京微齐力
HERCULES MICROELECTRONICS

HME MIPI CSI-2 Controller IP User Guide

Sept. 2026

Hercules Microelectronics Co., Ltd.

Notes

Copyright © 2026 Hercules Microelectronics Co., Ltd. All rights reserved.

No part of this document may be copied, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise, without the written permission of Hercules Microelectronics Co., Ltd. All trademarks are the property of their respective companies.

Version Part Number

V1.0

Contact Us

If you have any problems or requirements during using our product, please contact

Hercules Microelectronics Co., Ltd., or **your local distributors,** or send e-mail to

sales@hercules-micro.com

Environmental Considerations

To avoid the harmful substances being released into the environment or harming human health, we encourage you to recycle this product in an appropriate way to make sure that most of the materials are reused or recycled appropriately. Please contact your local authorities for disposal or recycle information.

Warranty

The information in this document has been carefully checked and is believed to be entirely reliable. However, no responsibility is assumed for inaccuracies. Furthermore, Hercules Microelectronics Co., Ltd. reserves the right to discontinue or make changes, without prior notice, to any products herein to improve reliability, function, or design.

Hercules Microelectronics Co., Ltd. advises its customers to obtain the latest version of the relevant information to verify, before placing orders, that the information being relied upon is current.

The product introduced in this book is not authorized for use as critical components in life support devices or systems without the express written approval of Hercules Microelectronics Co., Ltd. As used herein: 1. Life support devices or systems are devices or systems that (a) are intended for surgical implant into the body or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user. 2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Revision History

The table below shows the revision history for this document.

Date	Version	Revision
2026-09-20	V1.0	Initial release.

Table of Contents

Notes **1**

Revision History **2**

Table of Contents **3**

Figure Contents 4
Table Contents 5

Before You Start **6**

About this Datasheet 6
Legends 6

1 Introduction **7**

1.1 Features 7
1.2 Supported Devices 7
1.3 Internal Block Diagram 8

2 Port and Register Description **9**

2.1 Port Description 9
2.2 MIPI CSI-2 Parameter Description 11
2.3 Register List 11
2.4 Register Definition 11

3 IP Core Usage **12**

3.1 Clock Reset Design 12
3.2 MIPI CSI-2 Protocol 13
3.3 MIPI CSI-2 Interface timing 14
3.4 IP Constraint Method 17
3.4.1 Example of Clock Declaration 17
3.4.2 Cross-Clock Domain Processing 18
3.5 IP Generation of IP Core 18
3.5.1 IP Wizard Parameter Design 18
3.6 Resource and Performance Report 21
3.6.1 System Design Block Diagram 21
3.6.2 Resource Report 21
3.6.3 Performance Report 21
3.7 Usage Notes 21

4 Generate File Directory Structure **22**

Figure Contents

Figure 1-1 MIPI CSI-2 TX Controller System Block Diagram 8
Figure 1-2 MIPI CSI-2 RX Controller System Block Diagram 8
Figure 3-1 Reset Structure Block Diagram 12
Figure 3-2 Clock Structure Block Diagram 13
Figure 3-3 TX PPI Interface Timing (1-Lane Mode) 14
Figure 3-4 TX PPI Interface Timing (2-Lane Mode) 14
Figure 3-5 TX PPI Interface Timing (4-Lane Mode) 15
Figure 3-6 TX Long Packet Transmission Timing 15
Figure 3-7 TX Short Packet Transmission Timing 15
Figure 3-8 RX PPI Interface Timing (1-Lane Mode) 16
Figure 3-9 RX PPI Interface Timing (2-Lane Mode) 16
Figure 3-10 RX PPI Interface Timing (4-Lane Mode) 16
Figure 3-11 RX Short Packet Reception Timing 17
Figure 3-12 RX Long Packet Reception Timing 17
Figure 3-13 Page 1 of RX IP Wizard (Input Module Name) 19
Figure 3-14 Page 1 of TX IP Wizard (Input Module Name) 19
Figure 3-15 RX Parameter Page (Number of lanes = 4) 20
Figure 3-16 TX Parameter Page (Number of lanes = 4) 20
Figure 3-17 Reference Routine System Design Block Diagram 21

Table Contents

Table 1-1 Supported Devices 7
Table 2-1 MIPI CSI-2 TX Controller Ports 9
Table 2-2 MIPI CSI-2 RX Controller Ports 9
Table 3-1 Data Type Encoding 13
Table 4-1 File Structure 22

Before You Start

About this Datasheet

This document describes the architecture, interfaces, timing, and usage of the MIPI CSI-2 Controller IP core, and provides the design procedure, reference design, and file structure to help you integrate the IP into your own design.

For more details, please go to http://www.hercules-micro.com.

If you have any questions or suggestions, please contact us at support@hercules-micro.com.

Legends

Abbreviation	Full Name	Abbreviation	Full Name
CRC	**C**yclic **R**edundancy **C**heck	**LP**	**L**ow-**P**ower
CSI	**C**amera **S**erial **I**nterface	**LUT**	**L**ook-**U**p **T**able
DI	**D**ata **I**dentifier	**PF**	**P**acket **F**ooter
ECC	**E**rror **C**orrecting **C**ode	**PHY**	**Phy**sical Layer
EMB	**E**mbedded **M**emory **B**lock	**PPI**	**P**ort **P**hysical **I**nterface
FIFO	**F**irst **I**n **F**irst **O**ut	**RX**	**R**eceiver
FPGA	**F**ield **P**rogrammable **G**ate **A**rray	**TX**	**T**ransmitter
HS	**H**igh-**S**peed	**VC**	**V**irtual **C**hannel
IP	**I**ntellectual **P**roperty	**WC**	**W**ord **C**ount

1 Introduction

The MIPI CSI-2 Controller IP core is a soft-core solution that implements the MIPI CSI-2 protocol layer (packing and unpacking pixels to and from bytes), designed for high-speed serial data transmission between high-bandwidth image sensors and processors. The transmit controller packs application-layer image data into MIPI CSI-2 packets and sends them; the receive controller parses image data out of the serial data stream. Both interact with the D-PHY physical layer through a standard PPI interface.

1.1 Features

- Implements the MIPI CSI-2 protocol layer, supporting encapsulation (transmission) and parsing (reception) of long and short packets
- Interface types:
  - Physical layer: Standard PPI (PHY Protocol Interface) for D-PHY
  - Application side: Custom command and payload parallel interface with 32-bit payload data bus
  - Protocol layer: MIPI CSI-2 protocol layer (long/short packets, ECC, CRC16-CCITT)
- Configurable number of data lanes; supports 1, 2, and 4 lanes
- Transmitter automatically generates the packet header with ECC (corrects 1 error, detects 2 errors), and appends CRC16 checksum to payload data
- Receiver automatically performs header detection, command extraction, and payload restoration, enabling safe cross-clock domain data exchange
- Error indication output for header ECC errors and payload CRC errors
- Supports data formats (protocols): RAW (6/7/8/10/12/14/16/20), RGB (565/555/444/666/888), YUV (YUV420/422), and user-defined formats, extendable per customer requirements
- The operating frequency can reach up to 235 MHz
- Resource usage reference:
  - TX Controller (1 / 2 / 4 Lane): 522/505/478 LUTS, 728/726/724 REG
  - RX Controller (1 / 2 / 4 Lane): 105/130/191 LUTS, 186/221/302 REG

1.2 Supported Devices

Table 1-1 Supported Devices

FPGA Family	Description
P2 series	Supported
H7 series	Supported

1.3 Internal Block Diagram

The transmit controller packs application-layer commands and payload data into MIPI CSI-2 protocol packets and sends them to the D-PHY through the PPI interface. The application layer issues a packet request with VC / data type / byte count, and the controller automatically generates the ECC-protected header, computes CRC16, writes the TX buffer, and distributes bytes to the PPI interface per lane count. The command-acknowledge signal and the last-payload-word marker are both single-cycle pulses, while the payload-transfer enable stays asserted for the whole payload transfer.

