This user guide provides basic information about using the Microtronix Video LVDS Transmitter / Receiver IP, PN: 6246-xx-xx. The following table shows the document revision history.

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>April 2007</td>
<td>Initial Release – Version 1.0</td>
</tr>
<tr>
<td>September 2007</td>
<td>Minor updates – Version 1.1</td>
</tr>
<tr>
<td>November 2007</td>
<td>Minor updates – Version 1.2</td>
</tr>
<tr>
<td>January 2008</td>
<td>Added receive timing procedure – Version 1.3</td>
</tr>
<tr>
<td>March 2008</td>
<td>Added procedure to integrate the Core in a Quartus project – Version 1.4</td>
</tr>
<tr>
<td>February 2011</td>
<td>Updated for TimeQuest – Version 2.0</td>
</tr>
<tr>
<td>August 2011</td>
<td>Added Stratix III, Stratix IV and Arria devices - Version 2.1</td>
</tr>
<tr>
<td>July 2013</td>
<td>Added Cyclone V, Stratix V and Arria V devices - Version 2.2</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>
</tbody>
</table>

Code

`→` Indicates that there is no break between the current line and the next line.
Features

- 7:1 Serializer / Deserializer
- Single core supports transmitter and receiver functions
- Video LVDS SerDes transmitter and receiver
- Optimized for 8 and 10-bit RGB video applications
- Support for both 28-bit and 35-bit parallel data (mapped into 4/5 LVDS channels)
- Supports single, dual and quad link configurations
- LVDS TxClock aligned to data, no PLL fine tuning required
- Receiver auto aligns clock to encoded video data from input LVDS data stream
- Supports flat panel display resolutions up to 1080p 120 Hz
- Configuration GUI streamlines design process
- Complies with Open LDI Specification for Digital Display Interfaces
- Operation up to 85 MHz in Cyclone III, IV & V low cost devices
- Supports Cyclone and Arria device families and Stratix III & IV FPGA devices
- Support for OpenCore Plus evaluation
- Quartus Reference Design
The Microtronix Video LVDS SerDes Transmitter / Receiver IP Core provides a complete, easy-to-use solution to interface with a wide variety of video host systems and flat panel displays.

The core simplifies the design of video LVDS interfaces, improves data integrity and timing margins. For example, the Transmitter has the ability to generate a LVDS transmit clock synchronous to the video data stream thereby eliminating the need to fine-tune a PLL to the outputted LVDS data. Similarly, the Receiver auto aligns the receive clock to the encoded video to assure data synchronization. An easy to use GUI enables the user to select the number of LVDS links and to configure the number data channels for each link.

The Transmitter serializer converts 28-bit parallel data into 4 LVDS serial data streams and one LVDS clock stream. The Receiver deserializer converts 4 LVDS serial data streams into 28-bit parallel data. Adding a fifth channel increases the data width to 35 bits. Transmitter and Receiver modules can be cascaded to create dual and quad LVDS links.

Device Support

The Video LVDS SerDes Transmitter / Receiver IP Core is optimized for Altera Cyclone II, III, IV, V, Statix III, IV, V and Arria families of programmable logic devices.
**Transmitter**

A block diagram of the Transmitter core is shown in Figure 1. The LVDS transmitter signals are listed in Table 1.

![Block diagram of LVDS Transmit Serializer](image)

**Figure 1: Block diagram of LVDS Transmit Serializer**

**Table 1: LVDS 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_x2</td>
<td>IN</td>
<td>Clock input x7÷2</td>
</tr>
<tr>
<td>TA_IN[6-0]</td>
<td>IN</td>
<td>Channel A Parallel Data In</td>
</tr>
<tr>
<td>TB_IN[6-0]</td>
<td>IN</td>
<td>Channel B Parallel Data In</td>
</tr>
<tr>
<td>TC_IN[6-0]</td>
<td>IN</td>
<td>Channel C Parallel Data In</td>
</tr>
<tr>
<td>TD_IN[6-0]</td>
<td>IN</td>
<td>Channel D Parallel Data In</td>
</tr>
<tr>
<td>TE_IN[6-0]</td>
<td>IN</td>
<td>Channel E Parallel Data In</td>
</tr>
<tr>
<td>TA_OUT</td>
<td>OUT</td>
<td>LVDS Channel A Out</td>
</tr>
<tr>
<td>TB_OUT</td>
<td>OUT</td>
<td>LVDS Channel B Out</td>
</tr>
<tr>
<td>TC_OUT</td>
<td>OUT</td>
<td>LVDS Channel C Out</td>
</tr>
<tr>
<td>TD_OUT</td>
<td>OUT</td>
<td>LVDS Channel D Out</td>
</tr>
<tr>
<td>TE_OUT</td>
<td>OUT</td>
<td>LVDS Channel E Out</td>
</tr>
<tr>
<td>TCLK_OUT</td>
<td>OUT</td>
<td>LVDS Channel CLK Out</td>
</tr>
</tbody>
</table>
**Receiver**

A block diagram of the LVDS Receiver Deserializer core is shown in Figure 2 below. The LVDS Receiver signals are listed in Table 2.

