Document
Revision History

This User Manual provides basic information about using the Microtronix Camera Link IP Core, PN: 6283-xx-xx. The following table shows the document revision history.

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 31, 2011</td>
<td>Initial release – Version 1.0</td>
</tr>
<tr>
<td>June 17, 2011</td>
<td>Camera Link Medium added - Version 1.1</td>
</tr>
<tr>
<td>July 2012</td>
<td>Transmitter and PoCL - Version 1.3</td>
</tr>
<tr>
<td>August 2012</td>
<td>Link Aligner now supports Full; remove note about limitations – Version 1.4</td>
</tr>
<tr>
<td>June 13, 2013</td>
<td>Update pll clock settings</td>
</tr>
<tr>
<td>Nov 21, 2013</td>
<td>Add timing closure section. Improve Cyclone V support.</td>
</tr>
<tr>
<td>Apr 16, 2014</td>
<td>Change clocking for V family devices</td>
</tr>
<tr>
<td>June 7, 2016</td>
<td>Add support for Max 10 and add alternate I/O option for Altera V family devices.</td>
</tr>
</tbody>
</table>

How to Contact Microtronix

**E-mail**

Sales Information: sales@microtronix.com

Support Information: support@microtronix.com

**Website**

General Website: http://www.microtronix.com

Downloads: http://www.microtronix.com/downloads/

Support FTP site: http://microtronix.leapfile.com

**Phone Numbers**

General: (001) 519-690-0091

Fax: (001) 519-690-0092
## Typographic Conventions

<table>
<thead>
<tr>
<th>Path/Filename</th>
<th>A path/filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>[SOPC Builder]$: &lt;cmd&gt;</td>
<td>A command that should be run from within the Cygwin Environment.</td>
</tr>
<tr>
<td>Code</td>
<td>Sample code.</td>
</tr>
<tr>
<td>←</td>
<td>Indicates that there is no break between the current line and the next line.</td>
</tr>
</tbody>
</table>
Features

- 7:1 Camera Link Serializer / Deserializer (SerDes)
- Core supports Camera and Frame Grabber configurations
- Supports 8-bit 10-tap Base, Medium and Full configurations
- Supports 64-bit and 80-bit Full configuration
- Link alignment of Medium and Full sources
- Supports bi-directional serial Camera Link communication
- Supports Power Over Camera Link SafePower
- Configuration GUI streamlines design process
- Operation up to 85 MHz pixel clock in Cyclone, Stratix and Arria FPGA devices
- Supports Cyclone III, IV & V, Stratix III, IV & V and Arria II, & V (including GX) devices
- Support for OpenCore Plus evaluation

Deliverables

- Java Configuration GUI
- TimeQuest timing analyzer Synopsis Design Constraint (SDC) file
- VHDL ModelSim library
- Perpetual IP core license with 1 year of maintenance updates
- User Documentation
**Introduction**

The Microtronix **Camera Link IP Core** is designed for building vision systems incorporating Camera Link™ communication interfaces including Base, Medium & Full Channel Link in 64-bit and 80-bit configurations. The core supports camera control signals, serial communication, and video data. It is designed for building both Camera and Frame Grabber devices.

The Camera Link standard is based on Channel Link® technology developed by National Semiconductor. Channel Link uses LVDS (Low Voltage Differential Signaling) technology for transmitting digital data. Camera Link uses a parallel-to-serial transmitter and a serial-to-parallel-receiver to transmit data at rates up to 2.38 Gbps.

The Base Camera Link standard uses 28 bits to represent up to 24 bits of pixel data and 3 bits for Video Sync signals: namely Data Valid, Frame Valid, and Line Valid bits. The data is serialized 7:1 into four data streams with a dedicated clock and driven over five LVDS pairs. The Medium configuration adds another 24 bits of data and the Full configuration provides an additional 16 bits for a total of 64 bits of pixel data.

The Camera Link IP also includes optional support for Power Over Camera Link (PoCL) SafePower. When coupled with hardware supporting SafePower, this logic allows a Frame Grabber to provide power to PoCL cameras while also supporting non-PoCL cameras.

The **Camera Link IP Core** is optimized for the Cyclone (III & IV), Stratix (III & IV) and Arria II GX devices.

