Display Interface Multiplexer IP Core User’s Guide

November 2010
IPUG95_01.0

Lattice Semiconductor Corporation

Table of Contents

Chapter 1. Introduction 3
Quick Facts 4
Features 4
Chapter 2. Functional Description 5
Overview 5
LVDS Data Mapping 5
LVDS Rx 7
LVDS Tx 7
Parallel Multiplexer 8
Interface Description 8
Top-Level I/O Interface 9
Interface Description 10
LVDS A and B Inputs 10
LVDS C outputs 10
Interface Waveforms 10
Chapter 3. Parameter Settings 12
Parameters Group 12
Optional Ports Group 13
Chapter 4. IP Core Generation 14
Licensing the IP Core 14
Getting Started 14
IPexpress-Created Files and Top Level Directory Structure 16
Running Functional Simulation 18
Synthesizing and Implementing the Core in a Top-Level Design 19
Hardware Evaluation 19
Enabling Hardware Evaluation in Diamond 20
Enabling Hardware Evaluation in ispLEVER 20
Updating/Regenerating the IP Core 20
Regenerating an IP Core in Diamond 20
Regenerating an IP Core in ispLEVER 20
Chapter 5. Application Support 22
Using the IP Core 22
Simulation and Verification 22
Simulation Strategy 22
Display Interface Multiplexer IP Demo Design 22
Chapter 6. Core Verification 23
Chapter 7. Support Resources 24
Lattice Technical Support 24
Online Forums 24
Telephone Support Hotline 24
E-mail Support 24
Local Support 24
Internet 24
References 24
MachXO2 24
Revision History 25
Appendix A. Resource Utilization 26
Lattice MachXO2 FPGAs 26
Ordering Information 26

Introduction

Display interface standards such as flat panel display link (FPD link) [1] and the newer open LVDS display interface (Open LDI) [2] offer robust, high bandwidth connectivity between graphics controllers and flat panel displays. These display interfaces use low voltage differential signaling (LVDS) pairs to increase the transfer rate and to reduce EMI (electromagnetic interference) and power dissipation. Seven lines of TTL (transistor transfer logic) signals from the graphics controller are multiplexed into one pair of LVDS signal lines to achieve the narrow connectivity to the display. The number of LVDS pairs used in the interface depends on the color depth and the number of pixels transported per clock. As an example, for a video that has 8-bit colors and is transferred at the rate of one pixel per clock, the parallel data is 28 bits wide. The display interface standards also require that the low speed clock (corresponding to a block of 7 serially multiplexed data bits) be transmitted through a separate LVDS pair. Thus the 8-bit color, one pixel per clock example data would be transported through five LVDS pairs.

There is an increasing need for dynamically changing the graphics controller power consumption and display performance, especially in laptop computers. The laptop manufacturers are now offering the ability to drive the display from multiple sources including discrete and integrated graphics controllers to achieve this balance. It is also important that the switching between the graphics controllers happens fast and without visual artifact. This is usually achieved by a piece of digital logic that processes the video data, extracts the video timing signals and determines the ideal switching instant. With dedicated DDR macros for 7 to 1 multiplexing, LVDS I/Os (inputs/outputs), PLLs and programmable logic, the Lattice MachXO2™ is the ideal device to achieve efficient and smart display interface multiplexing. Lattice's Display Interface Multiplexer IP is an off-the-shelf solution for an easy and quick implementation of display interface multiplexer in Lattice MachXO2 devices.

The display multiplexer IP includes the LVDS I/Os, DDRx4 primitives for 7 to 1 multiplexing/de-multiplexing, PLLs and a parallel data multiplexer. The IP supports 3, 4, 6 or 8 LVDS data lanes for each of the input streams. The video streams from two asynchronous graphics controllers are connected through LVDS I/Os to the specially designed video DDR input modules which de-multiplex the LVDS signals into 7 parallel TTL outputs for each lane. The parallel data outputs are then multiplexed using a parallel logic multiplexer. The parallel data multiplexing is initiated by a user signal, but the IP offers the option to switch instantaneously or in synchronization with the vertical sync instant of either input stream. The multiplexed parallel data are then grouped and serialized using the specially designed Video DDR output modules. The DDR outputs are connected to the LVDS I/O pads. Refer to TN1203, *Implementing High-Speed Interfaces with MachXO2 Devices*, for details on the high speed DDR interfaces in MachXO2.

Refer to the Lattice Display Interface Multiplexer IP core web page at:

http://www.latticesemi.com/products/intellectualproperty/ipcores/displayinterfacemultiplex.cfm

Quick Facts

Table 1-1 gives quick facts about the Display Interface Multiplexer IP Core for Lattice MachXO2 devices.

Table 1-1. Display Interface Multiplexer IP Core Quick Facts

		Display Interface Multiplexer IP Core							
Number of PLLs		1	1	1	1	2	2	2	2
Pixels per clock		1	1	2	2	1	1	2	2
Color depth		6	8	6	8	6	8	6	8
Core Requirements	FPGA Families Supported	Lattice MachX02							
	Minimal Device Needed	2000	2000	2000	4000	4000	4000	4000	4000
Resource Utilization	Targeted Device	LCMXO2-4000HCCABGA256							
	DDR Output Lanes	3	4	6	8	3	4	6	8
	LUTs	287	319	390	448	436	461	533	591
	sysMEM EBRs	0	0	0	0	0	0	0	0
	Registers	170	195	239	279	245	266	308	352
Design Tool Support[1]	Lattice Implementation	Lattice Diamond™ 1.1 or ispLEVER® 8.1							
	Synthesis	Synplicity® Synplify™ Pro D-2010.03L-SP1							
	Simulation	Aldec® Active-HDL™ 8.2 Lattice Edition							
		Mentor Graphics® ModelSim™ SE 6.3F							

1. Design tool support for IP core version 1.0.

Features

The following are the major features of the Lattice Display Interface Multiplexer IP core:

- Includes LVDS Physical layer- two LVDS to parallel and one parallel to LVDS converters
- Supports one PLL and two PLL configurations to enable the use of smaller devices
- Supports one and two pixels per clock configurations
- Supports 6 bits and 8 bits color depths
- Supports all standard resolutions, refresh rates and pixel depths that are within the parallel data clock rate of 108 MHz
- Supports instantaneous and synchronized switching
- Provides optional sync and RGB data outputs for input and output video streams

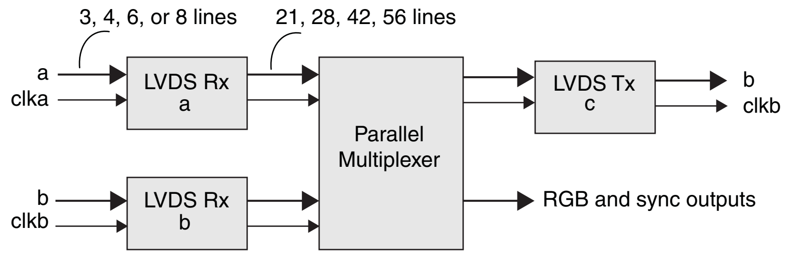
Functional Description

Overview

The top-level functional block diagram of the Display Interface Multiplexer IP core is shown in Figure 2-1. The main components of the IP are two LVDS receivers (LVDS Rx a and LVDS Rx b), a parallel multiplexer and a LVDS transmitter (LVDS Tx c). The size of the input and output busses depends on the number of pixels transmitted per clock and the color depth. The LVDS Rx modules convert each of the 7 to 1 multiplexed, serialized LVDS data signal to a 7-bit parallel data using the dedicated MachXO2 video DDR input modules. The parallel multiplexer block determines the vsync (vertical sync) instants for each of the input streams and multiplexes the data synchronizing the switching time at selected instances. This module also generates the optional RGB (red-green-blue), vsync, hsync (horizontal sync) and de (data enable) outputs. The output of the parallel multiplexer is converted to serialized LVDS output by the LVDS Tx c module.

For a detailed discussion on the design of LVDS display interface using MachXO2, refer to RD1093, *MachXO2 Display Interface*.