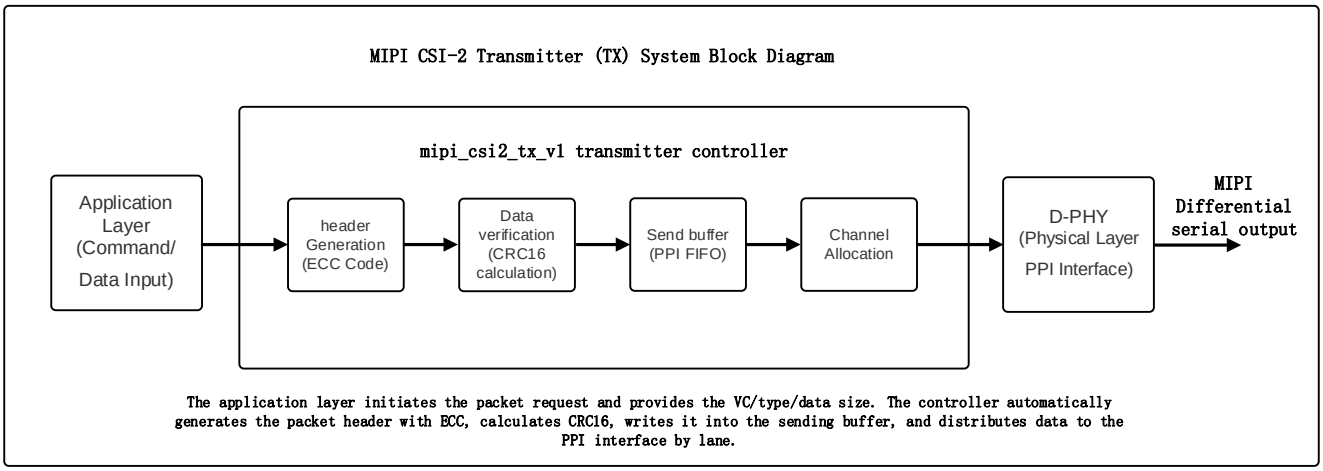

Figure 1-1 MIPI CSI-2 TX Controller System Block Diagram

The receive controller receives the data stream from the D-PHY PPI interface, parses the MIPI CSI-2 packet format, and extracts the command and payload data. It performs ECC on the packet header (correct 1, detect 2 bits) and CRC16 on the long-packet payload, restoring the byte stream into 32-bit parallel data output to the application layer, with check error flags output alongside.

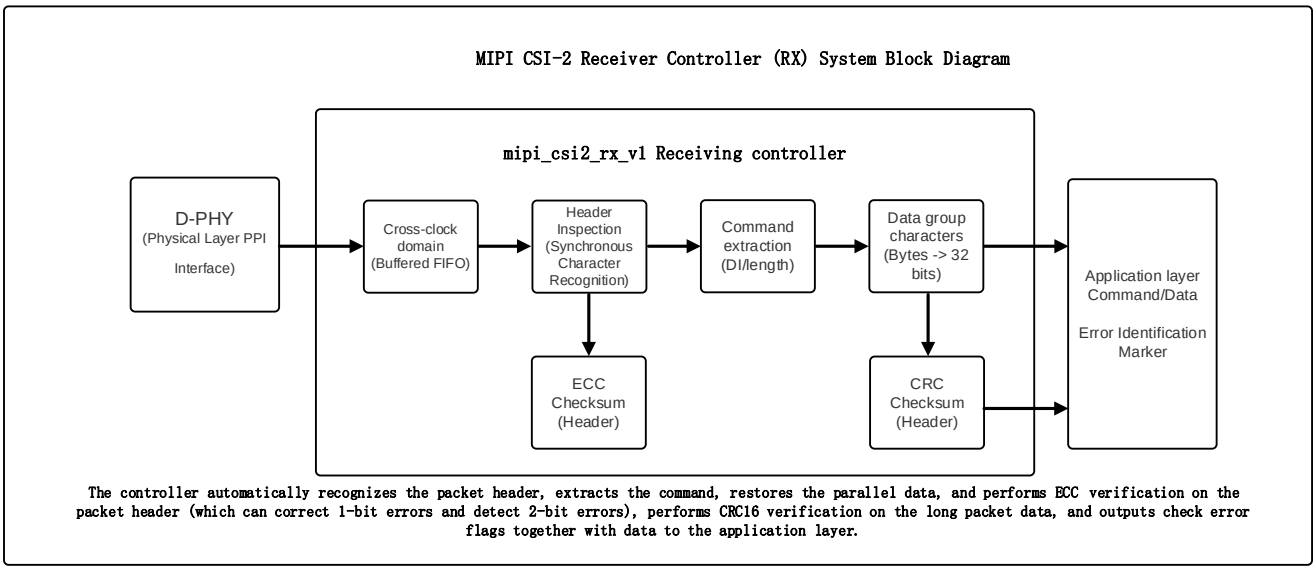

Figure 1-2 MIPI CSI-2 RX Controller System Block Diagram

2 Port and Register Description

2.1 Port Description

Table 2-1 MIPI CSI-2 TX Controller Ports

Port Name	Type	Width	Description
reset_n	input	1	Global reset, active low
PPI (PHY Protocol Interface)			
TxByteClkHS	input	1	PPI byte clock, the D-PHY HS-mode clock
TxReadyHS_ln0	input	1	PPI Lane0 is preparing the signal, indicating that the D-PHY can receive data
TxRequestHS_ln0	output	1	PPI Lane0 Request Signal, indicating that the controller has data to be sent
TxDataHS_ln0[7:0]	output	8	PPI Lane0 transmits the data bus
TxReadyHS_ln1	input	1	PPI Lane1 Preparation Signal
TxRequestHS_ln1	output	1	PPI Lane1 Request Signal
TxDataHS_ln1[7:0]	output	8	PPI Lane1 transmits the data bus
TxReadyHS_ln2	input	1	PPI Lane2 Preparation Signal
TxRequestHS_ln2	output	1	PPI Lane2 Request Signal
TxDataHS_ln2[7:0]	output	8	PPI Lane2 transmits the data bus
TxReadyHS_ln3	input	1	PPI Lane3 Preparation Signal
TxRequestHS_ln3	output	1	PPI Lane3 Request signal
TxDataHS_ln3[7:0]	output	8	PPI Lane3 transmits the data bus
Custom commands and parallel interface for load handling			
tx_cmd_req	input	1	The application layer sends a request signal, and when it is in a high state, it initiates the MIPI CSI-2 packet transmission.
tx_cmd_vc[1:0]	input	2	Virtual channel ID, embedded in the DI field of the packet header
tx_cmd_data_type[5:0]	input	6	MIPI CSI-2 data type code, embedded in the header DI field
tx_cmd_byte_count[15:0]	input	16	Packet payload byte count
tx_payload[31:0]	input	32	Input of application layer load data, 32-bit wide
tx_cmd_ack	output	1	The packet sends a confirmation signal, indicating that the current packet has been received.
tx_payload_en	output	1	Load transmission enable signal, indicating that the controller is ready to receive the load
tx_payload_en_last	output	1	The last load data marker
csi_tx_esc_clk	input	1	Reserved clock input port (ESC mode)

