



# MIPI DSI TX Controller Core User Guide

---

UG-CORE-MIPI-DSI-TX-v3.1  
February 2026  
[www.efinixinc.com](http://www.efinixinc.com)



# Contents

<b>Introduction.....</b>	<b>3</b>
<b>Features.....</b>	<b>3</b>
<b>Device Support.....</b>	<b>3</b>
<b>Resource Utilization and Performance.....</b>	<b>4</b>
<b>Release Notes.....</b>	<b>4</b>
<b>Functional Description.....</b>	<b>5</b>
Ports.....	5
Clocking.....	9
Register Definition.....	10
Video Mode Configuration.....	13
Command Packet Data Types.....	13
Sync Event Packet Data Type.....	14
Video Mode Pixel Encoding.....	14
MIPI Video Data DATA[63:0] Formats.....	15
Pixel Clock Calculation.....	16
Video Timing Parameters.....	17
Reset Sequence and Initialization.....	18
<b>IP Manager.....</b>	<b>19</b>
<b>Customizing the MIPI DSI TX Controller.....</b>	<b>20</b>
<b>MIPI DSI TX Controller Example Design.....</b>	<b>23</b>
<b>MIPI DSI TX Controller Testbench.....</b>	<b>25</b>
<b>Revision History.....</b>	<b>26</b>

# Introduction

The MIPI DSI specifies the physical link between the chip and display in devices such as smartphones, tablets, AR/VR headsets and connected cars<sup>(1)</sup>. It defines a serial bus and a communication protocol between the host, the source of the image data, and the destination, for example, display peripherals. The MIPI DSI TX Controller core implements the MIPI DSI interface in the FPGA and allows you to configure the related parameters.

Use the IP Manager to select IP, customize it, and generate files. The MIPI DSI TX Controller core has an interactive wizard to help you set parameters. The wizard also has options to create a testbench and/or example design targeting an Efinix<sup>®</sup> development board.

## Features

- Supports 1,2, and 4 lanes
- Supports continuous or discontinuous clock mode
- IP core clock frequency at 100 MHz
- 8-bit HS mode data width
- HS mode byte clock frequency from 10 MHz up to 187 MHz (from 80 Mbps up to 1,500 Mbps data rate)<sup>(2)</sup>
- Includes AXI4-Lite interface for register access
- Error correction code (ECC) generation for packet headers
- Cyclic redundancy check (CRC) generation for data bytes
- Supports non-burst with sync pulses, non-burst with sync events, and burst mode
- Supports end-of-transmission packet
- Supports command transmission in HS or LP mode

## Device Support

*Table 1: MIPI DSI TX Controller Core Device Support*

FPGA Family	Supported Device
Trion	-
Titanium and Topaz	All

<sup>(1)</sup> Source: MIPI Alliance.

<sup>(2)</sup> The maximum data rate of IP depends on the devices. Refer to the respective device data sheet for more accurate information.

# Resource Utilization and Performance



**Note:** The resources and performance values provided are based on some of the supported FPGAs. These values are just guidance and may change depending on the device resource utilization, design congestion, and user design.

**Table 2: Titanium Resource Utilization and Performance**

MIPI DSI TX Controller with 4 data lanes.

FPGA	Logic Elements (Logic, Adders, Flipflops,etc.)	Memory Blocks	DSP Blocks	$f_{MAX}$ (MHz) <sup>(3)</sup>				Efinity® Version <sup>(4)</sup>
				clk	axi_clk	clk_byte_HS	clk_pixel	
Ti60 F225 C4	5,737 / 60,800 (9.44%)	35/256 (13.67%)	0/160 (0%)	205	252	227	261	2023.2.307.4.14

## Release Notes

You can refer to the IP Core Release Notes for more information about the IP core changes. The IP Core Release Notes are available in the [Efinity page of the Support Center](#) under each Efinity software release version.



**Note:** You must be logged in to the Support Center to view the IP Core Release Notes.

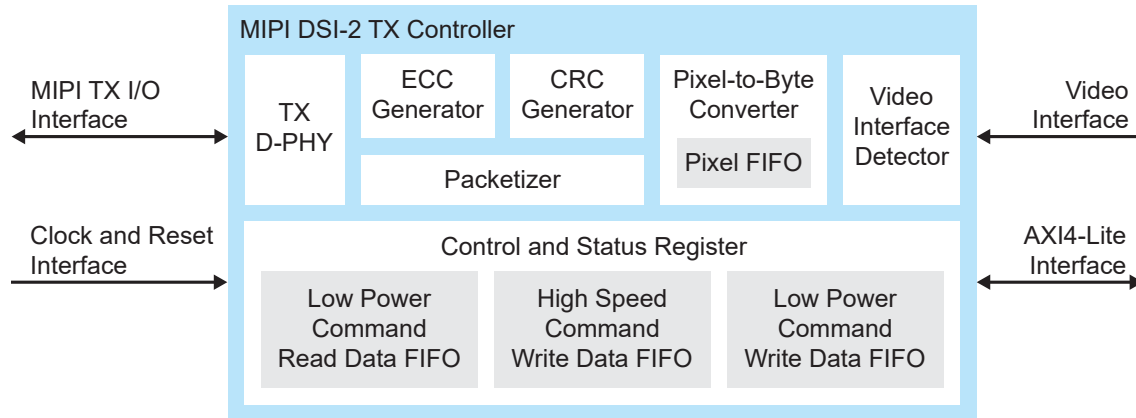
<sup>(3)</sup> Using default parameter settings.

<sup>(4)</sup> Using System Verilog.

