MIPI 2.5G CSI-2 RX Controller Core User Guide

UG-CORE-MIPI-2-5G-CSI2-RX-v1.7
October 2023
www.elitestek.com

Contents

Introduction

The MIPI CSI-2 interface, which defines a simple, high-speed protocol, is the most widely used camera interface for mobile[1]. Adding a MIPI interface to an FPGA creates a powerful bridge to transmit or receive high-speed video data easily to/from an application processor. The MIPI 2.5G CSI-2 RX Controller core allows you to perform complex video and image processing as a part of a complete system solution with a data rate of up to 2.5 Gbps. The MIPI 2.5G CSI-2 RX Controller uses the hard MIPI D-PHY blocks in supported 钛金系列 FPGAs.

Use the IP Manager to select IP, customize it, and generate files. The MIPI 2.5G CSI-2 RX 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 易灵思® development board.

Features

- Configurable data lanes: 1, 2, or 4
- High-speed (HS) mode and Low-power (LP) mode
- Arbitrary number of payload data bytes
- HS mode byte clock frequency from 5 to 156 MHz (80 to 2,500 Mbps data rate)
- Continuous HS mode byte clock and discontinuous HS mode byte clock
- 16-bit HS mode data width
- Pixel format:
  - RAW: RAW6, RAW7, RAW8, RAW10, RAW12,RAW14, RAW16, RAW20, RAW24, RAW28
  - RGB: RGB444, RGB555, RGB565, RGB888
  - YUV: YUV420 8-bit (legacy), YUV420 8-bit, YUV420, 10-bit, YUV420 8-bit (CSPS), YUV420 10-bit (CSPS), YUV422 8-bit, YUV422 10-bit
- User defined 8-bit data types
- Generic 8-bit long packet
- Null, blank and embedded 8-bit non image data
- PPI interface
- Generic frame mode and accurate frame mode
- Supports end of transmission error, start of transmission sync error, control error & LP escape error
- Supports control status register (CSR) for status and error assertion accessed through AXI4-Lite interface
- Supports initial auto-skew calibration and self-periodic skew calibration

Device Support

Table 1: MIPI 2.5G CSI-2 RX Controller Core Device Support

FPGA Family	Supported Device
Trion	–
钛金系列	Ti90, Ti120, Ti180 Not including F529 and G529 packages

[1] Source: MIPI Alliance.

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 change depending on the device resource utilization, design congestion, and user design.

Table 2: 钛金系列 Resource Utilization and Performance

MIPI 2.5G CSI-2 RX Controller with 4 data lanes.

FPGA	Logic and Adders	Flip-flops	Memory Blocks	DSP48 Blocks	f_{MAX} (MHz)[2]				Efinity® Version[3]
					clk	axi_clk	clk_byte_HS	clk_pixel	
Ti180 L484 C4	3,308	1,334	21	0	550	340	240	240	2022.1

Release Notes

You can refer to the IP Core Release Notes for more information about the IP core changes. The IP Core Release Notes is available in the Efinity Downloads page under each Efinity software release version.

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

[2] Using default parameter settings.
[3] Using Verilog HDL.

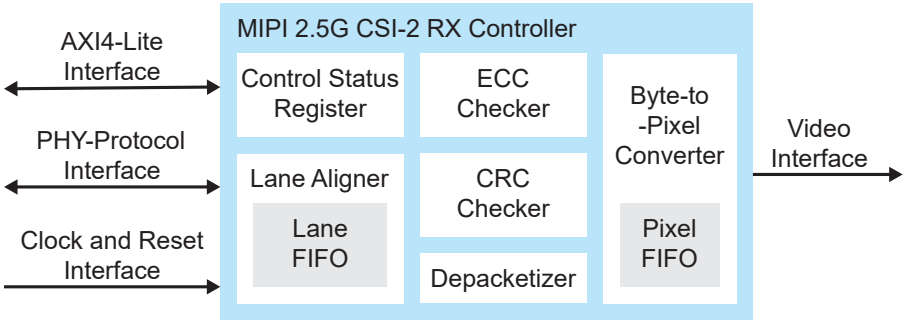
Functional Description

The MIPI 2.5G CSI-2 RX Controller consists of the following blocks:

