Design Guidelines for Implementing LVDS Interfaces in Cyclone Series Devices

June 2008, ver. 1.0 Application Note 479

Introduction

This application note describes the ways to use the Cyclone® series (Cyclone III, Cyclone II and Cyclone) devices for high-performance, low-voltage differential signaling (LVDS) interfaces. LVDS is a signaling standard that provides many benefits that allow high-speed data transfers. The cost-optimized Cyclone series devices offer easy integration of LVDS interfaces at speeds up to 875 Mbps for the receiver and 840 Mbps for the transmitter. This application note includes step-by-step design flow and interface guidelines. With simple settings and considerations using the altlvds megafunction in the Quartus® II software, the Cyclone series devices become part of the high-speed setup in your system, with increased system integration at reduced cost.

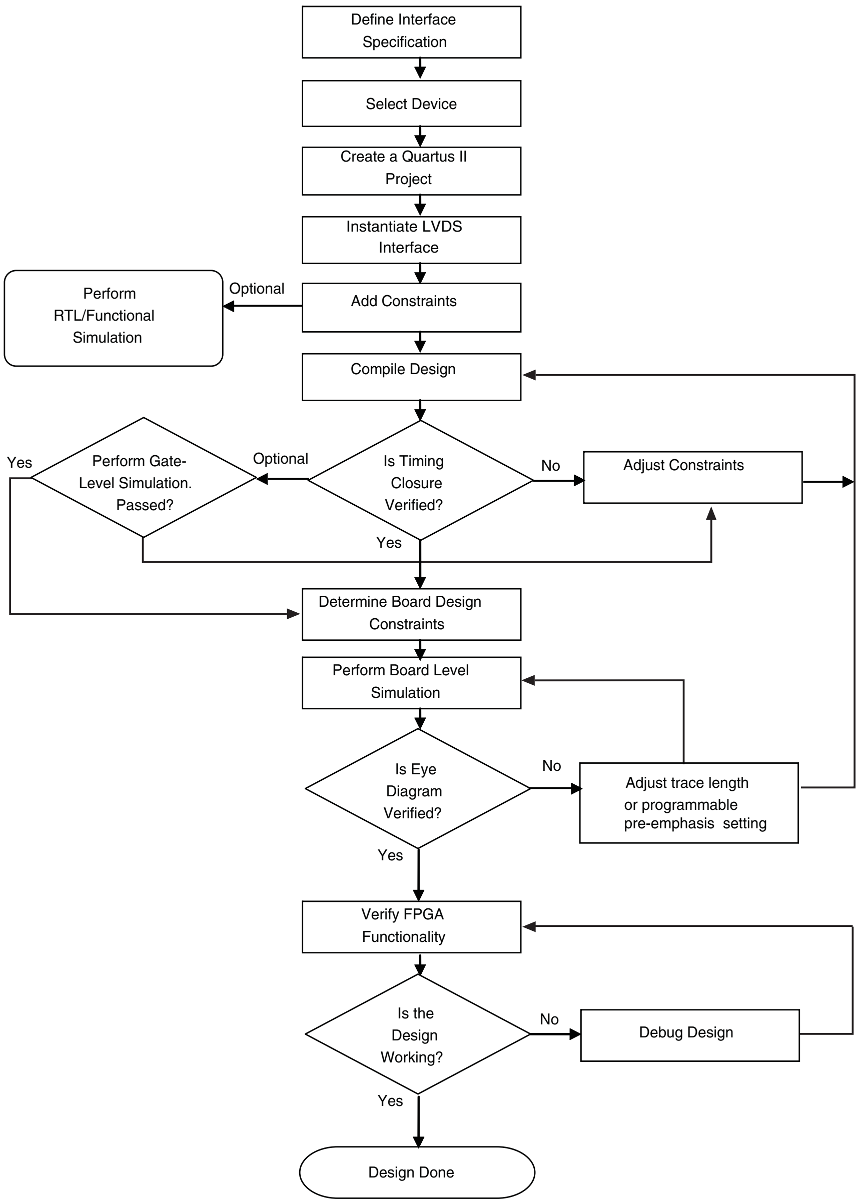
Use this application note in conjunction with the following chapters from the respective device family handbooks:

- *High-Speed Differential Interfaces in Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook.*
- *High-Speed Differential Interfaces in Cyclone II Devices* chapter in volume 1 of the *Cyclone II Device Handbook.*
- *High-Speed Differential Signaling in Cyclone Devices* chapter in volume 1 of the *Cyclone Device Handbook.*

FPGA Design Flow

This section describes the design flow to implement an LVDS interface in a Cyclone series device using the altlvds megafunction as illustrated in Figure 1. The following subsections detail the step-by-step design flow and simple guidelines to integrate a LVDS interface block to a Cyclone series devices system.