# Functional Description

The MIPI DSI TX Controller consists of a TX D-PHY block, control status registers, ECC generator, CRC generator, packetizer, pixel-to-byte converter, and video interface detector. The core has a video, AXI4-lite, MIPI TX I/O, and clock and reset interfaces.

Figure 1: MIPI DSI TX Controller System Block Diagram



## Ports

Table 3: Clock and Reset Ports

Port	Direction	Description
clk	Input	IP core clock consumed by controller logic. 100 MHz.
reset_n	Input	IP core reset signal.
clk_byte_HS	Input	MIPI TX parallel clock. This clock is a HS mode transmission clock.
reset_byte_HS_n	Input	MIPI TX parallel clock reset signal.
clk_pixel	Input	Pixel clock.
reset_pixel_n	Input	Pixel clock reset signal.
axi_clk	Input	AXI4-Lite interface clock.
axi_reset_n	Input	AXI4-Lite interface reset.

**Table 4: MIPI TX I/O interface**

Port	Direction	Description
Tx_LP_CLK_P	Output	LP mode TX clock single-ended P signal.
Tx_LP_CLK_P_OE	Output	Output enable for LP mode TX clock single-ended P signal.
Tx_LP_CLK_N	Output	LP mode TX clock single-ended N signal.
Tx_LP_CLK_N_OE	Output	Output enable for LP mode TX clock single-ended N signal.
Tx_HS_C [7:0]	Output	HS mode differential clock bus.
Tx_HS_enable_C	Output	Signal to enable HS mode clock lane.
Tx_LP_D_P [NUM_DATA_LANE-1:0]	Output	LP mode TX data single-ended P signal.
Tx_LP_D_P_OE [NUM_DATA_LANE-1:0]	Output	Output enable for LP mode TX data single-ended P signal.
Tx_LP_D_N [NUM_DATA_LANE-1:0]	Output	LP mode TX data single-ended N signal.
Tx_LP_D_N_OE [NUM_DATA_LANE-1:0]	Output	Output enable for LP mode TX data single-ended N signal.
Tx_HS_D_0[7:0]	Output	HS mode differential lane 0 data bus.
Tx_HS_D_1[7:0]	Output	HS mode differential lane 1 data bus.
Tx_HS_D_2[7:0]	Output	HS mode differential lane 2 data bus.
Tx_HS_D_3[7:0]	Output	HS mode differential lane 3 data bus.
Tx_HS_enable_D [NUM_DATA_LANE-1:0]	Output	Signal to enable HS mode data lane.
Rx_LP_D_P	Input	LP mode RX data single-ended P signal.
Rx_LP_D_N	Input	LP mode RX data single-ended N signal.

**Table 5: Video Interface**

All signals are clocked with clk\_pixel and reset\_pixel\_n.

Port	Direction	Description
hsync	Input	Active-low horizontal sync.
vsync	Input	Active-low vertical sync.
datatype [5:0]	Input	Data type of the HS packet. Sampled at Hsync rising edge.
pixel_data [63:0]	Input	Video Data. The actual width is dependent on pixel type. See <b>Video Mode Pixel Encoding</b> on page 14.
pixel_data_valid	Input	Active-high pixel data enable. Once the TX VALID signal goes high, the MIPI TX interface expects to receive pixel data every clock cycle until the entire line is sent. Additionally, the TX VALID signal must remain high for the entire line.
haddr [15:0]	Input	16 bit horizontal number of pixels. Sampled at Hsync rising edge.
vc [1:0]	Input	2-bit virtual channel signal.

**Table 6: Conduit Interface**

Port	Direction	Description
TurnRequest_dbg	Input	User control turnaround request. This active high signal is used to indicate that the protocol desires to initiate a bidirectional data lane turnaround, to allow the other side to begin transmissions. TurnRequest is valid on rising edge of clk. TurnRequest is only meaningful for a bidirectional data lane module that is currently the transmitter (Direction=0). If the bidirectional data lane module is in receive mode (Direction=1), this signal is ignored. A low-to-high transition on TurnRequest can only happen when Stopstate is asserted.
TurnRequest_done	Output	Indicates that the RX D-PHY acknowledges the bus turnaround or timeout. If this signal is high together with turnaround timeout, it indicates that there is no acknowledgement from the RX on the turnaround request.
irq	Output	Interrupt signal for Interrupt Status Register.

**Table 7: AXI4-Lite Interface**

All signals are clocked with axi\_clk.

Port	Direction	Description
axi_awaddr [6:0]	Input	AXI4-Lite write address bus.
axi_awvalid	Input	AXI4-Lite write address valid strobe.
axi_awready	Output	AXI4-Lite write address ready signal.
axi_wdata [31:0]	Input	AXI4-Lite write data.
axi_wvalid	Input	AXI4-Lite write data valid strobe.
axi_wready	Output	AXI4-Lite write ready signal.
axi_bvalid	Output	AXI4-Lite write response valid strobe.
axi_bready	Input	AXI4-Lite write response ready signal.
axi_araddr [6:0]	Input	AXI4-Lite read address bus.
axi_arvalid	Input	AXI4-Lite read address valid strobe.
axi_arready	Output	AXI4-Lite read address ready signal.
axi_rdata [31:0]	Output	AXI4-Lite read data.
axi_rvalid	Output	AXI4-Lite read data valid strobe.
axi_rready	Input	AXI4-Lite read data ready signal.