- Control Status Registers—Registers that user can configure in runtime and statuses that user can read using the AXI4-lite interface
- ECC Checker—Error correction code checker block
- CRC Checker—Cyclic redundancy checker block
- Depacketizer—Converts MIPI data to pixel data format
- Lane Aligner—Aligns data lanes to each other
- Lane FIFO—Manages clock domain conversion for the data from MIPI byte clock domain to pixel clock domain
- Byte-to-Pixel Converter—Converts the 64-bit data from the packetizer to pixel data format and rearranges the data according to video data type

The core has a video, AXI4-lite, MIPI D-PHY, and clock and reset interfaces.

Figure 1: MIPI 2.5G CSI-2 RX Controller System Block Diagram



Ports

Table 3: Clock and Reset Ports

Port	Direction	Description
clk	Input	IP core clock consumed by controller logics. 100 MHz.
reset_n	Input	IP core reset signal.
clk_byte_HS	Input	MIPI RX parallel clock. This is a HS transmission clock.
reset_byte_HS_n	Input	MIPI RX 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 active low reset.

Table 4: PHY-Protocol Interface

Port	Direction	Description
RxUlpsClkNot	Input	Receive ULPS on Clock Lane. This bus is synchronized to axi_clk in the controller and stored in the CSR. This active-low signal is asserted to indicate that the clock lane module has entered the ULPS due to the detection of a request to enter the ULPS.
RxUlpsActiveClkNot	Input	ULPS (not) Active. This bus is synchronized to axi_clk in the controller and stored in the CSR. This active-low signal is asserted to indicate that the lane is in ULPS.
RxClkEsc [NUM_DATA_LANE-1:0]	Input	Escape mode Receive Clock. This escape clock is not used in the controller in the current core version. It is supplied by the MIPI hard D-PHY. This signal is used to transfer received data to the protocol during escape mode.
RxErrEsc [NUM_DATA_LANE-1:0]	Input	Escape Entry Error. This bus is synchronized to axi_clk in the controller and stored in the CSR. If an unrecognized escape entry command is received in LP mode, this active-high signal is asserted and remains asserted until the next transaction starts, so that the protocol can properly process the error.
RxErrControl [NUM_DATA_LANE-1:0]	Input	Control Error. This bus is synchronized to axi_clk in the controller and stored in the CSR. This active-high signal is asserted when an incorrect line state sequence is detected in LP and ALP modes. Once asserted, this signal remains asserted until the next transaction starts, so that the protocol can properly process the error.
RxErrSotSyncHS [NUM_DATA_LANE-1:0]	Input	Start-of-Transmission Synchronization Error. This bus is be synchronized to axi_clk in the controller and stored in the CSR. If the HS SoT leader sequence is corrupted in a way that proper synchronization cannot be expected, this active-high signal is asserted for one cycle of RxWordClkHS. When ErrSotSyncHS is asserted, RxSyncHS, ErrSotHS, andRxValidHS is not asserted.
RxUlpsEsc [NUM_DATA_LANE-1:0]	Input	Escape ULPS. This bus is synchronized to axi_clk in the controller and stored in the CSR. This active-high signal is asserted to indicate that the lane module has entered the ULPS, due to the detection of a received ULPS command.
RxUlpsActiveNot [NUM_DATA_LANE-1:0]	Input	ULPS (not) Active. This bus is synchronized to axi_clk in the controller and stored in the CSR. This active-low signal is asserted to indicate that the lane is in ULPS.
RxSkewCalHS [NUM_DATA_LANE-1:0]	Input	HS Receive Skew Calibration. This bus is clocked by clk_byte_HS. This optional active-high signal indicates that the high speed deskew burst is being received.
RxStopState [NUM_DATA_LANE-1:0]	Input	Data Lane in Stop State. This asynchronous active-high signal indicates that the MIPI D-PHY data lane is currently in stop state.
RxSyncHS [NUM_DATA_LANE-1:0]	Input	Receiver Synchronization Observed. This bus is clocked by clk_byte_HS. This active-high signal indicates that the Data lane has seen an appropriate synchronization event.
RxDataHSn [HS_DATA_WIDTH-1:0]	Input	HS data received by MIPI D-PHY data lane. This bus is clocked by clk_byte_HS. n = lane 0 to 3
RxValidHSn [HS_DATA_WIDTH/8-1:0]	Input	This active-high signal indicates that the MIPI D-PHY data lane is driving data to the protocol layer on the RxDataHS output. This bus is clocked by clk_byte_HS. RxValidHSn[0]: RxDataHS[7:0] contains valid data received from the channel RxValidHSn[1]: RxDataHS[15:8] contains valid data received from the channel n = lane 0 to 3

