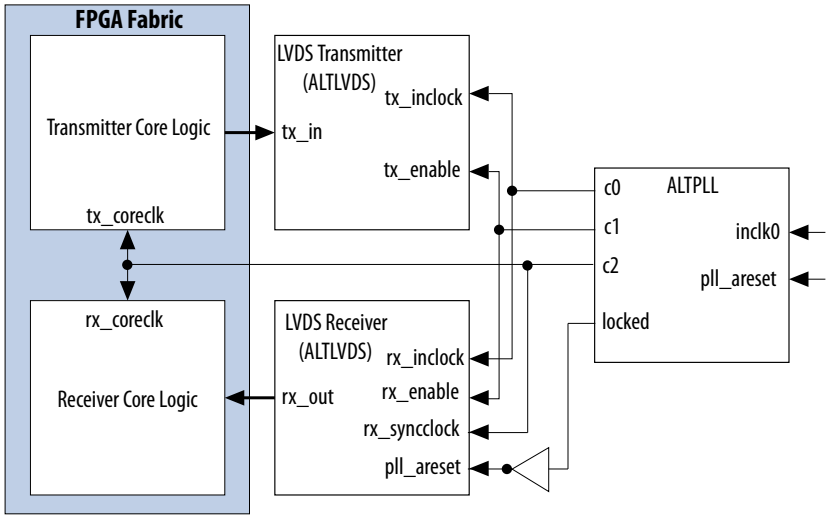
From the PLL	To the ALTLVDS Transmitter	To the ALTLVDS Receiver
Load enable output (c1)	`tx_enable` (load enable to the transmitter)	`rx_enable` (load enable for the deserializer)
Parallel clock output (c2)	Parallel clock used inside the transmitter core logic in the FPGA fabric	`rx_syncclock` (parallel clock input) and parallel clock used inside the receiver core logic in the FPGA fabric
~(locked)	—	`pll_areset` (asynchronous PLL reset port) **Note:** The `pll_areset` signal is automatically enabled for the LVDS receiver in external PLL mode. This signal does not exist for LVDS transmitter instantiation when the external PLL option is enabled.

The `rx_syncclock` port is not always required by the LVDS receiver in external PLL mode. If it is required, the Quartus II software automatically generates the port. Even if `rx_syncclock` (c2) is not used in the LVDS receiver, you must still use it to clock the FPGA fabric. The Quartus II compiler errors out if this port is not connected, as shown in the following figure.

Note: When generating the ALTPLL IP core for Arria II devices, select the **Left/Right PLL** PLL type to set up the PLL for LVDS.

The following figure shows the connection between the PLL IP core and the ALTLVDS IP core.

Figure 14: LVDS Interface with the PLL IP Core



Instantiation of pll_areset is optional for the ALTPLL instantiation.