Figure 1. Design Flow for Implementing LVDS Interfaces in Cyclone Series Devices


Define Interface Specification
Select Device
Create a Quartus II Project
Instantiate LVDS Interface
Add Constraints
Optional
Perform RTL/Functional Simulation
Compile Design
Is Timing Closure Verified?
No
Adjust Constraints
Optional
Perform Gate-Level Simulation. Passed?
Yes
Yes
Determine Board Design Constraints
Perform Board Level Simulation
Is Eye Diagram Verified?
No
Adjust trace length or programmable pre-emphasis setting
Yes
Verify FPGA Functionality
Is the Design Working?
No
Debug Design
Yes
Design Done

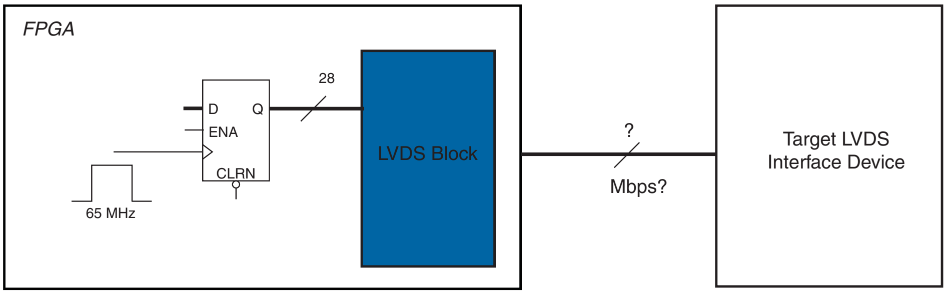
Step 1: Define Interface Specification

Properly defining the interface requirement would help you to allocate and optimize the resources used. At the first stage of integrating the LVDS interface, you can identify the number of channels required and the speed of data transmission. This would determine the number of physical pair of differential traces (or connections) that are required between your FPGA and the target device.

Begin by identifying the parallel data width and the frequency of the bus that you want to implement as an LVDS interface. From here, determine the number of channels that are required and the deserialization factor to use. The altlvds megafunction supports ×1 to ×10 deserialization factors, with the exception of ×3. Higher deserialization factor translates to less channels and faster data rates (in bits per second per channel).

Here is an example to determine the suitable number of channels and the deserialization factor. Take the case of a 28-bit transmit bus running at 65 MHz as illustrated in Figure 2.

Figure 2. Example of a LVDS Interface Setup



First, determine the suitable deserialization factor for the system. Divide the data width (28) with each available deserialization factors (from ×1 to ×10). The deserialization factors that yield zero remainder are suitable for the system. Table 1 shows the deserialization factor that yields zero remainder (×1, ×2, ×4 and ×7).

Secondly, calculate the number of channels required for each selected deserialization factor. This value is the result of the data width divided with the associated deserialization factor.

Lastly, multiply the bus frequency (65 MHz) with the deserialization factor to obtain the data-rates per channel.

Table 1. Deserialization Factor, Number of Channels and Data Rates per Channel Combinations Notes (1), (2)

Deserialization factor	Number of channels (1), (2)	Data rates per channel (Mbps)
×1	28	65
×2	14	130
×4	7	260
×7	4	455

Notes to Table 1

(1) Each channel uses a pair of pins; one pin each for inverted (n) and non-inverted (p) signal.

(2) For ×2 and higher deserialization factors, LVDS transmission employs double-data rate, where data are clocked at both rising and falling edge. Data rate is doubled without the increase in the number of channels.

For optimum resource utilization and performance, select the deserialization factor that yields the least number of channels and the highest data rate that can be supported. In example shown in Table 1, ×7 with four channels at 455 Mbps is the best choice which only requires four pairs of data outputs for a 28-bit bus chip-to-chip transmission.

Step 2: Select Device

After you have determined the interface specifications (deserialization factor, channels and data rates), identify which FPGA to use. Device selection can be divided into four main categories:

- "Timing Performance"
- "I/O Resources"
- "PLL Resources"
- "Clock Resources"

The following sections describe the four categories used for device selection. The discussion would only focus on the three families in the Cyclone series devices.

Timing Performance

Timing performance is often associated with data rate, which determines how many bits of data can be transferred in one second. However, there are skew-related performance parameters that require similar attention for device selection. This section discusses the transmission performance options and the skew-related parameters consideration.

Altera® offers the Cyclone series FPGAs in various performance options that are differentiated by speed grade. For example, commercial ranges in -6, -7 and -8 (slowest). Each device family and each speed grade supports a specific timing performance. Table 2 lists the maximum supported transmit data rate for LVDS interfaces.

