



DesignWare® Cores MIPI CSI-2 Device Controller

Databook

DWC MIPI CSI-2 HP Device Controller - Product Code: E822-0

DWC MIPI CSI-2 HP Device Combo Controller - Product Code: E823-0

DWC AP MIPI CSI-2 HP Device Controller - Product Code: E820-0

DWC AP MIPI CSI-2 HP Device Combo Controller - Product Code: E821-0

DWC MIPI CSI-2 Device Controller - Product Code: B148-0

DWC MIPI CSI-2 Device Combo Controller - Product Code: D740-0

DWC AP MIPI CSI-2 Device Controller - Product Code: B915-0

DWC AP MIPI CSI-2 Device Combo Controller - Product Code: D739-0

Copyright Notice and Proprietary Information

© 2021 Synopsys, Inc. All rights reserved. This Synopsys software and all associated documentation are proprietary to Synopsys, Inc. and may only be used pursuant to the terms and conditions of a written license agreement with Synopsys, Inc. All other use, reproduction, modification, or distribution of the Synopsys software or the associated documentation is strictly prohibited.

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at <https://www.synopsys.com/company/legal/trademarks-brands.html>

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.

www.synopsys.com

Contents

Revision History	7
Preface	11
Databook Organization	11
Related Documentation	11
Web Resources	12
Synopsys Statement on Inclusivity and Diversity	12
Customer Support	12
Chapter 1	
Product Overview	15
1.1 General Product Description	16
1.1.1 Applications	16
1.1.2 Standards Compliance	17
1.2 DWC_mipicsi2_device Features	18
1.2.1 Interfaces	20
1.2.2 Supported Resolutions and Frame Rates	21
1.2.3 Unsupported Features and Exceptions	23
1.3 Deliverables	24
1.4 Operational Model Overview	25
1.5 Speed and Clock Requirements	27
Chapter 2	
Architecture	29
2.1 Overview of Architecture	31
2.2 Image Data Interface	33
2.2.1 Overview of Image Data Interface	33
2.2.2 Description of Image Data Interface	33
2.2.3 Enabling IDI	46
2.2.4 Registers related to IDI	46
2.2.5 Signals related to IDI	47
2.3 Image Pixel Interface	48
2.3.1 Overview of Image Pixel Interface	48
2.3.2 Color Components Mapping	48
2.3.3 IPI Store and Forward Mode	50

2.3.4 IPI Cut-Through Mode	52
2.3.5 IPI Line Synchronization Omission Mode	54
2.3.6 IPI Backpressure Mode	55
2.3.7 IPI Embedded Data	57
2.3.8 Enabling IPI	58
2.3.9 Registers Related to IPI	58
2.3.10 Signals Related to IPI	58
2.3.11 Programming IPI	59
2.4 Multiple IPI Support.....	60
2.4.1 Multiple IPI with No Timing Delivery Mode	60
2.4.2 Multiple IPI with Timing Delivery Mode	60
2.4.3 Enabling Multiple IPI	68
2.4.4 Registers Related to Multiple IPI	68
2.4.5 Programming Multiple IPI	69
2.5 Simultaneous IDI and IPI Mode	70
2.5.1 Overview of Simultaneous IDI and IPI Mode	70
2.5.2 Insertion of IDI Packets in IPI Transmission	70
2.5.3 Time Calculation of IDI Insertion	72
2.5.4 Error Scenario	75
2.5.5 Enabling Simultaneous IDI and IPI Mode	76
2.5.6 Registers Related to Simultaneous IDI and IPI Mode	76
2.5.7 Programming Simultaneous IDI and IPI Mode	76
2.6 Stream Data Interface	77
2.6.1 Overview of Stream Data Interface	77
2.6.2 Description of Stream Data Interface	77
2.6.3 Registers Related to SDI	83
2.6.4 Signals Related to SDI	83
2.6.5 Programming SDI	84
2.7 Video Pattern Generator.....	85
2.7.1 Overview of the Video Pattern Generator	85
2.7.2 Color Bar Pattern	85
2.7.3 Color Coding	86
2.7.4 BER Testing Pattern	88
2.7.5 Video Pattern Generator Resolution	89
2.7.6 Enabling Video Pattern Generator	90
2.7.7 Registers Related to Video Pattern Generator	90
2.8 Memory.....	91
2.8.1 Image Data Interface (IDI) Minimum Memory Size	92
2.8.2 Image Pixel Interface (IPI) Memory Size	92
2.8.3 Memory Size for Automotive Safety	96
2.9 Protocol-PHY Interface (PPI).....	97
2.10 LRTE Support	98
2.10.1 Inter-packet Latency Reduction (ILR)	98
2.10.2 Enhanced Transport Efficiency	103
2.11 Data Scrambling	105
2.11.1 Overview of Data Scrambling	105
2.11.2 Enabling the Scrambler	107
2.11.3 Related Registers	108
2.12 Interrupts	109

2.13 Error Detection	111
----------------------------	-----

Chapter 3

Parameter Descriptions	113
3.1 Generic Configuration Parameters.....	114
3.2 Data Interface FIFO Configuration Parameters	118
3.3 Data Interface FIFO Configuration / IDI FIFO Parameters	119
3.4 Data Interface FIFO Configuration / IPI FIFO Parameters.....	121
3.5 Data Interface FIFO Configuration / IPI2 FIFO Parameters.....	123
3.6 Data Interface FIFO Configuration / IPI3 FIFO Parameters.....	125
3.7 Data Interface FIFO Configuration / IPI4 FIFO Parameters.....	127

Chapter 4

Signal Descriptions	129
4.1 AMBA Slave Interface Signals.....	131
4.2 Interrupt Signals	134
4.3 FMEDA Interrupt Signals	135
4.4 SSM Bus Interface Signals	136
4.5 Diagnosis fault Signals	139
4.6 IDI Interface Signals	143
4.7 IDI Payload RAM Signals	148
4.8 SDI Interface Signals.....	153
4.9 IPI Interface Signals.....	156
4.10 IPI Payload RAM Signals	166
4.11 PHY Signals	177
4.12 PHY Protocol Interface (PPI) Signals	180
4.13 PHY Protocol Interface (PPI) Signals for D-PHY Clock Lane..	181
4.14 PHY Protocol Interface (PPI) Signals for D-PHY	183
4.15 PHY Protocol Interface (PPI) Signals for Data Lane N	184
4.16 Parallel Port for PHY Configuration Signals.....	189
4.17 PHY Protocol Interface (PPI) Signals for Calibration Signals..	191
4.18 Scan Chain Signals	192

Chapter 5

Register Descriptions	193
5.1 Component Memory Maps	196
5.2 DWC_mipicsi2_device_MemMap Memory Map Registers	197
5.2.1 DWC_mipicsi2_device_MemMap/MAIN Registers	207
5.2.2 DWC_mipicsi2_device_MemMap/INT Registers	216
5.2.3 DWC_mipicsi2_device_MemMap/VPG Registers	297
5.2.4 DWC_mipicsi2_device_MemMap/PHY Registers	315
5.2.5 DWC_mipicsi2_device_MemMap/IDI Registers	351
5.2.6 DWC_mipicsi2_device_MemMap/SDI Registers	354
5.2.7 DWC_mipicsi2_device_MemMap/IPI Registers	356
5.2.8 DWC_mipicsi2_device_MemMap/AP_INT Registers	429
5.2.9 DWC_mipicsi2_device_MemMap/AP Registers	616
5.3 DW_safety_slave_map Memory Map Registers.....	638

5.3.1 DW_safety_slave_map/DW_safety_slave_block Registers	641
Appendix A	
Area and Power	699
A.1 The standard controller (DWC_mipicsi2_device).....	700
A.2 The standard controller with automotive (DWC_ap_mipicsi2_device)	703
A.3 The combo controller (DWC_mipicsi2_device_combo).....	706
A.4 The combo controller with automotive (DWC_ap_mipicsi2_device_combo)	708
A.5 The high-performance controller (DWC_mipicsi2_hp_device)	711
A.6 The high-performance controller with automotive (DWC_ap_mipicsi2_hp_device)	713
A.7 The high-performance and combo controller (DWC_mipicsi2_hp_device_combo)	716
A.8 The high performance and combo controller with automotive (DWC_ap_mipicsi2_hp_device_combo). 718	
Appendix B	
DWC_mipicsi2_device Automotive Safety	721
B.1 Overview of Automotive Safety Feature.....	722
B.1.1 ASIL B Certification	722
B.2 Automotive Safety Package Documents	723
B.3 Signals Related to Automotive Safety Feature	724
B.4 Programming Automotive Safety Feature	725
B.5 Configuring Automotive Safety Feature.....	726
B.6 Automotive Package Safety Memory Requirements.....	727
B.6.1 IPI RAM requirement	727
B.7 Memory Error Injection	728
B.8 Faulty Address Logger.....	729
B.9 Safety Slave	730
B.9.1 Overview of Safety Slave	730
B.9.2 Description of Safety Slave	730
Appendix C	
DWC_mipicsi2_device Packet Transmission Calculation	735
C.1 LRTE Disable.....	736
C.2 LRTE Enable	738
C.2.1 D-PHY EPD Options 1 and 2	738
C.2.2 C-PHY Option	739
Appendix D	
Internal Parameter Descriptions.....	741

Revision History

The following table provides a summary of changes made to this Databook:

Date	Version	Description
May 2021	1.32a	<p>Updated:</p> <ul style="list-style-type: none">■ “DWC_mipicsi2_device Features” on page 18■ “Signals Related to IPI” on page 58■ “Error Detection” on page 111■ Chapter 3, “Parameter Descriptions”■ Chapter 4, “Signal Descriptions”■ Chapter 5, “Register Descriptions”■ Appendix D, “Internal Parameter Descriptions”■ Appendix A, “Area and Power” <p>Added</p> <ul style="list-style-type: none">■ “IPI Embedded Data” on page 57
October 2019	1.31a	<p>Updated:</p> <ul style="list-style-type: none">■ “Parameter Descriptions” on page 89■ “Signal Descriptions” on page 97■ “Register Descriptions” on page 141■ “Internal Parameter Descriptions” on page 523■ “Area and Power” on page 49

Date	Version	Description
March 2019	1.30a	<p>Added:</p> <ul style="list-style-type: none">■ New section “RAW20 Data Reception” on page 39■ New section “LRTE Support” on page 79■ Figure 2-57 on page 83■ Figure 2-59 on page 84■ New chapter “DWC_mipicsi2_device Packet Transmission Calculation” on page 519 <p>Updated:</p> <ul style="list-style-type: none">■ “Standards Compliance” on page 19■ “DWC_mipicsi2_device Features” on page 20■ “Supported Resolutions and Frame Rates” on page 21■ Table 1-1 on page 22■ Table 1-4 on page 26■ “Unsupported Features and Exceptions” on page 23■ changed D-PHY title to “32-Bit D-PHY” on page 49■ “16-Bit-PHY” on page 48■ “32-Bit D-PHY” on page 49■ Table 2-11 on page 78

Date	Version	Description
May 2018	1.20a	<p>Added:</p> <ul style="list-style-type: none"> ■ Support for C-PHY ■ “IPI Backpressure Mode” on page 50 ■ “Multiple IPI Support” on page 52 ■ “Overview of Simultaneous IDI and IPI Mode” on page 60“ ■ “Data Scrambling” on page 82 ■ “Protocol-PHY Interface (PPI)” on page 78 <p>Updated:</p> <ul style="list-style-type: none"> ■ Chapter 1, “Product Overview” <ul style="list-style-type: none"> □ Figure 1-1 on page 18 □ Figure 1-2 on page 25 □ “Standards Compliance” on page 19 □ “DWC_mipicsi2_device Features” on page 20 □ “Interfaces” on page 21 □ Table 1-1 on page 22 □ Table 1-2 on page 23 □ Table 1-4 on page 26 ■ Chapter 2, “Architecture” <ul style="list-style-type: none"> □ “Overview of Architecture” on page 29 □ Figure 2-1 on page 29 □ Figure 2-3 on page 32 □ “IDI Driver Timing Requirement” on page 41 □ Figure 2-26 on page 45 □ “IPI Timing Driver Requirement for Backpressure Mode” on page 51 □ “ipi_data_timing Calculation” on page 47 □ “ppi_data_timing Calculation” on page 47 □ Figure 2-30 on page 48 □ “Signals related to IPI” on page 58 □ “Registers Related to Video Pattern Generator” on page 72 □ “Error Detection” on page 88 ■ Chapter 3, “Parameter Descriptions” ■ Chapter 4, “Signal Descriptions” ■ Chapter 5, “Register Descriptions” ■ Appendix D, “Internal Parameter Descriptions”
July 2017	1.11a	<ul style="list-style-type: none"> ■ Automotive Safety Package support is not included in this release ■ Added: <ul style="list-style-type: none"> □ “Memory Error Injection” □ “Faulty Address Logger” ■ “Parameter Descriptions”, “Signal Descriptions”, and “Register Descriptions” chapters updated and auto-extracted from source code

Date	Version	Description
January 2017	1.10a	<ul style="list-style-type: none">■ Added:<ul style="list-style-type: none">□ “For details about this option, see “Signal Descriptions”□ “VPG_STEP_LINE_NUM”□ “DWC_mipicsi2_device Automotive Safety”■ “Parameter Descriptions”, “Signal Descriptions”, and “Register Descriptions” chapters updated and auto-extracted from source code■ Restructured the “Architecture” to contain Functional Description chapter details■ Moved Area and Power information to an Appendix - “Area and Power”
May 2016	1.00a	First General Availability (GA) release

Preface

This databook describes the DesignWare® Cores MIPI CSI-2 Device Controller (DWC_mipicsi2_device), which along with Synopsys' DWC MIPI D-PHY is a part of the complete MIPI CSI-2 interface solution.

Databook Organization

The chapters of this databook are organized as follows:

- [Chapter 1, "Product Overview"](#), provides an introduction to the DWC_mipicsi2_device, including a block diagram, supported standards, supported features, deliverables, and so on.
- [Chapter 2, "Architecture"](#), describes the DWC_mipicsi2_device architecture and its main interfaces along with the functionalities.
- [Chapter 3, "Parameter Descriptions"](#), describes the hardware configuration parameters.
- [Chapter 4, "Signal Descriptions"](#), provides descriptions of inputs/outputs of the DWC_mipicsi2_device.
- [Chapter 5, "Register Descriptions"](#) provides the memory map of DWC_mipicsi2_device and the descriptions of the programmable software registers.
- [Appendix A, "Area and Power"](#), provides the area and power numbers of the DWC_mipicsi2_device for different configurations.
- [Appendix B, "DWC_mipicsi2_device Automotive Safety"](#), provides automotive features supported by the controller when you have the corresponding automotive license.
- [Appendix C, "DWC_mipicsi2_device Packet Transmission Calculation"](#), describes the packet transmission time calculation in PPI for each mode.
- [Chapter D, "Internal Parameter Descriptions"](#), provides a description of the internal parameters that might be indirectly referenced in expressions in the Signals, Parameters, or Registers chapters.

Related Documentation

Refer to the following documentation:

- [coreConsultant User Guide](#)
- [coreAssembler User Guide](#)

Web Resources

The following web links are various Synopsys online resources you may find useful:

- DesignWare IP product information: <https://www.synopsys.com/designware-ip.html>
- Your custom DesignWare IP page: <http://www.mydesignware.com>
- Documentation through SolvNet: <http://solvnet.synopsys.com> (Solvnet ID required)
- Synopsys Common Licensing (SCL): <http://www.synopsys.com/keys>

Synopsys Statement on Inclusivity and Diversity

Synopsys is committed to creating an inclusive environment where every employee, customer, and partner feels welcomed. We are reviewing and removing exclusionary language from our products and supporting customer-facing collateral. Our effort also includes internal initiatives to remove biased language from our engineering and working environment, including terms that are embedded in our software and IPs. At the same time, we are working to ensure that our web content and software applications are usable to people of varying abilities. You may still find examples of non-inclusive language in our software or documentation as our IPs implement industry-standard specifications that are currently under review to remove exclusionary language.

Customer Support

To obtain support for your product, prepare the required files and contact the support center using one of the methods described:

- Prepare the following debug information, if applicable:
 - For environment set-up problems or failures with configuration, simulation, or synthesis that occur within coreConsultant or coreAssembler, select the following menu:
 - **File > Build Debug Tar-file**
 - For simulation issues outside of coreConsultant or coreAssembler:
 - Create a waveforms file (such as VPD or VCD).
 - Identify the hierarchy path to the DesignWare instance.
 - Identify the timestamp of any signals or locations in the waveforms that are not understood.
 - *For the fastest response*, enter a case through SolvNetPlus:
 - a. <https://solvnetplus.synopsys.com>



SolvNetPlus does not support Internet Explorer. Use a supported browser such as Microsoft Edge, Google Chrome, Mozilla Firefox, or Apple Safari.

- b. Click the **Cases** menu and then click **Create a New Case** (below the list of cases).
- c. Complete the mandatory fields that are marked with an asterisk and click **Save**.

Make sure to include the following:

- **Product L1:** DesignWare Cores

- **Product L2:** L2_product_name

- d. After creating the case, attach any debug files you created.

For more information about general usage information, refer to the following article in SolvNet-Plus:

<https://solvnetplus.synopsys.com/s/article/SolvNetPlus-Usage-Help-Resources>

- Or, send an e-mail message to support_center@synopsys.com (your email will be queued and then, on a first-come, first-served basis, manually routed to the correct support engineer):

- ❑ Include the Product L1 and Product L2 names, and Version number in your e-mail so it can be routed correctly.
 - ❑ For simulation issues, include the timestamp of any signals or locations in waveforms that are not understood
 - ❑ Attach any debug files you created.

- Or, telephone your local support center:

- ❑ North America:

Call 1-800-245-8005 from 7 AM to 5:30 PM Pacific time, Monday through Friday.

- ❑ All other countries:

<https://www.synopsys.com/support/global-support-centers.html>

Product Overview

This chapter provides an introduction to the DWC_mipicsi2_device, including a block diagram, supported features, deliverables, supported standards, and so on. It has the following sections:

- “[General Product Description](#)” on page [16](#)
- “[DWC_mipicsi2_device Features](#)” on page [18](#)
- “[Deliverables](#)” on page [24](#)
- “[Operational Model Overview](#)” on page [25](#)
- “[Speed and Clock Requirements](#)” on page [27](#)

1.1 General Product Description

The DWC_mipicsi2_device implements the CSI-2 protocol on the device side. The CSI-2 link protocol specification is a part of communication protocols defined by MIPI Alliance standards intended for mobile system chip-to-chip communications. The CSI-2 specification is for the image application processor communication in cameras.

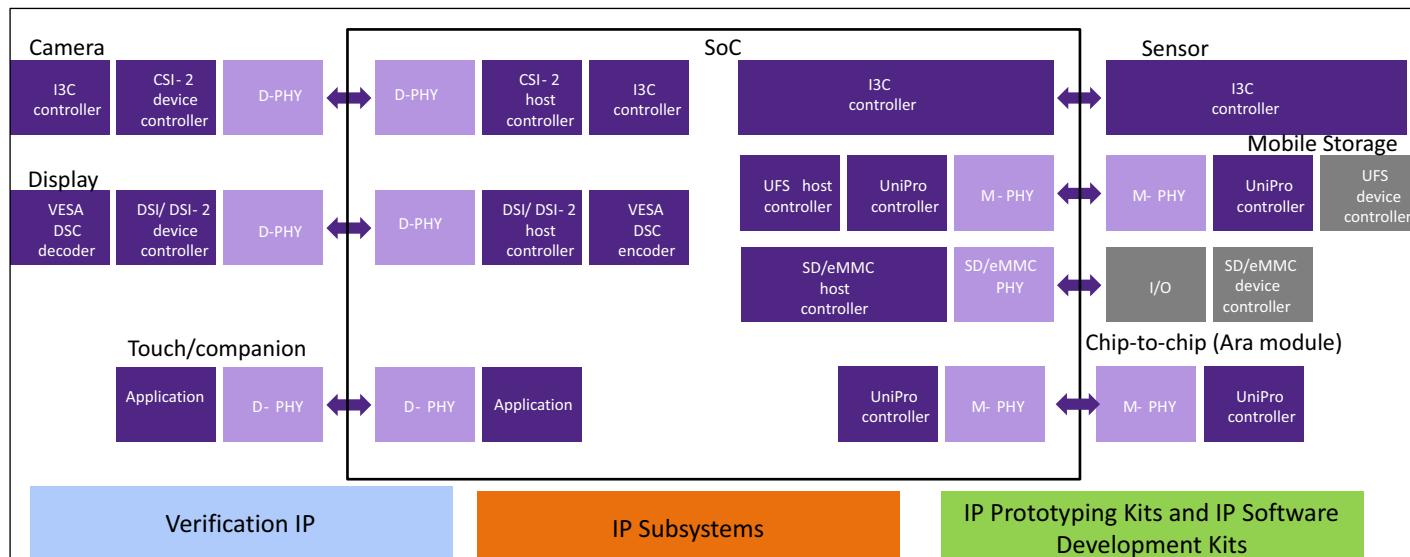
The DWC_mipicsi2_device is designed to send data to a CSI-2 compliant host. A D-PHY or C-PHY configured as a Master acts as the physical layer.

The DWC_mipicsi2_device includes the design and its verification environment. Synopsys also provides the coreConsultant tool for automated configuration, simulation, and synthesis of the controller.

The DWC_mipicsi2_device is designed to integrate with a Synopsys MIPI D-PHY. There is a broad range of D-PHYS that include bidirectional and TX D-PHYS with two lanes and four lanes for several technologies. For more information about the MIPI D-PHY, go to the Synopsys DesignWare MIPI D-PHY IP Solution page.

[Figure 1-1](#) shows the DWC_mipicsi2_device in an example system-on-chip design.

Figure 1-1 **DWC_mipicsi2_device in System-on-Chip Example**



1.1.1 Applications

Typical applications built with the DWC_mipicsi2_device are Mobile SoC, Application processors, and co-processors, that targets the following:

- Handheld devices
- Smartphone
- Multimedia tablets
- MID
- Navigation
- DSC
- DVC

- Automotive
- IoT
- Wearables

1.1.2 Standards Compliance

The DWC_mipicsi2_device conforms to the following standards:

- MIPI Alliance Specification for Camera Serial Interface 2 (CSI-2), Version 3.0, 10 September, 2019
- MIPI Alliance Specification for D-PHY, Version 2.5, 17 October, 2019
- MIPI Alliance Specification for C-PHY, Version 2.0, 9 September, 2019
- AMBA 2.0 Specification (APB) from Arm

1.2 DWC_mipicsi2_device Features

The standard controller DWC_mipicsi2_device supports the following features:

- Compliant with MIPI Alliance standards (See "[Standards Compliance](#)" on page [17](#))
- Optional support for Camera Control Interface (CCI) through the use of DesignWare Cores DW_apb_i2c controller
- Interface with MIPI D-PHY and MIPI C-PHY using PHY Protocol Interface (PPI), as defined in MIPI Alliance Specification for D-PHY and C-PHY, respectively
- D-PHY
 - Support 8-bit data width and up-to eight data lanes.
 - Up to 2.5 Gbps per lane
 - Programmable continuous and non-continuous clock behavior on clock lane
 - Up to 16 virtual channels
- C-PHY
 - Support 16-bit data width and up-to four trios.
 - Up to 2.5 Gsps (symbol per second) per trio
 - Up-to 32 virtual channels
- Long and Short packet encoding
- Automatic generation of ECC/PH-CRC for the Packet Header, and Checksum (CRC16) for the Packet Data
- Timing accurate signaling of Frame and Line synchronization packets
- 64-bit Image Data Interface (IDI)
- 48-bit Image Pixel Interface (IPI)
- Up to four simultaneous IPI interfaces with different virtual channel
- Embedded non-image data in IPI transmission
- Simultaneous IDI and IPI
- 64-bit Stream Data Interface
- Smart Region of Interest (SROI) capabilities using Stream Data Interface (SDI) or Image Data Interface (IDI)
- Support CSI2 v3.0 LRTE, EoTp
 - D-PHY LRTE option1 in 8-bit/16-bit/32-bit PPI mode.
 - D-PHY LRTE option2 in 8-bit PPI mode.
 - C-PHY LRTE in 16-bit/32-bit PPI mode.
- C-PHY alternative low power (ALP mode)
- PPI data scrambling
- PHY timing Auto Measurement
- All primary and secondary data formats
 - YUV420: 8-bit (legacy)/8-bit/10-bit/8-bit (CSPS)/10-bit (CSPS)
 - YUV422: 8-bit/10-bit

- ❑ RGB888/RGB666/RGB565/RGB555/RGB444
- ❑ RAW6/RAW7/RAW8/RAW10/RAW12/RAW14/RAW16/RAW20/RAW24
- ❑ Generic 8-bit long packet data types
- ❑ User-defined byte-based data
- Integrated Video Pattern Generator?
 - ❑ Vertical and horizontal color bar generation
 - ❑ Bit Error Rate (BER) pattern
- Memory Type?
 - ❑ Asynchronous dual port memory
 - ❑ Synchronous single port memory
- Error detection and recovery
 - ❑ PHY level – contention
 - ❑ Memory level – overflow/underflow
 - ❑ IDI/IPI level
 - ❑ VPG level

The DWC_mipicsi2_device_combo supports the following added features in the standard controller:

- Combo PHY 4L3T
 - ❑ Support DPHY 8-bit data width and four data lanes, up to 2.5 Gbps per lane
 - ❑ Support CPHY 16-bit data width and three trios, up to 2.5 Gsps (symbol per second) per trio
- Combo PHY 2L2T
 - ❑ Support DPHY 8-bit data width and two data lanes, up to 2.5 Gbps per lane
 - ❑ Support CPHY 16-bit data width and two trios, up to 2.5 Gsps (symbol per second) per trio

The DWC_mipicsi2_hp_device supports the following added features in the standard controller:

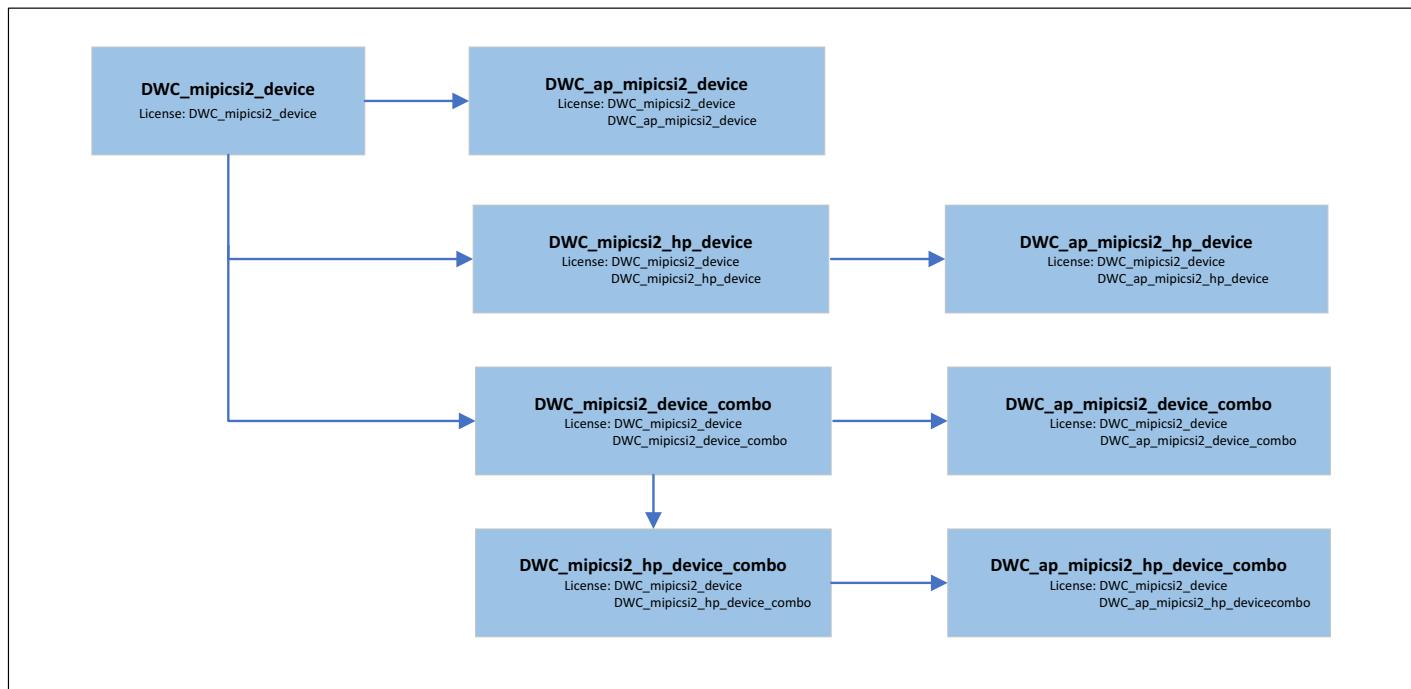
- D-PHY
 - ❑ Support 16-bit data width and up-to eight data lanes
 - ❑ Support 32-bit data width and up-to four data lanes
 - ❑ Up to 6.5 Gbps per lane
- C-PHY
 - ❑ Support 32-bit data width and up-to four trios
 - ❑ Up to 6.5 Gsps (symbol per second) per trio
- 128-bit Image Data Interface (IDI)
- 192-bit Image Pixel Interface (IPI)
- 128-bit Stream Data Interface (SDI)

The DWC_mipicsi2_hp_device_combo supports the following added features in the combo controller:

- Combo PHY 4L3T

- ❑ Support DPHY 16-bit/32-bit data width and four data lanes, up to 6.5 Gbps per lane
- ❑ Support CPHY 32-bit data width and three trios, up to 6.5 Gsps (symbol per second) per trio
- Combo PHY 2L2T
 - ❑ Support DPHY 16-bit/32-bit data width and two data lanes, up to 6.5 Gbps per lane
 - ❑ Support CPHY 32-bit data width and two trios, up to 6.5 Gsps (symbol per second) per trio
- 128-bit Image Data Interface (IDI)
- 192-bit Image Pixel Interface (IPI)
- 128-bit Stream Data Interface (SDI)
- The DWC_ap_mipicsi2_device, DWC_ap_mipicsi2_hp_device, DWC_ap_mipicsi2_device_combo, and DWC_ap_mipicsi2_hp_device_combo IPs are automotive controllers and support the following features:
 - ❑ Automotive support
 - ❑ Safety bus

Figure 1-2 CSI-2 Device Controller Products



1.2.1 Interfaces

Table 1-1 lists the DWC_mipicsi2_device interfaces and the functions.

Table 1-1 DWC_mipicsi2_device Interfaces

Interface	Function
Stream Data Interface (SDI)	<ul style="list-style-type: none"> ■ Receives the CSI-2 packet information from the system through a parallel bus with minimal latency. ■ Support packet format (Data on CSI-2 bus) and memory storage format (Data in receiver's buffer) in CSI2 spec.
Image Data Interface (IDI)	<ul style="list-style-type: none"> ■ Receives data from the sensor in 64-bit/128-bit packetized bus format, formatted according to CSI-2 Specification, with two/four words of 32-bit delivered per clock cycle. ■ Extracts packet header information from IDI, including Data Type, Virtual Channel, and Word Count.
Image Pixel Interface (IPI)	<ul style="list-style-type: none"> ■ Receives data from the sensor in 48-bit/192-bit pixel bus format, with one/four complete pixel provided per clock cycle. ■ Generates the Synchronous packet and data packet based on the IPI input. IPI reorders the received pixel to CSI-2 bus format of CSI-2 specification, based on data type.
AMBA APB Slave Bus	Used for configuration purposes.
D-PHY PPI	As recommended in the D-PHY specification.
C-PHY PPI	As recommended in the C-PHY specification.
ComboPHY mode (CSI2_DEVICE_PHY_TY PE=3)	This interface can be configured as either a D-PHY or C-PHY interface.

