LatticeECP3, LatticeECP2/M and LatticeXP2 7:1 LVDS Video Interface

April 2011 Reference Design RD1030

Introduction

Source synchronous interfaces consisting of multiple data bits and clocks have become a common method for moving image data within electronic systems. A prevalent standard is the 7:1 LVDS interface (employed in Channel Link, Flat Link, and Camera Link), which has become a common standard in many electronic products including consumer devices, industrial control, medical, and automotive telematics. In many of these applications, the practice of using low-cost FPGAs for image processing has become quite common. In particular, LatticeXP2™, LatticeECP2™, LatticeECP2M™ and LatticeECP3™ are well-suited to support the 7:1 LVDS standard.

Note: Since the 7:1 LVDS interface is supported in LatticeECP3 "EA" devices, but not the earlier "E" devices, all references to LatticeECP3 in this document refer to the "EA" devices only.

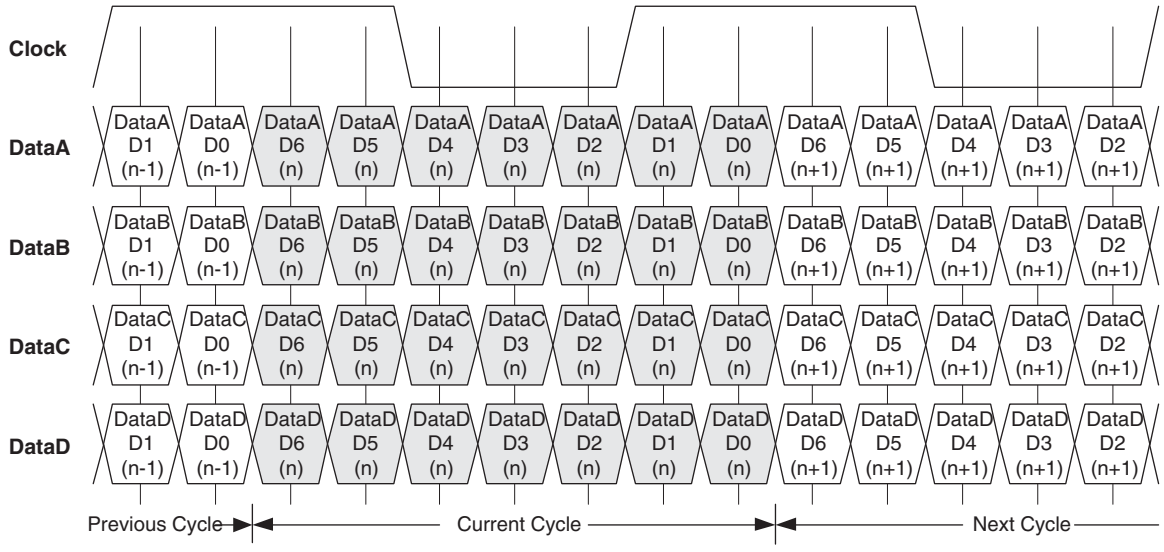
This document describes the requirements for implementing a 7:1 LVDS interface and the advantages of using these FPGAs in such an interface. By extension, support for the 7:1 LVDS interface in these devices proves the feasibility of hardware implementation for all other LVDS source synchronous requirements as well.

Two designs are included in the discussion of this document. The first design is a simple loopback test that illustrates the use of the 7:1 transmitter and 7:1 receiver. The second design is an example that brings video data into the FPGA device through the 7:1 receiver, processes it and transmits it out via the 7:1 transmitter. Both designs are verified using the Lattice 7:1 LVDS Video Demo Kit.

7:1 LVDS Interface Requirement

The 7:1 LVDS interface is a source synchronous LVDS interface. Seven data bits are serialized for each cycle of the low-speed clock as shown in Figure 1. Typically, the interface consists of four (three data, one clock) or five (four data, one clock) LVDS pairs. The four pairs translate to 21 parallel data bits and five pairs translate to 28 parallel data bits. Note that there is a 2-bit offset between the clock rising edge and the word boundary. Each word is 7 bits long.

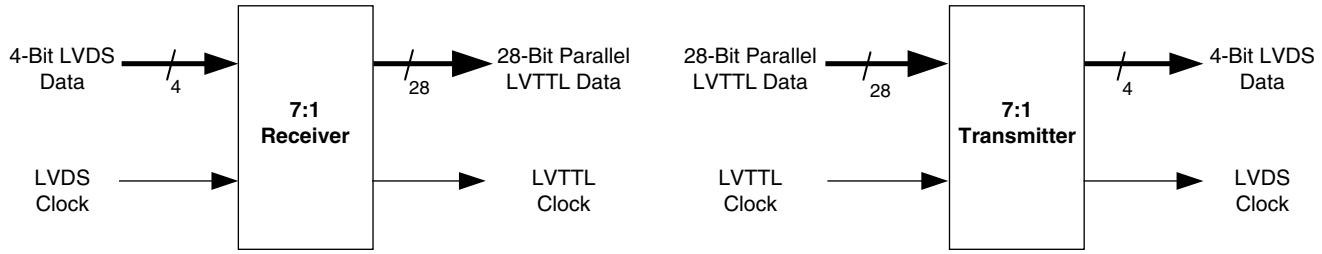
Figure 1. Basic Timing of the 7:1 LVDS Interface



Each channel includes a serial LVDS data pair along with a source synchronous LVDS clock pair. The receiver receives this serial LVDS data, deserializes it and aligns it to the original word boundary to generate seven parallel LVTTL data bits. The 7:1 transmitter serializes the seven LVTTL parallel data bits to a single LVDS data bit and transmits this serial data channel along with a LVDS clock.

Figure 2 shows the 7:1 receiver receiving four LVDS data channels. When deserialized, it generates 28-bit wide parallel data. Similarly, the 7:1 transmitter serializes 28-bit parallel data to generate four LVDS data channels.

Figure 2. 7:1 Receiver and Transmitter Function



The requirements for an FPGA-based solution to the Channel Link and Flat Link style interfaces consist of four key components: high-speed LVDS buffers, a PLL for generating the de-serialization clock, input data capture and gearing, and data formatting.

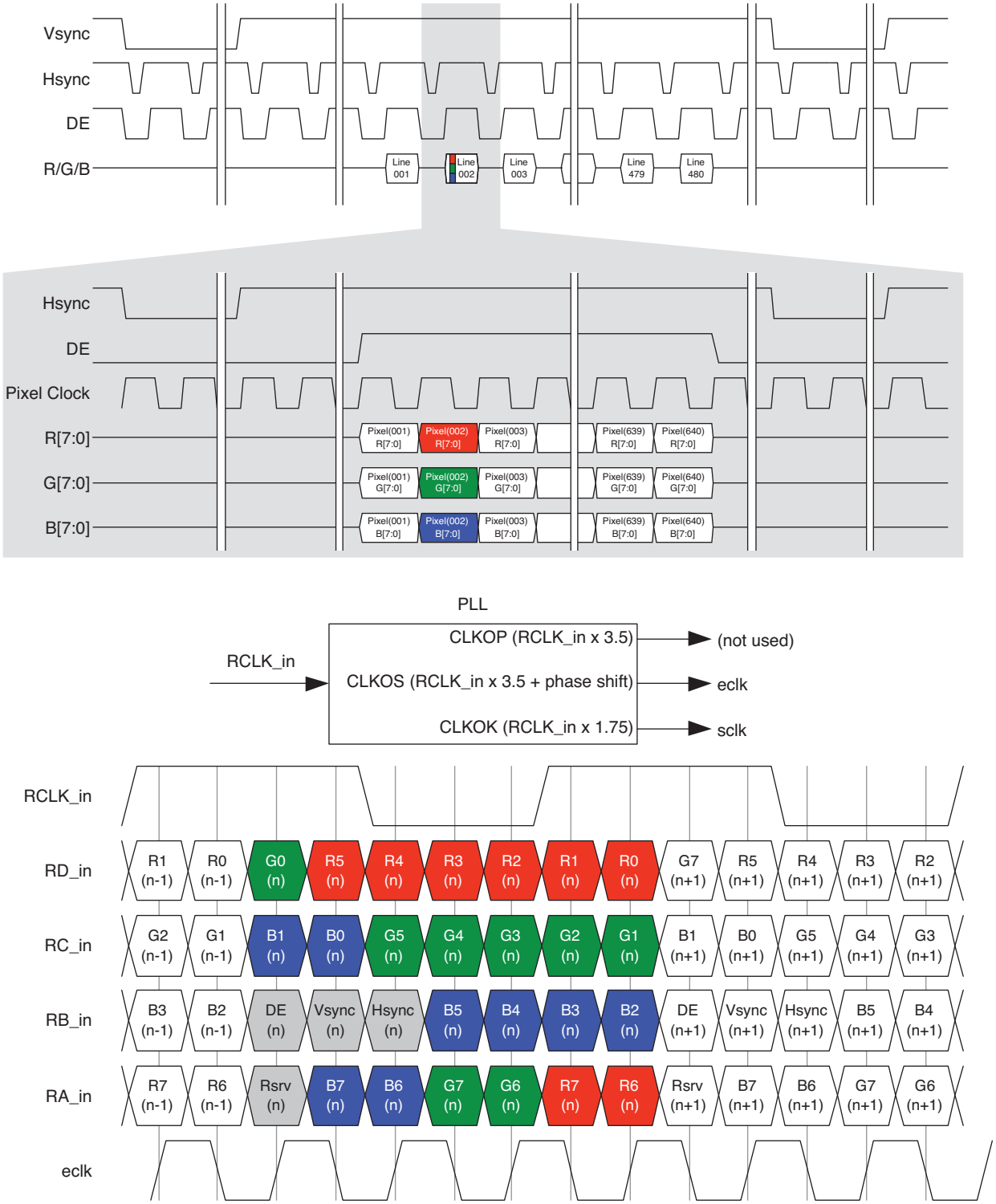
The data and clock are received or transmitted to or from the FPGA in LVDS format, with the data at relatively high speed. The exact speed depends on the resolution, frame rate and color depth used by the display. For example, 800x600 to 1024x768 displays require LVDS data to be transmitted from 40 MHz to 78.5 MHz for 60 Hz to 75 Hz refresh rates. This translates to LVDS data rates of 280 Mbps to 549 Mbps. Higher resolution displays, such as 1280x1024 60 Hz, require data to be transmitted with 108 MHz LVDS clocks. For this system, data will transmit at 756 Mbps.