Table 2-2 MIPI CSI-2 RX Controller Ports

Port Name	Type	Width	Description
reset_n	input	1	Global reset, active low

Port Name	Type	Width	Description
PPI (PHY Protocol Interface)			
RxByteClkHS	input	1	PPI byte clock, derived from the HS clock of D-PHY
csi_rx_ctrl_clk	input	1	Application layer clock, serving as the read clock for cross-clock-domain FIFO
RxActiveHS_ln0	input	1	PPI Lane0 HS activates the signal. 1 indicates active, 0 indicates idle.
RxSyncHS_ln0	input	1	PPI Lane0 Synchronous signal, indicating that the current byte is a synchronous sequence
RxValidHS_ln0	input	1	PPI Lane0 valid data signal
RxDataHS_ln0[7:0]	input	8	PPI Lane 0 Data Bus
RxActiveHS_ln1	input	1	PPI Lane1 HS activates the signal. 1 indicates active, 0 indicates idle.
RxSyncHS_ln1	input	1	PPI Lane1 synchronization signal, indicating that the current byte is a synchronization sequence
RxValidHS_ln1	input	1	PPI Lane1 valid data signal
RxDataHS_ln1[7:0]	input	8	PPI Lane 1 Data Bus
RxActiveHS_ln2	input	1	PPI Lane2 HS activates the signal. 1 indicates active, 0 indicates idle.
RxSyncHS_ln2	input	1	PPI Lane2 synchronization signal, indicating that the current byte is a synchronization sequence
RxValidHS_ln2	input	1	PPI Lane2 valid data signal
RxDataHS_ln2[7:0]	input	8	PPI Lane 2 Data Bus
RxActiveHS_ln3	input	1	PPI Lane3 HS activates the signal. 1 indicates active, 0 indicates idle.
RxSyncHS_ln3	input	1	PPI Lane3 synchronization signal, indicating that the current byte is a synchronization sequence
RxValidHS_ln3	input	1	PPI Lane3 valid data signal
RxDataHS_ln3[7:0]	input	8	PPI Lane 3 Data Bus
Custom commands and parallel interface for load handling			
rx_cmd_valid	output	1	Packet-header valid signal, indicating the start of a new MIPI CSI-2 packet
rx_cmd[23:0]	output	24	Packet-header data: [7:0]=DI (VC[7:6]+DataType[5:0]), [15:8]=WordCount_LSB, [23:16]=WordCount_MSB
rx_cmd_ecc[7:0]	output	8	ECC check byte is output only in 4-Lane mode (separately output when the packet header and ECC arrive simultaneously); in 1/2-Lane mode, this port is always 0.
rx_payload[31:0]	output	32	32-bit-wide payload data output
rx_payload_valid	output	1	Valid signal of load data
rx_payload_valid_last	output	1	Marker for the last 32-bit payload word of the packet
csi_tx_esc_clk	input	1	Reserved clock input port (ESC mode)
Verification and debugging interface (ECC/CRC error identification)			

Port Name	Type	Width	Description
ecc_one_bit_err	output	1	ECC 1-bit error indication: The packet header ECC check is set to 1 when a correctable 1-bit error is detected (output after hardware automatic correction), and the error position is given by ecc_one_bit_err_pos.
ecc_two_bit_err	output	1	ECC 2-bit error indication: The packet header ECC check is set to 1 when it detects an irreparable 2-bit error.
ecc_one_bit_err_pos[4:0]	output	5	ECC 1-bit error position: When "ecc_one_bit_err" is set, it is effective. It points to the index of the erroneous bit within the header (24-bit data + 8-bit ECC, totaling 32 bits). The index ranges from 0 to 31.
ecc_err	output	1	ECC Error General Indicator: Set when any ECC check error (1-bit or 2-bit) occurs
ecc_err_pos[4:0]	output	5	ECC error location: Valid when ecc_err is set. Indicates the index position of the error bit within the header, ranging from 0 to 31.
crc_err	output	1	CRC error flag: Set when the CRC16-CCITT check result of the long packet payload is inconsistent with the received CRC field.

2.2 MIPI CSI-2 Parameter Description

Table 2-1 MIPI CSI-2 parametric description

Parameter	Type	Range	Default	Description
Number for lanes	decimal	1,2,4	1	The number of MIPI CSI-2 data channels (Lanes) is configured on the second page of the Fuxi IP Wizard. Both the TX and RX controllers support 1/2/4. After generation, it corresponds to the RTL parameter NUM_LANES. In the 4-Lane mode, the RX additionally outputs the header ECC check byte rx_cmd_ecc[7:0].

2.3 Register List

This IP is a pure RTL implementation with no register configuration interface; all functionality is determined by port signals and timing. Configurable options (number of lanes) are set through the IP Wizard graphical user interface.

2.4 Register Definition

This IP is a pure RTL implementation with no register configuration interface; all functionality is determined by port signals and timing. Configurable options (number of lanes) are set through the IP Wizard graphical user interface.

3 IP Core Usage

3.1 Clock Reset Design

This design employs a single-PLL internal clocking scheme, generating all required system clocks using an on-chip RC oscillator combined with the Fuxi PLL IP. External PLL input mode switching is not supported.

- Clock generation chain: oscillator_v1 (on-chip RC oscillator) → pll_v1 → 5 system clock outputs
- PLL input: output from oscillator_v1

PLL outputs:

- clkout0 esc_clk: D-PHY TX escape channel clock, 0° reference
- clkout1 esc_clk_rx: D-PHY RX escape channel clock
- clkout2 TxBitClk: TX bit clock, 0° phase
- clkout3 TxBitClk_90: TX bit clock, 90° phase shift (configured via P3SEL=3'b100)
- clkout4 TxByteClk: TX byte clock, also serving as MCU cpuclk and clk_periph

The transmit side is driven by a unified PPI byte clock provided by D-PHY; the receive side operates with two asynchronous clock domains for write and read operations, and data exchange is handled via an internal cross-clock-domain FIFO. Reset uses a low-active main reset signal; asynchronous reset with synchronous release is recommended, and the relevant clocks must be stable before releasing the reset. Below is an example of clock and reset design:

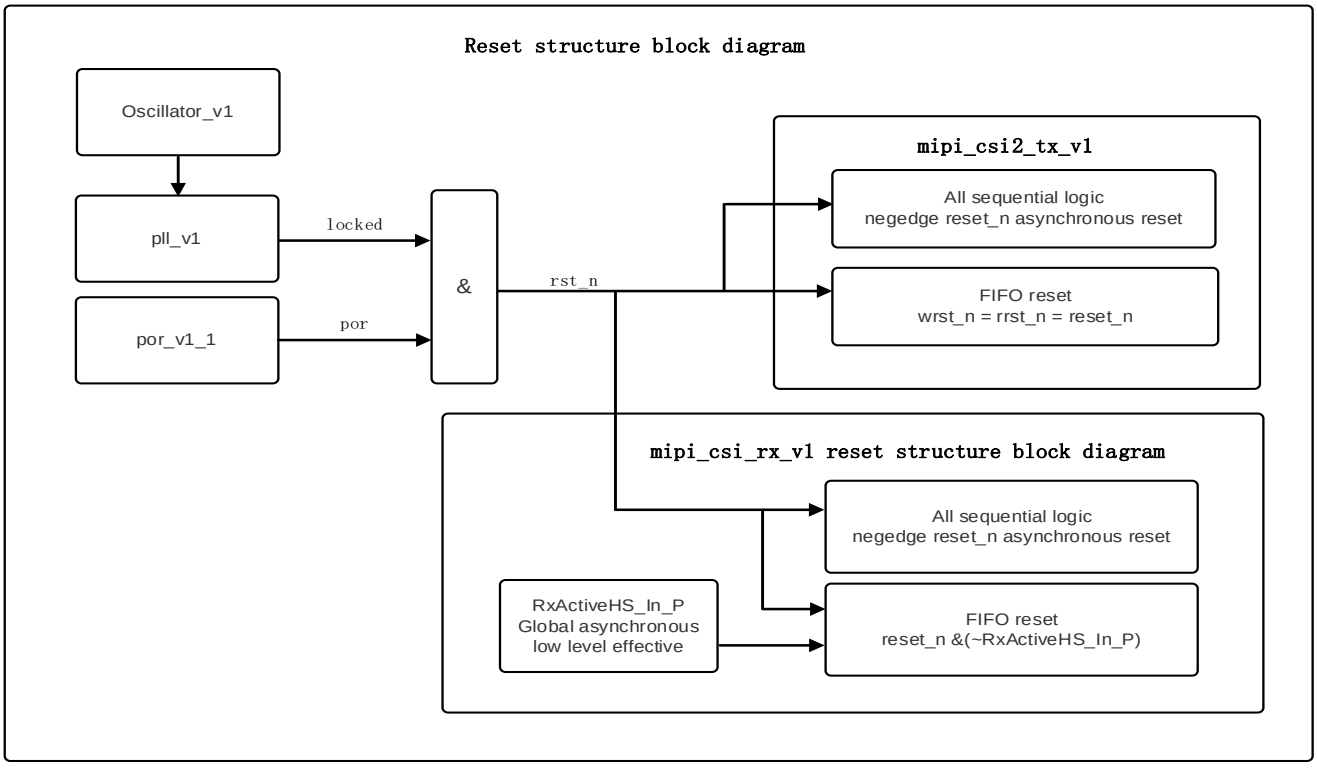

Figure 3-1 Reset Structure Block Diagram

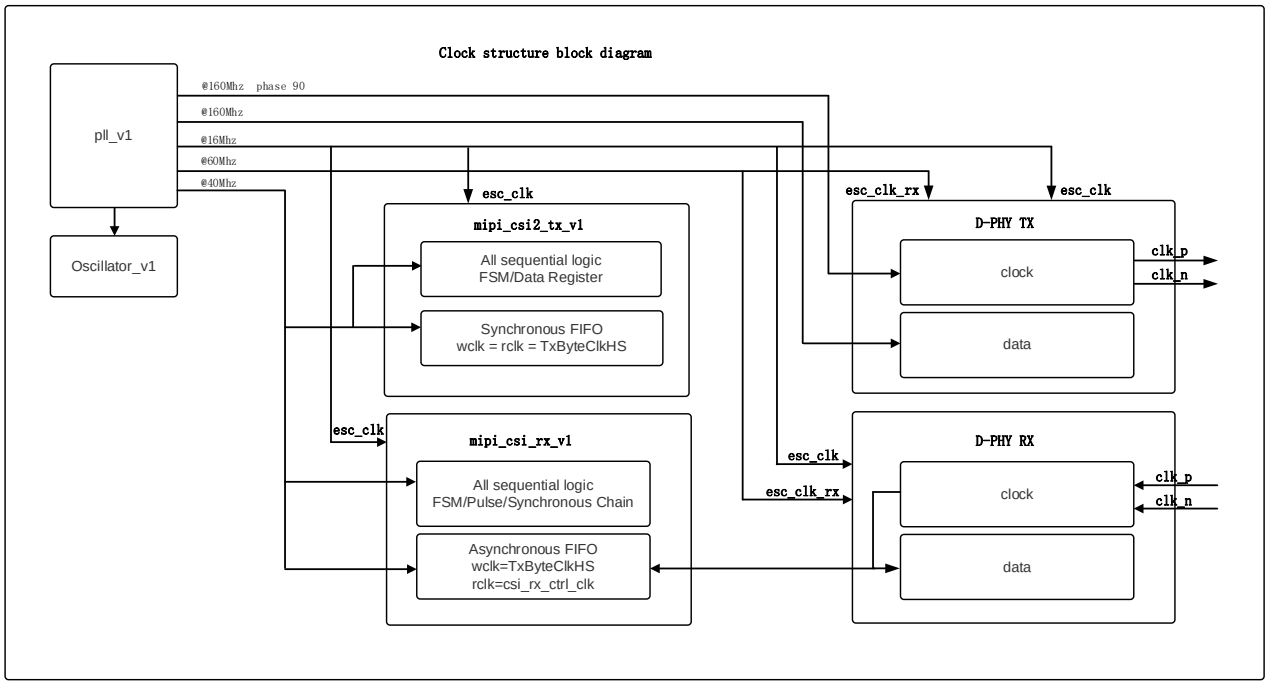

Figure 3-2 Clock Structure Block Diagram

3.2 MIPI CSI-2 Protocol

MIPI CSI-2 defines two packet types, short and long. The packet header consists of a Data Identifier (DI), a Word Count (WC), and an ECC check code. A short packet has no payload and no CRC and is used for frame / line synchronization. A long packet is followed by WC payload bytes and ends with a 16-bit CRC16-CCITT check. ECC uses a Hamming code over the 24-bit header to generate an 8-bit check, correcting 1 and detecting 2 bits. CRC16-CCITT uses polynomial 0x8408 (reflected; normal form 0x1021) with initial value 0xFFFF.

Table 3-1 Data Type Encoding

Data Type	Description	Category
0x00	Frame Start	Short packet
0x01	Frame End	Short packet
0x02	Line Start	Short packet
0x03	Line End	Short packet
0x08~0x0F	Generic Short Packet	Short packet
0x10	Null Packet	Long packet
0x11	Blanking Data	Long packet
0x12	Embedded 8-bit non-image data (Embedded Data)	Long packet
0x1E	YUV422 8-bit	Long packet
0x22	RGB565	Long packet
0x2A	RAW8	Long packet (4 pixels/word)
0x2B	RAW10	Long packet (4 pixels/5 bytes)
0x2C	RAW12	Long packet (2 pixels/3 bytes)
0x2D	RAW14	Long packet

Data Type	Description	Category
0x2E	RAW16	Long packet
0x30~0x37	User-defined 8-bit data	Long packet

3.3 MIPI CSI-2 Interface timing

The PPI interface timing is illustrated for Lane0 to show the interaction between the controller and the D-PHY. TX direction: When the FIFO is not empty, TxRequestHS is pulled high, and the D-PHY responds with TxReadyHS; when both are valid simultaneously, TxDataHS is output byte by byte, and the data is retained during the D-PHY backpressure period.

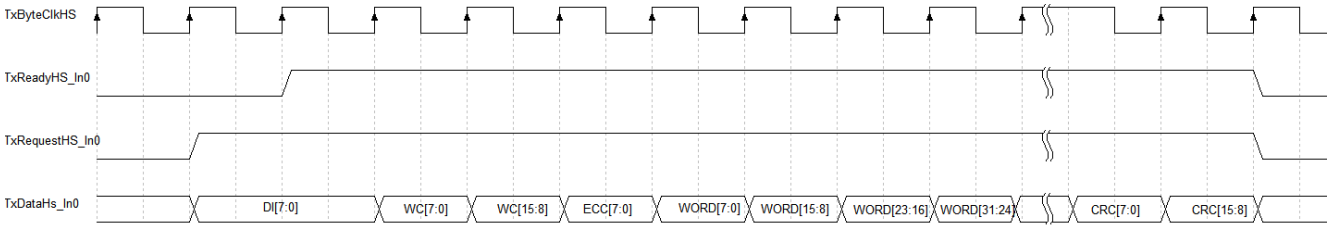

Figure 3-3 TX PPI Interface Timing (1-Lane Mode)

In the 2-Lane mode, the packet header and the payload bytes are alternately distributed to the two channels for parallel transmission, and the synchronous sequence only appears in Lane 0.