**Table 8: Debug Interface**

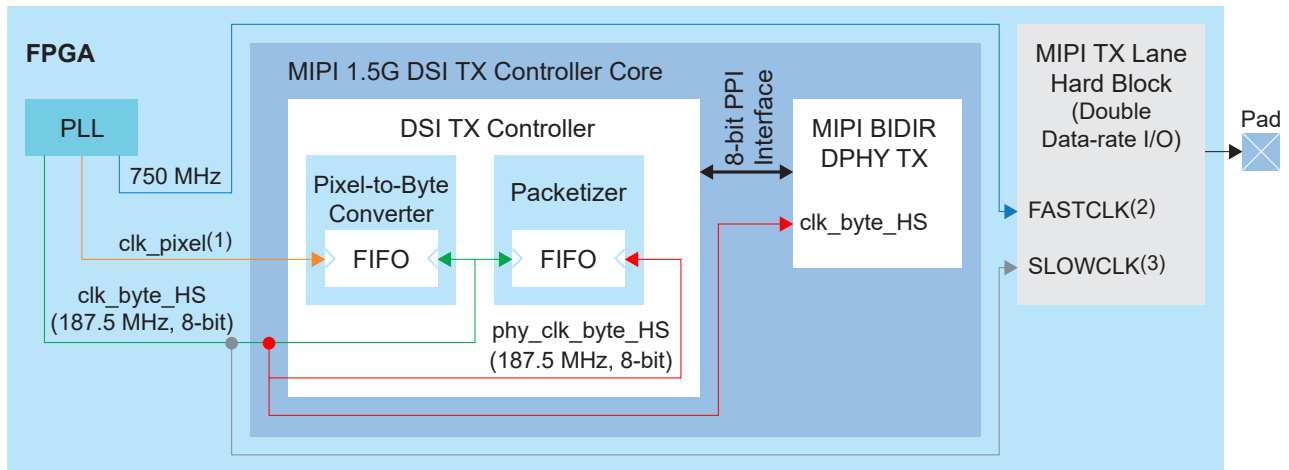
All signals are clocked with axi\_clk and axi\_reset\_n.

Port	Direction	Description
mipi_debug_out[31:0]	Output	<p>Debug port. Present if the parameter <b>MIPI_DSI_TX_DEBUG</b> is <b>Enabled</b>. The following is the list of internal signals that can be monitored.</p> <ul style="list-style-type: none"> <li>[0] = fifo_full</li> <li>[1] = fifo_empty</li> <li>[2] = non_support_longpkt</li> <li>[3] = turnaround_timeout</li> <li>[4] = ready_to_xmit (indicator for initialization done, prior to skewcal)</li> <li>[5] = init_skewcal_done (indicator for skewcal process completion)</li> <li>[6] = lp_dcs_rfifo_full</li> <li>[7] = lp_dcs_rfifo_empty</li> <li>[8] = lp_dcs_wfifo_full</li> <li>[9] = lp_dcs_wfifo_empty</li> <li>[10] = hs_dcs_wfifo_full</li> <li>[11] = hs_dcs_wfifo_empty</li> <li>[12] = lp_cmd_in_progress</li> <li>[13] = hs_cmd_in_progress</li> <li>[14] = TxStopState_0</li> <li>[15] = TxStopState_1</li> <li>[16] = TxStopState_2</li> <li>[17] = TxStopState_3</li> <li>[18] = TxStopState_4</li> <li>[19] = TxStopState_5</li> <li>[20] = TxStopState_6</li> <li>[21] = TxStopState_7</li> <li>[31:22] = Reserved</li> </ul>
mipi_debug_in[31:0]	Input	<p>Debug ports. Present if the parameter <b>MIPI_DSI_TX_DEBUG</b> is <b>Enabled</b>.</p> <p>Currently no function has been implemented. Tie all input bits to zero.</p> <p>[31:0] = reserved</p>

## Clocking

The following diagram illustrates the example of clock settings for a 1.5 Gbps DSI TX implementation.

Figure 2: 1.5 Gbps DSI TX Clocking Example



- (1) Refer to Pixel Clock Calculation section for the `clk_pixel` value.
- (2) **Serial Clock Pin Name** in the Interface Designer.
- (3) **Parallel Clock Pin Name** in the Interface Designer.

## Register Definition



**Table 9: Control Status Registers**

Word Offset	Bits	Name	R/W	Width (bits)
0x00	4:0	Interrupt status register.	R/W1C	5
0x04	4:0	Interrupt enable register.	R/W	5
0x08	7:0	PHY stop state status.	RO	8
0x0C	0	TxUlpsActiveClkNot.	RO	9
	8:1	TxUlpsActiveNot_7: TxUlpsActiveNot_0.		
0x10	7:0	Skew calibration high speed.	R/W	8
0x14	0	UlpsClk.	R/W	13
	8:1	UlpsEsc[7:0].		
	12:9	TxTriggerEsc.		
0x18	0	Reserved.	R/W	4
	1	Reserved.		
	2	Reserved.		
	3	video_stream_en. 1: Turn on HS video stream on the MIPI lane 0: Turn off HS video stream on the MIPI lane		
0x1C	HS Command Queue. Ensure that the bit <b>11 of the status register</b> is low before issuing the next command.		R/W	24
	7:0	datatype.		
	15:8	Parameter 1.		
	23:16	Parameter 2.		
0x20	LP Command Queue. Ensure that the bit <b>10 of the status register</b> is low before issuing the next command.		R/W	24
	7:0	datatype.		
	15:8	Parameter 1.		
	23:16	Parameter 2.		
0x24	19:0	Status register.	RO	20
0x28	31:0	Low power command write long data FIFO. You must write the LP write data to this FIFO before issuing the LP command packet to register 0x20. Store only one complete write packet data in the FIFO at a time.	WO	32