Table 5: AXI4-Lite Interface

Interface to access **Table 7: Control Status Registers** on page 8.
All signals are clocked with axi_clk and axi_reset_n.

Port	Direction	Description
axi_awaddr [15:0]	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 [15:0]	Input	AXI4-Lite read address bus.
axi_arvalid	Input	AXI4-Lite read address valid strobe.
axi_arready [31:0]	Output	AXI4-Lite read address ready signal.
axi_rdata	Output	AXI4-Lite read data.
axi_rvalid	Output	AXI4-Lite read data valid strobe.
axi_rready	Input	AXI4-Lite read data ready signal.

Table 6: Video Interface

All signals are clocked with clk_pixel and reset_pixel_n. The hsync_vc and vsync_vc are level signals and not pulse signals. See **Video Timing Parameters**.

Port	Direction	Description
hsync_vcx	Output	Active-high horizontal sync for virtual channel. x = virtual lane 0 to 15
vsync_vcx	Output	Active-high vertical sync for virtual channel. x = virtual lane 0 to 15
datatype [5:0]	Output	Data type of the long packet.
vc [1:0]	Output	Virtual channel.
pixel_data [63:0]	Output	Video Data. The actual data width of this port is dependent on pixel type. Refer to the pixel encoding table.
pixel_data_valid	Output	Active-high pixel data enable.
pixel_per_clk [15:0]	Output	Number of pixel per pixel clock.
word_count [15:0]	Output	Byte count of the long packet.
shortpkt_data_field [15:0]	Output	16-bit short packet data field for short packet.
vcx[1:0]	Output	Extra Virtual Channel

Pixel Clock Calculation

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

PIX_CLK_MHZ >= (DATARATE_MBPS * NUM_DATA_LANE) / 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 **Pixel Encoding** on page 11

Control Status Registers

Table 7: Control Status Registers

Word Address Offset	Name	R/W	Width (bits)
0x00	Interrupt Status Register	R/W1C[4]	15
0x04	Interrupt Enable Register	R/W	15
0x08	D-PHY status for lane 0	R	8
0x0C	D-PHY status for lane 1	R	8
0x10	D-PHY status for lane 2	R	8
0x14	D-PHY status for lane 3	R	8
0x18	D-PHY status for lane 4	R	8
0x1C	D-PHY status for lane 5	R	8
0x20	D-PHY status for lane 6	R	8
0x24	D-PHY status for lane 7	R	8
0x28	D-PHY status for clock lane	R	2

[4] Read register. Write 1 to clear the register.

Table 8: Interrupt Status Register Definition (0x00)

Bit	Name	Description
0	Pixel FIFO full	Pixel FIFO in the byte-to-pixel converter module is full.
1	Pixel FIFO empty	Pixel FIFO in the byte-to-pixel converter module is empty.
2	CRC error	CRC error indicator.
3	Ecc 1 bit error	ECC with 1 bit error indicator.
4	Ecc 2 bit error	ECC with 2 bit error indicator.
5	Undersize packet error	The incoming MIPI HS data byte is lesser than the wordcount value.
6	VC0 Line number synchronization error	Line number synchronization error for virtual channel 0.
7	VC1 Line number synchronization error	Line number synchronization error for virtual channel 1.
8	VC2 Line number synchronization error	Line number synchronization error for virtual channel 2.
9	VC3 Line number synchronization error	Line number synchronization error for virtual channel 3.
10	VC0 Frame number synchronization error	Frame number synchronization error for virtual channel 0.
11	VC1 Frame number synchronization error	Frame number synchronization error for virtual channel 1.
12	VC2 Frame number synchronization error	Frame number synchronization error for virtual channel 2.
13	VC3 Frame number synchronization error	Frame number synchronization error for virtual channel 3.
14	Initialization error	MIPI HS data is received before tInit is completed.