**Camera Link Receiver**

A block diagram of the Camera Link Base, Medium and Full Receivers core is shown in the figures below. In each core, Channel Link Receivers are used to deserialize the received data. In the Medium and Full implementations a Link Aligner is required to ensure the video data bits are aligned across the interface.

In the block diagrams, the items drawn in grey are logic resources supplied in the FPGA fabric. These include: a Camera Control Parallel IO (PIO) port, the LVDS receivers and transmitters, a (dedicated) PLL and a UART required for serial communications (SerTC/SerTFG) to/from the camera. The LVDS receivers and transmitters are logic blocks supplied in the I/O section in the FPGA.

The PLL clocking differs between device families. Refer to the clock generation section for detailed information about the PLL configurations required for the specific device used.
Figure 1: Block diagram of the Camera Link Base Receiver core
Figure 2: Block diagram of the Camera Link Medium Receiver core
Figure 3: Block diagram of the Camera Link Full Receiver core
**Camera Control Signals**

Four signals are used for general-purpose camera control. They are defined as camera inputs and frame grabber outputs. Camera manufacturers can define these signals to meet their needs for a particular product. The signals are:

- Camera Control 1 (CC1)
- Camera Control 2 (CC2)
- Camera Control 3 (CC3)
- Camera Control 4 (CC4)

The Camera Link IP does not support these signals. In the example projects, an Avalon Parallel IO port is used to interface to these signals.

**Communication UART**

The Camera Link communication UART interface provides two signals for asynchronous serial communication between the Frame Grabber and Camera. The SerTC is provided for communication to the Camera (TC) and SerTFG is provided for communication to the Frame-Grabber (TFG).

The Camera Link IP does not support these signals. In the example projects, an Avalon UART is used to interface to these signals. The UART serial interface is configured for: one start bit, one stop bit, no parity and no handshaking.

**Channel Link Receiver**

The Channel Link Receiver accepts the four LVDS data streams and one LVDS clock, and then derives 28 bits of parallel data and a clock for output to the system.

The Channel Link data encoding for 24-bit RGB is shown in Figure 4 below.

![Channel Link Data Encoding](image)

**Figure 4: Channel Link Data Encoding.**

**Channel Link Receiver Clock**

The received clock is fed through a PLL which generates two clocks for the IP. A fast bit clock is used to clock the data into the 7:1 deserializer and also a pixel
clock with the same frequency as the receive clock is used as a strobe for the parallel video data to the system. This PLL must be instantiated by the user (see Clock Generation).

A PLL is recommended for each Channel Link interface since it cannot be assured the clocks are in phase across the interface.

**Link Aligner**

When more than one Camera Link is in use (Medium or Full configurations), it is possible that the different clocks are not in sync with each other. The Link Aligner block compares the LVAL signal from each Channel Link and adjusts the delays through the IP to bring the links into alignment. Additionally, it synchronizes the output Ports to the STROBE/XCLK clock.

**Power Over Camera Link SafePower**

The optional Power Over Camera Link (PoCL) SafePower logic, when combined with supported hardware, allows a Frame Grabber to provide power to PoCL cameras and still remain compatible with non-PoCL cameras. The SafePower block has been tested with the Microtronix Camera Link Receiver Board.

The SafePower block requires a free-running clock separate from any Camera Link clocks. It also requires that the XCLK PLL have a "locked" output so it can detect an active clock.