Word Offset	Bits	Name	R/W	Width (bits)
0x2C	31:0	High speed command write long data FIFO. You must write the HS write data to this FIFO before issuing the HS command packet to register 0x1C. Store only one complete write packet data in the FIFO. <sup>(5)</sup>	WO	32
0x30	7:0	Low power command read long data FIFO. The bus turnaround read data is pushed into this FIFO. Store only one complete read packet data in the FIFO at a time. The MIPI DSI TX Controller stores all the return read data into the read FIFO and does not check whether the return read data matches maximum return packet size (MRPS).	RO	8
0x34, 0x38, 0x3C	Reserved			
0x40	31:0	Total H line word count in byte.	R/W	32
0x44	15:0	Horizontal sync active (HSA) in byte. Only write to this register when it is sync pulse mode.	R/W	16
0x48	15:0	Horizontal black porch (HBP) in byte. For burst event mode, factor in HSA value into the HBP value.	R/W	16
0x4C	15:0	Horizontal front porch (HFP) in byte.	R/W	16
0x50	7:0	Vertical sync active (VSA) in line. The minimum number of lines is 1.	R/W	8
0x54	7:0	Vertical black porch (VBP) in line. The minimum number of lines is 1.	R/W	8
0x58	7:0	Vertical front porch (VFP) in line. The minimum number of lines is 2.	R/W	8
0x5C	15:0	Vertical active (VACT) in line. The minimum number of lines is 1.	R/W	16

<sup>(5)</sup> The word count for a HS write long command data has to be larger or equal than the number of MIPI data lane (NUM\_DATA\_LANE).

**Table 10: 0x24 Status Register Definition**

Bit	Description
0	lp_dcs_rfifo_full. LP command read data FIFO full.
1	lp_dcs_rfifo_empty. LP command read data FIFO empty.
2	lp_dcs_wfifo_full. LP command write data FIFO full.
3	lp_dcs_wfifo_empty. LP command write data FIFO empty.
4	hs_dcs_wfifo_full. HS command write data FIFO full.
5	hs_dcs_wfifo_empty. HS command write data FIFO empty.
6	Reserved.
7	Reserved.
8	Reserved.
9	Reserved.
10	lp_cmd_in_progress. LP command transmission in LP lane is in progress.   <b>Note:</b> After writing LP cmd (addr: 0x20), lp_cmd_in_progress will not flag high immediately. Please wait at least 10 cycles of axi_clk prior to polling this indicator.
11	hs_cmd_in_progress. HS command transmission in HS lane is in progress.   <b>Note:</b> After writing HS cmd (addr: 0x1c), hs_cmd_in_progress will not flag high immediately. Please wait at least 10 cycles of axi_clk prior to polling this indicator.

**Table 11: 0x00 Interrupt Status Register Definition**

Bit	Description
0	Pixel FIFO full.
1	Pixel FIFO empty.
2	Unsupported video data type.
3	Turnaround timeout.

## Video Mode Configuration

The MIPI DSI TX Controller core supports the following video modes:

- Non-burst with sync pulses
- Non-burst with sync events
- Burst mode

The following table describes the configuration for each video mode based on blanking or low-power interval (BLLP) mode setting.

**Table 12: Video Mode Settings**

Mode	HSA	HBP	HFP	BLLP
Non-burst with Sync Pulse	HS Blank Packet	HS Blank Packet	HS Blank Packet	HS Blank Packet
Non-burst with Sync Event	HS Blank Packet	HS Blank Packet	HS Blank Packet	HS Blank Packet
Burst	HS Blank Packet	HS Blank Packet	LP-11	LP-11

## Command Packet Data Types

The following table describes the supported command packet data types. The command packets are sent through the command register. In LP mode, the command packets are sent through lane 0. Use the command to send non-video packets to display peripherals.

**Table 13: Command Packet Data Types**

Type	Data Type	Packet Size	Transfer Mode
0x2	Color mode off command	Short	LP/HS
0x12	Color mode on command	Short	LP/HS
0x22	Shutdown peripheral command	Short	LP/HS
0x32	Turn on peripheral command	Short	LP/HS
0x3	Generic short write, no parameters	Short	LP/HS
0x13	Generic short write, 1 parameters	Short	LP/HS
0x23	Generic short write, 2 parameters	Short	LP/HS
0x4	Generic short read, no parameters	Short	LP/HS
0x14	Generic short read, 1 parameters	Short	LP/HS
0x24	Generic short read, 2 parameters	Short	LP/HS
0x5	DCS short write, no parameters	Short	LP/HS
0x15	DCS short write, 1 parameters	Short	LP/HS
0x6	DCS read	Short	LP/HS
0x37	Set Max Return packet size	Short	LP/HS
0x29	Generic Long write	Long	LP/HS
0x39	DCS long write	Long	LP/HS

## Sync Event Packet Data Type

Sync events are short packets and can accurately represent events like the start and end of sync pulses.

*Table 14: Sync Events*

Data type	Description	Packet Size
0x1	V sync start	Short
0x11	V sync end	Short
0x21	H sync start	Short
0x31	H sync end	Short

## Video Mode Pixel Encoding

*Table 15: Video Mode Pixel Encoding*

TYPE[5:0]	Data Type	Bits per Pixel	Pixels per Pixel Clock	Bytes	Pack Bits	Packet Size	Transfer Mode
0xC	20-bit YCbCr	24	2	6	48	Long	HS
0x1C	24-bit YCbCr	24	2	6	48	Long	HS
0x2C	16-bit YCbCr	16	4	8	64	Long	HS
0x3D	12-bit YCbCr	12	4	6	48	Long	HS
0xE	RGB565	16	4	8	64	Long	HS
0x2E	RGB666 (24-bit)	24	2	6	48	Long	HS
0x3E	RGB888	24	2	6	48	Long	HS
0x0D	RGB101010	30	2	60/8	60	Long	HS

## MIPI Video Data DATA[63:0] Formats

The format depends on the data type. New data arrives on every pixel clock.