1.2.2 Supported Resolutions and Frame Rates

The CSI-2 specification does not define the supported standard resolutions or frame rates. Camera sensor resolution, blanking periods, synchronization events, frame rates, and pixel color depth play a fundamental role in the required bandwidth.

Table 1-2 presents some predefined and supported camera settings, assuming the following:

- D-PHY clock lane frequency is 500 MHz, 750 MHz, 1250 MHz or 2500 MHz, which results in a bandwidth of 1 Gbps, 1.5 Gbps, 2.5 Gbps, or 4.5 Gbps respectively, for each data lane.
- No significant reserved traffic is present on the link during pixel data transmission.

The last three columns of Table 1-2 show the minimum number of lanes required for each configuration.



The Table 1-2 is for reference purpose only, to evaluate the ability of the link to support the video format for DWC_mipicsi2_device.

Table 1-2 Supported Camera Settings

Mega Pixels	Number of Pixels with Overhead	Refresh Rate (Hz)	Color Depth (bpp)	CSI-2 Bandwidth (Mbits)	D-PHY at 1 Gbps Number of Lanes	D-PHY at 1.5 Gbps Number of Lanes	D-PHY at 2.5 Gbps Number of Lanes	C-PHY at 2.5 Gps (5.7 Gbps) Number of Lanes	D-PHY at 4.5Gbps Number of lanes	C-PHY at 3.5Gbps (8Gbps) Number of lanes
Camera Formats										
2 MP	2560000	15	24	922	1	1	1	1	1	1
3 MP	3840000	15	24	1382	2	1	1	1	1	1
5 MP	6400000	15	24	2304	3	2	1	1	1	1
8 MP	10240000	15	24	3686	4	3	2	1	1	1
12 MP	15360000	15	24	5530	6	4	3	1	2	1
13 MP	16640000	15	24	5990	6	4	3	2	2	1
16 MP	20480000	15	24	7373	8	5	3	2	2	1
20 MP	25600000	15	24	9216	Not enough	7	4	2	3	2
1280x720 pixels (720p)	921600	30	24	664	1	1	1	1	1	1
1280x720 pixels (720p)	921600	60	24	1327	2	1	1	1	1	1
1920x1080 pixels (1080p)	2073600	60	24	2986	3	2	2	1	1	1
3840x2160 pixels (2160p)	8294400	30	24	5972	6	4	3	2	2	1
3840x2160 pixels (2160p)	8294400	60	24	7963	8	6	4	2	2	1

According to CSI-2 protocol, a packet transmits a line data and the packet size should be the multiple of a byte. Therefore, for different data type, it limits the horizontal pixel number. The constraints of horizontal pixel number are as shown in [Table 1-3](#).

Table 1-3 Constraints of Horizontal Pixel Number

Data Type	Horizontal Pixel Number
RAW8/RAW16/RAW24/RGB444/RGB555/RGB565/RGB888	No
RAW12/RAW20/YUV422 8 bits/YUV422 10 bits/ Legacy YUV420 8 bits/YUV420 8 bits and CSPS	Multiple of 2
RAW6/RAW10/RAW14/RGB666/YUV420 10 bits and CSPS	Multiple of 4
RAW7	Multiple of 8

1.2.3 Unsupported Features and Exceptions

The unsupported features and exceptions are as follows:

- DWC_mipicsi2_device does not directly include support for the Camera Control Interface (CCI) as defined in the MIPI CSI-2 specification because this is an I2C-compliant control link. If a CCI/I2C-compliant link is required (in the absence of a pre-existing one), Synopsys DesignWare APB/I2C bridge (DW_apb_i2c) is recommended.
- I3C is recommended for CSI2 SPEC V2.0 and higher version.
- The DWC_mipicsi2_device does not perform data compression. For example, if a camera transmits the compressed data according to the Annex E of the CSI-2 Specification (data compression for RAW data types), the DWC_mipicsi2_device receives the data provided in the IDI interface as normal non-compressed RAW data.
- The DWC_mipicsi2_device controller only supports the D-PHY EPD option 2 under D-PHY 8-bit PPI mode and does not support it under D-PHY 16-bit/32-bit PPI mode.



Refer to SNPS PHY databook for a list of SNPS PHY supported/unsupported features.

1.3 Deliverables

The DWC_mipicsi2_device is packaged as a .run file. The license file, required to use this Synopsys DesignWare IP, is delivered separately. Use the Synopsys coreConsultant tool to configure, synthesize, and simulate the DWC_mipicsi2_device controller. The DWC_mipicsi2_device image includes the following:

- Verilog RTL source code
- A UVM-based testbench that can be configured to use a generic PHY model (D-PHY or Combo PHY) or a Synopsys Verilog model
- Synthesis scripts for Synopsys Design Compiler and Synplify Pro
- Regression scripts for the simulator: Synopsys VCS
- SpyGlass checker rules used for linting and CDC
- Gate-level validation scripts for:
 - Formal Verification (using Formality)
 - Static timing analyses (using PrimeTime)
 - ATPG patterns verification (using Tetramax)
- DWC_mipicsi2_device Databook (this document), User Guide, Installation Guide, and Release Notes

1.4 Operational Model Overview

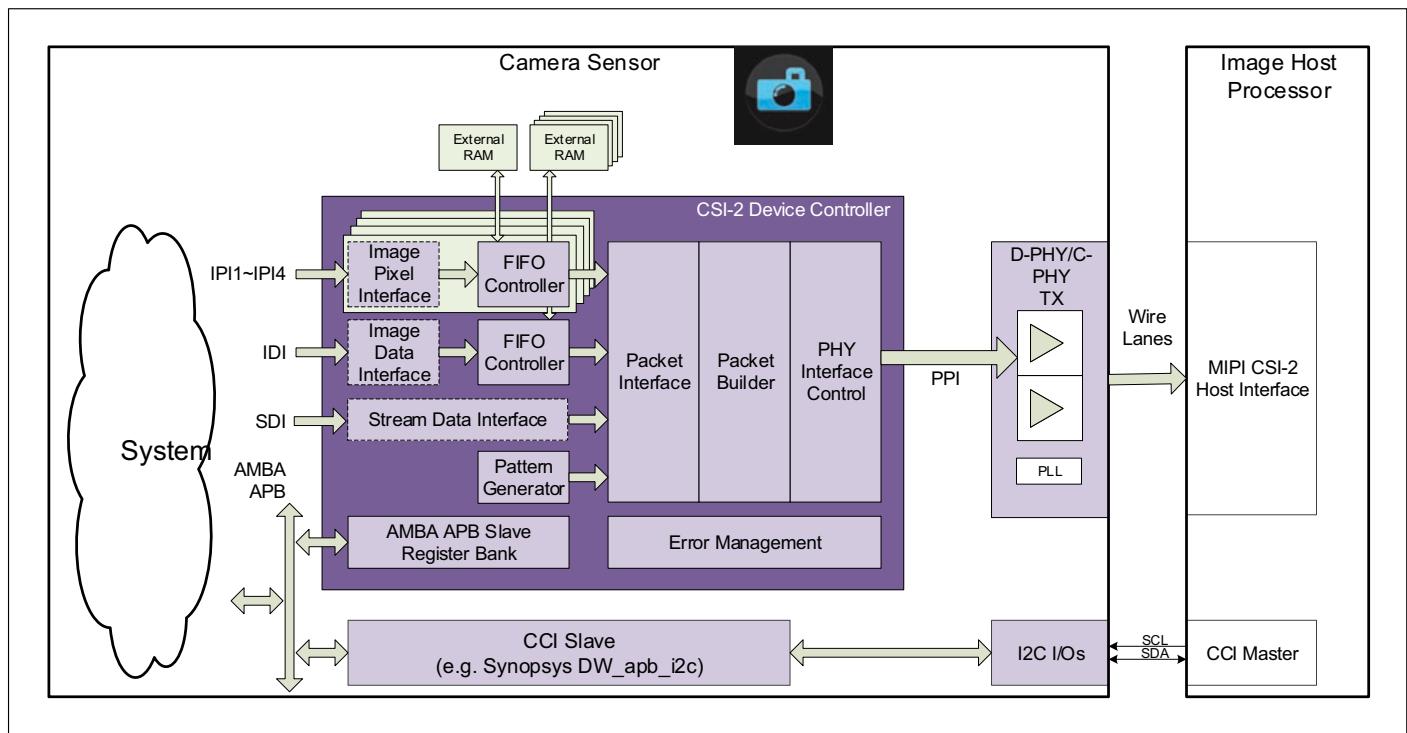
The DWC_mipicsi2_device is a digital controller that implements all protocol functions defined in the MIPI CSI-2 Specification, providing an interface between the system and the MIPI D-PHY, allowing the communication with a MIPI CSI-2 compliant host processor.

The DWC_mipicsi2_device provides the means for seamless integration with Synopsys D-PHYs through coreConsultant. Optionally, you can configure the controller for a non-Synopsys D-PHY. In such a configuration, it exhibits a PPI-compliant interface to connect to D-PHY and/or C-PHY.

[Table 1-4](#) lists the main functional blocks of the controller.

Table 1-4 DWC_mipicsi2_device Functional Blocks

Functional Block	Description
Image Data Interface (IDI)	Provides a system interface to receive packed data format in the recommended memory storage format.
Image Pixel Interface (IPI)	Provides a system interface to receive unpacked pixels and timing accurate video synchronization signals.
Stream Data Interface (SDI)	Provides a system interface to receive the CSI-2 packet information through a parallel bus with minimal latency.
Video Pattern Generator	Provides auto-generated test pattern according to the register bank configuration.
Packet Interface	Receives image header and payload from multiple image source.
Packet Builder	Builds CSI-2 protocol packets.
PHY Interface Control	Controls D-PHY or C-PHY to transmit CSI-2 protocol packets.
AMBA-APB Register Bank	Provides access to configuration and control registers.

Figure 1-3 Functional Block Diagram of DWC_mipicsi2_device

1.5 Speed and Clock Requirements

The register bank relies on the AMBA APB `pclk` clock, which should be a free-running clock.

[Table 1-5](#) shows the clock frequency of `DWC_mipicsi2_device`.

Table 1-5 DWC_mipicsi2_device Clock Frequency

Clock		Frequency Range					
		Minimum Data Rate/Frequency	Maximum Data Rate/Frequency		Generic 16nm library	Simulation range	
			Generic 28nm library	Generic 7nm library			
lanebyteclk/ sdi_clk	DPHY	8-bit PPI	80Mbps/10Mhz	2.5Gbps/312.5Mhz	4.5Gbps/562.5Mhz	80Mbps ~ 6.5Gbps	
		16-bit PPI	160Mbps/10Mhz	4.5Gbps/281.3Mhz	6.5Gbps/406.3Mhz	80Mbps ~ 6.5Gbps	
		32-bit PPI	320Mbps/10Mhz	4.5Gbps/140.7Mhz	6.5Gbps/203.2Mhz	80Mbps ~ 6.5Gbps	
	CPHY	16-bit PPI	80Msps/11.4Mhz	2.5Gsps/357.2Mhz	3.5Gsps/500Mhz	80Msps ~ 6.5Gsps	
		32-bit PPI	160Msps/11.4Mhz	3.5Gsps/250Mhz	6.5Gsps/464.2Mhz	80Msps ~ 6.5Gsps	
pclk		15MHz	250Mhz	250Mhz	5MHz ~ 500Mhz		
idi_clk		10MHz	400Mhz	600Mhz	10MHz ~ 1Ghz		
ipi_clk		10MHz	400Mhz	600Mhz	10MHz ~ 1Ghz		
txclkes		5MHz	20MHz	1.25MHz ~ 20Mhz			
ssm_pclk		15MHz	250MHz	250Mhz	5Mhz ~ 500Mhz		



1. Minimum Frequency of lanebyteclk and txclkes are limited by the solution of CSI2 device controller and SNPS CDPHY TX.
2. Maximum Frequency of txclkes is limited by MIPI D-PHY/C-PHY Specification
3. Maximum Frequency of pclk, idi_clk and ipi_clk is the default value and depends on the selected technology library and configuration.
4. All the frequencies of Simulation range are tested in simulation.

2

Architecture

This chapter describes the DWC_mipicsi2_device architecture. It contains the following sections:

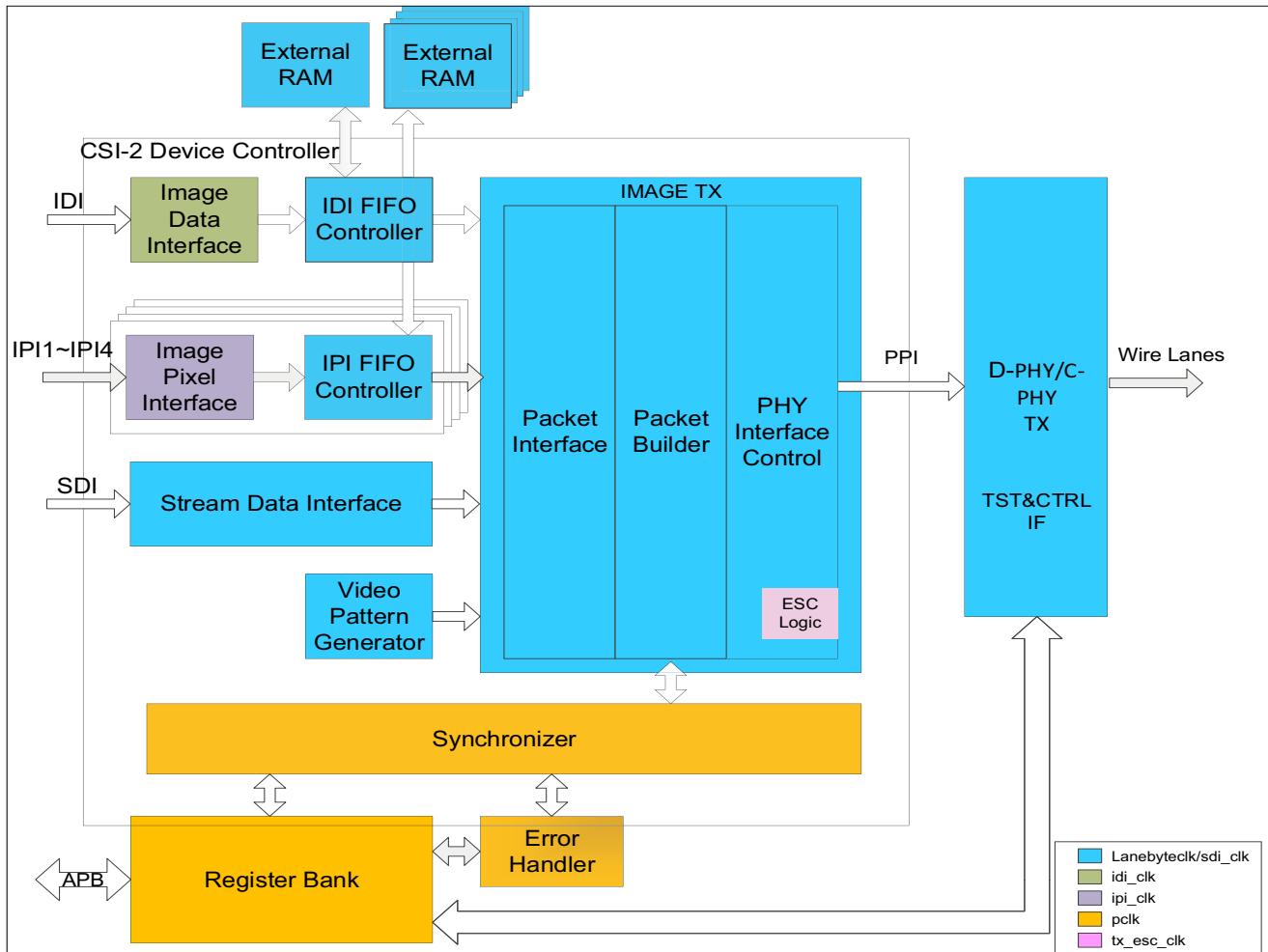
- “Overview of Architecture” on page 31
- “Image Data Interface” on page 33
 - “Overview of Image Data Interface” on page 33
 - “Description of Image Data Interface” on page 33
 - “Enabling IDI” on page 46
 - “Registers related to IDI” on page 46
 - “Signals related to IDI” on page 47
- “Image Pixel Interface” on page 48
 - “Overview of Image Pixel Interface” on page 48
 - “Color Components Mapping” on page 48
 - “IPI Store and Forward Mode” on page 50
 - “IPI Cut-Through Mode” on page 52
 - “IPI Line Synchronization Omission Mode” on page 54
 - “IPI Backpressure Mode” on page 55
 - “IPI Embedded Data” on page 57
 - “Enabling IPI” on page 58
 - “Registers Related to IPI” on page 58
 - “Signals Related to IPI” on page 58
 - “Programming IPI” on page 59
- “Multiple IPI Support” on page 60
 - “Multiple IPI with No Timing Delivery Mode” on page 60
 - “Multiple IPI with Timing Delivery Mode” on page 60
 - “Enabling Multiple IPI” on page 68
 - “Registers Related to Multiple IPI” on page 68
 - “Programming Multiple IPI” on page 69

- “Simultaneous IDI and IPI Mode” on page 70
 - “Overview of Simultaneous IDI and IPI Mode” on page 70
 - “Insertion of IDI Packets in IPI Transmission” on page 70
 - “Time Calculation of IDI Insertion” on page 72
 - “Error Scenario” on page 75
 - “Enabling Simultaneous IDI and IPI Mode” on page 76
 - “Registers Related to Simultaneous IDI and IPI Mode” on page 76v
 - “Programming Simultaneous IDI and IPI Mode” on page 76
- “Stream Data Interface” on page 77
 - “Overview of Stream Data Interface” on page 77
 - “Description of Stream Data Interface” on page 77
 - “Registers Related to SDI” on page 83
 - “Signals Related to SDI” on page 83
 - “Programming SDI” on page 84
- “Video Pattern Generator” on page 85
 - “Overview of the Video Pattern Generator” on page 85
 - “Color Bar Pattern” on page 85
 - “Color Coding” on page 86
 - “BER Testing Pattern” on page 88
 - “Video Pattern Generator Resolution” on page 89
 - “Enabling Video Pattern Generator” on page 90
 - “Registers Related to Video Pattern Generator” on page 90
- “Memory” on page 91
 - “Image Data Interface (IDI) Minimum Memory Size” on page 92
 - “Image Pixel Interface (IPI) Memory Size” on page 92
 - “Memory Size for Automotive Safety” on page 96
- “Protocol-PHY Interface (PPI)” on page 97
- “LRTE Support” on page 98
 - “Inter-packet Latency Reduction (ILR)” on page 98
 - “Enhanced Transport Efficiency” on page 103
- “Data Scrambling” on page 105
 - “Overview of Data Scrambling” on page 105
 - “Enabling the Scrambler” on page 107
 - “Related Registers” on page 108
- “Interrupts” on page 109
- “Error Detection” on page 111

2.1 Overview of Architecture

Figure 2-1 shows the overall architecture of the DWC_mipicsi2_device.

Figure 2-1 **DWC_mipicsi2_device Architecture**



The main blocks of DWC_mipicsi2_device are as follows:

- **Image Data Interface (IDI)**: Provides a system interface to receive 64-bit packed data format as the recommended memory storage format.
- **Image Pixel Interface (IPI)**: Provides a system interface to receive unpacked pixels, embedded non-image data, and timing accurate video synchronization signals.
- **Stream Data Interface (SDI)**: Provides a system interface to receive the CSI-2 packet information through a parallel bus with minimal latency.
- **FIFO controller**: Controls storage and retrieval of the header FIFO and payload FIFO.
- **Video Pattern Generator**: Provides auto-generated test pattern according to the register bank configuration.
- **Image Tx**: Contains the following blocks:

- **Packet Interface:** Receives image header and payload from multiple image source, VPG, or IPI/IDI/SDI, and transmits to Packet Builder.
- **Packet Builder:** Receives the packet information from the Packet Interface module and combines them into CSI-2 protocol packets, and transmits them to the phy_if_ctrl module.
- **PHY Interface Control:** Receives the packet data from the Packet Builder module and controls D-PHY or C-PHY to transmit CSI2 protocol packets, also including D-PHY or C-PHY UPLS control.
- **Error Handler:** Manages all interrupts from the controller.
- **AMBA-APB Register Bank:** Provides access to configuration and control registers.



DWC_mipicsi2_device does not use tx_esc_clk. Use this divided clock as enable.

2.2 Image Data Interface

This section describes the Image Data Interface (IDI) of DWC_mipicsi2_device.

2.2.1 Overview of Image Data Interface

The IDI is a Synopsys proprietary interface used to receive the CSI-2 packet information from the system through a parallel bus. The interface receives the packet information related to virtual channel, data type, word count, and data.

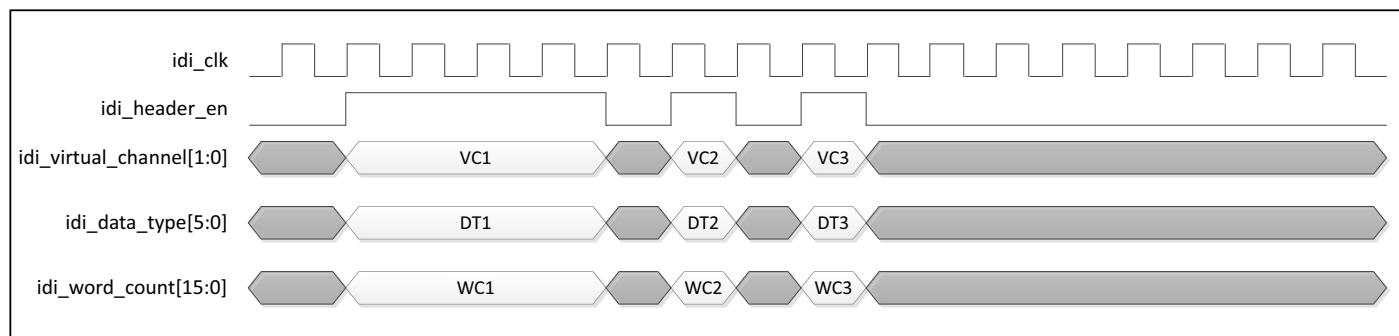
2.2.2 Description of Image Data Interface

The packet payload information is received in a 64-bit data memory organization and follows data format template defined in the CSI-2 specification.

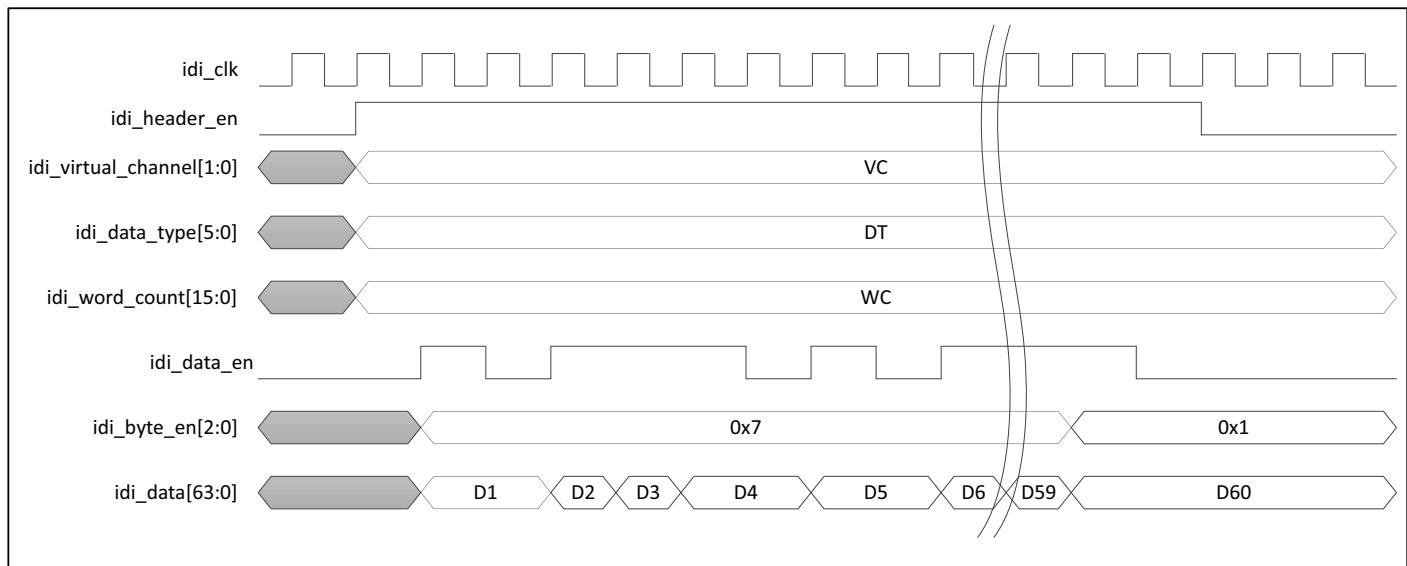
The signal `idi_header_en` indicates that the new data is available. This signal rises when a new packet is sampled on `idi_virtual_channel`, `idi_data_type`, and `idi_word_count` fields and falls at the end of the packet transmission. During the `idi_header_en` assert, `idi_virtual_channel`, `idi_datatype`, and `idi_word_count` fields must remain unchanged.

For a short packet, there is no restriction on the length of `idi_header_en`. [Figure 2-2](#) shows an example of transferring short packets. The rising edge of `idi_header_en` captures the fields of `idi_virtual_channel`, `idi_data_type`, and `idi_word_count` and dispatches at the falling edge.

Figure 2-2 Image Data Interface Short Packet Timing



The long packet transmission requires at least one `idi_clk` cycle between `idi_header_en` and `idi_data_en`. To indicate the completion of packet transmission, `idi_header_en` is de-asserted. The `idi_byte_en` indicates the number of valid bytes of `idi_data`. [Figure 2-3](#) shows an example of long packet transmission.

Figure 2-3 Image Data Interface Long Packet Timing

In the generic case and for arbitrary data, the first byte of payload data transmitted maps to the least significant byte of the 64-bit memory word, and the eighth byte of payload data transmitted maps to the most significant byte of the 64-bit memory word.

[Figure 2-4](#) shows the generic CSI-2 byte to 64-bit memory word mapping rule.

Figure 2-4 Generic/Arbitrary Data Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
General / Arbitrary data	-	-	-	BYTE8[7:0]				BYTE7[7:0]				BYTE6[7:0]				BYTE5[7:0]				BYTE4[7:0]				BYTE3[7:0]				BYTE2[7:0]				BYTE1[7:0]																																			

2.2.2.1 RGB888 Data Reception

The RGB888 data format byte to 64-bit memory word mapping follows the generic CSI-2 rule.

Figure 2-5 RGB888 Data Format Reception

Video Format				ID1_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGB888	8	-	1	G3				B3				R2				G2				B2				R1				G1				B1																																			
		-	2	B6				R5				G5				B5				R4				G4				B4				R3																																			
		-	3	R8				G8				B8				R7				G7				B7				R6				G6																																			

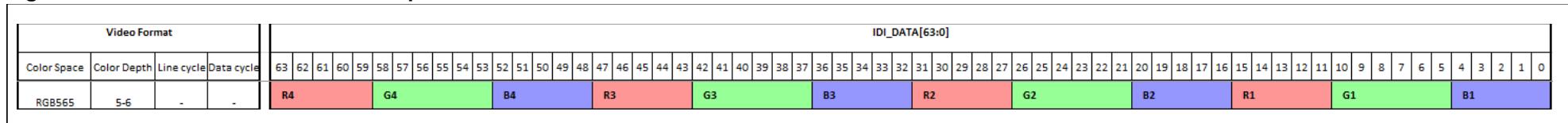
2.2.2.2 RGB666 Data Reception

Figure 2-6 RGB666 Data Format Reception

Video Format				ID1_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGB666	6	-	1	G4[3:0]	B4			R3			G3			B3			R2			G2			B2			R1			G1			B1			G4[5:4]																																
		-	2	B8[1:0]	R7			G7			B7			R6			G6			B6			R5			G5			B5			R4			G4[5:4]																																
		-	3	G11			B11			R10			G10			B10			R9			G9			B9			R8			G8			B8[5:2]																																	

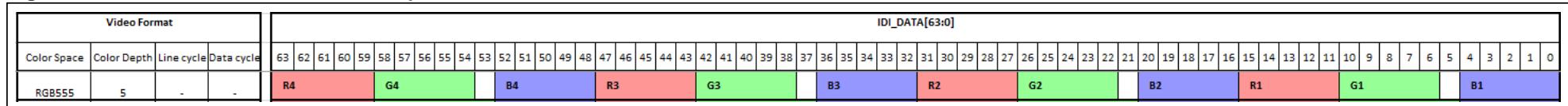
2.2.2.3 RGB565 Data Reception

Figure 2-7 RGB565 Data Format Reception



2.2.2.4 RGB555 Data Reception

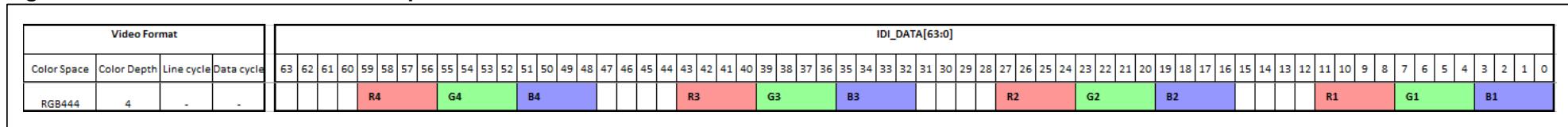
Figure 2-8 RGB555 Data Format Reception



2.2.2.5 RGB444 Data Reception

The RGB444 data format byte to 64-bit memory word mapping has a special transform as shown in [Figure 2-9](#):

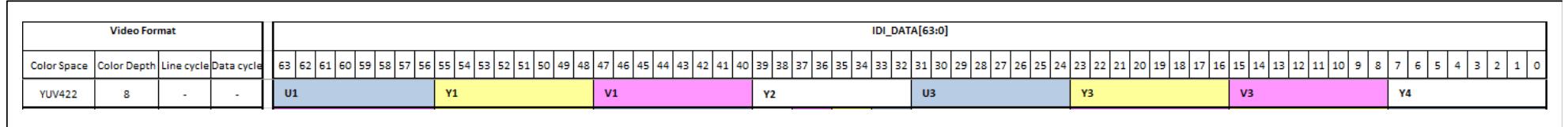
Figure 2-9 RGB444 Data Format Reception



2.2.2.6 YUV422 8-Bit Data Reception

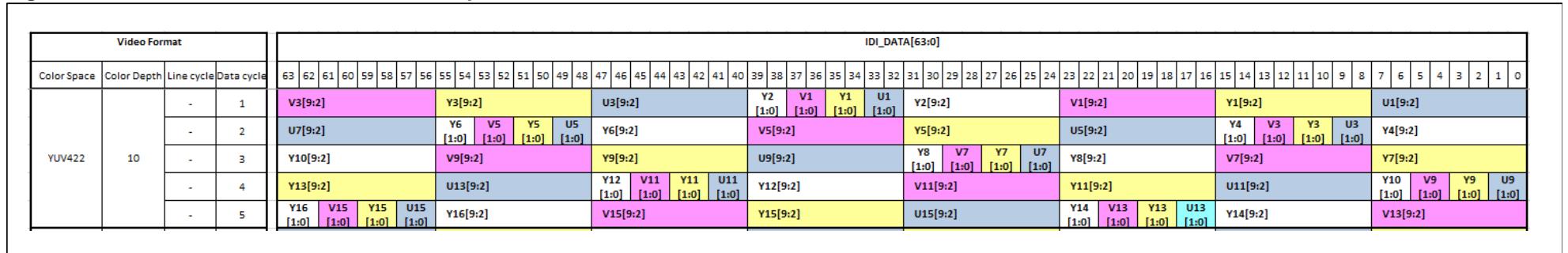
The YUV422 8-bit data format byte to 64-bit memory word mapping does not follow the generic CSI-2 rule:

- The first byte of payload data transmitted maps to the most significant byte of the 64-bit memory word.
- The eighth byte of payload data transmitted maps to the least significant byte of the 64-bit memory word.