Figure 2-1. Functional Block Diagram of the 7 to 1 LVDS Mux IP



LVDS Data Mapping

The packing of RGB and sync data in the LVDS pairs depends on the color depth and whether one or two pixels are transferred in a clock cycle. The four different data mappings described in *VESA Notebook Panel Standard* [3] are shown in Figure 2-2 through Figure 2-5. They are 6-bit color, one pixel per clock, 6-bit color, two pixels per clock, 8-bit color, one pixel per clock, and 8-bit color, two pixels per clock mappings.

Figure 2-2. 6-Bit, One Pixel Per Clock Data Mapping

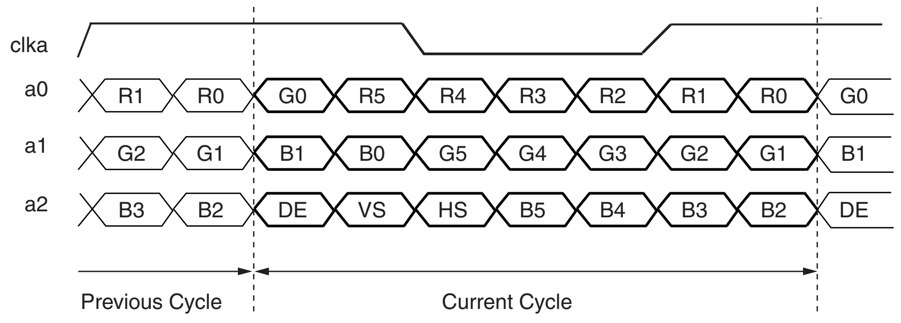

clka
a0
R1 R0 G0 R5 R4 R3 R2 R1 R0 G0
a1
G2 G1 B1 B0 G5 G4 G3 G2 G1 B1
a2
B3 B2 DE VS HS B5 B4 B3 B2 DE
Previous Cycle
Current Cycle

Figure 2-3. 8-Bit, One Pixel Per Clock Data Mapping

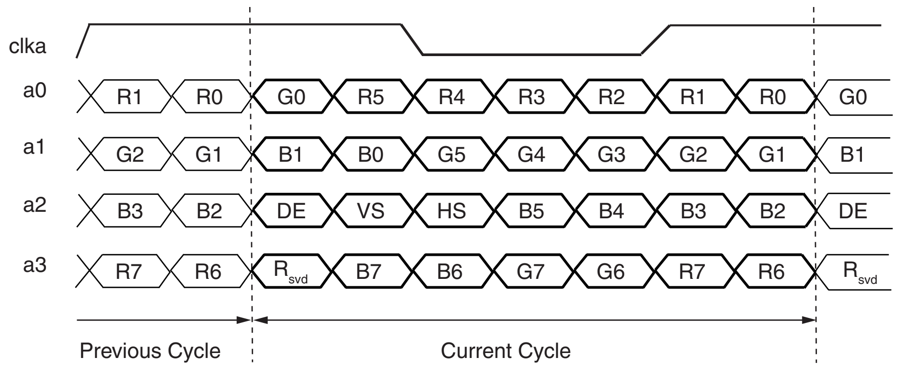

clka
a0: R1 R0 G0 R5 R4 R3 R2 R1 R0 G0
a1: G2 G1 B1 B0 G5 G4 G3 G2 G1 B1
a2: B3 B2 DE VS HS B5 B4 B3 B2 DE
a3: R7 R6 Rsvd B7 B6 G7 G6 R7 R6 Rsvd
Previous Cycle
Current Cycle

Figure 2-4. 6-Bit, Two Pixels Per Clock Data Mapping

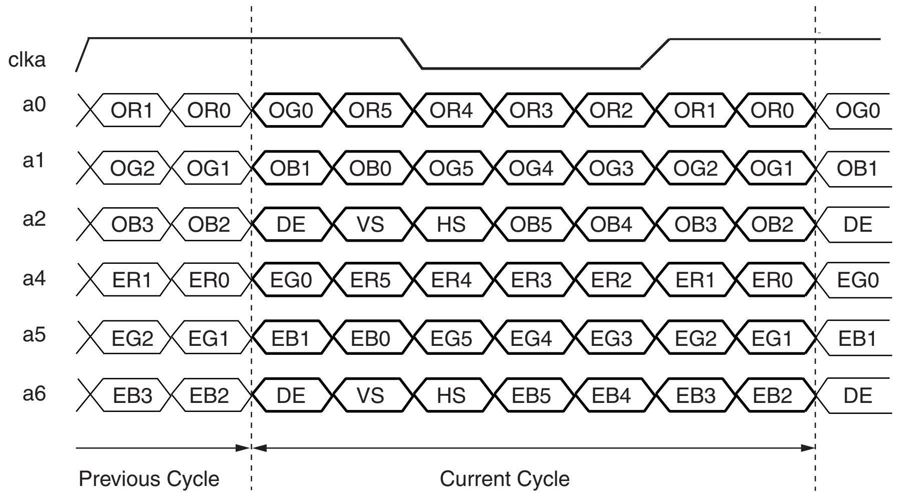
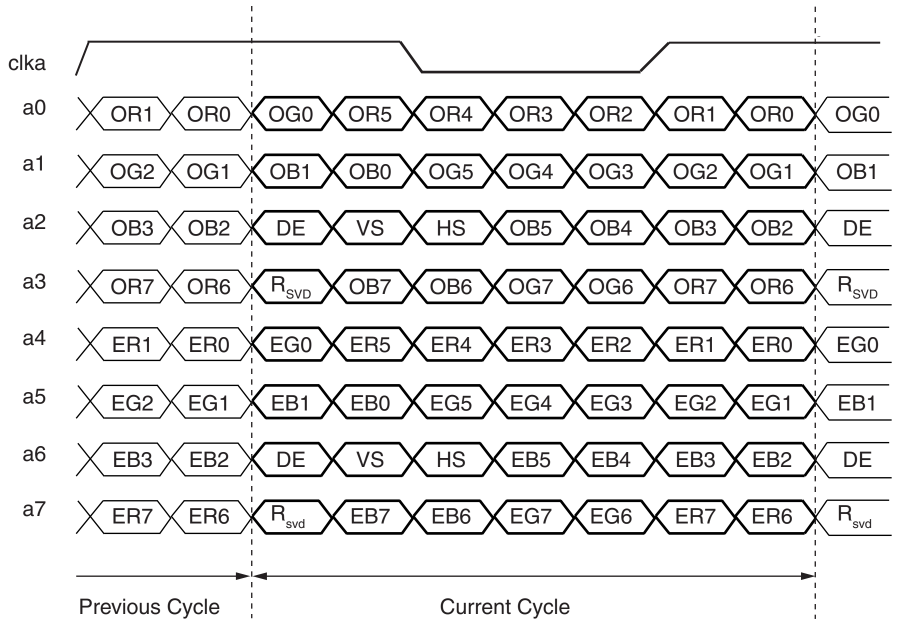

clka
a0: OR1 OR0 OG0 OR5 OR4 OR3 OR2 OR1 OR0 OG0
a1: OG2 OG1 OB1 OB0 OG5 OG4 OG3 OG2 OG1 OB1
a2: OB3 OB2 DE VS HS OB5 OB4 OB3 OB2 DE
a4: ER1 ER0 EG0 ER5 ER4 ER3 ER2 ER1 ER0 EG0
a5: EG2 EG1 EB1 EB0 EG5 EG4 EG3 EG2 EG1 EB1
a6: EB3 EB2 DE VS HS EB5 EB4 EB3 EB2 DE
Previous Cycle
Current Cycle

Figure 2-5. 8-Bit, Two Pixels Per Clock Data Mapping


clka
a0: OR1 OR0 OG0 OR5 OR4 OR3 OR2 OR1 OR0 OG0
a1: OG2 OG1 OB1 OB0 OG5 OG4 OG3 OG2 OG1 OB1
a2: OB3 OB2 DE VS HS OB5 OB4 OB3 OB2 DE
a3: OR7 OR6 RSVD OB7 OB6 OG7 OG6 OR7 OR6 RSVD
a4: ER1 ER0 EG0 ER5 ER4 ER3 ER2 ER1 ER0 EG0
a5: EG2 EG1 EB1 EB0 EG5 EG4 EG3 EG2 EG1 EB1
a6: EB3 EB2 DE VS HS EB5 EB4 EB3 EB2 DE
a7: ER7 ER6 Rsvd EB7 EB6 EG7 EG6 ER7 ER6 Rsvd
Previous Cycle
Current Cycle