Clock Generation

In a LatticeECP3, LatticeECP2/M or LatticeXP2 implementation, the input capture circuitry uses Double Data Rate (DDR) registers with data captured on both the rising and falling edges of the clock. When operating as a receiver the low-speed clock that is provided with the data must be multiplied by 3.5 times in order to capture the data on both clock edges. If the input capture circuitry operates on only one edge of the clock, a multiplication factor of seven must be used. As an alternative, seven phase-shifted versions of the low-speed clock can be generated and used to capture the input data with seven different registers. However, the challenges of clock generation and distribution discourage this approach for an FPGA implementation. The clock must have relatively low jitter since its jitter must be accounted for in the overall timing budget. Similarly, the skew of the clock distribution network used to provide this clock to input or output registers must be accounted for in any timing analysis.

In order to transmit high-speed data, a transmitter must multiply the clock used to transfer low-speed parallel data into the interface by 3.5. Again, the jitter of the clock and the skew of its distribution are important as they impact the timing budget for the interface. Figure 3 shows the PLL clock generation and how the R, G, B bits, Vsync, Hsync, and DE of a pixel on line 2 of a video frame get assigned to the four LVDS data pairs. The data bits are sampled on both rising and falling edges of the eclk clock.

Figure 3. Timings of Video Signals and the 7:1 LVDS Channel Link Interface



Data Capture

The registers that follow the LVDS input buffer must accurately capture the data. A tight control of the clock and data relationship is important to capture the incoming high-speed data stream. It is also necessary to gear, or reduce, the speed of the data before it is passed on to the FPGA fabric. Let us take LatticeECP2/M and

LatticeECP3 as examples. LatticeECP2/M FPGAs specify the operation of individual circuit elements to around 350 MHz. For LatticeECP3, it will be around 470 MHz. A practical operating frequency with a reasonable amount of logic is 225 MHz for LatticeECP2/M and 350 MHz for LatticeECP3. Therefore, the greater the gearing that can be done in the I/O structure, the lower the likelihood that the FPGA fabric will be the limit on overall performance. A similar discussion is applicable to the transmit path.

Data Formatting

The final step is to take the data from the I/O cells and format it into the original 7-bit width clocked by the low-speed clock. This logic can easily be constructed within the FPGA fabric.

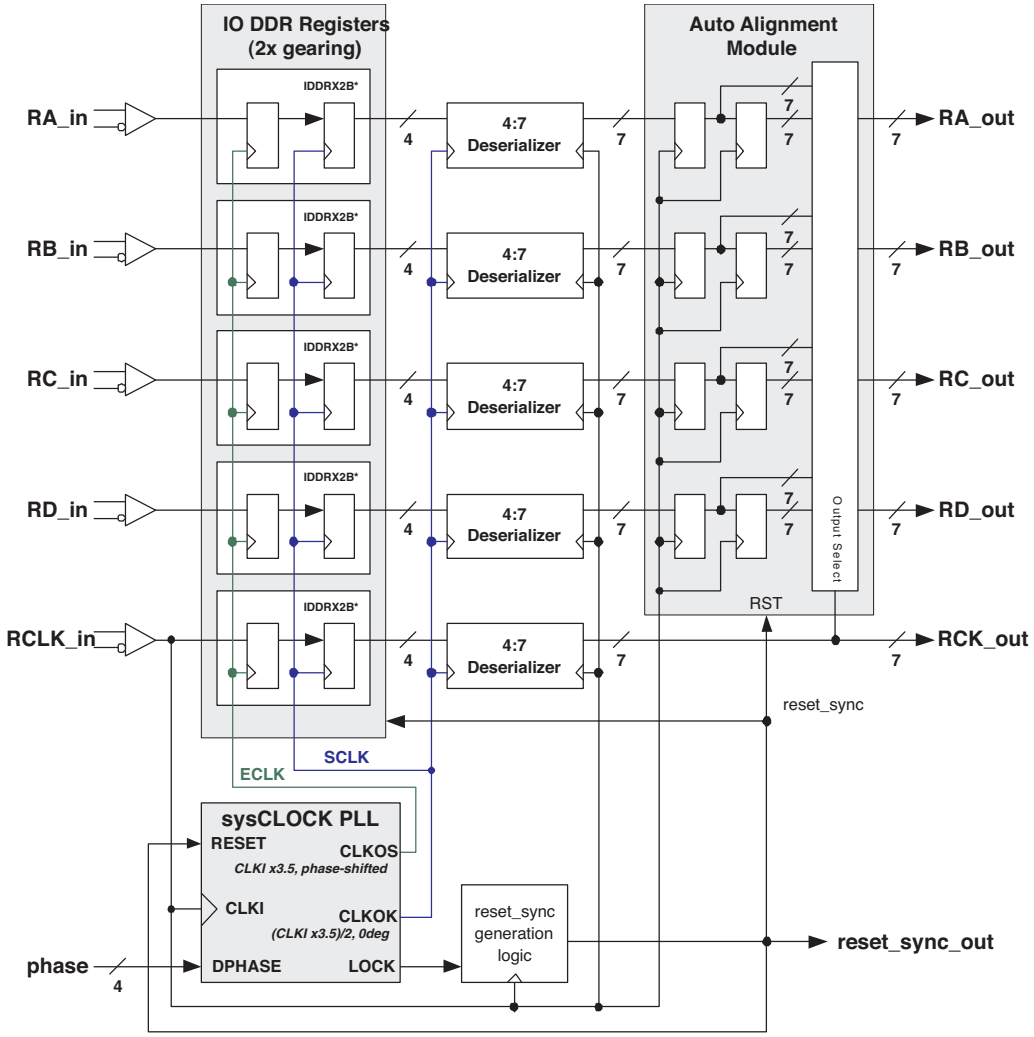
LatticeECP3, LatticeECP2/M and LatticeXP2 7:1 LVDS Interface

The LatticeECP3, LatticeECP2/M and LatticeXP2 architectures provide an ideal solution for this interface. This section describes implementation of the 7:1 receiver and 7:1 transmitter using the LatticeECP3, LatticeECP2/M and LatticeXP2 device I/O structures.

7:1 Receiver

Figure 4 shows the block diagram of the receive side of an intra-system display interface within a LatticeECP3, LatticeECP2/M or LatticeXP2 device. The receiver receives four LVDS data channels (seven bits each) and one LVDS clock.