Figure 2-10 YUV422 8-Bit Data Format Reception

2.2.2.7 YUV422 10-Bit Data Reception

The YUV422 10-bit data format byte to 64-bit memory word mapping follows the generic CSI-2 rule.

Figure 2-11 YUV422 10-Bit Data Format Reception

2.2.2.8 YUV420 8-Bit (Legacy) Data Reception

The YUV420 8-bit (legacy) data format byte to 64-bit memory word mapping does not follow the generic CSI-2 rule:

- The first byte of payload data transmitted maps to the most significant byte of the 64-bit memory word.
- The eighth byte of payload data transmitted maps to the least significant byte of the 64-bit memory word.

Figure 2-12 YUV420 8-Bit (Legacy) Data Format Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
YUV420 legacy	8	odd line	1	U1				Y1				Y2				U3				Y3				Y4				U5				Y5																																			
			2	Y6				U7				Y7				Y8				U9				Y9				Y10				U11																																			
			3	Y11				Y12				U13				Y13				Y14				U15				Y15				Y16																																			
	8	even line	1	V1				Y1				Y2				V3				Y3				Y4				V5				Y5																																			
			2	Y6				V7				Y7				Y8				V9				Y9				Y10				V11																																			
			3	Y11				Y12				V13				Y13				Y14				V15				Y15				Y16																																			

2.2.2.9 YUV420 10-Bit Data Reception

The YUV420 10-bit data format byte to 64-bit memory word mapping follows the generic CSI-2 rule.

Figure 2-13 YUV420 10-Bit Data Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
YUV420	8	odd line	-	Y8				Y7				Y6				Y5				Y4				Y3				Y2				Y1																																			
		even line	-	Y4				V3				Y3				U3				Y2				V1				Y1				U1																																			
	10	odd line	1	Y7[9:2]				Y6[9:2]				Y5[9:2]				Y4 [1:0]		Y3 [1:0]		Y2 [1:0]		Y1 [1:0]		Y4[9:2]				Y3[9:2]				Y2[9:2]				Y1[9:2]																															
			2	Y13[9:2]				Y12 [1:0]		Y11 [1:0]		Y10 [1:0]		Y9 [1:0]		Y12[9:2]				Y11[9:2]				Y10[9:2]				Y9[9:2]				Y8 [1:0]		Y7 [1:0]		Y6 [1:0]		Y5 [1:0]		Y8[9:2]																											
			3	Y20[9:2]				Y19[9:2]				Y18[9:2]				Y17[9:2]				Y16 [1:0]		Y15 [1:0]		Y14 [1:0]		Y13 [1:0]		Y16[9:2]				Y15[9:2]				Y14[9:2]																															
			4	Y26[9:2]				Y25[9:2]				Y24 [1:0]		Y23 [1:0]		Y22 [1:0]		Y24[9:2]				Y23[9:2]				Y22[9:2]				Y21[9:2]				Y20 [1:0]		Y19 [1:0]		Y18 [1:0]		Y17 [1:0]																											
			5	Y32 [1:0]		Y32[9:2]				Y31[9:2]				Y30[9:2]				Y29[9:2]				Y28 [1:0]		Y27 [1:0]		Y26 [1:0]		Y25 [1:0]		Y28[9:2]				Y27[9:2]																																	
	10	even line	1	V3[9:2]				Y3[9:2]				U3[9:2]				Y2 [1:0]		V1 [1:0]		Y1 [1:0]		U1 [1:0]		Y2[9:2]				V1[9:2]				Y1[9:2]				U1[9:2]																															
			2	U7[9:2]				Y6 [1:0]		V5 [1:0]		Y5 [1:0]		U5 [1:0]		Y6[9:2]				V5[9:2]				Y5[9:2]				U5[9:2]				Y4 [1:0]		V3 [1:0]		Y3 [1:0]		U3 [1:0]		Y4[9:2]																											
			3	Y10[9:2]				V9[9:2]				Y9[9:2]				U9[9:2]				Y8[9:2]				V8[9:2]				V7[9:2]				Y7[9:2]																																			
			4	Y13[9:2]				U13[9:2]				Y12 [1:0]		V11 [1:0]		Y11 [1:0]		Y12[9:2]				V11[9:2]				Y11[9:2]				U11[9:2]				Y10 [1:0]		V9 [1:0]		Y9 [1:0]		U9 [1:0]																											
			5	Y16 [1:0]		V15 [1:0]		Y15 [1:0]		Y16[9:2]				V15[9:2]				Y15[9:2]				U15[9:2]				Y14[9:2]				V13[9:2]																																					

2.2.2.10 RAW6 Data Reception

Figure 2-14 RAW6 Data Format Reception

2.2.2.11 RAW7 Data Reception

Figure 2-15 RAW7 Data Format Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW7	7	-	1	P1[0]	P9[6:0]				P8[6:0]				P7[6:0]				P6[6:0]				P5[6:0]				P4[6:0]				P3[6:0]				P2[6:0]				P1[6:0]																														
		-	2	P19[1:0]	P18[6:0]				P17[6:0]				P16[6:0]				P15[6:0]				P14[6:0]				P13[6:0]				P12[6:0]				P11[6:0]				P10[6:1]																														
		-	3	P28[2:0]				P27[6:0]				P26[6:0]				P25[6:0]				P24[6:0]				P23[6:0]				P22[6:0]				P21[6:0]				P20[6:0]				P19[6:2]																											
		-	4	P37[3:0]				P36[6:0]				P35[6:0]				P34[6:0]				P33[6:0]				P32[6:0]				P31[6:0]				P30[6:0]				P29[6:0]				P28[6:3]																											
		-	5	P46[4:0]				P45[6:0]				P44[6:0]				P43[6:0]				P42[6:0]				P41[6:0]				P40[6:0]				P39[6:0]				P38[6:0]				P37[6:4]																											
		-	6	P55[5:0]				P54[6:0]				P53[6:0]				P52[6:0]				P51[6:0]				P50[6:0]				P49[6:0]				P48[6:0]				P47[6:0]				P46[6:5]																											
		-	7	P64[6:0]				P63[6:0]				P62[6:0]				P61[6:0]				P60[6:0]				P59[6:0]				P58[6:0]				P57[6:0]				P56[6:0]				P55[5]																											

2.2.2.12 RAW8 Data Reception

The RAW8 data format byte to 64-bit memory word mapping follows the generic CSI-2 rule.

Figure 2-16 RAW8 Data Format Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW8	8	-	-	P8[7:0]				P7[7:0]				P6[7:0]				P5[7:0]				P4[7:0]				P3[7:0]				P2[7:0]				P1[7:0]																																			

2.2.2.13 RAW10 Data Reception

The RAW10 data format byte to 64-bit memory word mapping follows the generic CSI-2 rule.

Figure 2-17 RAW10 Data Format Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW10	10	-	1	P7[9:2]				P6[9:2]				P5[9:2]				P4 [1:0]	P3 [1:0]	P2 [1:0]	P1 [1:0]	P4[9:2]				P3[9:2]				P2[9:2]				P1[9:2]																																			
		-	2	P13[9:2]				P12 [1:0]	P11 [1:0]	P10 [1:0]	P9 [1:0]	P12[9:2]				P11[9:2]				P10[9:2]				P9[9:2]				P8 [1:0]	P7 [1:0]	P6 [1:0]	P5 [1:0]	P8[9:2]																																			
		-	3	P20[9:2]				P19[9:2]				P18[9:2]				P17[9:2]				P16 [1:0]	P15 [1:0]	P14 [1:0]	P13 [1:0]	P16[9:2]				P15[9:2]				P14[9:2]																																			
		-	4	P26[9:2]				P25[9:2]				P24 [1:0]	P23 [1:0]	P22 [1:0]	P21 [1:0]	P24[9:2]				P23[9:2]				P22[9:2]				P21[9:2]				P20 [1:0]	P19 [1:0]	P18 [1:0]	P17 [1:0]	P20[9:2]																															
		-	5	P32 [1:0]	P31 [1:0]	P30 [1:0]	P29 [1:0]	P32[9:2]				P31[9:2]				P30[9:2]				P29[9:2]				P28 [1:0]	P27 [1:0]	P26 [1:0]	P25 [1:0]	P28[9:2]				P27[9:2]																																			

2.2.2.14 RAW12 Data Reception

The RAW12 data format byte to 64-bit memory word mapping follows the generic CSI-2 rule.

Figure 2-18 RAW12 Data Format Reception

2.2.2.15 RAW14 Data Reception

Figure 2-19 RAW14 Data Format Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line cycle	Data cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW14	14	-	1	P5[13:6]					P4[5:0]			P3[5:0]			P2[5:0]			P1[5:0]			P4[13:6]					P3[13:6]			P2[13:6]			P1[13:6]																																			
		-	2	P10[13:6]					P9[13:6]					P8[5:0]			P7[5:0]			P6[5:0]			P5[5:0]			P8[13:6]					P7[13:6]			P6[13:6]																																	
		-	3	P15[13:6]					P14[13:6]					P13[13:6]					P12[5:0]			P11[5:0]			P10[5:0]			P9[5:0]			P12[13:6]					P11[13:6]																															
		-	4	P20[13:6]					P19[13:6]					P18[13:6]					P17[13:6]					P16[5:0]			P15[5:0]			P14[5:0]			P13[5:0]			P16[13:6]																															
		-	5	P22[1:0]			P21[5:0]			P24[13:6]					P23[13:6]					P22[13:6]					P21[13:6]			P20[5:0]			P19[5:0]			P18[5:0]			P17[5:0]																														
		-	6	P27[3:0]			P26[5:0]			P25[5:0]			P28[13:6]					P27[13:6]					P26[13:6]			P25[13:6]			P24[5:0]			P23[5:0]			P22[5:2]																																
		-	7	P32[5:0]					P31[5:0]					P30[5:0]			P29[5:0]			P32[13:6]					P31[13:6]			P30[13:6]			P29[13:6]			P28[5:0]			P27[5:4]																														

2.2.2.16 RAW16 Data Reception

Figure 2-20 RAW16 Data Format Reception

2.2.2.17 RAW20 Data Reception

Figure 2-21 RAW20 Data Format Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line Cycle	Data Cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW/20	20	1		P4[19:12]				P3[9:2]				P3[19:12]				P2[1:0]				P1[1:0]				P2[9:2]				P2[19:12]				P1[9:2]				P1[19:12]																															
		2		P7[19:12]				P4[1:0]				P4[11:10]				P5[1:0]				P6[9:2]				P6[19:12]				P5[9:2]				P5[19:12]				P4[1:0]				P4[11:10]				P3[11:10]				P4[9:2]																			
		3		P10[9:2]				P10[19:12]				P9[9:2]				P9[19:12]				P8[1:0]				P8[11:10]				P7[19:12]				P8[9:2]				P8[19:12]				P7[9:2]																											
		4		P13[9:2]				P13[19:12]				P12[1:0]				P12[11:10]				P11[1:0]				P12[9:2]				P12[19:12]				P11[9:2]				P11[19:12]				P10[1:0]				P10[11:10]				P9[11:10]																			
		5		P16[1:0]				P16[11:10]				P15[1:0]				P16[9:2]				P16[19:12]				P15[9:2]				P15[19:12]				P14[1:0]				P14[11:10]				P13[1:0]				P13[11:10]				P14[9:2]				P14[19:12]															

2.2.2.18 RAW24 Data Reception

Figure 2-22 RAW24 Data Format Reception

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line Cycle	Data Cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW24	24		1	P3[11:4]				P3[23:16]				P2[3:0]				P2[15:12]				P2[11:4]				P2[23:16]				P1[3:0]		P1[15:12]		P1[11:4]				P1[23:16]																															
			2	P6[23:16]				P5[3:0]		P5[15:12]		P5[11:4]				P5[23:16]				P4[3:0]		P4[15:12]		P4[11:4]				P4[23:16]				P3[3:0]		P3[15:12]																																	
			3	P8[3:0]		P8[15:12]		P8[11:4]				P8[23:16]				P7[3:0]		P7[15:12]		P7[11:4]				P7[23:16]				P6[3:0]		P6[15:12]		P6[11:4]																																			

2.2.2.19 IDI 128-Bit Memory Word Mapping

Except YUV420 8-Bit legacy and YUV422 8-Bit, for all other data types, the mapping at each 64-bit of the 128-bit IDI is the same as the mapping of the 64-bit IDI.

For example, mapping of RAW8 data type at 128-bit IDI is as follows:

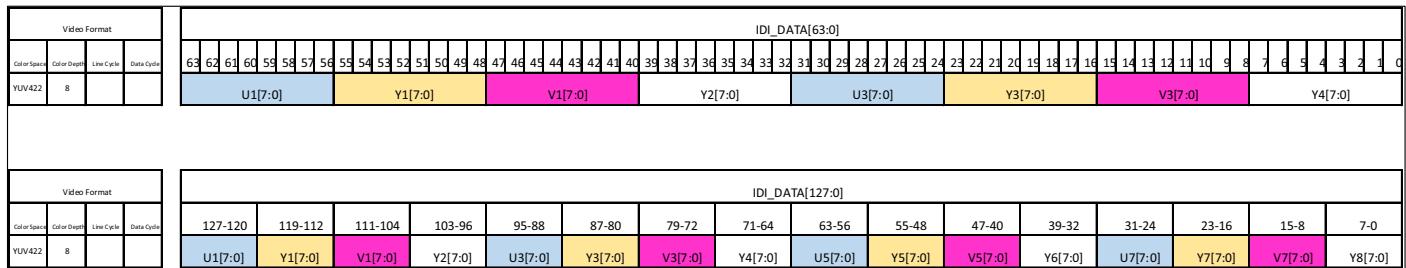
Figure 2-23 RAW8 data mapping at 128-bit IDI

Video Format				IDI_DATA[63:0]																																																															
Color Space	Color Depth	Line Cycle	Data Cycle	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW 8	8			P8[7:0]				P7[7:0]				P6[7:0]				P5[7:0]				P4[7:0]				P3[7:0]				P2[7:0]				P1[7:0]																																			

The mapping for YUV422 8-Bit is as follows:

YUV420 8-Bit legacy is also similar.

Figure 2-24 YUV422 8-Bit and YUV420 8-Bit Legacy Data Mapping at 128-bit IDI



2.2.2.20 Valid Bytes of CSI-2 Packet

A CSI-2 packet size is always an integer number of bytes, but that number is not necessarily a multiple of eight. The `idi_data` signal is 64-bit wide therefore, part of the bytes may not contain valid packet data at the end of a packet. The `idi_byte_en[2:0]` signal indicates the number of bytes that are valid in the `idi_data` input signal as shown in [Table 2-1](#).

Table 2-1 Valid idi_data Bits According to idi_byte_en

idi_byte_en [2:0]	[63:56]	[48:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
YUV420 8-Bit Legacy and YUV422 8-Bit DT								
3'b000	Valid	Don't care						
3'b001	Valid	Valid	Don't care					
3'b010	Valid	Valid	Valid	Don't care				
3'b011	Valid	Valid	Valid	Valid	Don't care	Don't care	Don't care	Don't care
3'b100	Valid	Valid	Valid	Valid	Valid	Don't care	Don't care	Don't care
3'b101	Valid	Valid	Valid	Valid	Valid	Valid	Don't care	Don't care
3'b110	Valid	Don't care						
3'b111	Valid							
All Other Data Types								
3'b000	Don't care	Valid						
3'b001	Don't care	Valid	Valid					
3'b010	Don't care	Valid	Valid	Valid				
3'b011	Don't care	Don't care	Don't care	Don't care	Valid	Valid	Valid	Valid
3'b100	Don't care	Don't care	Don't care	Valid	Valid	Valid	Valid	Valid
3'b101	Don't care	Don't care	Valid	Valid	Valid	Valid	Valid	Valid
3'b110	Don't care	Valid						
3'b111	Valid							

During the payload data transfer, the idi_byte_en signal is at 3'b111 to indicate that the eight bytes in the idi_data are valid bytes from the packet payload. When the last word is available, idi_byte_en changes according to the number of remaining valid bytes. For all data types, the Least Significant (LS) bytes contain the remaining bytes. The exceptions are data types YUV420 8-bit legacy and YUV422 8-bit DT that are of the reverse order as shown in [Table 2-1](#).

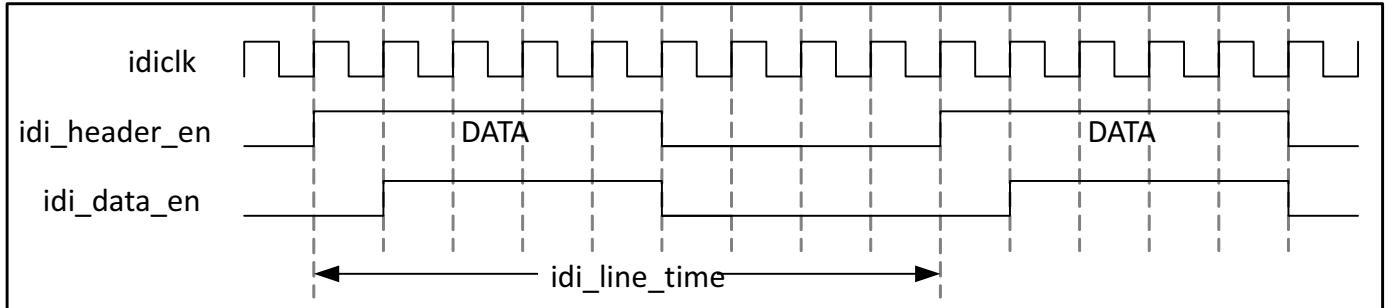
2.2.2.21 IDI Driver Timing Requirement

When not using the Backpressure feature, the PPI throughput must be greater than the IDI throughput to avoid FIFO overflow. The throughputs vary according to the line synchronization packet mode.

Line synchronization packet mode disabled

In this mode, the time between the beginnings of two data packets defines the idิ_line_time.

Figure 2-25 IDI Line Time Definition when Line synchronization Packet Mode is Disabled

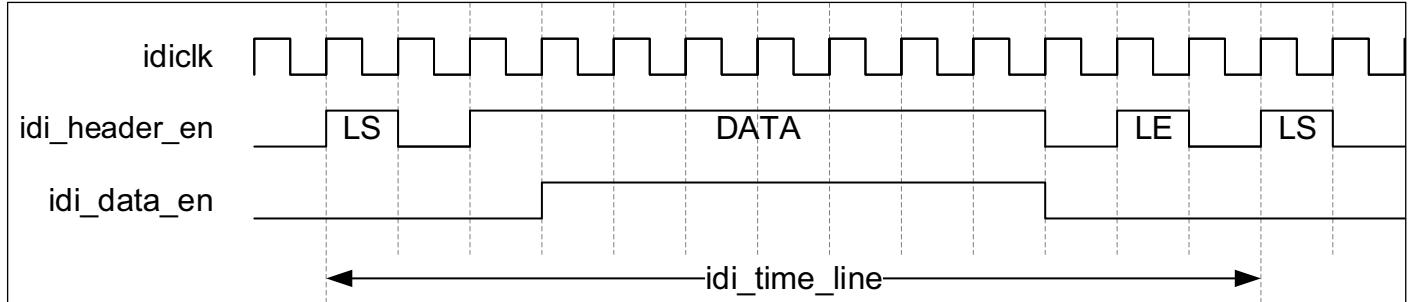


The following formula defines the timing: $\text{idи_hline_time} > \text{LP_trans_time}$

Line synchronization packet mode enabled

In this mode, the time between the start of two-line start packets ($\text{idи_data_type} = 6'h02$) defines the idи_line_time.

Figure 2-26 IDI Line Time Definition when Line Synchronization Packet Mode is Enabled

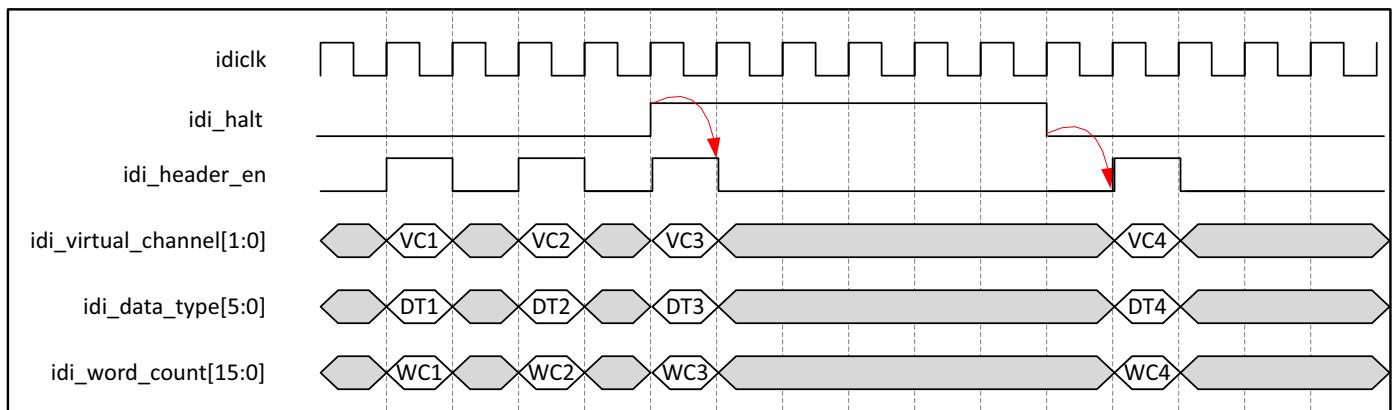
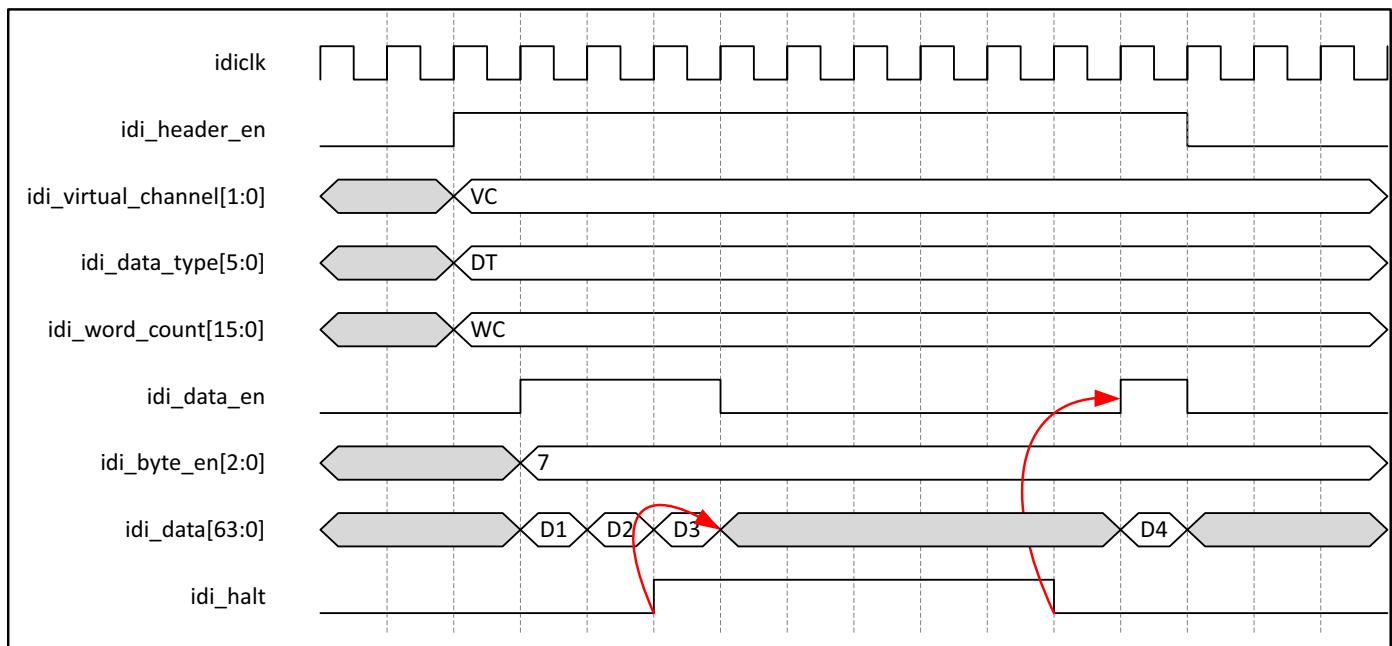


The following formula defines the timing: $\text{idи_hline_time} > \text{LP_trans_time} + 2 * \text{SP_trans_time}$
For more information about SP_trans_time and LP_trans_time calculation, see [DWC_mipicsi2_device Packet Transmission Calculation](#) on page 735.

2.2.2.22 IDI Backpressure

DWC_mipicsi2_device provides a Backpressure mechanism to inform the IDI driver that the memory is almost full. The idи_halt output signal rises each time either the header or the payload FIFO can accept just one more incoming data.

Figure 2-27 and Figure 2-28 show the idи_halt behavior for a packet header and packet payload reception.

Figure 2-27 Header FIFO Becomes Full Causing idi_halt Assertion**Figure 2-28 Packet Payload FIFO Becomes Full Causing idi_halt Assertion**

2.2.3 Enabling IDI

To enable IDI, select “IDI” from the Data Interface > Select System Interface drop-down option in the Basic Configuration window in the coreConsultant GUI. You can also enable 64-bit data interface option, if required.

For more information, see the *Parameter Descriptions* chapter.

2.2.4 Registers related to IDI

The following registers are related to IDI:

- INT_ST_IDI

- INT_MASK_N_IDI
- INT_FORCE_IDI
- IDI_FIFO_STATUS

For more information, see the *Parameter Descriptions* chapter.

2.2.5 Signals related to IDI

For more information, see the “IDI Interface Signals” section in the *Signal Descriptions* chapter.

2.3 Image Pixel Interface

This section describes the Image Pixel Interface (IPI) of the DWC_mipicsi2_device.

2.3.1 Overview of Image Pixel Interface

The IPI is a Synopsys proprietary interface used to receive the pixel information through a parallel bus from the system. The Image Pixel Interface is 48-bit pixel bus, with one complete pixel provided per clock cycle, transmitted using one programmed Virtual Channel.

2.3.2 Color Components Mapping

The pixel color coding at the 48-bit interface is shown in [Figure 2-29](#) and the [Table 2-2](#) lists the different types of data input.

Table 2-2 Data Inputs

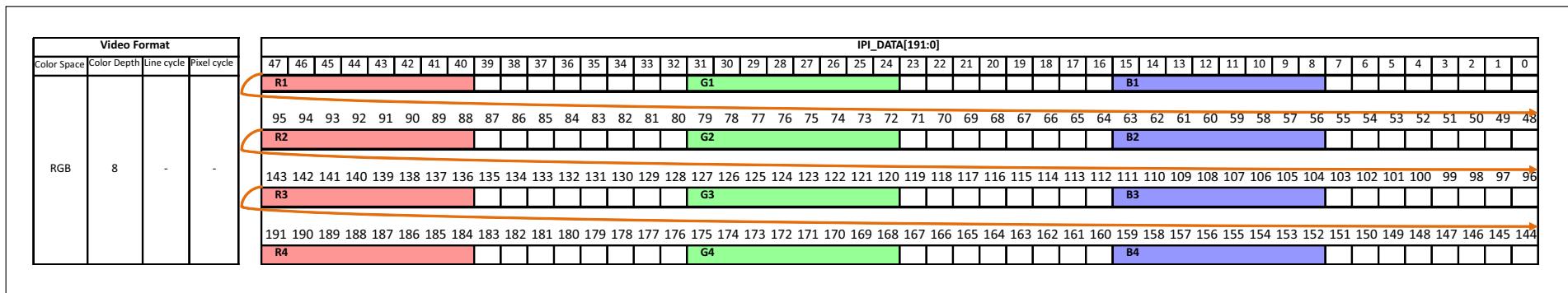
Data Input	Description
User defined	Delivers 8-bit user-defined data in 48-bit or 192-bit data.r
RAW data types and YUV 8-bit legacy	Transfers two or more pixels in parallel, reducing the number of clock cycles that is necessary to send the same image data.
Other types	Delivers one pixel per clock cycle.

In [Figure 2-29](#):

- All empty bits that do not carry information from a color component (represented as white) are filled with 0s.
- Indexes are used along with a letter identifying the color component to distinguish different pixels of the same data type. This does not apply to RGB because the R, G, and B components all belong to the same pixel.
- For YUV422 data types, there is a distinction between even and odd pixels, since only Y component is received for odd pixels.
- For RAW and YUV420 8-bit legacy, two pixels are present for each clock cycle and there is a distinction between odd and even lines.
- For all other YUV 420 data types, there is a distinction between odd and even pixels and between odd and even lines. Appropriately flags these distinctions at the interfaces through the odd_pixel and odd_line signals.
- Chroma-shifted versions of YUV 420 data types have the same sequence of component as the non-chroma-shifted versions. Therefore, it shows only once.

Figure 2-29 Pixel Color Coding at the 48-bit Interface

The pixel color coding at each 48-bit of the 192-bit IPI is the same as the 48-bit IPI pixel mapping. For example, mapping of RGB888 data type at 192-bit IPI is as follows:

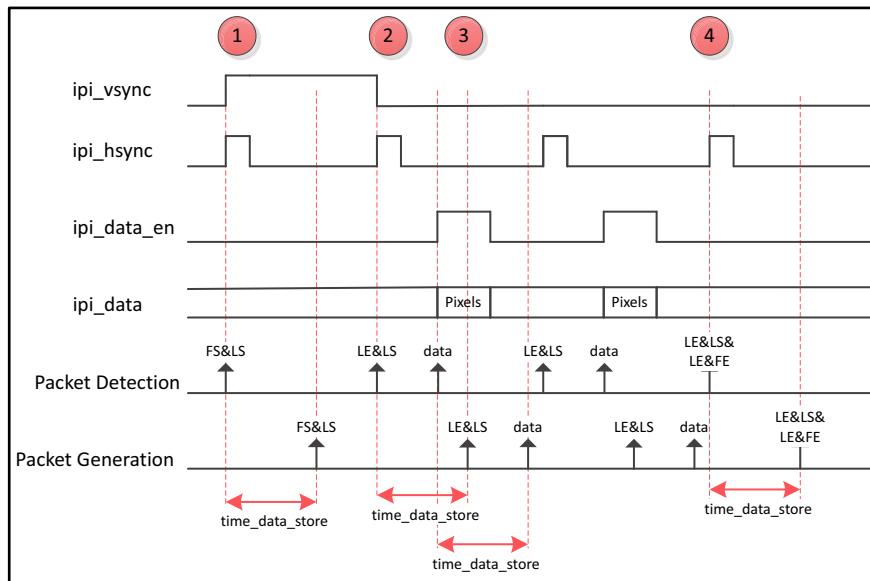
Figure 2-30 Pixel Color Coding at 192-bit IPI

2.3.3 IPI Store and Forward Mode

The system generates the synchronous packet and data packet based on the IPI inputs. It reorders the received pixel to CSI-2 bus format, based on the data type configuration, as shown in [Figure 2-29](#).

time_data_store

To avoid the underflow of the FIFO during packet transmission, it is a must to receive the complete data, before the controller sends the data. There is a delay named time_data_store between packet check and packet generation. The time_data_store is the expected width of ipi_data_en, and can be calculated based on the value of ipi_pixels_qst and ipi_dt_qst.

