Using LVDS
in APEX 20KE Devices

May 2002, ver. 1.3 Application Note 120

Introduction

Because complex designs continually demand more bandwidth, designers need a high-performance solution that offers fast data transfer and low power consumption. To address this need, Altera has incorporated LVDS technology into APEX™ 20KE devices.

LVDS is a low-voltage swing, general-purpose I/O standard that has high-speed, low-power, and low-noise advantages. Capable of extremely high data transmission rates across a variety of interconnect media, such as printed circuit board (PCB) traces, backplanes, and cables, LVDS meets today's complex design requirements for high data rates and low power consumption because it is a low-voltage differential signal that can transmit data at rates up to 840 megabits per second (Mbps). With LVDS on APEX 20KE devices, one chip can interface with high-speed, low-voltage backplanes, or data channels.

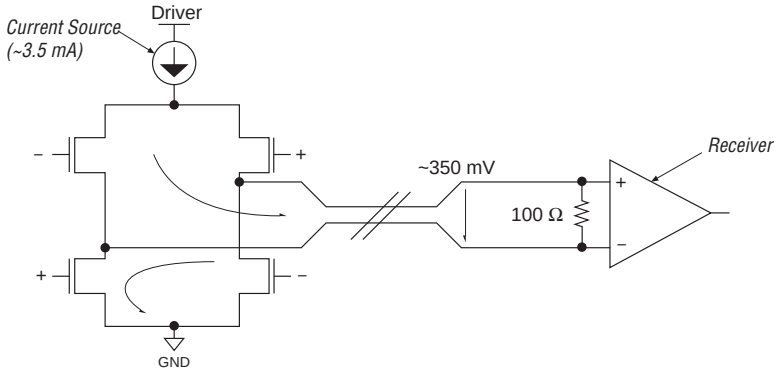
This application note explains the LVDS standard and describes how to use and benefit from LVDS-integrated APEX 20KE devices.

LVDS Standards

Two key industry standards define LVDS: IEEE Std. 1596.3 SCI-LVDS and ANSI/TIA/EIA-644. Although both standards have similar features, the IEEE Std. 1596.3 SCI-LVDS standard supports a maximum data transfer rate of only 250 Mbps. APEX 20KE devices are designed to meet the ANSI/TIA/EIA-644 standard while supporting a maximum data transfer rate of 840 Mbps.

The ANSI/TIA/EIA-644 standard defines driver output and receiver input characteristics. Figure 1 shows how the current-mode LVDS driver works.

Figure 1. LVDS Current-Mode Driver



LVDS Differential Transmission

The LVDS I/O standard utilizes a low-voltage differential data transmission scheme without requiring an input reference voltage. Differential transmission means that every LVDS signal uses two lines. The voltage difference between the two lines defines the logic state of the LVDS signal. For each signal pair, there is a true signal and a complementary signal. The differential signal is the difference between the true signal and the complementary signal (i.e., `LVDSRX01p` minus `LVDSRX01n`).

The differential transmission scheme has several key advantages over single-ended schemes:

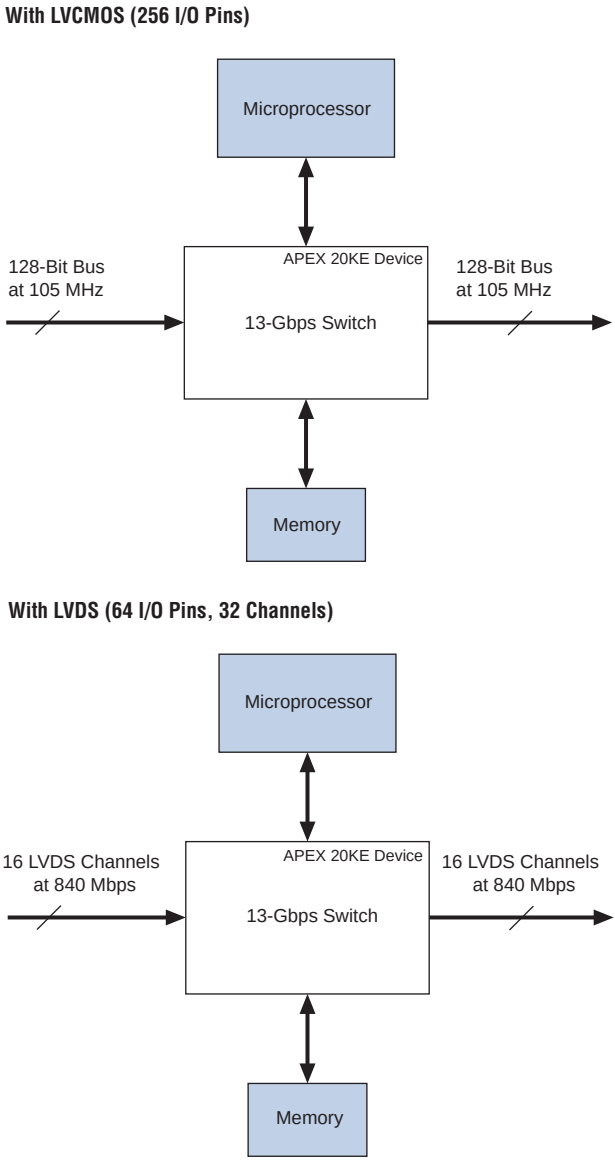
- Increased performance
- Reduced power consumption
- Minimized electromagnetic interference (EMI) generation

Increased Performance

Low-voltage swing is important for high performance; the smaller the voltage swing, the faster a signal can change logic levels. The faster the transition time (i.e., edge rate), the higher the potential data rate. To provide switching speeds in the hundreds-of-Mbps range, the LVDS standard defines a typical low-voltage signal swing of 350 mV. The two signals are referenced to each other, not to `GND` or another static signal level. Therefore, a differential standard has a much smaller switching region.

For example, the same bandwidth for a low-voltage CMOS (LVCMOS) data bus can be achieved with LVDS using one-fourth as many pins by operating the LVDS signals at eight times the data rate. Figure 2 shows a 128-bit LVCMOS data bus that can be implemented with 16 LVDS channels (32 pins).

Figure 2. LVCMOS & LVDS Performance



Power Efficiency

LVDS is a power-efficient standard. Because LVDS has a low switching voltage (typically 350 mV), the AC power dissipation per signal is small. Furthermore, the DC current is typically 3.5 mA per channel. Table 1 shows the equations that calculate the load power dissipation.

Table 1. Calculating LVDS Power Dissipation

Calculation	Equation	Example
DC power per channel (P_{DC})	$P_{DC} = V \times I$	3.3 V × 3.5 mA = 11.55 mW *(1)*
AC power per channel (P_{AC})	$P_{AC} = 2CV^2F$	2×16 pF $\times$ (0.35 V)$^2 \times$ 311 MHz = 1.219 mW
Total power	$P_{AC} + P_{DC}$	11.55 mW + 1.219 mW = 12.77 mW

Note:
(1) The voltage level is 3.3 V because the V_{CCIO} of the APEX LVDS driver is driven by a 3.3-V supply.

To understand how LVDS power consumption compares to LVTTL, consider the following example in which both LVDS and LVCMOS are operating at a 622.08-Mbps bandwidth. The comparison is shown in Table 2.

Table 2. LVDS Power Consumption Compared to LVCMOS Power Consumption

Parameter	LVDS	LVCMOS	Unit
Number of pins/channels	1 channel *(1)*	8 pins *(1)*	–
Frequency	622.08	77.76	MHz
Total bandwidth	622.08	622.08	Mbps
Data voltage swing	0.35	3.3	V
Power per pin	–	3.39	mW
Total power	12.77	27.090	mW

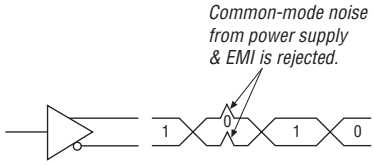
Note:
(1) 1 channel operating in ×8 mode is equivalent (in terms of data rate) to 8 LVCMOS pins.

Reduced Electromagnetic Interference (EMI)

Using the LVDS standard also provides the important advantage of reduced electromagnetic interference (EMI). EMI is radiated noise created from the acceleration of electric charge within a device and across the transmission medium between devices. Device-generated EMI is dependent on frequency, output voltage swing, and slew rate. Due to the low-voltage swing of the LVDS standard, the EMI effects are much smaller than in CMOS, TTL, or other I/O standards.

Furthermore, LVDS is less susceptible to common-mode noise because LVDS is a differential standard. Figure 3 shows that system and power supply noise is equally coupled to both LVDS signals, and thus does not affect signal quality.

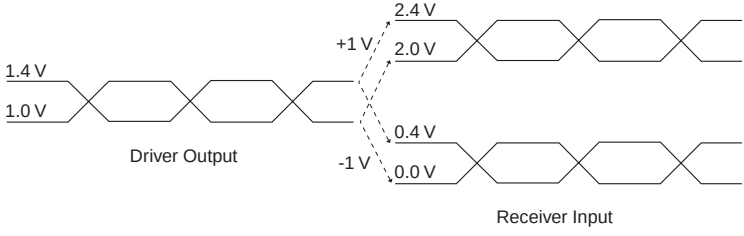
Figure 3. System-Level Noise Rejection



Common-Mode Noise Range

An LVDS receiver can tolerate a maximum of ±1-V ground shift between the driver and the receiver ground. The recommended input voltage range is from 0.0 V to 2.4 V. Because the typical voltage offset is 1.2 V, the common mode range of the receiver is 0.2 V to 2.2 V. The LVDS driver has an output voltage swing between 1.4 V and 1.0 V, with respect to ground. When there is a +1-V ground shift, the voltage swing ranges from 2.0 V to 2.4 V, which is within the input voltage range. Similarly, if there is a –1.0-V shift, the output voltage swing ranges from 0.0 V to 0.4 V. Figure 4 shows the ground shift tolerance.

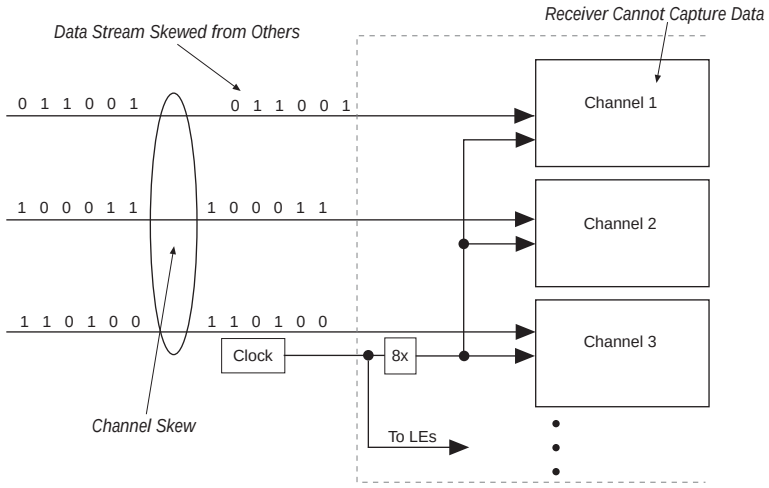
Figure 4. Common-Mode Voltage Range



Deskew Circuitry

APEX 20KE devices incorporate an optional deskew circuitry, which can be used to ensure successful data capture at high data rates. The deskew circuitry can be used to achieve high performance even with substantial board skew. The deskew circuitry can be used to increase the RSKM parameter as seen in the section "LVDS Timing" on page 9. The deskew circuitry is implemented inside the APEX 20KE device to compensate for board skew on the data channels, as shown in Figure 5.

Figure 5. Channel-to-Channel & Clock-to-Channel Skew



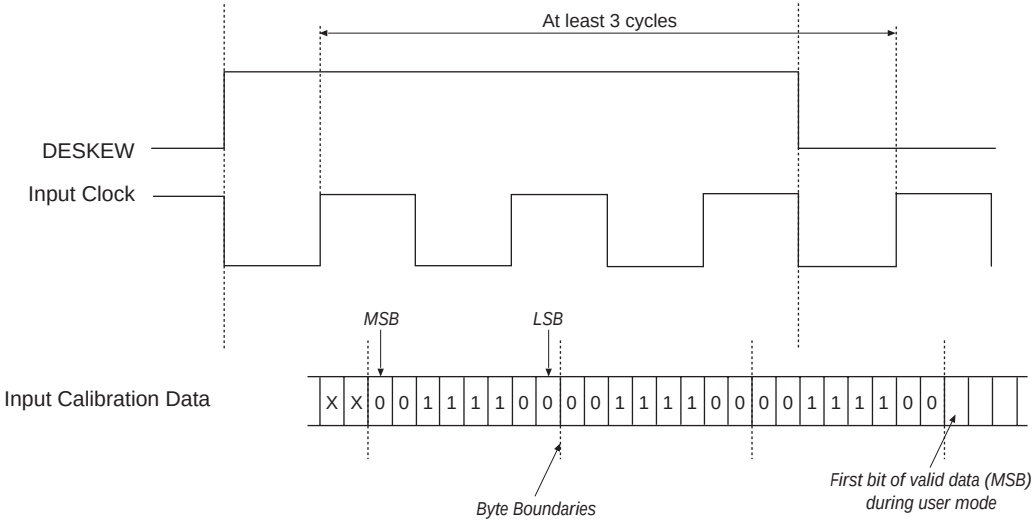
APEX 20KE devices use the deskew circuitry to provide high data transfer rates even with large channel skew on the board. An over-sampling circuit is used in the receiver to accurately capture the data. The data inputs are captured by four phases of the same clocks, and the results are examined to determine which clock successfully captured the data. The deskew circuitry can compensate for skew as much as ±25% of the time unit interval (TUI).

A calibration pattern is required to phase-align the clock with the incoming LVDS data. The calibration data values depend on the operating mode of the LVDS PLL. Like user mode data, the first bit of the calibration data is the third bit after the rising edge of the input clock. The calibration data is shown in Table 3.

Table 3. Calibration Data Pattern for Deskew Circuitry	
LVDS Operating Mode	**Calibration Pattern**
4	1100
7	0011100
8	00111100

Driving the dual-function DESKEW pin high places the LVDS inputs in calibration mode. The calibration pattern must be applied for a minimum of three input clock cycles (see Figure 6). For the device sending the deskew pattern, the deskew pin should be synchronous to the falling edge of the input clock to ensure that setup time and hold times are met. All channels are calibrated simultaneously. Each LVDS input channel can independently align the clock with the received data to account for differences in channel skew. After all channels have been successfully calibrated, the LVDS data pins are ready to transmit and receive data.

Figure 6. Deskew Circuitry Calibration Waveform for 8× Mode

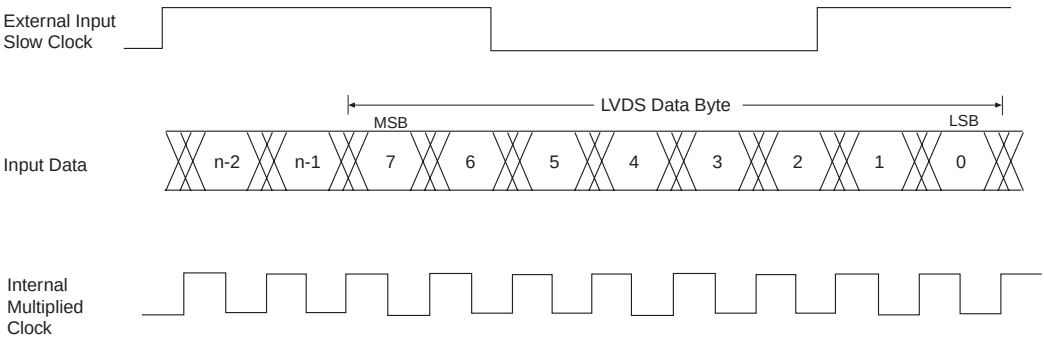


Changes in temperature and voltage can affect the receiver input skew margin (RSKM). RSKM is the tolerance of the difference between the input clock and the input data. The deskew circuitry needs to be re-calibrated often enough to ensure the skew in the system never exceeds RSKM. An analysis of the circuit must be performed to determine if the RSKM specification is violated.

Data Orientation

A set relationship exists between the external clock and the incoming data. For operation at 840 Mbps in ×8 mode, the external clock is multiplied by 8 and phase-aligned by the PLL to coincide with the sampling window of each data bit. Figure 7 shows the data bit orientation of the ×8 data conversion mode, as defined in the Quartus™ II software's `altlvds_rx` megafunction.

Figure 7. Data Bit Orientation



LVDS Timing

This section discusses the timing budget, waveforms, and specifications for LVDS in APEX 20KE devices.

LVDS allows data to be transmitted at very high speeds. This high data transmission rate results in better overall system performance. To take advantage of fast system performance, designers need to understand how to analyze timing for LVDS. LVDS timing analysis is different from traditional synchronous timing analysis techniques. Rather than focusing on clock-to-output and setup times, LVDS timing analysis is based on the skew between the data and the clock signals.

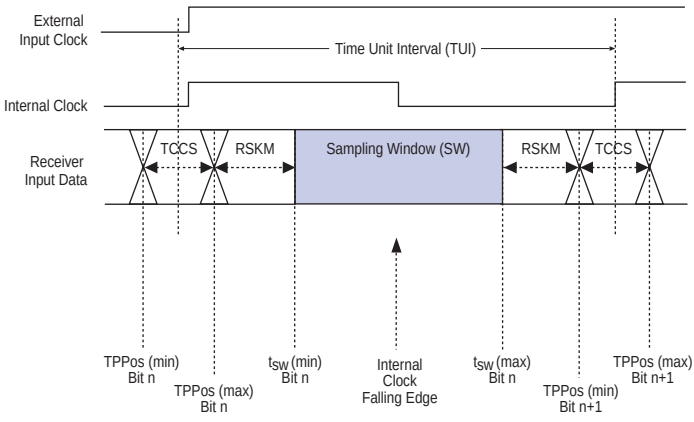
High-speed LVDS data transmission requires designers to use LVDS timing parameters provided by Altera and other LVDS vendors. Designers must consider board skew, cable skew, and clock jitter. This section defines LVDS timing parameters for APEX 20KE devices, and it explains how to use LVDS timing parameters to determine a design's maximum performance.