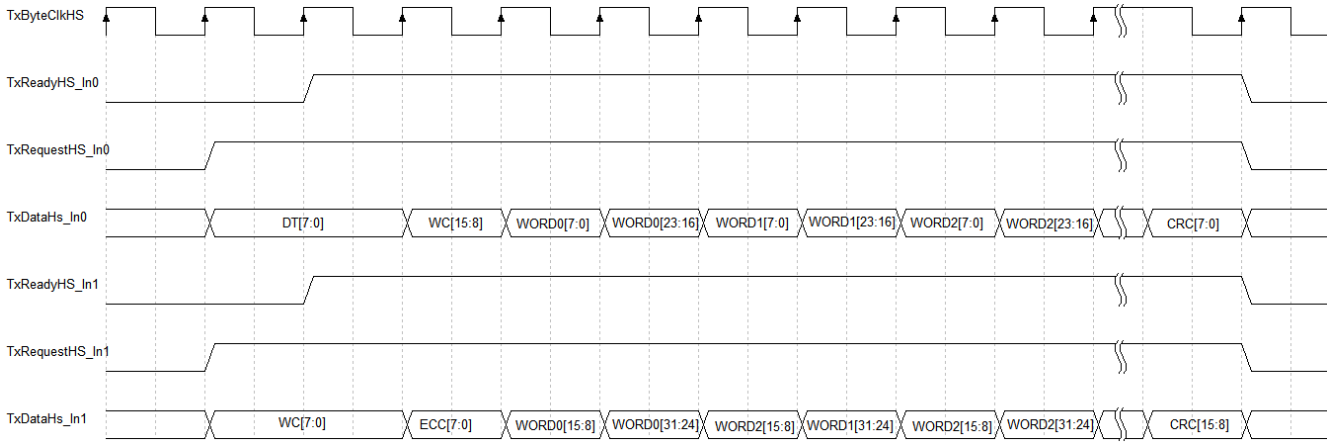

Figure 3-4 TX PPI Interface Timing (2-Lane Mode)

In 4-Lane mode, each transmit buffer directly corresponds to one lane. The packet header and payload are distributed in parallel to four lanes for transmission at 4 bytes per cycle.

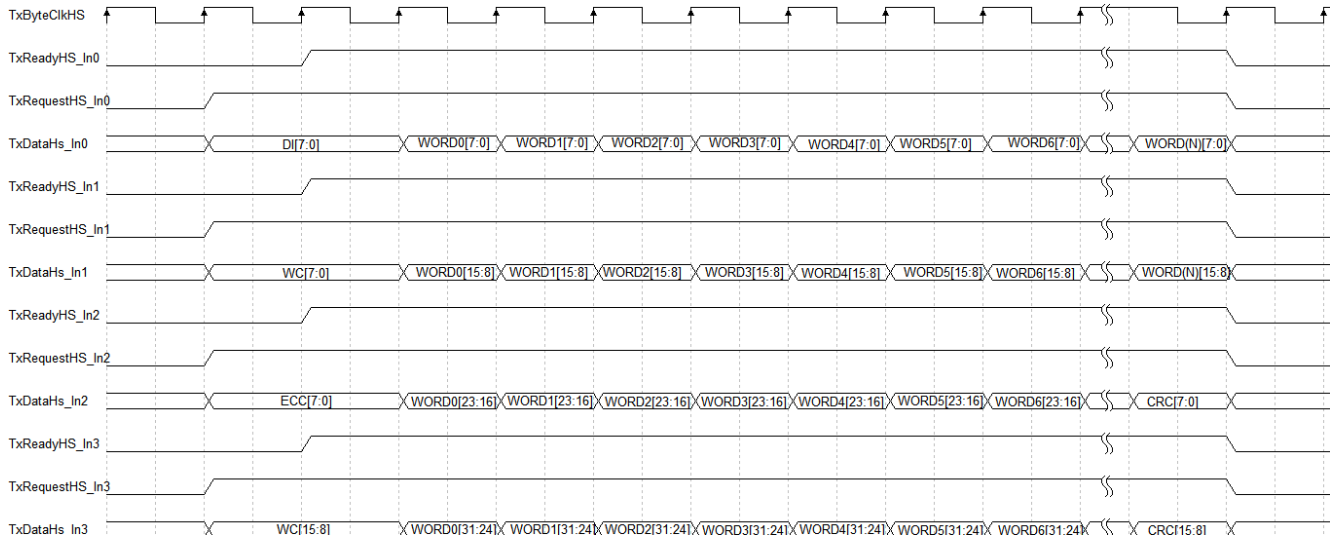

Figure 3-5 TX PPI Interface Timing (4-Lane Mode)

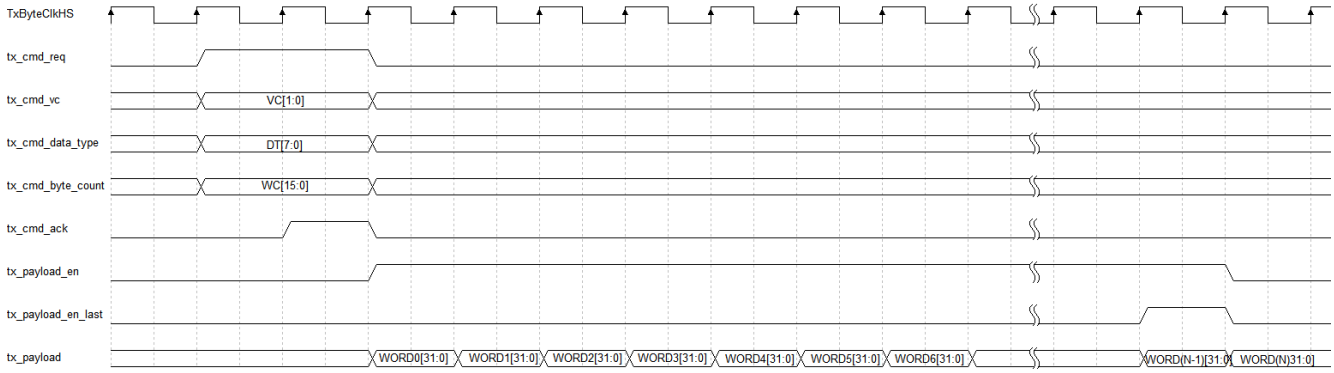

Figure 3-6 TX Long Packet Transmission Timing

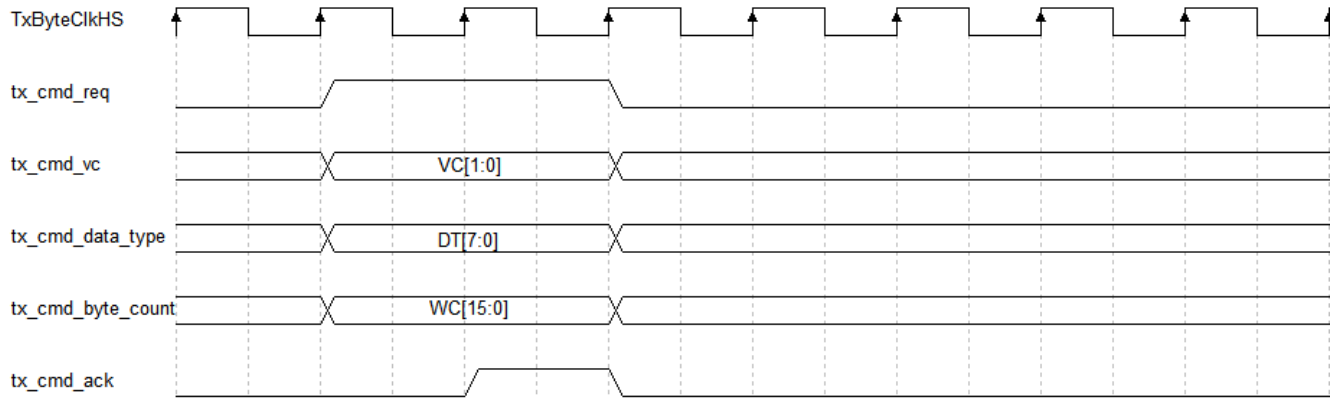

Figure 3-7 TX Short Packet Transmission Timing

In single-Lane mode: The 2-Lane mode transfers 1 byte per cycle, and the packet header is collected over 4 read cycles (DT+WC_L, WC_H+ECC).