Figure 4. 7:1 Receiver Side Block Diagram

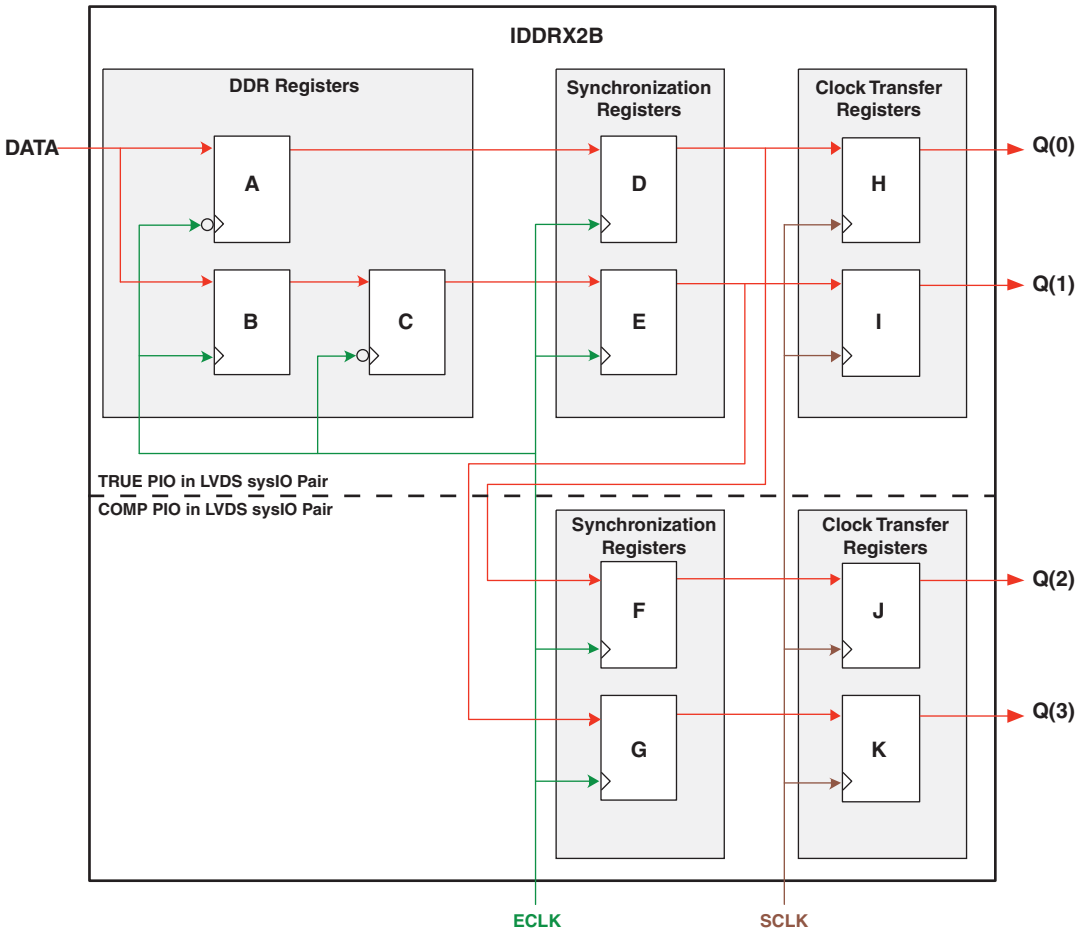


The data and clock enter the LatticeECP3, LatticeECP2/M or LatticeXP2 device through the LVDS buffers in the Programmable I/O Cell (PIC) block. When the 2x gearing function is used, these buffers operate at up to 420 MHz (i.e., 840Mbps) for LatticeXP2 and LatticeECP2/M devices, or 500 MHz (i.e., 1.0Gbps) for LatticeECP3 devices, supporting most high resolution and display refresh rates.

The LVDS data is fed to the I/O logic DDR register and the source synchronous LVDS clock is fed into a PLL. The PLL is used to multiply the clock by 3.5 and create a phase shift which is normally 90 degrees. This phase shift allows for placing the clock in the middle of the data valid window. This faster phase-shifted clock is then distributed via a low skew edge clock net to double data rate input capture registers. The PLL is also used to generate a slower clock that is half the frequency of the faster edge clock. This clock is fed to the second stage of DDR registers in the I/O logic block using the primary clock tree.

The I/O DDR register with the 2x gearing function (IDDRX2B) is used for the design with LatticeXP2 and LatticeECP2/M FPGAs. A 2x DDR element provides four FPGA side data bits for every I/O side data bit at half the clock rate. The gearing allows muxing/demuxing of the I/O data clocked with the high-speed Edge clock (ECLK) to the slower speed FPGA clock rate (SCLK). In the end, all the data is received at the rising edge of SCLK. Figure 5 is a detailed diagram of the IDDRX2B.

Figure 5. IDDRX2B Detailed Block Diagram

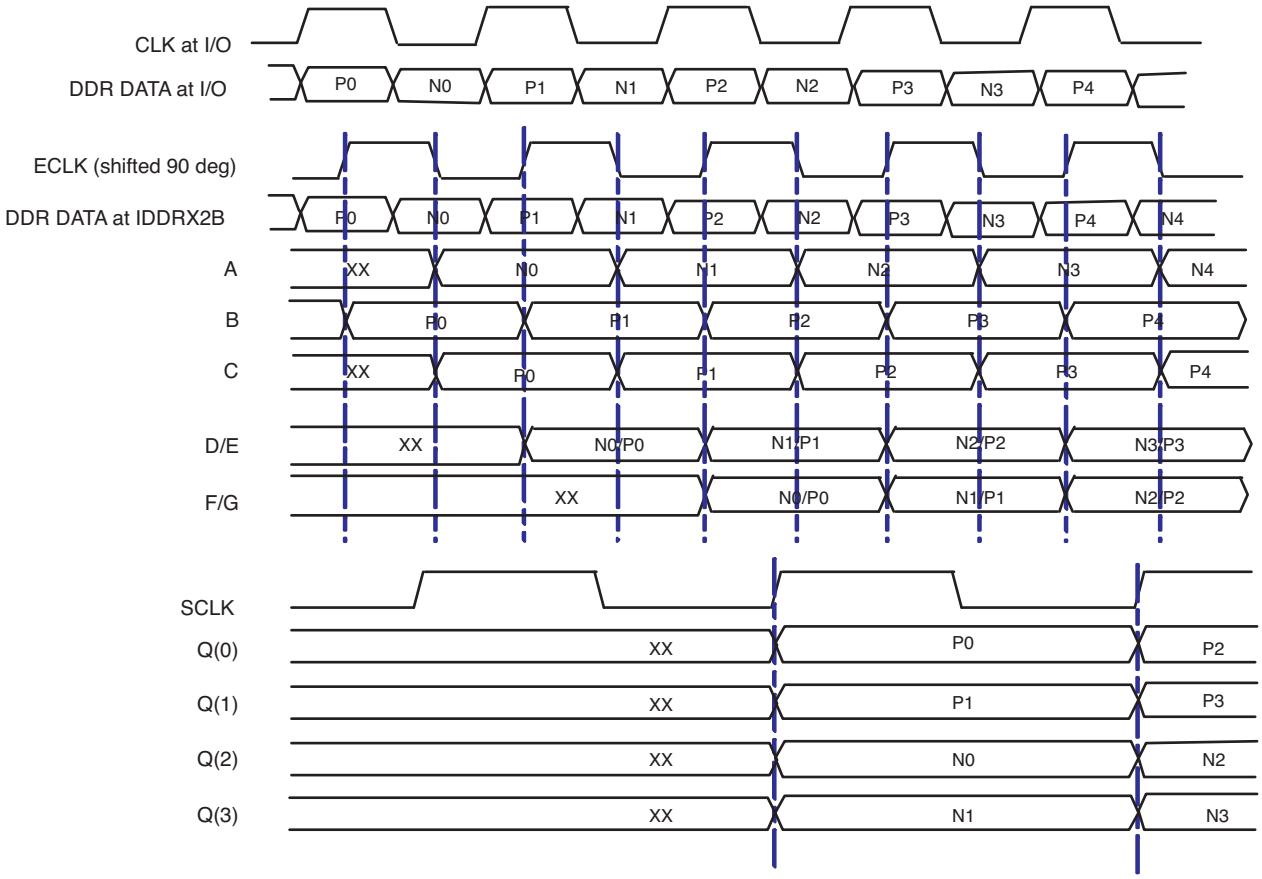


The IDDRX2B module inputs the DDR data at both edges of the Edge clock and generates four streams of data, all at the rising edge of the slower FPGA clock. The shaded portion of Figure 5 shows the I/O registers used to do the 2x gearing mode. The I/O registers of the complementary PIO are used in DDR gearing mode. For more informa-

tion on the DDR registers and the various modes, refer to TN1105, LatticeECP2/M High-Speed I/O Interface, TN1138, LatticeXP2 High-Speed I/O Interface and TN1180, LatticeECP3 High-Speed I/O Interface.

Figure 6 shows an example of input gearing using the IDDRX2B block.

Figure 6. Example of Input Gearing Using IDDRX2B



The four bits of parallel data are then converted to 7-bit data at the correct speed in the 4:7 deserializer module. The deserializer stores the 4-bit output of the IDDRX2B in a 28-bit wide shift register. The incoming LVDS clock is then used as a framing signal to detect the start and end of the 7-bit data frame. The ordering of the 7-bit data can be modified in the design files if required.