Figure 2-31 Generation of CSI Packets Based on IPI Events

The following are the events shown in Figure 2-31:

1. When `ipi_vsync` and `ipi_hsync` are set to 1, DWC_mipicsi2_device sequentially generates the Frame Start packet and the first Line Start packet.
2. When only `ipi_hsync` is set to 1, DWC_mipicsi2_device sequentially generates the Line End packet (if a Line Start is already transmitted) and the Line Start packet.
3. When `ipi_data_en` is set to 1, DWC_mipicsi2_device generates the data packet base on `ipi_data`.
4. At the last `ipi_hsync` of a frame, DWC_mipicsi2_device sequentially generates the four consecutive packet (LE/LS/LE/FE).

2.3.3.1 IPI Timing Driver Requirement for IPI Store and Forward Mode

To maintain the IPI timing information, respect the following constraints:

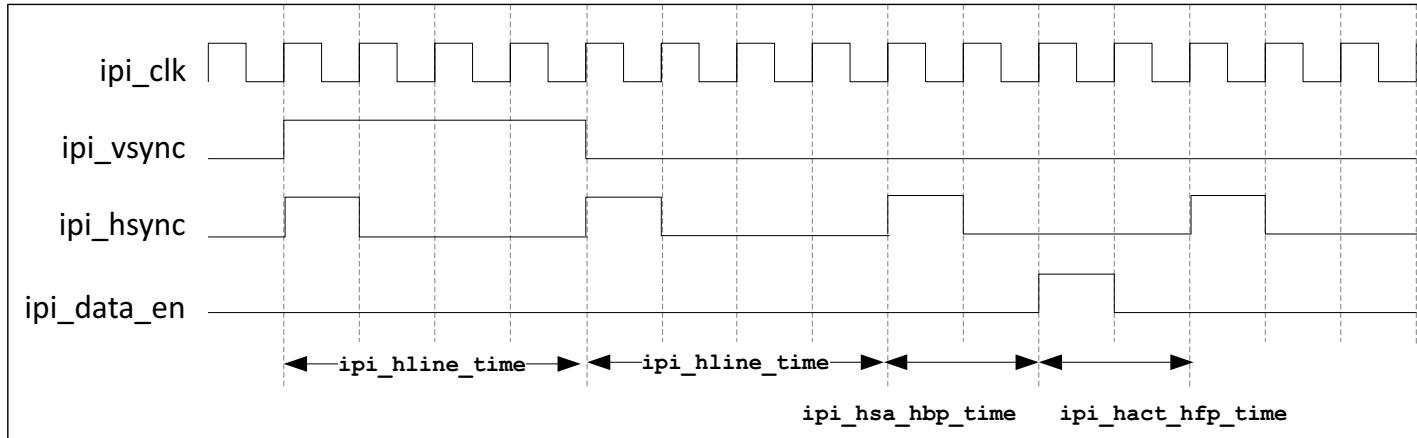
- Line synchronization packet mode disabled
 $\text{ipi_hline_time} > \text{LP_trans_time}$
- Line synchronization packet mode enabled
 $\text{ipi_hsa_hbp_time} > 2 * \text{SP_trans_time}$
 $\text{ipi_hact_hfp_time} > \text{LP_trans_time}$
 $\text{ipi_hline_time} > 4 * \text{SP_trans_time}$



For `SP_trans_time` and `LP_trans_time` calculation, see
“DWC_mipicsi2_device Packet Transmission Calculation” on page 735

Due to implementation limitation, follow these constraints between the IPI events:

- Insert at least four `ipi_clk` cycles between two `ipi_hsync`.
- Insert at least 10 `ipi_clk` cycles between the falling edge of `ipi_data_en` and the next rising edge of `ipi_data_en`.
- Insert at least 2 `ipi_clk` cycles between the rising edge of `ipi_hsync` and the rising edge of `ipi_data_en` ($HSA+HBP \geq 2 \text{ } ipi_clk$).

Figure 2-32 IPI Timing Information

Note The Header FIFO depth parameter limits the number of packets buffered in the controller. For more information, see “Core Configuration Parameters” in Chapter 3, “Parameter Descriptions”.

2.3.4 IPI Cut-Through Mode

DWC_mipicsi2_device provides a mechanism to write and read IPI RAM at the same time allows using `ipi_data_en` in continuous mode.

This mechanism maintains the IPI timing information and reduces memory requirements. For more memory information, see “[Memory](#)” on page [91](#).

IPI Timing Driver Requirement for IPI Cut-Through Mode, this requirement is the same as “[IPI Timing Driver Requirement for IPI Store and Forward Mode](#)” on page [51](#).

2.3.4.1 IPI Cut-Through Mode Configuration

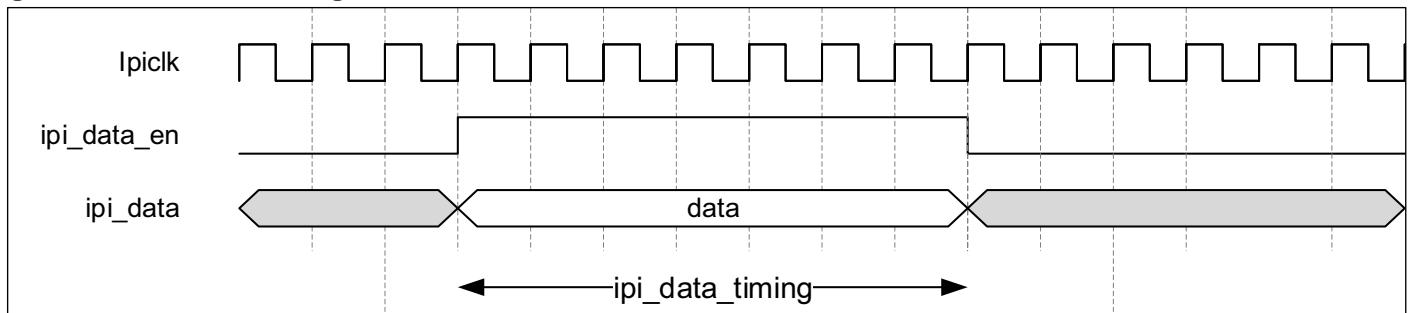
IPI Cut through mode is configurable using the `ipi_mode` field of the `IPI_PKT_CFG` register and the `IPI_DATA_SEND_START` register.

Program the `ipi_data_send_start` to avoid the FIFO underflow. This value depends upon the following:

- `ipi_data_timing`: The instant the last byte is transmitted on the IPI interface, expressed in `ipi_clk`.
- `ppi_data_timing`: The instant the last byte is transmitted on the PPI interface, expressed in `ipi_clk`.

2.3.4.1.1 `ipi_data_timing` Calculation

[Figure 2-33](#) shows the `ipi_data_timing` parameter.

Figure 2-33 IPI Data Timing

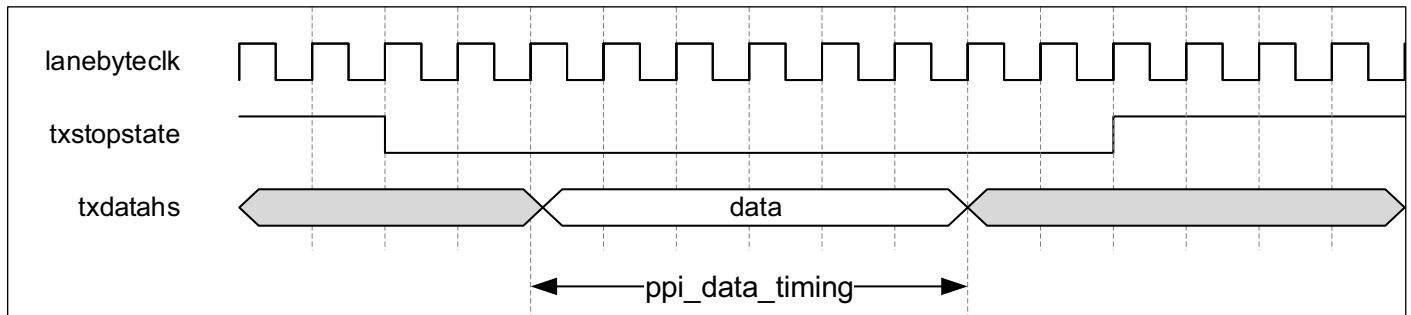
ipi_data_timing (in **ipi_clk**) = **Pixelline/ppc**

Where:

- **Pixelline** = Number of pixels in one line.
- **ppc** = pixels per **ipi_clk** cycle, according to [Figure 2-33](#).

2.3.4.1.2 ppi_data_timing Calculation

[Figure 2-34](#) shows **ppi_data_timing** parameter:

Figure 2-34 ppi_data_timing

[Table 2-3](#) lists the formula to calculate the **ppi_data_timing** value.

Table 2-3 Formula to Calculate the ppi_data_timing Value

PHY Type	Formula
D-PHY	$\text{ppi_data_timing(ipi_clk)} = (\text{T_LANEBYTECLK}/\text{Tipi_clk}) * ((6 + \text{pixelline} * \text{bpp}) / (\text{phydatawidth} * \text{nlanes} / 8))$
C-PHY 16-Bit	$\text{ppi_data_timing(ipi_clk)} = (\text{T_LANEBYTECLK}/\text{Tipi_clk}) * (7 + ((2 + \text{pixelline} * \text{bpp}) / (2 * \text{nlanes})))$
C-PHY 32-Bit	$\text{ppi_data_timing(ipi_clk)} = (\text{T_LANEBYTECLK}/\text{Tipi_clk}) * (4 + ((2 + \text{pixelline} * \text{bpp}) / (4 * \text{nlanes})))$

PHY Type	Formula
Where:	
<ul style="list-style-type: none"> ■ Pixel line = Number of pixels in one line ■ Bpp = Byte Per Pixel ■ phydatawidth = The data width of PHY ■ nlanes = The number of active lanes 	



For the following color coding, the Pixel line considered should be for the odd line:

- CSI2_Y420_8B
- CSI2_CSPS_Y420_8B
- CSI2_CSPS_Y420_10B
- CSI2_Y420_10B

2.3.4.1.3 ipi_data_send_start Configuration

According to ipi_data_timing and ppi_data_timing values, the timing requirements are different.

When the width of the payload FIFO is 64 bits:

- ipi_data_timing - ppi_data_timing <= 12
ipi_data_send_start = 12 + (T_LANEBYTECLK/Tipi_clk) * 12
- ipi_data_timing - ppi_data_timing > 12
ipi_data_send_start = ipi_data_timing - ppi_data_timing + (T_LANEBYTECLK/Tipi_clk) * 12

When the width of the payload FIFO is 128 bits:

- ipi_data_timing - ppi_data_timing <= 24
ipi_data_send_start = 24 + (T_LANEBYTECLK/Tipi_clk) * 24
- ipi_data_timing - ppi_data_timing > 24
ipi_data_send_start = ipi_data_timing - ppi_data_timing + (T_LANEBYTECLK/Tipi_clk) * 24

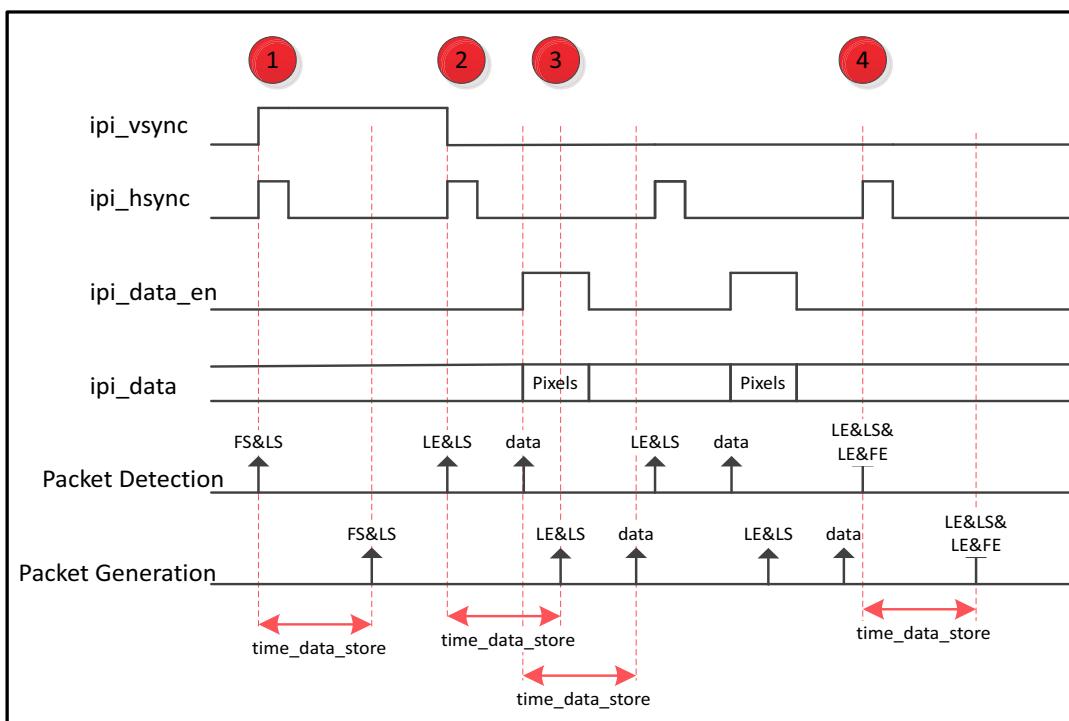
2.3.5 IPI Line Synchronization Omission Mode

The line synchronization packets are optional, therefore CSI-2 and DWC_mipicsi2_device provide a mode to omit their transmission.

You can configure IPI Line Synchronization Omission mode using the ipi_hsync_pkt_en field of the IPI_PKT_CFG register.

Figure 2-35 shows the following events:

- When ipi_vsync and ipi_hsync are set to 1, DWC_mipicsi2_device generates the Frame Start packet.
- When ipi_data_en is set to 1, DWC_mipicsi2_device generates the data packet based on ipi_data.
- At the last ipi_hsync of a frame, DWC_mipicsi2_device generates the Frame End packet.

Figure 2-35 Generation of CSI Packets Without Line Synchronous Packet Based on IPI Events

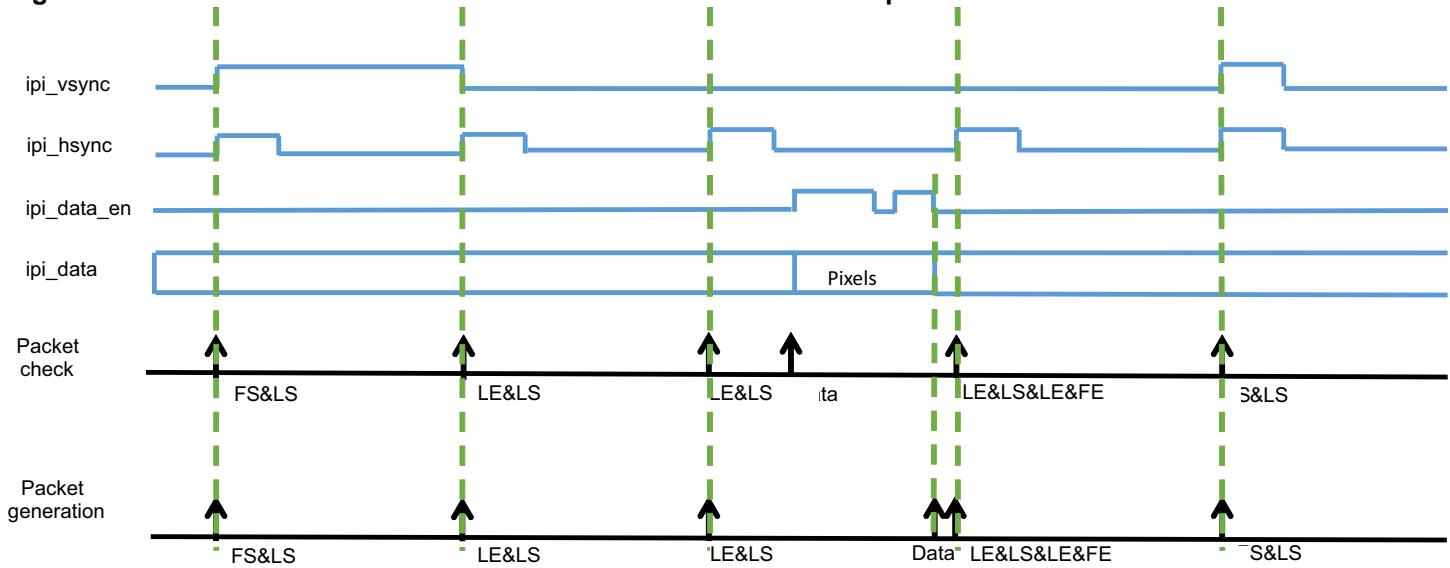
2.3.6 IPI Backpressure Mode

DWC_mipicsi2_device provides a Backpressure mechanism to inform the IPI driver that the memory is almost full. The `ipi_halt` signal enforces the Backpressure mechanism. When this signal is asserted, DWC_mipicsi2_device stops receiving (sync events and payload) in the IPI interface. After `ipi_halt` is de-asserted, the controller resumes receiving these events and payload. The Backpressure mode allows the pixel stream to be non-continuous. In this mode, DWC_mipicsi2_device does not support IPI timing delivery.

Configure the IPI Backpressure mode using the `IPI_PKT_CFG.ipi_mode` register field.

Figure 2-36 shows the following events:

- The controller generates the sync packet immediately when it detects the sync event.
- The controller generates the data packet when it receives the data completely.

Figure 2-36 Generation of CSI Packets Based on IPI Events for Backpressure Mode

2.3.6.1 IPI Timing Driver Requirement for Backpressure Mode

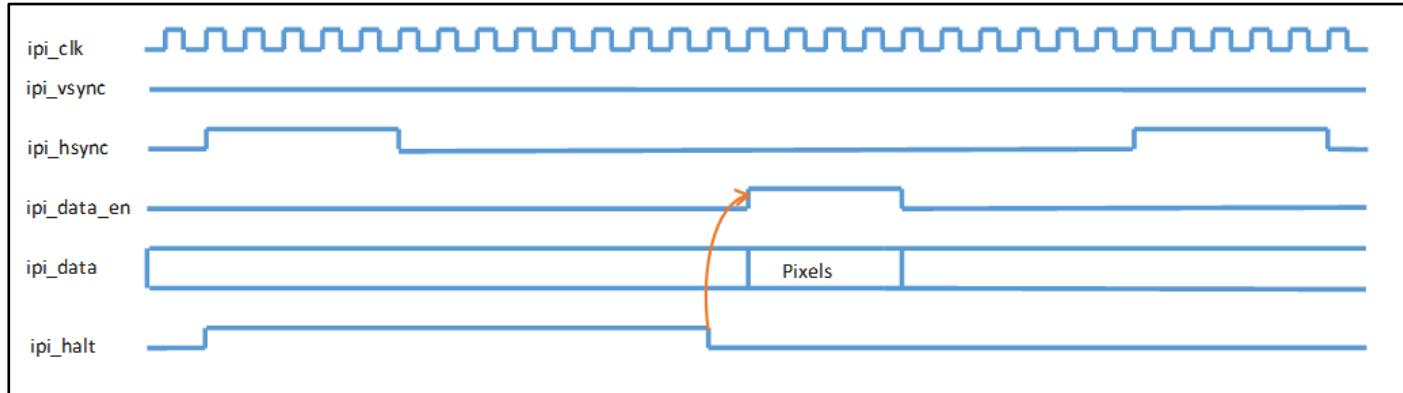
Due to implementation, follow these constraints between the IPI events:

- Insert at least four `ipi_clk` cycles between two `ipi_hsync`.
- Insert at least 10 `ipi_clk` cycles between the falling edge of `ipi_data_en` and the next rising edge of `ipi_hsync`.

2.3.6.2 Halt Functionality

DWC_mipicsi2_device provides a Backpressure mechanism to inform the IPI driver that the memory is almost full. The following examples demonstrate the behavior of the IPI interface under different scenarios. The `ipi_halt` Signal asserts during all IPI_HSA_TIME and IPI_HBP_TIME.

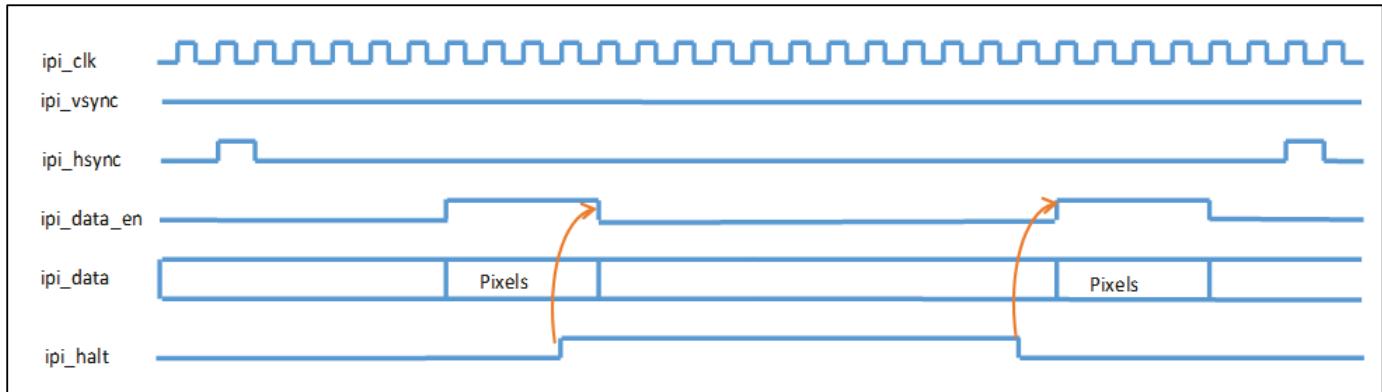
When `ipi_halt` is asserted during HSYNC signal, no update is required. In Figure 2-37, when `ipi_halt` is de-asserted, the `ipi_data_en` can be asserted.

Figure 2-37 ipi_halt signal asserts during all IPI_HSA_TIME and IPI_HBP_TIME

The `ipi_halt` signal asserts during IPI_ACTIVE_TIME.

[Figure 2-38](#) shows the `ipi_halt` behavior when the `ipi_halt` signal asserts during IPI_ACTIVE_TIME.

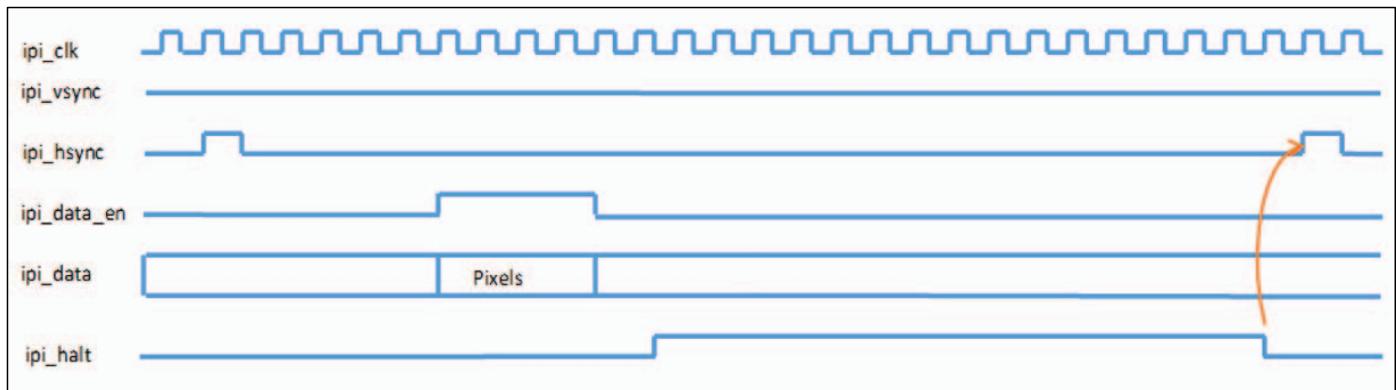
Figure 2-38 ipi_halt signal asserts during IPI_ACTIVE_TIME



The `ipi_halt` signal asserts during IPI_HFP_TIME.

[Figure 2-39](#) shows the `ipi_halt` behavior when the `ipi_halt` signal asserts during IPI_HFP_TIME.

Figure 2-39 ipi_halt signal asserts during IPI_HFP_TIME



2.3.7 IPI Embedded Data

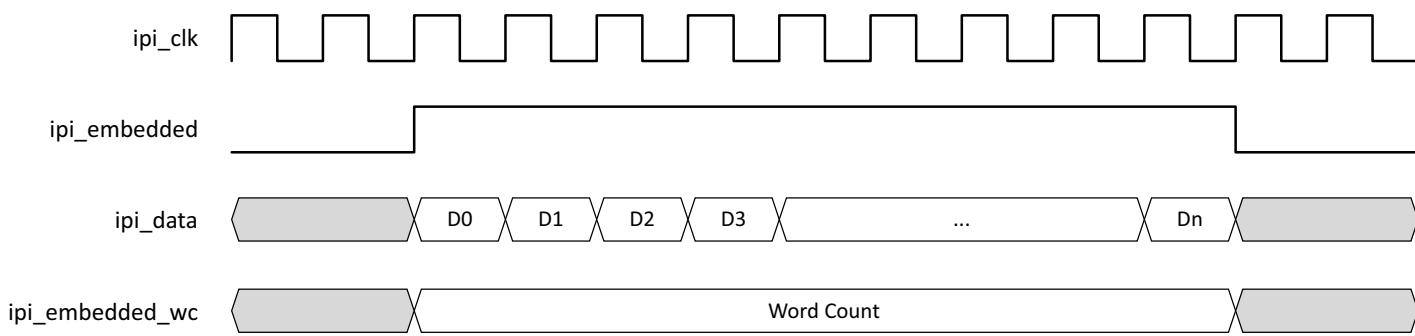
You can embed extra lines that contain additional information in VSA, VBP, and VFP lines. The VSA, VBP, and VFP lines may have zero or more lines of embedded data (data type 0x12).

The following ports help in embedding the extra lines:

- `ipi_embedded`: This port asserts when `ipi_data` bus transmits the embedded data.
- `ipi_embedded_wc`: This port indicates the word count of the current embedded data packet.

The DWC_mipicsi2_device supports:

- The dynamic word count for embedded data.
- The IPI embedded data transmitted in IPI “store and forward mode” and “cut through mode”.

Figure 2-40 IPI Embedded Data Timing

The prerequisites to embed the extra lines in the IPI frame are as follows:

- It is not allowed to insert embedded data in the last line of VFP.
- The embedded data size cannot be larger than the video line.
- The simultaneous IDI and IPI mode do not support the embedded data in IPI.
- The embedded lines should keep the same HSA and HBP as video lines, to keep the same IPI timing.
- The memory-size should be at least 1*long-video-packet-size if embedded data are inserted in the cut-through mode.

2.3.8 Enabling IPI

The IPI stores the received data in memory. Upon retrieving data from memory, it reorganizes the data in a pixel stream format. And it conveys the stream and additional frame timing signals containing vertical and horizontal information to the interface through a pixel data bus. This interface requires an externally supplied pixel clock.

To enable IPI, select “IPI” from the Data Interface > Select System Interface drop-down option in the Basic Configuration window in the coreConsultant GUI. In parallel, select the Number of IPI Interfaces. You can also enable 64-bit IDI interface option if required.

For details about this option, see the *Parameter Descriptions* chapter.

2.3.9 Registers Related to IPI

Following are the registers related to IPI:

- `IPI_PIXELS`
- `IPI_PKT_CFG`
- `IPI_MAX_FRAME_NUM`
- `IPI_LINES`
- `IPI_DATA_SEND_START`
- `IPI_STEP_LINE_NUM`
- `IPI_START_LINE_NUM`
- `IPI_FIFO_STATUS`

2.3.10 Signals Related to IPI

Following are the signals related to IPI:

- ipi_clk
- ipi_data
- ipi_vsync
- ipi_vsync
- ipi_hsync
- ipi_embedded
- ipi_halt
- ipi_embedded_wc
- ipi_data_parity
- ipi_mem_wclk
- ipi_mem_waddr
- ipi_mem_waddr_parity
- ipi_mem_wen
- ipi_mem_wdata
- ipi_mem_rclk
- ipi_mem_raddr
- ipi_mem_raddr_parity
- ipi_mem_ren

For more information, see the “IPI Interface Signals” and the “IPI Payload RAM Signals” sections in *Signal Descriptions* chapter.

2.3.11 Programming IPI

For more information, see the “IPI Initialization Flow” section in the *MIPI CSI-2 Device Controller User Guide*.

2.4 Multiple IPI Support

It is possible to support up to four IPI interfaces simultaneously with different Virtual Channels (VC). Multiple IPI transmission has two operating modes:

- Multiple IPI with no Timing Delivery mode
- Multiple IPI with Timing Delivery mode

Each IPI requires the same *ipi* clock. This is because `DWC_mipicsi2_device` controls the transmission based on the *Multiple IPI Transmission Model* which is constructed by frame parameters with unified *ipi* clock. The methodology to build the *Multiple IPI Transmission Model* is described later in this chapter.

2.4.1 Multiple IPI with No Timing Delivery Mode

In this mode, `DWC_mipicsi2_device` does not deliver timing for all the IPIs. However, the transmission of each interface is based on the following polling methods:

- All IPIs do not deliver timing.
- All IPIs work in Backpressure mode.

2.4.2 Multiple IPI with Timing Delivery Mode

In this mode, the `DWC_mipicsi2_device` can deliver timing for all the IPIs, but there are some requirements for all the IPIs listed as below:

- Each IPI requires timing delivery.
- All IPIs work in Store and Forward mode.
- For all IPIs, any two frame rates require multiple relationship, including 1:1. For example, 15fps/30fps/60fps.

This mode requires constructing the *Multiple IPI Transmission Model* Multiple IPI Transmission Model based on the resolutions and frame rates of all IPIs and configuring the corresponding registers. Then `DWC_mipicsi2_device` controls multiple IPI transmission based on the configurations.

User logic should control all IPIs input based on these frame parameters in *Multiple IPI Transmission Model* and make sure they are consistent with the configuration of each IPI registers and multiple IPI control registers.

The basic principles for constructing the mode are as follows:

- Let all IPIs have the same overall number of vertical lines (VSA+VBP+VACT+VFP+Frame_Blank_lines).
- Since the frame rate of each IPI is required to have a multiple relationship, so the Hline of each IPI also has a multiple relationship.
If the frame rate of each IPI is 1:1, the Hline of each IPI is the same.
- Minimize the HSA, and HPB of each IPI as much as possible, because the HFP is reserved for other IPIs transmissions.