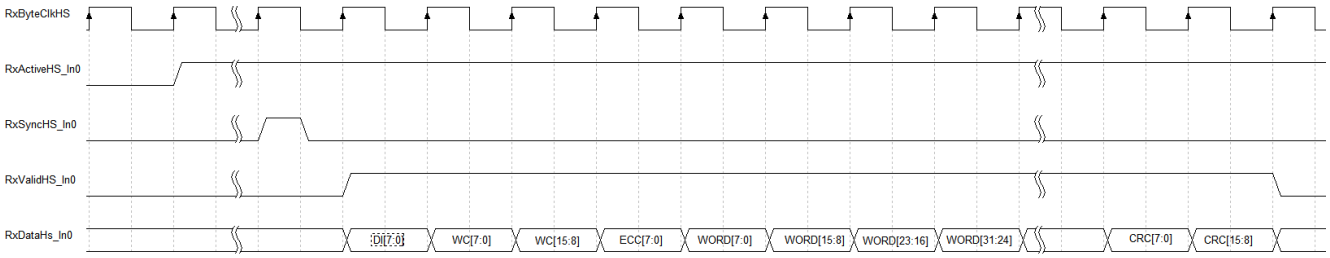

Figure 3-8 RX PPI Interface Timing (1-Lane Mode)

In the multi-Lane mode, each Lane concurrently carries byte streams: In the 2-Lane mode, 2 bytes are transmitted per cycle (even-numbered bytes for Lane 0 and odd-numbered bytes for Lane 1), and the header is collected in 2 read cycles (DT + WC_L, WC_H + ECC); in the 4-Lane mode, 4 bytes are transmitted per cycle, and the header is collected in 1 read cycle (DI, WC_L, WC_H, ECC each occupying one Lane), and the ECC byte can be output independently by rx_cmd_ecc. The load is output in 32-bit groups under different Lane numbers. The timing for 2/4-Lane is as shown in the figure below. Note: rx_cmd_ecc only outputs valid values in the 4-Lane mode, and is always 0 in the 1/2-Lane mode.

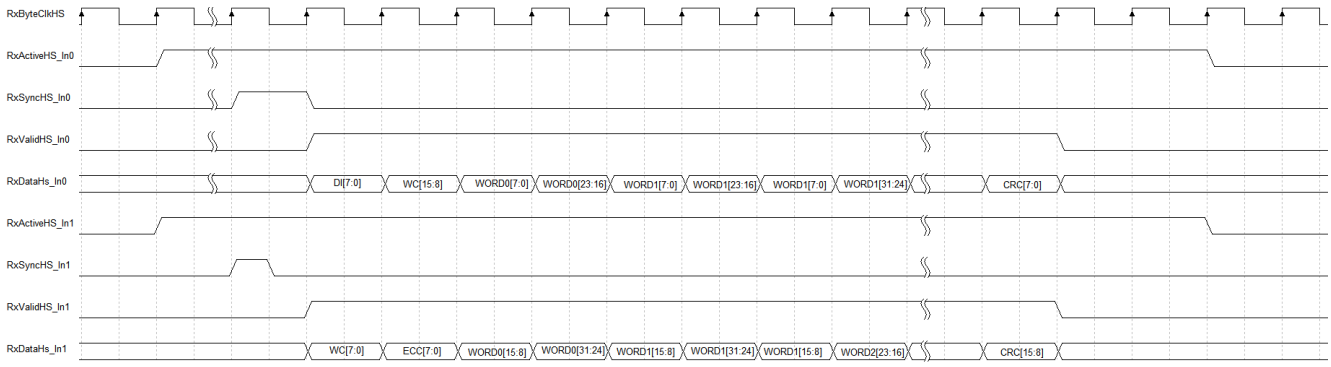

Figure 3-9 RX PPI Interface Timing (2-Lane Mode)



Figure 3-10 RX PPI Interface Timing (4-Lane Mode)

The packet reception timing is shown in the figure below: After the controller receives the packet header, it asserts a single-cycle rx_cmd_valid pulse together with rx_cmd header information. Short packets carry no payload, so rx_payload_valid and rx_payload_valid_last remain deasserted. For long packets, payload is received subsequently; rx_payload_valid stays asserted during payload transmission, and

rx_payload_valid_last is asserted at the last cycle of the payload.

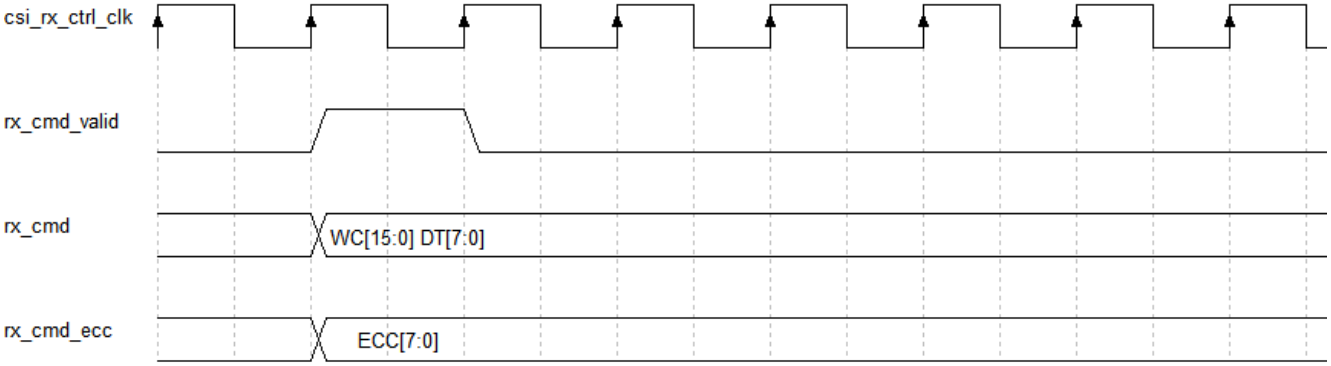

Figure 3-11 RX Short Packet Reception Timing

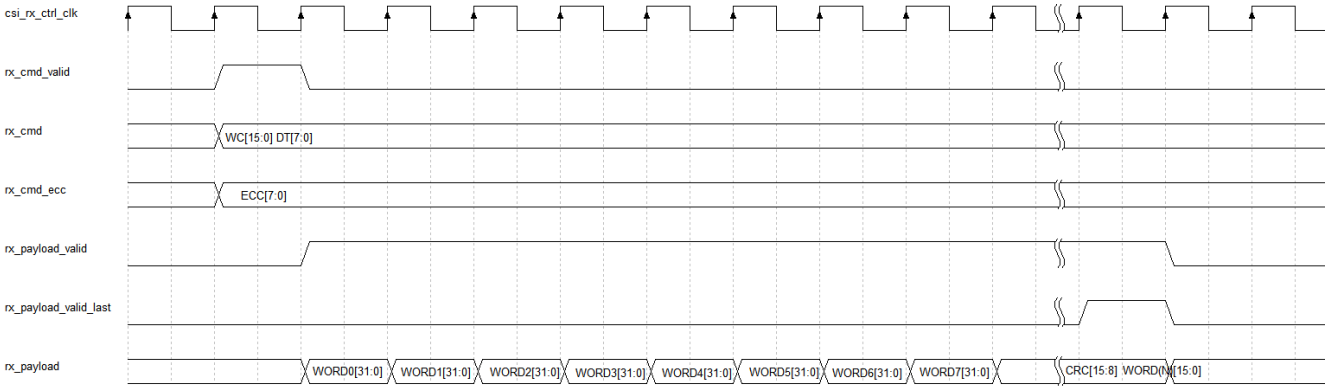

Figure 3-12 RX Long Packet Reception Timing

3.4 IP Constraint Method

3.4.1 Example of Clock Declaration

Note: The following clock declarations are excerpted from the reference project for illustration only. The periods of the input clock and each PLL output (CO0~CO4) depend on the specific design. Users must recalculate each clock period based on the actual line rate and PHY reference clock, and shall not directly copy the example constraints.

```
create_clock -period 12.5 -name {clkin_25m} [get_pins {u_osc/OSC}]

create_clock -period 62.5 -name {u_pll_pll_u0/CO0} [get_pins {u_pll_pll_u0/CO0}]

create_clock -period 16.6667 -name {u_pll_pll_u0/CO1} [get_pins {u_pll_pll_u0/CO1}]

create_clock -period 2.5 -name {u_pll_pll_u0/CO2} [get_pins {u_pll_pll_u0/CO2}]

create_clock -period 2.5 -name {u_pll_pll_u0/CO3} [get_pins {u_pll_pll_u0/CO3}]
```