Table 2. LVDS Maximum Transmit Performance for Cyclone Series Devices *Notes (1), (2)*

Devices	Connection Type	-6 Speed Grade Commercial (Mbps)	-7 Speed Grade Commercial (Mbps)	-8 Speed Grade Commercial (Mbps)
Cyclone III	Dedicated *(1)*	840	740	640
	Three-Resistor	640	640	550
Cyclone II	Three-Resistor	640	640 (550) *(2)*	550 (311) *(2)*
Cyclone	Three-Resistor	640	640	550

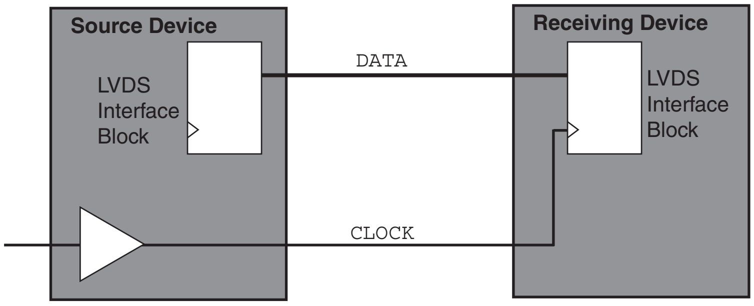
Notes to Table 2:

(1) Cyclone III devices support the LVDS output operations through dedicated differential transmitters on the row I/O banks. LVDS output operations can also be emulated using a pair of single-ended output with external three-resistor network on column I/O pins.

(2) The maximum data rate may not comply with duty cycle distortion of 45-55%. The numbers in parentheses indicate the maximum data rate that comply with duty cycle distortion of 45-55%.

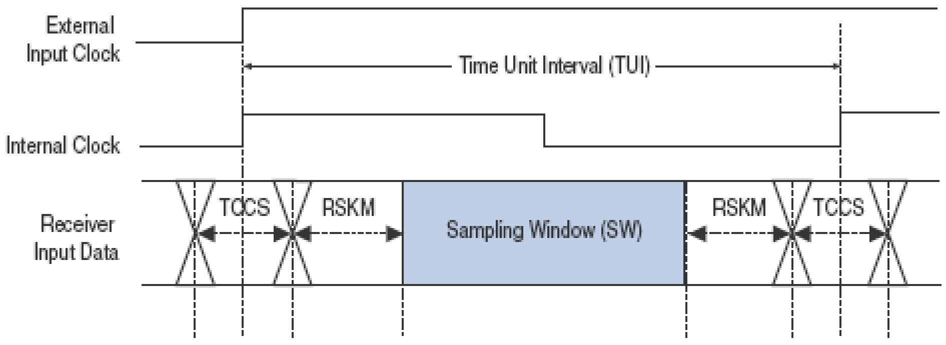
LVDS is a source-synchronous signaling interface where the clock is sourced from the same device as the data rather than another source, such as a common clock network as illustrated in Figure 3.

Figure 3. Simplified Source-Synchronous Interface



Therefore, the timing for LVDS interface is based on the skew between the data and clock signals. The skew-related parameters are transmitter channel-to-channel skew (TCCS) and sampling window (SW). Lower device-level skew values give you more timing margin to account for various board-level effects such as skew, jitter and noise. The timing margin for the interface is represented by receiver input skew margin (RSKM) as shown in Figure 4.

Figure 4. High-Speed I/O Timing Diagram



For detailed descriptions of the timing parameters, refer to the *High-Speed I/O Timing* section in *High-Speed Differential Interfaces* of the device handbook.

I/O Resources

The number of differential channels vary across device family, density and package. Use the device pin-outs to determine if the number of differential channels available in the FPGA meets your system requirements. Differential channels are identified by pins with `DIFFIO_xx` as **Optional Function(s)** in Cyclone III devices and `LVDSxx` as **Optional Function(s)** in Cyclone II and Cyclone devices. Requirements that affect the differential channels usage are:

- Bank V_{CCIO} level for LVDS inputs and outputs: LVDS input and output must be powered with a 2.5-V V_{CCIO} supply in the banks where it resides, except for the below conditions where a 2.5-V V_{CCIO} supply is not required:
  - All LVDS inputs for the Cyclone device, powered by V_{CCINT}.
  - All LVDS dedicated clock inputs for Cyclone II, powered by V_{CCINT}.

Supplying higher than supported voltage to V_{CCIO} for LVDS would violate the rated specifications and may possibly damage the buffer.

- Differential pad placement guidelines: There are requirements for differential pad placement with respect to single-ended I/O pads and total differential outputs per V_{CCIO} and ground pair. These requirements ensure an acceptable noise level on the V_{CCIO} supply in banks where the differential I/Os reside. Only placement requirements of single-ended I/O pads to differential pads are checked automatically by the Quartus II software. Violating the guidelines may put your system at risk of functional failure due to undesirable noise on the signals.

For the exact requirements, refer to the *Differential Pad Placement* section in the *I/O* chapter of the respective device handbook.

PLL Resources

The altlvds megafunction utilizes a phase-locked loop (PLL) to create various clocks for SERDES operations in the LVDS interface. You need at least one PLL for ×4 and higher deserialization factor operations.