Table 9: Interrupt Enable Register Definition (0x04)

Each enabled interrupt status bit is aggregated to the `irq` output port as indicator. By default, all interrupt enable registers are set to 1'b0 (disabled).

Bit	Name	Description
0	Pixel FIFO full full interrupt enable	Enable interrupt generation for Pixel FIFO full status bit.
1	Pixel FIFO empty full interrupt enable	Enable interrupt generation for Pixel FIFO empty status bit.
2	CRC error full interrupt enable	Enable interrupt generation for CRC error status bit.
3	ECC 1 bit error full interrupt enable	Enable interrupt generation for ECC 1 bit error status bit.
4	ECC 2 bit error full interrupt enable	Enable interrupt generation for ECC 2 bit error status bit.
5	Undersize packet error full interrupt enable	Enable interrupt generation for Undersize packet error status bit.
6	VC0 Line number synchronization error full interrupt enable	Enable interrupt generation for VC0 Line number synchronization error status bit.
7	VC1 Line number synchronization error full interrupt enable	Enable interrupt generation for VC1 Line number synchronization error status bit.
8	VC2 Line number synchronization error full interrupt enable	Enable interrupt generation for VC2 Line number synchronization error status bit.
9	VC3 Line number synchronization error full interrupt enable	Enable interrupt generation for VC3 Line number synchronization error status bit.
10	VC0 Frame number synchronization error full interrupt enable	Enable interrupt generation for VC0 Frame number synchronization error status bit.
11	VC1 Frame number synchronization error full interrupt enable	Enable interrupt generation for VC1 Frame number synchronization error status bit.
12	VC2 Frame number synchronization error full interrupt enable	Enable interrupt generation for VC2 Frame number synchronization error status bit.
13	VC3 Frame number synchronization error full interrupt enable	Enable interrupt generation for VC3 Frame number synchronization error status bit.
14	Initialization error full interrupt enable	Enable interrupt generation for Initialization error status bit.

Table 10: D-PHY Status for Data Lanes Register Definition (0x08 – 0x24)

Bit	Name	Description
0	RxErrSotSyncHS	Start-of-Transmission (SoT) Synchronization Error. The core asserts this signal high for one cycle of RxWordClkHS if the HS SoT leader sequence is corrupted in a way that proper synchronization cannot be expected.
1	RxErrControl	Control Error. The core asserts this signal high when an incorrect Line state sequence is detected in LP and ALP modes. Once asserted, this signal remains asserted until the next transaction starts, so that the protocol can properly process the error.
2	RxErrEsc	Escape Entry Error. The core asserts this signal high if an unrecognized escape entry command is received in LP mode. Once asserted, this signal remains asserted until the next transaction starts, so that the protocol can properly process the error.
3	RxStopState	Lane is in stop state.
4	Reserved	Reserved
5	RxUlpsActiveNot	Ultra Low Power State (ULPS) (not) Active. The core deasserts this signal low to indicate that the data lane is in ULP state.
6	RxUlpsEsc	Escape ULPS (Receive) mode. The core asserts this signal high to indicate that the lane module has entered the ULPS, due to the detection of a received ULPS command. The lane module remains in this mode with RxUlpsEsc asserted until a Stop state is detected on the lane interconnect.
7	Reserved	Reserved

Table 11: D-PHY Status for Clock Lane Register Definition (0x28)

Bit	Name	Description
0	RxUlpsActiveClkNot	ULPS (not) Active. The core asserts this signal high to indicate that the clock lane is in ULPS.
1	RxUlpsClkNot	Receive ULPS on Clock Lane. The core deasserts this signal low to indicate that the clock lane module has entered the ULPS due to the detection of a request to enter the ULPS. The lane module remains in this mode with RxUlpsClkNot asserted until a stop state is detected on the lane interconnect.