Table 14: Example Settings to Generate Three Output Clocks using PLL IP Core

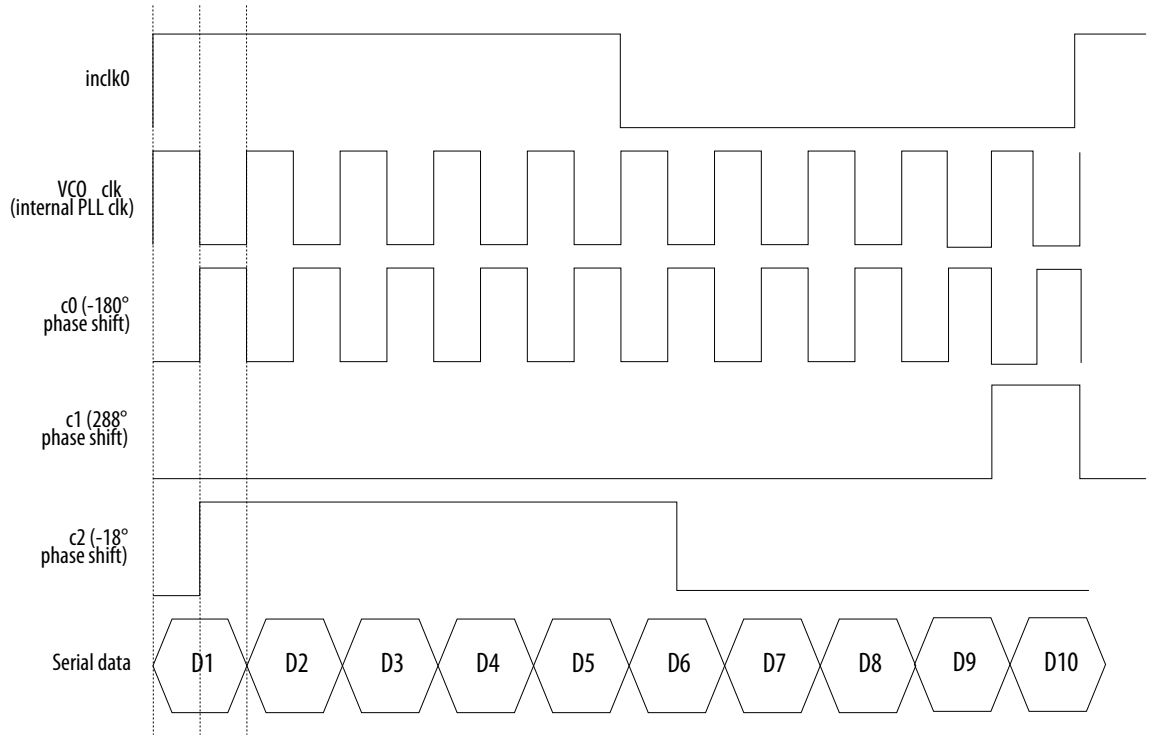
This table shows an example with the parameter values that you can set in the PLL IP core parameter editor to generate three output clocks.

Parameter/Clock	Setting
Serial clock	Frequency = **1000 MHz**
Parallel clock	Frequency = **100 MHz** (serial clock divided by the serialization factor)
LVDS data rate	1 Gbps
Serialization factor	10
Input reference clock	Frequency = **100 MHz**
c0	• Frequency = **1000 MHz** (multiplication factor = 10 and division factor = 1) • Phase shift = **–180°** with respect to the voltage-controlled oscillator (VCO) clock • Duty cycle = **50%**
c1	• Frequency = (1000/10) = **100 MHz** (multiplication factor = 1 and division factor = 1) • Phase shift = (10 - 2) × 360/10 = **288°** [(deserialization factor - 2)/deserialization factor] × 360° • Duty cycle = (100/10) = **10%** (100 divided by the serialization factor)
c2	• Frequency = (1000/10) = **100 MHz** (multiplication factor = 1 and division factor = 1) • Phase shift = (–180/10) = **–18°** (c0 phase shift divided by the serialization factor) • Duty cycle = **50%**

Phase shift calculations using RSKM equation assume that the input clock and serial data are edge aligned. The following figure shows that by introducing a phase shift of –180° to sampling clock (c0) ensures that the input data is center-aligned with respect to the `c0`.

Note: The phase shift example used in this section assumes that the clock and data are edge-aligned at the FPGA pins. For other clock relationships, Altera recommends that you create the ALTLVDS_TX and ALTLVDS_RX IP cores initially without using the external PLL option. Set the phase shifts you require in the parameter editor and then note the phase shift and duty cycle settings for the three PLL output clocks in the Quartus II software Compilation Report (**Resource** > **Fitter** > **PLL Usage** section). Once you have the correct phase shift and duty cycle settings for your parameterization, you can implement the external PLL mode in your design. In the parameter editor for the PLL IP core, enter the phase shift and duty cycle values for each output clock based on the values you previously noted from the **PLL Usage** report.

Figure 15: Phase Relationship for External PLL Interface Signals



Related Information

- **DC and Switching Characteristics for Stratix IV Devices**
- **Receiver Skew Margin and Transmitter Channel-to-Channel Skew** on page 54

External PLL Compensation Mode for ALTLVDS IP Core in External PLL Mode

If you instantiate the ALTLVDS IP core in external PLL mode, Altera recommends that you set up the data rate and clocking with the PLL IP core.