There are a maximum of two PLLs available in Cyclone devices and four PLLs in Cyclone III and Cyclone II devices. The number of PLL varies according to device densities.

For exact PPL resources for a specific device density, refer to the *Clock Networks and PLLs* for Cyclone III devices or the *Clock Management* section for Cyclone II and Cyclone.

Cyclone III and Cyclone II devices support source-synchronous compensation mode PLL. Altera recommends using the source-synchronous compensation mode PLL for LVDS receiver applications to maintain the clock and data relationship at the pin.

Cyclone devices do not support source-synchronous compensation mode PLL. Therefore, use the normal compensation mode PLL in Cyclone devices for LVDS receiver applications. In such cases, you need to ensure the correct phase settings in the PLL to account for the delay from the clock pin to the input data register clock port.

More details on how to make the PLL phase settings for Cyclone devices (or Cyclone series devices that use normal compensation mode PLL) are discussed in "Step 6: Adjust Constraints" on page 17.

Clock Resources

One pair of differential dedicated clock input pins is required for the destination device. The clock output pins are used to route the received clock directly to the PLL. The Cyclone III and Cyclone II devices offer up to 16 pairs of dedicated differential clock inputs, while Cyclone devices offer up to four pairs.

Use general purpose differential channels on the Cyclone series device to implement the differential clock output when it is a source device to maintain the skew between clock and data outputs.

Step 3: Create a Quartus II Project and Instantiate the interface

Once you have selected the appropriate FPGA device, create a Quartus II project targeting the FPGA device you have chosen.

For a step-by-step guide on creating a Quartus II project, refer to the Tutorial in the Quartus II software.

There are two ways that you can instantiate the components for the LVDS interface:

- Use Altera's altlvds megafunction, which builds the SERDES functions and instantiates the PLL. You can make the necessary PLL settings in the MegaWizard® Plug-In Manager under **Frequency/PLL Settings** and **Transmitter** settings tab. This option integrates the PLL into the LVDS block, therefore simplifies the clocking setup for the system. The drawback is the reduced flexibility in PLL utilization as the PLL can only be used for that particular LVDS block instantiation and cannot be shared for other operations.
- Use Altera's altlvds megafunction with External PLL option, which only builds the SERDES functions. A notification window appears when you checked the **Use External PLL** option box. Follow the required clock setting to the input ports as listed in the notification window. You can create your own clocking source with Altera's altpll megafunction. This option is mainly for users who want to optimize the usage of the PLL with other operations in the core.

Figures 5 and 6 shows the modules of a typical LVDS interface design using internal and external PLL respectively.

Figure 5. Typical LVDS Interface Modules with altlvds Megafunction

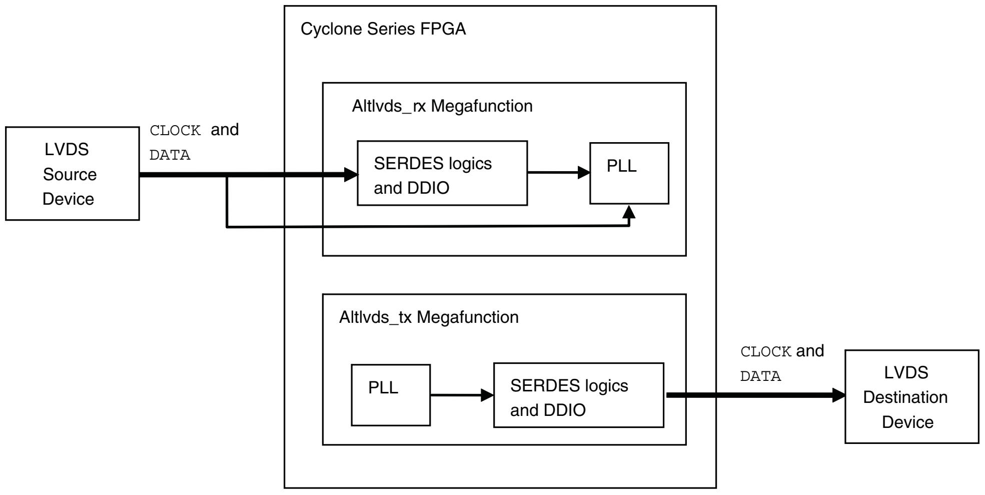
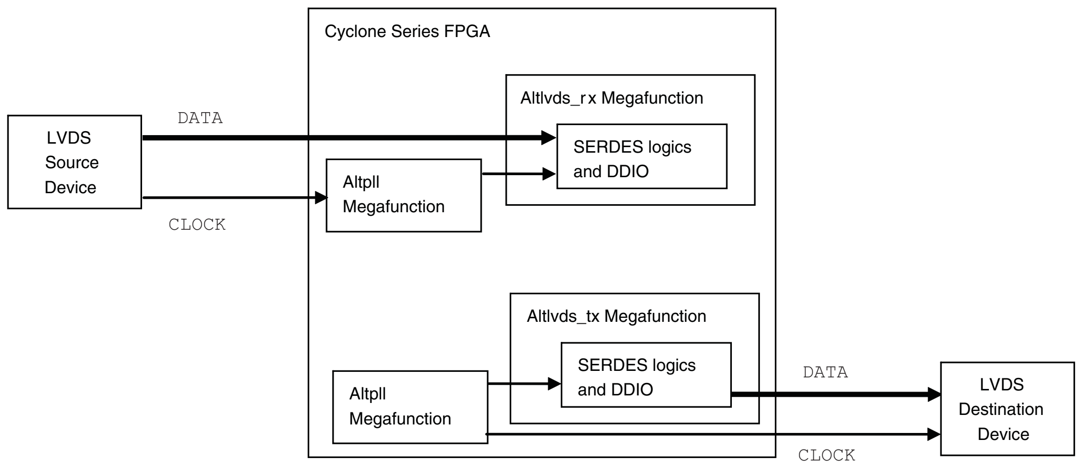