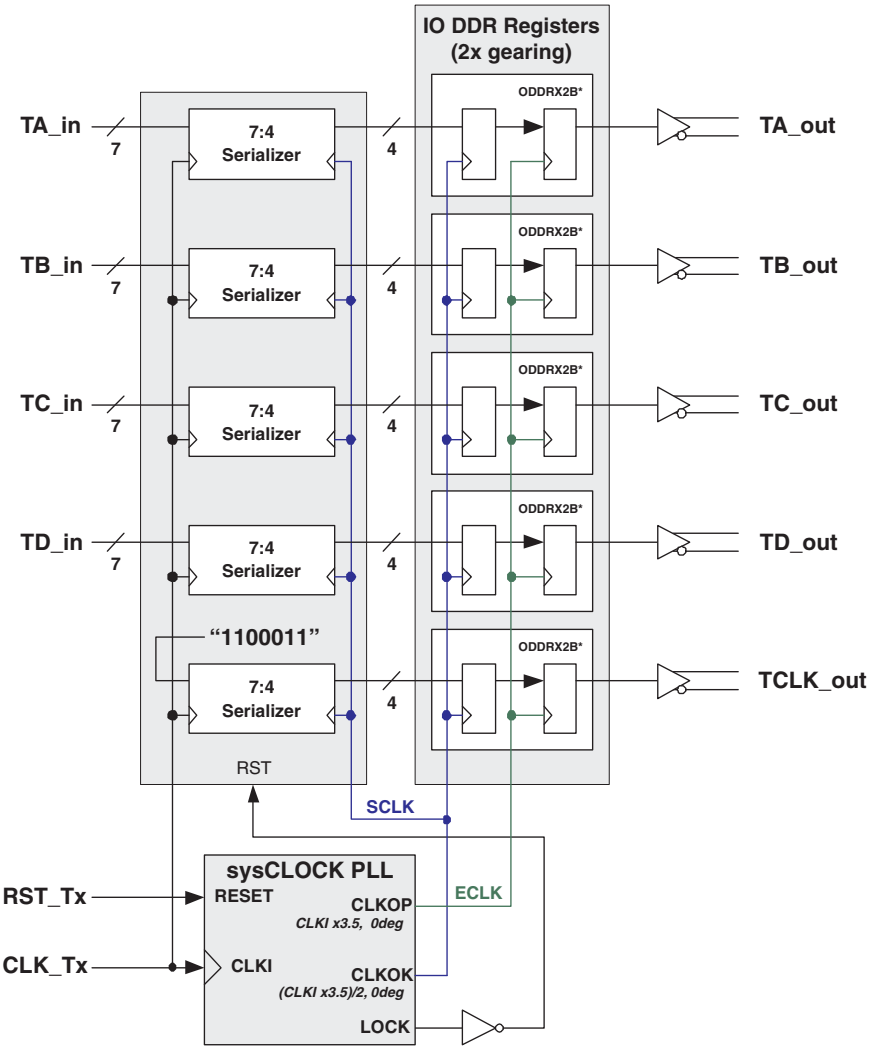
7:1 Transmitter

Figure 7 shows the transmit side of the 7:1 implementation. In this case, the LatticeECP3, LatticeECP2/M or LatticeXP2 device receives four channels of 7-bit parallel data and the slow clock. All 28 bits of parallel data are aligned to the slow clock received.

The slow input clock is fed to the PLL. The PLL is used to multiply the clock 3.5 times (ECLK). The PLL is also used to generate a clock at half the frequency of the 3.5x clock. This clock is represented by the SCLK in Figure 7. The DDR register in the I/O Logic module is used to generate the serial data output. LatticeXP2 and LatticeECP2/M FPGAs support output DDR register modules with 2x gearing similar to the input DDR registers. The advantage of using the output DDR registers with the 2x gearing (ODDRX2B) over 1x gearing is that the FPGA core can run at half the speed of the clock used by the output DDR registers

The seven bits of parallel data need to be converted to four bits of serial data before they are sent to each of the output DDR registers. The 7:4 Serializer module is used to do this. Each of the seven bits of parallel data is stored in a 28-bit wide buffer and four bits of data aligned to the SCLK clock are sent to the ODDRX2B module.

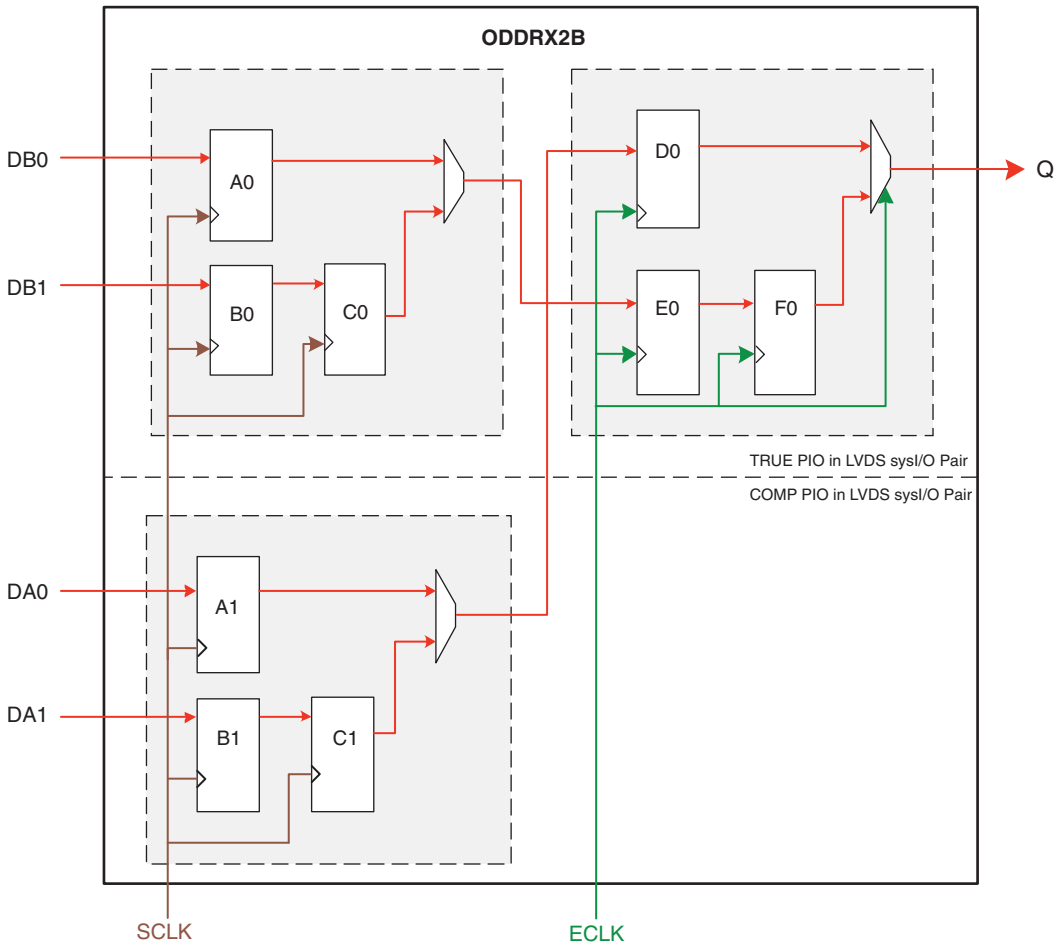
Figure 7. 7:1 Transmitter Side Block Diagram



The ODDRX2B also receives the faster ECLK from the PLL and performs the gearing function. The gearing allows multiplexing of the I/O data clocked with the slow-speed FPGA Clock (SCLK) to the high-speed Edge clock. All of the data is transmitted at the rising edge of the ECLK.

Figure 8 shows a detailed diagram of the ODDRX2B.

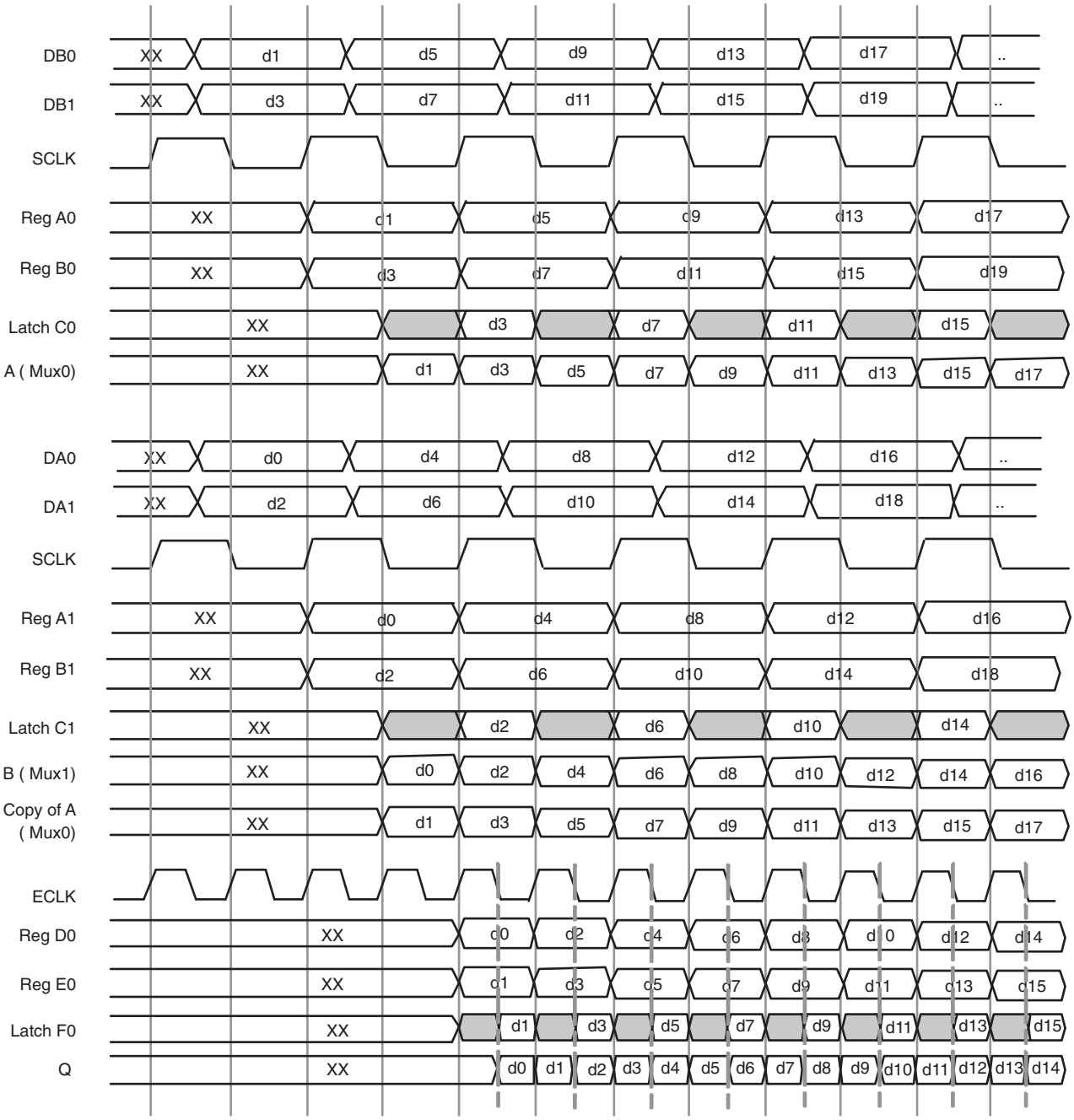
Figure 8. ODDRX2B Gearing Function



The ODDRX2B module inputs come from the four bits of data from the FPGA fabric at both edges of the slow FPGA Clock (SCLK). These inputs also generate a single stream of data at both edges of the faster Edge Clock (ECLK). The shaded portion of Figure 8 shows the I/O registers used in 2x gearing mode. The I/O registers of the complementary PIO are used in DDR 2x Gearing mode. For more information on the DDR registers and various modes, refer to TN1105, LatticeECP2/M High-Speed I/O Interface, TN1138, LatticeXP2 High-Speed I/O Interface and TN1180, LatticeECP3 High-Speed I/O Interface.