**Table 16: Loosely Packed Pixel Stream, 20-bit YCbCr (2-pixels per clock)**

63	48	47					0
0	Pixel 1 and Pixel 2						
N/A	[47:44] 4'h0	[43:34] Y	[33:24] Cr	[23:20] 4'h0	[19:10] Y	[9:0] Cb	

**Table 17: Packed Pixel Stream, 24-bit YCbCr (2-pixels per clock)**

63	48	47					0
0	Pixel 1 and Pixel 2						
N/A	[47:36] Y	[35:24] Cr	[23:12] Y	[11:0] Cb			

**Table 18: Packed Pixel Stream, 16-bit YCbCr (4-pixels per clock)**

63	32			31				0
Pixel 3 and Pixel 4				Pixel 1 and Pixel 2				
[63:56] Y	[55:48] Cr	[47:40] Y	[39:32] Cb	[31:24] Y	[23:16] Cr	[15:8] Y	[7:0] Cb	

**Table 19: Packed Pixel Stream, 12-bit YCbCr (4-pixels per clock)**

63	48	47	24		23			0
Odd Lines								
0	Pixel 3 and Pixel 4			Pixel 1 and Pixel 2				
N/A	[47:40] Y	[39:32] Y	[31:24] Cb	[23:16] Y	[15:8] Y	[7:0] Cb		
Even Lines								
0	Pixel 3 and Pixel 4			Pixel 1 and Pixel 2				
N/A	[47:40] Y	[39:32] Y	[31:24] Cr	[23:16] Y	[15:8] Y	[7:0] Cb		

**Table 20: RGB565 (4-pixels per clock)**

63		48			47			32			31			16			15			0		
Pixel 4			Pixel 3			Pixel 2			Pixel 1													
[63:59]	[58:53]	[52:48]	[47:43]	[42:37]	[36:32]	[31:27]	[26:21]	[20:16]	[15:11]	[10:5]	[4:0]											
Blue	Green	Red	Blue	Green	Red	Blue	Green	Red	Blue	Green	Red											

**Table 21: RGB666 (2-pixels per clock)**

63		48			47			24			23			0		
0		Pixel 2			Pixel 1											
N/A		[47:42]	[41:36]	[35:30]	[29:24]	[23:18]	[17:12]	[11:6]	[5:0]							
		6'h0	Blue	Green	Red	6'h0	Blue	Green	Red							

**Table 22: RGB888 (2-pixels per clock)**

63		48			47			24			23			0		
0		Pixel 2			Pixel 1											
N/A		[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]									
		Blue	Green	Red	Blue	Green	Red									

**Table 23: RGB101010 (2-pixels per clock)**

63		60			59			30			29			0		
0		Pixel 2			Pixel 1											
N/A		[59:50]	[49:40]	[39:30]	[29:20]	[19:10]	[9:0]									
		Blue	Green	Red	Blue	Green	Red									

## Pixel Clock Calculation

The following formula calculates the pixel clock frequency that you need to drive the pixel clock input port, `clk_pixel`.

$$\text{PIX\_CLK\_MHZ} < (\text{DATARATE\_MBPS} * \text{NUM\_DATA\_LANE}) / \text{PACK\_BIT},$$

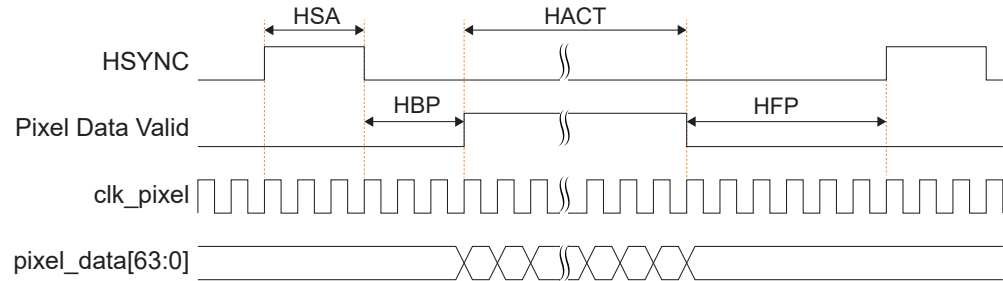
where:

- `PIX_CLK_MHZ` is the pixel clock in MHz
- `DATARATE_MBPS` is the MIPI data rate in Mbps
- `NUM_DATA_LANE` is the number of data lanes
- `PACK_BIT` is the Pixel data bits per pixel clock from **Video Mode Pixel Encoding** on page 14.

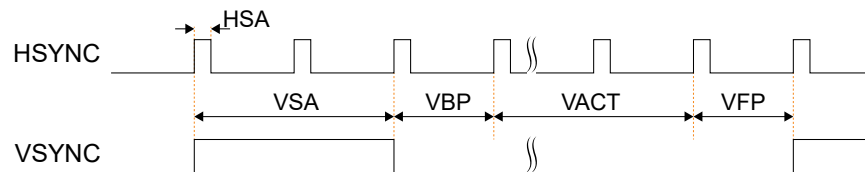
## Video Timing Parameters

The following waveforms show the video interface signals relationship.