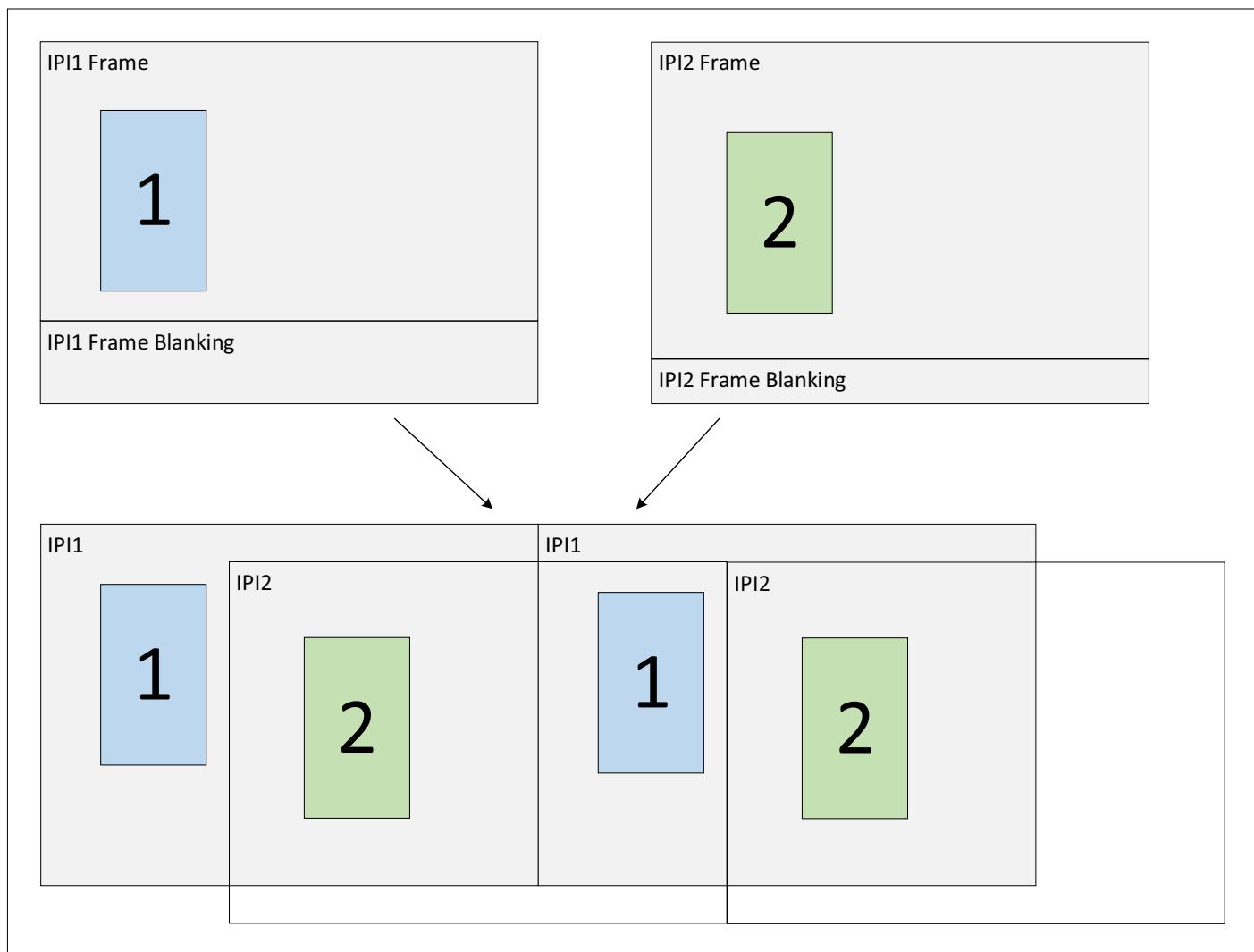
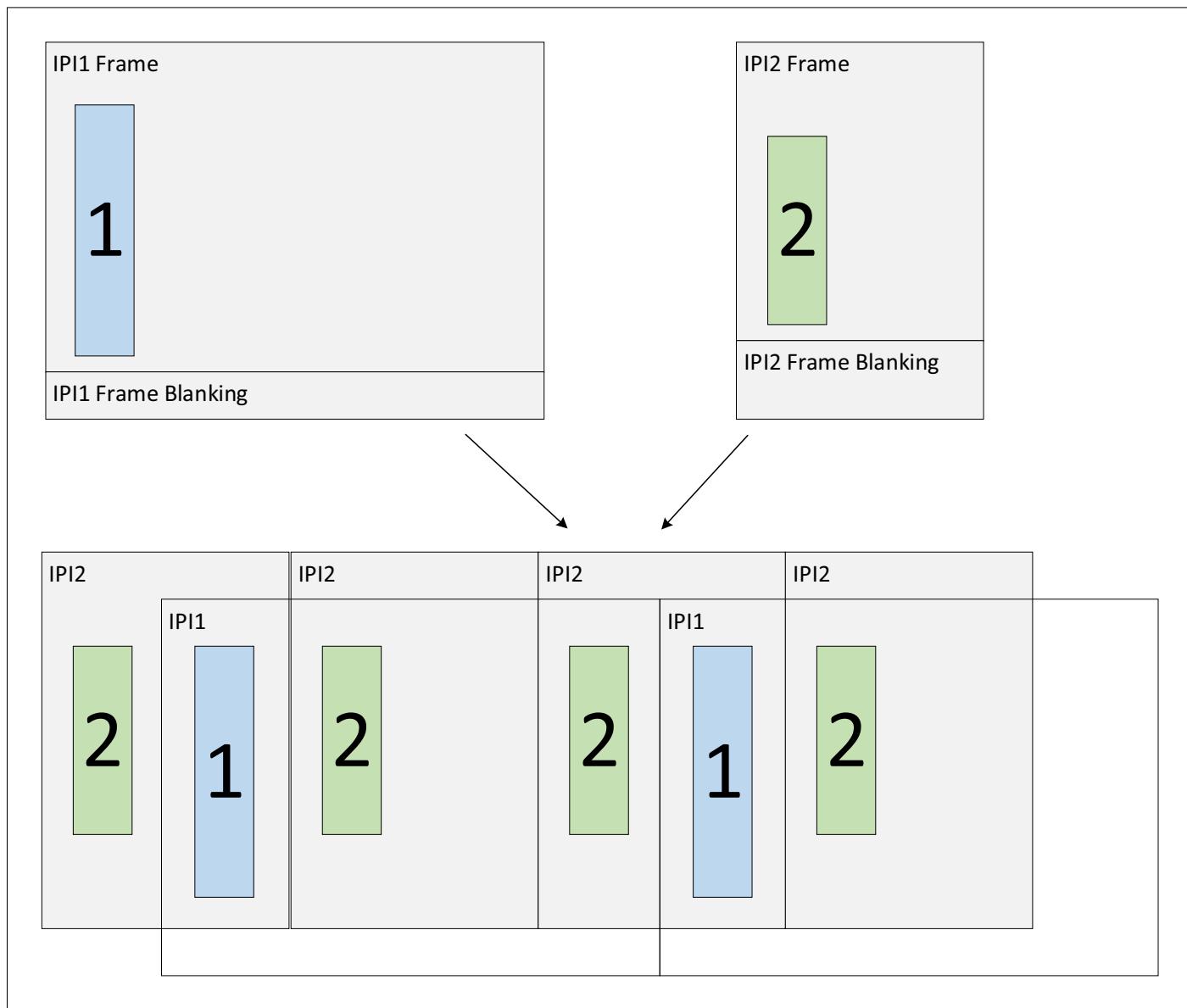
Figure 2-41 FPS 1:1 Example

Figure 2-42 FPS 1:2 Example

The following steps describe how to construct the *Multiple IPI Transmission Model*:

1. Select the smallest frame rates of all IPIs.
2. Based on multiple relationship between all IPIs and smallest frame rate, find out the least common multiple. Then set the LCM to field number.
3. Set the field line number for the field model, which should be greater than the maximum of all IPI's vertical resolution.
4. Set the IPI transmission order in each field and ensure the field bandwidth is enough to transmit. If the field bandwidth is not enough, two ways are provided to optimize bandwidth:
 - Return to Step 3 to reduce the line number so that the line time can get bigger.

- Stay in step 4 to rearrange the order of IPIs in each field. See [Chapter 2.4.2.2, "Example 2"](#) for further explanation about why change order is possible to optimize bandwidth.

To ensure the field bandwidth is enough for the allocated IPI transmission, the following equation must be tenable.

- $1/\text{smallest_frame_rate} = \text{field_line_number} * \text{field_line_time} * \text{field_number}$
- $\text{field_line_number} > \text{Max}(\text{IPI}[n]_\text{VACT_LINES})$
- $\text{field_line_time} \geq \text{Sum}(\text{ipi}[n]_\text{hsa_hbp_time}, \text{ipi}[n]_\text{lp_time})$

Where:

- $n = 1, 2, 3$ or 4
- $\text{ipi}[n]_\text{hsa_hbp_time}$ is the time between the rising edge of **ipi_hsync** and **ipi_data_en**.
- $\text{ipi}[n]_\text{lp_time}$ is the time for data packet transmission.

In addition to the IPI timing delivery requirements, you must configure the corresponding register to respect the following constraints:

- Line Sync Enable mode
 - $\text{ipi}[n]_\text{hsa_hbp_time} \geq 2 * \text{SP_trans_time}$
 - $\text{ipi}[n]_\text{lp_time} \geq \text{LP_trans_time}$
 - $\text{ipi}[n]_\text{hsa_hbp_time} + \text{ipi}[n]_\text{lp_time} \geq 4 * \text{SP_trans_time}$
- Line Sync Disable Mode
 - $\text{ipi}[n]_\text{hsa_hbp_time} \geq 1$

For more information about SP_trans_time and LP_trans_time calculation, See "["DWC_mipicsi2_device Packet Transmission Calculation"](#) on page [735](#)".

2.4.2.1 Example 1

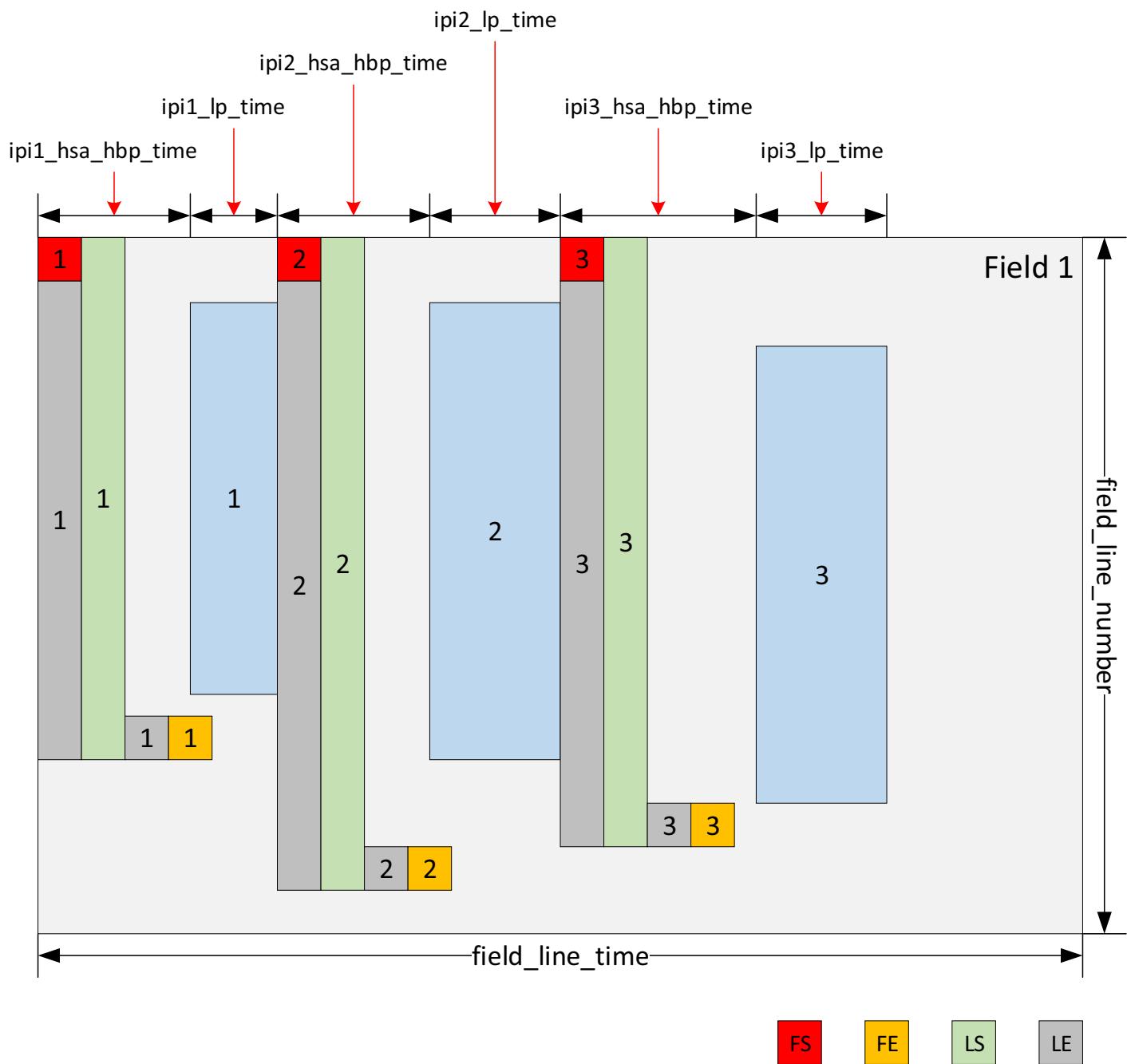
Bandwidth on D-PHY at 2.5 Gbps, 4 Lanes 10Gbps:

- IPI1: Video Formats RGB888 / 1280x720 (720p) / 15fps
- IPI2: Video Formats YUV420 8-Bit / 1920x1080 (1080p) / 15fps
- IPI3: Video Formats RGB888 / 1920x1080 (1080p) / 15fps

Follow the *Multiple IPI Transmission Model* construction steps:

- Select the smallest frame rate to be 15fps.
- IPI1:min=1:1, IPI2:min=1:1 and IPI3:min=1:1, so LCM(1,1,1)=1. Then set the field number to 1.
- Set the field line number to 1500 as an example, so that following condition is satisfied:
 $\text{field_line_number} > \text{Max}(720, 1080, 1080)$.
- IPI1, IPI2 and IPI3 are all sent in field1.
 But if field_line_time is not enough for IPI1/2/3 after setting field_line_number to 1500, then go back to step 3 to set 1200 as an example and re-check.

[Figure 2-43](#) shows the Multiple IPI Transmission Model of example 1.

Figure 2-43 Example 1 of Multiple IPI Transmission Model

After the model is constructed, then user logic should control the input of all IPIs based on this model and configure registers for each IPI as below:

- For each IPI configuration, the overall number of vertical lines is
 - $\text{IPI}[n]_{\text{VSA_LINES}} + \text{IPI}[n]_{\text{VACT_LINES}} + \text{IPI}[n]_{\text{VFP_LINES}} + \text{IPI}[n]_{\text{FRAME_BLANKING_LINES}} = \text{field_line_number}$

- IPI1
 - The number of pixels in a single video pattern packet is 1280 pixels.
 - The time of HSA and HBP is ipi1_has_hbp_time.
 - Overall time for the horizontal line is field_line_time.
 - The vertical resolution (IPI1_VACT_LINES) is 720 lines.
 - The overall number of vertical lines is 1200 lines.
- IPI2
 - The number of pixels in a single video pattern packet is 1920 pixels.
 - The time of HSA and HBP is ipi2_has_hbp_time.
 - Overall time for the horizontal line is field_line_time.
 - The vertical resolution (IPI2_VACT_LINES) is 1080 lines.
 - The overall number of vertical lines is 1200 lines.
- IPI3
 - The number of pixels in a single video pattern packet is 1920 pixels.
 - The time of HSA and HBP is ipi3_has_hbp_time.
 - Overall time for the horizontal line is field_line_time.
 - The vertical resolution (IPI3_VACT_LINES) is 1080 lines.
 - The overall number of vertical lines is 1200 lines.

2.4.2.2 Example 2

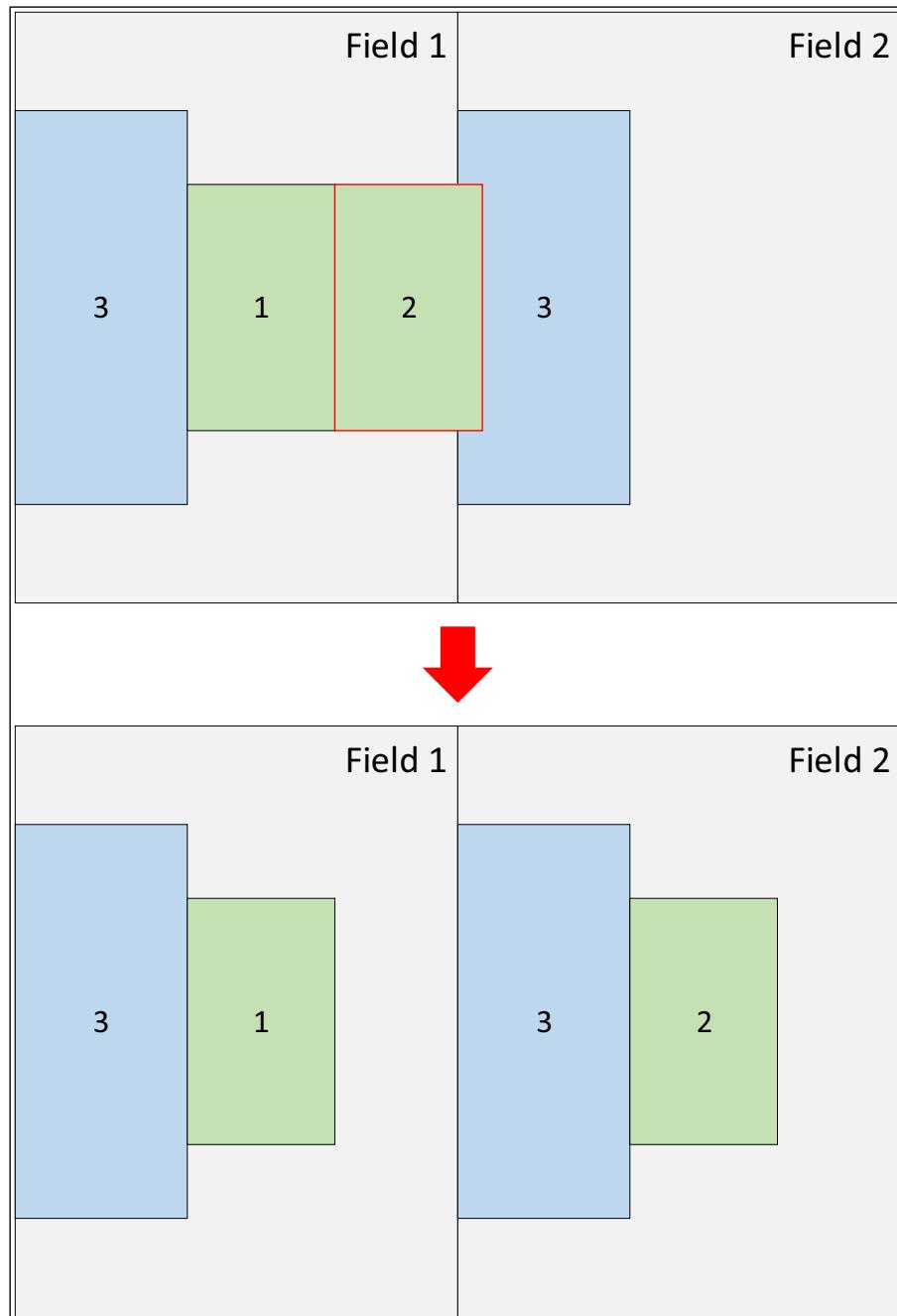
Bandwidth on D-PHY at 2.5 Gbps, 4 Lanes 10Gbps:

- IPI1: Video Formats RGB888 / 1280x720 (720p) / 15fps
- IPI2: Video Formats RGB888 / 1280x720 (720p) / 15fps
- IPI3: Video Formats RAW20 / 1920x1080 (1080p) / 30fps

Follow the *Multiple IPI Transmission Model* construction steps:

1. Select the smallest frame rate to be 15fps.
2. IPI1:min=1:1, IPI2:min=1:1 and IPI3:min=1:2, so LCM(1,1,2)=2. Then set the field number to 2.
3. Set the field line number to 1200 as an example, so that following condition is satisfied:
field_line_number > Max (720, 720, 1080).
4. Then the transmission order can have a variety of combinations.

Figure 2-44 shows two possible transmission orders and explain why change order can optimize bandwidth.

Figure 2-44 Change Transmission Order to Optimize Bandwidth

As shown in the [Figure 2-44](#) above, first transmission order is to send IPI1/2/3 in field 1 and only IPI3 in field 2.

But if the field_line_time is not enough to send all IPI1/2/3, so rearrange as the second transmission order. To send IPI1/3 in field 1 and IPI2/3 in field 2, and with this order it matches the bandwidth well.

[Figure 2-45](#) below shows the Multiple IPI Transmission Model of example 2.

Figure 2-45 Example 2 of Multiple IPI Transmission Model

After the model is constructed, then user logic should control the input of all IPIs based on this model and configure registers for each IPI as following:

- For each IPI configuration, the overall number of vertical lines is
 - IPI[n].VSA_LINES + IPI[n].VACT_LINES + IPI[n].VFP_LINES + IPI[n].FRAME_BLANKING_LINES = field_line_number
- IPI1

- ❑ The number of pixels in a single video pattern packet is 1280.
- ❑ The time of HSA and HBP is ipi1_has_hbp_time.
- ❑ Overall time for the horizontal line is 2*field_line_time.
- ❑ The vertical resolution (IPI1_VACT_LINES) is 720 lines.
- ❑ The overall number of vertical lines is 1200 lines.
- IPI2
 - ❑ The number of pixels in a single video pattern packet is 1280.
 - ❑ The time of HSA and HBP is ipi2_has_hbp_time.
 - ❑ Overall time for the horizontal line is 2*field_line_time.
 - ❑ The vertical resolution (IPI2_VACT_LINES) is 720 lines.
 - ❑ The overall number of vertical lines is 1200 lines.
- IPI3
 - ❑ The number of pixels in a single video pattern packet is 1920.
 - ❑ The time of HSA and HBP is ipi3_has_hbp_time.
 - ❑ Overall time for the horizontal line is field_line_time.
 - ❑ The vertical resolution (IPI3_VACT_LINES) is 1080 lines.
 - ❑ The overall number of vertical lines is 1200 lines.

2.4.3 Enabling Multiple IPI

To enable this feature, set the “Select System Interface” parameter option to “IPI” in the coreConsultant GUI. Then select the Number of IPI Interfaces.

For details about the option, see the [Chapter 3, “Parameter Descriptions”](#) chapter.

2.4.4 Registers Related to Multiple IPI

Following are the registers related to multiple IPI:

- IPI(n)_PKT_CFG
- IPI(n)_PIXELS
- IPI(n)_MAX_FRAME_NUM
- IPI(n)_START_LINE_NUM
- IPI(n)_STEP_LINE_NUM
- IPI(n)_LINES
- IPI(n)_SEND_START
- IPI(n)_FIFO_STATUS
- IPI(n)_HSA_HBP_TIME
- IPI(n)_LP_TIME
- MT_IPI(n)_TRANS_CFG
- MT_IPI_CFG
- MT_IPI_DF_TIME

- MT_FIFO_STATUS

For more details, see the [Chapter 5, “Register Descriptions”](#) chapter.

2.4.5 Programming Multiple IPI

For more information, see the “Multiple IPI Initialization Flow” section in the *MIPI CSI-2 Device Controller User Guide*.

2.5 Simultaneous IDI and IPI Mode

This section describes how DWC_mipicsi2_device operates when both IDI and IPI interfaces are enabled.

2.5.1 Overview of Simultaneous IDI and IPI Mode

IPI is a Synopsys proprietary interface that can deliver pixel data and timing information. According to the frame structure, there are some blanking areas or low-power periods between sync event packets and pixel data that the DWC_mipicsi2_device does not use. DWC_mipicsi2_device uses IDLE time to insert IDI packets during IPI frame transmission.

When IDI and IPI both exist in the design, DWC_mipicsi2_device supports simultaneous IDI and IPI; you can also configure the controller so that IDI or IPI work independently (only one interface is working).

2.5.2 Insertion of IDI Packets in IPI Transmission

The shaded area in the [Figure 2-46](#) indicates the idle periods of IPI transmission.

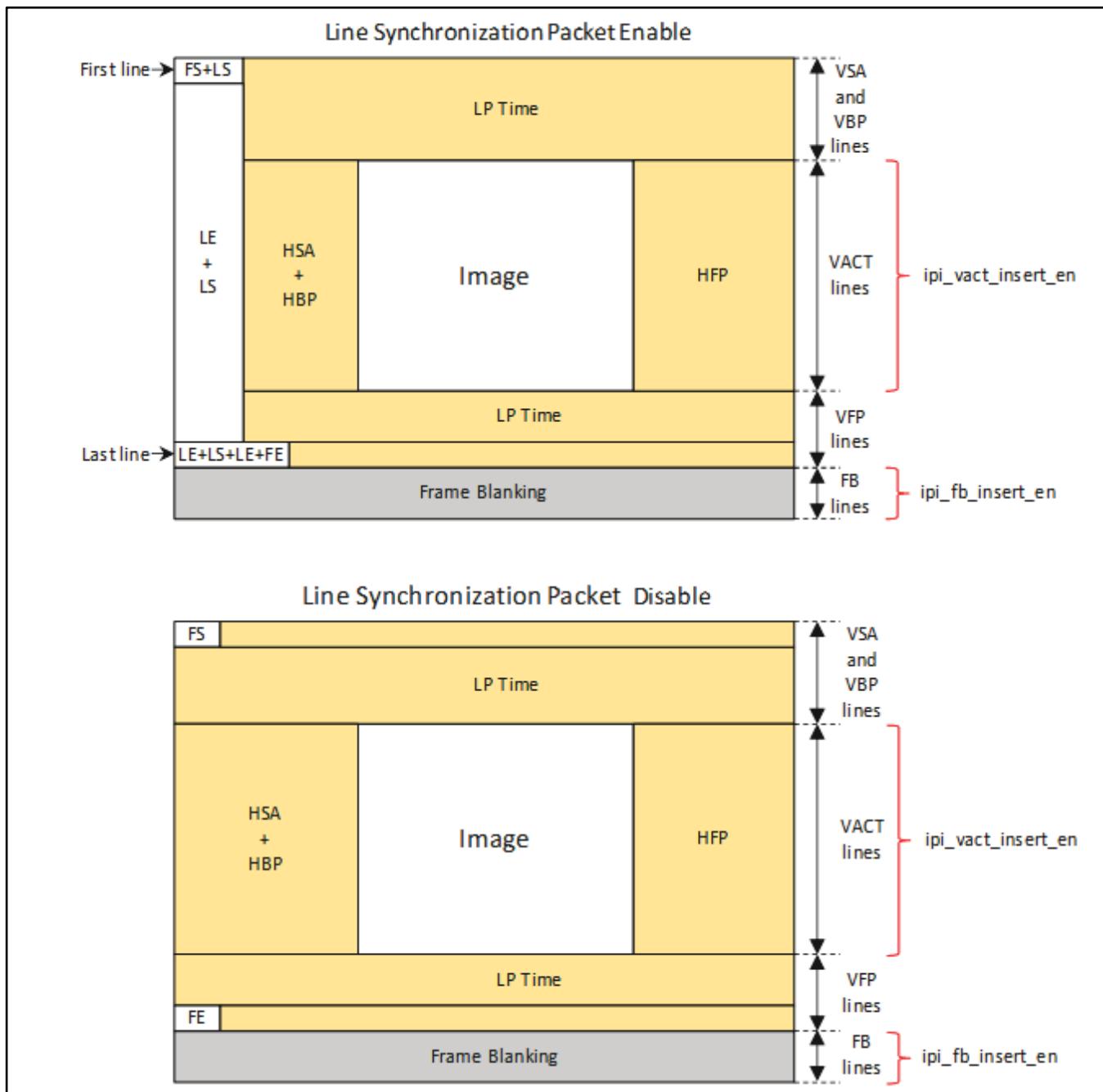
After the following packets/state, it transmits the IDI packets in the blanking periods:

- Frame Start (FS) packets, when Line Synchronization Packet is disabled
- Frame End (FE) packets, including last line and frame blanking regions
- Line Start (LS) packets, in VSA, VBP, VACT and VFP regions, except last line
- Horizontal Active (HACT) state

As shown in [Figure 2-46](#), you can configure the insertion of IDI packets in VACT and Frame Blanking regions through the following `IPI_INSERT_CTRL` register fields:

- `ipi_vact_insert_en`
 - 1: Enable insertion in VACT field
 - 0: Disable insertion in VACT field
- `ipi_fb_insert_en`
 - 1: Enable insertion in FB field
 - 0: Disable insertion in FB field

Meanwhile, insertion in other regions (VSA/VBP/VFP) is enabled by default and not configured through a register.

Figure 2-46 IDI Insertion Area Within IPI Transmission

Note

When enabling simultaneous IDI and IPI interfaces, take note of the following:

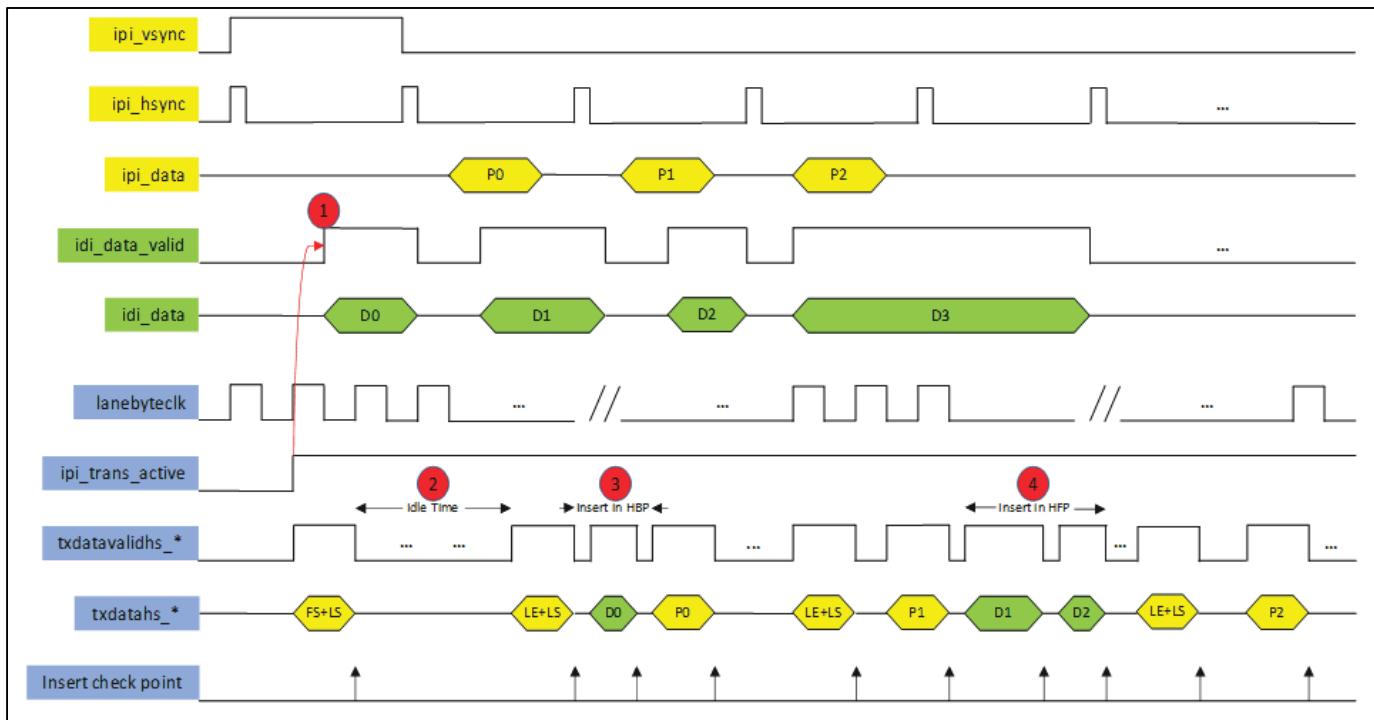
- IPI timing have higher priority, therefore, IDI does not affect IPI transmission.
- IPI must launch before IDI input. When `ipi_trans_active` is high, IDI inputs data and the `DWC_mipicsi2_device` determines where to insert IDI packets.
- When there is no space, or the bandwidth is too slow to insert IDI packets, use Backpressure mode to control IDI input.
- If there is ample time, it inserts the Multiple IDI packets in one IPI blanking field (HSA/HBP/HFP).