**Table 1: Channel Link Receiver signal assignments**

<table>
<thead>
<tr>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RST</td>
<td>IN</td>
<td>Active high reset input</td>
</tr>
<tr>
<td>XCLK</td>
<td>IN</td>
<td>Channel Link X Clock</td>
</tr>
<tr>
<td>XCLK_x1</td>
<td>IN</td>
<td>Channel Link X Clock used for Cyclone V, Arria V, Stratix V</td>
</tr>
<tr>
<td>XCLK_x7</td>
<td>IN</td>
<td>Channel Link X Clock x 7 used for Cyclone V, Arria V, Stratix V</td>
</tr>
<tr>
<td>XCLK_x7_2</td>
<td>IN</td>
<td>Channel Link X Clock x 7 ÷ 2 used for devices other than Cyclone V, Arria V and Stratix V</td>
</tr>
<tr>
<td>X[3..0]</td>
<td>IN</td>
<td>Channel Link X In</td>
</tr>
<tr>
<td>YCLK</td>
<td>IN</td>
<td>Channel Link X Clock</td>
</tr>
<tr>
<td>YCLK_x1</td>
<td>IN</td>
<td>Channel Link Y Clock used for Cyclone V, Arria V, Stratix V</td>
</tr>
<tr>
<td>YCLK_x7</td>
<td>IN</td>
<td>Channel Link Y Clock x 7 used for Cyclone V, Arria V, Stratix V</td>
</tr>
<tr>
<td>YCLK_x7_2</td>
<td>IN</td>
<td>Channel Link Y Clock x 7 ÷ 2 used for devices other than Cyclone V, Arria V and Stratix V</td>
</tr>
<tr>
<td>Y[3..0]</td>
<td>IN</td>
<td>Channel Link Y In</td>
</tr>
<tr>
<td>ZCLK</td>
<td>IN</td>
<td>Channel Link Z Clock</td>
</tr>
<tr>
<td>ZCLK_x1</td>
<td>IN</td>
<td>Channel Link Z Clock used for Cyclone V, Arria V, Stratix V</td>
</tr>
<tr>
<td>ZCLK_x7</td>
<td>IN</td>
<td>Channel Link Z Clock used for Cyclone V, Arria V, Stratix V</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>ZCLK_x7_2</td>
<td>IN</td>
<td>Channel Link Z Clock x 7 ÷ 2 used for devices other than Cyclone V, Arria V and Stratix V.</td>
</tr>
<tr>
<td>Z[3..0]</td>
<td>IN</td>
<td>Channel Link Z In</td>
</tr>
<tr>
<td>PORT_A[7..0]</td>
<td>OUT</td>
<td>Port A Parallel Data Out</td>
</tr>
<tr>
<td>PORT_B[7..0]</td>
<td>OUT</td>
<td>Port B Parallel Data Out</td>
</tr>
<tr>
<td>PORT_C[7..0]</td>
<td>OUT</td>
<td>Port C Parallel Data Out</td>
</tr>
<tr>
<td>PORT_D[7..0]</td>
<td>OUT</td>
<td>Port D Parallel Data Out</td>
</tr>
<tr>
<td>PORT_E[7..0]</td>
<td>OUT</td>
<td>Port E Parallel Data Out</td>
</tr>
<tr>
<td>PORT_F[7..0]</td>
<td>OUT</td>
<td>Port F Parallel Data Out</td>
</tr>
<tr>
<td>PORT_G[7..0]</td>
<td>OUT</td>
<td>Port G Parallel Data Out</td>
</tr>
<tr>
<td>PORT_H[7..0]</td>
<td>OUT</td>
<td>Port H Parallel Data Out</td>
</tr>
<tr>
<td>PORT_I[7..0]</td>
<td>OUT</td>
<td>Port I Parallel Data Out (80 bit)</td>
</tr>
<tr>
<td>PORT_J[7..0]</td>
<td>OUT</td>
<td>Port J Parallel Data Out (80 bit)</td>
</tr>
<tr>
<td>FVAL</td>
<td>OUT</td>
<td>Frame Valid</td>
</tr>
<tr>
<td>LVAL</td>
<td>OUT</td>
<td>Line Valid</td>
</tr>
<tr>
<td>DVAL</td>
<td>OUT</td>
<td>Data Valid</td>
</tr>
<tr>
<td>STROBE</td>
<td>OUT</td>
<td>Data Strobe signal</td>
</tr>
<tr>
<td>POCL_CLK</td>
<td>IN</td>
<td>PoCL Clock</td>
</tr>
<tr>
<td>POCL_RESET</td>
<td>IN</td>
<td>PoCL Reset</td>
</tr>
<tr>
<td>POCL_PLL_LOCKED</td>
<td>IN</td>
<td>Locked signal from XCLK PLL</td>
</tr>
<tr>
<td>POCL_SENSE1</td>
<td>IN</td>
<td>PoCL Voltage Sense 1</td>
</tr>
<tr>
<td>POCL_SENSE2</td>
<td>IN</td>
<td>PoCL Voltage Sense 2</td>
</tr>
<tr>
<td>POCL_ENA_CURRENT</td>
<td>OUT</td>
<td>PoCL Current Source Enable</td>
</tr>
<tr>
<td>POCL_ENA_12VDC</td>
<td>OUT</td>
<td>PoCL Enable 12V to Camera</td>
</tr>
<tr>
<td>POCL_ENA_GND</td>
<td>OUT</td>
<td>PoCL Enable GND to Camera</td>
</tr>
</tbody>
</table>
Block diagrams of the standard 64-bit and 80-bit Camera Link Full Transmitter core variants are shown in the following three figures. The Camera Link Full Transmitter signals are listed in Table 2 below.
Figure 6: Block diagram of 80-bit Camera Link Full Deca Transmitter Serializer
Figure 7: Block diagram of 80-bit Camera Link Full Octo Transmitter Serializer
## Table 2: Channel Link Full Transmitter signal assignments