**Figure 3: Video Timing Waveform (Horizontal)**



**Figure 4: Video Timing Waveform (Vertical)**



**Table 24: Video Timing Parameter Definitions**

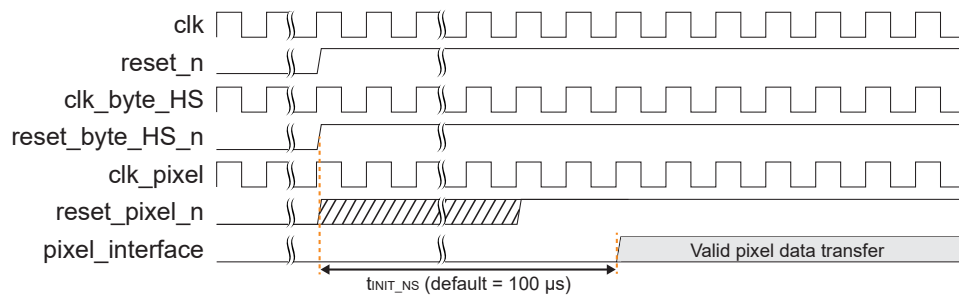
MIPI Video Timing Parameters	Definition
HACT	Total number of pixel per line. To ensure no flaws in the pixel-to-byte conversion, apply the following formula and the result must be a rounded number: $HACT \times \text{bits\_per\_pixel} / 64$
VACT	Total number of line per frame
HSA	HSYNC pulse width
HBP	Horizontal back porch
HFP	Horizontal front porch
VSA	VSNC pulse width
VBP	Vertical back porch
VFP	Vertical front porch
Pixel Clock	Video stream pixel clock frequency in MHz
MIPI Speed	DSI TX MIPI speed in Mbps
No. data lane	Number of MIPI data lane

## Reset Sequence and Initialization

During the initial power-up state, an initialization time,  $t_{INIT\_NS}$  of  $100\ \mu s$  is the minimum requirement for the MIPI D-PHY receiver to function properly before an LP/HS data transfer ( $100\ \mu s$  of initialization time is required when Efinix MIPI soft DPHY transmitter is used, you can adjust this timing depending on other transmitter's specifications). Typically, `reset_pixel_n` can be deasserted at any time after deassertion of other reset domains, but before the end of initialization process. The output of valid pixel data can be decoded from the PPI interface after the initialization process is completed. For any reset assertion event during user mode, you must follow the reinitialization sequence for reset deassertion procedure.

The following figure describes the reset sequencing and initialization requirements.

**Figure 5: Reset Sequence and Initialization**



# IP Manager

The Efinity® IP Manager is an interactive wizard that helps you customize and generate Efinity® IP cores. The IP Manager performs validation checks on the parameters you set to ensure that your selections are valid. When you generate the IP core, you can optionally generate an example design targeting an Efinity development board and/or a testbench. This wizard is helpful in situations in which you use several IP cores, multiple instances of an IP core with different parameters, or the same IP core for different projects.



**Note:** Not all Efinity IP cores include an example design or a testbench.

## Generating the MIPI DSI TX Controller Core with the IP Manager

The following steps explain how to customize an IP core with the IP Configuration wizard.

1. Open the IP Catalog.
2. Choose **MIPI > MIPI DSI TX Controller** core and click **Next**. The **IP Configuration** wizard opens.
3. Enter the module name in the **Module Name** box.



**Note:** You cannot generate the core without a module name.

4. Customize the IP core using the options shown in the wizard. For detailed information on the options, refer to the *Customizing the MIPI DSI TX Controller* section.
5. (Optional) In the **Deliverables** tab, specify whether to generate an IP core example design targeting an Efinity® development board and/or testbench. These options are turned on by default.
6. (Optional) In the **Summary** tab, review your selections.
7. Click **Generate** to generate the IP core and other selected deliverables.
8. In the **Review configuration generation** dialog box, click **Generate**. The Console in the **Summary** tab shows the generation status.



**Note:** You can disable the **Review configuration generation** dialog box by turning off the **Show Confirmation Box** option in the wizard.

9. When generation finishes, the wizard displays the **Generation Success** dialog box. Click **OK** to close the wizard.

The wizard adds the IP to your project and displays it under **IP** in the Project pane.

## Generated Files

The IP Manager generates these files and directories:

- **<module name>\_define.vh**—Contains the customized parameters.
- **<module name>\_tpl.v**—Verilog HDL instantiation template.
- **<module name>\_tpl.vhd**—VHDL instantiation template.
- **<module name>.v**—IP source code.
- **settings.json**—Configuration file.
- **<kit name>\_devkit**—Has generated RTL, example design, and Efinity® project targeting a specific development board.
- **Testbench**—Contains generated RTL and testbench files.

# Customizing the MIPI DSI TX Controller

The core has parameters so you can customize its function. You set the parameters in the **General** tab of the core's IP Configuration window.

*Table 25: MIPI DSI TX Controller Core Parameter*

Name	Option	Description
t <sub>LPX_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. The t <sub>LPX_NS</sub> ratio between host processor and peripheral must not exceed 3:2. The host processor is responsible for controlling its own clock frequency to match the peripheral. The host processor LP clock frequency must be in the range of 67% to 150% of peripheral LP clock frequency. Default: 50
t <sub>INIT_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 100000
t <sub>LP_EXIT_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 100
Bus Turnaround Timeout Period	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 100000
t <sub>D_TERM_EN_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. This parameter dedicates to the receiver timing from RX to TX during BTA function. Default: 35
t <sub>HS_PREPARE_ZERO_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns (value before adding UI). This parameter describes the expected timing (combination of t <sub>HS_PREPARE</sub> and t <sub>HS_ZERO</sub> ) driven by transmitter from RX to TX during BTA function. Default: 145
t <sub>CLK_ZERO_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 262
t <sub>CLK_TRAIL_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 60
t <sub>CLK_PRE_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 10
t <sub>CLK_POST_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 60
t <sub>CLK_PREPARE_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 38
t <sub>HS_PREPARE_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns (value before adding UI). Default: 40