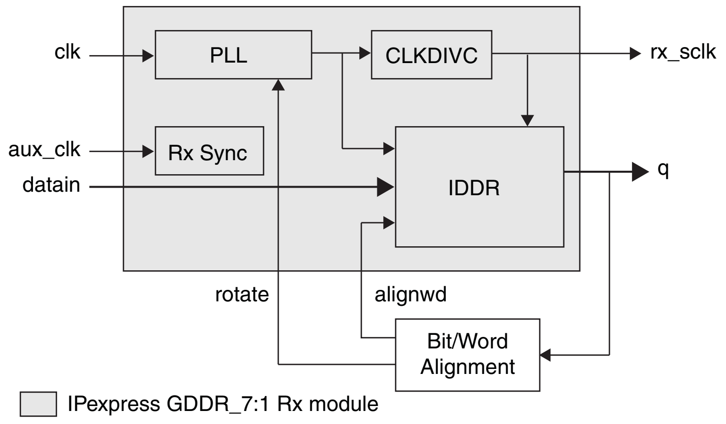
Display panels use one of the four LVDS interface pixel mapping formats shown in Figure 2-2 through Figure 2-5. The two main categories are single channel or one pixel per clock and dual channel or two pixels per clock. In each of these categories, there is a 6-bit and 8-bit mapping format. The 8-bit mapping format is an extension of the 6-bit in that it uses three LVDS pairs exactly same as that of 6-bit format with an additional LVDS pair for the upper two bits of each color component.

The format of the video stream, i.e. the duration of horizontal and vertical sync pulses and the gap between sync pulses and data, varies with each monitor or panel used. The actual hsync and vsync signals to the panel are generated by the TCON (timing controller).

LVDS Rx

The LVDS Rx blocks shown in Figure 2-1 convert the serially multiplexed LVDS inputs to parallel data using DDR I/Os. This block is built around the MachXO2 IPexpress module “GDDR_7:1”. The block diagram of the LVDS Rx block is shown in Figure 2-6. The GDDR_7:1 Rx module generated from IPexpress is shaded. The bit/word alignment logic generates the rotate signal for the PLL and the alignwd signal for the IDDR to achieve bit and word alignments.

Figure 2-6. LVDS Rx Block

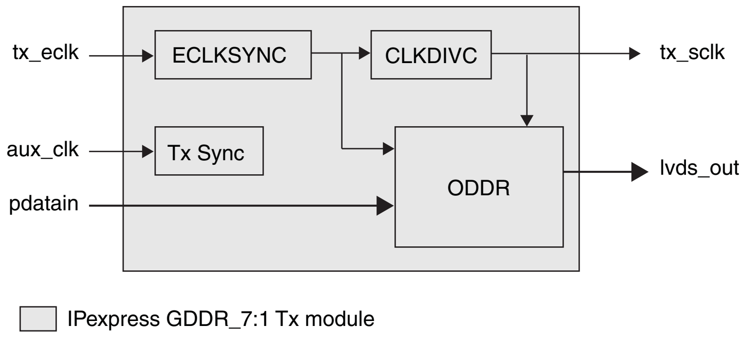


For IP configurations that use one PLL, the Rx module will be constructed using primitives rather than from IPexpress.

LVDS Tx

A block diagram of LVDS Tx block is shown in Figure 2-7. As shown in the figure, this block is built around the IPexpress MachXO2 architecture module, GDDR_7:1 Tx.

Figure 2-7. LVDS Tx Block



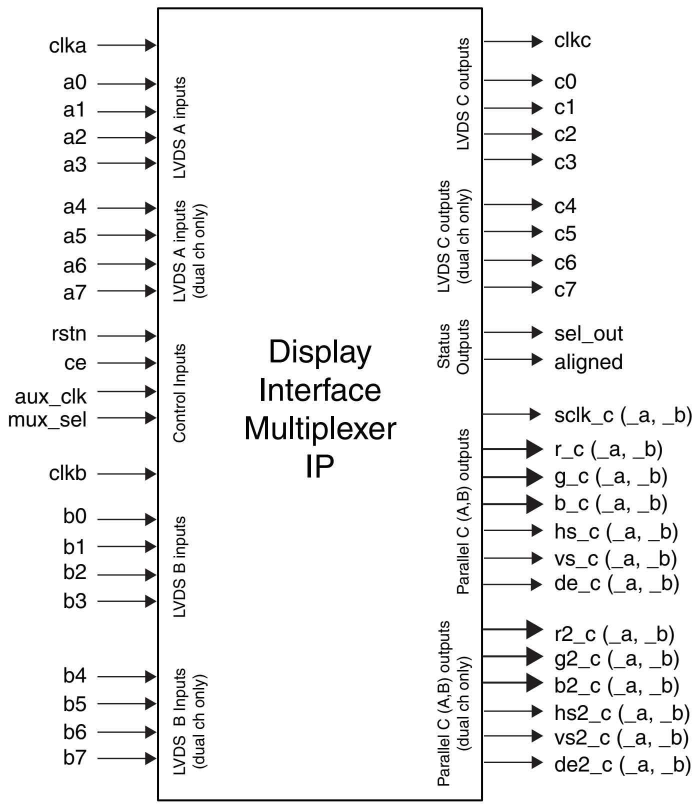
Parallel Multiplexer

The parallel multiplexer gets parallel data and the corresponding clock from the LVDS receivers of each video stream. The output of the multiplexer has to be read relative to the output clock from the LVDS Tx block. A system of two asynchronous FIFOs is used to achieve multiplexing of the parallel data and synchronizing the result with the Tx clock. Several synchronized switching options are provided by the IP. In the simplest option, the data are multiplexed instantaneously, that is as soon as the external switch signal is received. The other options are to switch at the start of vsync or end of vsync. The vsync itself can be chosen to correspond to the current output stream or the newly selected output stream. To perform synchronized switching, the vsync timing is extracted from the vsync bit in the data streams.

Interface Description

The top-level interface diagram for the Display Interface Multiplexer IP is shown in Figure 2-8.

Figure 2-8. Display Interface Multiplexer IP Interfaces



Top-Level I/O Interface

Table 2-1 provides the list of ports and descriptions of each port for the Display Interface Multiplexer IP core.

Table 2-1. Display Interface Multiplexer IP Core Port Description

Port	Bits	I/O	Description
LVDS A inputs			
clka	1	I	LVDS clock input for stream A.
a0 - a7	1	I	LVDS data inputs 0 through 7 for stream A. The inputs available for different configurations are as follows: 1 pixel per clock, 6-bit color- a0, a1, a2 1 pixel per clock, 8-bit color- a0, a1, a2, a3 2 pixels per clock, 6-bit color- a0, a1, a2, a4, a5, a6 2 pixels per clock, 8-bit color- a0, a1, a2, a3, a4, a5, a6, a7
LVDS B inputs			
clkb	1	I	LVDS clock input for stream B.
b0 - b7	1	I	LVDS data inputs 0 through 7 for stream B. The inputs available for different configurations are as follows: 1 pixel per clock, 6-bit color- b0, b1, b2 1 pixel per clock, 8-bit color- b0, b1, b2, b3 2 pixels per clock, 6-bit color- b0, b1, b2, b4, b5, b6 2 pixels per clock, 8-bit color- b0, b1, b2, b3, b4, b5, b6, b7
Control Inputs			
rstn	1	I	Asynchronous system reset input. A zero on this input resets FIFO pointers and all the registers in the design.
ce	1	I	Optional clock enable input. A zero on this input freezes the register switching in the design. It must be tied to 1 for normal operation.
mux_sel	1	I	Video multiplexer select input. A zero on this input selects video stream A and a one selects video stream B. Depending on the configuration, the output switching instant may not immediately follow the `mux_sel` switching instant.
aux_clk	1	I	Auxiliary clock used to perform startup synchronization of MachXO2 DDR input/output modules. This clock needs to be available during initialization and therefore cannot be derived from data clock or PLL output clocks. The frequency of this clock can be anywhere between 10 and 110 MHz.
LVDS C outputs			
clkc	1	O	LVDS clock output for stream C.
c0 - c7	1	O	LVDS data outputs 0 through 7 for stream C. Stream C is the multiplexed version of inputs A and B. The outputs available for different configurations are as follows: 1 pixel per clock, 6-bit color- c0, c1, c2 1 pixel per clock, 8-bit color- c0, c1, c2, c3 2 pixels per clock, 6-bit color- c0, c1, c2, c4, c5, c6 2 pixels per clock, 8-bit color- c0, c1, c2, c3, c4, c5, c6, c7
Status outputs			
sel_out	1	O	This output identifies the input that is selected by the multiplexer. This output closely follows (with a latency) `mux_sel` when the IP is configured for immediate switching. For synchronized switching configurations, this output changes state during the appropriate vsync instant immediately following the `mux_sel` switching instant.
aligned	1	O	This output indicates whether the LVDS receivers have achieved bit and word alignments.
Parallel C (A,B) outputs			
sclk_c (_a,_b)	1	O	Parallel data output clock for C (A,B) outputs.
r_c (_a,_b)	w	O	Parallel R component output for stream C (A, B). The bus width, w is 6 and 8 for 6-bit color and 8-bit color respectively.
g_c (_a,_b)	w	O	Parallel G component output for stream C (A, B). The bus width, w is 6 and 8 for 6-bit color and 8-bit color respectively.