<table>
<thead>
<tr>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RST</td>
<td>IN</td>
<td>Active high reset input</td>
</tr>
<tr>
<td>CLK_IN</td>
<td>IN</td>
<td>Clock input</td>
</tr>
<tr>
<td>CLK_IN_x7</td>
<td>IN</td>
<td>Clock input x7, used for devices other than Cyclone V, Arria V, Stratix V.</td>
</tr>
<tr>
<td>CLK_IN_x7_2</td>
<td>IN</td>
<td>Clock input x7_2, used for devices other than Cyclone V, Arria V, Stratix V.</td>
</tr>
<tr>
<td>CLK_IN_x1</td>
<td>IN</td>
<td>Clock input, used for Cyclone V, Arria V, Stratix V.</td>
</tr>
<tr>
<td>XCLK</td>
<td>OUT</td>
<td>X Clock Output</td>
</tr>
<tr>
<td>X[3..0]</td>
<td>OUT</td>
<td>LVDS Channel X Out</td>
</tr>
<tr>
<td>YCLK</td>
<td>OUT</td>
<td>Y Clock Output</td>
</tr>
<tr>
<td>Y[3..0]</td>
<td>OUT</td>
<td>LVDS Channel Y Out</td>
</tr>
<tr>
<td>ZCLK</td>
<td>OUT</td>
<td>Z Clock Output</td>
</tr>
<tr>
<td>Z[3..0]</td>
<td>OUT</td>
<td>LVDS Channel Z Out</td>
</tr>
<tr>
<td>PORT_A[7..0]</td>
<td>IN</td>
<td>Port A Parallel Data In</td>
</tr>
<tr>
<td>PORT_B[7..0]</td>
<td>IN</td>
<td>Port B Parallel Data In</td>
</tr>
<tr>
<td>PORT_C[7..0]</td>
<td>IN</td>
<td>Port C Parallel Data In</td>
</tr>
<tr>
<td>PORT_D[7..0]</td>
<td>IN</td>
<td>Port D Parallel Data In</td>
</tr>
<tr>
<td>PORT_E[7..0]</td>
<td>IN</td>
<td>Port E Parallel Data In</td>
</tr>
<tr>
<td>PORT_F[7..0]</td>
<td>IN</td>
<td>Port F Parallel Data In</td>
</tr>
<tr>
<td>PORT_G[7..0]</td>
<td>IN</td>
<td>Port G Parallel Data In</td>
</tr>
<tr>
<td>PORT_H[7..0]</td>
<td>IN</td>
<td>Port H Parallel Data In</td>
</tr>
<tr>
<td>PORT_I[7..0]</td>
<td>IN</td>
<td>Port I Parallel Data In (80 bit)</td>
</tr>
<tr>
<td>PORT_J[7..0]</td>
<td>IN</td>
<td>Port J Parallel Data In (80 bit)</td>
</tr>
<tr>
<td>FVAL</td>
<td>IN</td>
<td>Frame Valid</td>
</tr>
<tr>
<td>LVAL</td>
<td>IN</td>
<td>Line Valid</td>
</tr>
<tr>
<td>DVAL</td>
<td>IN</td>
<td>Data Valid</td>
</tr>
</tbody>
</table>
Design Flow

The following steps describe how to integrate the **Camera Link IP core** in a Quartus project.

- Open Windows command prompt.
- Browse to the Camera Link wizard directory `<install_dir>/wizard`.
- Start the wizard by typing `java –jar mtx_camera_link_gui.jar`
- Alternatively, start the wizard from the Microtronix->Camera Link Transceiver program group in the Start menu

![Figure 8: Wizard Overview](image-url)
• Click on the Project tab.
• Use the browse button to select a new project or load an existing project.
• Select the appropriate FPGA device family and speed grade.