Figure 9 shows an example of input gearing using the ODDRX2B block.

Figure 9. Example of Output Gearing Using ODDRX2B



The serialized data output of the ODDRX2B is sent out of the device using high-speed LVDS buffers.

The LatticeECP3 I/O structure is different from that of the LatticeXP2 and LatticeECP2/M devices. The DQSBUF primitive (e.g., DQSBUFE for 2x gearing) has to be used to generate the strobe logic and delay used in the output DDR modules to correctly mux the DDR data. This DQSBUF primitive is required for the outputs of generic DDR implementations such as 7:1 LVDS. Since all I/Os in a DQS group share the same DQSBUF, it is recommended to group as many I/Os of the same 7:1 LVDS bus as possible within one DQS group. Since each DQS group includes only a limited number of True LVDS pins (normally two I/Os per DQS group), if True LVDS I/Os are used for 7:1 LVDS outputs, more DQSBUF primitives will be required to span the True LVDS outputs to adjacent DQS groups. Note that this does not apply to the design using emulated LVDS outputs. Also, the I/O DDR primitives in

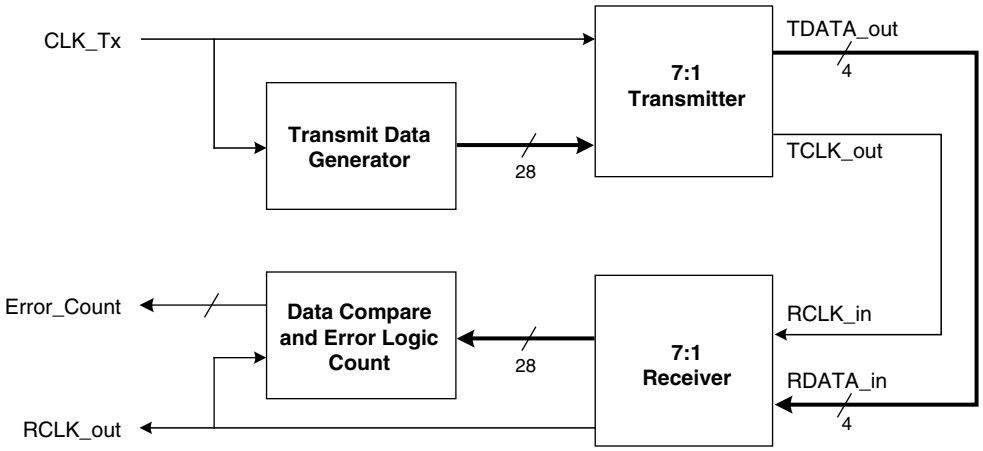
LatticeECP3 devices (IDDRX2D1/ODDRX2D) are different from those in LatticeXP2 and LatticeECP2/M devices (IDDRX2B/ODDRX2B) in port definitions. For more detailed information, see TN1180, LatticeECP3 High-Speed I/O Interface.

Design Example 1: Loopback Test

The loopback test design included with this document uses the Lattice FPGA to implement both the 7:1 transmitter and receiver. Figure 10 shows the design implementation. For more detailed information about the 7:1 transmitter and receiver, refer to Figures 4 and 7.

28-bit transmit data is generated in the FPGA logic using counter values. This data is then serialized and transmitted as four bits of LVDS data using the 7:1 transmitter logic. The 4-bit LVDS data is then looped back into the LatticeECP3, LatticeECP2/M or LatticeXP2 device receiver side and deserialized using the 7:1 receiver logic. This deserialized data is then fed to the data compare logic module which compares the deserialized receiver data to the original counter values transmitted. The error count is increased at every mismatch detected between the two data values. The 7:1 transmitter and receiver logic is explained in detail in the sections above.

Figure 10. Loopback Test Block Diagram



Loopback Test Implementation Results

The loopback design was tested using the LatticeECP2 Advanced Evaluation Board, the LatticeXP2 Advanced Evaluation Board and the LatticeECP3 Video Protocol Board. Both the Lattice FPGA transmit and receive sides were successfully run at 108 MHz transmit and receive pixel clock for LatticeECP3, LatticeECP2/M and LatticeXP2. For LatticeECP3, it can run up to 135 MHz. Table 1 shows the resources utilized by the design.

Table 1. Loopback Test Design Performance and Resource Utilization

Device Family	Language	Speed Grade	Utilization (LUTs)	f_{MAX} (MHz)	I/Os	Slices	Registers	sysMEM EBRs	sysDSP Blocks
LatticeECP3[1]	VHDL	-7	832 (1%)	>108	36	771	910	0 (0%)	0 (0%)
	Verilog		819 (1%)	>108	36	766	916	0 (0%)	0 (0%)
LatticeECP2/M[2]	VHDL	-6	858 (2%)	>108	36	794	914	0 (0%)	0 (0%)
	Verilog		834 (2%)	>108	36	778	916	0 (0%)	0 (0%)
LatticeXP2[3]	VHDL	-6	839 (5%)	>108	36	785	916	0 (0%)	0 (0%)
	Verilog		825 (5%)	>108	36	774	915	0 (0%)	0 (0%)

1. Performance and utilization characteristics are generated using LFE3-95EA-7FN1156C with Lattice Diamond™ 1.2 design software. When using this design in a different device, density, speed, or grade, performance and utilization may vary.
2. Performance and utilization characteristics are generated using LFE2-50E-6F672C with Lattice Diamond 1.2 design software. When using this design in a different device, density, speed, or grade, performance and utilization may vary.
3. Performance and utilization characteristics are generated using LFXP2-17E-6F484C with Lattice Diamond 1.2 design software. When using this design in a different device, density, speed, or grade, performance and utilization may vary.

In this design, the Lattice FPGA functions as both the transmitter and receiver.

Design Example 2: Demonstration of 7:1 LVDS Interface with Video Processing Functions

In order to verify the operation of the 7:1 LVDS interfaces within the Lattice FPGA, Lattice has developed the test system shown in Figures 11 and 12. The test system on the LatticeECP3 Video Protocol Board is the same as the one on the LatticeXP2 Advanced Evaluation Board. Detailed information regarding the test system on the LatticeECP2 Advanced Evaluation Board, including Boards #1, #2, and #3, and the LatticeECP2 Advanced Evaluation Board, is included in TN1134, Lattice 7:1 LVDS Video Demo Kit User's Guide. This system takes video data supplied in DVI format from a source such as a PC or a DVD player and converts it to the 7:1 LVDS source synchronous format using a National Semiconductor Channel Link Transmitter Device. This image data is fed to the Lattice FPGA where the 7:1 Receiver module is used to deserialize the data. This data is then converted back into serial data using the 7:1 Transmitter module within the Lattice FPGA device. It is then transmitted using a source synchronous 7:1 LVDS interface to a National Semiconductor Channel Link Receiver device and ultimately to a display.