Table 2-1. Display Interface Multiplexer IP Core Port Description (Continued)

Port	Bits	I/O	Description
b_c (_a,_b)	w	O	Parallel B component output for stream C (A, B). The bus width, w is 6 and 8 for 6-bit color and 8-bit color respectively.
hs_c (_a,_b)	1	O	Horizontal sync output for stream C (A, B).
vs_c (_a,_b)	1	O	Vertical sync output for stream C (A, B).
de_c (_a,_b)	1	O	Data enable output for stream C (A, B).
r2_c (_a,_b)	w	O	Parallel R component output for stream C (A, B), channel 2. The bus width, w is 6 and 8 for 6-bit color and 8-bit color respectively.
g2_c (_a,_b)	w	O	Parallel G component output for stream C (A, B), channel 2. The bus width, w is 6 and 8 for 6-bit color and 8-bit color respectively.
b2_c (_a,_b)	w	O	Parallel B component output for stream C (A, B), channel 2. The bus width, w is 6 and 8 for 6-bit color and 8-bit color respectively.
hs2_c (_a,_b)	1	O	Horizontal sync output for stream C (A, B), channel 2.
vs2_c (_a,_b)	1	O	Vertical sync output for stream C (A, B), channel 2.
de_c (_a,_b)	1	O	Data enable output for stream C (A, B), channel 2.

Interface Description

This section describes the interface ports and their usage.

LVDS A and B Inputs

The LVDS data and clock inputs typically come from the graphics controllers as LVDS pairs. The single ended ports of the IP have to be connected to the LVDS input pads of the MachXO2 device. A sample connection is shown in the top-level Verilog file generated with the IP core.

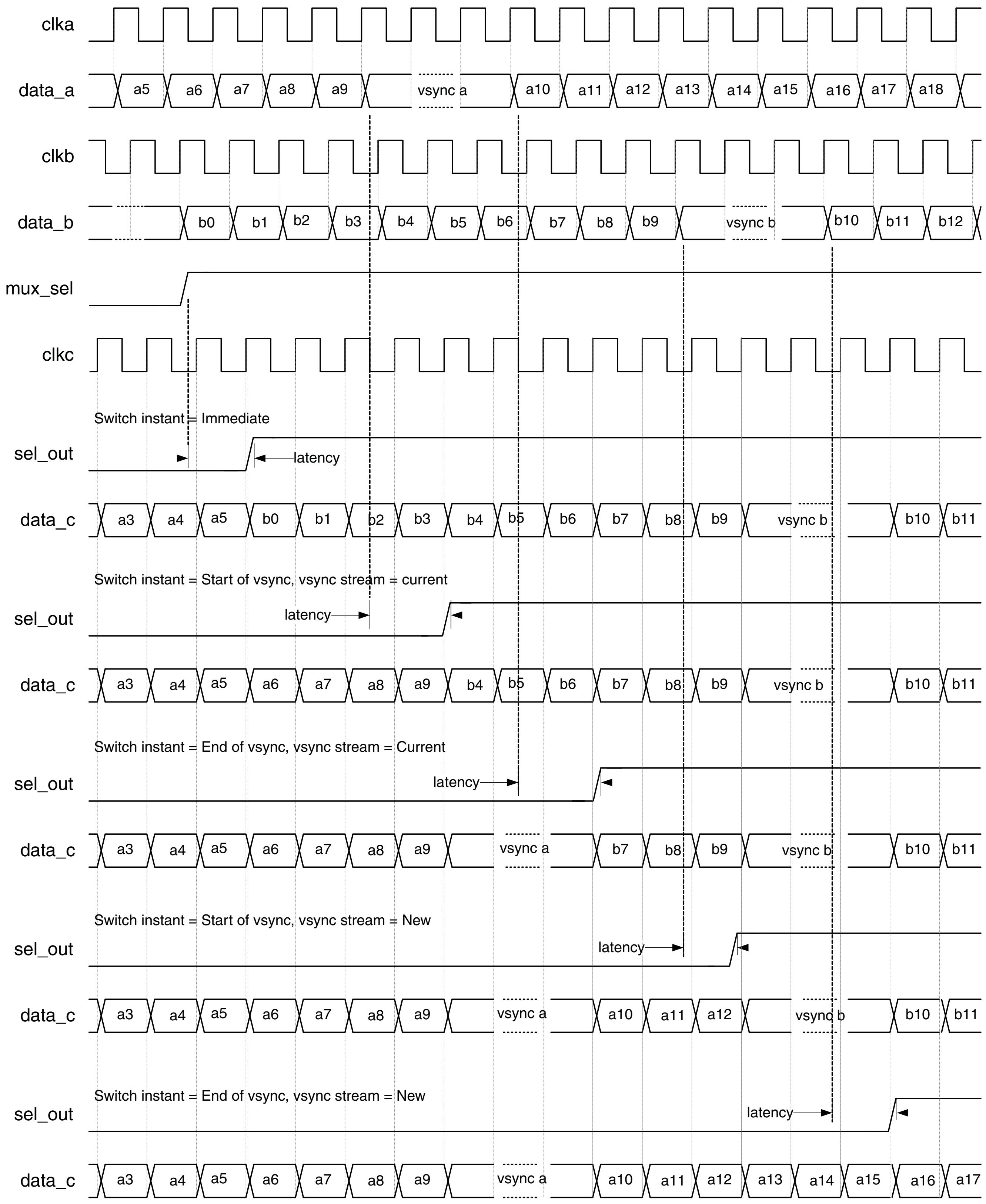
LVDS C outputs

These are the multiplexed video outputs of the IP. The single ended ports out of the IP have to be connected to the LVDS output buffers of the MachXO2 device. The sample top-level Verilog file generated with the IP core shows this connection.

Interface Waveforms

The switching instances for different configurations are illustrated in Figure 2-9. For simplicity, the video streams are assumed to have ten pixels per horizontal line. The figure shows the data and sel_out outputs for five different configurations. As shown in the figure the output appears after a latency after the selected event. The typical latency is in the order of 15 clock cycles between the LVDS inputs and LVDS outputs. The event that triggers the switching depends on the parameter settings and it is one of the following: mux_sel, start of vsync of current stream, end of vsync of current stream, start of vsync of new stream and end of vsync of new stream. In the example shown, the signal mux_sel goes from 0 to 1, thus making stream A as the current stream and stream B as the new stream.