![Project Tab](image)

*Figure 9: Project Tab*
Click on the Camera Link tab to select the Camera Link settings.
- Select the desired IP Architecture, either Frame Grabber or Camera.
- Select the link size required by the design: Base, Medium or Full.
- When Full select Normal, Deca or Octo

**Figure 10: Camera Link Tab**
- When a family V series FPGA is selected Alternate I/O is selected. The Frame Grabber only allows Alternate I/O to be used. In Camera mode you can select to use regular I/O. Using regular I/O allows for a more relaxed pin assignment.
- If Power Over Camera Link SafePower is enabled, click on the PoCL tab to adjust its settings.
- Enter the frequency of the clock that will be connected to the POCL_CLK input of the IP.
- If necessary, adjust the remaining parameters for your SafePower hardware. The default values have been tested on the Microtronix Camera Link Receiver Board.
Once all the appropriate options are selected, click on the Generate button to start the Camera Link IP core generation.

- The wizard writes a top level Camera Link entity.
- Start Quartus II and open the project.
- Add Camera Link component to the project and connect the signals.
- Add the directory `<install_dir>/synthesis` to the Quartus user libraries (Assignments -> Settings -> User Libraries).
- Start the compilation.

**Clock Generation**

The clocks required by the Camera Link IP Core are dependent on the FPGA family being used. For Cyclone V, Arria V, and Stratix V devices, three PLL generated clocks are used for alternate I/O function (the receiver is always alternate I/O). For all other devices two PLL generated clocks are required. The following sections describe clock generation.

**Cyclone III & IV, Stratix III & IV and Arria II Devices and Family V Transmitter Non-Alternate I/O**

A PLL is required to generate two clocks for the Camera Link IP. The settings vary slightly between a camera (transmitter) and frame grabber (receiver) configuration. In both cases, two clocks are required: firstly the base pixel clock and second the LVDS bit clock.
The multiplication and division factors of the transmitter PLL base pixel clock will depend upon the available clock inputs to the FPGA and the desired pixel frequency. The LVDS bit clock’s multiplication factor must always be 7 times that of the pixel clock and its division factor must be 2 times that of the pixel clock. For example, if the input clock to the FPGA was 27MHz and the desired pixel clock was 74.25MHz, the multiplication factor would be 11 and the division factor would be 4 (74.25 = 27 × 11 ÷ 4) and the duty cycle is 43%. The multiplication factor for the LVDS bit clock is 3.5 times the pixel clock (i.e. ×7÷2) and would therefore be 77 (11 × 7) divided by 8 (4 × 2). See Figure 12 for an example transmitter setup.

Figure 12: Camera Link LVDS SerDes Transmitter

The multiplication and division factors of the receiver PLL will always be fixed. The base pixel clock will have a multiplication factor of 1 and a division factor of 1. The LVDS bit clock will have a multiplication factor of 7 and a division factor of 2. Both output clocks on the receive PLL must be shifted by one bit position to ensure correct data capture. This is most easily accomplished by setting a phase shift of 51.43 degrees and a duty cycle of 43% for the pixel clock and a phase shift of 180 degrees for the LVDS bit clock. See Figure 13 for an example LVDS receiver setup.

Figure 13: Camera Link LVDS SerDes Receiver

Under “Operation Mode” in the PLL MegaWizard, the second clock (c1) must be selected as the clock to be compensated for. This is required for both transmitter and receiver PLLs.
On the Receiver side, the input to the PLL is the clock from the LVDS transmitter. The PLL input clock pin on the FPGA must be a dedicated clock (i.e. the general function of the pin must be that of a clock).

**Cyclone V, Arria V, and Stratix V Devices Alternate I/O**

A PLL is required to generate three clocks for the Camera Link IP. Different PLL settings are used for the frame grabber (receiver) and the camera (transmitter), but both require a pixel clock, an LVds bit clock, and a data clock.

The transmitter PLL should use SOURCE SYNCHRONOUS operation mode. To configure the transmitter PLL settings, first select the desired camera link operating frequency. The Camera Link standard specifies a permissible frequency range of between 20 MHz to 85 MHz. The first PLL output clock is the LVds bit clock. The frequency of this clock is 7 times the camera link frequency, the phase shift is 180 degrees and the duty cycle is 50%. The second PLL output is the data clock. The frequency of the data clock is the same as the selected camera link frequency, the duty cycle is 14% and the phase shift is 0 degrees. The third PLL output generates the camera link pixel clock. The PLL settings are the same as the data clock except that the duty cycle is 50%.