Pixel Encoding

Table 12: Pixel Encoding

TYPE[5:0]	Data Type	Pixels per Clock	Bits per Pixel	Pixel Data Bits per Pixel Clock
0x20	RGB444	4	12	48
0x21	RGB555	4	15	60
0x22	RGB565	4	16	64
0x23	RGB666	3	18	54
0x24	RGB888	2	24	48
0x28	RAW6	8	6	48
0x29	RAW7	8	7	56
0x2A	RAW8	8	8	64
0x2B	RAW10	4	10	40
0x2C	RAW12	4	12	48
0x2D	RAW14	4	14	56
0x2E	RAW16	4	16	64
0x2F	RAW20	2	20	40
0x27	RAW24	2	24	48
0x26	RAW28	2	28	56
0x18	YUV420 8 bit	Odd line: 8 Even line: 4	Odd line: 8 Even line: 8, 24	Odd line: 64 Even line: 64
0x19	YUV420 10 bit	Odd line: 4 Even line: 2	Odd line: 10 Even line: 10, 30	Odd line: 40 Even line: 40
0x1A	Legacy YUV420 8 bit	4	8, 16	48
0x1C	YUV420 8 bit (CSPS)	Odd line: 8 Even line: 4	Odd line: 8 Even line: 8, 24	Odd line: 64 Even line: 64
0x1D	YUV420 10 bit (CSPS)	Odd line: 4 Even line: 2	Odd line: 10 Even line: 10, 30	Odd line: 40 Even line: 40
0x1E	YUV422 8 bit	4	8, 24	64
0x1F	YUV422 10 bit	2	10, 30	40
0x30 - 37	User defined 8 bit	8	8	64
0x13 – 0x16	Generic 8-bit long packet	8	8	64
0x12	Embedded 8-bit non image data	8	8	64

MIPI RX Video Data DATA[63:0] Formats

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

Table 13: RAW6 (8 Pixels per Clock)

63 48	47 42	41 36	35 30	29 24	23 18	17 12	11 6	5 0
0	Pixel 8	Pixel 7	Pixel 6	Pixel 5	Pixel 4	Pixel 3	Pixel 2	Pixel 1

Table 14: RAW7 (8 Pixels per Clock)

63 56	55 49	48 42	41 35	34 28	27 21	20 14	13 7	6 0
0	Pixel 8	Pixel 7	Pixel 6	Pixel 5	Pixel 4	Pixel 3	Pixel 2	Pixel 1

Table 15: RAW8 (8 Pixels per Clock)

63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
Pixel 8	Pixel 7	Pixel 6	Pixel 5	Pixel 4	Pixel 3	Pixel 2	Pixel 1

Table 16: RAW10 (4 Pixels per Clock)

63 40	39 30	29 20	19 10	9 0
0	Pixel 4	Pixel 3	Pixel 2	Pixel 1

Table 17: RAW12 (4 Pixels per Clock)

63 48	47 36	35 24	23 12	11 0
0	Pixel 4	Pixel 3	Pixel 2	Pixel 1

Table 18: RAW14 (4 Pixels per Clock)

63 56	55 42	41 28	27 14	13 0
0	Pixel 4	Pixel 3	Pixel 2	Pixel 1

Table 19: RAW16 (4 Pixels per Clock)

63 48	47 32	31 16	15 0
Pixel 4	Pixel 3	Pixel 2	Pixel 1

Table 20: RAW20 (2 Pixels per Clock)

63 40	39 20	19 0
0	Pixel 2	Pixel 1

Table 21: RAW24 (2 Pixels per Clock)

63 48	47 24	23 0
0	Pixel 2	Pixel 1

Table 22: RAW28 (2 Pixels per Clock)

63 56	55 28	27 0
0	Pixel 2	Pixel 1

Table 23: RGB444 (4 Pixels per Clock)