Figure 2-9. Interface Waveforms


clka
data_a
a5
a6
a7
a8
a9
vsync a
a10
a11
a12
a13
a14
a15
a16
a17
a18
clkb
data_b
b0
b1
b2
b3
b4
b5
b6
b7
b8
b9
vsync b
b10
b11
b12
mux_sel
clkc
Switch instant = Immediate
sel_out
latency
data_c
a3 a4 a5 b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 vsync b b10 b11
Switch instant = Start of vsync, vsync stream = current
sel_out
latency
data_c
a3 a4 a5 a6 a7 a8 a9 b4 b5 b6 b7 b8 b9 vsync b b10 b11
Switch instant = End of vsync, vsync stream = Current
sel_out
latency
data_c
a3 a4 a5 a6 a7 a8 a9 vsync a b7 b8 b9 vsync b b10 b11
Switch instant = Start of vsync, vsync stream = New
sel_out
latency
data_c
a3 a4 a5 a6 a7 a8 a9 vsync a a10 a11 a12 vsync b b10 b11
Switch instant = End of vsync, vsync stream = New
sel_out
latency
data_c
a3 a4 a5 a6 a7 a8 a9 vsync a a10 a11 a12 a13 a14 a15 a16 a17

Parameter Settings

The IPexpress™ tool is used to create IP and architectural modules in the Diamond or ispLEVER software. Refer to "IP Core Generation" on page 14 for a description on how to generate the IP.

Table 3-1 provides the list of user configurable parameters for the Display Interface Multiplexer IP Core.

Table 3-1. IP Core Parameters

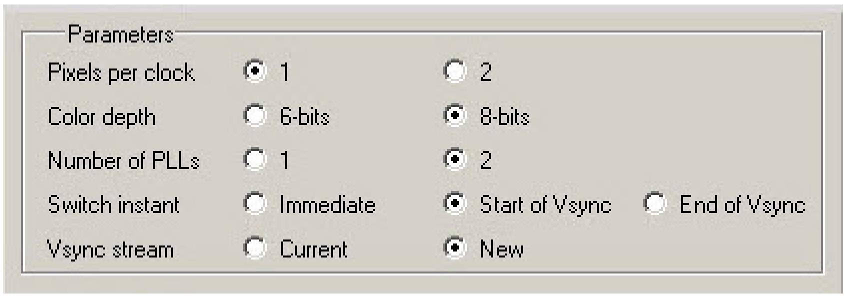
Name	Description	Range	Defaults
Pixels per clock	Specifies the transfer rate in pixels per clock	{1, 2}	1
Color depth	Color depth for the transported video stream.	{6, 8}	8
Number of PLLs	Number of PLLs used by the LVDS receivers in the design.	{1, 2}	2
Switch instant	Defines the parallel multiplexer switching option.	{Immediate, Start of vsync, End of vsync}	Start of vsync
Vsync stream	Defines whether the current or the new (to be switched to) stream is used for vsync determination.	{Current, New}	New
Optional Ports			
Clock enable port	Configures if a clock enable port is required in the IP.	{Yes, No}	No
RGB outputs for A input	Selects whether RGB parallel outputs are required for the input stream A.	{Yes, No}	No
RGB outputs for B input	Selects whether RGB parallel outputs are required for the input stream B.	{Yes, No}	No
RGB outputs for C output	Selects whether RGB parallel outputs are required for the output stream C.	{Yes, No}	No

The parameter settings are specified using the Display Interface Multiplexer IP core Configuration GUI in IPexpress. The Display Interface Multiplexer parameter options are grouped into two categories: Parameters Group and Optional Ports Group.

Parameters Group

Figure 3-1 shows the controls in the Parameters Group.

Figure 3-1. Display Interface Multiplexer IP Core Parameters Group



Parameters

Pixels per clock: [✓ 1] 2

Color depth: 6-bits [✓ 8-bits]

Number of PLLs: 1 [✓ 2]

Switch instant: Immediate [✓ Start of Vsync] End of Vsync

Vsync stream: Current [✓ New]

Pixels per clock specifies the number of video pixels transmitted per clock. The interface standards allow transport of one pixel or two pixels in one clock. If one pixel per clock is chosen, three or four (for 6-bit color and 8-bit color respectively) LVDS data pairs are used for the transport. The data mapping schemes for one pixel per clock transport are shown in Figure 2-2 on page 5 and Figure 2-3 on page 6. If two pixels per clock transport is chosen, six or eight (for 6-bit color and 8-bit color respectively) LVDS data pairs are used. The data mapping schemes for two pixels per clock transport are shown in Figure 2-4 on page 6 and Figure 2-5 on page 6.

Color depth is the number of bits used to represent each color component of the pixel. Depending on the value of the color depth, three or four LVDS data pairs will be used for each pixel.

The control, “Number of PLLs” lets the user select one or two PLLs for implementing the LVDS receivers for input streams A and B. Two PLLs (one for each input stream) are necessary to do interruption free synchronized switching of the input streams. However, if only one PLL is available in the device, the one PLL option can be used. In that case, since the PLL has to be re-trained for the second stream and the clock and data aligned for the new stream, there will be a break in the output during switching.

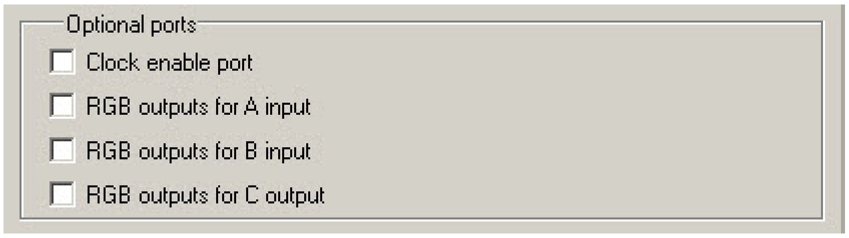
Switch instant allows the user to specify when the multiplexer switches the output after receiving the multiplexer select (mux_sel) control signal. With the “immediate” option, the multiplexer switches to the new output immediately after mux_sel changes state. With “Start of vsync” or “End of vsync” options, the switching happens at the start or end of vsync interval of the chosen stream.

The Vsync stream option specifies the stream whose vsync signal is used in determining the switching instant.

Optional Ports Group

Figure 3-2 shows the Optional Ports Group.

Figure 3-2. Display Interface Multiplexer IP Core Optional Ports Group


Optional ports
☐ Clock enable port
☐ RGB outputs for A input
☐ RGB outputs for B input
☐ RGB outputs for C output

If the clock enable port is checked, an input port named “ce” is made available which is used to control the switching operations in the IP core. As the use of clock enable control increases the resource utilization of the IP core, it must be used only when necessary.

RGB output ports for C output (A input or B input) option brings out color component data and timing signals (vsync, hsync and data enable) for the C output (A input or B input) video stream. The size of the color component outputs (R, G and B) is the same as the chosen color depth (6 or 8). When the number of pixels per clock is two, additional component and sync outputs (with a 2 suffix) corresponding to the second pixel are made available.

IP Core Generation

This chapter provides information on licensing the Display Interface Multiplexer IP core, generating the core using the Diamond or ispLEVER software IPexpress tool, running functional simulation, and including the core in a top-level design.

Licensing the IP Core

An IP license is required to enable full, unrestricted use of the Display Interface Multiplexer IP core in a complete, top-level design. An IP license that specifies the IP core (Display Interface Multiplexer) and device family (MachX02) is required to enable full use of the Display Interface Multiplexer IP core in Lattice MachXO2 devices. Instructions on how to obtain licenses for Lattice IP cores are given at:

http://www.latticesemi.com/products/intellectualproperty/aboutip/isplevercoreonlinepurchas.cfm

Users may download and generate the Display Interface Multiplexer IP core for MachXO2 and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The Display Interface Multiplexer IP core also supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited time (approximately four hours) without requiring an IP license (see "Hardware Evaluation" on page 19 for further details). However, a license is required to enable timing simulation, to open the design in the Diamond or ispLEVER EPIC tool, and to generate bitstreams that do not include the hardware evaluation timeout limitation.