Timing Budget

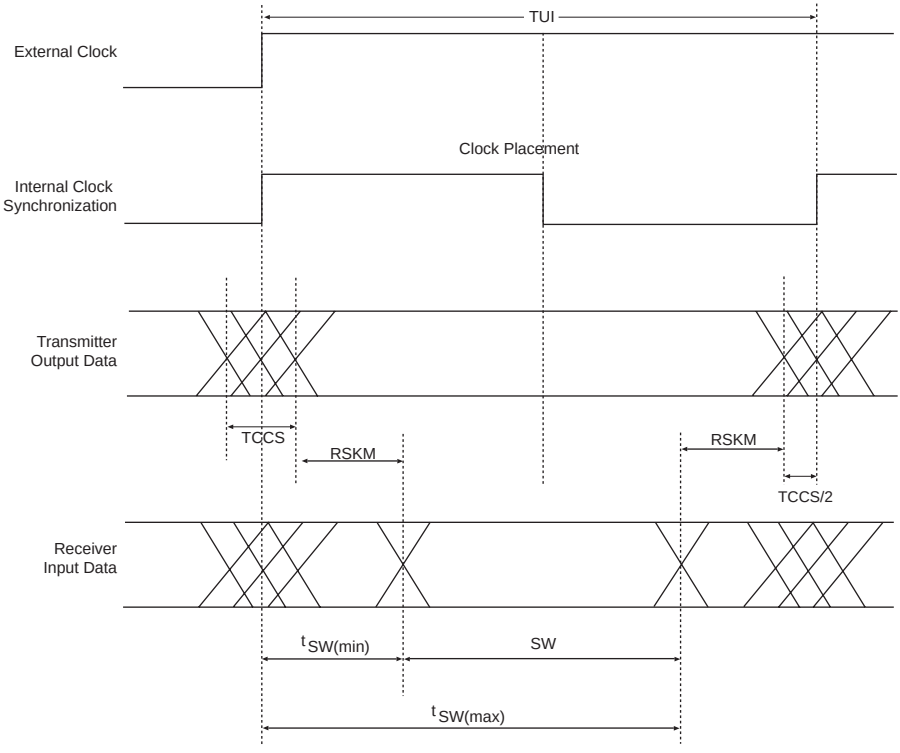
The internally generated PLL clock is positioned to meet the requirements of the timing budget. Figure 8 shows a timing diagram that includes the relationships between the LVDS timing parameters and the bit positions. Figure 8 also shows the waveforms defining the timing specifications for high-speed LVDS operation.

Figure 8. LVDS Timing Diagram & Timing Budget

Timing Diagram



Timing Budget



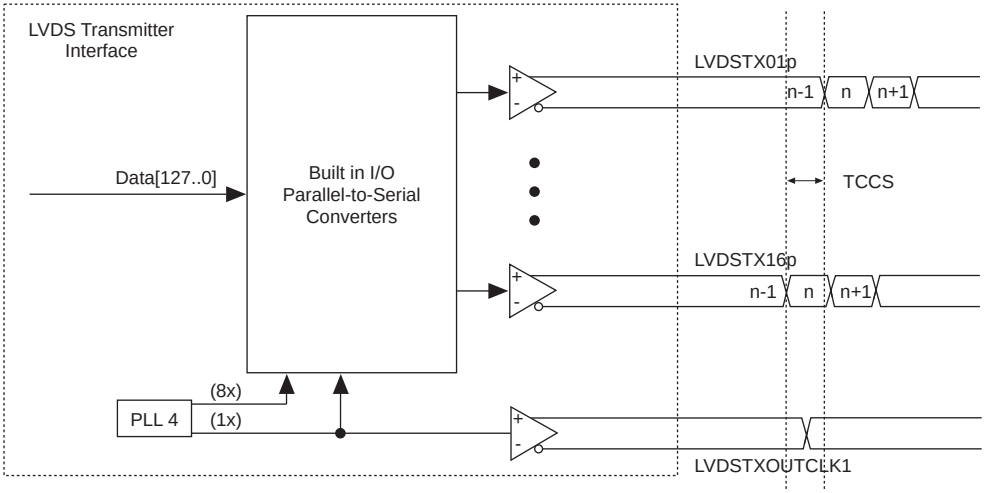
Sampling Window

The sampling window (SW) is the period of time that the input data must be stable to ensure that it can be successfully sampled by the LVDS receiver (Rx). The receiver requires the data to be stable for a period of time before it may be sampled (setup time) and it must be held for a period of time after sampling (hold time). The SW is also defined by the worst-case setup and hold times that take into account the worst-case variation in clock strobe placement.

Channel-to-Channel Skew

The channel-to-channel skew (TCCS) is the difference between the fastest and slowest data output transitions, which includes the clock-to-output (t_{CO}) variation and the clock skew of the transmitter. Skew is the variation in arrival time of two signals that are specified to arrive at the same time. Figure 9 shows a diagram of TCCS.

Figure 9. Channel-to-Channel Skew (TCCS)



Receiver Input Skew Margin

RSKM is defined by the total margin left after accounting for the sampling window and TCCS. The RSKM equation is shown below:

RSKM = (TUI – SW – TCCS) / 2

The RSKM parameter is large enough to allow for clock jitter, cable and board skew, and to provide extra margin for your design. To meet a system's requirements, designers must consider that jitter and system skew can not exceed RSKM. This equation for an application's margin is shown below:

Margin = RSKM – (input clock jitter + system skew)

System skew is the difference in propagation delays of signals between devices and includes skew introduced from cables, connectors, and differences in signal lengths on PCB traces. The input clock jitter is the jitter on the transmit clock that will be received by the APEX 20KE LVDS PLL.

LVDS Interconnect

System skew is made up of cable skew, connector skew, and PCB trace skew. A wide variety of cables and connectors for LVDS interconnect are available. Cable skew is determined by cable type, cable length, and cable quality and is normally specified in picoseconds per unit length. The longer the cable the greater the skew.

Connector skew is normally much less than cable skew. "Zero skew" connectors consist of a single row of pins that minimize the skew. The PCB traces should be routed with equal length to minimize the skew.

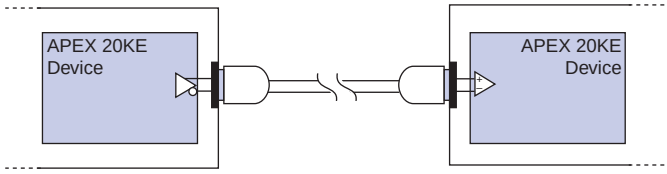
For more information on routing LVDS traces, see the *Board Design Guidelines for LVDS Systems White Paper*.

As much as possible, designers should maintain equal distance between traces in an LVDS pair. Routing the pair of traces close together will maximize the common-mode rejection ratio (CMRR).

Design Example

This section describes an LVDS design example using an APEX 20KE-to-APEX 20KE connection and a data transfer rate of 624 Mbps over a 5-m cable. This design uses a 3M Corporation cable (14526-EZ5B) and connector (10226-1A10VE). Figure 10 shows a design example of an APEX 20KE-to-APEX 20KE (specifically for EP20K400E-1X and EP20K600E-1X devices) connection with 3M cable assembly. The timing parameters used in this example are described in Table 6.

Figure 10. LVDS Design Example



The design in Figure 10 has the following characteristics:

SW = 0.44 ns, TCCS = 0.4 ns

RSKM = (TUI – SW – TCCS) / 2 = (1.6 – 0.44 – 0.4) / 2 = 380 ps

Cable skew per meter (max) = 50 ps, Connector skew (max) = 17 ps (Values obtained from 3M Corporation)

Since the LVDS balls are located on the outer edge of the FineLine BGA™ packages, the traces can be easily routed with a low skew.

PCB skew = 30 ps (based on the electrical length of the PCB traces)

System skew = Cable skew + Connector skew + PCB skew
= 50 ps / m × 5 m + 17 ps + 30 ps = 297 ps

Margin = RSKM – (input clock jitter + system skew)
= 380 ps – (10 ps + 297 ps) = 73 ps

Because there is positive margin, the circuit will operate at the required speed. If a longer cable is desired, there may not be sufficient margin. In this case, the APEX 20KE deskew circuit can be used to increase RSKM and assure circuit functionality.

When designing for high-speed data transfer rates, designers must consider various factors that affect the margin for correct data sampling. By completing the calculations described in this section, designers can calculate the margin in LVDS designs that use APEX 20KE devices and calculate LVDS transfer speed over cables and connectors. Low-skew cables and connectors can also improve margin and overall system performance. The APEX 20KE LVDS circuit provides low TCCS and SW parameters that allow high-speed LVDS data transfers.

Table 4 shows the LVDS timing specifications and terminology.

Table 4. LVDS Timing Specifications & Terminology

LVDS Timing Specification	Terminology
t_C	LVDS receiver/transmitter input and output clock period.
$f_{LVDSCLK}$	LVDS receiver/transmitter input and output clock frequency.
t_{LHT}	Low-to-high transmission time.
t_{HLT}	High-to-low transmission time.
TUI	The TUI is the timing budget allowed for skew, propagation delays, and data sampling window. (TUI = 1/(Receiver Input Clock Frequency × Multiplication Factor) = t_C/w).
f_{LVDSDR}	Maximum LVDS data transfer rate (f_{LVDSDR} = 1/TUI).
Channel-to-channel skew (TCCS)	TCCS is the timing difference between the fastest and slowest output edges, including t_{CO} variation and clock skew. The clock is included in the TCCS measurement.
Receiver input skew margin (RSKM)	RSKM is the timing margin between clock input and data input for user board design, which allows for LVDS cable skew and jitter on the LVDS PLL. RSKM = (TUI – TCCS – SW)/2.
Sampling window (SW)	This parameter defines the period of time during which the data must be valid in order to be correctly captured. The setup and hold times determine the ideal strobe position within the sampling window. SW = t_{SU} (max) – t_{SW} (min).
Input jitter (peak-to-peak)	Peak-to-peak input jitter on LVDS PLLs.
Output jitter (RMS)	RMS output jitter on LVDS PLLs.
t_{DUTY}	Duty cycle on LVDS transmitter output clock.
t_{LOCK}	Lock time for LVDS transmitter and receiver PLLs.

LVDS Timing Specifications

Tables 5 and 6 show the LVDS timing specifications as described in Figure 8 and Table 4.

Table 5. LVDS Multiplication Rate

Symbol	Description	Value	Unit
w (LVDS mode)	Width of parallel data and PLL multiplication factor	8	Integer
		7	Integer
		4	Integer

Table 6. EP20K400E & EP20K600E LVDS Timing Requirement (Part 1 of 2)

Symbol	Conditions	Commercial -1 Speed Grade			Commercial -2 Speed Grade			Industrial -2 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
t_C	×8 mode	9.52		33	11.43		33	12.80		33	ns
	×7 mode	9.52		33	11.43		33	11.43		33	ns
	×4 mode	5.71		20	6.88		20	6.88		20	ns
$f_{LVDSCLK}$	×8 mode	30		105	30		87.5	30		78.125	MHz
	×7 mode	30		105	30		87.5	30		87.5	MHz
	×4 mode	50		175	50		145.25	50		145.25	MHz
t_{LHT}	All	260	400		260	400		260	400		ps
t_{HLT}	All	260	400		260	400		260	400		ps
TUI	×8 mode	1.190	t_C/w	4.167	1.429	t_C/w	4.167	1.600	t_C/w	4.167	ns
	×7 mode	1.361	t_C/w	4.762	1.633	t_C/w	4.762	1.633	t_C/w	4.762	ns
	×4 mode	1.429	t_C/w	5	1.721	t_C/w	5	1.721	t_C/w	5	ns
f_{LVDSDR}	×8 mode (f_{LVDSDR} = 1/TUI)	240	w/t_C	840	240	w/t_C	700	240	w/t_C	625	Mbps
	×7 mode (f_{LVDSDR} = 1/TUI)	210	w/t_C	735	210	w/t_C	612.5	210	w/t_C	612.5	Mbps
	×4 mode (f_{LVDSDR} = 1/TUI)	200	w/t_C	700	200	w/t_C	581	200	w/t_C	581	Mbps
TCCS	All			400			450			450	ps
	All			400			450			450	ps
	All			400			450			450	ps
RSKM	×8 mode (no deskew at maximum f_{LVDSDR})			175			199			285	ps
	×7 mode (no deskew at maximum f_{LVDSDR})			190			301			301	ps
	×4 mode (no deskew at maximum f_{LVDSDR})			224			346			346	ps
	×8 mode (with deskew at maximum f_{LVDSDR})			473			556			685	ps
	×7 mode (with deskew at maximum f_{LVDSDR})			530			709			709	ps
	×4 mode (with deskew at maximum f_{LVDSDR})			581			776			776	ps

Table 6. EP20K400E & EP20K600E LVDS Timing Requirement (Part 2 of 2)

Symbol	Conditions	Commercial -1 Speed Grade			Commercial -2 Speed Grade			Industrial -2 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
SW	750 < f_{LVDSDR} ð 840 Mbps	440									ps
	200 ð f_{LVDSDR} ð 750 Mbps	580			580			580			ps
Input jitter (peak-to-peak)	All			2% of t_C			2% of t_C			2% of t_C	ns
Output jitter (RMS)	All			0.25% of t_C			0.25% of t_C			0.25% of t_C	ns
t_{DUTY}	All	49	50	51	49	50	51	49	50	51	%
t_{LOCK}	All			5			5			5	µs

Table 7. EP20K1000E & EP20K1500E LVDS Timing Requirement (Part 1 of 2)

Symbol	Conditions	Commercial -1 Speed Grade			Commercial -2 Speed Grade			Industrial -2 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
t_C	×8 mode	10.67		33	12.80		33	12.80		33	ns
	×7 mode	9.52		33	11.43		33	11.43		33	ns
	×4 mode	5.71		20	6.88		20	6.88		20	ns
$f_{LVDSCLK}$	×8 mode	30		93.75	30		78.125	30		78.125	MHz
	×7 mode	30		105	30		87.5	30		87.5	MHz
	×4 mode	50		175	50		145.25	50		145.25	MHz
t_{LHT}	All	260	400		260	400		260	400		ps
t_{HLT}	All	260	400		260	400		260	400		ps
TUI	×8 mode	1.333	t_C/w	4.167	1.600	t_C/w	4.167	1.600	t_C/w	4.167	ns
	×7 mode	1.361	t_C/w	4.762	1.633	t_C/w	4.762	1.633	t_C/w	4.762	ns
	×4 mode	1.429	t_C/w	5	1.721	t_C/w	5	1.721	t_C/w	5	ns
f_{LVDSDR}	×8 mode (f_{LVDSDR} = 1/TUI)	240	w/t_C	750	240	w/t_C	625	240	w/t_C	625	Mbps
	×7 mode (f_{LVDSDR} = 1/TUI)	210	w/t_C	735	210	w/t_C	612.5	210	w/t_C	612.5	Mbps
	×4 mode (f_{LVDSDR} = 1/TUI)	200	w/t_C	700	200	w/t_C	581	200	w/t_C	581	Mbps
TCCS	All			400			450			450	ps
	All			400			450			450	ps
	All			400			450			450	ps
RSKM	×8 mode (no deskew at maximum f_{LVDSDR})			177			285			285	ps
	×7 mode (no deskew at maximum f_{LVDSDR})			190			301			301	ps
	×4 mode (no deskew at maximum f_{LVDSDR})			224			346			346	ps
	×8 mode (with deskew at maximum f_{LVDSDR})			510			685			685	ps
	×7 mode (with deskew at maximum f_{LVDSDR})			530			709			709	ps
	×4 mode (with deskew at maximum f_{LVDSDR})			581			776			776	ps

Table 7. EP20K1000E & EP20K1500E LVDS Timing Requirement (Part 2 of 2)

Symbol	Conditions	Commercial -1 Speed Grade			Commercial -2 Speed Grade			Industrial -2 Speed Grade			Unit
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
SW	200 ð f_{LVDSDR} ð 750 Mbps	580			580			580			ps
Input jitter (peak-to-peak)	All			2% of t_C			2% of t_C			2% of t_C	ns
Output jitter (RMS)	All			0.25% of t_C			0.25% of t_C			0.25% of t_C	ns
t_{DUTY}	All	49	50	51	49	50	51	49	50	51	%
t_{LOCK}	All			5			5			5	µs

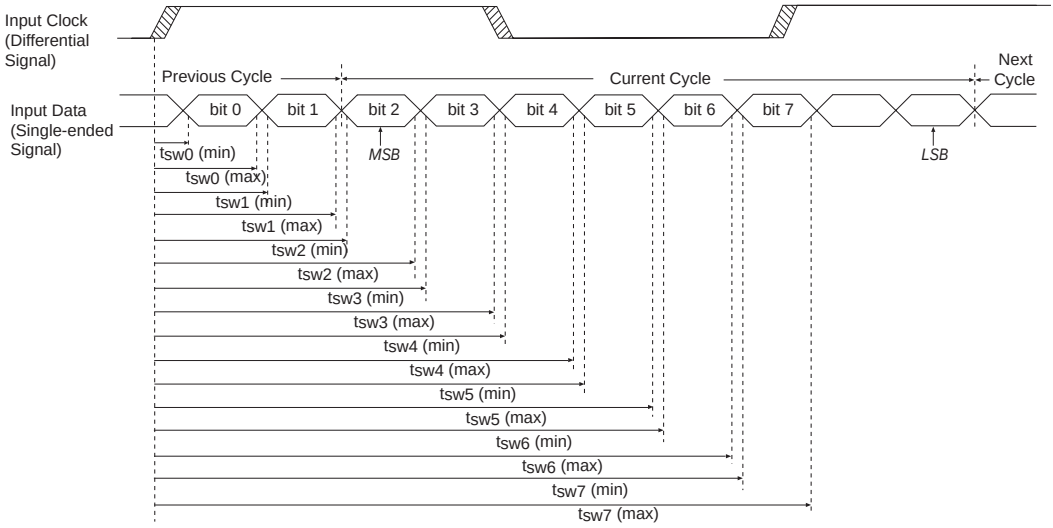
Receiver & Transmitter Data Valid Windows

For internally generated LVDS PLLs to be properly phase-aligned at the serial-to-parallel converter for data capture, the data sampling window must be properly positioned with respect to the clock.

The location of the sampling window and data transmit windows are defined in this section. They are compatible with source-synchronous LVDS buffers offered by National Semiconductor and Texas Instruments.