Example of IDI Insertion Timing

Consider the following scenario as shown in [Figure 2-47](#):

1. When IPI starts transmission (`ipi_trans_active` is high), IDI starts operation.
2. Either there is not enough time to insert IDI packet D0 or not sent the internal D0 request.
3. Insert only one IDI packet in the HBP field.
4. Insert multiple IDI packets in the HFP field because the idle time is adequate.

Figure 2-47 Overview of IDI Insertion Timing

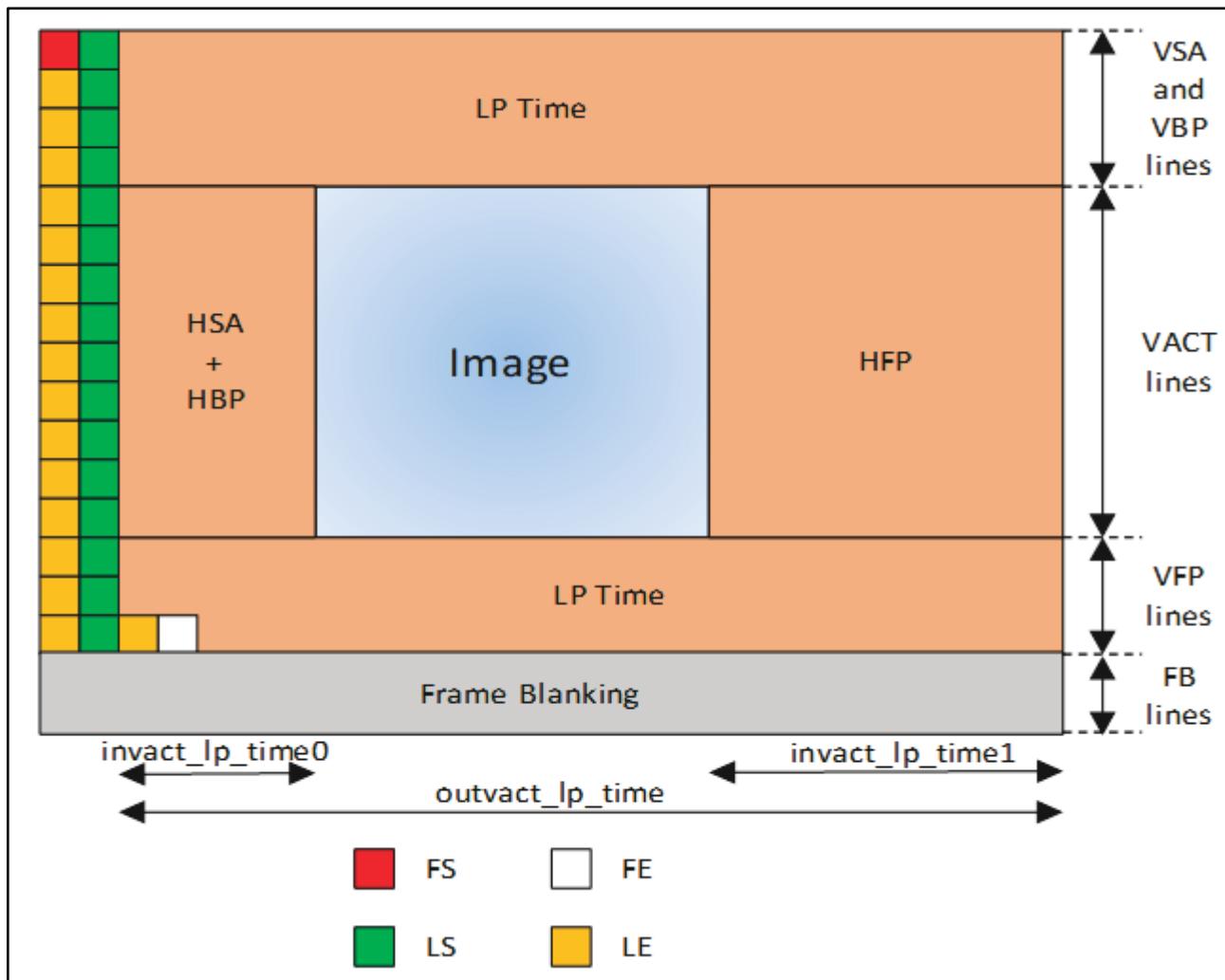


2.5.3 Time Calculation of IDI Insertion

When configuring the `IPI_INSERT_CTRL` register, the controller evaluates whether the time of the blanking field is enough to insert an IDI packet. [Figure 2-48](#) and the following section describe the method to calculate the blanking time.

2.5.3.1 Line Synchronization Packet Enabled

Figure 2-48 Time Interval with Line Synchronization Packet Enabled



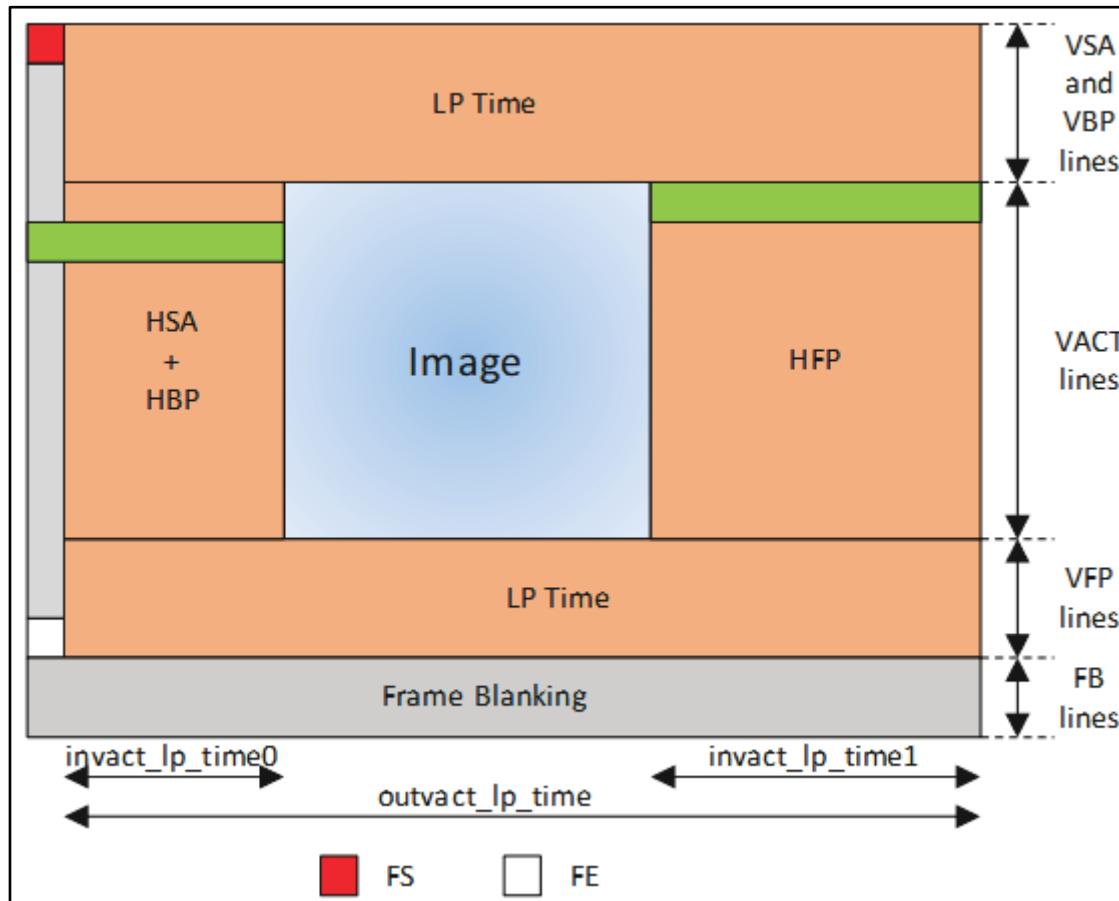
- When inside the VACT field:
 - $\text{invact_lp_time0} = \text{HSA_time} + \text{HBP_time} - 2 * \text{SP_trans_time}$
 - $\text{invact_lp_time1} = \text{HFP_time}$
 Where

$$\text{HFP_time} = \text{HLINE_time} - \text{HSA_time} - \text{HBP_time} - \text{LP_trans_time}$$
- When outside VACT field:
 - $\text{outvact_lp_time} = \text{HSA_time} + \text{HBP_time} - 2 * \text{SP_trans_time}$.
If last line, can add Frame Blanking field as a whole
 - $\text{outvact_lp_time} = \text{HLINE_time} - 4 * \text{SP_trans_time} + (\text{HLINE_time} * \text{ipi_fb_lines})$

For more information about SP_trans_time and LP_trans_time calculation, see “[DWC_mipicsi2_device Packet Transmission Calculation](#)” on page [735](#).

2.5.3.2 Line Synchronization Packet Disabled

Figure 2-49 Time Interval with Line Synchronization Packet Disabled



- When inside VACT field:
 - $\text{invact_lp_time0} = \text{HSA_time} + \text{HBP_time}$
 - $\text{invact_lp_time1} = \text{HFP_time}$

Because there is no hsync, time0 and time1 can be treated as a whole when outside VACT field:

- If first line
 $\text{outvact_lp_time} = \text{HLINE_time} - \text{SP_trans_time}$
- If last line, can add Frame Blanking field as a whole.
 $\text{outvact_lp_time} = \text{HLINE_time} - \text{SP_trans_time} + (\text{HLINE_time} * \text{ipi_fb_lines})$

Besides, VSA and VBP fields can be a whole:

$$(VSA_lines + VBP_lines) * HLINE_time + invact_lp_time0 - SP_trans_time$$

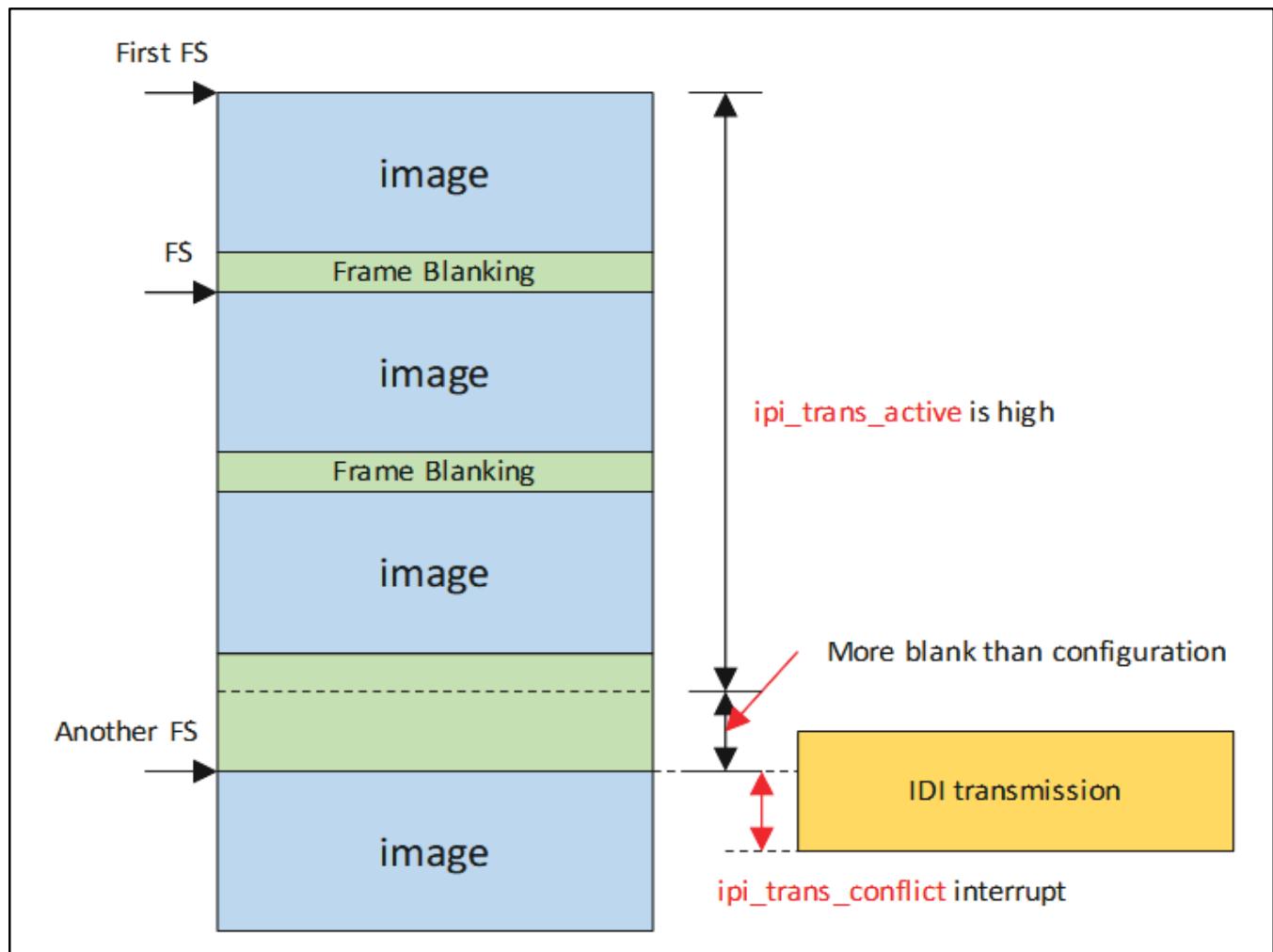
VFP field can be a whole:

$$(VFP_lines - 1) * HLINE_time + invact_lp_time1$$

2.5.4 Error Scenario

As mentioned previously, IPI timing transmission has a higher priority. This means that the controller immediately sends IPI packets to the PHY when it receives an IPI request. An error occurs if IDI occupies PHY at the same time.

Figure 2-50 IPI Transmission Error Detection



The controller counts to configured Frame Blanking lines to determine whether IPI transmission is finished as shown in [Figure 2-50](#). If the next FS packet is not received by then, the controller pulls down the `ipi_trans_active` signal to indicate the IPI has stopped. The IDI can work independently, and if it re-starts without being disabled, this can cause an `ipi_trans_conflict` error.

To ensure IPI timing delivery, take note of the following requirements:

1. During initialization, the system must enable IPI first. After detecting `ipi_trans_active` is high, IDI can be enabled.
2. If IDI is working independently and wants to enable IPI, the system must stop the upstream IDI data and ensure that IDI data transmission on the PHY is finished, and then can enable IPI input. Otherwise, the `ipi_trans_conflict` interrupt is issued the IDI is not stopped. After the IPI starts working again, repeat Step 1 if you want to enable IDI.



Note When operating in simultaneous IDI and IPI mode, VPG is not supported to work simultaneously.

2.5.5 Enabling Simultaneous IDI and IPI Mode

To enable this feature, set the “Select System Interface” parameter option to “IDI&IPI” (`CSI2_DEVICE_DATAINTERFACE=3`) in coreConsultant.

For details about this option, refer to Chapter Parameter Descriptions

2.5.6 Registers Related to Simultaneous IDI and IPI Mode

The following registers are related to simultaneous IDI and IPI operation:

- `PHY_SWITCH_TIME`
- `IPI_PKT_CFG`
- `IPI_HSA_HBP_PPI_TIME`
- `IPI_HLINE_PPI_TIME`
- `IPI_VSA_LINES`
- `IPI_VBP_LINES`
- `IPI_VFP_LINES`
- `IPI_ACT_LINES`
- `IPI_FB_LINES`
- `IPI_INSERT_CTRL`
- `IPI_TRANS_STATUS`
- `HS_IDLE register`

For details about these registers, see *Chapter “Register Descriptions”*.

2.5.7 Programming Simultaneous IDI and IPI Mode

For more information, see “Programming Simultaneous IDI and IPI Mode” in the *MIPI CSI-2 Device Controller User Guide*.

2.6 Stream Data Interface

This section describes the Stream Data Interface (SDI) of the DWC_mipicsi2_device.

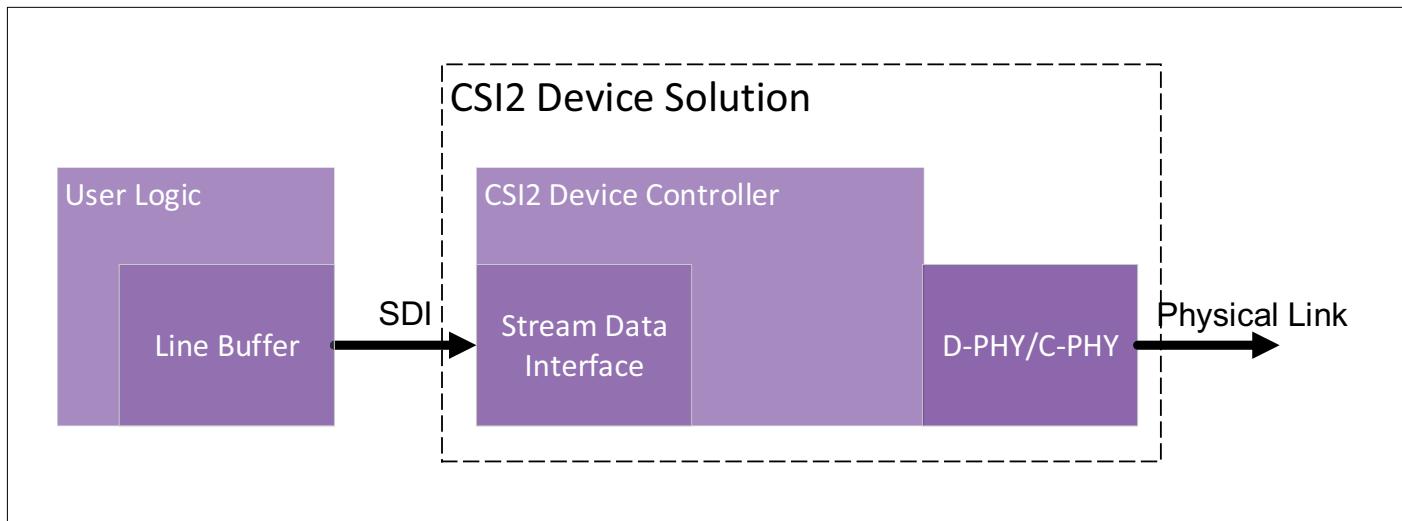
2.6.1 Overview of Stream Data Interface

Stream Data Interface (SDI) is a Synopsys proprietary interface to receive the CSI-2 packet information from the system through a parallel bus with minimal latency.

Use-Case

This interface is useful to transmit stream data directly from a memory to PHY in the system. It should have a line buffer outside CSI2 device controller. The line buffer should always be able to provide data to the CSI2 controller as it requests.

Figure 2-51 CSI2 Device Solution



2.6.2 Description of Stream Data Interface

The following options are supported for the data received on this interface:

- Bypass internal FIFO and CDC processing, minimize internal processing.
- Support packet format (Data on CSI-2 bus) and memory storage format (Data in receiver's buffer) in CSI2 spec.
 - Packet format supports ECC/CRC Bypass or Generation mode:

Bypass mode

- Bypass internal ECC/PH-CRC generation and use CSI header ECC/PH-CRC (without corrections of Header or ECC/PH-CRC field) provided by user.
- Bypass internal CRC generation and use the CSI Footer CRC (without correcting the CRC field) provided by user.

Generation mode

- Enable Internal ECC/PH-CRC generation
- Enable Internal CRC generation

- ❑ Memory storage format support Generation Mode only

Packet Header Representation

Each packet is completely delimited in time by pulses in sdi_pkt_start and sdi_pkt_end signals. When sdi_pkt_start is asserted, a packet header is available in sdi_data bus with the structure shown in Figure 2-52.

Figure 2-52 Format for Fitting D-PHY and C-PHY Headers in sdi_data

For D-PHY, some of the bits are permanently kept at 0, as indicated above. For both D-PHY and C-PHY, virtual channel (VC) and virtual channel extension (VCX) are merged into a single entity, the VC field. The whole header is protected by either ECC (for D-PHY) or redundancy and PH-CRC (for C-PHY).

- When payload mode is packet format (Data on CSI-2 bus), it supports ECC/PH-CRC Bypass or Generation mode.
 - When payload mode is memory storage format (Data in receiver's buffer), it only supports ECC/PH-CRC Generation mode.

ECC/PH-CRC Bypass mode

In ECC/PH-CRC bypass mode, controller receives the value of ECC/PH-CRC in SDI without corrections of Header or ECC/PH-CRC field.

ECC/PH-CRC Generation mode

In ECC/PH-CRC generation mode, the controller ignores the value of ECC/PH-CRC in SDI and generates the ECC/PH-CRC field.

Packet Payload Representation

If sdi_pkt_end is pulsed simultaneously with sdi_pkt_start, that means it is a short packet. Otherwise, payload is delivered between the pulses of sdi_pkt_start and sdi_pkt_end.

For packet format, the reception doesn't need to be adjusted because it's already the mapping of Data on CSI-2 bus. The sdi pld byte en indicates the valid bytes of sdi data when payload transmission.

Figure 2-53 Packet Payload Representation for all Data Types

sdi_pld_byte_en	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
All Data Types								
4'b0001	Don't care	Valid						
4'b0010	Don't care	Valid	Valid					
4'b0011	Don't care	Valid	Valid	Valid				
4'b0100	Don't care	Don't care	Don't care	Don't care	Valid	Valid	Valid	Valid
4'b0101	Don't care	Don't care	Don't care	Valid	Valid	Valid	Valid	Valid
4'b0110	Don't care	Don't care	Valid	Valid	Valid	Valid	Valid	Valid
4'b0111	Don't care	Valid	Valid	Valid	Valid	Valid	Valid	Valid
4'b1000	Valid	Valid						
Others	Don't care							

For memory storage format, controller must re-order data from memory storage format to CSI2 packet format based on data type. The mapping is same with IDI 64-bit/128-bit memory word mapping rule. The sdi_pld_byte_en indicates the valid bytes of sdi_data when payload transmission.

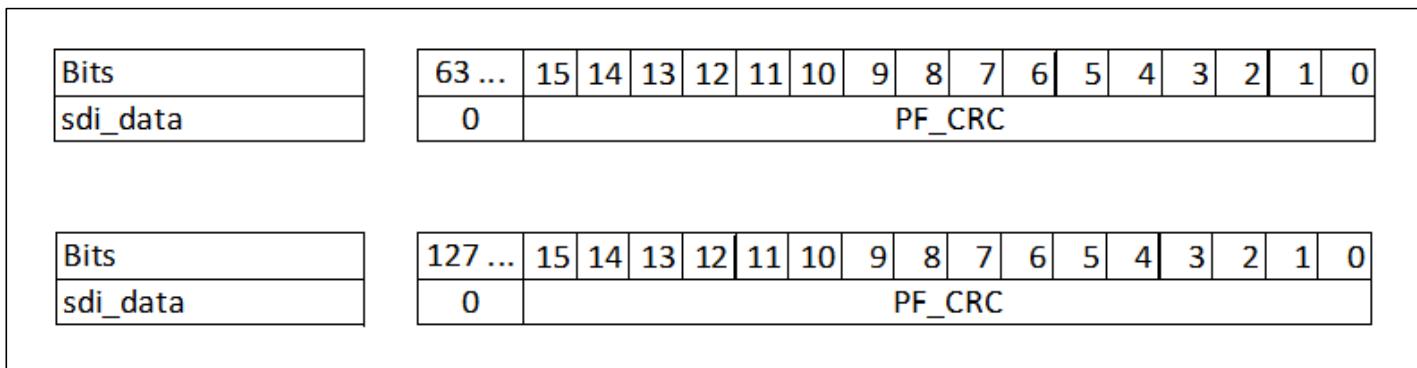
Figure 2-54 Packet Payload Representation for YUV420 8-Bit Legacy and YUV422 8-Bit

sdi_pld_byte_en	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
YUV420 8-Bit Legacy and YUV422 8-Bit								
4'b0001	Valid	Don't care						
4'b0010	Valid	Valid	Don't care					
4'b0011	Valid	Valid	Valid	Don't care				
4'b0100	Valid	Valid	Valid	Valid	Don't care	Don't care	Don't care	Don't care
4'b0101	Valid	Valid	Valid	Valid	Valid	Don't care	Don't care	Don't care
4'b0110	Valid	Valid	Valid	Valid	Valid	Valid	Don't care	Don't care
4'b0111	Valid	Don't care						
4'b1000	Valid							
Others	Don't care							
All Data Types								
4'b0001	Don't care	Valid						
4'b0010	Don't care	Valid	Valid					
4'b0011	Don't care	Valid	Valid	Valid				
4'b0100	Don't care	Don't care	Don't care	Don't care	Valid	Valid	Valid	Valid
4'b0101	Don't care	Don't care	Don't care	Valid	Valid	Valid	Valid	Valid
4'b0110	Don't care	Don't care	Valid	Valid	Valid	Valid	Valid	Valid
4'b0111	Don't care	Valid						
4'b1000	Valid							
Others	Don't care							

Packet Footer Representation

Packet footer is delivered upon the pulse of sdi_pkt_end using structure shown in [Figure 2-55](#).

- When payload mode is packet format (Data on CSI-2 bus), it supports PF-CRC16 Bypass or Generation mode.
- When payload mode is memory storage format (Data in receiver's buffer), it only supports PF-CRC16 Generation mode.

Figure 2-55 Format for footer in sdi_data

PF-CRC16 Bypass mode

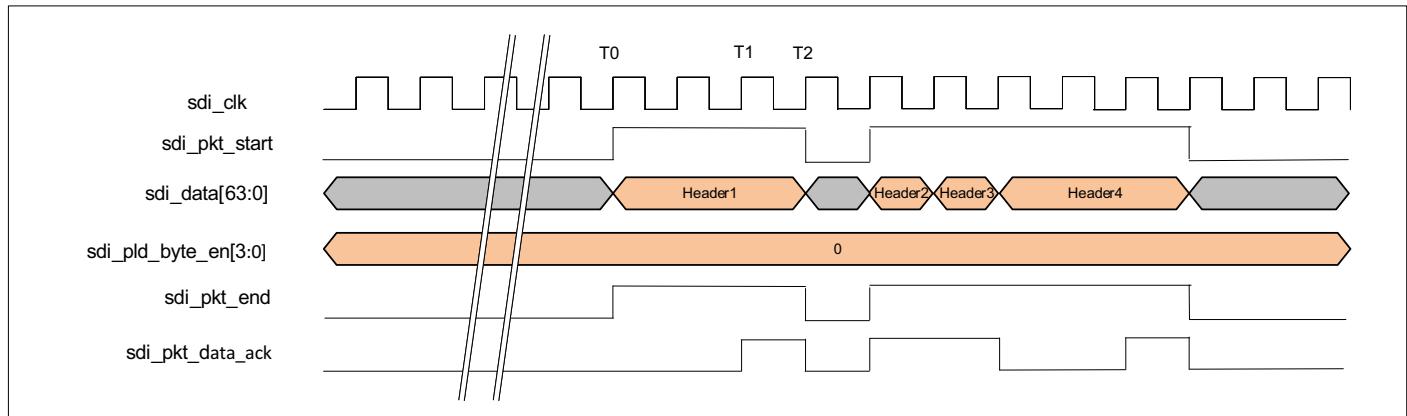
In ECC/PF-CRC16 bypass mode, controller receives the value of PF-CRC16 in SDI without correcting the CRC field.

PF-CRC16 Generation mode

In ECC/PF-CRC16 generation mode, controller ignores the value of PF-CRC16 in SDI and generate PF-CRC16 field.

Sample transfers

Payload never overlaps with the header or footer in time. sdi_byte_en always indicates the number of bytes in sdi_data that are valid in each cycle, located on the LSB side. Payload is delivered always using 8 bytes (8 bytes is for 64 bit SDI and 16 bytes is for 128bit SDI) of data per clock cycle, meaning that 0-byte correspond to wait cycles. Only for the last cycle of payload of a packet, it is possible to deliver less than 16 bytes, as required to finish it. Two consecutive packets never overlap in time, but can be delivered in consecutive clock cycles, in a non-stop flow.

Figure 2-56 Short Packet Transmission Timing

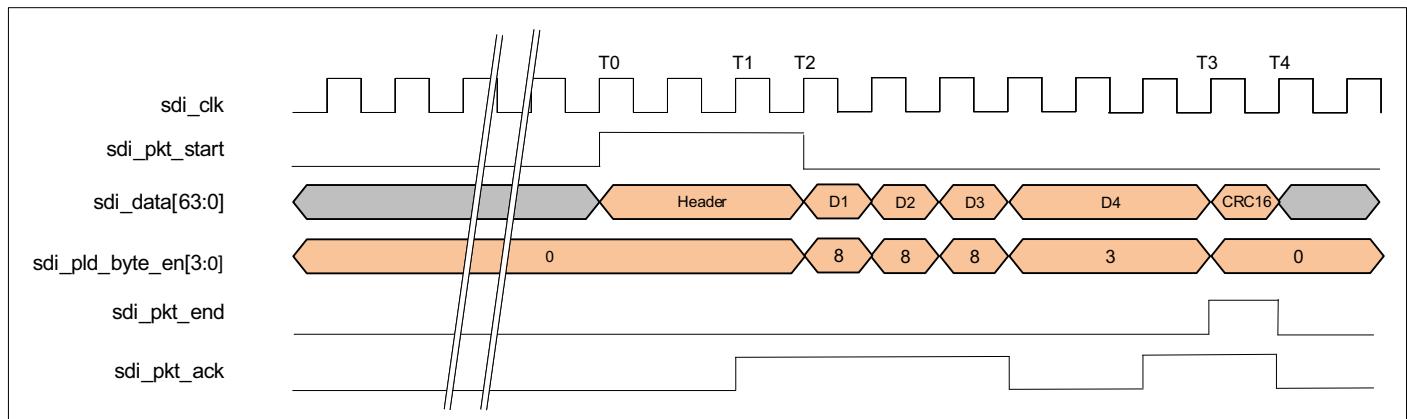
Notes:

At T0: When there is a short packet to transmit, assert sdi_pkt_start. At the same time, send sdi_data to the controller. This packet is a short packet, sdi_pkt_end should be asserted.

At T1: When sdi_pkt_data_ack is asserted, indicate the sdi_data is received by controller.

At T2: When sdi_pkt_data_ack is asserted, sdi_pkt_start and sdi_pkt_end should be de-asserted. You can then send the next packet.

Figure 2-57 Long Packet Transmission Timing



Notes:

At T0: When there is a long packet to transmit, assert sdi_pkt_start. At the same time, send sdi_data to the controller.

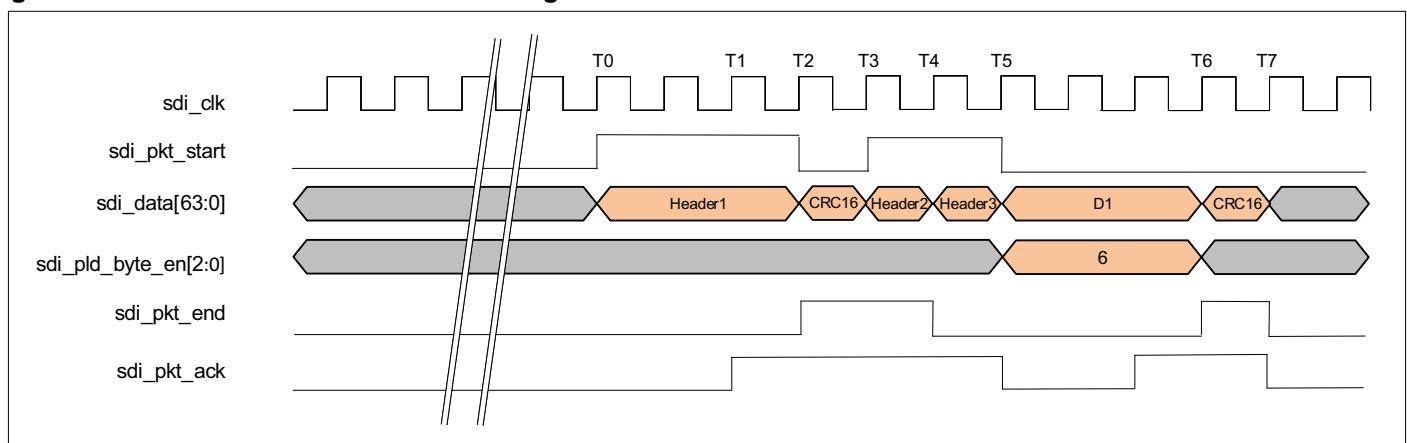
At T1: If controller can receive the sdi data, assert sdi_pkt_data_ack.