Note: For Stratix IV, Arria II, and Cyclone IV devices, use the ALTPLL IP core. For Stratix V, Arria V, and Cyclone V devices use the Altera PLL IP core.

- For Arria V, Arria V GZ, and Stratix V devices with ALTLVDS_RX configured in non-DPA mode, the external PLL must be in LVDS compensation mode.
- For Cyclone V devices, LVDS interfaces placed on the all edges must be in LVDS compensation mode.

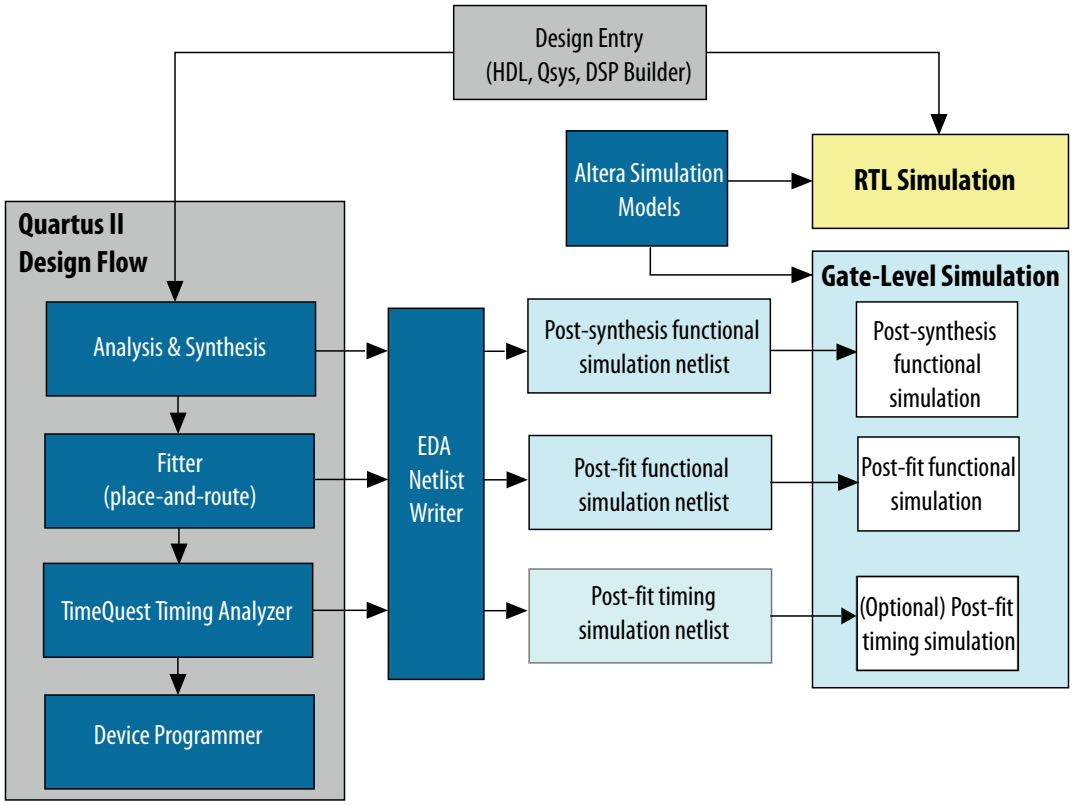
For more information about PLL compensation modes, refer to the PLL chapter of the relevant device handbook.

Simulating Altera IP Cores in other EDA Tools

The Quartus II software supports RTL and gate-level design simulation of Altera IP cores in supported EDA simulators. Simulation involves setting up your simulator working environment, compiling simulation model libraries, and running your simulation.

You can use the functional simulation model and the testbench or example design generated with your IP core for simulation. The functional simulation model and testbench files are generated in a project subdirectory. This directory may also include scripts to compile and run the testbench. For a complete list of models or libraries required to simulate your IP core, refer to the scripts generated with the testbench. You can use the Quartus II NativeLink feature to automatically generate simulation files and scripts. NativeLink launches your preferred simulator from within the Quartus II software.