Figure 11. 7:1 Interface Test System on LatticeECP2 Advanced Evaluation Board

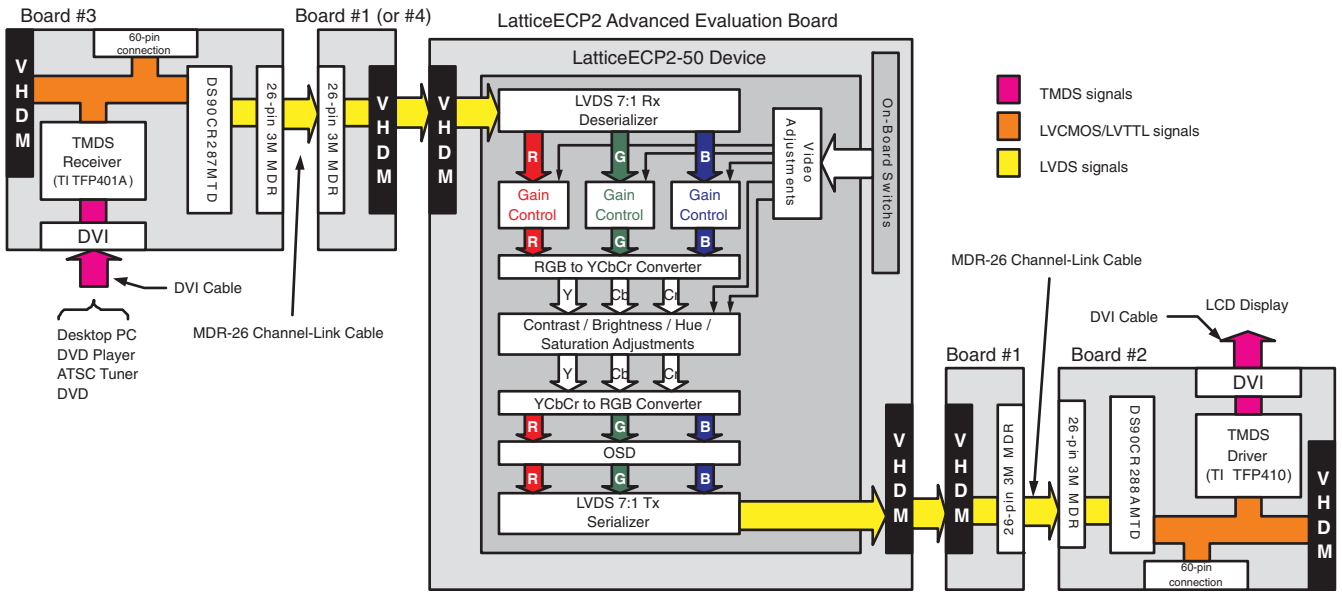
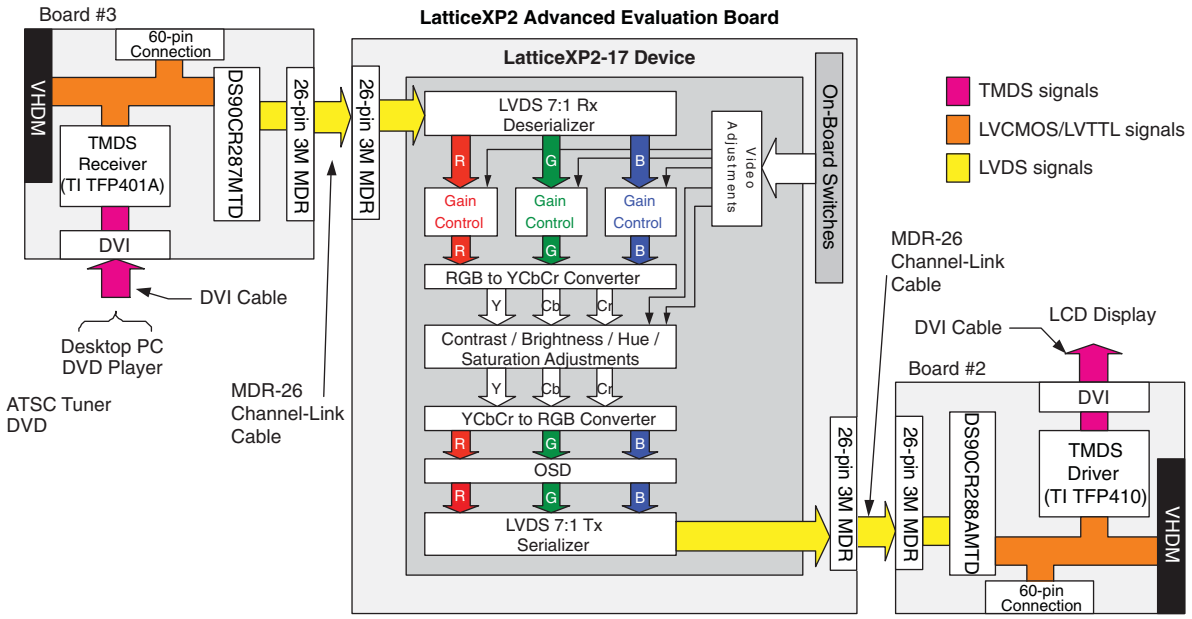


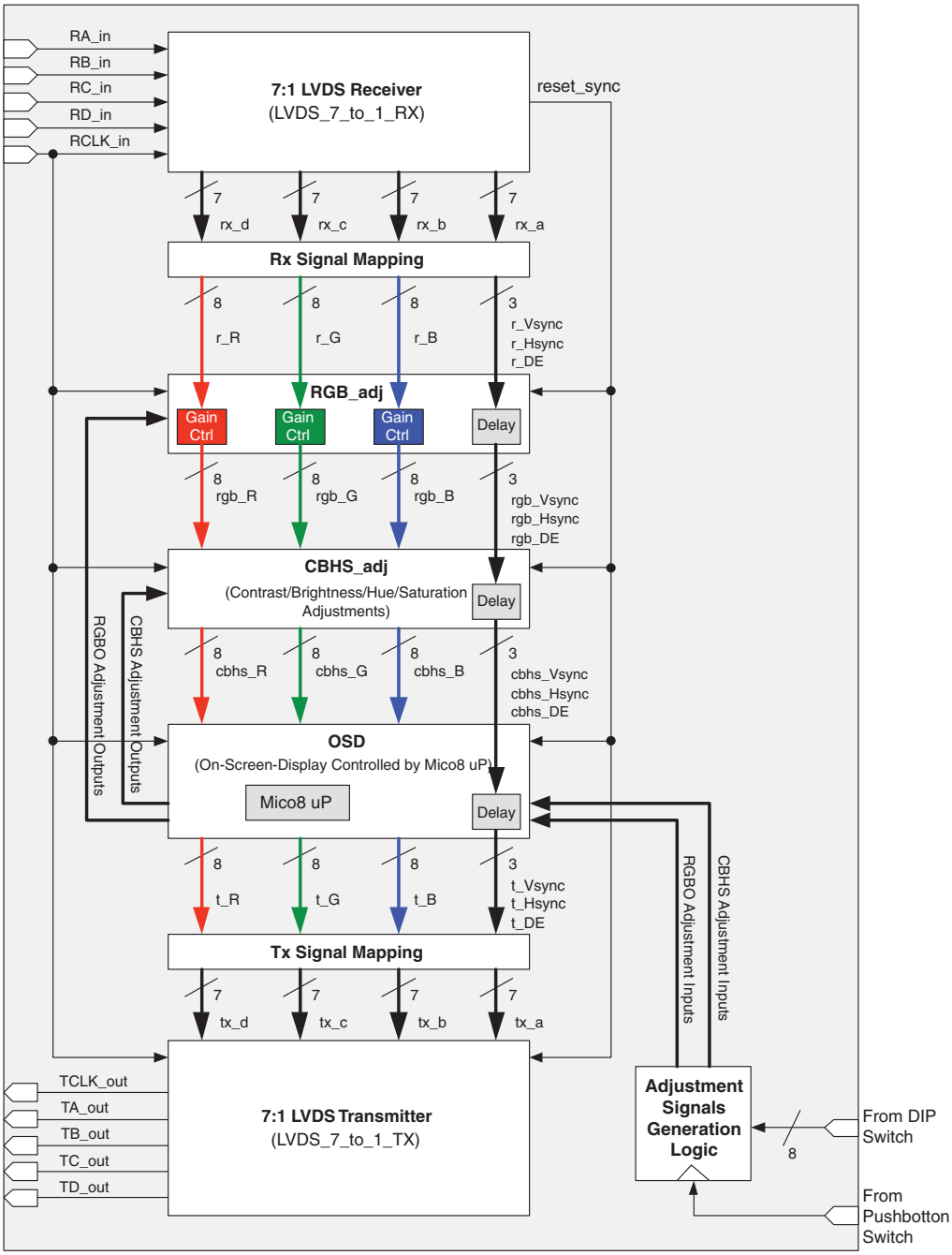
Figure 12. 7:1 Interface Test System on LatticeXP2 Advanced Evaluation Board



Figures 11 and 12 show a simplified block diagram of the design inside the FPGA device. Other than the receiver and transmitter modules, the center logic block can be any customized video processing design. For demonstration purposes, the designs shown in Figures 11 and 12 were created to include the following features.

- R-gain, G-gain, B-gain controls
- Contrast, Brightness, Hue, Saturation controls
- On-Screen-Display controlled by LatticeMico8 microprocessor
- On-Screen-Display opacity control

Figure 13. Video Processing Design Example



The block diagram of this design example is shown in Figure 13. The design includes five sub-modules: Receiver, RGB_adj, CBHS_adj, OSD and Transmitter. On the LatticeECP2 Advanced Evaluation Board, the 8-position DIP-switch SW5 is used for adjusting the R, G, B gains, Contrast, Brightness, Hue, Saturation, and OSD opacity. When the specific controls are selected, the push-button SW4 (i.e., Control Switch) needs to be toggled to activate the adjustment. SW5 is also used for enabling and disabling the OSD and the Auto-Demo feature. The functions of SW5 pins are listed in Table 2. On the LatticeXP2 and LatticeECP3 evaluation boards, the corresponding switches and their functions are also listed in Table 2.