3.4.2 Cross-Clock Domain Processing

```
set_false_path -from [get_clocks {clkin_25m}] -to [get_clocks {u_mipi_tx_u_mipi2/clk}]

set_false_path -from [get_clocks {u_mipi_tx_u_mipi2/clk}] -to [get_clocks {clkin_25m}]

set_false_path -from [get_clocks {clkin_25m}] -to [get_clocks {u_pll_pll_u0/CO0}]

set_false_path -from [get_clocks {u_pll_pll_u0/CO0}] -to [get_clocks {clkin_25m}]

set_false_path -from [get_clocks {u_pll_pll_u0/CO0}] -to [get_clocks {u_pll_pll_u0/CO1}]

set_false_path -from [get_clocks {u_pll_pll_u0/CO1}] -to [get_clocks {u_pll_pll_u0/CO0}]
```

Note: The above false path constraints are all examples. The cross-clock domain paths must be set according to the actual clock domain relationship in the design. The clock names and module-level paths (such as u_pll_pll_u0/CO0, u_mipi_tx_u_mipi2/clk) in the example may vary depending on the project, and users should modify them according to their own projects.

3.5 IP Generation of IP Core

3.5.1 IP Wizard Parameter Design

The MIPI CSI-2 transmit/receive controller in the ip_core directory of the routine project was generated by the IP Wizard of Fuxi Software (FX). If you need to adjust the number of channels or change the module name, you can open the configuration wizard on the IP core entry in the project view (or in the menu Tools → Wizard Manager) to regenerate it. The IP Wizard consists of two pages.The first page (Input Module Name) sets the module information: module name (default corresponding to the RX/TX controller of this IP, can be modified), output file save location, generation language (Verilog) and target device (such as P2L50N0V324-M0X1C7), and supports Match project / default to use the default settings of the project; the left side of this page displays the IP symbol and all ports in real time.

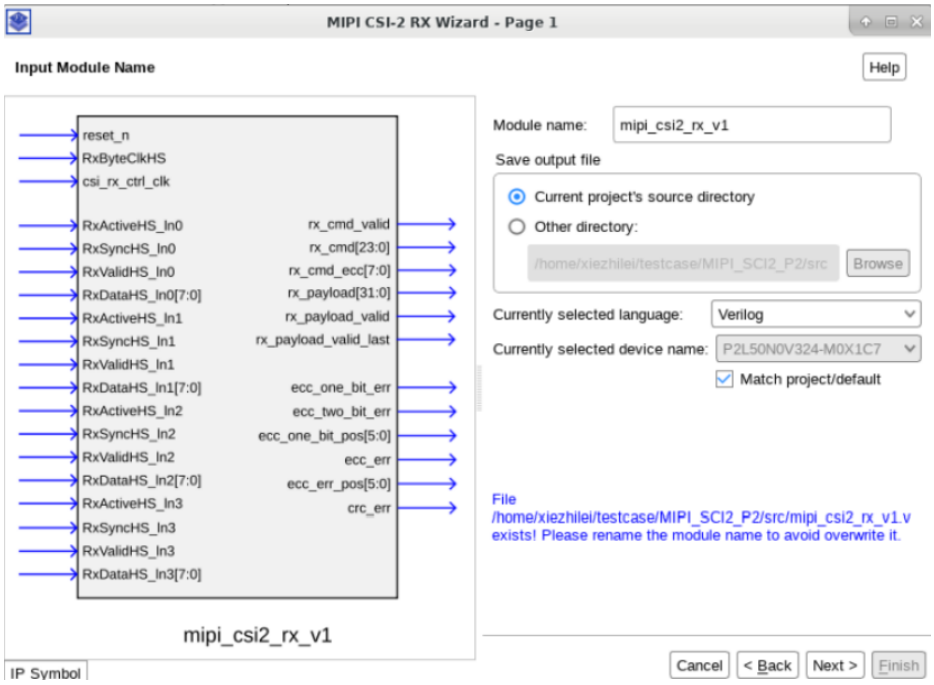

Figure 3-13 Page 1 of RX IP Wizard (Input Module Name)

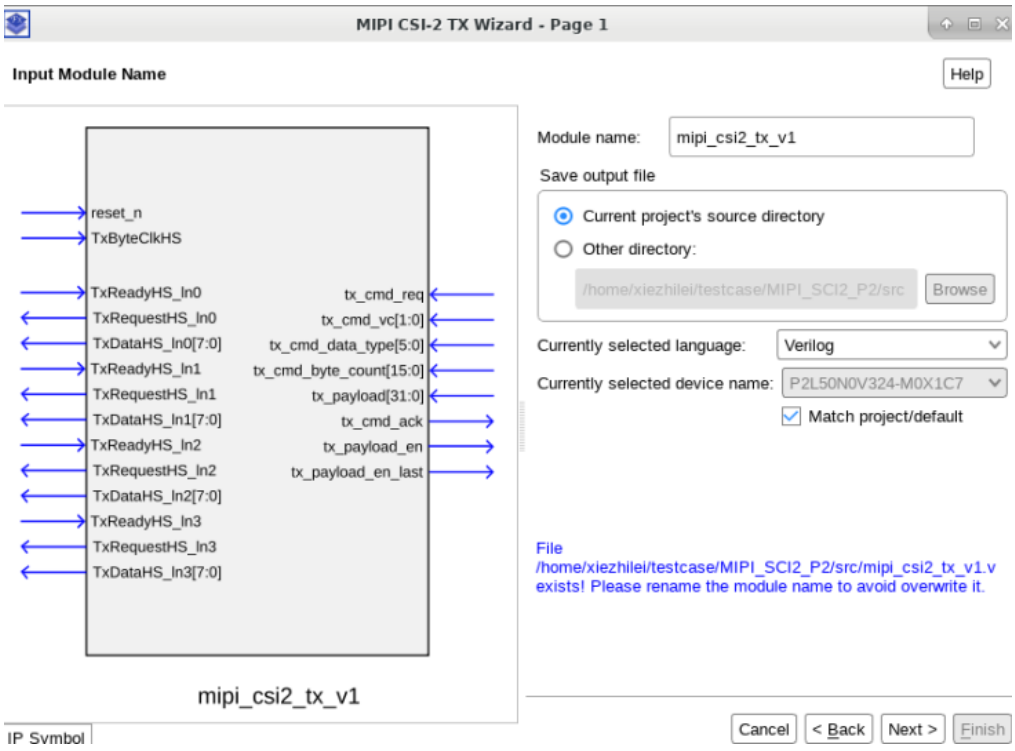

Figure 3-14 Page 1 of TX IP Wizard (Input Module Name)

The second page is the parameter page, which only provides one configuration item named "Number for lanes" (i.e., NUM_LANES, both TX and RX support 1/2/4). The IP symbol on the left updates the number of ports in real time according to the parameters; after clicking "Finish", the top-level encapsulation and source file will be generated based on the selected parameters. By default, the routine uses the 4-Lane configuration. If it is changed to 1-Lane or 2-Lane, the dangling handling for the unused Lane signals in the top-level of the routine needs to be adjusted synchronously.