Figure 16: Simulation in Quartus II Design Flow



Note: Post-fit timing simulation is not supported for 28nm and later device archetectures. Altera IP supports a variety of simulation models, including simulation-specific IP functional simulation models and encrypted RTL models, and plain text RTL models. These are all cycle-accurate models. The models support fast functional simulation of your IP core instance using industry-standard VHDL or Verilog HDL simulators. For some cores, only the plain text RTL model is generated, and you can simulate that model. Use the simulation models only for simulation and not for synthesis or any other purposes. Using these models for synthesis creates a nonfunctional design.

Related Information

Simulating Altera Designs

Generating ALTLVDS IP Core Using Clear Box Generator

Apart from the IP core parameter editor, you can also use the clear box generator, a command-line executable, to configure parameters that are in the ALTLVDS_TX and ALTLVDS_RX parameter editors. The clear box generator creates or modifies custom IP core variations that you can instantiate in a design file. The clear box generator generates IP core variation file in Verilog HDL or VHDL format.

1. Create a text file (**.txt**) that contains your clear box ports and parameter settings in your working directory.
2. Open the command prompt and change the current directory to your working directory by typing: `cd c:\altera\11.0\quartus\work\`

   The clear box executable file name is **clearbox.exe**.
3. To view the available ports and parameters for this IP core, type one of the following commands: `clearbox altlvds_tx -h` or `clearbox altlvds_rx -h`.
4. To generate the ALTLVDS_TX and ALTLVDS_RX IP cores variation file based on the ports and parameter settings in the text file, type one of the following commands: `clearbox altlvds_tx -f *.txt` or `clearbox altlvds_rx -f *.txt`.

   For example, `clearbox altlvds_tx -f sample_param_test.txt`
5. After the clear box generator generates the IP core variation files, instantiate the IP core module in a HDL file or a block diagram file in the Quartus II software.
6. To view the estimated hardware resources that the ALTLVDS_TX and ALTLVDS_RX IP cores use, type one of the following commands: `clearbox altlvds_tx -f sample_param_test.txt -resc_count` or `clearbox altlvds_rx -f sample_param_test.txt -resc_count`.

   This command does not generate a HDL file.

Document Revision History

Date	Version	Changes
December 2014	2014.12.15	• Added footnotes to clarify the availability of DPA and soft-CDR modes in Stratix series. • Removed Cyclone series from the list of series with soft-CDR support. • Added guidelines about the time required for tx_outclock to stabilize if you turn on the **Implement Deserializer circuitry in logic cells** option. • Updated the statement that refers to selecting "Left/Right PLL" to set up PLL in LVDS mode to clarify that the option is required only for Arria II devices. • Updated information about the PLL IP core to clarify that for Stratix IV, Arria II, and Cyclone IV devices, the PLL IP core is ALTPLL IP, and for Stratix V, Arria V, and Cyclone V devices, the PLL IP core is Altera PLL.

Date	Version	Changes

November 2014	2014.11.17	• Restructured and updated sections that describe the external PLL mode and the relevant ALTPLL IP core parameters.
		  • Added recommendations about getting the correct ALTPLL phase shift and duty cycle values for the external PLL mode.
		  • Clarified that the `rx_syncclock` is automatically created by the Quartus II software only when it is required.
		• Updated the ALTLVDS_RX ports list to clarify that the `rx_cda_reset` port is not supported in Arria V and Cyclone V devices. In these devices, use the `rx_channel_data_align` signal instead.

June 2014	2014.06.30	• Replaced MegaWizard Plug-In Manager information with IP Catalog. • Added standard information about upgrading IP cores. • Added standard installation and licensing information. • Removed outdated device support level information. IP core device support is now available in IP Catalog and parameter editor. • Updated a statement about valid data availability for `rx_channel_data_align` signal in the topic about ALTLVDS_RX parameter settings.
November 2013	2013.11.08	Updated the following parameters: • `outclock_alignment`: clarify that this parameter is only used by the RTL simulation model and has no affect on how the Fitter sets the PLL parameters. • `outclock_phase_shift`: clarified that this parameter is used to set the phase shift parameters used by the PLL. Updated the following parameters: • `inclock_data_alignment`: clarified that this parameter is only used by the RTL simulation model and has no affect on how the Fitter sets the PLL parameters. • `inclock_phase_shift`: clarified that this parameter is used to set the phase shift parameters used by the PLL.