At T2: When sdi_pkt_data_ack is asserted, User Logic must send next sdi_data and sdi_pld_byte_en to the controller due HS transmission has begun.

At T3: When sdi_pkt_data_ack is asserted, User Logic must send next sdi_data and sdi_pld_byte_en to the controller due HS transmission has begun. If this sdi_data is PF_CRC, sdi_pkt_end should be asserted.

At T4: When sdi_pkt_data_ack is asserted, sdi_pkt_end should be de-asserted. User Logic can send next packet or stop to send packet.

Figure 2-58 Additional Transmission Timing



Notes:

At T0: When there is a long packet to transmit, assert sdi_pkt_start. At the same time, send sdi_data to the controller.

At T1: When sdi_pkt_data_ack is asserted, indicate the sdi_data is received by controller.

At T2: When sdi_pkt_data_ack is asserted, User Logic must send next sdi_data and sdi_pld_byte_en to the controller due HS transmission has begun. This packet is a long packet with word counter 0, sdi_data is PF_CRC and sdi_pkt_end should be asserted.

At T3: When sdi_pkt_data_ack is asserted and there is next short packet to transmit, assert sdi_pkt_start. At the same time, send sdi_data to the controller. This packet is a short packet, sdi_pkt_end should be asserted.

At T4: When sdi_pkt_data_ack is asserted and there is next long packet to transmit, assert sdi_pkt_start. At the same time, send sdi_data to the controller.

At T5: When sdi_pkt_data_ack is asserted, User Logic must send next sdi_data and sdi_pld_byte_en to the controller due HS transmission has begun.

At T6: When sdi_pkt_data_ack is asserted, User Logic must send next sdi_data and sdi_pld_byte_en to the controller. If this sdi_data is PF_CRC, sdi_pkt_end should be asserted.

At T7: When sdi_pkt_data_ack is asserted, sdi_pkt_end should be de-asserted. User Logic can send next packet.

Error check

In SDI, the controller checks the following errors.

- Incorrect Frame/Line Sequence
- Inconsistency between the byte number of the received payload and the word count filed in packet header.
- Inconsistency between the internal ECC/PH-CRC generation and ECC/PH-CRC of SDI bus in ECC/PH-CRC Bypass mode. For this error, the ECC/PH-CRC field which is even wrong in SDI bus will still be received and transmitted by the controller.
- Inconsistency between the internal packet foot CRC generation and PF-CRC of SDI bus in ECC/PF-CRC16 Bypass mode. For this error, the PF_CRC field which is even wrong in SDI bus will be received and transmitted by controller.

2.6.3 Registers Related to SDI

Following are the registers related to SDI:

- SDI_CFG
- INT_ST_SDI
- INT_ST_SDI_VCX
- INT_ST_SDI_VCX2

2.6.4 Signals Related to SDI

Following are the signals related to SDI:

- sdi_clk
- sdi_clkrstz
- sdi_pkt_start
- sdi_pkt_end

- sdi_data
- sdi_pld_byte_en
- sdi_pkt_data_ack

2.6.5 Programming SDI

To programming SDI, only need to configure SDI_CFG register for Payload Format/ECC/CRC Mode during the “Controller Configuration” phase.

For more information about “Controller Configuration” phase, refer to the *MIPI CSI-2 Device Controller User Guide*.

2.7 Video Pattern Generator

This section describes the Video Pattern Generator (VPG) of DWC_mipicsi2_device.

2.7.1 Overview of the Video Pattern Generator

The VPG allows the transmission of horizontal/vertical color bar and D-PHY Bit Error Rate testing pattern without any stimuli.

The functions of video pattern generator are:

- Helps output a color bar by the controller without any kind of stimuli.
- It transmits a static color bar image and supports both horizontal and vertical orientation.
- It supports RAW8 and RGB888 Bit Error Rate.
- Supports all color codings.

The frame requirements must be defined in video pattern registers that are listed in [Table 2-4](#)

Table 2-4 Frame Requirement Configuration Registers

Register Name	Address Offset	Access Type	Description	Value after Reset
VPG_MODE_CFG		R/W	Test Mode configuration	0x00000000
VPG_PKT_CFG		R/W	Video pattern Configuration	0x00000000
VPG_PKT_SIZE		R/W	Video packet size	0x00000000
VPG_HSA_TIME		R/W	Horizontal Sync Active time	0x00000000
VPG_HBP_TIME		R/W	Horizontal Back Porch time	0x00000000
VPG_HLINE_TIME		R/W	Line time	0x00000000
VPG_VSA_LINES		R/W	Vertical Sync Active period	0x00000000
VPG_VBP_LINES		R/W	Vertical Back Porch period	0x00000000
VPG_VFP_LINES		R/W	Vertical Front Porch period	0x00000000
VPG_ACT_LINES		R/W	Vertical resolution	0x00000000
VPG_MAX_FRAME_NUM		R/W	Maximum Frame Number	0x00000000
VPG_START_LINE_NUM		R/W	Start Line Number	0x00000000
VPG_STEP_LINE_NUM		R/W	Step Line Number	0x00000000
VPG_BK_LINES		R/W	Frame Blanking period	0x00000000

2.7.2 Color Bar Pattern

The color bar pattern comprises eight color bars: white, yellow, cyan, green, magenta, red, blue, and black.

Calculate each color width by dividing the line pixel size (vertical pattern) or the number of lines (Horizontal pattern) by eight:

- In the vertical color bar mode, each single-color bar has a width equal to the number of pixels in a line divided by eight. In case the number of pixels in a line is not divisible by eight, the last color (black) contains the remaining pixels.
- In the horizontal color bar mode, each color line has a color width equal to the number of lines in a frame divided by eight. In case the number of lines in a frame is not divisible by eight, the last color (black) contains the remaining lines.

Figure 2-59 Vertical Color Bar Mode

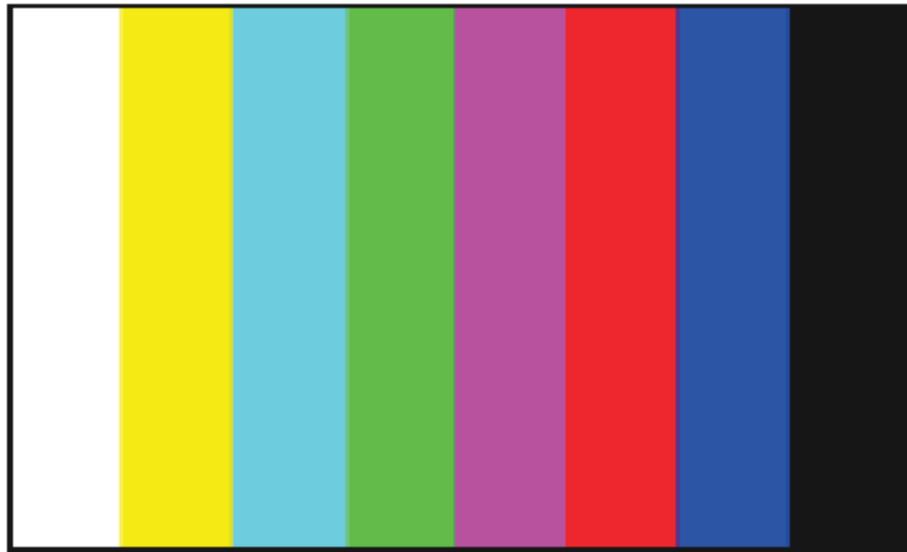


Figure 2-60 Horizontal Color Bar Mode



2.7.3 Color Coding

This section provides details on how DWC_mipicsi2_device handles color coding.

2.7.3.1 RAW Components

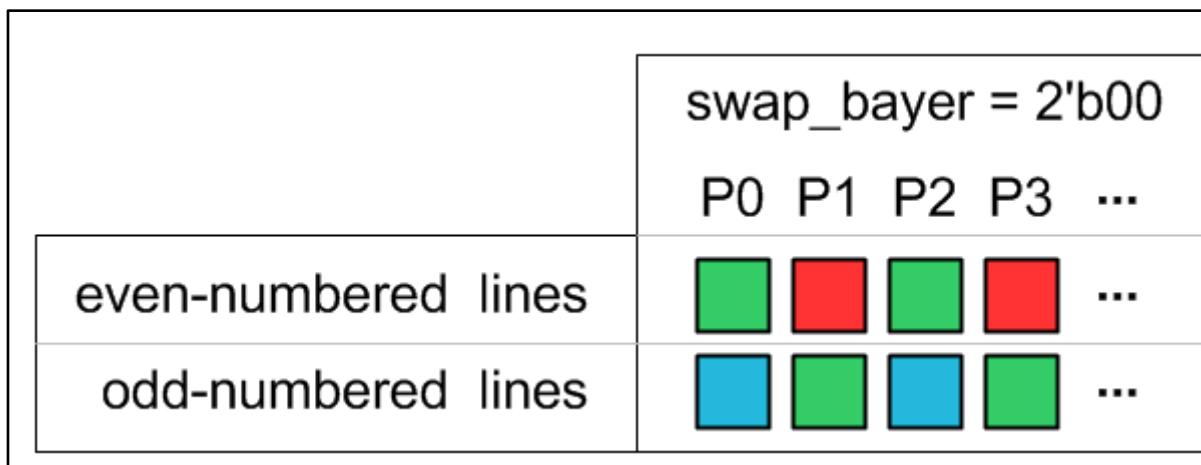
[Table 2-5](#) shows the RAW components used during the color coding.

Table 2-5 RAW Components

	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
R	Maximum	Maximum	Minimum	Minimum	Maximum	Maximum	Minimum	Minimum
G	Maximum	Maximum	Maximum	Maximum	Minimum	Minimum	Minimum	Minimum
B	Maximum	Minimum	Maximum	Minimum	Maximum	Minimum	Maximum	Minimum

For RAW data types, use `swap_bayer` to change the order of the color component.

Figure 2-61 Color Selection Made by `swap_bayer` for RAW Data Types



2.7.3.2 RGB Components

[Table 2-6](#) shows the RGB components used during color coding.

Table 2-6 RGB Components

	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
R	Maximum	Maximum	Minimum	Minimum	Maximum	Maximum	Minimum	Minimum
G	Maximum	Maximum	Maximum	Maximum	Minimum	Minimum	Minimum	Minimum
B	Maximum	Minimum	Maximum	Minimum	Maximum	Minimum	Maximum	Minimum

2.7.3.3 YUV Components

To avoid YUV negative components, YUV color coding is coded with the following offset:

- Y has an excursion of 219 and an offset of +16
- U and V have excursions of ± 112 and offset of +128

Table 2-7 shows YUV422 8-bit components.

Table 2-7 YUV422 8-bit Components

	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
Y [7:0]	235	210	170	145	106	81	41	16
U [7:0]	128	16	166	54	202	90	240	128
V [7:0]	128	146	16	34	222	240	110	128

Each component of the YUV422 10-bit loosely packed is calculated by multiplying the respective component of the YUV422 8-bit by 4.

Table 2-8 YUV422 10-bit Components

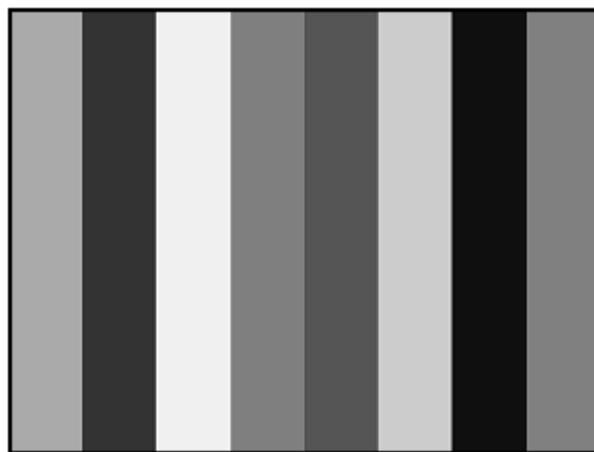
	White	Yellow	Cyan	Green	Magenta	Red	Blue	Black
Y [9:0]	940	840	680	580	424	324	164	64
U [9:0]	512	64	664	216	808	360	960	512
V [9:0]	512	584	64	136	888	960	440	512

2.7.4 BER Testing Pattern

The BER testing pattern simplifies conformance testing. This pattern tests the RX D-PHY capability to receive the data correctly. BER test supports RAW8 and RGB888 format. Requires the following data patterns:

- X bytes of 0xAA (high-frequency pattern, inverted)
- X bytes of 0x33 (mid-frequency pattern)
- X bytes of 0xF0 (low-frequency pattern, inverted)
- X bytes of 0x7F (lone 0 pattern)
- X bytes of 0x55 (high-frequency pattern)
- X bytes of 0xCC (mid-frequency pattern, inverted)
- X bytes of 0x0F (low-frequency pattern)
- Y bytes of 0x80 (lone 1 pattern)

In most cases, Y is equal to X. However, depending on line length and the color coding used, Y may be a different value than X. With RGB888 color coding and vertical resolution in multiples of eight, the pattern shown in Figure 2-62 appears on the DSI display.

Figure 2-62 RGB888 BER Testing Pattern

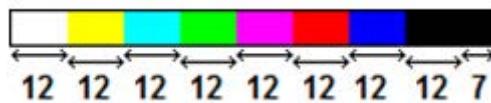
2.7.5 Video Pattern Generator Resolution

Depending on the orientation, BER mode, and color coding, the smallest resolutions accepted by the test mode pattern generator are:

- BER mode: 8x8
- Horizontal color bar mode: 8x8
- Vertical color bar mode: 8x8

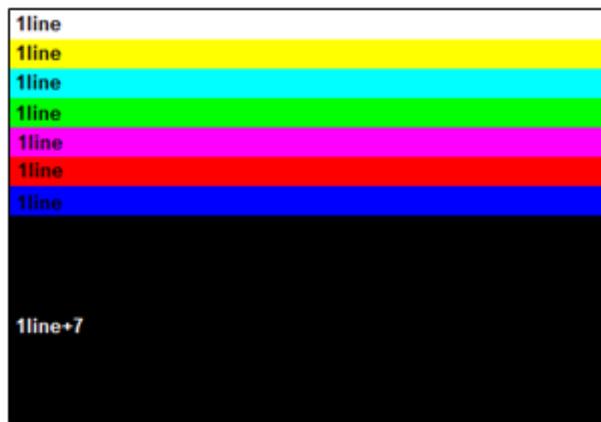
2.7.5.1 Vertical Pattern

The width of each color bar is determined by the division of vertical resolution (pixels) for eight video pattern colors. If the vertical resolution is not divisible by eight, the last color (black) extends to fill the resolution. In the example shown in [Figure 2-63](#), the vertical resolution is 103.

Figure 2-63 Vertical Pattern (103*15)

2.7.5.2 Horizontal Pattern

The width of each color bar is determined by the division of the number of horizontal resolution (lines) for eight video pattern colors. If the horizontal resolution is not divisible by eight, the last color (Black) is extended to fill the resolution.

Figure 2-64 Horizontal Pattern (103*15)

2.7.6 Enabling Video Pattern Generator

This feature is not active by default. You can enable the feature through register configuration. In the Color Bar mode, you can configure VPG_CTRL register to switch to either vertical or horizontal color bar image.

2.7.7 Registers Related to Video Pattern Generator

Following are the registers related to Video Pattern Generator:

- VPG_CTRL
- VPG_STATUS
- VPG_MODE_CFG
- VPG_PKT_CFG
- VPG_PKT_SIZE
- VPG_HSA_TIME
- VPG_HBP_TIME
- VPG_HLINE_TIME
- VPG_VSA_LINES
- VPG_VBP_LINES
- VPG_VFP_LINES
- VPG_ACT_LINES
- VPG_MAX_FRAME_NUM
- VPG_START_LINE_NUM
- VPG_STEP_LINE_NUM
- VPG_BK_LINES

2.8 Memory

DWC_mipicsi2_device supports to use asynchronous dual port memory or synchronous single port memory to implement FIFO for data storage. It can be selected by using the parameter CSI2_DEVICE_EXT_RAM_TYPE in coreConsultant GUI.

[Figure 2-65](#) shows the timing diagram of asynchronous dual port memory access.

Figure 2-65 Timing Representation of asynchronous dual port memory access

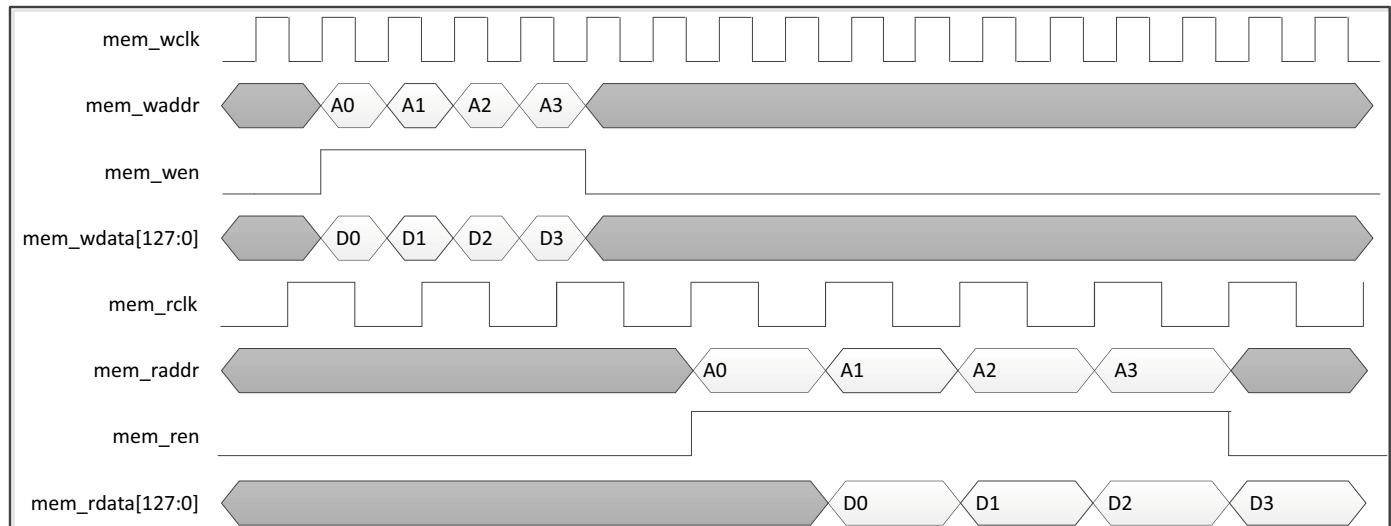
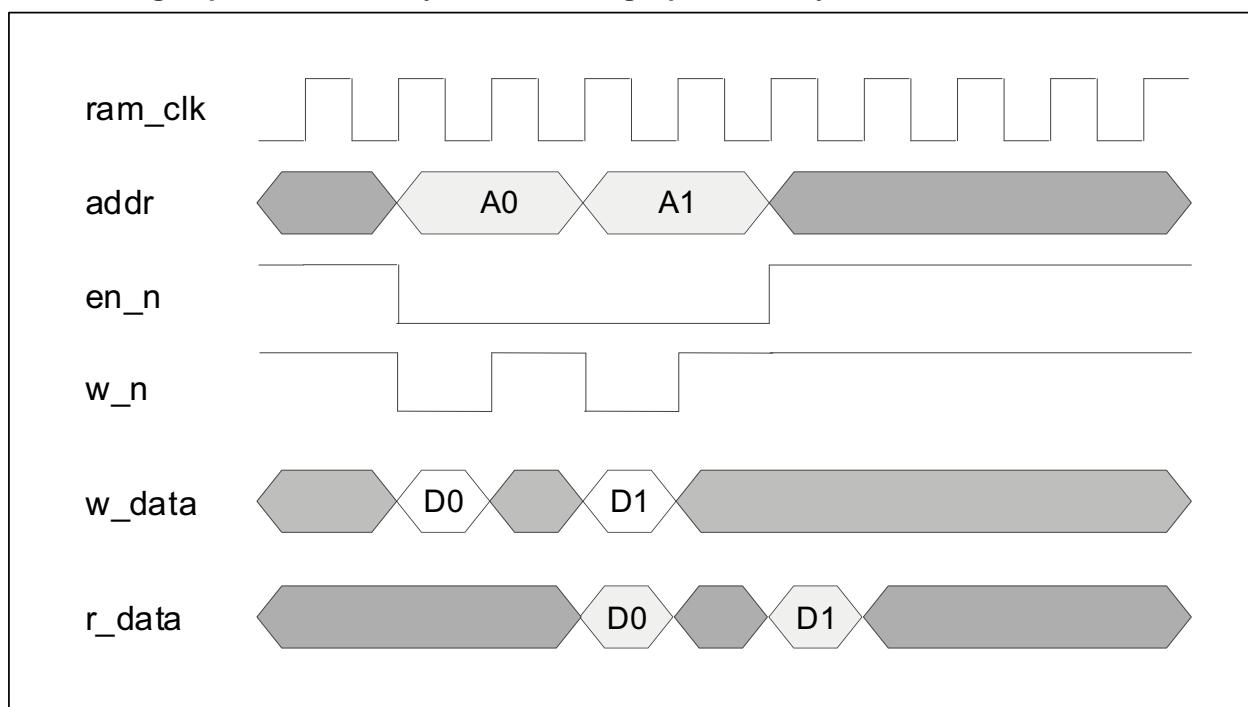


Figure 2-66 Timing Representation of synchronous single port memory



If synchronous single port memory is selected, then

1. The total FIFO depth should be even.
2. Two external SPRAMs are needed, $\text{odd_ram_size} = \text{even_ram_size} = \text{PTP_SPRAM_DEPTH}/2$.



Note The depth of Pseudo Two-Port (PTP) RAM will be automatically shown in cC GUI.

2.8.1 Image Data Interface (IDI) Minimum Memory Size

When not using the Backpressure feature, “[IDI Driver Timing Requirement](#)” on page 45 must be followed. The size of the 2-port RAMs must be at least the double size of long packet size.

When using Backpressure feature, there are no IDI timing requirements to follow. The size of the 2-port RAMs size must be at least one entire data line + $3*64$ bits.

[Table 2-9](#) summarizes the required 2-port RAM sizes.

Table 2-9 IDI 2-Port RAM Requirements

RAM Requirements	IDI Timing Requirement	2-port RAMs Size
Backpressure feature is enabled	No	<ul style="list-style-type: none"> ■ When the width of the payload FIFO is 64bits without Automotive Package function <ul style="list-style-type: none"> □ Long packet size + $3*64$ bits ■ When the width of the payload FIFO is 128bits without Automotive Package function <ul style="list-style-type: none"> □ Long packet size + $3*128$ bits
Backpressure feature is disabled	Yes	$2*\text{Long packet size}$

2.8.2 Image Pixel Interface (IPI) Memory Size

This section describes the memory size requirements for the IPI.

2.8.2.1 Minimum Memory Size

The requirements for the 2-port RAM size in various IPI modes are as follows:

- Store and Forward mode: at least double the pixel data packet size
- Cut-Through mode: equal to the pixel data packet
- Backpressure mode: equal to the pixel data packet + $5*64$ bits

[Table 2-10](#) summarizes the required 2-port RAM sizes.

Table 2-10 IPI 2-Port RAM Requirements

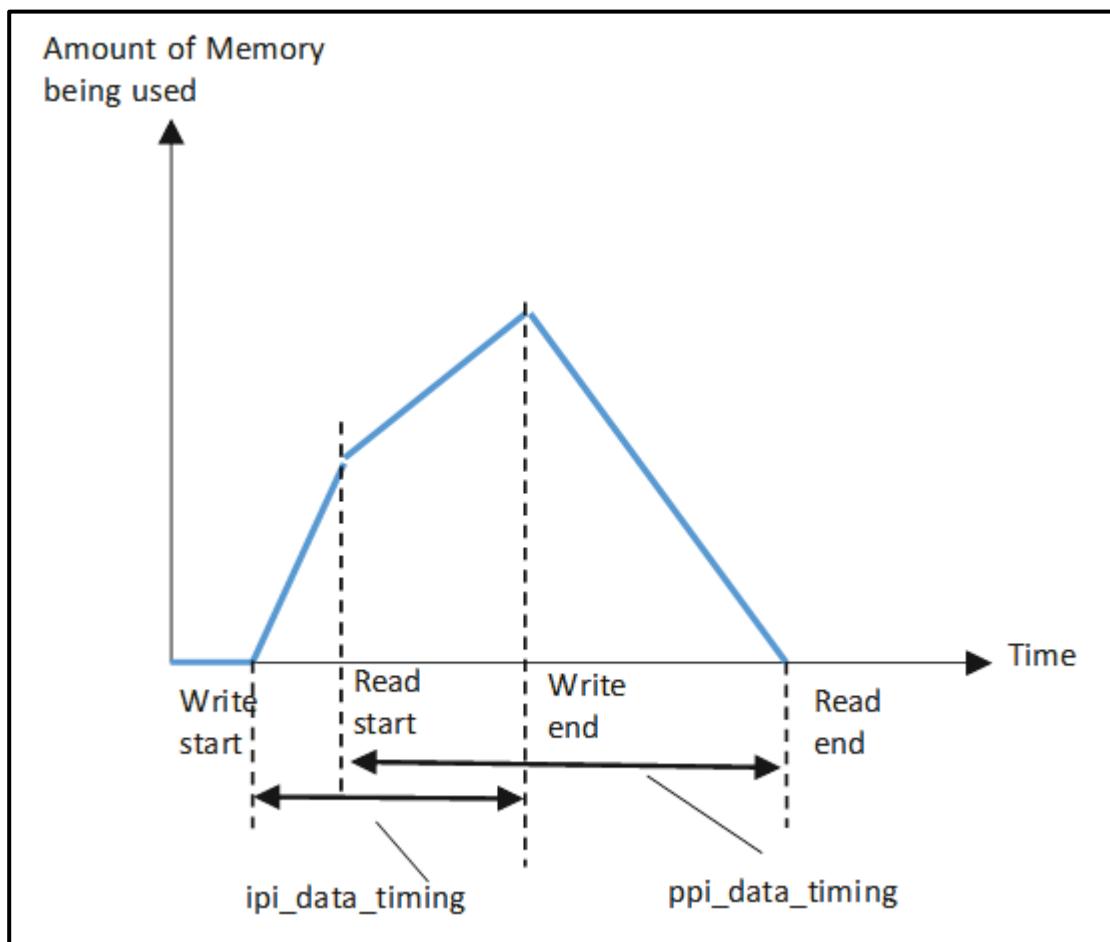
Mode	IPI Timing Requirements	2-Port RAM Size
Store and Forward Mode	Yes	$2*\text{Long packet size}$

Mode	IPI Timing Requirements	2-Port RAM Size
Cut-Through mode	Yes	<p>Long packet size</p> <p>Note: This is a safe value. In "Minimum memory calculation of IPI Cut-Through mode", the formula is used to calculate the Minimum value for LRTE Disable Mode ONLY.</p>
Backpressure Mode	No	<ul style="list-style-type: none"> ■ When the width of the payload FIFO is 64bits without Automotive Package function <ul style="list-style-type: none"> □ Long packet size + 5*64bits ■ When the width of the payload FIFO is 128bits without Automotive Package function <ul style="list-style-type: none"> □ Long packet size + 5*128bits

2.8.2.2 Minimum Memory Calculation of IPI Cut-Through Mode with LRTE Disable

Figure 2-67 shows when the writing FIFO is faster than reading FIFO, which corresponds to the following calculations:

- For 64-bit FIFO width:
 $\text{ipi_data_timing} - \text{ppi_data_timing} \leq 12$
- For 28-bit FIFO width:
 $\text{ipi_data_timing} - \text{ppi_data_timing} \leq 24$

Figure 2-67 Writing FIFO Faster than Reading FIFO

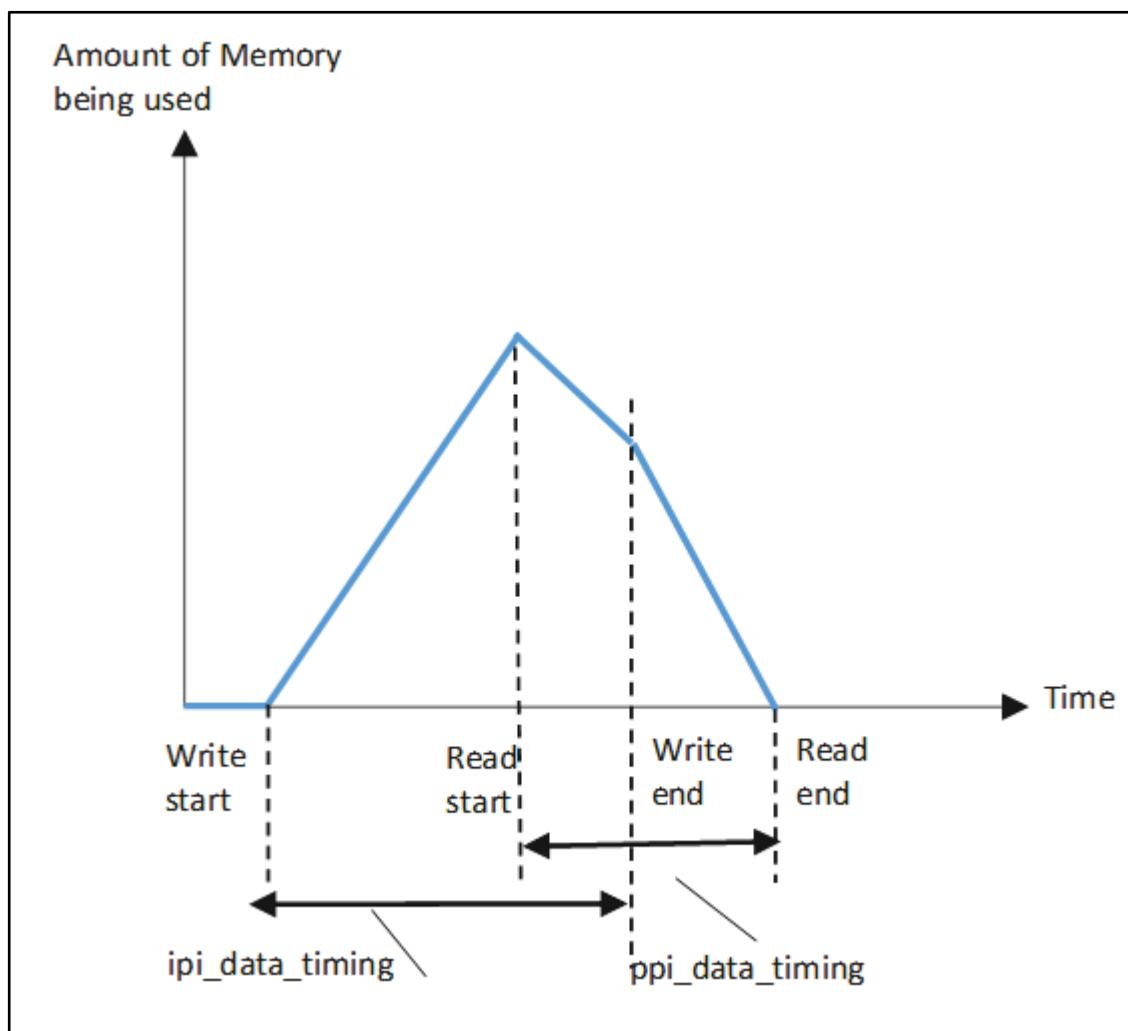
```
min_memory_size = data_packet_size - ((ipi_data_timing - ipi_data_send_start - Internal_latency_ipi) * Tipiclk/Tlanebyteclk - Internal_latency_ppi - PHY_LP2HS_time) * phy_lane_width * number of active lines
```

Where:

- `data_packet_size` = the bit number of one data packet
- `Internal_latency_ipi` = 20, the internal latency between packet generation and the packet delivery in `ipi_clk`
- `Internal_latency_ppi` = 20 + synchronization stages, the internal latency between packet generation and the packet delivery in `ppi_clk`

Figure 2-68 shows when the reading FIFO is faster than writing FIFO, which corresponds to the following calculation:

- For 64-bit FIFO width:
 $\text{ipi_data_timing} - \text{ppi_data_timing} > 12$
- For 128-bit FIFO width
 $\text{ipi_data_timing} - \text{ppi_data_timing} > 24$

Figure 2-68 Reading FIFO Faster than Writing FIFO

```
min_memory_size = (ipi_data_send_start + Internal_latency_ipi+(Internal_latency_ppi
+ PHY_LP2HS_time)/Tipiclk)* Pixel_per_cycles *Bits_per_pixel
```

Where:

- Pixel_per_cycles = pixels per ipi_clk cycle
- Bits_per_pixel = Bits per pixel



Note For the following color coding, the Pixel line considered must be for the even line:

- CSI2_Y420_8B
- CSI2_CSFS_Y420_8B
- CSI2_CSFS_Y420_10B
- CSI2_Y420_10B

2.8.3 Memory Size for Automotive Safety

For memory requirements related to automotive safety, see [Chapter B.6, “Automotive Package Safety Memory Requirements” chapter](#).