Use the Altera MegaWizard Plug-In Manager to create the PLL. Set the operation mode and enter the required frequency, phase shift, and duty cycle for the three clocks.

![Diagram of PLL settings](image)

**Figure 14: Camera Link 60 MHz Transmitter Example**

For the Receiver, the input to the PLL is the clock from the camera link interface. The PLL input clock pin on the FPGA must be a dedicated clock (i.e. the general function of the pin must be that of a clock). If the camera link frequency is known then use that frequency as the reference clock when
configuring the PLL, otherwise use the maximum operating frequency expected.

Use the Altera MegaWizard Plug-In Manager to create the PLL. Set the operation mode to SOURCE SYNCHRONOUS and enter the selected reference clock frequency. Next, enter the required frequency, phase shift, and duty cycle for the three clocks. The first receiver PLL output clock is the lVDS bit clock. The frequency of this clock is 7 times the reference clock frequency, the phase shift is 45 degrees and the duty cycle is 50%. The second PLL output is used for the data clock. The frequency of the data clock is the same as the reference clock, the duty cycle is 14% and the phase shift is 180 degrees. The third PLL output generates the camera link pixel clock. The PLL settings are the same as the data clock except that the duty cycle is 50%.

![Figure 15: Camera Link 60 MHz Receiver Example](image)

If the PLL loses its lock to the camera link clock, for example because the camera link signals are interrupted, the PLL may not re-acquire lock automatically. It is recommended to either turn on PLL Auto Reset in the PLL MegaWizard Plug-In Manager, or to supply external reset logic to reset the PLL.

### Assignments

Before starting the compilation in Quartus II the I/O-standard for the Camera Link LVDS receiver inputs and Camera Link LVDS transmitter outputs must be set to LVDS. The LVDS receiver inputs need to have differential termination which can be done externally with a resistor or by setting the differential termination with the assignment editor in Quartus.

### SDC Timing Constraints

**Constraints for Cyclone III & IV, Stratix III & IV and Arria II Devices and Family V Transmitter Non-Alternate I/O**

The Camera Link Receiver requires a number of timing constraints to ensure proper operation. Before entering the Camera Link base clock constraints a couple of calculations must be made.
rising edge time = \( \frac{<\text{LVDS base clock period in ns}>}{7 \times 5} \) 
falling edge time = \( \frac{<\text{LVDS base clock period in ns}>}{7 \times 9} \)

Once the rising and falling edge times of the LVDS base clock have been calculated, they can be entered into the Camera Link receiver constraints. When entering the calculated values into an SDC file, round the numbers to three decimal places.

```tcl
set lvds_base_period <LVDS base clock period in ns>
create_clock -name lvds_rxx_base -period $lvds_base_period \ 
- waveform {<rising edge time> <falling edge time>} [get_ports <LVDS clock input pin>]
derive_pll_clocks

set lvds_rx_bit_clk {<TimeQuest name of LVDS bit clock PLL output>}
set lvds_rx_data {<space separated list of LVDS receive data pins>}
create_clock -name lvds_rx_ddr_clk -period [expr {$lvds_base_period / 7 * 2}]
set max_tco <maximum tCO of LVDS source in ns>
set min_tco <minimum tCO of LVDS source in ns>
set skew <absolute value of maximum clock to data skew on board>
set_input_delay -clock lvds_rx_ddr_clk -max [expr {$max_tco + $skew}] \ 
[get_ports $lvds_rx_data]
set_input_delay -clock lvds_rx_ddr_clk -min [expr {$min_tco - $skew}] \ 
[get_ports $lvds_rx_data]
set_input_delay -clock lvds_rx_ddr_clk -max [expr {$max_tco + $skew}] \ 
[get_ports $lvds_rx_data] -clock_fall -add_delay
set_input_delay -clock lvds_rx_ddr_clk -min [expr {$min_tco - $skew}] \ 
[get_ports $lvds_rx_data] -clock_fall -add_delay

set_false_path -setup -rise_from [get_clocks lvds_rx_ddr_clk] -fall_to \ 
[get_clocks $lvds_rx_bit_clk]
set_false_path -setup -fall_from [get_clocks lvds_rx_ddr_clk] -rise_to \ 
[get_clocks $lvds_rx_bit_clk]
set_false_path -hold -rise_from [get_clocks lvds_rx_ddr_clk] -rise_to \ 
[get_clocks $lvds_rx_bit_clk]
set_false_path -hold -fall_from [get_clocks lvds_rx_ddr_clk] -fall_to \ 
[get_clocks $lvds_rx_bit_clk]
```