Figure 6. Typical LVDS Interface Modules with altlvds Megafunction using External PLL Option (with altpll Megafunction)



You need to consider some of the requirements when creating the settings in the altlvds MegaWizard. This section describes the:

- "Clock Settings"
- "PLL Sharing"
- "Odd SERDES Factor"
- "Data Word Alignment"

For information on ways to perform the settings discussed in this section, refer to the *altlvds Megafunction User Guide*.

Clock Settings

The altlvds megafunction requires a PLL for ×4 and higher SERDES factors to generate a fast clock with a frequency at half the data rate. The Cyclone series devices use the DDIO registers as part of the SERDES interface (except for ×1 SERDES factor), hence the frequency at half the data rate. Additionally, there are various phase-settings available in the MegaWizard for input and output clock-to-data alignment.

The external PLL option allows you to access all the PLL clocks for other functions. However, you need to ensure that all the clocking sources to the megafunction ports are in correct frequency and phase setting.

Clock-to-data phase alignment is one of the main clock settings that you need to pay attention to both the source and destination device. The clock is often set to edge-aligned with the data. However, there are other settings such as center-aligned and random phase-setting that you can use, depending on your target source or destination device and board routing conditions. If you choose to implement the altlvds megafunction which also instantiates the PLL, you can make such phase settings under the **Transmitter** settings or **Receiver** settings tab in the MegaWizard. Otherwise, you have to manually make the phase settings in the altpll megafunction.

The clock is often centered in the data valid window at destination device to maximize timing margin for data capture. There are a few recommended settings that you can make in the receiver's altlvds megafunction based on different clock-to-data alignment. Each alignment below refers to the clock-and-serial-data alignment.

- Edge-aligned: Check the **Use source-synchronous mode of the PLL** and **Align clock to center of data window at capture point** option. This option sets the receiver PLL to source-synchronous mode PLL and maintains the clock and data phase relationship at the pins to the input register. It also sets the phase of fast clock to -90° or -180° for double-data rate or single-data rate operations respectively.

For further details, refer to the *Source-Synchronous Mode* section under *Clock Feedback Modes* in the *PLL* chapter for the *Cyclone II and Cyclone III devices handbooks*.

- Center-aligned: Check the **Use source-synchronous mode of the PLL** option. Incoming clock is aligned to the center of the data valid window, hence clock centering in the destination device is not required.
- Random-aligned: Manually determine the required phase-shift setting for the incoming clock of the destination device to capture the data correctly.

For a Cyclone device used as an LVDS receiver, follow the guidelines for the random-aligned clock-to-data scenario.

PLL Sharing

In applications where both an LVDS transmitter and receiver are required, two PLLs may be required to implement each of the interfaces. The altlvds megafunction allows for PLL sharing between the transmitter and receiver to reduce the PLL utilization to only one PLL. Here are the guidelines to implement this feature:

- To share a PLL for both operations, several PLL settings must be the same, such as PLL feedback mode, clock frequency and phase settings. This means that both the LVDS transmitters and receivers must use the same input clock frequency, deserialization factor and data rates.
- For transmitters, check the **Align clock to center of data window** option if you have this option enabled for your receiver.
- Check the **Use shared PLL(s) for receivers and transmitters** option to allow the Quartus II compiler to share the same PLL.

For instructions on other PLL signals for PLL sharing, refer to the altlvds Setting Options description and comments in the *altlvds Megafunction User Guide* .