Name	Option	Description
t <sub>WAKEUP_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 1000
t <sub>HS_EXIT_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns. Default: 100
t <sub>HS_ZERO_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns (value before adding UI). Default: 105
t <sub>HS_TRAIL_NS</sub>	Values according to MIPI D-PHY specifications.	Soft D-PHY timing parameter in ns (value before adding UI). Default: 60
NUM_DATA_LANE	1, 2, 4	Number of data lanes. Default: 4
MIPI Parallel Clock Frequency	10 - 187	MIPI parallel clock frequency in MHz. Round down the frequency value to an integer if there is any floating point from your calculation. Default: 125
IP Core Clock Frequency	40 - 100	IP core clock frequency in MHz. Default: 100
DPHY_CLOCK_MODE	Continuous, Discontinuous	DHY clock mode. Default: Continuous
Pack Type 60	Enable, Disable	Turn on pack 60-bit datatype. For example, RGB101010. Default: Disable
Pack Type 48	Enable, Disable	Turn on pack 48-bit datatype, for example, 20-bit YCbCr, 24-bit YCbCr, 12-bit YCbCr, RGB666 (24-bit), or RGB888. Default: Enable
Pack Type 64	Enable, Disable	Turn on pack 64-bit datatype, for example, 16-bit YCbCr, or RGB565. Default: Enable
Enable Bus Turnaround in vertical low power mode	Enable, Disable	Enables the bus turnaround during the last vertical front porch line which goes into LP-11 state. Default: Disable
Maximum Horizontal Resolution	Values according to video display	Maximum horizontal pixel resolution. Default: 1080
Video Transmission Packet Sequences	Non-burst mode with Sync Pulses, Burst mode, Non-burst mode with Sync Events	Select video mode: 0: Non-Burst Mode with Sync Pulses 1: Non-Burst Mode with Sync event (default) 2: Burst Mode
High Speed Write Data FIFO DEPTH	8 - 2048 <sup>(6)</sup>	HS command write data FIFO depth. Default: 64
Low power Write Data FIFO DEPTH	8 - 2048 <sup>(6)</sup>	LP command write data FIFO depth. Default: 64

<sup>(6)</sup> 2<sup>n</sup>, where n can be from 3 to 11.

Name	Option	Description
Low power Read Data FIFO DEPTH	8 - 2048 <sup>(6)</sup>	Bus turnaround read data depth. Default: 64
Pixel Data FIFO Depth Size	256 - 8192 <sup>(6)</sup>	FIFO depth size to store the pixel packet data. (need to be set to power of 2 value). Minimum FIFO depth required > horizontal_pixel (HACT) x bits_per_pixel / 64 Default: 2048
Enable End Of Transmission Packet	Enable, Disable	Enables or disables the End Of Transmission Packet. Default: Disable
Enable bidirectional DPHY	Enable, Disable	To instantiate a unidirectional or bidirectional soft D-PHY. Default: Enable
MIPI_DSI_TX_DEBUG	Enable, Disable	Enable debug ports for internal signal observation and monitoring. Default: Disable

# MIPI DSI TX Controller Example Design

You can choose to generate the example design when generating the core in the IP Manager Configuration window. Compile the example design project and download the **.hex** or **.bit** file to your board.

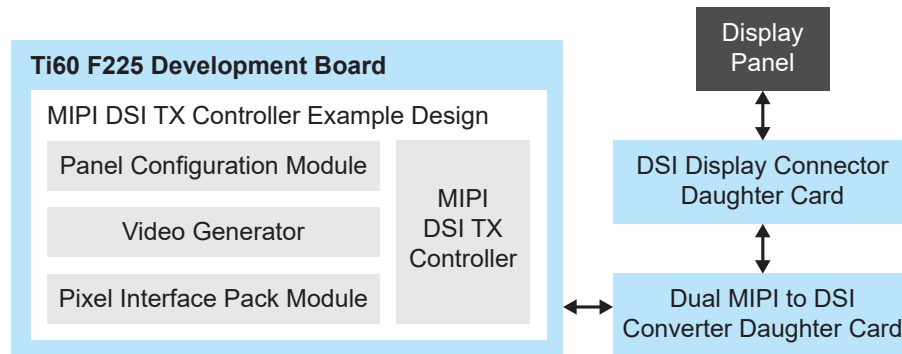


**Important:** Efinix tested the example design generated with the default parameter options only.

The example design targets the Titanium Ti60 F225 Development Board. This design generates a video stream and sends the video data to a display panel through the MIPI DSI TX Controller. Apart from the Titanium Ti60 F225 Development Board, the example design requires the following hardware:

- Dual MIPI to DSI Converter Daughter Card
- Mini-DSI Panel Connector Daughter Card
- Mini-DSI Panel display

*Figure 6: MIPI DSI TX Controller Core Example Design*



After power-up, press the reset button (SW7), then you should be able to see a video displayed on the panel module.

Figure 7: MIPI DSI TX Controller Example Design Design Set Up



Table 26: Example Design Implementation

FPGA	Logic and Adders	Flip-flops	Memory Blocks	$f_{MAX}$ (MHz) <sup>(7)</sup>					Efinity® Version <sup>(8)</sup>
				i_sys_clk	dsi_pclk	i_mipi_pclk	mipi_clk	i_mipi_tx_pclk	
Ti60 F225 C4	4,608	1,861	15	200	457	304	210	199	2021.2

<sup>(7)</sup> Using default parameter settings.

<sup>(8)</sup> Using Verilog HDL.

# MIPI DSI TX Controller Testbench

You can choose to generate the testbench when generating the core in the IP Manager Configuration window.