Date	Version	Changes
June 2013	2013.06.10	• Removed **Use clock pin** parameter. This parameter is no longer available for the megafunction beginning from ACDS 13.0. • Updated **Table 1** to include Arria V, Arria V GZ, and Stratix V device family support. Also added a note to clarify that Altera recommends implementing the Bus LVDS (BLVDS) I/O with user logic, instead of the ALTLVDS_TX and ALTLVDS_RX megafunctions. • Updated **Table 5** and **Table 6** to remove Stratix V device family support and to clarify that In Cyclone series, except Cyclone V, the SERDES is always implemented in logic cells for the Implement Deserializer circuitry in logic cells option. • Updated **Table 5** to clarify that the values for the **What is the phase alignment of 'tx_in' with respect to the rising edge of 'tx_inclock'? (in degrees)** option is device dependent. • Updated **Table 5** and **Table 6** to remove Stratix V device family support for the **Enable self-reset on lost lock in PLL**, **Enable PLL Calibration**, and **Use 'dpa_pll_recal' input port** options. • Updated **Table 6** to add Arria V and Arria V GZ devices support for the **Enable Dynamic Phase Alignment mode**, **Use 'rx_divfwdclk' output port and bypass the DPA FIFO**, **Use 'rx_dpa_locked' output port**, **Use a DPA initial phase selection of**, and **Align DPA to rising edge of data only** options. • Updated **Table 6** to clarify that the values for the **What is the phase alignment of 'rx_in' with respect to the rising edge of 'rx_inclock'?** option is device dependent. • Updated **Table 6** to add the **Is this interface constrained to the left, or right banks?** option. • Updated to add Arria V and Arria V GZ devices support for `common_rx_tx_pll`. • Updated to remove Stratix V device family support for the `deserialization_factor`, `use_no_phase_shift`, `use_external_pll`, and `pll_self_reset_on_loss_lock` (Stratix V devices do not support SERDES using logic cells). • Updated to add Arria V and Arria V GZ devices support for `deserialization_factor`. • Updated to add Arria V and Arria V GZ devices support for `inclock_data_alignment`, `outclock_divide_by`, `outclock_duty_cycle`, `outclock_resource`, `registered_input`, and `use_external_pll`. • Updated to add Arria V, Arria V GZ, Cyclone V, and Stratix V devices.

Date	Version	Changes
June 2013	2013.06.10	• Updated **Standard Mode** on page 47 to add a note to recommend using `rx_divfwdclk` (instead of any static clock) as the SignalTap capturing clock. • Updated **Receiver Skew Margin and Transmitter Channel-to-Channel Skew** on page 54to fix the error in RSKM equation by replacing TCCS with RCCS. Also added information on how to apply the RCCS figure to the RSKM calculation in TimeQuest. • Updated **Arria II GX, Arria V, Arria V GZ, Cyclone V, and Stratix V LVDS Package Skew Compensation Report Panel** on page 61 to add Arria V, Arria V GZ, and Cyclone V devices. • Updated **Figure 2** to fix the waveform error for c1 (288 degrees phase shift) • Updated `tx_enable` and `rx_enable ports` in**Table 11** and **Table 10** to clarify that the **Set up PLL in LVDS mode** option and the `enable0` and `enable1` ports are only for Stratix II devices. • Updated **Parameters Used by the ALTPLL Megafunction**. • Added a link to the **High-Speed Differential I/O Interfaces and DPA in Arria V Devices**
October 2012	v9.1	• Updated Table 2-2 on page 2-7 to fix content error for the What is the deserialization factor? and Use 'rx_dpa_locked' output port options. • Updated "Clock Forwarding" on page 3-1. • Updated "DPA PLL Calibration" on page 3-4 to fix device family support. • Updated "Dedicated SERDES" on page 3-9 to add a note on TimeQuest Timing Analyzer. • Updated Table 3-5 on page 3-25 to update description for rx_in[]and rx_inclock. • Updated Table 3-6 on page 3-28 to update description for tx_inclock and tx_out[].
February 2012	v9.0	• Updated "Source-Synchronous Timing Analysis and Timing Constraints" section. • Added design examples. • Updated "Parameter Settings" chapter to include "Use Clock Pin" parameter.