The input timing waveform is shown in Figure 11. To be compatible with National Semiconductor Corporation (NSC) and Texas Instruments, Inc. (TI) source synchronous LVDS devices, there is a 2-bit cycle phase delay from the input clock to the input data.

Figure 11. Input Timing Waveform *Note (1)*

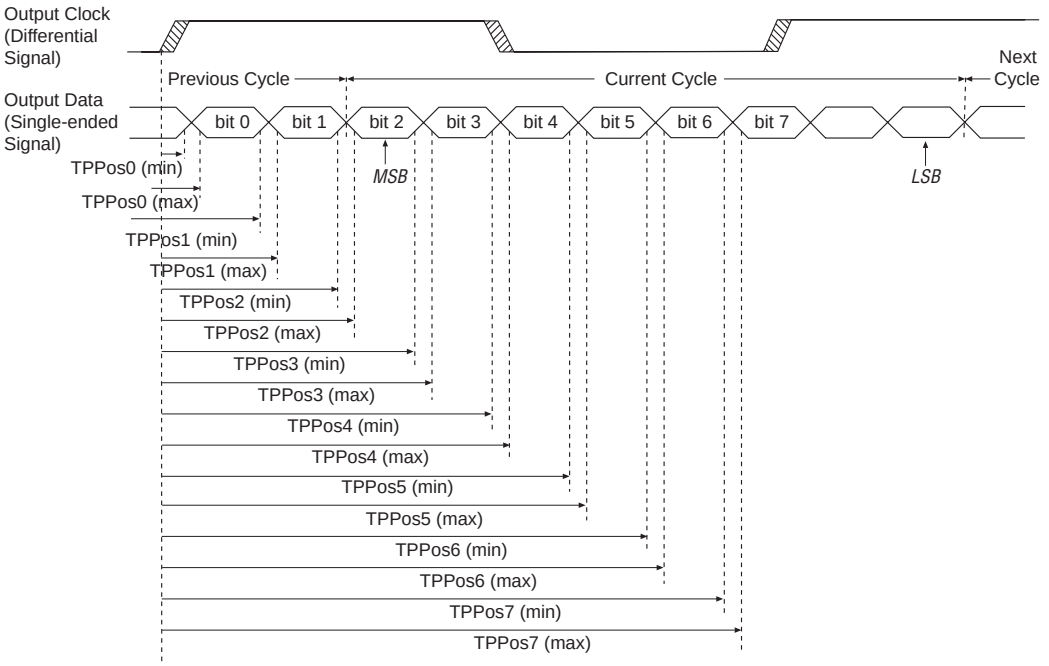


Note:

(1) The timing specifications are referenced at a 100 mV differential voltage.

The output timing waveform in Figure 12 shows the relationship between the output LVDS clock and the serial output data stream.

Figure 12. Output Timing Waveform



Note:

(1) The timing specifications are referenced at a 250 mV differential voltage.

Tables 8 through 13 define the positions of the receiver and transmitter data valid windows for maximum data rates in ×8, ×7, and ×4 LVDS modes (as shown in Figures 11 and 12). Tables 8, 10, and 12 define the sampling window for each bit with respect to the receiver input clock. Tables 9, 11, and 13 define the switching time between transmitted data bits with respect to the transmitter output clock. For the transmitter, data is valid on the transmitter data pins from TPPos[*n*] (maximum) to TPPos[*n*+1] (minimum).

There is an Automated LVDS Bit Positions Calculator on the Altera web site (**http://www.altera.com**), which will calculate the bit positions based on your specific design settings.

Table 8. LVDS Receiver Sampling Window Positions for ×8 Mode (840-Mbps Transfer)

Symbol	Parameter	Minimum	Typical	Maximum	Unit
t_{SW0}	Sampling window position for bit 0	0.38	0.60	0.82	ns
t_{SW1}	Sampling window position for bit 1	1.57	1.79	2.01	ns
t_{SW2}	Sampling window position for bit 2	2.76	2.98	3.20	ns
t_{SW3}	Sampling window position for bit 3	3.95	4.17	4.39	ns
t_{SW4}	Sampling window position for bit 4	5.14	5.36	5.58	ns
t_{SW5}	Sampling window position for bit 5	6.33	6.55	6.77	ns
t_{SW6}	Sampling window position for bit 6	7.52	7.74	7.96	ns
t_{SW7}	Sampling window position for bit 7	8.71	8.93	9.15	ns

Table 9. LVDS Transmitter Switching Characteristics for ×8 Mode (840-Mbps Transfer)

Symbol	Parameter	Minimum	Typical	Maximum	Unit
TPPos0	Transmitter output pulse position for bit 0	–0.20	0.00	0.20	ns
TPPos1	Transmitter output pulse position for bit 1	0.99	1.19	1.39	ns
TPPos2	Transmitter output pulse position for bit 2	2.18	2.38	2.58	ns
TPPos3	Transmitter output pulse position for bit 3	3.37	3.57	3.77	ns
TPPos4	Transmitter output pulse position for bit 4	4.56	4.76	4.96	ns
TPPos5	Transmitter output pulse position for bit 5	5.75	5.95	6.15	ns
TPPos6	Transmitter output pulse position for bit 6	6.94	7.14	7.34	ns
TPPos7	Transmitter output pulse position for bit 7	8.13	8.33	8.53	ns

Table 10. LVDS Receiver Sampling Window Positions for ×7 Mode (735-Mbps Transfer)

Symbol	Parameter	Minimum	Typical	Maximum	Unit
t_{SW0}	Sampling window position for bit 0	0.39	0.68	0.97	ns
t_{SW1}	Sampling window position for bit 1	1.75	2.04	2.33	ns
t_{SW2}	Sampling window position for bit 2	3.11	3.40	3.69	ns
t_{SW3}	Sampling window position for bit 3	4.47	4.76	5.05	ns
t_{SW4}	Sampling window position for bit 4	5.83	6.12	6.41	ns
t_{SW5}	Sampling window position for bit 5	7.19	7.48	7.77	ns
t_{SW6}	Sampling window position for bit 6	8.55	8.84	9.13	ns

Table 11. LVDS Transmitter Switching Characteristics for ×7 Mode (735-Mbps Transfer)

Symbol	Parameter	Minimum	Typical	Maximum	Unit
TPPos0	Transmitter output pulse position for bit 0	–0.20	0.00	0.20	ns
TPPos1	Transmitter output pulse position for bit 1	1.16	1.36	1.56	ns
TPPos2	Transmitter output pulse position for bit 2	2.52	2.72	2.92	ns
TPPos3	Transmitter output pulse position for bit 3	3.88	4.08	4.28	ns
TPPos4	Transmitter output pulse position for bit 4	5.24	5.44	5.64	ns
TPPos5	Transmitter output pulse position for bit 5	6.60	6.80	7.00	ns
TPPos6	Transmitter output pulse position for bit 6	7.96	8.16	8.36	ns

Table 12. LVDS Receiver Sampling Window Positions for ×4 Mode (700-Mbps Transfer)

Symbol	Parameter	Minimum	Typical	Maximum	Unit
t_{SW0}	Sampling window position for bit 0	0.42	0.71	1.00	ns
t_{SW1}	Sampling window position for bit 1	1.85	2.14	2.43	ns
t_{SW2}	Sampling window position for bit 2	3.28	3.57	3.86	ns
t_{SW3}	Sampling window position for bit 3	4.71	5.00	5.29	ns

Table 13. LVDS Transmitter Switching Characteristics for ×4 Mode (700-Mbps Transfer)

Symbol	Parameter	Minimum	Typical	Maximum	Unit
TPPos0	Transmitter output pulse position for bit 0	–0.20	0.00	0.20	ns
TPPos1	Transmitter output pulse position for bit 1	1.23	1.43	1.63	ns
TPPos2	Transmitter output pulse position for bit 2	2.66	2.86	3.06	ns
TPPos3	Transmitter output pulse position for bit 3	4.08	4.28	4.48	ns

LVDS Electrical Specifications

Table 14 shows the recommended operating conditions for the LVDS I/O block.

Table 14. 3.3-V LVDS I/O Specifications

Symbol	Parameter	Conditions	Min	Typ	Max	Units
V_{CCINT}	Supply voltage for internal logic and input buffers		1.71	1.80	1.89	V
V_{CCIO}	Supply voltage		3.15	3.30	3.45	V
V_{OD}	Differential output voltage	R_L = 100 Ω	250		450 *(1)*	mV
ΔV_{OD}	Change in V_{OD} between H and L	R_L = 100 Ω			50	mV
V_{OS}	Output offset voltage	R_L = 100 Ω	1.125	1.250	1.375	V
ΔV_{OS}	Change in V_{OS} between H and L	R_L = 100 Ω			50	mV
V_{TH}	Differential input threshold	V_{CM} = 1.2 V	–100		100	mV
R_L	Receiver differential input resistor		90	100	110	Ω

Note:

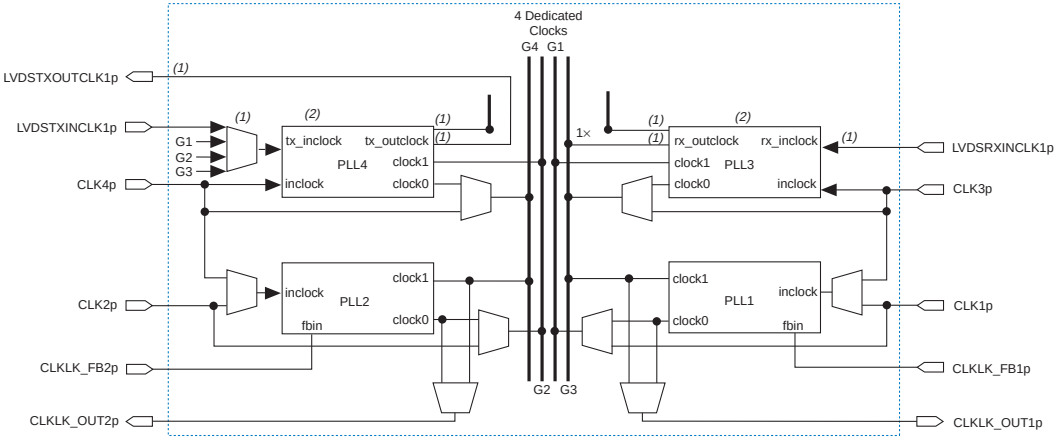
(1) Devices that do not have the "X" suffix in their ordering codes have a maximum V_{OD} of 550 mV.

Data Conversion Modes

The PLL is the key to successful transmission of the high data rates supported by LVDS. The PLL minimizes skew and phase-aligns the clock at the parallel-to-serial and serial-to-parallel data converters.

In EP20K400E and larger devices, two of the ClockLock™ PLLs can be configured for use in the LVDS I/O interfaces. One LVDS PLL is used for the input block, and another is used for the output block. Figure 13 shows a block diagram of the APEX 20KE LVDS PLLs, including LVDS-specific pin names.

Figure 13. LVDS PLL Block Diagram



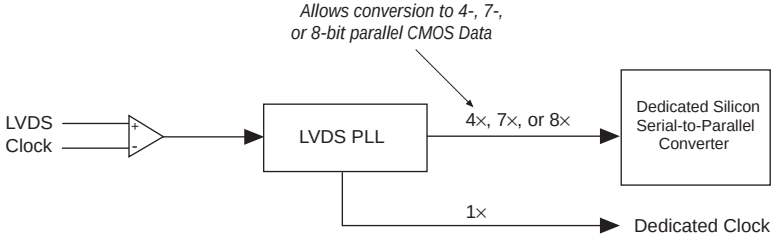
Notes:

(1) These PLL I/O ports are only used in LVDS mode.

(2) PLL3 and PLL4 can be used for either LVDS or as a general-purpose PLL.

When using LVDS, the clocks can be multiplied to support high-speed data transfer rates and convert between LVDS and CMOS data. You can multiply the input clock by 4, 7, or 8 for use in the dedicated data conversion circuitry. A general-purpose PLL is recommended for LVDS in bypass mode. Figure 14 shows the connections to the LVDS PLL on the receiver side.

Figure 14. LVDS PLL Block Diagram



An LVDS PLL is used to boost the LVDS input clock for internally clocking the LVDS data. The PLL also phase-aligns the clock with the incoming data.

The incoming serial LVDS channels can either use or bypass the serial-to-parallel converter. The parallel converter can operate in different data-conversion modes (e.g., ×8, ×7, ×4 LVDS operating modes). When operating in ×1 mode, the dedicated LVDS circuitry is bypassed, and the data directly feeds the logic elements (LEs). The 1× generated clock from the LVDS receiver PLL (PLL3) can also be used to clock internal logic within the device. Figure 15 shows the block diagram of the LVDS receiver circuitry.

Figure 15. Dedicated LVDS Receiver Circuitry Block Diagram

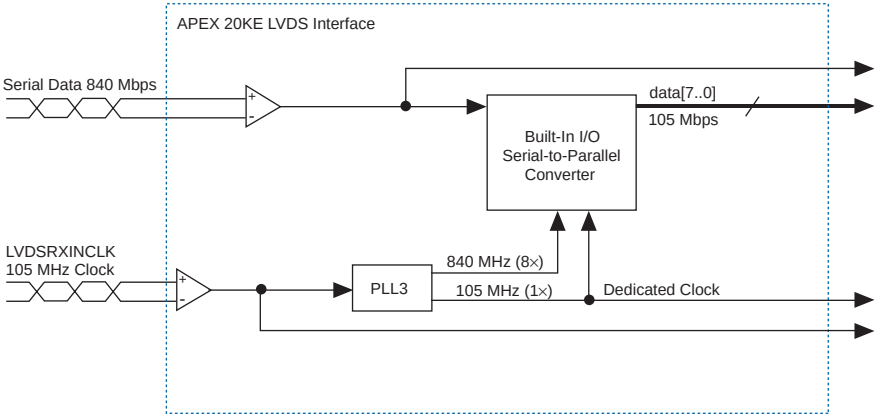
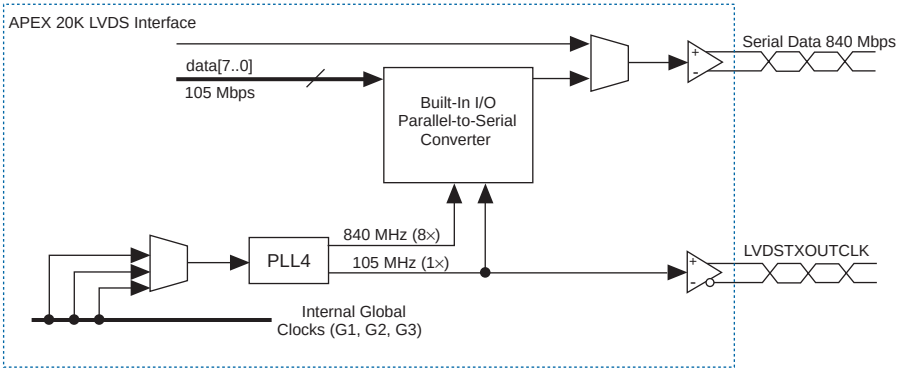


Figure 16 shows a block diagram of the LVDS transmitter circuitry. The transmitter PLL (PLL4) can be driven externally or by the output of the receiver PLL (PLL3) via an internal global line, G3, or by dedicated clock pins via G1, G2, or G3. However, the output of the general-purpose PLLs cannot drive the transmitter PLL (PLL4). The output of the transmitter PLL can be driven off-chip to clock other LVDS devices in the system.

Figure 16. Dedicated LVDS Transmitter Circuitry Block Diagram

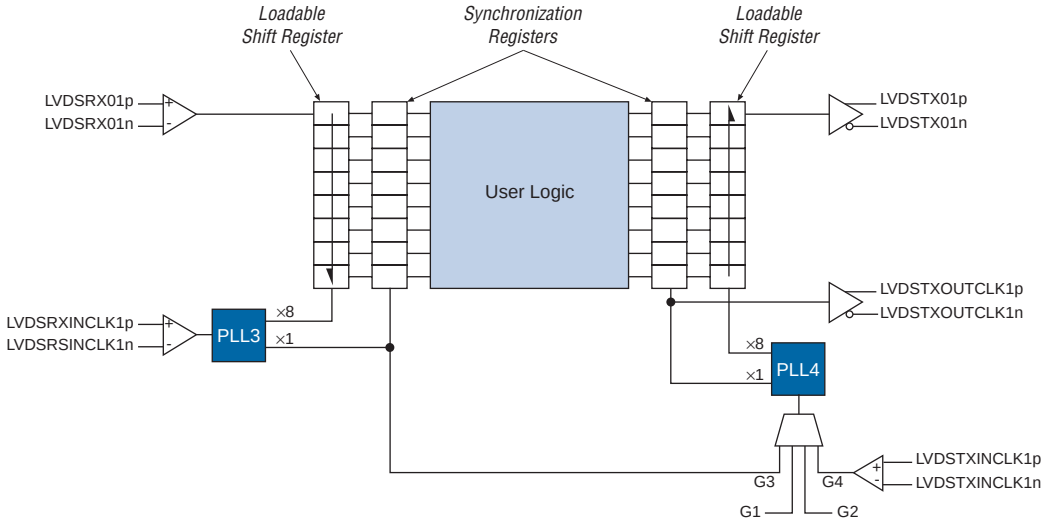


For more information on the ClockLock and ClockBoost™ circuitry, refer to *Application Note 115 (Using the ClockLock & ClockBoost PLL Features in APEX Devices)*.

LVDS Interface

Figure 17 shows how the LVDS receiver and transmitter internally interface with the logic and the other devices in the system. There are 16 input LVDS channels in the input block, with an LVDS PLL used to clock the serial-to-parallel converter in the receiver. Two PLLs (PLL3 for the receiver and PLL4 for the transmitter) generate phase-locked clock signals for the serial-to-parallel and parallel-to-serial data converters. The receiver has 16 input channels, and the transmitter has 16 output LVDS channels. The LVDS receiver converts a maximum of 16 LVDS signals into 128 parallel data bits, which feeds internal LEs within the device. Similarly, the LVDS transmitter converts a maximum of 128 on-chip data bits into 16 LVDS data streams, using an 8-to-1 parallel-to-serial converter.