Table 2. Switch for Video Color Adjustments, Demo and OSD Controls

SW5 Pin Number on the LatticeECP2 Board	SW8 Pin Number on the LatticeXP2 Board	SW1/SW2 Pin Number on the LatticeECP3 Board	OFF	ON
SW5-8	SW8-8	SW2-4	R-gain or Contrast deselected	R-gain or Contrast selected
SW5-7	SW8-7	SW2-3	G-gain or Brightness deselected	G-gain or Brightness selected
SW5-6	SW8-6	SW2-2	B-gain or Hue deselected	B-gain or Hue selected
SW5-5	SW8-5	SW2-1	Opacity or Saturation deselected	Opacity or Saturation selected
SW5-4	SW8-4	SW1-4	OSD enabled	OSD disabled
SW5-3	SW8-3	SW1-3	Auto-Demo enabled	Auto-Demo disabled
SW5-2	SW8-2	SW1-2	Select RGBO group	Select CBHS group
SW5-1	SW8-1	SW1-1	Decrease the selected controls when Control Switch is toggled	Increase the selected controls when Control Switch is toggled

Note: Control Switch is SW4 for the LatticeECP2 Advanced Evaluation Board, SW5 for the LatticeXP2 Advanced Evaluation Board, or SW6 for the LatticeECP3 Video Protocol Evaluation Board.

Video Processing Design Implementation Results

The video processing demo design was verified using the Lattice 7:1 LVDS Demo Kit that comes with the LatticeECP3, LatticeECP2 and LatticeXP2 evaluation boards and other daughter boards. The video source was running at 108 MHz at 1280x1024 image resolution. Table 3 shows the resources utilized by the design.

Table 3. Video Processing Design Performance and Resource Utilization

Device	Language	Speed Grade	Utilization (LUTs)	f_{MAX} (MHz)	I/Os	Slices	Registers	sysMEM EBRs	sysDSP Blocks
LatticeECP3[1]	VHDL	-7	1848 (2%)	>108	35	1420	1347	10 (4%)	4.125 (12%)
	Verilog		1852 (2%)	>108	35	1415	1315	10 (4%)	4.125 (12%)
LatticeECP2/M[2]	VHDL	-6	1804(4%)	>108	35	1428	1293	8 (38%)	4.125 (23%)
	Verilog		1857 (4%)	>108	35	1433	1253	10 (48%)	4.125 (22%)
LatticeXP2[3]	VHDL	-6	1803 (11%)	>108	35	1492	1292	8 (53%)	4.125 (82%)
	Verilog		1848 (11%)	>108	35	1482	1254	10 (67%)	4.125 (82%)

1. Performance and utilization characteristics are generated using LFE3-95EA-7FN1156C with Lattice Diamond™ 1.2 design software. When using this design in a different device, density, speed, or grade, performance and utilization may vary.
2. Performance and utilization characteristics are generated using LFE2-50E-6F672C with Lattice Diamond 1.2 design software. When using this design in a different device, density, speed, or grade, performance and utilization may vary.
3. Performance and utilization characteristics are generated using LFXP2-17E-6F484C with Lattice Diamond 1.2 design software. When using this design in a different device, density, speed, or grade, performance and utilization may vary.

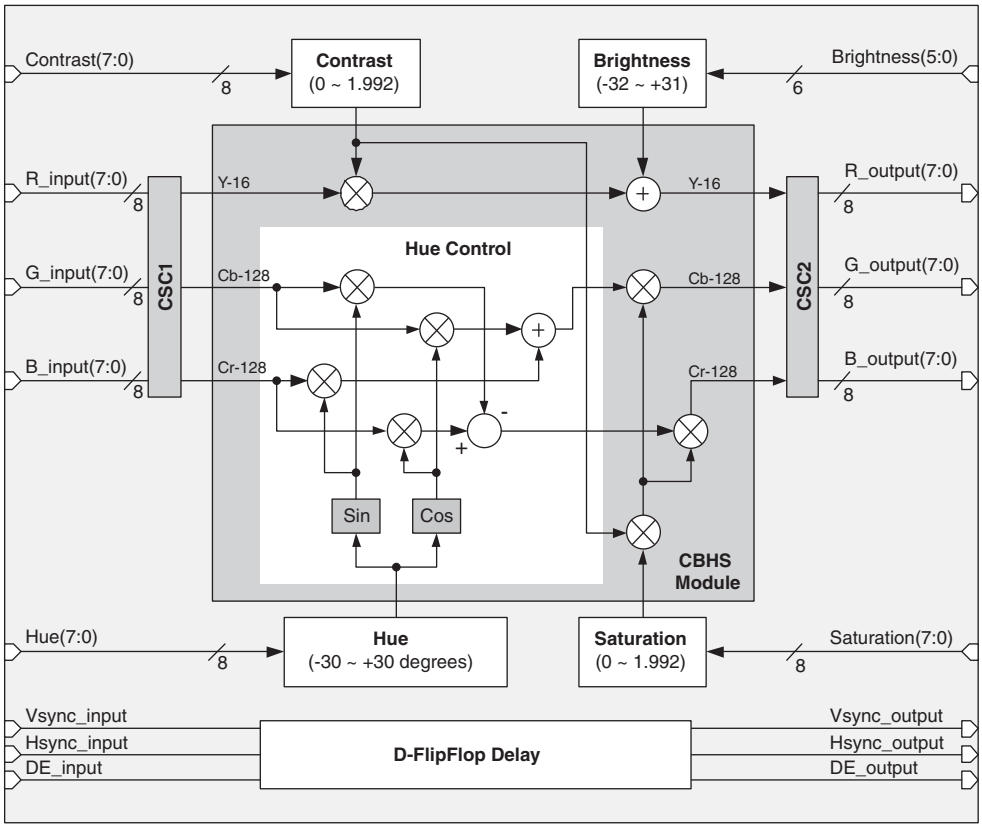
Module RGB_adj

With the 9x9 multipliers implemented using the sysDSP blocks, the RGB_adj module multiplies the 8-bit R, G, B color datum with the R-, G-, B-gain. These gains are real numbers with value between 0 and 1. Nine data bits represent the real number with bit 8 representing the integer part and the rest of the bits representing the fractional part of the real number. For the fractional part, bit 7 represents 2-1 (i.e. 0.5), bit 6 represents 2-2 (i.e. 0.25), bit 5 represents 2-3 (i.e. 0.125), and so on. For example, the 9-bit data “011000000” will be representing the real value 0.5 + 0.25 = 0.75; the 9-bit data “100000000” will be representing the real value 1.0. The similar method to represent a non-integer real value is used in many modules of the design. The number of the integer bits and the fractional bits may be changed to represent real numbers in different range.

Module CBHS_adj

For adjusting contrast, brightness, hue and saturation of the video image, the pixel data in the RGB color space needs to be converted to the YCbCr color space. Figure 14 shows the block diagram of the CBHS_adj module. After the adjustment, the pixel data in the YCbCr color space will be converted back to the RGB color space. There are offsets in the YCbCr color space. The offsets of Y, Cb and Cr are 16, 128 and 128 respectively. When performing the contrast, brightness, hue and saturation adjustments, these offsets need to be removed. Therefore, the color space converters CSC1 and CSC2 convert the pixel data between the RGB and the YCbCr without adding the Y, Cb and Cr offsets.

Figure 14. Contrast, Brightness, Hue and Saturation Adjustments



The equations used in the CSC1 and CSC2 converters are:

CSC1

$$\begin{bmatrix} Y - 16 \\ C_b - 128 \\ C_r - 128 \end{bmatrix} = \begin{bmatrix} 0.2567890625 & 0.50412890625 & 0.09790625 \\ 0.14822265625 & 0.2909921875 & 0.43921484375 \\ 0.43921484375 & 0.3677890625 & 0.07142578125 \end{bmatrix} \bullet \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