**Note:** You must include all `.v` files generated in the `/testbench` directory in your simulation.



**Important:** Efinix tested the testbench generated with the default parameter options only.

Efinix provides a simulation script for you to run the testbench quickly using the Modelsim software. To run the Modelsim testbench script, run `vsim -do modelsim.do` in a terminal application. You must have Modelsim installed on your computer to use this script.

**Table 27: Testbench Files**

The IP Manager generates different encrypted source codes for you to simulate with different simulators.

Directory	Note
<code>../Testbench</code>	Contains the example design and testbench files.
<code>../Testbench/aldec</code>	Contains the generated encrypted source code to simulate with the Aldec simulator.
<code>../Testbench/modelsim</code>	Contains the generated encrypted source code to simulate with the Modelsim simulator.
<code>../Testbench/ncsim</code>	Contains the generated encrypted source code to simulate with the NCSIM simulator.
<code>../Testbench/synopsys</code>	Contains the generated encrypted source code to simulate with the VCS simulator.

The simulation testbench simulates the example design. This design generates a video pattern to the MIPI DSI TX controller. The MIPI interface transaction from the controller is checked by a checker in the testbench.

After running the simulation successfully, the test prints the following message:

```
Received frame no. 1
Received frame no. 2
Received frame no. 3
Received frame no. 4
```

# Revision History

**Table 28: Revision History**

Date	Document Version	IP Version	Description
February 2026	3.1	5.15	Updated HACT in table Video Timing Parameters Definition. (DOC-2928)
May 2025	3.0	5.14	Updated Reset Sequence and Initialization. (DOC-2485) Update default parameter value. (SIP-910) Example design I/O bank update (HVIO 3.3 V). (SIP-907) Added description on debug port for ready_to_xmit and init_skewcal_done in table Debug Interface. Updated Customizing the MIPI DSI TX Controller.
December 2024	2.9	5.12	Updated mapping in <b>Table 16: Loosely Packed Pixel Stream, 20-bit YCbCr (2-pixels per clock)</b> on page 15 and <b>Table 21: RGB666 (2-pixels per clock)</b> on page 16. (DOC-2277)
December 2024	2.8	5.12	Added debug ports for internal signal observation and monitoring in Ports and Customizing the MIPI DSI TX Controller. (SIP-580) Added Pack Type 60 parameter from MIPI DSI TX Controller Core Parameter table. (SIP-803)
November 2024	2.7	5.11	Updated <b>Table 5: Video Interface</b> on page 6. (DOC-2109) Added Topaz in Features and Device Support. (DOC-2102) Added IP Version in Revision History. (DOC-2185)
September 2024	2.6	-	Updated pixel_data[63:0] in <b>Figure 3: Video Timing Waveform (Horizontal)</b> on page 17.
June 2024	2.5	-	Updated notes for lp_cmd_in_progress and hs_cmd_in_progress in table 0x24 Status Register Definition. (DOC-1949) Added Reset Sequence and Initialization sub-section.
April 2024	2.4	-	Updated figure 1.5 Gbps DSI TX Clocking Example by changing 700 MHz to 750 MHz from PLL to MIPI.
March 2024	2.3	-	Added important note in Testbench regarding using default parameters options only. (DOC-1781) Added testbench file for Aldec simulation model support. (DOC-1782)

Date	Document Version	IP Version	Description
January 2024	2.2	-	Updated video data at sections: Video Mode Pixel Encoding and MIPI Video Data DATA[63:0] Formats. (DOC-1624) Updated HS mode data width in Features section. (DOC-1675)
June 2023	2.1	-	Added MIPI Video Data Formats. (DOC-1233) Added Device Support and release notes sections. (DOC-1234) Updated supported data rate. (DOC-1217) Updated FIFO Pixel Depth Size parameter. Editorial changes.
April 2023	2.0	-	Updated MIPI Parallel Clock frequency parameter options and default value. (DOC-1186)
February 2023	1.9	-	Added note about the resource and performance values in the resource and utilization table are for guidance only. Corrected <b>Titanium MIPI Utility-v&lt;version&gt;.xslm</b> file name.
January 2023	1.8	-	Corrected DATARATE_MPBBS typo in Pixel Clock Calculation topic. Corrected Video Timing Waveform (Horizontal).
December 2022	1.7	-	Updated block diagram and added timing parameter waveforms. (DOC-1023)
August 2022	1.6	-	Added description to port tables about the clock domain used. (DOC-819) Corrected Control Status Registers definition. Corrected hsync and vsync video interface description. (DOC-879)
April 2022	1.5	-	Added minimum number of line for VSA, VBP, VFP, and VACT. (DOC-790)
March 2022	1.4	-	Updated description to HS and LP command queue register. (SIP-170)
January 2022	1.3	-	Updated resource utilization table. (DOC-700)
December 2021	1.2	-	Updated and added new IP Manager parameters. Update porch file to Titanium MIPI Utility. Updated register definition. Updated core block diagram. Added simulation testbench.

Date	Document Version	IP Version	Description
October 2021	1.1	-	<p>Added note to state that the <math>f_{MAX}</math> in Resource Utilization and Performance, and Example Design Implementation tables were based on default parameter settings.</p> <p>Updated design example target board to production Titanium Ti60 F225 Development Board and updated Resource Utilization and Performance, and Example Design Implementation tables. (DOC-553)</p> <p>Updated the display panel used in example design.</p> <p>Added 1 and 2 lanes support.</p> <p>Updated the MIPI TX I/O interface ports, Control Status Registers definition, Video Mode Configuration table, Video Mode Pixel Encoding table, and MIPI DSI TX Controller Core Parameters.</p>
July 2021	1.0	-	Initial release.