Figure 17. LVDS Receiver & Transmitter Interface



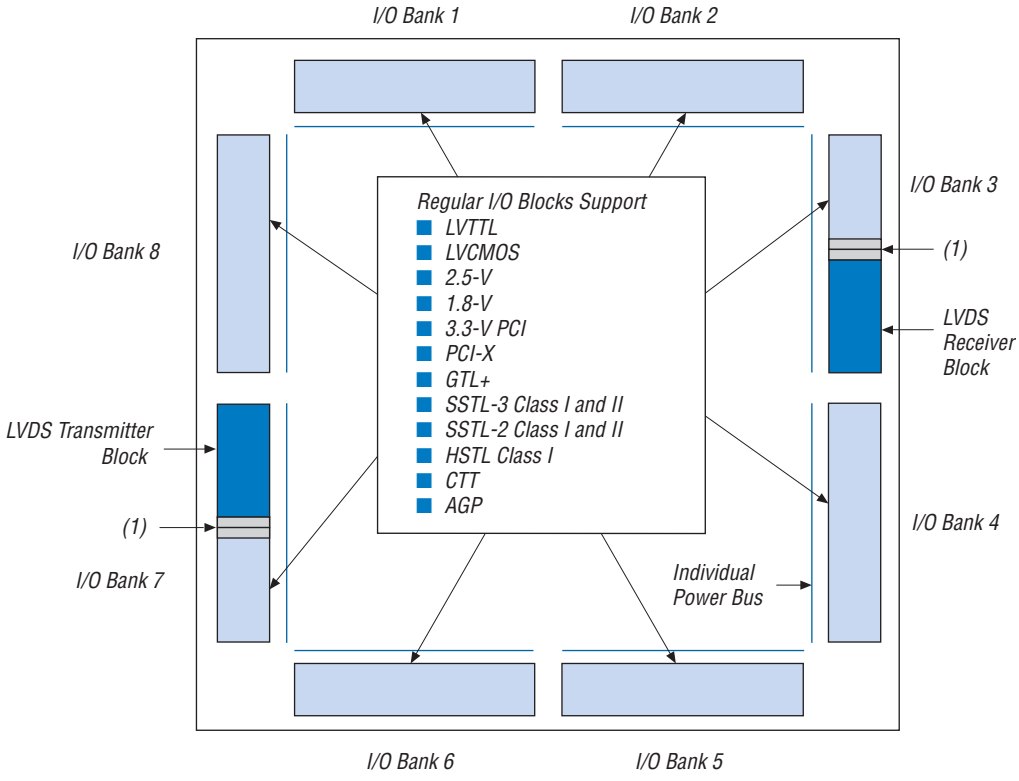
The internal LVDS PLL clocks have an 8× maximum multiplication rate. The LVDS transmitter has the ability to drive the 1× locked PLL clock off-chip. The external transmitter clock output and output data signals are in-phase. Every cycle of transmit and receive clock data—up to 128 bits of input and output data—are sampled via the 16 LVDS I/O channels.

The LVDS input pins are row pins located on the right side of the device. Each LVDS receiver channel interfaces with dedicated shift registers and drives row lines. Similarly, the LVDS output pins are also row pins located on the left side of the device. Each LVDS transmitter channel interfaces with dedicated shift registers, driven by LEs in the adjacent MegaLAB structure.

APEX 20KE I/O Structure

APEX 20KE devices have eight programmable I/O banks and two dedicated LVDS I/O blocks. Figure 18 shows a representation of the APEX 20KE I/O banks and LVDS blocks. The LVDS receiver block is located on the right, and the transmitter block is located on the left.

Figure 18. APEX 20KE I/O Banks



Note:
(1) For more information on placing I/O pins within LVDS blocks, see "Guidelines for Using LVDS Blocks" on page 67.

All EP20K400E and larger devices support using LVDS on dedicated clock signals and LVDS data in bypass (×1) mode. EP20K400E and larger devices with an "X" suffix in their ordering codes support LVDS in ×8, ×7, and ×4 modes. EP20K300E devices support using LVDS on dedicated clock signals and LVDS data in bypass (×1) mode in the 652-pin ball-grid array (BGA) and 672-pin FineLine BGA packages.

EP20K200E and smaller devices support using LVDS on dedicated clock signals. For the EP20K300E and smaller devices, the "X" suffix indicates that the device includes PLLs. All APEX 20KE devices, including devices without an "X" suffix in their ordering codes, support LVDS on dedicated clocks.

LVDS support is summarized in Table 15.

Table 15. LVDS Support in APEX 20KE Devices

Device Density	Feature	Devices with PLLs	Devices without PLLs
EP20K200E and smaller	LVDS Clock	Input, output, and feedback	Input
	LVDS I/O pins	Not supported	Not supported
EP20K300E	LVDS Clock	Input, output, and feedback	Input
	LVDS I/O pins	1× mode	1× mode
EP20K400E and larger	LVDS Clock	Input, output, and feedback	Input
	LVDS I/O pins	All modes	1× mode

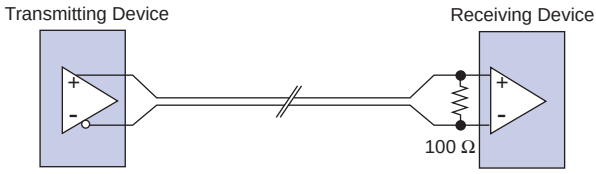
The LVDS transmitter and receiver blocks support all of the I/O standards and can be used as input, output, or bi-directional pins at 3.3 V, 2.5 V, and 1.8 V. The first two I/O pins that border the LVDS blocks are input only to maintain an acceptable noise level on the internal V_{CCIO} supply.

The programmable I/O element (IOE) blocks have individual power planes with separate VCCIO pins for each I/O bank. The V_{CCIO} planes support 3.3-V, 2.5-V, and 1.8-V levels. If the I/O pins are used for LVDS I/O standards, always connect the LVDS power bus-associated VCCIO pins to 3.3 V. When not used for LVDS, the two LVDS I/O blocks support all of the standards supported by APEX 20KE devices.

Board Termination

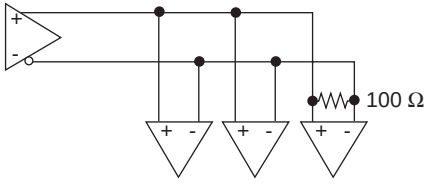
The LVDS I/O standard requires a termination resistor between the signals at the receiver side. This termination resistor generates the differential output voltage (V_{OD}) at the receiver input. The termination resistor should match the differential load impedance of the bus (typically 100 Ω, but the values may range between 90 Ω and 110 Ω). Figure 19 shows LVDS board termination at the receiver.

Figure 19. LVDS Board Termination at the Receiver



For multi-drop configurations where one transmitter drives multiple receivers, only one termination resistor is used, and it should be placed at the furthest receiver from the transmitter device, as shown in Figure 20.

Figure 20. Multi-Drop Configuration Termination



LVDS Design Guidelines

Because of the high data transmission rates used with LVDS, skew can be a problem. To prevent skew and maintain signal integrity, follow the recommendations below:

- Keep stub lengths less than 12 mm (0.5 inch).
- Place drivers and receivers as close to connectors as possible.
- Match the electrical lengths of all bus LVDS lines.
- Minimize the distance between traces of a pair of LVDS lines to maximize Common Mode Rejection Ratio (CMRR).
- Separate TTL / CMOS signals from LVDS signals onto different board layers.
- Use a multi-layer PCB with a ground plane beneath the LVDS bus lines.
- Avoid 90 degree PCB bends and multiple vias. Use the same number of bends and vias for each signal pair to match delays.
- Use good decoupling techniques. Use four surface-mount bypass capacitors (2.2 µF, 0.1 µF, 0.01 µF, and 0.001 µF) placed close to the `GND_CKLK3` / `VCC_CKLK3` and `GND_CKLK4` / `VCC_CKLK4` pairs to eliminate switching noise. These capacitors can be vertically stacked to conserve board space.
- Place a parallel termination resistor at the receiver input.
- Match trace differential impedance with load impedance (100 Ω).

Software

This section provides information on implementing the LVDS I/O standard in the Quartus II software.

LVDS can be easily implemented in APEX devices using the Quartus II software and the `altlvds` megafunction, saving design time and reducing board space.

LVDS Paired Pin Labeling

Information on the dual-purpose paired LVDS pins are displayed in the same text string as the other information on a pin, similar to other pins that have secondary functions such as `INIT_DONE`. For example, in Figure 21 on page 33, Row I/O is shown as Row I/O, `LVDSRXINCLK1p`.

LVDS pins have a specific naming convention: all LVDS pin names begin with LVDS. The next two characters for data pins indicate whether they belong to the receiver (RX) or transmitter (TX), followed by the two-digit channel <*number*> which ranges from 01 to 16. The last character at the end of the pin name indicates polarity; p for positive polarity and n for negative polarity.

Table 16 summarizes all of the LVDS pin names.

Table 16. LVDS Pin Naming Convention (Part 1 of 2)

Pin Names	Function
LVDSRX<*number*>p	Receiver positive data pin
LVDSRX<*number*>n	Receiver negative data pin
LVDSTX<*number*>p	Transmitter positive data pin
LVDSTX<*number*>n	Transmitter negative data pin
LVDSRXINCLK1p	Receiver input clock positive pin
LVDSRXINCLK1n	Receiver input clock negative pin
LVDSTXINCLK1p	Transmitter input clock positive pin
LVDSTXINCLK1n	Transmitter input clock negative pin
LVDSTXOUTCLK1p	Transmitter output clock positive pin
LVDSTXOUTCLK1n	Transmitter output clock negative pin
CLK1p	Dedicated clock 1 positive pin (PLL 1)
CLK1n	Dedicated clock 1 negative pin (PLL 1)
CLK2p	Dedicated clock 2 positive pin (PLL 2)
CLK2n	Dedicated clock 2 negative pin (PLL 2)
CLK3p	Dedicated clock 3 positive pin (PLL 3)
CLK3n	Dedicated clock 3 negative pin (PLL 3)
CLK4p	Dedicated clock 4 positive pin (PLL 4)

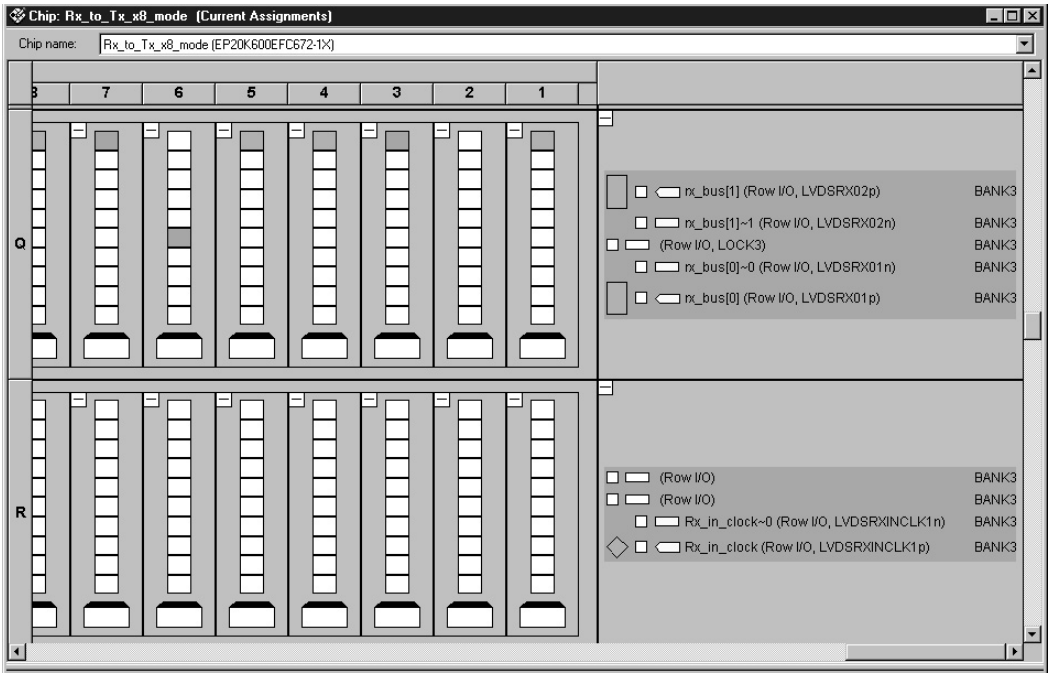
Table 16. LVDS Pin Naming Convention (Part 2 of 2)

Pin Names	Function
CLK4n	Dedicated clock 4 negative pin (PLL 4)
CLKLK_FB1p	Dual-purpose ClockLock feedback positive pin (PLL 1)
CLKLK_FB1n	Dual-purpose ClockLock feedback negative pin (PLL 1)
CLKLK_FB2p	Dual-purpose ClockLock feedback positive pin (PLL 2)
CLKLK_FB2n	Dual-purpose ClockLock feedback negative pin (PLL 2)
CLKLK_OUT1p	Dual-purpose ClockLock output positive pin (PLL 1)
CLKLK_OUT1n	Dual-purpose ClockLock output negative pin (PLL 1)
CLKLK_OUT2p	Dual-purpose ClockLock output positive pin (PLL 2)
CLKLK_OUT2n	Dual-purpose ClockLock output negative pin (PLL 2)

The dedicated clock pins (CLK1p, CLK2p, CLK3p, CLK4p) support LVDS and have optional dual-purpose negative polarity pins associated with them. The PLL feedback pins (CLKLK_FB1p, CLKLK_FB2p) and the PLL output pins (CLKLK_OUT1p, CLKLK_OUT2p) also support LVDS following the same convention as the dedicated clock pins.

Figure 21 shows the LVDS receiver in the Floorplan Editor. The receiver data channel, represented by LVDSRX01p and LVDSRX01n, feeds the dedicated serial-to-parallel converter. The LVDS clock (LVDSRXINCLK1p, LVDSINCLK1n) clocks the serial-to-parallel converter. The serial-to-parallel converter is shown by the filled rectangle adjacent to the IOE register associated with each positive polarity LVDS data and clock pin.

Figure 21. LVDS Receiver in the Floorplan Editor



LVDS Bit Positions

Data synchronization is necessary for successful data transmission at high frequencies. Figure 22 shows the data bit orientation for a receiver channel operating in ×8 mode. Unlike in calibration mode, the first bit of data for the current clock cycle is the third bit because the first two bits belong to the previous cycle. Similar positioning exists for the most significant bits (MSBs) and least significant bits (LSBs) after deserialization, as seen in Table 17.

Figure 22. Bit Order for One Channel of LVDS Data

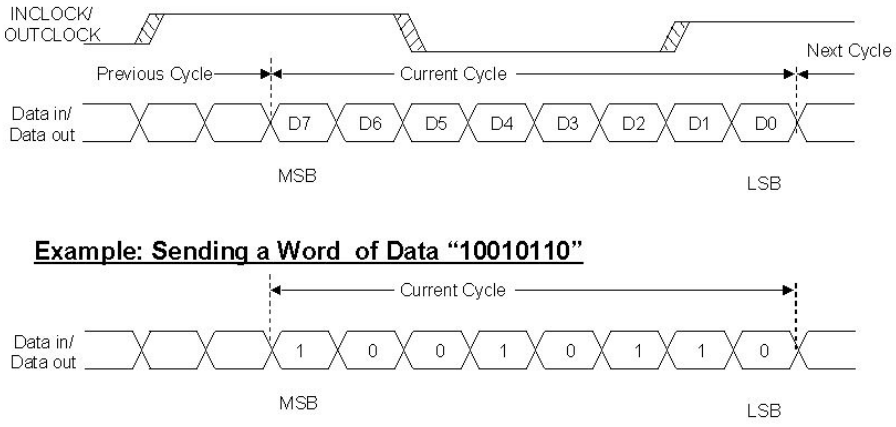


Table 17 shows the conventions for LVDS bit naming.

Table 17. LVDS Bit Naming

Receiver Data Channel Number	Internal CMOS 8-bit Parallel Data	
	MSB Position	LSB Position
1	7	0
2	15	8
3	23	16
4	31	24
5	39	32
6	47	40
7	55	48
8	63	56
9	71	64
10	79	72
11	87	80
12	95	88
13	103	96
14	111	104
15	119	112
16	127	120

The altlvds Megafunction

Figures 23 and 24 show the symbols for the `altlvds` megafunction transmitter and receiver, respectively. Each module represents the dedicated LVDS silicon present in APEX 20KE devices as well as the dedicated LVDS PLLs that are present for clock generation. A single module represents either one or multiple LVDS channels.

Figure 23. The altlvds Megafunction Transmitter Module Symbol

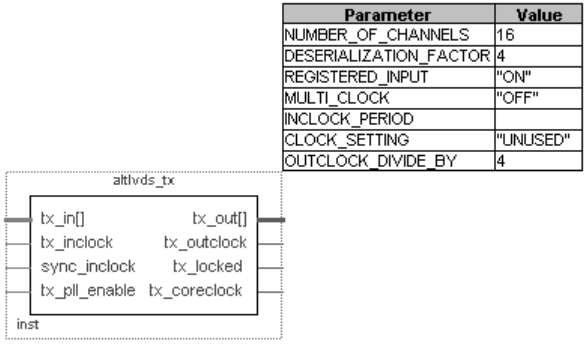
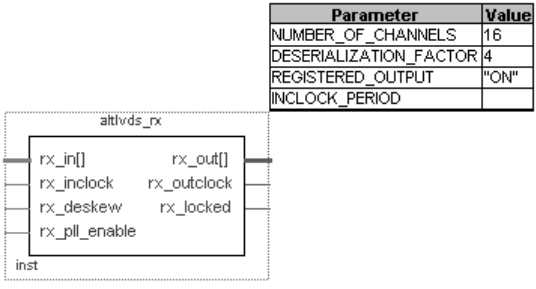


Figure 24. The altlvds Megafunction Receiver Module Symbol



Figures 25 through 28 show the AHDL Function Prototype (port name and order also apply to Verilog HDL) and VHDL Component Declaration sample scripts for both the LVDS transmitter and receiver.