CSC2

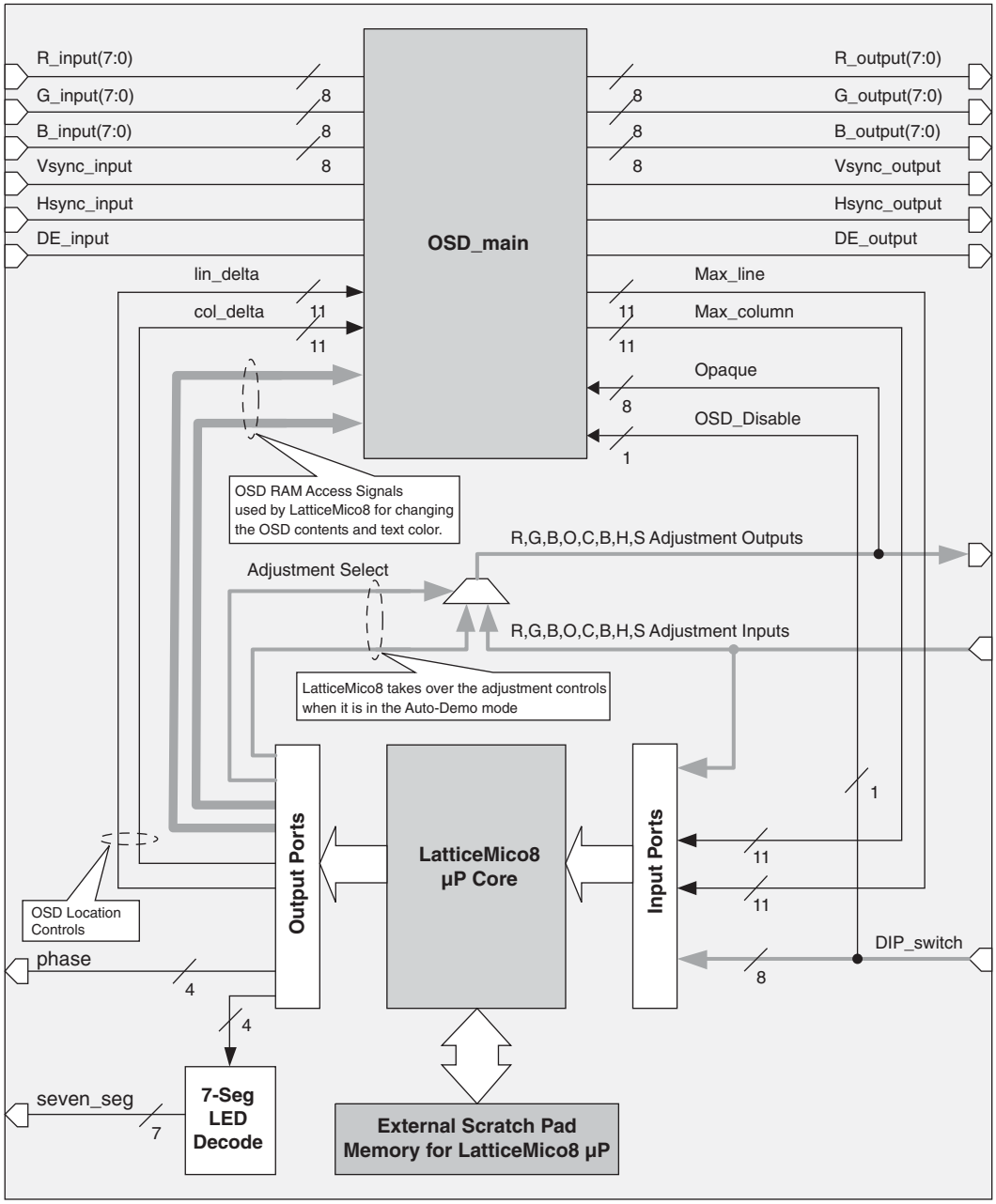
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.1643828125 & 0 & 1.59602734375 \\ 1.1643828125 & -0.39176171875 & -0.81296875 \\ 1.1643828125 & 2.01723046875 & 0 \end{bmatrix} \bullet \begin{bmatrix} Y - 16 \\ C_b - 128 \\ C_r - 128 \end{bmatrix}$$

The Sine and Cosine functions are required for the hue adjustment. A lookup table ROM is used for implementing these two functions. A Tcl script is designed to create the memory file used for the Sine/Cosine ROM contents initialization.

Module OSD

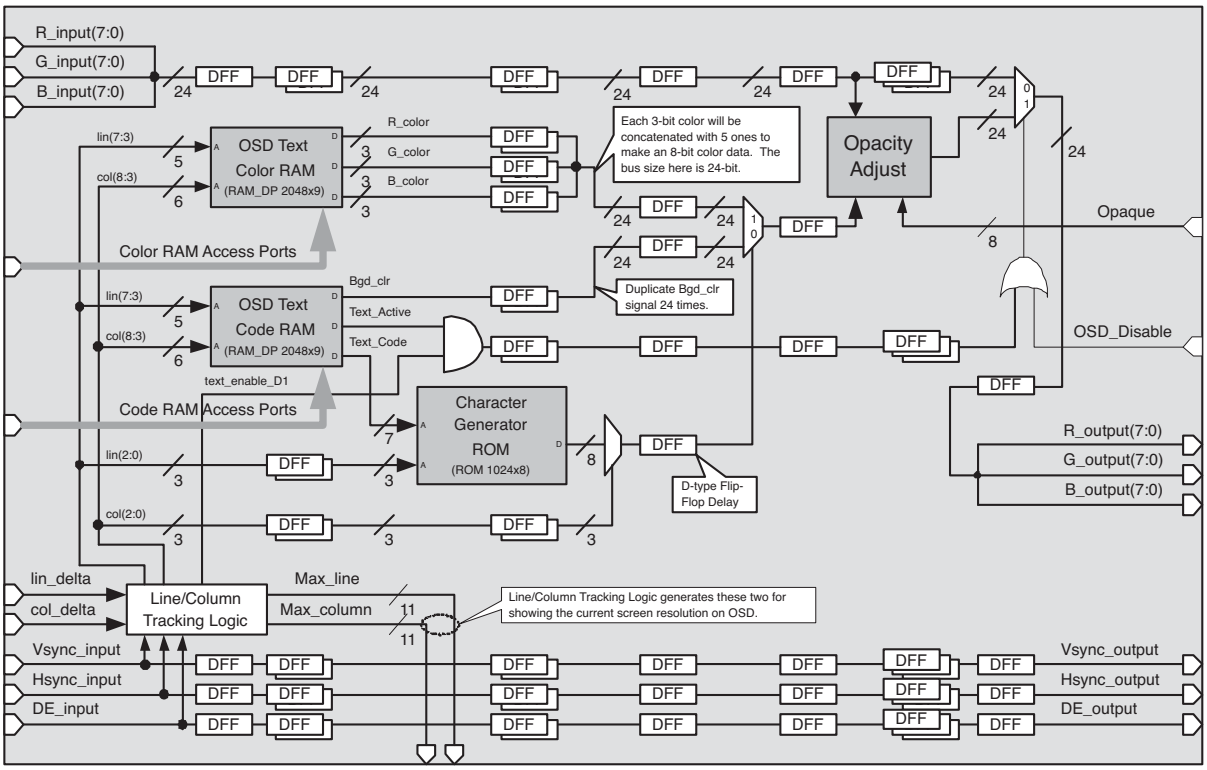
The contents of the On-Screen-Display are controlled by the LatticeMico8 microcontroller. Figure 15 shows the block diagram of the OSD module. The LatticeMico8™ microprocessor, a free 8-bit microcontroller soft core optimized for Lattice FPGAs, will update the dual-port RAM contents in the OSD_main sub-module to reflect the current RGB gains, Contrast, Brightness, Hue, Saturation and OSD Opacity values. It also controls these adjustment values when the Auto-Demo mode is enabled.

Figure 15. On-Screen-Display Module



The block diagram of the OSD_main sub-module is shown in Figure 16. There two dual-port RAMs to hold the character codes and the character colors displayed on the OSD. The OSD contents will be changed whenever the RAMs contents are updated by the LatticeMico8 microcontroller. The character patterns are stored in the Character Generator ROM.

Figure 16. OSD_main Sub-module



The Line/Column Tracking Logic block controls the position of the OSD and is also used for tracking the current resolution of the video image. The Max_line and Max_column will be read back by the LatticeMico8 to display the resolution on the OSD.

The OSD opacity control is implemented in the OSD_main sub-module as well. The Opaque value is a real number between 0 and 1 with 1 as the default value. When the value is reduced, the OSD will become semi-transparent. Figure 17 shows the block diagram of the OSD opacity control.

Figure 17. OSD Opacity Control

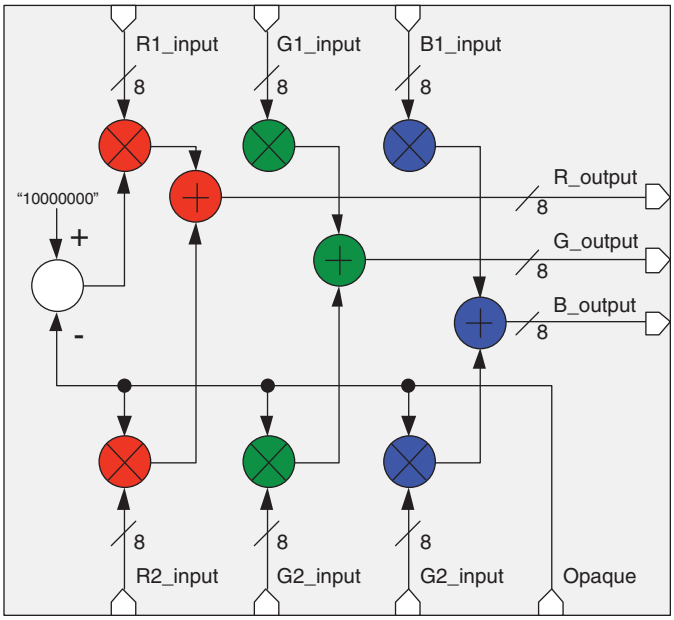


Figure 18. Semitransparent OSD Showing On the Display Screen

Summary

The LatticeECP3, LatticeECP2/M and LatticeXP2 FPGA families are well-suited for high-speed LVDS video applications. In addition to capturing the video data at high speeds, these families are capable of processing video data using the on-chip sysDSP block and the Embedded Block RAM.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
September 2006	01.0	Initial release.
March 2007	01.1	Updated 7:1 Receive Side Block Diagram.
		Updated IDDRX2B Detailed Block Diagram.
		Updated Example of Input Gearing Using IDDRX2B diagram.
		Updated Transmitter Side Block Diagram.
		Updated ODDRX2B Gearing Function diagram.
		Updated Example of Output Gearing Using ODDRX2B diagram.
May 2007	01.2	Added the video demo design example that includes the color adjustments, OSD and auto-demo features. Updated the Performance and Resource Utilization tables to include numbers for both VHDL and Verilog version. Updated figures.
September 2007	01.3	Added LatticeXP2 family support and removed the timing analysis section.
September 2009	01.4	Added LatticeECP3 family “E” series support.
April 2011	01.5	Added LatticeECP3 family “EA” series support.