63 48	47 36	35 24	23 12	11 0
0	Pixel 4	Pixel 3	Pixel 2	Pixel 1

63 48	47 36			35 24			23 12			11 0		
—	[47:44] Red	[43:40] Green	[39:36] Blue	[35:32] Red	[31:28] Green	[27:24] Blue	[23:20] Red	[19:16] Green	[15:12] Blue	[11:8] Red	[7:4] Green	[3:0] Blue

Table 24: RGB555 (4 Pixels per Clock)

63 60	59 45			44 30			29 15			14 0		
0	Pixel 4			Pixel 3			Pixel 2			Pixel 1		
—	[59:55] Red	[54:50] Green	[49:45] Blue	[44:40] Red	[39:35] Green	[34:30] Blue	[29:25] Red	[24:20] Green	[19:15] Blue	[14:10] Red	[9:5] Green	[4:0] Blue

Table 25: RGB565 (4 Pixels per Clock)

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

Table 26: RGB888 (2 Pixels per Clock)

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

Table 27: YUV420 8 bit Odd Line (8 Pixels per Clock), Even Line (4 Pixels per Clock)

63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
Odd Lines							
Pixel 8 Y8	Pixel 7 Y7	Pixel 6 Y6	Pixel 5 Y5	Pixel 4 Y4	Pixel 3 Y3	Pixel 2 Y2	Pixel 1 Y1
Even Lines							
Pixel 4	Pixel 3			Pixel 2	Pixel 1		
Y4	V3	Y3	U3	Y2	V1	Y1	U1

Table 28: Legacy YUV420 8 bit (4 Pixels per Clock)

63 48	47 40	39 32	31 24	23 16	15 8	7 0
0	Pixel 4	Pixel 3		Pixel 2	Pixel 1	
Odd Lines	Y4	Y3	U3	Y2	Y1	U1
Even Lines	Y4	Y3	V3	Y2	Y1	V1

Table 29: YUV420 10 bit Odd Line (4 Pixels per Clock), Even Line (2 Pixels per Clock)

63 40	39 30	29 20	19 10	9 0
Odd Lines				
0	Pixel 4 Y4	Pixel 3 Y3	Pixel 2 Y2	Pixel 1 Y1
Even Lines				
0	Pixel 1	Pixel 2	Pixel 1	
	Y2	V1	Y1	U1

Table 30: YUV422 8 bit (4 Pixels per Clock)

63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
Pixel 4	Pixel 3			Pixel 2	Pixel 1		
Y4	V3	Y3	U3	Y2	V1	Y1	U1

Table 31: YUV422 10 bit (2 Pixels per Clock)

63 40	39 30	29 20	19 10	9 0
0	Pixel 1	Pixel 2	Pixel 1	
	Y2	V1	Y1	U1

Video Timing Parameters

The following waveforms show the video interface signals relationship.

Figure 2: Video Timing Waveform (Horizontal)

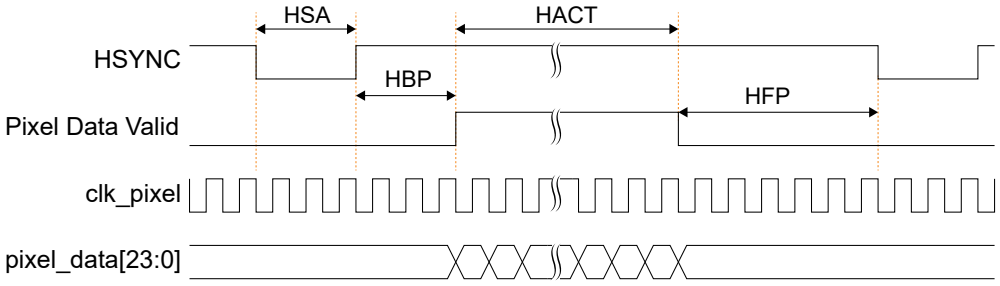


Figure 3: Video Timing Waveform (Vertical)