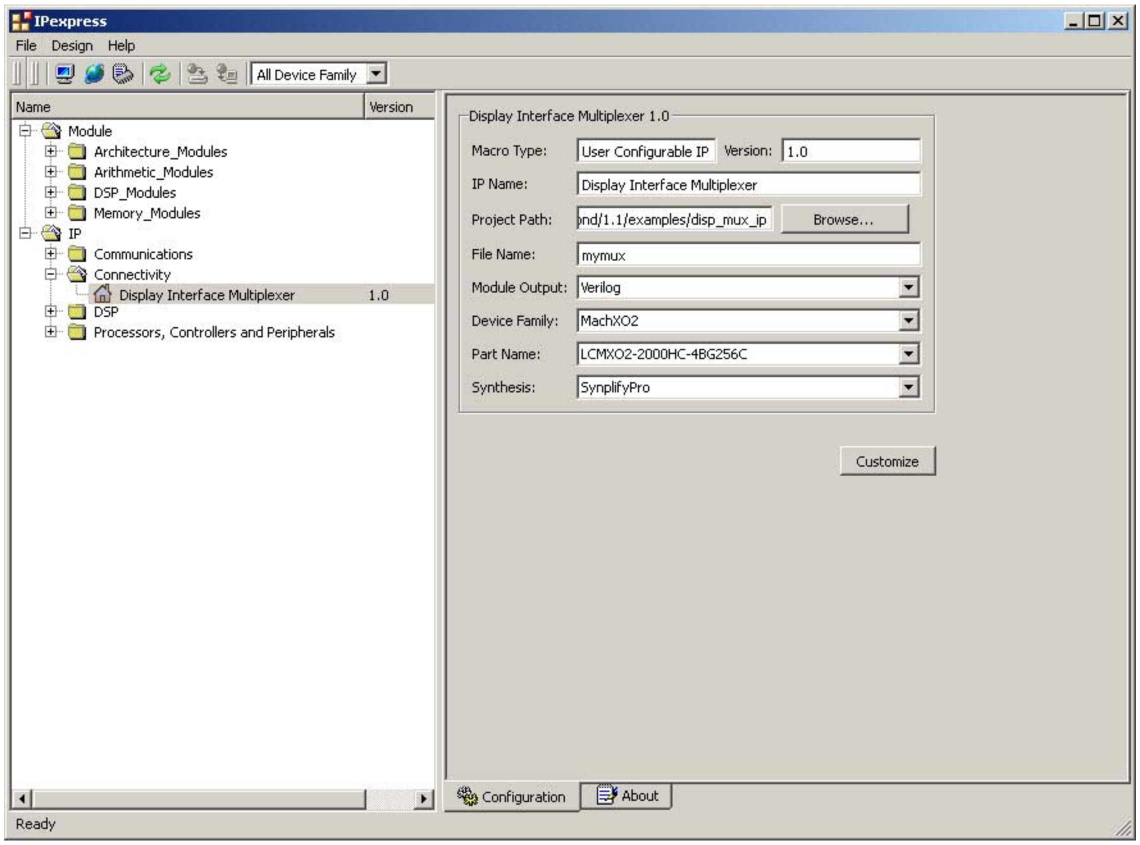
Getting Started

The Display Interface Multiplexer IP core is available for download from the Lattice IP Server using the IPexpress tool in Diamond or ispLEVER. The IP files are automatically installed using ispUPDATE technology in any customer-specified directory. After the IP core has been installed, it will be available in the IPexpress GUI dialog box shown in Figure 4-1.

The IPexpress tool GUI dialog box for the Display Interface Multiplexer IP core is shown in Figure 4-1. To generate a specific IP core configuration the user specifies:

- **Project Path** – Path to the directory where the generated IP files will be loaded.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **(Diamond) Module Output** – Verilog or VHDL.
- **(ispLEVER) Design Entry Type** – Verilog HDL or VHDL
- **Device Family** – Device family to which IP is to be targeted (e.g. MachXO2, etc.). Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

Figure 4-1. IPexpress Dialog Box (Diamond Version)



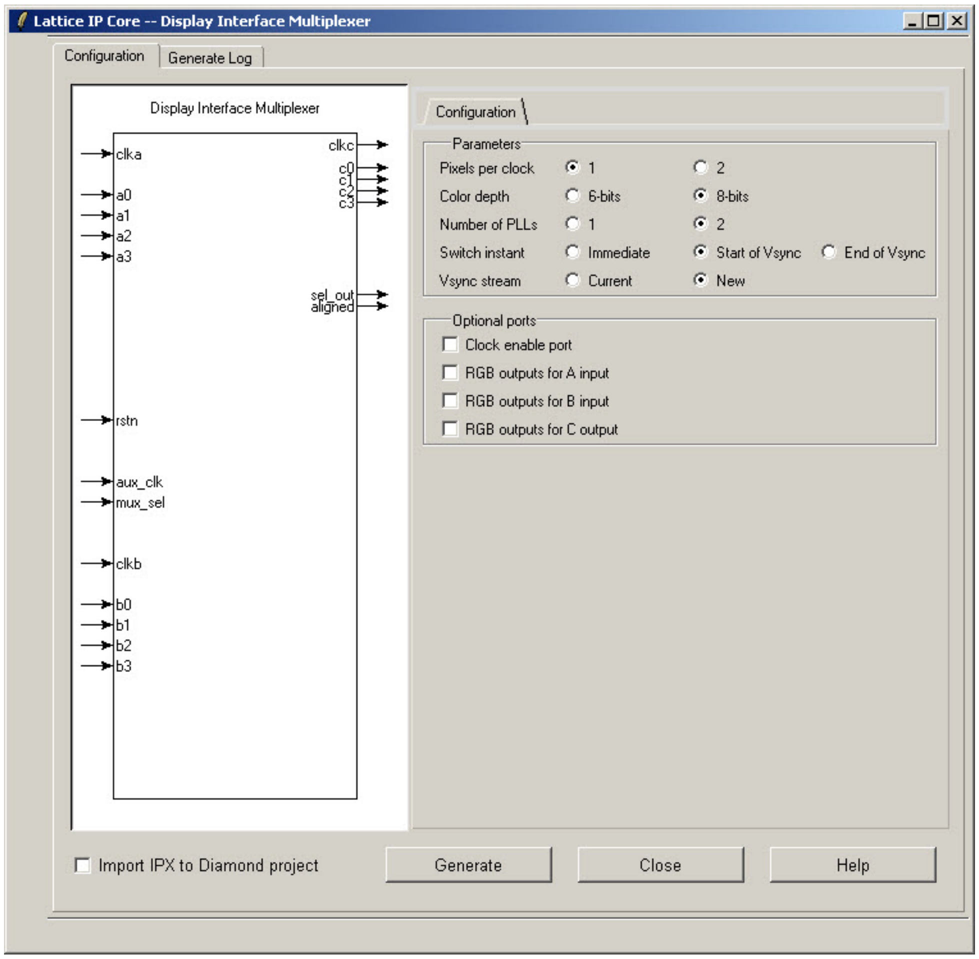
Display Interface Multiplexer 1.0

Macro Type:	User Configurable IP
Version:	1.0
IP Name:	Display Interface Multiplexer
Project Path:	ond/1.1/examples/disp_mux_ip
File Name:	mymux
Module Output:	Verilog
Device Family:	MachXO2
Part Name:	LCMXO2-2000HC-4BG256C
Synthesis:	SynplifyPro

Note that if the IPexpress tool is called from within an existing project, Project Path, Module Output (Design Entry in ispLEVER), Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To generate an IP configuration, the user clicks the **Customize** button in the IPexpress tool dialog box to display the Display Interface Multiplexer IP core Configuration GUI, as shown in Figure 4-2. From this dialog box, the user can select the IP parameter options specific to their application. Refer to “Parameter Settings” on page 12 for more information on the Display Interface Multiplexer parameter settings.

Figure 4-2. Configuration GUI (Diamond Version)



Parameters

Pixels per clock	[✓ 1]	2	
Color depth	6-bits	[✓ 8-bits]	
Number of PLLs	1	[✓ 2]	
Switch instant	Immediate	[✓ Start of Vsync]	End of Vsync
Vsync stream	Current	[✓ New]	

Optional ports

- ☐ Clock enable port
- ☐ RGB outputs for A input
- ☐ RGB outputs for B input
- ☐ RGB outputs for C output

☐ Import IPX to Diamond project

IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button in the IP Configuration dialog box, the IP core and supporting files are generated in the specified “Project Path” directory. The directory structure of the generated files is shown in Figure 4-3.