Odd SERDES Factor

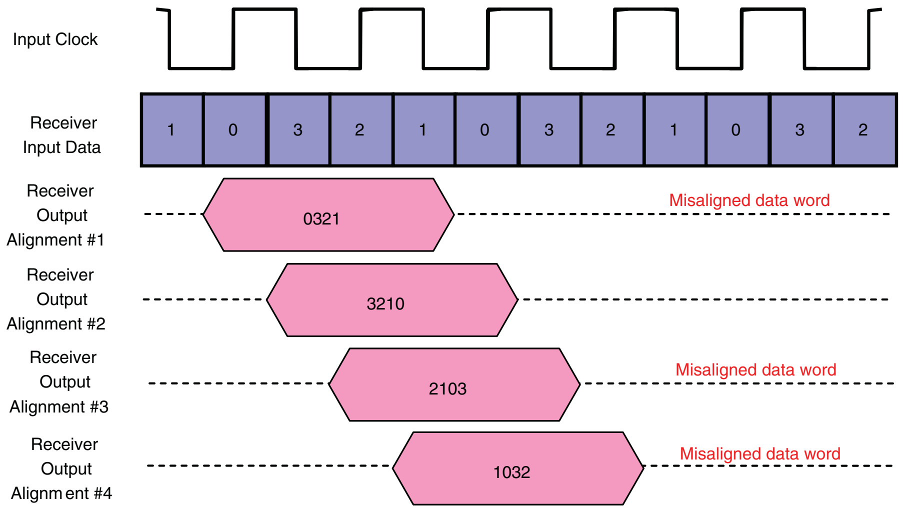
Display applications often use an odd SERDES factor such as ×7 in LVDS interfaces. When designing with odd SERDES factor, the Quartus II software by default sets the `txcoreclock` frequency to a value of `data rate` per (`SERDES factor×2`). This setting improves the timing performance in the SERDES logics and allows the device to operate at the maximum data rate. The altlvds megafunction implements the odd SERDES factor interface differently than the even SERDES factor due to the odd number of data width being fed to the double data input/output (DDIO) registers.

At certain lower data rates, you may choose a higher `txcoreclock` frequency (which is at `data rate per SERDES factor`). Higher `txcoreclock` usage reduces logic element (LE) utilization by half than with lower `txcoreclock`. Run timing simulation and check the functionality of the interface to determine if your system can operate at a higher `txcoreclock` to save on resource utilization.

Data Word Alignment

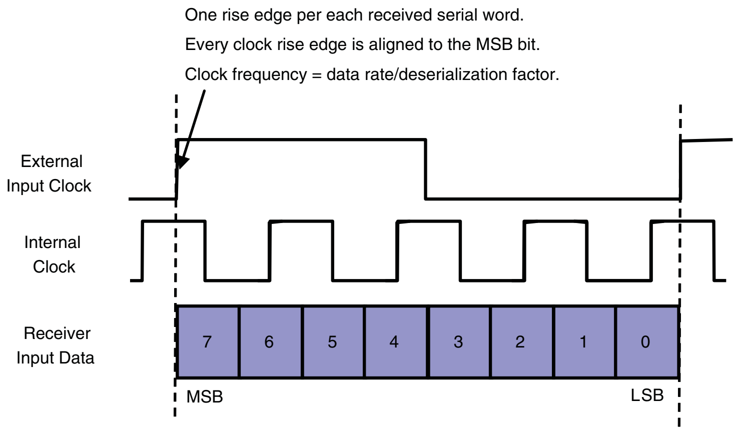
In a parallel interface, each bit in a bus is transmitted over a separate pin simultaneously. If you want to transmit an 8-bit data bus over a parallel interface, the receiving device will receive the 8 bits of data in the same order that was transmitted; most significant bit (MSB) to least significant bit (LSB) in the correct data word boundary. In a serial interface like LVDS, the data bus is serialized and transmitted one bit at a time over one output (a pair of pins). With the same transmission of an 8-bit data bus but over LVDS interface, the receiving device will not be able to determine the correct data word boundary, not knowing the position of the MSB. The deserialized data may not be the same as the original 8-bit data that was transmitted earlier. Figure 7 illustrates the possible data word alignment from a receiver for a 4-bit serial transmission. From the figure there is only 25% possibility that the alignment would be correct. Incorrect alignment will corrupt the received data.

Figure 7. Four Possible Receiver Data Word Alignment Cases for a 4-bit Serial Transmission Interface



In Cyclone series devices, the PLL locks to the rising edge of the input clock. If you have one rising edge for each serial word received, the PLL will lock to the same place within the serial word. Therefore, the easiest way for data word alignment in a Cyclone series device is to align the MSB bit to every rise edge of the clock during transmission from the source device. This is possible with the input clock frequency equal to the data bus frequency. In other words, if the input clock frequency is equal to the data rate per deserialization factor, data word alignment in a Cyclone series device used as a LVDS receiver is guaranteed without any additional setting or logic. In this setup, the word boundary at the Cyclone series receiver will be deterministic, regardless of power-up cycle or PLL reset event. Figure 8 illustrates the input clock requirement and correct data word alignment in a Cyclone series device.