![LVDS Receiver Deserializer Block Diagram](image)

**Figure 2: Block diagram of LVDS Receive Deserializer core**

**Table 2: LVDS 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>CLK_IN</td>
<td>IN</td>
<td>Clock input</td>
</tr>
<tr>
<td>CLK_IN_x2</td>
<td>IN</td>
<td>Clock input x7÷2</td>
</tr>
<tr>
<td>RA_IN</td>
<td>IN</td>
<td>LVDS Channel A In</td>
</tr>
<tr>
<td>RB_IN</td>
<td>IN</td>
<td>LVDS Channel B In</td>
</tr>
<tr>
<td>RC_IN</td>
<td>IN</td>
<td>LVDS Channel C In</td>
</tr>
<tr>
<td>RD_IN</td>
<td>IN</td>
<td>LVDS Channel D In</td>
</tr>
<tr>
<td>RE_IN</td>
<td>IN</td>
<td>LVDS Channel E In</td>
</tr>
<tr>
<td>RA_OUT[6-0]</td>
<td>OUT</td>
<td>Channel A Parallel Data Out</td>
</tr>
<tr>
<td>RB_OUT[6-0]</td>
<td>OUT</td>
<td>Channel B Parallel Data Out</td>
</tr>
<tr>
<td>RC_OUT[6-0]</td>
<td>OUT</td>
<td>Channel C Parallel Data Out</td>
</tr>
<tr>
<td>RD_OUT[6-0]</td>
<td>OUT</td>
<td>Channel D Parallel Data Out</td>
</tr>
<tr>
<td>RE_OUT[6-0]</td>
<td>OUT</td>
<td>Channel E Parallel Data Out</td>
</tr>
</tbody>
</table>
The LVDS Receiver clock input connects to a PLL that is used to generate CLK_IN and CLK_IN_x2 for the LVDS Deserializer. This PLL must be instantiated by the user (see Clock Generation).

**Data Mapping**

Figure 3 below shows the LVDS data mapping for the Transmitter and Receiver.

![LVDS Mapping Diagram](image)

**Figure 3: LVDS Mapping**

**Design Flow**

The following steps describe how to integrate the *LVDS SerDes Transmitter / Receiver IP core* in a Quartus project.

- Open Windows command prompt.
- Browse to the LVDS wizard directory `<install_dir>/wizard`.
- Start the wizard by typing `java -jar mtx_lvds_gui.jar`
- Alternatively, start the wizard from the Microtronix->HD Video LVDS Transmitter Receiver program group in the Start menu.
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 (Cyclone II, Cyclone III, Cyclone III LS, Cyclone IV E, Cyclone IV GX, Stratix III, Stratix IV (GT/CX/E), Arria GX, Arria II GX or Arria II GZ) and speed grade.
• Click on the LVDS tab to select the LVDS settings.

• Select the desired IP Architecture, either Transmitter or Receiver.

• Select the number of links and channels required by the design. Channel value represents the number of LVDS data streams and the Link value represents the number of groups of channels.

![LVDS Tab](image)

**Figure 6: LVDS Tab**

- Once all the appropriate options are selected, click on the Generate button to start the LVDS IP core generation.

- The wizard writes a top level LVDS entity.

- Start Quartus II and open the project.

- Add LVDS component to the project and connect the signals.

- Add the mtx_lvds_package to the project files (Assignments -> Settings -> Files). The mtx_lvds_package is located in the directory `<install_dir>/synthesis`.

- Add the directory `<install_dir>/synthesis` to the Quartus user libraries (Assignments -> Settings -> User Libraries).

- Start the compilation.

**Clock Generation**

A PLL is required to generate the clocks for the LVDS IP. The settings vary slightly between the transmitter and the receiver. In both cases, two clocks are required. The first is the base pixel clock and the second is 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). 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 7 for an example transmitter setup.

Figure 7: LVDS 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 for the pixel clock and 180 degrees for the LVDS bit clock. See Figure 8 for an example LVDS receiver setup.

Figure 8: LVDS 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).

**Assignments**

Before starting the compilation in Quartus II the I/O-standard for the LVDS receiver inputs and LVDS transmitter outputs must be set to LVDS.

**Timing Constraints**

The LVDS receiver requires a number of timing constraints to ensure proper operation. Before entering the LVDS 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 LVDS receiver constraints. When entering the calculated values into an SDC file, round the numbers to three decimal places.

```literate
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

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:

```
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.

**Table 3: FPGA Resource requirements**

<table>
<thead>
<tr>
<th>Module</th>
<th>LE</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVDS Transmitter</td>
<td>255</td>
</tr>
<tr>
<td>LVDS Receiver</td>
<td>260</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_lvds <install_dir>/simulation/mtx_lvds
   ```

   If you use a newer version of ModelSim, you must refresh the precompiled library. At the ModelSim prompt type;

   ```
   vcom -refresh -work mtx_lvds
   ```

3. Compile all of the design files
4. Start the ModelSim simulation by typing;
   
   ```
   vsim -t ps -L mtx_lvds <top_level>
   ```

Verification

The video LVDS module has been verified on Microtronix & Altera development boards. Table 4 shows the hardware tested platforms and supported memory devices.

Table 4: Hardware Platforms

<table>
<thead>
<tr>
<th>Development Board</th>
<th>Altera Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stratix III</td>
<td>EP3SL150F1152</td>
</tr>
<tr>
<td>Arria II GX</td>
<td>EP2AGX125EF35</td>
</tr>
<tr>
<td>Cyclone V SoC</td>
<td>5CSXC6N/ES</td>
</tr>
<tr>
<td>ViClaro III</td>
<td>EP3C120F780C7</td>
</tr>
<tr>
<td>ViClaro II</td>
<td>EP2C35F484C6</td>
</tr>
<tr>
<td>ViClaro</td>
<td>EP2C20F256C7</td>
</tr>
<tr>
<td>Vivien</td>
<td>EP2C5T144C6</td>
</tr>
</tbody>
</table>
Installazione

Follow these steps to install the Microtronix video LVDS modules on your computer.

1. Insert the Microtronix video LVDS modules 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 Video SerDes LVDS Transmitter/Receiver IP Core may be supplied with either a OpenCores Plus Evaluation license of 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\licences\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 Video SerDes Transmitter/Receiver 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 Floating Server or a 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.