Figure 25. AHDL Function Prototype (Transmitter)

```
FUNCTION altlvds_tx (tx_in[DESERIALIZATION_FACTOR*NUMBER_OF_CHANNELS-
1..0], tx_inclock, sync_inclock, tx_pll_enable)
   WITH (NUMBER_OF_CHANNELS, DESERIALIZATION_FACTOR, REGISTERED_INPUT,
MULTI_CLOCK, INCLOCK_PERIOD, OUTCLOCK_DIVIDE_BY, INTENDED_DEVICE_FAMILY)
   RETURNS (tx_out[NUMBER_OF_CHANNELS-1..0], tx_outclock, tx_locked);
```

Figure 26. VHDL Component Declaration (Transmitter)

```
COMPONENT altlvds_tx
   GENERIC (NUMBER_OF_CHANNELS: NATURAL;
      DESERIALIZATION_FACTOR: NATURAL;
      OUTCLOCK_DIVIDE_BY: NATURAL:= 0;
      REGISTERED_INPUT: STRING:= "ON";
      MULTI_CLOCK: STRING:= "OFF";
      INCLOCK_PERIOD: NATURAL);
      CLOCK_SETTING: STRING:= "UNUSED");
      INTENDED_DEVICE_FAMILY: STRING:= “APEX 20KE”);
    PORT (tx_in: IN
STD_LOGIC_VECTOR(DESERIALIZATION_FACTOR*NUMBER_OF_CHANNELS-1 DOWNTO 0);
      tx_inclock: IN STD_LOGIC;
      sync_inclock: IN STD_LOGIC:= '0';
      tx_pll_enable: IN STD_LOGIC:= '1'
      tx_out: OUT STD_LOGIC_VECTOR(NUMBER_OF_CHANNELS-1 DOWNTO 0);
      tx_outclock, tx_locked: OUT STD_LOGIC);
END COMPONENT;
```

Figure 27. AHDL Function Prototype (Receiver)

```
FUNCTION altlvds_rx (rx_in[NUMBER_OF_CHANNELS-1..0], rx_inclock,
rx_deskew, rx_pll_enable)
   WITH (NUMBER_OF_CHANNELS, DESERIALIZATION_FACTOR, REGISTERED_OUTPUT,
INCLOCK_PERIOD, INTENDED_DEVICE_FAMILY)
   RETURNS (rx_out[DESERIALIZATION_FACTOR*NUMBER_OF_CHANNELS-1..0],
rx_outclock, rx_locked);
```

Figure 28. VHDL Component Declaration (Receiver)

```
COMPONENT altlvds_rx
   GENERIC(NUMBER_OF_CHANNELS: NATURAL;
      DESERIALIZATION_FACTOR: NATURAL;
      REGISTERED_OUTPUT: STRING := "ON";
      INCLOCK_PERIOD: NATURAL;
      CLOCK_SETTING: STRING := "UNUSED");
      INTENDED_DEVICE_FAMILY: STRING:= “APEX20KE”);
   PORT (rx_in: IN STD_LOGIC_VECTOR(NUMBER_OF_CHANNELS-1 DOWNTO 0);
      rx_inclock: IN STD_LOGIC;
      rx_deskew: IN STD_LOGIC := '0';
      rx_pll_enable: IN STD_LOGIC:= '1'
      rx_out: OUT
STD_LOGIC_VECTOR(DESERIALIZATION_FACTOR*NUMBER_OF_CHANNELS-1 DOWNTO 0);
      rx_outclock, rx_locked: OUT STD_LOGIC);
END COMPONENT;
```

The altlvds megafunction input and output ports are described in Tables 18 and 19, respectively. Table 20 lists the parameters that are used to configure the altlvds megafunction.

Table 18. Input Ports of the altlvds Megafunction

Port Name	Required	Description	Notes
LVDS Transmitter Input Ports			
tx_in[]	Yes	Input data	Input port [(DESERIALIZATION_FACTOR) × (NUMBER_OF_CHANNELS-1..0)] wide.
tx_inclock	Yes	LVDS reference input clock	The LVDSTXINCLK pin cannot clock LE registers; the Quartus II software will return an error if the LVDSTXINCLK tries to clock LE registers. When using the LVDSTXINCLK in ×7 or ×8 mode, set the MULTI_CLOCK parameter to ON.
sync_inclock	No	Optional clock for the input registers	If the MULTI_CLOCK parameter is turned on, you must use the sync_inclock port.
tx_pll_enable	No	Enable control for the LVDS PLL	
rx_in[]	Yes	LVDS input data channel	Input port [NUMBER_OF_CHANNELS–1..0] wide.
rx_inclock	Yes	LVDS reference input clock	
rx_deskew	No	Specifies whether to activate calibration mode	For more information on the rx_deskew port, contact Altera Applications.
rx_pll-enable	No	Enable control for the LVDS PLL	

Table 19. Output Ports of the altlvds Megafunction

Port Name	Required	Description	Notes
LVDS Transmitter Output Ports			
tx_out[]	Yes	Serialized LVDS data signal	Output port [NUMBER_OF_CHANNELS–1..0] wide.
tx_outclock	No	External reference clock	
tx_locked	No	Gives the status of the LVDS PLL	When the PLL is locked, this signal is VCC. When the PLL fails to lock, this signal is GND.
LVDS Receiver Output Ports			
rx_out[]	Yes	Deserialized data signal	Output port [(DESERIALIZATION_FACTOR) × (NUMBER_OF_CHANNELS-1..0)] wide.
rx_outclock	No	Internal reference clock	This signal can only be used to clock internal logic.
rx_locked	No	Gives the status of the LVDS PLL	When the PLL is locked, this signal is V_{CC}. When the PLL fails to lock, this signal is GND.

Table 20. The altlvds Megafunction Parameters (Part 1 of 2)

Parameter	Type	Required	Description
LVDS Transmitter Parameters			
NUMBER_OF_CHANNELS	Integer	Yes	Specifies the number of LVDS channels.
DESERIALIZATION_FACTOR	Integer	Yes	Specifies the number of bits per channel. Values are 8, 7, or 4.
REGISTERED_INPUT	String	No	Indicates whether the tx_out[] and tx_in[]port should be registered. Values are "ON" and "OFF." If omitted the default is "ON."
MULTI_CLOCK	String	No	Indicates whether the sync_inclock port is used for input registering. Values are "ON" and "OFF." If omitted the default is "OFF."
INCLOCK_PERIOD	String	Yes	Specifies the period or frequency of the input clocks. The default time unit is ps.

Table 20. The altlvds Megafunction Parameters (Part 2 of 2)

Parameter	Type	Required	Description
LVDS Transmitter Parameters			
OUTCLOCK_DIVIDE_BY	Integer	No	Specifies the period 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. For APEX 20KE devices, OUTCLOCK_DIVIDE_BY = DESERIALIZATION_FACTOR
INTENDED_DEVICE_FAMILY	String	No	This parameter is used for modeling and behavioral simulation purposes. Create the PLL with the MegaWizard Plug-in Manager to calculate the value for this parameter.
LVDS Receiver Parameters			
NUMBER_OF_CHANNELS	Integer	Yes	Specifies the number of LVDS channels.
DESERIALIZATION_FACTOR	Integer	Yes	Specifies the number of bits per channel. Values are 8, 7, or 4.
REGISTERED_OUTPUT	String	No	Indicates whether the rx_out[] port should be registered. Values are "ON" and "OFF." If omitted, the default is "ON."
INCLOCK_PERIOD	String	Yes	Specifies the period or frequency of the rx_inclock port. The default time unit is ps.
INTENDED_DEVICE_FAMILY	String	No	This parameter is used for modeling and behavioral simulation purposes. Create the PLL with the MegaWizard Plug-in Manager to calculate the value for this parameter.

The MegaWizard Interface

The MegaWizard® interface allows users to customize the LVDS megafunction. The MegaWizard Plug-In Manager automatically generates the following files:

- Component Declaration File (**.cmp**) that can be used in VHDL Design Files (**.vhd**)
- Include File (**.inc**) that can be used in Text Design Files (**.tdf**) and Verilog HDL Design Files (**.v**)
- Quartus II Block Symbol File (**.bsf**) that can be used in Quartus II Block Design Files (**.bdf**)
- Custom Megafunction variation file (TDF, VHD, or V file)

The MegaWizard Plug-In Manager can be invoked in two ways:

- Choosing the **MegaWizard Plug-In Manager** command from the Tools menu, as seen in Figure 29.
- Selecting **MegaWizard Plug-In Manager** from the **Symbol** dialog box in the Block Editor, as seen in Figure 30.

Figure 29. Invoking the MegaWizard Plug-In Manager from the Tools Menu

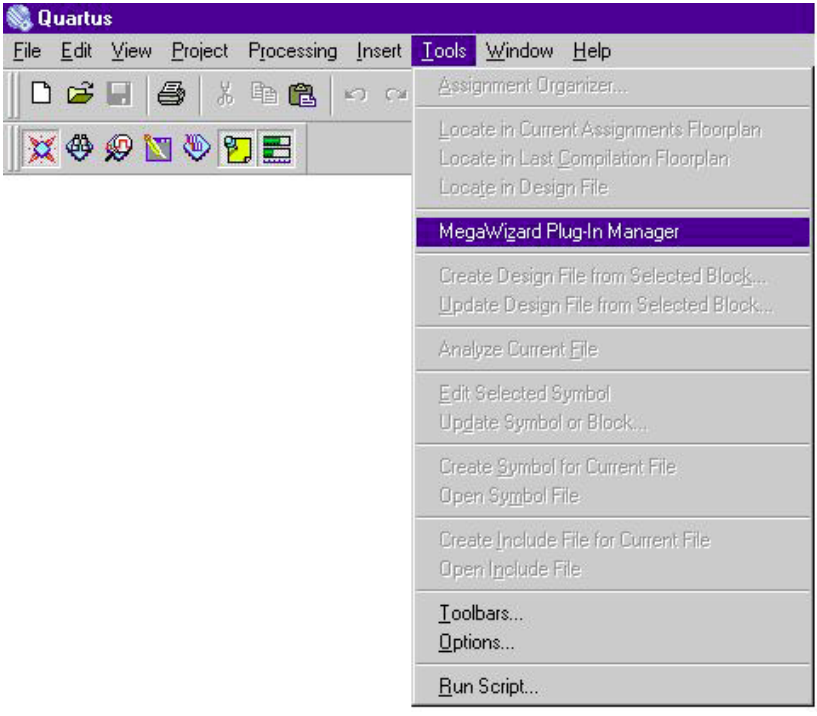
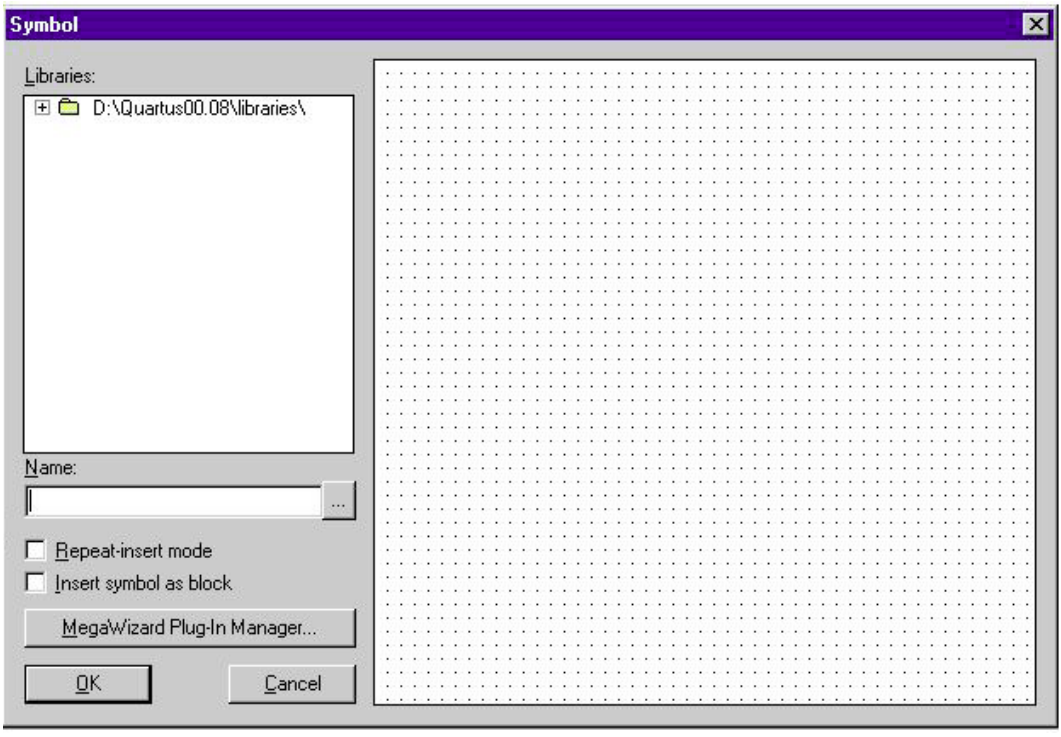


Figure 30. Invoking the MegaWizard Plug-In Manager from the Block Editor



The MegaWizard Plug-In Manager takes a step-by-step approach to generating customized LVDS transmitter and receiver modules. Each page of the MegaWizard Plug-In Manager allows the user to select from a set of customizable features that tailors the modules to the needs of the design.

Figure 31 displays the third page of the `altlvds` megafunction in the MegaWizard Plug-In Manager when instantiating an LVDS transmitter. Figure 32 shows the third page for a receiver instantiation. These pages allow the user to customize the LVDS transmitter and receiver modules.

Figure 31. Page 3 of the altlvds Transmitter MegaWizard Plug-In Manager

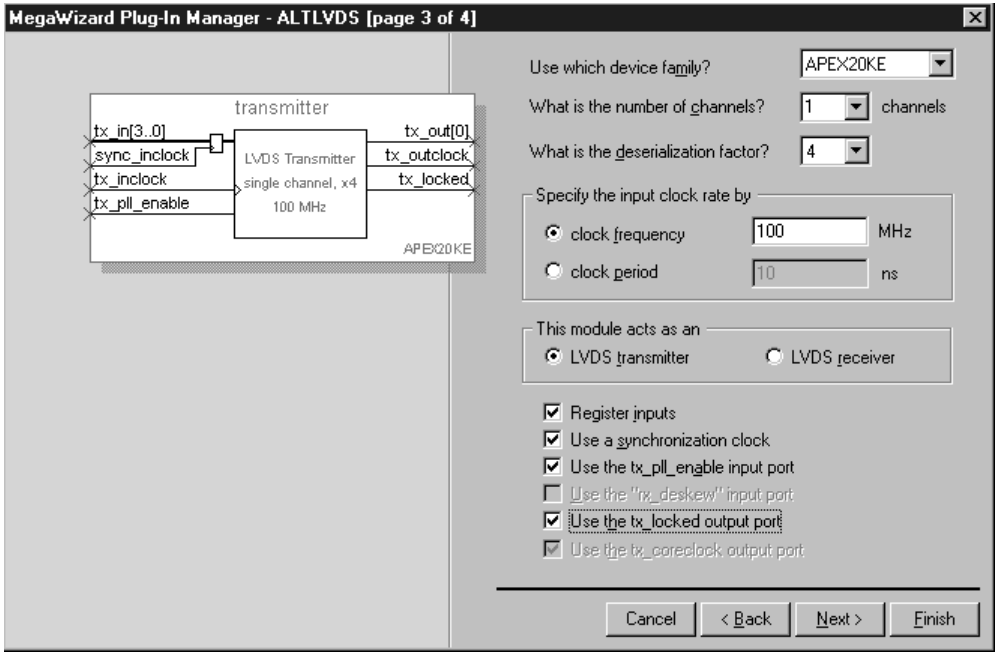
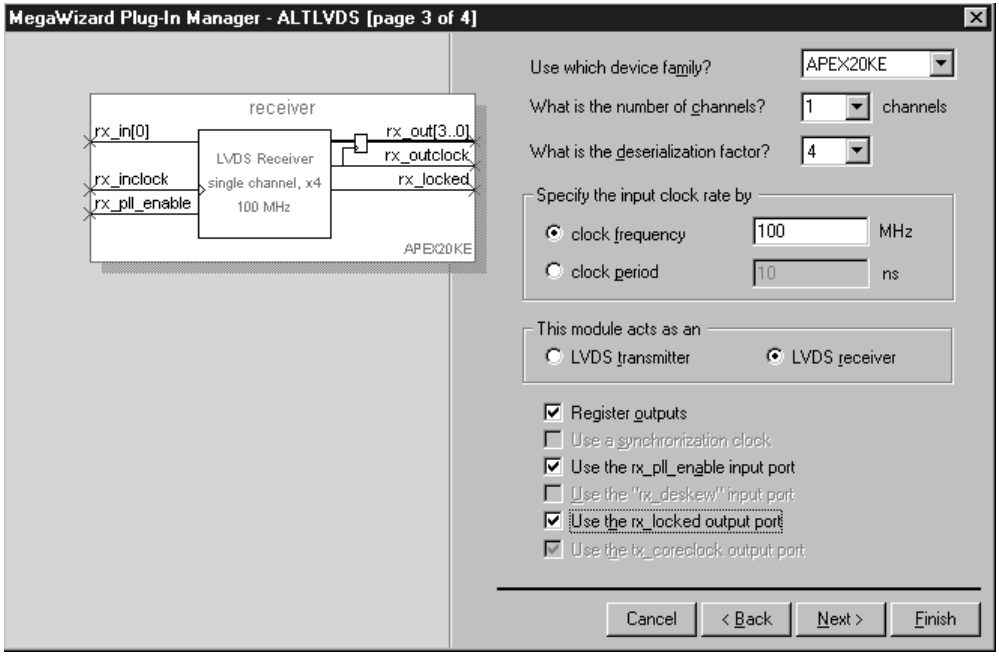


Figure 32. Page 3 of the altlvds Receiver MegaWizard Plug-In Manager



Described below are the various customizable features that are available in the MegaWizard interface:

- Number of channels – this option allows the user to select the number of LVDS channels to be used in the design. The desired value can be either typed or selected from the pop-up menu, up to a maximum of 16 channels. This simplifies the complexity of the design in that only one transmitter or receiver module needs to be instantiated to represent multiple LVDS channels.
- Deserialization factor – this option specifies the number of bits per channel. The user can either type or select 4, 7, or 8 from the pop-up menu.
- Clock frequency / period – this option specifies the clock frequency or period of the LVDS input clock.
- LVDS transmitter / receiver – this option specifies the function of the LVDS module.
- Register inputs / outputs – specifies whether or not to register the inputs for the transmitter and outputs for the receiver. If the signals are not registered in the adjacent MegaLAB structure, then the registered inputs / outputs option must be turned on.