Figure 4-3. Display Interface Multiplexer Core Directory Structure

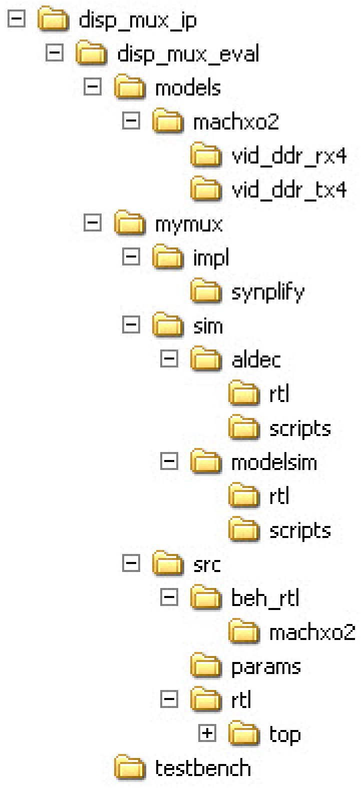


Table 4-1 provides a list of key files and directories created by the IPexpress tool and how they are used. The IPexpress tool creates several files that are used throughout the design cycle. The names of most of the created files are customized to the user's module name specified in the IPexpress tool.

Table 4-1. File List

File	Simulation	Synthesis	Description
<*username*>_inst.v			This file provides an instance template for the IP.
<*username*>_beh.v	Yes		This file provides the front-end simulation library for the Display Interface Multiplexer IP core.
disp_mux_params.v	Yes		This file provides the user options of the IP for the simulation model.
<*username*>_bb.v		Yes	This file provides the synthesis black box for the user's synthesis.
<*username*>.ngo		Yes	This file provides the synthesized IP core used by the Diamond or ispLEVER software. This file needs to be pointed to by the Build step by using the search path property.
<*username*>.lpc			This file contains the IPexpress tool and Display Interface Multiplexer IP GUI options used to recreate or modify the core in the IPexpress tool.
<*username*>.ipx			The IPX file holds references to all of the elements of an IP or Module after it is generated from the IPexpress tool (Diamond version only). The file is used to bring in the appropriate files during the design implementation and analysis. It is also used to re-load parameter settings into the IP/Module generation GUI when an IP/Module is being re-generated.
pmi_*.ngo			These files contain the distributed memories used by the IP core. These files need to be pointed to by the Build step by using the search path property.

Table 4-1. File List (Continued)

disp_mux_eval			This directory contains a sample design. This design is capable of performing a simulation and running through the ispLEVER software.

Most of the files required to use the Display Interface Multiplexer IP core in a user's design reside in the root directory created by the IPexpress tool. This includes the synthesis black box, simulation model, and post synthesis NGO files for the PMI modules.

The `\disp_mux_eval` and subtending directories provide files supporting Display Interface Multiplexer IP core evaluation. The `\disp_mux_eval` directory contains files/folders with content that is constant for all configurations of the Display Interface Multiplexer IP core. The `\<username>` subfolder (`\mymux` in this example) contains files/folders with content specific to the <*username*> configuration.

The Display Interface Multiplexer ReadMe document is also provided in the `\disp_mux_eval` directory.

For example information and known issues on this core, see the Lattice Display Interface Multiplexer ReadMe document. This file is available when the core is installed in the Diamond or ispLEVER software. The document provides information on creating an evaluation version of the core for use in Diamond or ispLEVER and simulation.

The `\disp_mux_eval` directory is created by the IPexpress tool the first time the core is generated and updated each time the core is regenerated. A `\<username>` directory is created by the IPexpress tool each time the core is generated and regenerated each time the core with the same file name is regenerated. A separate `\<username>` directory is generated for cores with different names, e.g. `\<my_core_0>`, `\<my_core_1>`, etc.

The `\disp_mux_eval` directory provides an evaluation design which can be used to determine the size of the IP core and a design which can be pushed through the Diamond or ispLEVER software including front-end and timing simulations. The models directory provides the library element for the DDR I/Os.

The `\<username>` directory contains the sample design for the configuration specified by the customer. The \<username>\impl directory provides project files supporting Synplify synthesis flow. The sample top-level design pulls the user ports out to external pins. This design and associated project files can be used to determine the size of the core and to push it through the mechanics of the Diamond or ispLEVER software design flow.

The `\<username>\sim` directory provides project files supporting RTL and timing simulation for both the Active-HDL and ModelSim simulators. The `\<username>\src` directory provides the top-level source code for the eval design. The \testbench directory provides a top-level testbench file.

Running Functional Simulation

Simulation support for the Display Interface Multiplexer IP core is provided for Aldec and ModelSim simulators. The Display Interface Multiplexer IP core simulation model is generated from the IPexpress tool with the name `<username>_beh.v`. This file is an obfuscated simulation model. An obfuscated simulation model is Lattice's unique IP protection technique which scrambles the Verilog HDL while maintaining logical equivalence. VHDL users will use the same Verilog model for simulation.

When compiling the Display Interface Multiplexer IP core, the file disp_mux_params.v must be compiled with the model. This files provides define constants that are necessary for the simulation model.

The ModelSim environment is located in `\<project_dir>\disp_mux_eval\<username>\sim\modelsim`. Users can run the ModelSim simulation by performing the following steps:

1. Open ModelSim.
2. Under the File tab, select **Change Directory** and choose folder `\<project_dir>\disp_mux_eval\<username>\sim\modelsim\scripts`.
3. Under the Tools tab, select **Tcl > Execute Macro** and execute the ModelSim "do" script shown.

The Aldec Active-HDL environment is located in
`\<project_dir>\disp_mux_eval\<username>\sim\aldec`. Users can run the Aldec evaluation simulation by performing the following steps:

1. Open Active-HDL.
2. Under the Tools tab, select **Execute Macro**.
3. Browse to the directory `\<project_dir>\disp_mux_eval\<username>\sim\aldec\scripts` and execute the Active-HDL "do" script shown.

Synthesizing and Implementing the Core in a Top-Level Design

The Display Interface Multiplexer IP core itself is synthesized and provided in NGO format when the core is generated through the IPexpress tool. You can combine the core in your own top-level design by instantiating the core in your top level file as described above in the "Instantiating the Core" section and then synthesizing the entire design with Synplify.

The top-level file <*username*>_top.v provided in
`\<project_dir>\disp_mux_eval\<username>\src\top\<device_family>`
supports the ability to implement the Display Interface Multiplexer IP core in isolation. Push-button implementation of this top-level design with Synplify is supported via the Diamond or ispLEVER software project files <*username*>_top.ldf or <*username*>_top.syn located in the
`\<project_dir>\disp_mux_eval\<username>\impl\synplify` directory.

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to
   `\<project_dir>\disp_mux_eval\<username>\impl\synplify` in the Open Project dialog box.
3. Select and open <*username*>.ldf. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the **Process** tab in the left-hand GUI window.
5. Implement the complete design via the standard Diamond GUI flow.

To use this project file in ispLEVER:

1. Choose **File > Open Project.**
2. Browse to
   `\<project_dir>\disp_mux_eval\<username>\impl\synplify` in the Open Project dialog box.
3. Select and open <*username*>.syn. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the device top-level entry in the left-hand GUI window.
5. Implement the complete design via the standard ispLEVER GUI flow.

Hardware Evaluation

The Display Interface Multiplexer IP core supports supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs.

Enabling Hardware Evaluation in Diamond

Choose **Project > Active Strategy > Translate Design Settings**. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default.

Enabling Hardware Evaluation in ispLEVER

In the Processes for Current Source pane, right-click the **Build Database** process and choose **Properties** from the dropdown menu. The hardware evaluation capability may be enabled/disabled in the Properties dialog box. It is enabled by default.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, you can modify any of its settings including device type, design entry method, and any of the options specific to the IP core. Regenerating can be done to modify an existing IP core or to create a new but similar one.

Regenerating an IP Core in Diamond

To regenerate an IP core in Diamond:

1. In IPexpress, click the **Regenerate** button.
2. In the Regenerate view of IPexpress, choose the IPX source file of the module or IP you wish to regenerate.
3. IPexpress shows the current settings for the module or IP in the Source box. Make your new settings in the T**arget** box.
4. If you want to generate a new set of files in a new location, set the new location in the **IPX Target File** box. The base of the file name will be the base of all the new file names. The IPX Target File must end with an .ipx extension.
5. Click **Regenerate.** The module's dialog box opens showing the current option settings.
6. In the dialog box, choose the desired options. To get information about the options, click **Help**. Also, check the About tab in IPexpress for links to technical notes and user guides. IP may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.
7. To import the module into your project, if it's not already there, select **Import IPX to Diamond Project** (not available in stand-alone mode).
8. Click **Generate**.
9. Check the Generate Log tab to check for warnings and error messages.
10. Click **Close**.