The LVDS transmitter does not require any timing constraints for proper operation. It relies on the FPGAs dedicated output registers to generate the clock and data properly aligned with each other. To avoid unconstrained path warnings, the following constraint, with the correct transmit pin names substituted, can be used:

```tcl
set_false_path -from * -to [get_ports {<space separated list of LVDS transmit pins>}] 
```

For example timing constraints, see the SDC file included with the Quartus II reference design project.
Constraints for Cyclone V, Arria V, and Stratix V Devices Alternate I/O

The camera link receiver requires clock assignments for each camera link input clock. Input delay constraints can optionally be used to improve the timing analysis by taking into account delay and skew in the camera link signals.

Before entering the Camera Link base clock constraints a couple of calculations must be made.

\[
\text{rising edge time} = \frac{\text{LVDS base clock period in ns}}{7 \times 5} \\
\text{falling edge time} = \frac{\text{LVDS base clock period in ns}}{7 \times 9}
\]

Once the rising and falling edge times of the LVDS base clock have been calculated, they can be entered into the Camera Link receiver constraints. When entering the calculated values into an SDC file, round the numbers to three decimal places.

```
set lvds_base_period <LVDS base clock period in ns>
create_clock -name lvds_rx_base -period $lvds_base_period \ 
    -waveform {<rising edge time> <falling edge time>} \ 
        [get_ports <LVDS clock input pin>] 
derive_pll_clocks
derive_clock_uncertainty
set lvds_rx_data {<space separated list of LVDS receive data pins>}

set max_tco <maximum tCO of LVDS source in ns>
set min_tco <minimum tCO of LVDS source in ns>
set skew <absolute value of maximum clock to data skew on board>
set lvds_rx_data {<space separated list of LVDS receive data pins>}

create_clock -name camera_link_v_clock -period [expr {$lvds_base_period / 7}] 
set_input_delay -clock camera_link_v_clock -max [expr {$max_tco + $skew}] \ 
    [get_ports $camera_link_rx_data]
set_input_delay -clock camera_link_v_clock -min [expr {$min_tco - $skew}] \ 
    [get_ports $camera_link_rx_data]
```

The LVDS transmitter does not require any timing constraints for proper operation. It relies on the FPGAs dedicated output registers to generate the clock and data properly aligned with each other.

Timing Closure Recommendations

1) Create assignments to use global clock networks for the camera link PLL generated clocks. An assignment may cause the fitter to assign a more optimal global clock network as compared to allowing the fitter to automatically promote the clocks to global networks.

2) The choice of LVDS pins has an effect on the maximum clock rate for which timing closure is possible. All pins should be in the same bank. A test design can be created to allow Quartus to select optimal pins before board layout.
Table 3 shows the typical size in logic elements (LE) for the various Camera Link IP Core configurations (including the LVDS core). The actual number of logic elements may vary depending on the device family and Quartus settings.

### Table 3: IP Core FPGA Resource requirements

<table>
<thead>
<tr>
<th>Module</th>
<th>LE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera Link Base Receiver</td>
<td>280</td>
</tr>
<tr>
<td>Camera Link Medium Receiver</td>
<td>560</td>
</tr>
<tr>
<td>Camera Link Full Receiver</td>
<td>840</td>
</tr>
<tr>
<td>Camera Link Base Transmitter</td>
<td>290</td>
</tr>
<tr>
<td>Camera Link Medium Transmitter</td>
<td>580</td>
</tr>
<tr>
<td>Camera Link Full Transmitter</td>
<td>870</td>
</tr>
</tbody>
</table>
Simulation

A precompiled simulation library is provided for performing simulations using ModelSim. The library is located in the `<install_dir>/simulation` directory. Perform the following steps to simulate your design with the video LVDS modules.