Date	Version	Changes
June 2011	v.8.0	• Reorganized the document format. • Added "Source-Synchronous Timing Analysis and Timing Constraints" section. • Added "Generating Clock Signals for LVDS Interface" section. • Updated the timing diagram in the "Receiver Skew Margin and Transmitter Channel-to-Channel Skew" section. • Updated "Parameter Settings" chapter. • Added "Using Clear Box Generator" section.
August 2010	v.7.0	• Updated "DPA PLL Calibration in Stratix III and Stratix IV E Devices" section. • Added Verilog HDL prototypes. • Added VHDL LIBRARY-USE declaration. • Added VHDL Component Declarations. • Added new ports and parameters. • Added new parameter settings. • Removed Design Examples for this release.
November 2009	v6.1	Added "Arria II GX and Stratix V LVDS Package Skew Compensation Report Panel".
September 2009	v6.0	• Added "Device Support". • Updated "Specifications" section to include "Ports and Parameters in ALTLVDS_RX Megafunction" and "Ports and Parameters in ALTLVDS_TX Megafunction". • Added "Specifications".
March 2009	v5.0	• Updated Table 4, and Table 12. • Added DPA Misalignment Issue, Figure 3, and "DPA PLL Calibration", Figure 20 and Figure 21. • Added Table 11 ALTLVDS Receiver DPA settings 3 option (page 7) and Table 19 Configuration Settings for Design Example 4 (LVDS Receiver). • Added description about "Design Example 4: Stratix III ALTLVDS Receiver with DPA PLL Calibration.

Date	Version	Changes
December 2008	v4.0	Updated for the Quartus II software 8.1: • Removed figures. • Added Stratix IV to Device Family Support. • Updated Table 3, Table 4, Table 5, Table 6, Table 7, Table 8, Table 12, Table 13, Table 15,Table 3-1, Table 3-2, Table 3-3, Table 3-4, and Table 3-6. • Added Enable bitslip control, Enable independent bitslips controls for each channel, and Register the bitslip control input using 'rx_ outclock' parameters and descriptions Table 11. • Updated steps in Functional Results-Simulate the ALTLVDS Receiver/Transmitter Design in the ModelSim-Altera Software, Functional Results-Simulate the ALTLVDS Receiver/Transmitter Design in the Quartus II Software, "Functional Results-Simulatethe ALTLVDS Receiver/Transmitter Design in the ModelSim-Altera Software". • Added tx_syncclock and descriptions in Table 3-1. • Added rx_data_align and rx_syncclock in Table 3-4. • Updated descriptions in Table 3-6.
May 2008	v3.4	Small changes to Table 2-7 on page 2-27 and Table 2-9 on page 2-32.
November 2007	v3.3	Updated for the Quartus® II software v7.2, including: • Added soft-CDR mode. • Added description of new receiver output port rx_divfwdclk[]. • Added description of new receiver parameters enable_soft_cdr, is_ negative_ppm_drift,net_ppm_variation, enable_dpa_align_to_ rising_edge_only, dpa_initial_phase_ value, and enable_dpa_ initial_phase _selection. • Updated two design examples. • Added third design example using soft-CDR mode.
March 2007	v3.2	Updated for Quartus II software 7.0, including Cyclone® III information.
December 2006	v3.1	Updated Table 1-1 to include Stratix® III information.
November 2006	v3.0	Updated for the Quartus II software 6.1.
June 2006	v2.0	Updated for the Quartus II software 6.0.
August 2005	v1.1	Minor content changes.
December 2004	v1.0	Initial release.