The IPexpress package file (.ipx) supported by Diamond holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing the .ipx file to the associated Diamond project. To change the option settings of a module or IP that is already in a design project, double-click the module's .ipx file in the File List view. This opens IPexpress and the module's dialog box showing the current option settings. Then go to step 6 above.

Regenerating an IP Core in ispLEVER

To regenerate an IP core in ispLEVER:

1. In the IPexpress tool, choose **Tools > Regenerate IP/Module**.

2. In the Select a Parameter File dialog box, choose the Lattice Parameter Configuration (.lpc) file of the IP core you wish to regenerate, and click **Open**.
3. The Select Target Core Version, Design Entry, and Device dialog box shows the current settings for the IP core in the Source Value box. Make your new settings in the Target Value box.
4. If you want to generate a new set of files in a new location, set the location in the LPC Target File box. The base of the .lpc file name will be the base of all the new file names. The LPC Target File must end with an .lpc extension.
5. Click **Next**. The IP core's dialog box opens showing the current option settings.
6. In the dialog box, choose desired options. To get information about the options, click **Help**. Also, check the About tab in the IPexpress tool for links to technical notes and user guides. The IP core might come with additional information. As the options change, the schematic diagram of the IP core changes to show the I/O and the device resources the IP core will need.
7. Click **Generate**.
8. Click the **Generate Log** tab to check for warnings and error messages.

Application Support

This chapter provides application support information for the MachXO2 Display Interface Multiplexer IP core.

Using the IP Core

This section provides supporting information on how to use the Display Interface Multiplexer IP core in complete designs. Topics discussed include IP simulation and verification and Display Interface Multiplexer top-level design.

Simulation and Verification

This section discusses strategies and approaches for verifying the proper functionality of the Display Interface Multiplexer IP core through simulation.

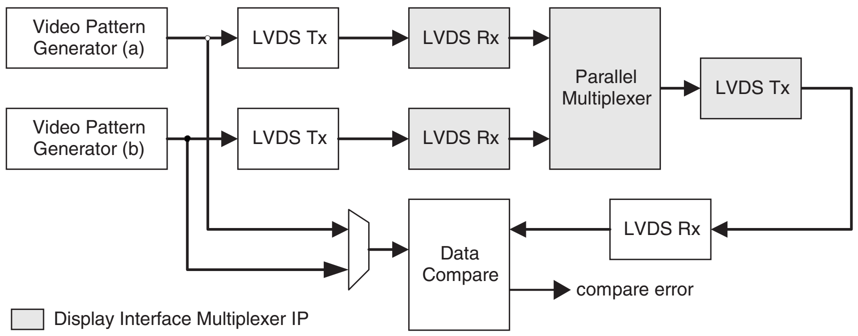
Simulation Strategy

Included with the core from the IPexpress tool is the evaluation testbench located in the *<username>* directory. The intent of the evaluation testbench and the demo design is to show the core performing in simulation.

Display Interface Multiplexer IP Demo Design

The IP sample design is a self-checking top-level design that contains the IP, stimulus generator and comparison logic. The general scheme of the demo design is shown in Figure 5-1.

Figure 5-1. Display Interface Multiplexer IP Demo Design



A brief description of each of the major blocks of the demo design is given below.

Video Pattern Generator
A video pattern generator is required in order to effectively test the synchronized switching functions. The pattern generator that is part of the demo design generates pseudo video pattern that has active video, horizontal blanking and vertical blanking instants, but at a scaled down level. The number of pixels in an active line, blanking periods and number of lines per frame are made very small to allow faster simulation time.

Data Comparison
A data comparison module is provided in the demo design to verify the correctness of the received data. This block compares the delayed version of the stimulus data with the parallelized version of the IP output data. A training sequence is used to align the stimulus and IP output data.

Testbench
The testbench instantiates the demo design and supplies all the required clocks to it. Since the self-checking capability is provided in the top-level demo design, the test bench is not required to do data comparisons. The testbench provides the control signals such as mux_sel and counts the number of compare errors.

Core Verification

The functionality of the Lattice Display Interface Multiplexer IP core has been verified via RTL and netlist simulations.

The RTL simulation used pseudo video patterns using asynchronous clocks and non-synchronized video frames.

Netlist simulations were carried out by using the post-route Verilog netlist generated by Diamond software in place of the IP RTL model and verified for accuracy.

Support Resources

This chapter contains information about Lattice Technical Support, additional references, and document revision history.

Lattice Technical Support

There are a number of ways to receive technical support.

Online Forums

The first place to look is Lattice Forums (http://www.latticesemi.com/support/forums.cfm). Lattice Forums contain a wealth of knowledge and are actively monitored by Lattice Applications Engineers.

Telephone Support Hotline

Receive direct technical support for all Lattice products by calling Lattice Applications from 5:30 a.m. to 6 p.m. Pacific Time.

- For USA & Canada: 1-800-LATTICE (528-8423)
- For other locations: +1 503 268 8001

In Asia, call Lattice Applications from 8:30 a.m. to 5:30 p.m. Beijing Time (CST), +0800 UTC. Chinese and English language only.

- For Asia: +86 21 52989090

E-mail Support

- techsupport@latticesemi.com
- techsupport-asia@latticesemi.com

Local Support

Contact your nearest Lattice Sales Office.

Internet

www.latticesemi.com

References

1. Flat Panel Display (FPD) Link, National Semiconductors, 2009
2. Open LDI- Open LVDS Display Interface Standard Specification, National Semiconductors, http://www.national.com/analog/displays/open_LDI
3. VESA Notebook Panel Standard, Ver 1, October 2007

MachXO2

- DS1035, *MachXO2 Family Datasheet*
- RD1093, *MachXO2 Display Interface*
- TN1203, *Implementing High-Speed Interfaces with MachXO2 Devices*

Revision History

Date	Document Version	IP Core Version	Change Summary
November 2010	01.0	1.0	Initial release.

Resource Utilization

This appendix gives resource utilization information for Lattice devices using the Display Interface Multiplexer IP Core. The IP configurations shown in this chapter were generated using the IPexpress software tool. IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the Diamond and ispLEVER design tools. Details regarding the usage of IPexpress can be found in the IPexpress and Diamond and ispLEVER help systems. For more information on the Diamond or ispLEVER design tools, visit the Lattice web site at: www.latticesemi.com/software.

Lattice MachXO2 FPGAs

Table A-1. Performance and Resource Utilization[1]

Sample Configuration	Slices	LUTs	Registers	PIOs	Tx sclk	Rx sclk	Rx sclka	Rx sclkb
1	145	287	170	29	235	125	-	-
2	161	319	195	35	218	121	-	-
3	196	390	239	47	186	118	-	-
4	224	448	279	59	191	109	-	-
5	221	436	245	29	231	-	126	119
6	232	461	266	35	216	-	120	129
7	268	533	308	47	183	-	117	118
8	298	591	352	59	174	-	115	119

1. Performance and utilization characteristics are generated using an LCMXO2-4000HCCABGA256 device in Diamond 1.1 software. Performance may vary when using this IP core in a different density, speed or grade within the MachXO2 family.

Table A-2. Parameter Settings for Sample Configurations[1]

Parameters	1	2	3	4	5	6	7	8
Number of PLLs	1	1	1	1	2	2	2	2
Pixels per clock	1	1	2	2	1	1	2	2
Color depth	6	8	6	8	6	8	6	8
LVDS data pairs	3	4	6	8	3	4	6	8

1.The Display Interface Multiplexer IP core is an IPexpress user-configurable core and can be used to generate any allowable configuration.

Ordering Information

The Ordering Part Number (OPN) for all configurations of the Display Interface Multiplexer IP core targeting Lattice MachXO2 devices is DIM-M2-U1.