- Use the rx_pll_enable/tx_pll-enable option: optional input ports for the receiver and transmitter which enable control for the LVDS PLL.
- Use the tx_locked /rx_locked port – this option enables the use of the locked pin for the transmitter and receiver. When the phase-locked loop (PLL) locks onto the incoming clock and generates an internal clock, the locked signal is driven high. It remains high as long as the input clock remains within specification.
- Use a synchronization clock – this option activates the synchronization clock for the transmitter. If this option is activated, the synchronization clock must have the same frequency and phase as the transmitter clock in order to avoid hold time violations. To use the synchronization clock, the registered_input option must be set. To set the registered_input option: from the MegaWizard interface, check the *Register Inputs* box, then check the *Use a synchronization clock* box. For more guidelines on designing for LVDS with APEX 20KE devices, see "Guidelines for Using LVDS Blocks" on page 67.
- Use the rx_deskew input port – this option activates the deskew input port for the receiver that is used to calibrate the module.

MegaWizard Examples

Figure 33 shows an altlvds transmitter module generated by the MegaWizard Plug-In Manager with an input frequency of 50 MHz. 16 channels are used with a deserialization factor of 4. In this example, the sync_inclock input and the tx_locked output are both used as well as the input registers.

Figure 33. 50-MHz 16-Channel ×4 LVDS Transmitter

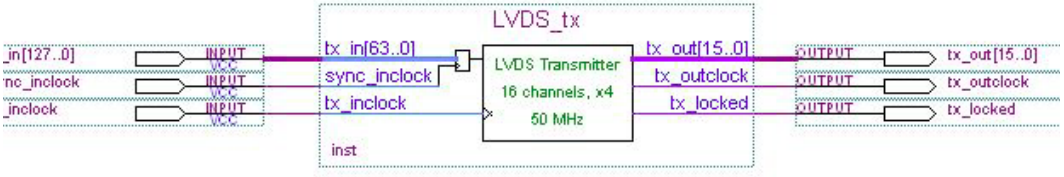
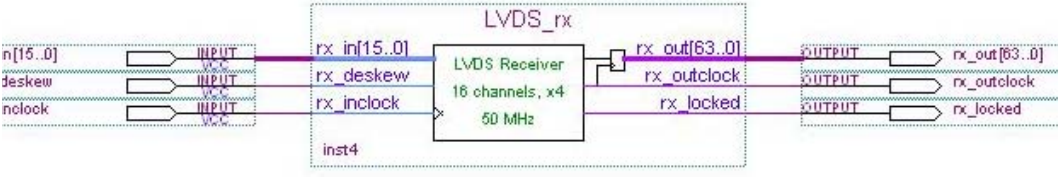


Figure 34 shows the instantiation of an LVDS receiver with 16 channels and an input frequency of 50 MHz. The deserialization factor is set to 4. The rx_deskew, rx_outclock, and output registers are all used in this example.

Figure 34. 50-MHz 16-Channel ×4 LVDS Receiver



HDL Examples

The following examples show the altlvds megafunction in transmitter mode and receiver mode, respectively, in both VHDL and Verilog HDL. These examples instantiate the LVDS modules and connect them to input and output pins.

VHDL

The LVDS transmitter is a 16-channel module operating in ×8 mode with an input clock of 70 MHz. The transmitter's output clock is fed out of the module in ×8 mode through the tx_outclock pin. The status of the PLL can be monitored from the tx_locked pin. The input registers are also used.

Figure 35. lvds_tx.vhd (Part 1 of 2)

```
library ieee;
use ieee.std_logic_1164.all;
entity lvds_tx is
   port
      ( tx_in:            in    std_logic_vector(127 downto 0);
        tx_inclock:       in    std_logic;
        sync_inclock:     in    std_logic;
        tx_out:           out   std_logic_vector(15 downto 0);
        tx_outclock:      out   std_logic;
        tx_locked:        out   std_logic
      );
end lvds_tx;
```

Figure 35. lvds_tx.vhd (Part 2 of 2)

```
architecture apex of lvds_tx is
   component altlvds_tx
   generic
        (number_of_channels:     positive;
        deserialization_factor:  positive;
        registered_input:        string       := "ON";
        multi_clock:             string       := "OFF";
        inclock_period:          positive;
        clock_setting:           string       := "UNUSED"
      );
   port
      ( tx_in:             in     std_logic_vector
        (deserialization_factor*number_of_channels-1 downto 0);
        tx_inclock:        in     std_logic;
        sync_inclock:      in     std_logic := '0';
        tx_out:            out    std_logic_vector
        (number_of_channels-1 downto 0);
        tx_outclock:       out    std_logic;
        tx_locked:         out    std_logic
      );
end component;
begin
U0:altlvds_tx
   generic map    (    number_of_channels => 16,
                       deserialization_factor => 8,
                       inclock_period => 9524
                  )
   port map       (    tx_in => tx_in,
                       tx_inclock => tx_inclock,
                       sync_inclock => sync_inclock,
                       tx_out => tx_out,
                       tx_outclock => tx_outclock,
                       tx_locked => tx_locked
                  );
end apex;
```

The LVDS receiver is a 16-channel module operating in ×8 mode with an input clock of 70 MHz. The receiver's output clock cannot be fed out directly to an output pin; therefore, it feeds the clock port of a dff register.

Figure 36. lvds_rx.vhd (Part 1 of 2)

```
library ieee;
use ieee.std_logic_1164.all;

entity lvds_rx is
   port
      ( rx_in:            in     std_logic_vector(15 downto 0);
        rx_inclock:       in     std_logic;
        rx_deskew:        in     std_logic;
        rx_out:           out    std_logic_vector(127 downto 0);
        rx_outclock:      out    std_logic;
        rx_locked:        out    std_logic;
        inpin:            in     std_logic;
        outpin:           out    std_logic;
        clear:            in     std_logic;
        preset:           in     std_logic
      );
end lvds_rx;

architecture apex of lvds_rx is
   component altlvds_rx
      generic
            (     number_of_channels:        positive;
                  deserialization_factor:    positive;
                  registered_output:         string = "ON";
                  inclock_period:            positive;
                  clock_setting:             string:= "UNUSED"
            );
      port
            (  rx_in:          in
std_logic_vector(number_of_channels-1 downto 0);
               rx_inclock:     in     std_logic;
               rx_deskew:      in     std_logic := '0';
               rx_out:         out
std_logic_vector(deserialization_factor * number_of_channels-1 downto 0);
               rx_outclock:    out    std_logic;
               rx_locked:      out    std_logic
            );
   end component;

   component dff
```

Figure 36. lvds_rx.vhd (Part 2 of 2)

```
   port
      (
      d   : in std_logic;
      clk : in std_logic;
      clrn: in std_logic;
      prn : in std_logic;
      q   : out std_logic
      );
   end component;

signal clock: std_logic;
begin
U0: altlvds_rx
   generic map
      (      number_of_channels => 16,
             deserialization_factor => 8,
             inclock_period => 9524
      )
   port map
      (      rx_in => rx_in,
             rx_inclock => rx_inclock,
             rx_deskew => rx_deskew,
             rx_out => rx_out,
             rx_outclock => clock,
             rx_locked => rx_locked
      );

U1: dff
   port map (d => inpin,
        clk => clock,
        clrn => clear,
        prn => preset,
        q => outpin);
end apex;
```

Verilog HDL

The following examples show the `altlvds` megafunction in Verilog HDL for both the transmitter and receiver, respectively. The LVDS transmitter is a 16-channel module, operating with an input clock of 105 MHz and a deserialization factor of ×8. The output clock of the PLL is fed directly to the `tx_outclock` output pin.

Figure 37. lvds_tx.v

```
module lvds_tx (tx_in, tx_inclock, tx_out, tx_outclock, tx_locked);
       input[127:0] tx_in;
       input tx_inclock;
       output[15:0] tx_out;
       output tx_outclock;
       output tx_locked;
altlvds_tx U0 (.tx_in (tx_in), .tx_inclock (tx_inclock), .tx_out (tx_out),
.tx_outclock (tx_outclock), .tx_locked (tx_locked));
defparam
       U0.number_of_channels    =    16,
       U0.deserialization_factor=    8,
       U0.registered_input      =    "On",
       U0.multi_clock           =    "Off",
       U0.inclock_period        =    9524;
endmodule
```

The receiver is also a 16-channel module with an input clock frequency of 70 MHz and a deserialization factor of ×8. The output clock of the PLL is fed to the clock port of register D1 since it cannot be fed directly to an output pin.

Figure 38. lvds_rx.v

```
module lvds_rx ( rx_in, rx_inclock, rx_deskew, rx_out, rx_locked, inpin,
outpin);
       input[15:0] rx_in;
       input rx_inclock;
       input rx_deskew;
       output[127:0] rx_out;
       output rx_locked;
       input inpin;
       output outpin;
reg pll_out;
altlvds_rx U0 (.rx_in (rx_in), .rx_inclock (rx_inclock), .rx_deskew
(rx_deskew), .rx_out (rx_out), .rx_outclock (pll_out), .rx_locked
(rx_locked));
defparam
       U0.number_of_channels    =    16,
       U0.deserialization_factor=    8,
       U0.registered_output     =    "ON",
       U0.inclock_period        =    9524;
dff D1 (.d(inpin), .q(outpin), .clk(pll_out));
endmodule
```

Synthesis with Third-Party Tools

To synthesize the design successfully in third-party tools such as Synplify, FPGA Compiler / II, FPGA Express, and LeonardoSpectrum, the LVDS design component must be treated as a black box. By declaring the module a black box, synthesis tools will refrain from synthesizing the module. However, the correct port connections will be made in the output EDIF netlist file (.**edf**) or Verilog Quartus II mapping file (.**vqm**). When the netlist file is brought into the Quartus II software, native synthesis on the black-boxed module is automatically performed.

The LVDS module must first be generated by the MegaWizard Plug-In Manager, which involves specifying the name of the module and the ports that are used. Below are examples of an LVDS transmitter design in VHDL and Verilog HDL for several third-party tools. The file named `mylvds_tx` is the MegaWizard-generated file.

Figure 39. Exemplar Synplify & FPGA Compiler/II: lvds_tx.vhd

```
library ieee;
use ieee.std_logic_1164.all;

entity lvds_tx is
   port
      ( tx_in:            in    std_logic_vector(127 downto 0);
        tx_inclock:       in    std_logic;
        sync_inclock:     in    std_logic;
        tx_out:           out   std_logic_vector(15 downto 0);
        tx_outclock:      out   std_logic;
        tx_locked:        out   std_logic
      );
end lvds_tx;

architecture apex of lvds_tx is
component mylvds_tx
      port
      ( tx_in:            in    std_logic_vector(127 downto 0);
        tx_inclock:       in    std_logic;
        sync_inclock:     in    std_logic;
        tx_out:           out   std_logic_vector(15 downto 0);
        tx_outclock:      out   std_logic;
        tx_locked:        out   std_logic
      );
end component;

attribute black_box: boolean;
attribute black_box of mylvds_tx: component is true;

begin
U0:mylvds_tx
      port map (tx_in => tx_in, tx_inclock => tx_inclock, sync_inclock =>
      sync_inclock, tx_out => tx_out, tx_outclock => tx_outclock,
tx_locked =>
      tx_locked);
end apex;
```

Figure 40. Exemplar LeonardoSpectrum: lvds_tx.vhd

```
library ieee;
use ieee.std_logic_1164.all;

entity lvds_tx is
       port
       ( tx_in:            in     std_logic_vector(127 downto 0);
         tx_inclock:       in     std_logic;
         sync_inclock:     in     std_logic;
         tx_out:           out    std_logic_vector(15 downto 0);
         tx_outclock:      out    std_logic;
         tx_locked:        out    std_logic
       );
end lvds_tx;

architecture apex of lvds_tx is
       component mylvds_tx
       port
       ( tx_in:            in     std_logic_vector(127 downto 0);
         tx_inclock:       in     std_logic;
         sync_inclock:     in     std_logic;
         tx_out:           out    std_logic_vector(15 downto 0);
         tx_outclock:      out    std_logic;
         tx_locked:        out    std_logic
       );
end component;

attribute noopt: boolean;
attribute noopt of mylvds_tx: component is true;

begin
U0:mylvds_tx
       port map (tx_in => tx_in, tx_inclock => tx_inclock, sync_inclock =>
sync_inclock, tx_out => tx_out, tx_outclock => tx_outclock, tx_locked =>
tx_locked);
end apex;
```

The code below demonstrates black-boxing in Verilog HDL using the same transmitter module generated by the MegaWizard Plug-In Manager.

Figure 41. Exemplar Synplify/FPGA Complier/II: lvds_tx.v

```
module mylvds_tx (tx_in, tx_inclock, tx_out, tx_outclock, tx_locked);
/*synthesis black_box*/
       input[127:0] tx_in;
       input tx_inclock;
       output[15:0] tx_out;
       output tx_outclock;
       output tx_locked;
endmodule
module lvds_tx (tx_in, tx_inclock, tx_out, tx_outclock, tx_locked);
       input[127:0] tx_in;
       input tx_inclock;
       output[15:0] tx_out;
       output tx_outclock;
       output tx_locked;

mylvds_tx U0 (.tx_in (tx_in), .tx_inclock (tx_inclock), .tx_out (tx_out),
.tx_outclock (tx_outclock), .tx_locked (tx_locked));

endmodule
```

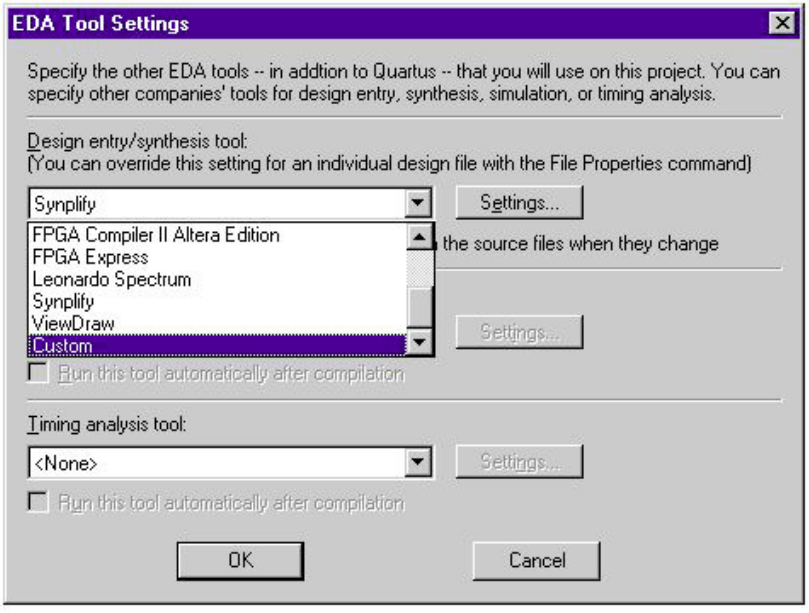
Figure 42. Exemplar LeonardoSpectrum: lvds_tx.v

```
module mylvds_tx (tx_in, tx_inclock, tx_out, tx_outclock, tx_locked);
       input[127:0] tx_in;
       input tx_inclock;
       output[15:0] tx_out;
       output tx_outclock;
       output tx_locked;
       endmodule
// higher level file
module lvds_tx (tx_in, tx_inclock, tx_out, tx_outclock, tx_locked);
       input[127:0] tx_in;
       input tx_inclock;
       output[15:0] tx_out;
       output tx_outclock;
       output tx_locked;

mylvds_tx U0 (.tx_in (tx_in), .tx_inclock (tx_inclock), .tx_out (tx_out),
.tx_outclock (tx_outclock), .tx_locked (tx_locked));
//exemplar attribute U0 NOOPT TRUE
endmodule
```

The Quartus II software must be configured so that it recognizes the EDF or VQM netlist file from the third-party synthesis tool. The synthesis tool can be selected from the **EDA Tool Settings** dialog box (Project menu) in the Quartus II software, as seen in Figure 43. More information regarding the Quartus II software's integration with third-party tools can be found on the Altera web site at **http://www.altera.com/html/nativelink/nativelink.html**.

Figure 43. EDA Tool Settings in the Quartus II Software



Testbenches

The following testbench examples, which can be used in third-party simulators such as ModelSim, show the functionality of the LVDS behavioral model. The receiver's output is connected directly to the transmitter's input. The deskew pin is asserted, and the calibration pattern is applied first. If this is not done correctly, the model will not allow data to exit the transmitter. The test pattern enters the receiver's input port and subsequently leaves the transmitter's output port after two clock cycles of latency. When running this test bench, ensure that the time resolution of the simulator is set to picoseconds. More information on receiver calibration can be found in the following VHDL and Verilog HDL testbench examples.