1. Launch ModelSim
2. Map the sdram memory controller library. At the ModelSim prompt type:
   ```
   vmap mtx_camera_link
   <install_dir>/simulation/mtx_camera_link
   ```
   If you use a newer version of ModelSim, you must refresh the precompiled library. At the ModelSim prompt type;
   ```
   vcom -refresh -work mtx_camera_link
   ```
3. Compile all of the design files
4. Start the ModelSim simulation by typing;
   ```
   vsim -t ps -L mtx_camera_link <top_level>
   ```

Verification

The Camera Link IP Core has been verified on Microtronix and Altera FPGA development boards. Table 4 shows the hardware platforms and FPGA devices the core has been tested on.

<table>
<thead>
<tr>
<th>Development Board</th>
<th>Altera Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>ViClaro III</td>
<td>EP3C120F780C7</td>
</tr>
<tr>
<td>Altera Cyclone IV GX</td>
<td>EP4CGX150DF31C7</td>
</tr>
<tr>
<td>ViClaro IV</td>
<td>EP4CGX110DF31C7</td>
</tr>
<tr>
<td>Altera Stratix V</td>
<td>EP4SGX230KF40C2N</td>
</tr>
<tr>
<td>Altera Arria V</td>
<td>5AGXF8B3H4F35C5</td>
</tr>
<tr>
<td>Altera Cyclone V</td>
<td>5CGXFC7D6F31C7ES</td>
</tr>
</tbody>
</table>
Installations

Follow these steps to install the Microtronix **Camera Link Transceiver IP Core** module on your computer.

1. Insert the Microtronix **Camera Link Installation CD** into your CD-ROM (or equivalent)

2. The setup program for the package should start. If it doesn’t, browse to the CD using Windows Explorer and double-click on the setup icon.

3. Follow all the prompts. The setup program will attempt to auto-detect the installation location of the Quartus II. Please correct the specified paths if the setup program doesn’t or incorrectly detects them.

IP Core License

The Camera Link IP Core may be supplied with either an OpenCores Plus Evaluation license or a Full Node Locked or a Floating Server license.

**OpenCore Plus Evaluation License**

An OpenCore Plus Evaluation license enables you to design and evaluate your design in circuit on a hardware test platform. Microtronix requires the customer NIC or Guard ID (from a Server or PC workstation) in order to generate an Evaluation license to support OpenCore Plus compilation.

To generate an Evaluation license, Microtronix requires one of two things:

1. Your Altera Software Guard ID (dongle), this is a 9-digit number starting with T. (Example: T000012345) or.

2. Your 12-digit Network MAC Address (Example: 0123456789AB)

Your NIC number is a 12-digit hexadecimal network card number that identifies the Windows workstation serving the Quartus II Web Edition license. You can find the NIC number by typing `ipconfig /all` at the command prompt. Your NIC number is the number on the physical address line, minus the dashes, for example, 00C04FA392EF.

Once either are received, Microtronix will send you the license file(s) to enable Quartus to generate a .sof file for you to run on your target board.

**Installing the Microtronix IP Core license**

To install an IP Core license, follow these steps:

1) Run the Altera Quartus II program and from the menu select > Tools > License Setup. This menu gives the location of the folder and name of the master license file used by Quartus. For example: `C:\altera\licenses\T000085155.dat`.

2) Open this license file with a text editor (i.e. Notepad).

3) In a separate text editor window, open the `license_filename.dat` file provided by Microtronix.

4) Select all of the text in the Microtronix license file.
5) Paste this text into the Altera license.dat file at the end of the file.

6) If it is a Server License, you may need to edit the Server Name or TCP Port in the header per notes in Server License.

7) Save this file and close the text editor.

8) Return to Quartus and start your project. For more information, please visit http://altera.com/literature/an/an340.pdf

**Full IP Core License**

To generate .pof program files incorporating the Camera Link IP core requires the user to have purchased a Full IP Core License. These licenses are generated by Microtronix based on a NIC or Guard ID supplied by the user. They can be supplied as either a multi-user Floating Server or a single user Node Locked PC workstation license.

After purchasing a Full License you receive your license file. Copy the license file (license.dat) to your current Quartus license file and the LVDS core (CC21_6246) will show in the Quartus License Setup (Tools->License Setup).

Please contact sales at Microtronix (sales@microtronix.com) for additional licensing details.