Figure 8. Input Clock and Serial Received Data Relation for Correct Data Word Alignment for an 8-bit 1 Channel LVDS Interface


One rise edge per each received serial word.
Every clock rise edge is aligned to the MSB bit.
Clock frequency = data rate/deserialization factor.
External Input Clock
Internal Clock
Receiver Input Data
7
6
5
4
3
2
1
0
MSB
LSB

If you interface a Cyclone series device to another Cyclone series device using LVDS, follow the setting below for the transmitter device using the altlvds megafunction:

- Select deserialization factor value as the outclock divide factor (`B`) or just ensure that the `outclock` frequency is equal to `coreclock` frequency.

In certain interfaces, you may meet the input clock frequency requirement (where `clock=data rate/deserialization factor`) but the MSB bit is not aligned with the rising edge of the input clock. The data word boundary will be incorrect and may corrupt the received data. In such cases, use `RX_DATA_ALIGN` port (a bit-slipping feature) in the altlvds megafunction to realign the data word boundary. The `RX_DATA_ALIGN` port is a user input port based on pulse response. Hence, you need to determine the number of times to pulse it to achieve the correct data word boundary. Run timing simulation on the interface to determine how many time to pulse the `RX_DATA_ALIGN` port of the altlvds megafunction each time you power up or reset the PLL to realign the word boundary.

In cases where the input clock frequency is different than the requirement, you need to have a data checking mechanism in place. Take a case of an 800 Mbps interface and a deserialization factor of 8. For example, if the clock rate is at 200 MHz, there are two rising clock edges for each of the serial word received. In that case, the PLL could lock to either of those rising edges, and thus you would have a 50% chance of having the same word alignment from one power-up cycle to another or after a PLL reset. Design a data checking mechanism with the `RX_DATA_ALIGN` port to determine the correct word boundary before establishing the data transmission. Typically, a training sequence is used to establish the correct word boundary as commonly used in serial protocols such as SPI 4.2.

For details of ways to use the `RX_DATA_ALIGN` port in data-word alignment, refer to the *altlvds Megafunction User Guide.*

Step 4: Add Constraints

The next step in the design process is to add all timing, location and physical constraints related to the LVDS interface. This includes timing, pin locations, I/O standards and pin loading assignments.

Differential Pin Placement

Use the Quartus II Pin Planner Package view to ease differential I/O assignment planning. On the View menu, click **Show Differential Pin Pair Connections** to highlight the differential pin pairing. The differential pin pairs will be connected with red lines. For differential pins, you only need to assign the signal to a positive pin. The negative pin is assigned automatically by the Quartus II software if the positive pin is assigned with a differential I/O standard.

Internal to the FPGA, each differential pin pair has matched routing with minimal skew between the positive and the negative pins. Internal routings are matched even for pins in a pair that are not next to each other on the package. Externally the printed circuit board (PCB) designer should take care to have the same trace lengths to the receiving device to minimize skew and maximize performance.

Logic Placement

The Quartus II software automatically places the SERDES logics at the best location to meet timing requirements. Therefore, you are not required to perform placement constraints on the altlvds megafunction logics.

TCCS parameter is guaranteed per datasheet specification to the entire bank of differential I/O when the transmitter SERDES logic is placed within the logic block array (LAB) adjacent to the output pins. If you find that the TCCS performance is not satisfactory and does not meet the TCCS specification using the default Quartus II fitting, create LogicLock™ regions in the device floorplan for the transmitter SERDES logics. Constrain the transmitter SERDES logics to the LAB adjacent to the data output pins and clock output pins to improve the TCCS performance.

For step-by-step instructions on ways to create a design floorplan with LogicLock location assignments, refer to *Design and Synthesis* in volume 1 of the *Quartus II Handbook*.

Timing Constraint

The LVDS transmitter and receiver functions for Cyclone series devices with altlvds megafunction have been 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 logics using the altlvds megafunction.

The Quartus II compiler automatically ensures the associated delay chain setting are set correctly for the data path at LVDS receiver that use source-synchronous compensation mode of PLL operation. If the input clock and data are not edge or center-aligned at the receiver, timing constraints may be necessary for the Timing Analyzer tool in the Quartus II software to indicate whether the SERDES will capture the data as expected or otherwise.

Step 5: Compile Design and Verify Timing Closure

After you have made the proper constraints to the design, compile the design in the Quartus II software. Upon completion of the compilation, check the timing report in either Classic Timing Analyzer or TimeQuest Timing Analyzer to see if there is any timing violation on clock setup and clock hold paths.

The `report_tccs` and `report_rskm` commands in **TimeQuest Timing Analyzer** tool are not available for the Cyclone series devices. The commands are only available for dedicated transmitter and receiver with SERDES, as in Stratix® series devices.