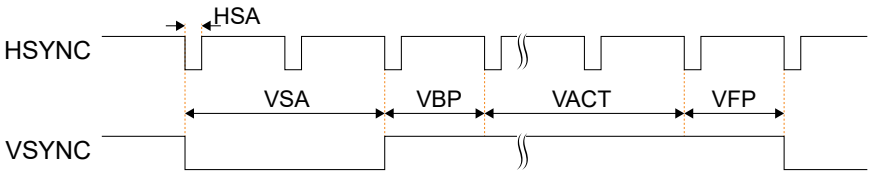


Table 32: Video Timing Parameter Definitions

MIPI Video Timing Parameters	Definition
HACT	Total number of pixel per line
VACT	Total number of line per frame
HSA	HSYNC pulse width
HBP	Horizontal back porch
HFP	Horizontal front porch
VSA	VSYNC pulse width
VBP	Vertical back porch
VFP	Vertical front porch
Pixel Clock	Video stream pixel clock frequency in MHz
MIPI Speed	CSI-2 RX MIPI speed in Mbps
No. data lane	Number of MIPI data lane

IP Manager

The Efinity® IP Manager is an interactive wizard that helps you customize and generate 易灵思® 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 易灵思 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 易灵思 IP cores include an example design or a testbench.

Generating the MIPI 2.5G CSI-2 RX 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 2.5G CSI-2 RX 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 2.5G CSI-2 RX Controller section.
5. (Optional) In the **Deliverables** tab, specify whether to generate an IP core example design targeting an 易灵思® 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>_tmpl.v**—Verilog HDL instantiation template.
- **<module name>_tmpl.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 2.5G CSI-2 RX 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 33: MIPI 2.5G CSI-2 RX Controller Core Parameter

Name	Options	Description
Data Lanes	1, 2, 4	Number of data lanes. Default: 4
IP Core Clock Frequency	40 - 100	IP core clock frequency in MHz. Default: 100
Pixel Data FIFO Depth Size	256 - 8192	FIFO depth size that stores the pixel packet data. Set to the power of 2 value that is bigger than the 2 * (max horizontal pixel / 8). For example, when maximum horizontal pixel is 1280, set this parameter to 512. Default: 1024
Image Frame Mode	Generic, Accurate	Selects image frame mode. Generic mode: Frame format without accurate synchronization timing via Line Start and Line End. Accurate mode: Frame format with accurate synchronization timing via Line Start and Line End. Default: Generic
Enable Extra Bit on Virtual Channel	Disable, Enable	Enables 16 virtual channel support. Default: Disable
Enable Pipeline State for RXStopState Signal	8 – 15	Enable pipeline stage for RXStopState signal. The pipeline registers are clocked with HS mode byte clock. Compensates the MIPI HSIO deserializer, read FIFO and data synchronizer latency in designs with low MIPI data rate. Default: 8
Number of Asynchronous Register Stages	2 – 8	Cross clock domain control signal synchronization stage. Default: 2
tINIT_NS	Values according to MIPI D-PHY specifications.	PHY initialization period in ns. Value must be 100000 or more. Default: 100000
Pack Type 40	Enable, Disable	Enables the controller to pack RAW10, RAW20, YUV_420_10, and YUV_422_10 data type.(5) Default: Enable
Pack Type 48	Enable, Disable	Enables the controller to pack RAW6, RAW12, RAW24, RGB888, and YUV_420_8_legacy data type.(5) Default: Enable
Pack Type 56	Enable, Disable	Enables the controller to pack RAW7 RAW14, and RAW28.(5) Default: Enable
Pack Type 64	Enable, Disable	Enables the controller to pack RAW8, RAW16, RGB444, RGB565, RGB555, YUV_422_8, YUV_420_8, generic long packet, user define 8-bit, and embedded 8-bit non image packet.(5) Default: Enable
PPI Interface Data Width	16	HS mode data width. Default: 16

(5) Only enable the pack type that you are using to save logic resources.