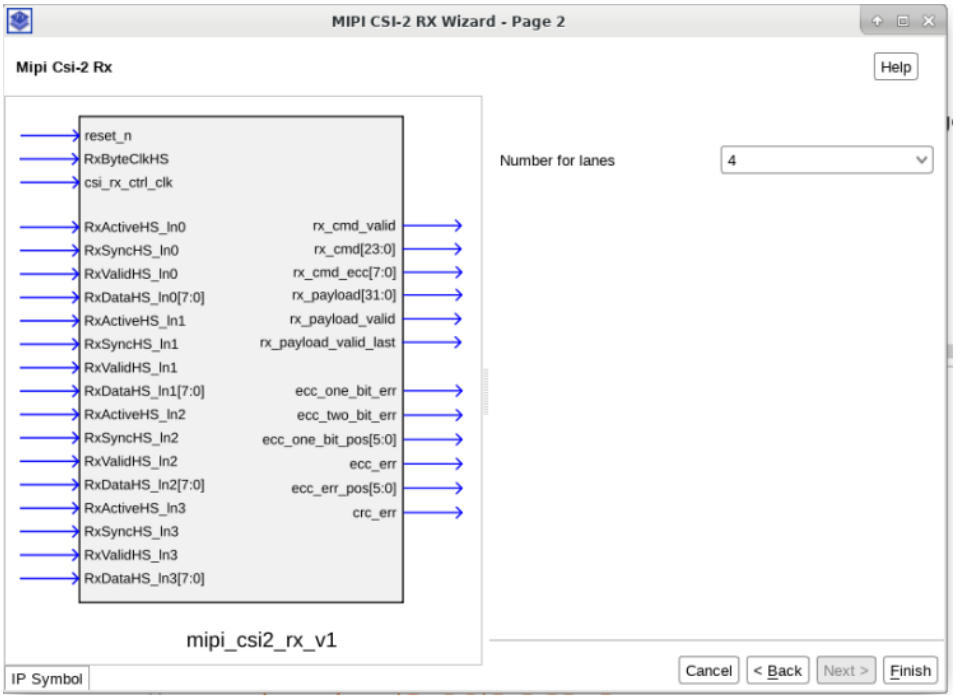

Figure 3-15 RX Parameter Page (Number of lanes = 4)

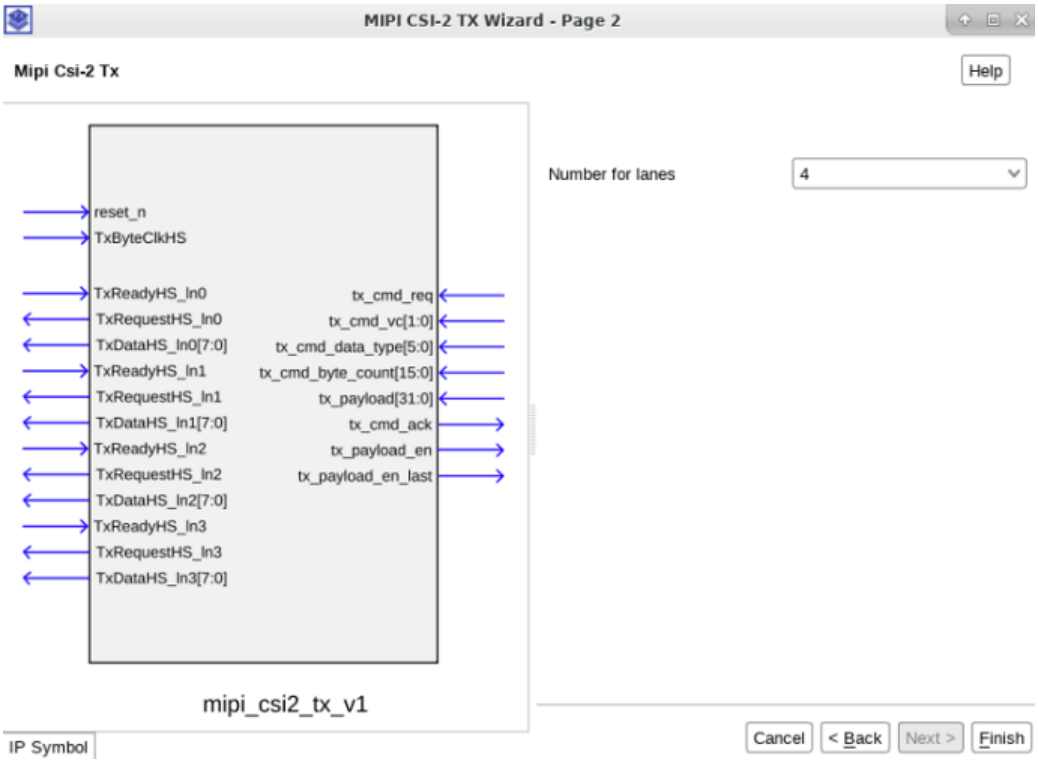

Figure 3-16 TX Parameter Page (Number of lanes = 4)

3.6 Resource and Performance Report

3.6.1 System Design Block Diagram

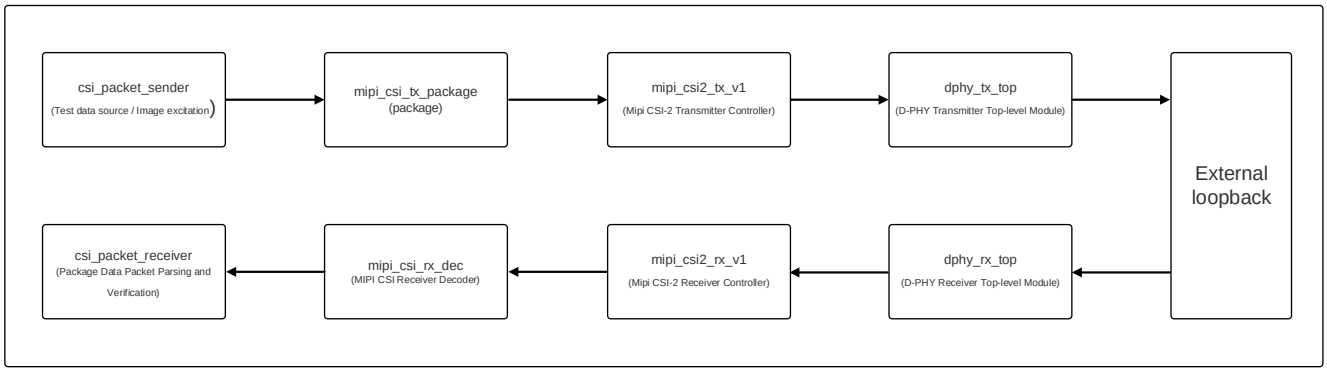

Figure 3-17 Reference Routine System Design Block Diagram

3.6.2 Resource Report

Working Mode	LUT	REG	BRAM	PLL
TX Controller (Pure Protocol Core, 1/2/4 Lane)	522/505/478	728/726/724	-	-
RX Controller (Pure Protocol Core, 1/2/4 Lane)	105/130/191	186/221/302	-	-

3.6.3 Performance Report

Device	FMAX
P2L50N0V324-M4H1C7	235 MHz

3.7 Usage Notes

- Cross-clock domain: The read and write operations of the RX FIFO are performed using two independent clocks. Before releasing the reset, it is necessary to ensure the stability of both clocks.
- Application layer bandwidth: MIPI does not allow the output side of PPI to backpressure. Data must be continuously supplied during the validity period of tx_payload_en.

4 Generate File Directory Structure

The MIPI CSI-2 IP file is automatically generated by the wizard and mainly consists of: source files (src) and related documentation. The details are shown in the following table.

Table 4-1 File Structure

File Path	Description
Source file directory	
src/p2_h7	IP Core file, used for synthesis
doc	
doc/HME_MIPI_CSI-2_Controller_IP_User_Guide_CN.pdf doc/HME_MIPI_CSI-2_Controller_IP_User_Guide_EN.pdf	IP usage guide