Figure 44. VHDL HDL: lvds_test.vhd (Part 1 of 3)

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_arith.ALL;
USE ieee.std_logic_unsigned.ALL;
USE work.apex20ke_mf_components.ALL;
USE std.textio.ALL;
ENTITY lvds_test IS
END lvds_test;

ARCHITECTURE testbench OF lvds_test IS

       SIGNAL rx_in        : std_logic_vector(3 downto 0) := "0000";
       SIGNAL rx_inclock   : std_logic := '1';
       SIGNAL rx_deskew    : std_logic := '0';
       SIGNAL synch_inclock: std_logic := '0';
       SIGNAL tx_out       : std_logic_vector(3 downto 0);
       SIGNAL tx_outclock  : std_logic;
       SIGNAL rx_out       : std_logic_vector(31 downto 0);
       SIGNAL rx_outclock  : std_logic;
       SIGNAL lvds_data_clk: std_logic := '1';

       TYPE rx_buffer IS ARRAY(0 to 17, 0 to 3) OF std_logic;

BEGIN

-- Instantiate the LDVS Receiver
L0: altlvds_rx
       GENERIC MAP (       number_of_channels => 4,
                           deserialization_factor => 8,
                           inclock_period => 9524,
                           registered_output => "ON")
       PORT MAP (          rx_in => rx_in,
                           rx_inclock => rx_inclock,
                           rx_deskew => rx_deskew,
                           rx_out => rx_out,
                           rx_outclock => rx_outclock);

-- Instantiate the LVDS Transmitter
L1: altlvds_tx

       GENERIC MAP (       number_of_channels => 4,
                           deserialization_factor => 8,
                           inclock_period => 128000,
                           registered_input => "ON")
         PORT MAP (        tx_in => rx_out,
```

Figure 44. VHDL HDL: lvds_test.vhd (Part 2 of 3)

```
                          tx_inclock => rx_inclock,
                          sync_inclock => synch_inclock,
                          tx_out => tx_out,
                          tx_outclock => tx_outclock);

-- Create a 7,812,500 Hz clock
PROCESS(rx_inclock)
BEGIN
       rx_inclock <= NOT rx_inclock AFTER 64 ns;
END PROCESS;

-- Create a 62,500,000 lvds data clock to synch data inputs
PROCESS(lvds_data_clk)
BEGIN
       lvds_data_clk <= NOT lvds_data_clk AFTER 8 ns;
END PROCESS;

PROCESS
       VARIABLE deskew_pattern : std_logic_vector(23 downto 0) :=
"000011110000111100001111";
       VARIABLE test_pattern : std_logic_vector(15 downto 0) :=
"0000011000000001";
       VARIABLE cnt : integer range 0 to 15 := 0;
       VARIABLE deskew_cnt: integer range 0 to 48 := 0;
BEGIN
       IF deskew_cnt < 48 THEN
         IF deskew_cnt > 23 THEN
                          rx_deskew <= '1';
                          rx_in(0) <= deskew_pattern(deskew_cnt -24);
                          rx_in(1) <= deskew_pattern(deskew_cnt -24);
                          rx_in(2) <= deskew_pattern(deskew_cnt -24);
                          rx_in(3) <= deskew_pattern(deskew_cnt -24);
         END IF;
         deskew_cnt := deskew_cnt + 1;
         wait until ((lvds_data_clk'event) and (lvds_data_clk = '1'));
       ELSIF cnt < 15 THEN
                          rx_deskew <= '0';
                          rx_in(0) <= test_pattern(cnt);
                          rx_in(1) <= test_pattern(cnt);
                          rx_in(2) <= test_pattern(cnt);
                          rx_in(3) <= test_pattern(cnt);
                          cnt := cnt + 1;
                          wait until ((lvds_data_clk'event) and
(lvds_data_clk = '1'));
       ELSE cnt := 0;
```

Figure 44. VHDL HDL: lvds_test.vhd (Part 3 of 3)

```
      END IF;
END PROCESS;
END testbench;
```

Figure 45. Verilog HDL: lvds_test.v (H4) (Part 1 of 2)

```
‘timescale 1ps/1ps

module lvds_test();
reg [3:0] rx_in;
reg rx_inclock;
reg rx_deskew;
reg synch_inclock;
wire [3:0] tx_out;
wire tx_outclock;
wire [31:0] rx_out;
wire rx_outclock;
reg lvds_data_clk;

reg [23:0] deskew_pattern;
reg [15:0] test_pattern;

reg [3:0] cnt;
reg [5:0] deskew_cnt;

altlvds_rx L0(.rx_in(rx_in), .rx_inclock(rx_inclock),
.rx_deskew(rx_deskew), .rx_out(rx_out), .rx_outclock(rx_outclock));
defparam L0.number_of_channels = 4;
defparam L0.deserialization_factor = 8;
defparam L0.registered_output = "ON";
defparam L0.inclock_period = 128000;

altlvds_tx L1(.tx_in(rx_out), .tx_inclock(rx_inclock),
.sync_inclock(synch_inclock), .tx_out(tx_out),
.tx_outclock(tx_outclock));
defparam L1.number_of_channels = 4;
defparam L1.deserialization_factor = 8;
defparam L1.registered_input = "ON";
defparam L1.inclock_period = 128000;

initial
begin
      rx_in = 4’b0000;
```

```
        rx_deskew = 1'b0;
```

Figure 45. Verilog HDL: lvds_test.v (H4) (Part 2 of 2)

```
       synch_inclock = 1'b0;
       deskew_cnt = 6'b000001;
       cnt = 4'b0000;
       deskew_pattern = 24'b000011110000111100001111;
       test_pattern = 16'b1000011110000001;
end

initial
begin
       rx_inclock = 1'b1;
       forever #64000 rx_inclock = ~rx_inclock;
end

initial
begin
       lvds_data_clk = 1'b1;
       forever #8000 lvds_data_clk = ~lvds_data_clk;
end