MIPI 2.5G CSI-2 RX Controller Example Design

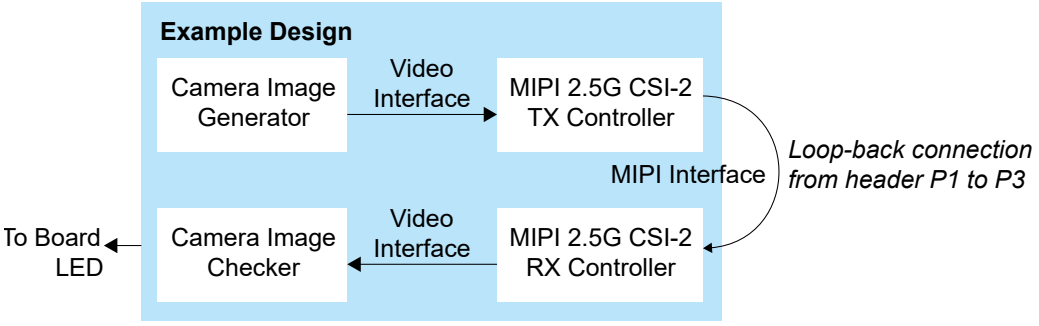
You can choose to generate the example design when generating the core in the IP Manager Configuration window. Compile the example design project and download the **.hex** or **.bit** file to your board.

Important: 易灵思 tested the example design generated with the default parameter options only.

The example design targets the 钛金系列 Ti180 M484 Development Board. The design instantiates both MIPI 2.5G CSI-2 TX and RX Controller cores. This design requires a QTE header-compatible cable.

The design generates an image and sends the image data to the camera image checker through the MIPI 2.5G CSI-2 TX Controller. The data is then sent through a hardware loopback on the board using a 4-lane MIPI interface to the MIPI 2.5G CSI-2 RX Controller. The camera image checker compares the data received with the one created by the image generator, and outputs the results using the board LEDs.

Figure 4: MIPI 2.5G CSI-2 RX Controller Core Example Design



After programming the bitstream file into the development board, press pushbutton SW3 to reset the design. Then press pushbutton SW4 to restart the pixel data generator. LED5 blinks continuously and LED4 turns on if the received image matches the generated image.

Note: You can use the **钛金系列 MIPI Utility-v<version>.xlsm** to check if your own design will work. Enter the related design information then verify whether your selections pass various tests. You can download the 钛金系列 MIPI utility from the Design Support page in the Support Center.

Table 34: Example Design Implementation

FPGA	Logic and Adders	Flip-flops	Memory Blocks	DSP48 Blocks	f_{MAX} (MHz)[6]				Efinity® Version[7]
					clk1	clk2	clk3	clk4	
Ti180 M484 C4	4,175	2,626	161	0	500	220	300	160	2022.1

- clk1—mipi_clk
- clk2—mipi_dphy_rx_clk_CLKOUT
- clk3—clk_pixel
- clk4—mipi_dphy_tx_SLOWCLK

[6] Using default parameter settings.
[7] Using Verilog HDL.

Revision History

Table 35: Revision History

Date	Version	Description
October 2023	1.7	Updated MIPI video data format tables to include RGB information. (DOC-1474)
September 2023	1.6	Corrected supported HS data width. (DOC-1437)
July 2023	1.5	Added more description for Accurate and Generic image frame modes. (DOC-1343)
June 2023	1.4	Corrected signal width and updated the description for RxValidHSn [HS_DATA_WIDTH/8-1:0]. (DOC-1340) Added Device Support and release notes sections. (DOC-1234) Updated supported data rate. (DOC-1217) Updated port descriptions. Added RAW16, RAW20, RAW24, and RAW28 format support. Updated IP Core Frequency, Pixel Data FIFO Depth Size, Pack Type40, Pack Type48, Pack Type56, Pack Type64 parameters. Improved Interrupt Enable Register Definition descriptions. Editorial changes.
February 2023	1.3	Added note about the resource and performance values in the resource and utilization table are for guidance only.
December 2022	1.2	Updated PHY-Protocol Interface descriptions by indicating the clock domains. (DOC-1022) Added New in Version section.
August 2022	1.1	Updated parameters for IP Manager support in Efinity software v2022.1.
August 2022	1.0	Initial release.