For receiver design not utilizing source-synchronous mode of PLL operation or in Cyclone devices, you need to ensure proper timing constraint for the data to be captured correctly. Here are some guidelines for you to do so:

- Check the t_{su} for serial data input pin.
- Make sure clock is -90° or -180° to data at input register to maximize the read capture margin respectively for double and single-data rate operation.
- If you find the clock is not centered to the data at the input register, follow "Step 6: Adjust Constraints" on page 17 to improve the timing.

☞ The Receiver Skew Margin (RSKM) will be reduced significantly if the clock is shifted away from the center of the data window (refer to Figure 4 on page 6). Ideally, you will have a maximum RSKM margin when capture clock is centered at the data window. Determine if you have a sufficient margin as RSKM to account for board-level skews.

Step 6: Adjust Constraints

You can make adjustment to the constraints if the timing result does not fulfill the requirement or the design simply needs fine-tuning to improve margin.

In cases where the clock is shifted from the center of the data window without using source-synchronous compensated PLL, adjust the PLL phase settings of serial data `rxin` with respect to the rising edge of `rx_inclock`. Recompile the design and check if the clock is centered at the data window at data input registers.

Step 7: Determine Board Design Constraints and Perform Board Level Simulations

Once you have closed the timing for the design, examine the board design to determine how different factors can have an effect on the signal integrity, thus affecting overall timing seen at the receiving device of the LVDS interface. Time margin for the LVDS receiver (indicated by RSKM) is the timing budget allocation for board level effects, such as skew, jitter and noise. Exceeding the timing budget causes the LVDS receiver megafunction to fail.

Board-level skew (the difference in arrival time of bits transmitted at the same time for both clock and data) could arise from variation in board trace lengths, use of connectors and parasitic circuits. Jitter effects are derived from electromagnetic interference (EMI) such as cross-talk. On board resources with imperfect power supplies and reference planes may also contribute to the noise factor.

Altera recommends you to follow the board design considerations as highlighted in the *Design Guideline* section of *High-Speed Differential Interfaces* in the respective device handbook.

For PCB layout guidelines, refer to *AN 224: High-Speed Board Layout Guidelines* and *AN 315: Guidelines for Designing High-Speed FPGA PCBs.*

After determining the system requirements for the board design and finalizing the right board constraints, run board-level simulation to see if the setup is optimum. Determine if the data window meets the input specification (electrical and timing) of the LVDS receiver. Cyclone III devices support the programmable pre-emphasis feature on dedicated LVDS output buffers to compensate the frequency-dependent attenuation of the transmission line. This feature helps to maximize the data-eye opening at the far-end receiver, especially on long transmission lines.

There are various electronic design automation (EDA) simulation tools available to perform board-level simulations. Perform simulations with IBIS or HSPICE models for both the FPGA and the target LVDS interface device.

Step 8: Verify FPGA Functionality

You can obtain useful information about the LVDS interface performance with board-level verification using the FPGA prototype. While the focus here is to ensure the FPGA functionality in your end system, you can take additional steps to examine margins using oscilloscopes to verify the predicted size of the data-valid window, setup and hold margins at the I/O interface.

You can also use Altera's SignalTap® II Embedded Logic Analyzer to perform system level verification to correlate the system against your design targets.

For detailed information about using SignalTap II, refer to the *Design Debugging Using the SignalTap II Embedded Logic Analyzer* chapter in volume 3 of the *Quartus II Handbook* .

Conclusion

The Cyclone series devices support high-speed LVDS interfaces at speeds up to 840 Mbps. The altlvds megafunction in the Quartus II software provides smooth integration of the LVDS interfaces into Cyclone series devices in cost-sensitive applications. FPGA design flow and guidelines for the LVDS interface implementation provides a good out-of-the-box experience with the Cyclone series devices.

Referenced Documents

This application note references the following documents:

- *High-Speed Differential Interfaces in Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook.*
- *High-Speed Differential Interfaces in Cyclone II Devices* chapter in volume 1 of the *Cyclone II Device Handbook.*
- *High-Speed Differential Signaling in Cyclone Devices* chapter in volume 1 of the *Cyclone Device Handbook.*
- *AN 224: High-Speed Board Layout Guidelines.*
- *AN 315: Guidelines for Designing High-Speed FPGA PCBs.*
- *Design Debugging Using the SignalTap II Embedded Logic Analyzer* chapter in volume 3 of the *Quartus II Handbook.*
- *Design and Synthesis* in volume 1 of the *Quartus II Handbook.*
- *High-Speed I/O Timing* section in *High-Speed Differential Interfaces.*
- Tutorial in the Quartus II software.
- *altlvds Megafunction User Guide.*

Document Revision History

Table 3 shows the revision history for this application note.

Table 3. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
June 2008 v1.0	● Initial release.	—

101 Innovation Drive
San Jose, CA 95134
www.altera.com
Technical Support:
www.altera.com/support/
Literature Services:
literature@altera.com