always@(posedge lvds_data_clk)
begin
       if (deskew_cnt < 48)
         begin
         if (deskew_cnt > 23)
                           begin
                           rx_deskew = 1'b1;
                           rx_in[0] = deskew_pattern[deskew_cnt-24];
                           rx_in[1] = deskew_pattern[deskew_cnt-24];
                           rx_in[2] = deskew_pattern[deskew_cnt-24];
                           rx_in[3] = deskew_pattern[deskew_cnt-24];
                           end
         deskew_cnt = deskew_cnt + 1;
         end
       else if (cnt <= 15)
         begin
         #1 rx_deskew = 1'b0;
         rx_in[0] = test_pattern[cnt];
         rx_in[1] = test_pattern[cnt];
         rx_in[2] = test_pattern[cnt];
         rx_in[3] = test_pattern[cnt];
         cnt = cnt + 1;
         end
       else cnt = 0;
end
endmodule
```

Quartus II LVDS Reporting

The Quartus II software reports LVDS usage in the compilation report file. The report file documents all information pertaining to LVDS resource usage and placement in the APEX device under the following categories:

- All Package Pins
- Control Signals
- Global and Other Fast Signals
- LVDS
- ClockLock

This section briefly describes each category.

All Package Pins

This category of the report file indicates the function and location of all package pins. LVDS pins are displayed with their names and pin numbers, as seen in the Figure 46 example.

Figure 46. All Package Pins Category of the Report File

All Package Pins

Pin #	Usage
D2	GND
D3	GND
D4	GND
D5	VCCIO
D6	rx_out[0]
D7	rx_out[6]
D8	GND*
D9	GND*
D10	GND*
D11	GND*
D12	VCCINT
D13	GND*
D14	GND*

The Quartus II software adheres to the previously discussed banking rules and will not place non-LVDS outputs in LVDS-enabled banks. In such configurations, the design yields a no-fit, indicating that these non-LVDS outputs are illegally placed.

For more information on using I/O standards in the Quartus II software, refer to *Application Note 117 (Using I/O Standards in the Quartus Software)*.

Control Signals

The Control Signals category reports the control signals that are present in the design. LVDS control signals, such as input clocks and PLL output clocks, are reported as seen in Figure 47. PLL output clocks are denoted as either `pll_clk0` or `pll_clk1`. The `pll_clk1` output clock can be fed directly out of the transmitter in 1× mode.

Figure 47. Control Signals Category of the Report File

Control Signals

Name	Pin #	Fan-Out	Usage
|rec:inst1|altlvds_rx:altlvds_rx_component|pll_clk0	LVDS_PLLRX_1	0	Clock
|rec:inst1|altlvds_rx:altlvds_rx_component|pll_clk1	LVDS_PLLRX_1	0	Clock
|sync_inclock	N8	16	Clock
|tra:inst|altlvds_tx:altlvds_tx_component|pll_clk0	LVDS_PLLTX_1	0	Clock
|tra:inst|altlvds_tx:altlvds_tx_component|pll_clk1	LVDS_PLLTX_1	0	Clock
|rx_inclock	AB25	16	Clock
|tx_inclock	D1	16	Clock

Global and Other Fast Signals

The Global and Other Fast Signals category displays the globally routed signals in the design. When LVDS is used, only the PLL-generated clocks and the synchronization clocks are routed globally as seen in Figure 48. The number of fan-out nodes for the global signal is also displayed.

Figure 48. Global and Other Fast Signals Category of the Report File

Global & Other Fast Signals

Name	Pin #	Fan-Out	Global
|rec:inst1|altlvds_rx:altlvds_rx_component|pll_clk0	LVDS_PLLRX_1	1	yes
|rec:inst1|altlvds_rx:altlvds_rx_component|pll_clk1	LVDS_PLLRX_1	10	yes
|sync_inclock	N8	16	yes
|tra:inst|altlvds_tx:altlvds_tx_component|pll_clk0	LVDS_PLLTX_1	1	yes
|tra:inst|altlvds_tx:altlvds_tx_component|pll_clk1	LVDS_PLLTX_1	2	yes

LVDS

This category reports LVDS usage in the design, as seen in Figure 49. The instance name is displayed along with its function and deserialization factor. Both PLL output clocks for the LVDS modules are also shown. The LVDS category is omitted when LVDS is not used in the APEX device.

Figure 49. LVDS Section of the Report File

LVDS

Name	Function	Clock0	Clock1	Data Width
|lvds:inst5|altlvds_rx:altlvds_rx_component|rx[0]	Transmitter	|lvds:inst5|altlvds_rx:altlvds_rx_component|pll_clk0	|lvds:inst5|altlvds_rx:altlvds_rx_component|pll_clk1	4
|tx:inst14|altlvds_tx:altlvds_tx_component|tx_out[0]	Receiver	|tx:inst14|altlvds_tx:altlvds_tx_component|pll_clk0	|tx:inst14|altlvds_tx:altlvds_tx_component|pll_clk1	4

ClockLock

The ClockLock category of the report file, as seen in Figure 50, gives the specifications of each PLL that was used. The input frequency is indicated as well as the various resulting clock frequencies after multiplication by the deserialization factor.

Figure 50. ClockLock Section of the Report File

ClockLock

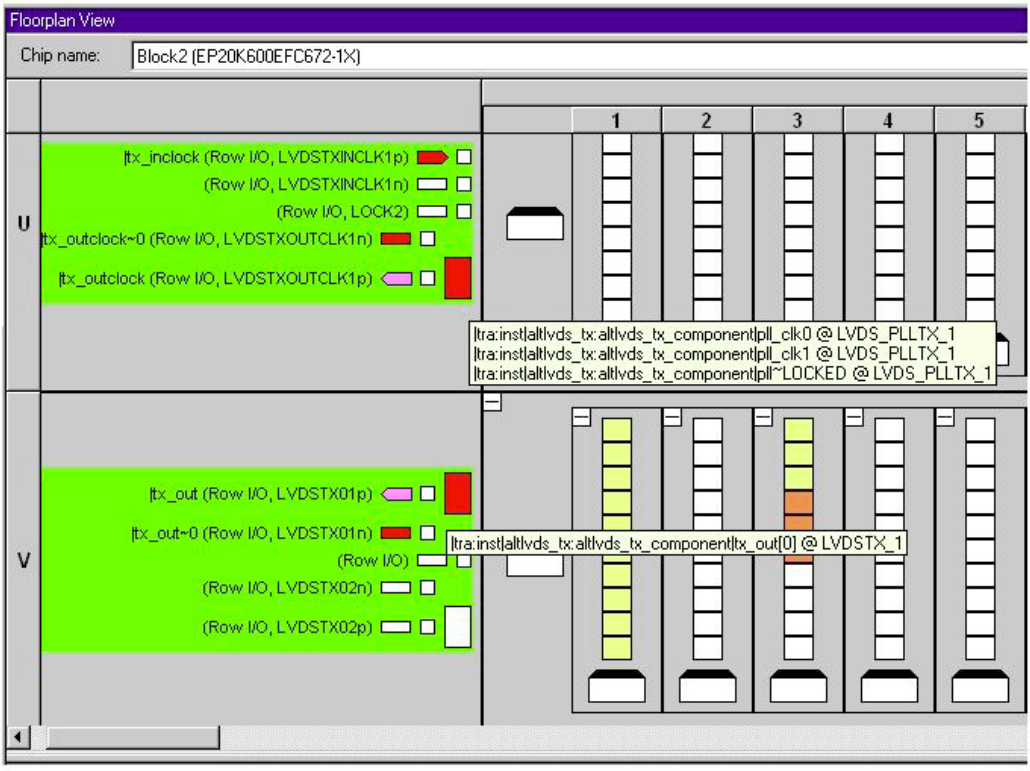
Name	Mode	Input Frequency	Multiply Clock0	Divide Clock0	Ouput Frequency	Multiply Clock1	Divide Clock1	Ouput Frequency	Phase Shift	External Feedback
|rec:inst1|altlvds_rx:altlvds_rx_component	LVDS	50.0 MHz	8	1	400.0 MHz	1	1	50.0 MHz	0 ps	no
|tra:inst|altlvds_tx:altlvds_tx_component	LVDS	50.0 MHz	8	1	400.0 MHz	1	1	50.0 MHz	0 ps	no

Floorplanner

The Floorplanner gives a visual representation of the internal routing and placement of logic within the device.

Figure 51 shows the Floorplan view for an LVDS transmitter. The transmitter is divided between two or more colored blocks: the LVDS PLL is located adjacent to the transmitter output clock pins (`LVDSTXOUTCLK1p` and `LVDSTXOUTCLK1n`), and the individual parallel-to-serial converters are located adjacent to each pair of LVDS data-out pins (e.g., `LVDSTX01p` and `LVDSTX01n`).

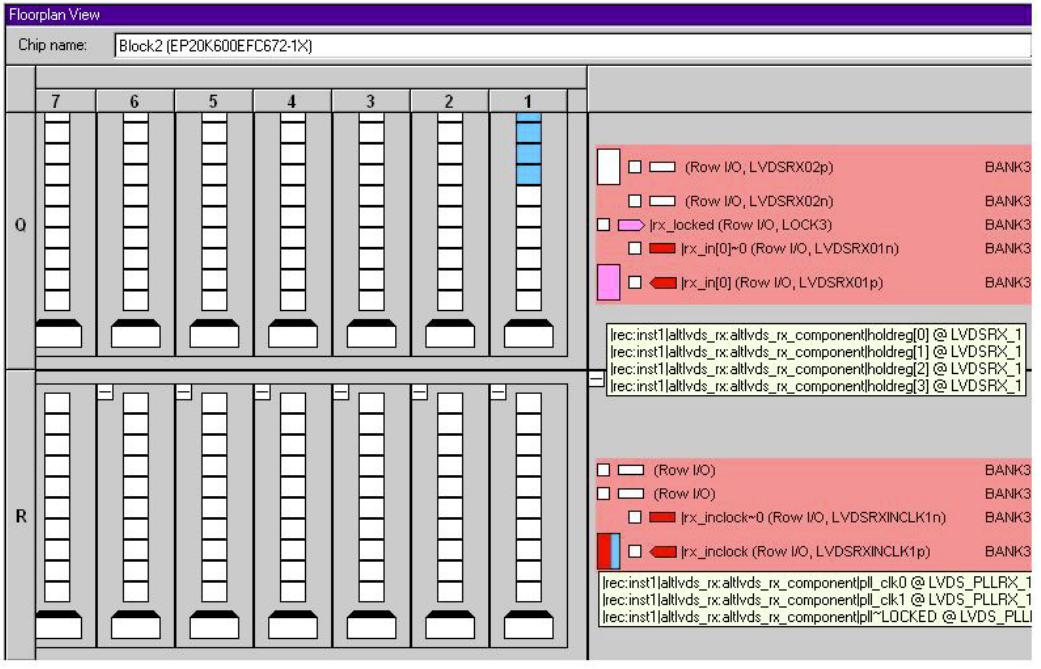
Figure 51. Floorplan View of LVDS Transmitter



In Figure 51, the PLL appears in the equations as LVDS_PLLTX_1, and the single parallel-to-serial converter appears as LVDSTX_1. The logic cells that appear to the right of the LVDS modules represent additional logic that is consumed during implementation. The transmitter's locked PLL clock (pll_clk1) can be driven off-chip in 1× mode through the transmitter output clock pins.

The Quartus II software displays the LVDS receiver module in a similar fashion, as seen in Figure 52. The receiver is divided between two or more colored blocks: the LVDS PLL is located adjacent to the receiver input clock pins (LVDSRXINCLK1p and LVDSRXINCLK1n), and the individual serial-to-parallel converters are located adjacent to the LVDS data-in pins (LVDSRX01p and LVDSRX01n). The PLL appears as LVDS_PLLRX_1, and the serial-to-parallel converter appears as LVDSRX_1 in this example. The logic cells that appear to the left of the LVDS modules represent additional logic that is consumed during implementation. Because the PLL receiver output clock cannot be fed externally, it does not fan-out to any I/O pins in the Floorplan view.

Figure 52. Floorplanner View of LVDS Receiver

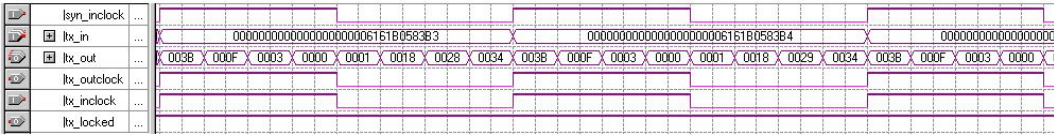


Simulation in Quartus II

The Quartus II development tool provides users with the capability to conveniently and efficiently simulate the LVDS design. Vector waveform files (**.vwf**), which are used as inputs to the native simulation tool, can be created within the Quartus II software. The simulation model for the LVDS receiver is essentially a serialization shift-register that is driven by an LVDS data channel and clocked by an LVDS PLL multiplied by the serialization value. The shift-register drives a bank of data registers clocked by the original clock. The LVDS transmitter module is the inverse of the receiver. A data register is driven by internal parallel data signals and clocked by the original LVDS clock. It then loads a shift-register that drives the LVDS output pin and is clocked by the multiplied output of the LVDS PLL. For more information on simulation in the Quartus II software, see Quartus II Help.

Figure 53 shows the results of an example functional simulation of an LVDS transmitter. The 16-channel transmitter is operating at 60 MHz with the synchronization clock activated and a deserialization factor of 8.

Figure 53. Example Functional Simulation Waveform of LVDS Transmitter



The locked pin `tx_locked` remains high as long as the input frequency is valid. The input clock `tx_inclock` and the synchronization clock `syn_inclock` must have the exact same phase and frequency for the module to function correctly. The clocks must also have the same frequency specified in the design files. If the frequency differs, the Simulator will warn that the PLL was unable to lock onto the incoming clocks.

The incoming data `tx_in` is synchronized with the input clocks `tx_inclock` and `syn_inclock`. The output data `tx_out` is synchronized with the output clock `tx_outclock` that has the same frequency as the input clock. The output clock has the same frequency as the input clock, (not the internally multiplied clock), because only the 1× version of the PLL-generated clock can be fed out. The output data transitions 8 times within one period of the 1× clock, indicating that the deserialization factor is 8.

Figure 54 shows the results of an example functional simulation of the deskew circuitry in the LVDS receiver. The deskew pin rx_deskew is asserted for at least three clock cycles after the PLL locks onto the incoming clock and the deskew calibration pattern is applied to all channels. If the deskew pin is prematurely de-asserted or the deskew calibration pattern is incorrect, the Quartus II software will warn that the deskew pin was de-asserted at an invalid time.

Figure 54. Example Functional Simulation Waveform of Receiver Calibration

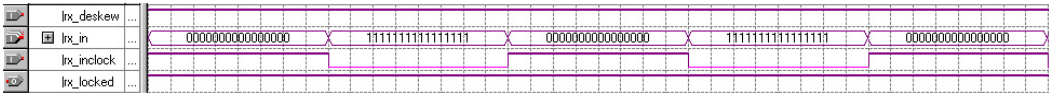
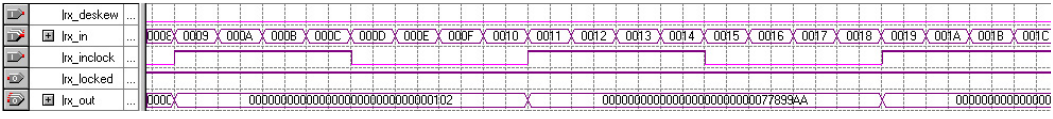


Figure 55 shows the results of an example functional simulation of an LVDS receiver. The locked pin rx_locked is asserted as long as the incoming clock signal is valid. The valid frequency is determined by the value that is set in the MegaWizard Plug-In Manager. If this frequency does not correspond, the simulation will indicate that the PLL could not lock onto the incoming clock signal.

Figure 55. Example Functional Simulation Waveform of LVDS Receiver



The incoming data is synchronized with the incoming clock signal rx_inclock. The input data transitions 8 times within one incoming clock period, indicating that the deserialization factor is 8. The output data is synchronized with the output clock of the receiver module, which is not displayed in the figure.

Figure 56 indicates the bit mapping performed by the Quartus II software for a single receiver channel operating in 8× mode at 60 MHz. The functional simulation waveforms show that the data for the current clock cycle of rx_inclock (beginning at 41.66 ns) maps to the appropriate positions, as indicated in Table 17 on page 34. The first bit of data for the current clock cycle is not accepted until 45.83 ns, which is immediately after the last two bits of the previous cycle are accepted.

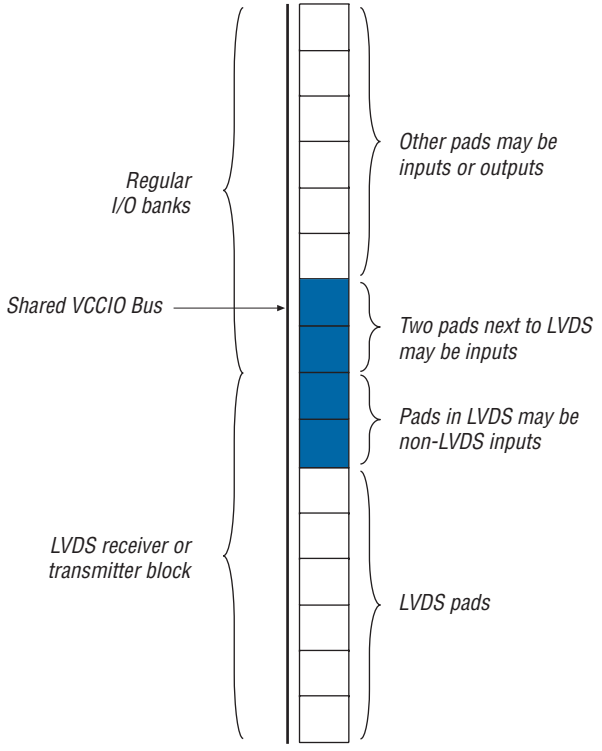
Figure 56. Bit Mapping Sample Waveform

Guidelines for Using LVDS Blocks

When using differential signaling in receiver and / or transmitter LVDS blocks, you cannot place non-differential output pins within two I / O pads of the LVDS receiver or transmitter blocks. You cannot place single-ended outputs on any LVDS block pins when using differential signaling on any of the channels. Using switching outputs on LVDS block pins (except the PLL LOCK pin) could affect True-LVDS pins and degrade performance. You can use switching outputs on the PLL LOCK pin because it rarely changes. As shown in Figure 57, output pins must be at least two pads away from the LVDS receiver and transmitter blocks unless separated by a power or ground pin.

The two-pad guideline also applies to the dedicated LVDS clock pins and the global clock pins when using differential signaling. You cannot place output pins within two pads of the LVDS clock pins (both dedicated and non-dedicated) unless separated by a power or ground pin. You can use unused True-LVDS pins as input pins without compromising the acceptable noise level on the V_{CCIO} plane. The **Show Pads** view in the Quartus II Floorplan Editor can be used to see the pad order.

Figure 57. I/O Pin Placement in the I/O Bank Adjacent to the LVDS Blocks



Required Guidelines for LVDS in APEX 20KE Devices

Use the information in this section when designing for LVDS with APEX 20KE devices. Altera recommends that you read these guidelines before laying out PCB boards.

Global line G3 is always used in designs using the LVDS receiver. There are registers internal to the megafunction that are clocked using the 1× clock generated by the LVDS receiver PLL. Therefore, the global line G3 will be used even if the `rx_outclock` port is not feeding any clock input ports in the design files. LE registers necessary for synchronization between the LVDS block and the core are created internally in the megafunction under the following circumstances:

- When the LVDS receiver is in 4× mode. In the 4× mode case, there is an extra set of LE hold registers required at the output of the LVDS block. When registered outputs in the megafunction are disabled, these hold registers will still exist.
- When the LVDS receiver is in 7× or 8× mode and registered outputs are enabled in the megafunction.

The only case where registers are not created inside the megafunction is when the LVDS receiver is 7× or 8× mode and the registered outputs are disabled. In this case the LVDS receiver will not function correctly unless the data is fed into core logic (LEs, ESBs). In this case, the `rx_outclock` is required, and as a result, the global line G3 is still necessary.

The `LVDSTXINCLK` pins cannot feed core logic (LEs, ESBs) because the data transfer from the core logic to the LVDS transmitter block creates hold time violations. Care should be taken when designing with the `LVDSTXINCLK` pins. In the following cases, the `altlvds` transmitter megafunction must have the **MULTI_CLOCK** setting set to **Yes** and the `sync_inclock` port must be used. The global clock pin(s) that feeds the `sync_inclock` must be the same phase and frequency as the `LVDSTXINCLK` pins.

- Required when the LVDS transmitter is in 7× and 8× modes, which have an extra set of hold registers instantiated by the megafunction at the input to the LVDS transmitter block. So even if you disable registered inputs in the megafunctions, these hold registers will still exist.
- When the LVDS transmitter is in 4× mode and registered inputs are enabled in the megafunction.

The easiest way to implement dual frequency applications is to use the `LVDSRXINCLK` to feed the receiver PLL and a dedicated clock pin to feed the transmitter PLL.

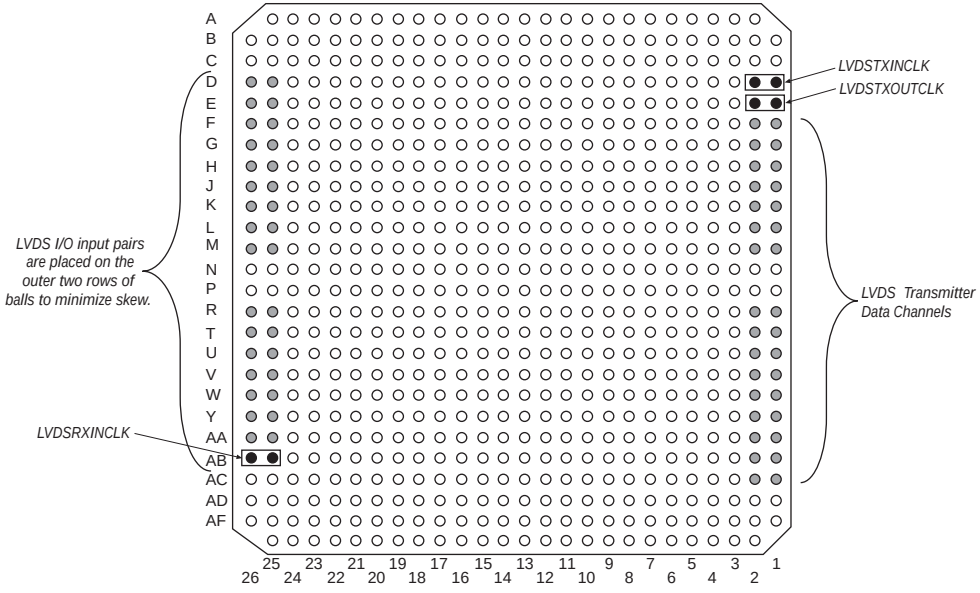
When designing with LVDS in the transmitter or receiver blocks, the two pads around any LVDS signal cannot be used as an output pin. The Quartus II software will report an error message for violating the two-pad rule next to LVDS pins. These two I/O pads are input only because the LVDS blocks share the same V_{CCIO} supply with the I/O bank that it is located in. Noise from the switching of the non-LVDS output pins would degrade the LVDS performance in the LVDS block.

Packaging

The low inductance and capacitance of 1.27-mm BGA and 1.0-mm FineLine BGA packages makes them ideal for the LVDS feature.

The balls used for LVDS signals are located on the outer two rows of balls on the FineLine BGA package. Figure 58 shows the LVDS ball placement on a 672-pin FineLine BGA package for EP20K400E, EP20K600E, and EP20K1000E devices. The marked pins include the 16 LVDS input signals and LVDS clock input on the left side of the package (bottom view). The 16 LVDS output signals, clock signal, and clock output signal are shown on the right side of the package.

Figure 58. Location of LVDS I/O Balls on a 672-Pin FineLine BGA Package



Applications

APEX devices offer different LVDS modes with multiple ways to connect the receiver and transmitter LVDS PLLs. You can interface multiple LVDS devices by using a method that accommodates your design needs. The following LVDS applications are supported with APEX 20KE devices:

- Point-to-point configurations
- Multi-drop LVDS
- Bypassing the dedicated LVDS converter circuitry

Point-to-Point Configurations

Point-to-point LVDS applications involve two devices communicating data via LVDS. For point-to-point communication, the receiver PLL can be clocked from either of two sources: the same source as the transmitting devices or the PLL output clock generated by the transmitting devices. Figures 59 and 60 show both cases.

For high performance, use the source synchronous clocking scheme shown in Figure 59. Standard I/O timing parameters (setup time and clock-to-output) must be taken into consideration for the application shown in Figure 60. For the source synchronous application, designers should follow the LVDS timing budget defined in "Data Orientation" on page 8.

Figure 59. Transmitter PLL Clocks Receiver PLL

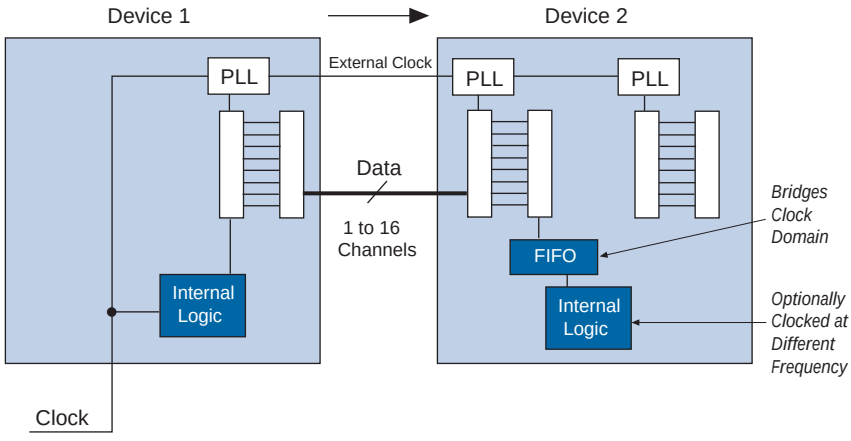
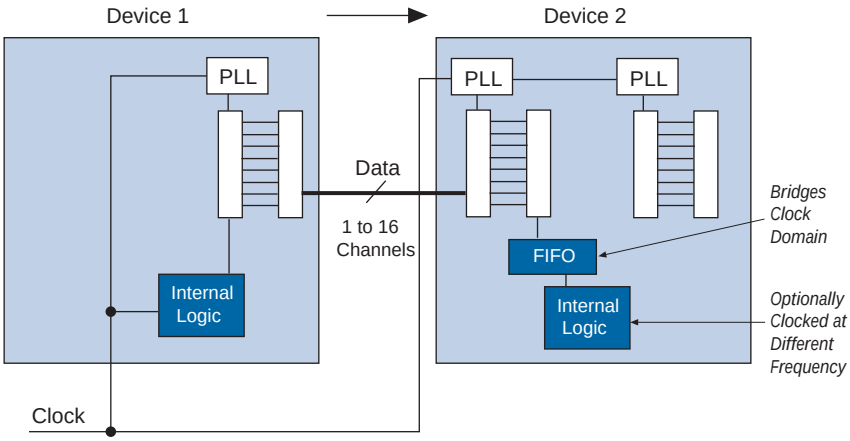


Figure 60. Receiver PLLs Clocked by Board Clock

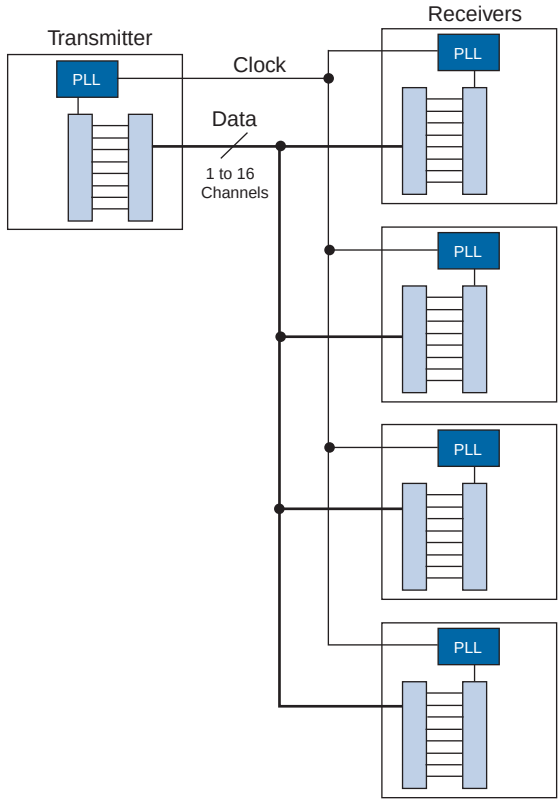


Multi-Drop Configurations

Multi-drop configurations have one transmitter and multiple receivers. The transmitter clock from the source device clocks the LVDS PLLs in the receiving devices. Performance is affected by the number of loads that the transmitter is required to drive. Preliminary information shows that an APEX 20KE device can support up to 16 loads at 400 MHz. Contact Altera Applications for up-to-date information on multi-drop configurations.

Figure 61 shows a multi-drop configuration with four loads.

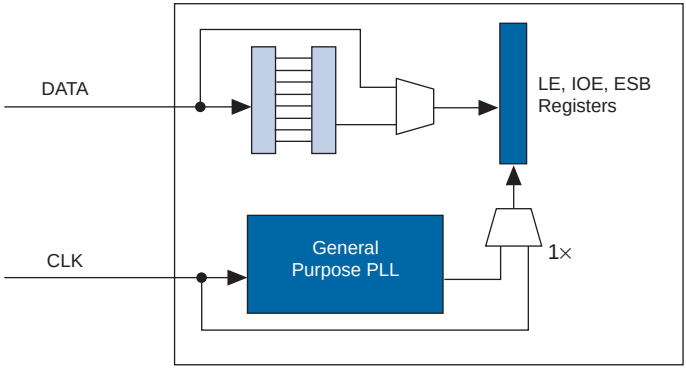
Figure 61. Multi-Drop Configuration



Bypassing the Dedicated LVDS Converter Circuitry

For low data rate LVDS signals that are less than 155 Mbps, data can bypass the dedicated serial-to-parallel converters and feed LEs directly, as shown in Figure 62. The global clock pin can either use a general-purpose PLL or bypass it and globally clock the registers directly. The PLL can be used for phase shifting the clock with respect to the data to achieve maximum I/O timing. The setup and hold times are sufficient to meet the 155 Mbps requirements. In this application method, the clock and data are running at the same rate. The clock can be any standard, and one of the general-purpose PLLs can be used with 1× multiplication to clock the LE registers. The general-purpose PLL supports LVDS signals and can operate up to an input frequency of 420 MHz.

Figure 62. Data Feeds LEs Directly



Summary

The APEX 20KE device is the first PLD to offer a system-on-a-programmable-chip (SOPC) LVDS solution. LVDS integration brings higher data transmission rates and broader bandwidth, all with lower power consumption. LVDS also improves I/O interface performance and simplifies board design by minimizing the number of devices used to interface with backplanes. LVDS-integrated APEX 20KE devices raise the standard and solve the challenge of tomorrow's complex design demands.

References

Electrical Characteristics of Low Voltage Differential Signaling (LVDS) Interface Circuits, ANSI/TIA/EIA-644, American National Standards Institute/Telecommunication Industry Association/Electronic Industries Association.

Revision History

The information contained in the *Application Note 120 (Using LVDS in APEX 20KE Devices)* version 1.3 supersedes information published in previous versions.

Version 1.3

The following change was made to the *Application Note 120 (Using LVDS in APEX 20KE Devices)* version 1.3:

- Updated Figure 7.

Version 1.2

The following changes were made to the *Application Note 120 (Using LVDS in APEX 20KE Devices)* version 1.2:

- Updated Figure 18.
- Updated *Note (1)* to Figure 18.
- Updated Figure 57.
- Updated "Guidelines for Using LVDS Blocks" on page 67.

Version 1.1

The following changes were made to the *Application Note 120 (Using LVDS in APEX 20KE Devices)* version 1.1:

- Corrected the equation for margin on page 13.
- Updated Table 19 to include a note for `rx_outclock`.
- Corrected Figures 59 and 60.

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