2.9 Protocol-PHY Interface (PPI)

The controller can support either C-PHY or D-PHY interfaces.

For timing of D-PHY interface or C-PHY interface, refer to D-PHY and C-PHY specifications.

Data on the PPI interface can be scrambled to reduce EMI as described in “[Data Scrambling](#)” on page [105](#).

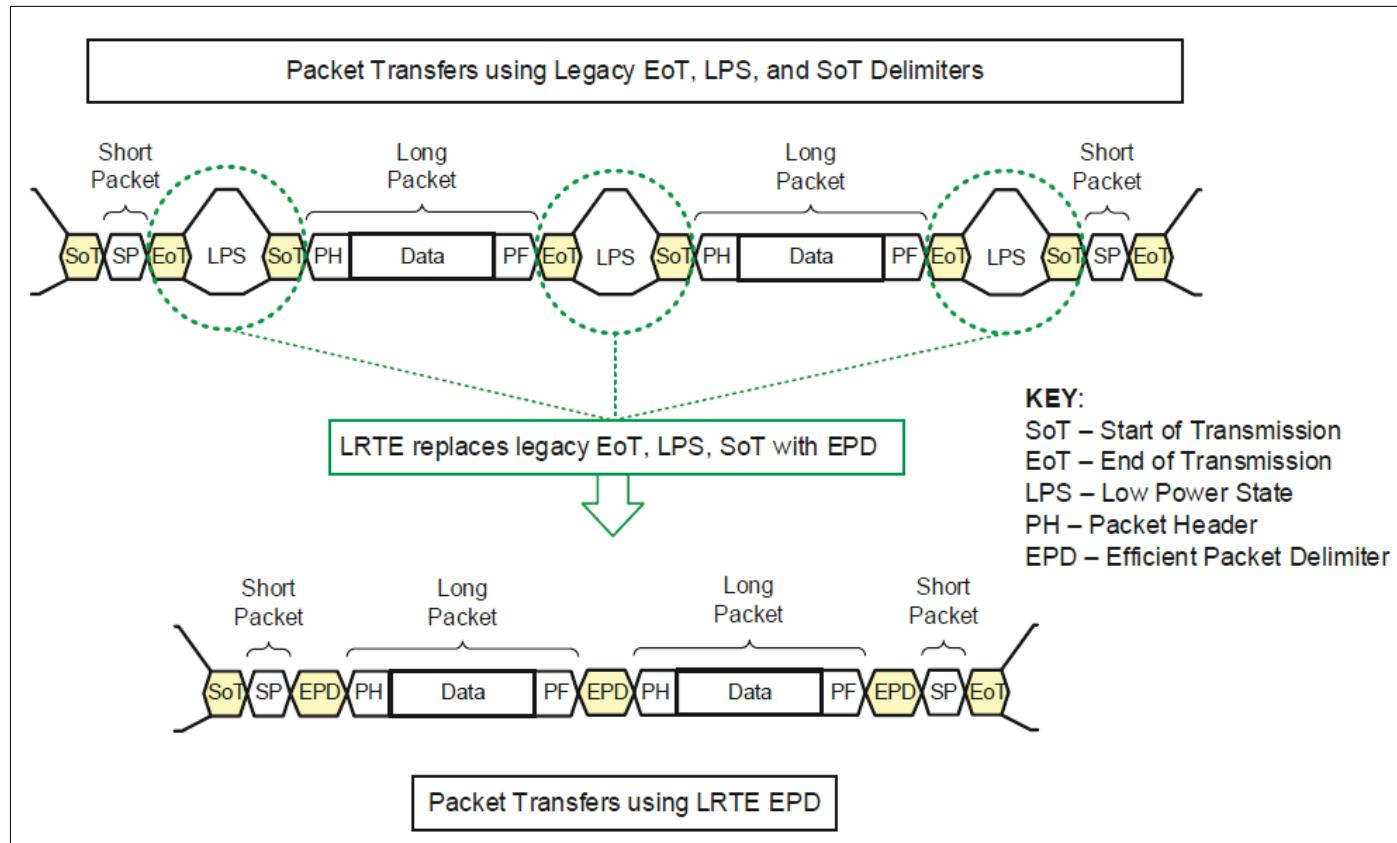
2.10 LRTE Support

Latency Reduction and Transport Efficiency (LRTE) is an optional CSI-2 feature that facilitates optional transport, in order to support a number of emerging imaging applications.

2.10.1 Inter-packet Latency Reduction (ILR)

Inter-packet latency reduction (ILR) replaces legacy EoT, LPS and SoT packet delimiters with a more Efficient Packet Delimiter (EPD) signaling mechanism that avoids the need for high-speed-low-power-high-speed transitions.

Figure 2-69 Inter-packet Latency Reduction Using LRTE EPD

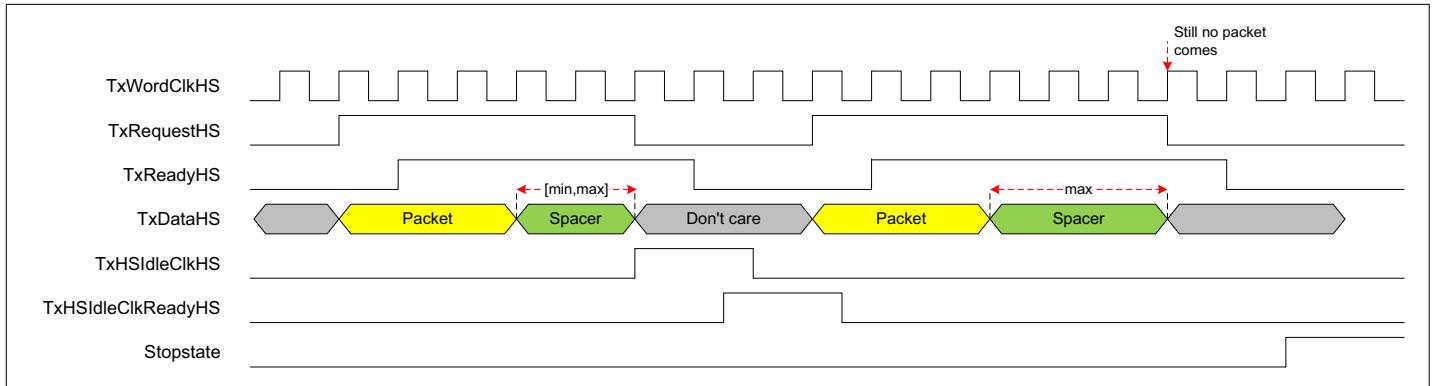


2.10.1.1 D-PHY EPD Option 1

For CSI2 v3.0 specification or later, insertion of Spacers-without-PDQ after CSI-2 packets just before D-PHY EoT is register programmable. Timing is shown as in [Figure 2-70](#) and [Figure 2-71](#).

Enable Spacers-without-PDQ

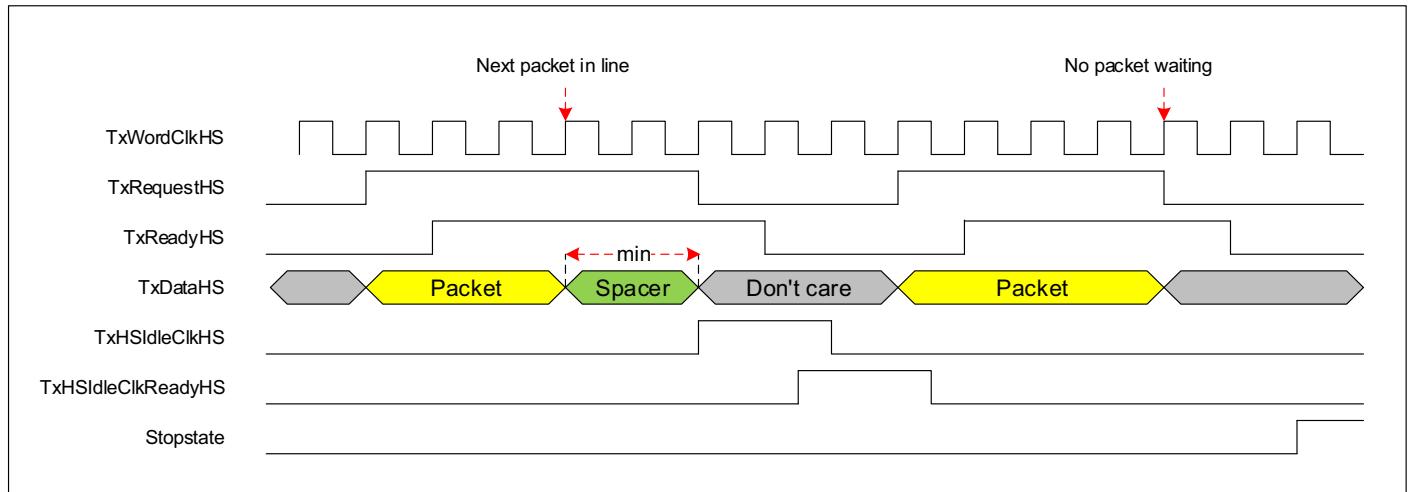
Figure 2-70 Timing of D-PHY EPD Option 1 with Spacers-without-PDQ Enabled



1. Every packet is followed by spacers. Minimum and maximum number of spacers are register programmable.
2. Minimum number of spacers are sent before requesting HS-Idle if next packet comes before minimum number of spacers are finished.
3. If next packet comes between minimum and maximum number of spacers, controller immediately stops sending spacers and requests PHY for HS-Idle.
4. If next packet does not come until maximum number of spacers are finished, controller stops transmission and lets PHY go to Stop State.

Disable Spacers-without-PDQ

Figure 2-71 Timing of D-PHY EPD Option 1 with Spacers-without-PDQ Disabled



When Spacers-without-PDQ is disabled,

1. Only minimum number of spacers need to be programmed.
2. If next packet comes before current packet is finished sending, then minimum number of spacers are be added and after that controller will require for HS-Idle. Otherwise, no spacer will be added, and PHY goes to Stop State after current packet is finished.

2.10.1.2 D-PHY EPD Option 2

D-PHY EPD Option 2 is limited to optional CSI-2 protocol-generated and CSI-2 protocol-consumed Spacers for back-to-back transfers (there is no use of PHY-generated and PHY-consumed PDQ).

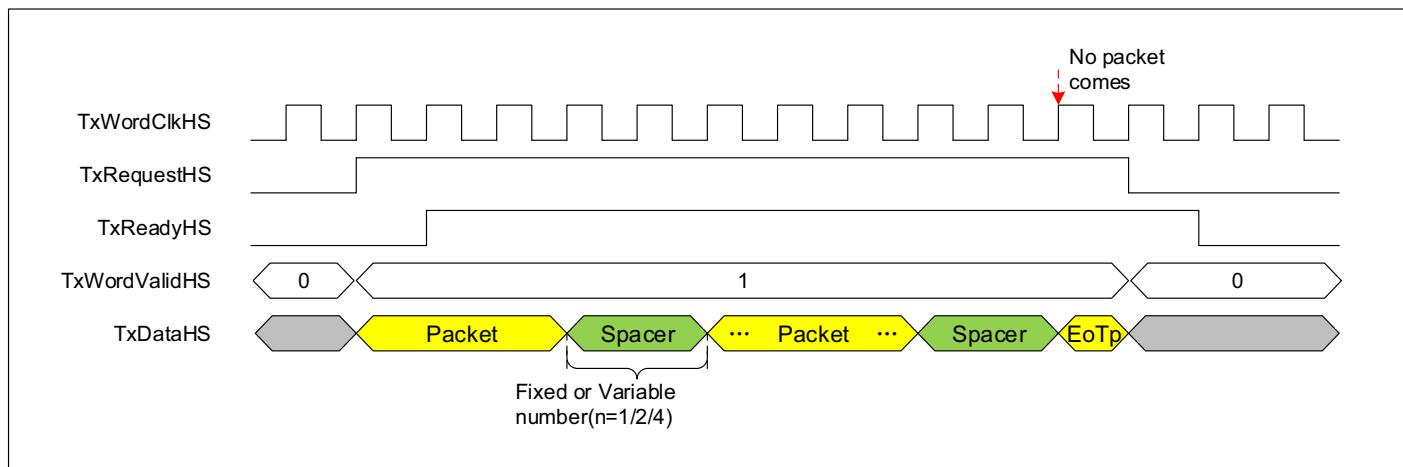
Option 2 is primarily intended for use with legacy D-PHYs not supporting Option 1. Currently, controller only support EPD Option 2 under D-PHY 8-bit Mode.

For D-PHY EPD Option 2, the End-of-Transmission short packet (EoTp) removes the need for the CSI-2 protocol receiver to perform EoT processing by unambiguously signaling the last CSI-2 packet in a D-PHY high-speed burst payload.

Enabling or disabling EoTp is register programmable. Timing is shown in [Figure 2-72](#) and [Figure 2-73](#).

Enable EoTp Packet

Figure 2-72 Timing of D-PHY EPD Option 2 with EoTp Enabled

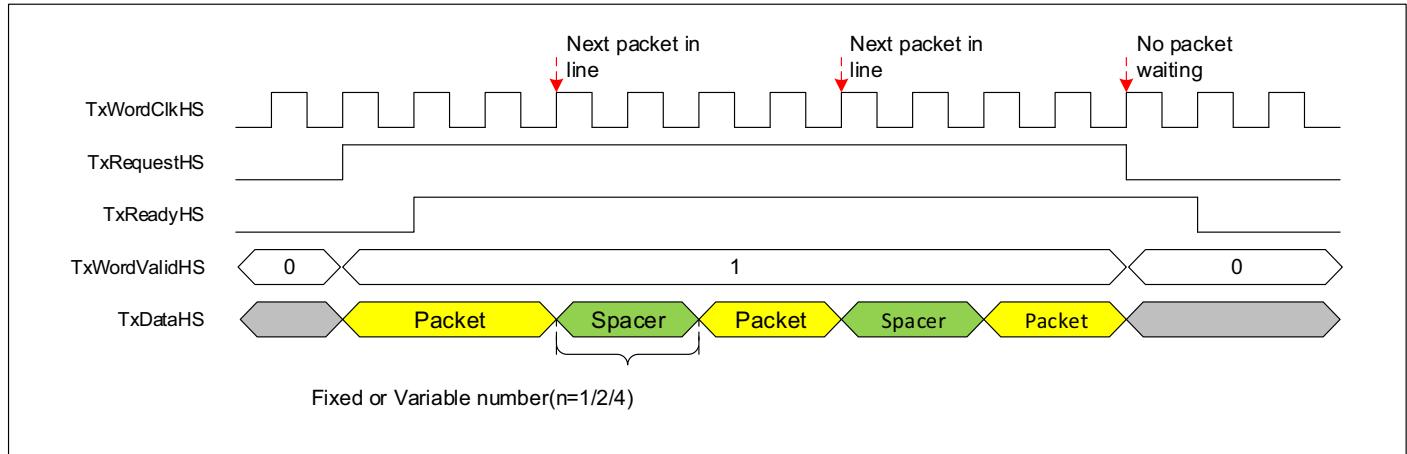


When EoTp packet is enabled,

1. Every packet is followed by spacers.
2. If fixed number of spacers mode is selected, then only minimum number of spacers are appended to packet. And if next packet still doesn't come after minimum number of spacers are finished, then EoTp packet will be inserted to finish current HS transmission.
3. If variable number of spacers mode is selected, controller will send spacers between minimum and maximum number. When spacer number is reaching maximum and still no packet comes, then EoTp will be sent and finish current HS transmission.

Disable EoTp Packet

Figure 2-73 Timing of D-PHY EPD Option 2 with EoTp Disabled



When EoTp packet is disabled,

1. Only fixed or minimum number of spacers need to be programmed.
2. If next packet comes before current packet is finished sending, then minimum number of spacers are added. Otherwise, no spacer are added, and PHY goes to Stop state after the current packet is finished.

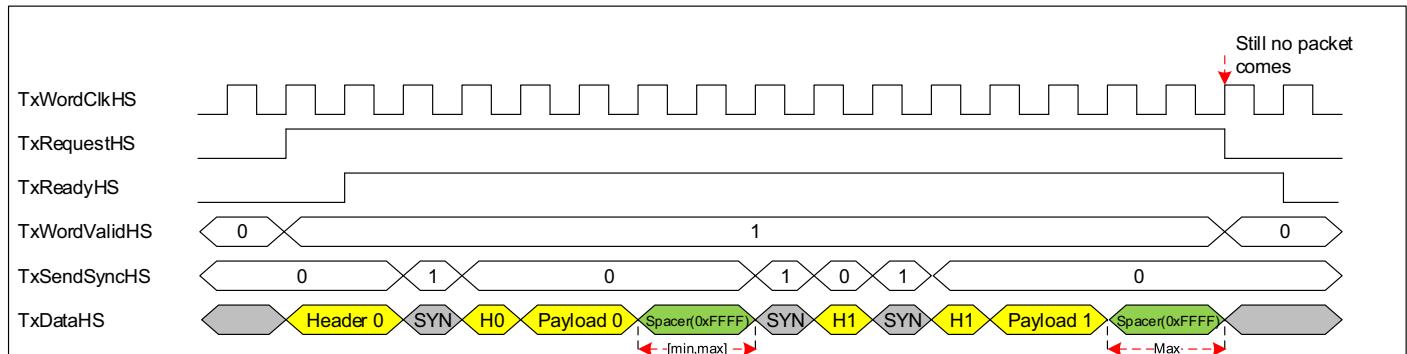
2.10.1.3 C-PHY EPD

The EPD for the C-PHY physical layer option uses one or more instances of the PHY-generated and PHY-consumed 7-UI Sync Word for the PDQ signaling. Controller should reuse TxSendSyncHS at the PPI in order to generate the PDQ control code by the C-PHY transmitter.

Similar to the D-PHY EPD Option 1, insertion of Spacers-without-PDQ after CSI-2 packets just prior to C-PHY EoT is register programmable. Timing is shown in Figure 2-74 and Figure 2-75 below.

Enable Spacers-without-PDQ

Figure 2-74 Timing of C-PHY EPD with Spacers-without-PDQ Enabled

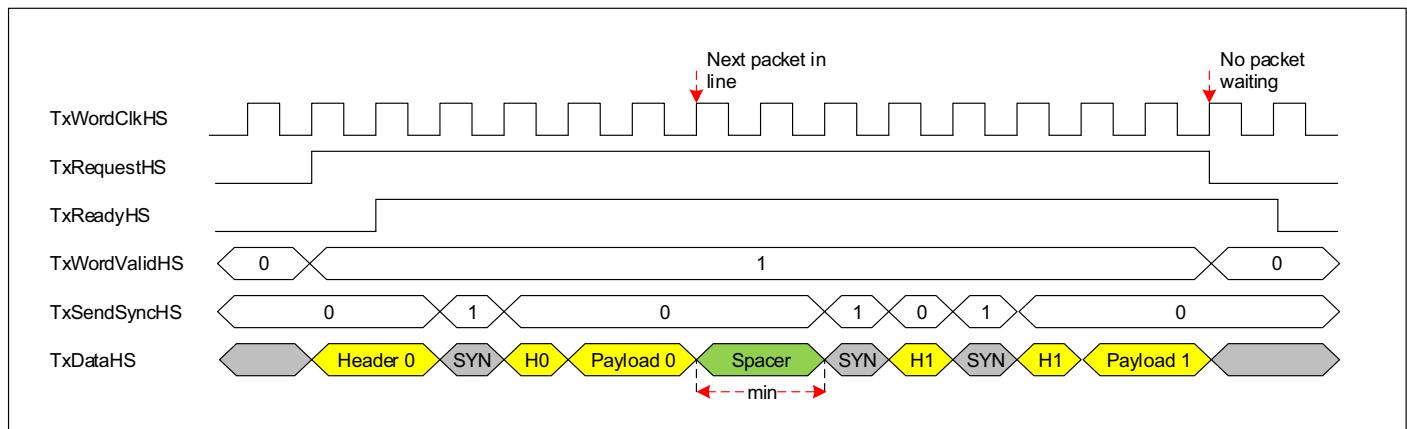


When Spacers-without-PDQ is enabled,

1. Every packet will be followed by spacers. Minimum and maximum number of spacers are register programmable.
2. Minimum number of spacers will be sent if next packet comes before minimum number of spacers are finished.
3. If next packet comes between minimum and maximum number of spacers, controller will immediately stop sending spacers and start transmission of next packet.
4. If next packet does not come until maximum number of spacers are finished, controller will stop transmission and let PHY go to Stop State.

Disable Spacers-without-PDQ

Figure 2-75 Timing of C-PHY EPD with Spacers-without-PDQ Disabled



When Spacers-without-PDQ is disabled,

1. Only minimum number of spacers need to be programmed.
2. If next packet comes before current packet is finished, then minimum number of spacers will be added and controller will start transmission of next packet after that. Otherwise, no spacer will be added, and PHY will go to Stop State after current packet is finished.

2.10.1.4 EPD Modes Supported by Controller

In summary, the modes of EPD supported by the controller are shown in [Table 2-11](#).

Table 2-11 EPD Modes Supported by the Controller

TYPE	Mode	Spacer Number	Usage
D-PHY Option 1	Enable Spacers-without-PDQ	[min, max]	SDI/IDI/IPI
	Disable Spacers-without-PDQ	min	SDI/IDI

TYPE	Mode		Spacer Number	Usage
D-PHY Option 2 (8-bit ONLY)	Enable EoTp	Fixed	min	SDI/IDI
		Variable	[min, max] (1n/2n/4n)	SDI/IDI/IPI
	Disable EoTp	Fixed	min	SDI/IDI
		Variable	[min, max] (1n/2n/4n)	SDI/IDI
C-PHY	Enable Spacers-without-PDQ		[min, max]	SDI/IDI/IPI
	Disable Spacers-without-PDQ		min	SDI/IDI



- When maximum value needs to be programmed, then: max > min.
- For DPHY EPD Option 2:
 - min \geq 8.
 - Both the minimum and maximum need to satisfy multiple relationships when 1n/2n/4n is programmed in variable mode.

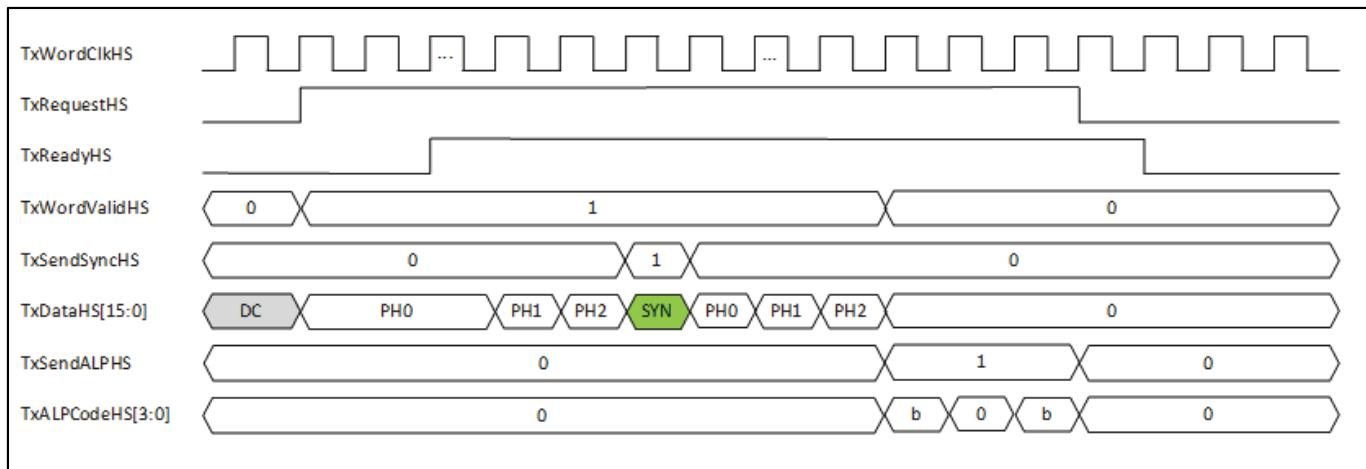
2.10.1.5 Related Registers

EPD related registers:

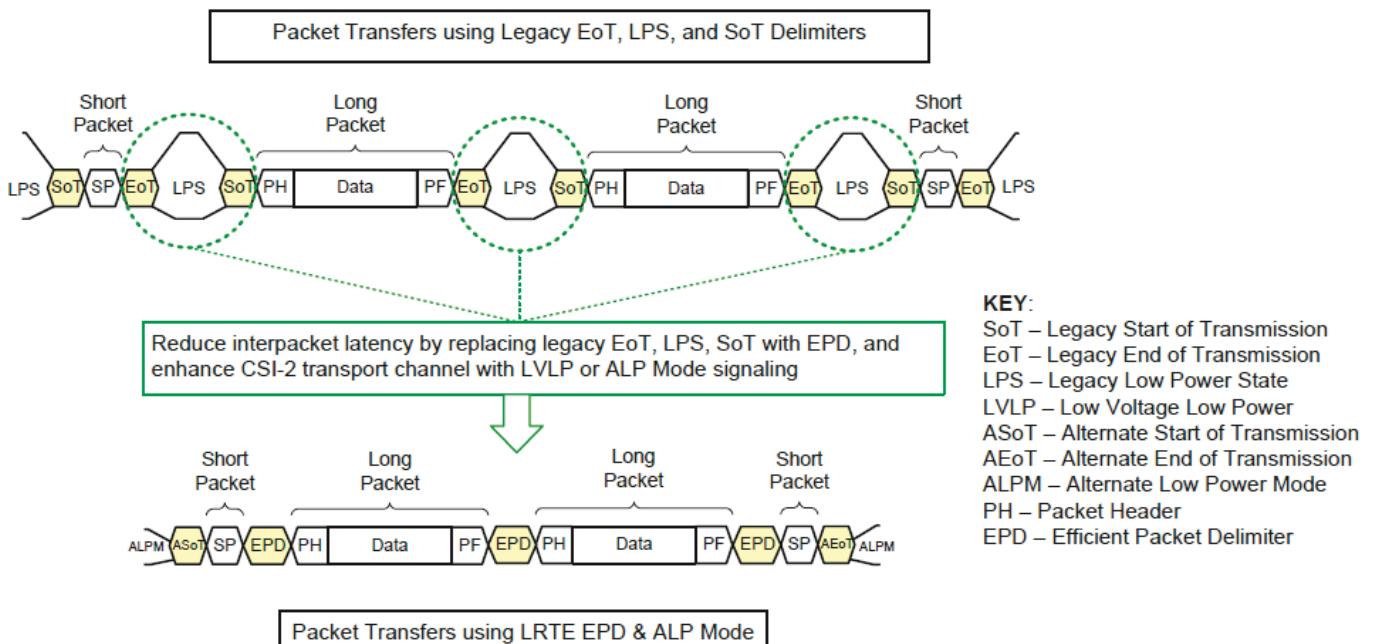
- TX_REG_CSI_EPD_EN_SSP (0x348)
- TX_REG_CSI_EPD_OP_SLP (0x34C)
- TX_REG_CSI_EPD_MISC_OPTIONS (0x350)
- TX_REG_CSI_EPD_MAX (0x354)

2.10.2 Enhanced Transport Efficiency

C-PHY ALPs are now provided to enhanced channel transport.

Figure 2-76 Overview of C-PHY ALPs after Burst Transmission

[Figure 2-76](#) shows ALP command after a high-speed Data Burst. EPD and ALPs, the two LRTE provisions may be used together in order to benefit from CSI-2 ILR and enhanced channel transport.

Figure 2-77 Using EPD and ALPs Together

2.11 Data Scrambling

This section describes the data scrambling feature of the DWC_mipicsi2_device controller.

2.11.1 Overview of Data Scrambling

Data scrambling is used to mitigate the effects of EMI and RF self-interference by spreading the transmission energy of the link over a possibly large frequency band, using a data randomization technique. The data that is being transmitted is scrambled with a pseudo-random-bit-stream (PRBS) to reduce the likelihood of a repetitive pattern on a lane basis.

The PRBS is generated using the Galois form of a Linear Feedback Shift Register (LFSR) implementing the generator polynomial:

$$G(x) = x^{16} + x^5 + x^4 + x^3 + 1$$

Each data lane has its own instance of a scrambler block and a different 16-bit LFSR seed defined by the CSI2 protocol.

[Figure 2-78](#) and [Figure 2-80](#) list the seed values after the controller resets.

Table 2-12 D-PHY Scrambler PRBS Initial Seed Values for Lanes 1 Through 8

Lane	Initial Seed Value
1	0x0810
2	0x0990
3	0x0a51
4	0x0bd0
5	0x0c30
6	0x0db0
7	0x0e70
8	0x0ff0

Table 2-13 C-PHY Scrambler PRBS Initial Seed Values for Lanes 1 Through 8

Lane	Initial Seed Value			
	Sync Type 0	Sync Type 1	Sync Type 2	Sync Type 3
1	0x0810	0x0001	0x1818	0x1008
2	0x0990	0x0180	0x1998	0x1188
3	0x0a51	0x0240	0x1a59	0x1248
4	0x0bd0	0x03c0	0x1bd8	0x13c8
5	0x0c30	0x0420	0x1c38	0x1428
6	0x0db0	0x05a0	0x1db8	0x15a8

Lane	Initial Seed Value			
	Sync Type 0	Sync Type 1	Sync Type 2	Sync Type 3
7	0x0e70	0x0660	0x1e79	0x1668
8	0x0ff0	0x07e0	0x1ff8	0x17e8

Data is scrambled when `txdatavalidhs` is high, and the LFSR seed is reset before its rising edge.

The C-PHY has the following features:

- Re-initialize the LFSR seed when `txsendsynchs` is high
- Two modes for choosing synchronize type, which can be configured through DATA_SCRAMBLING registers
 - Not single seed mode: Randomly choose Sync Type 1/2/3/4 when `txsendsynchs` is high
 - Single seed mode: Only Sync Type 3 used

[Figure 2-78](#) and [Figure 2-80](#) show the details of scrambling and seed re-initialization timing.

Figure 2-78 D-PHY Scrambling and Re-initialization Timing with LRTE Disabled

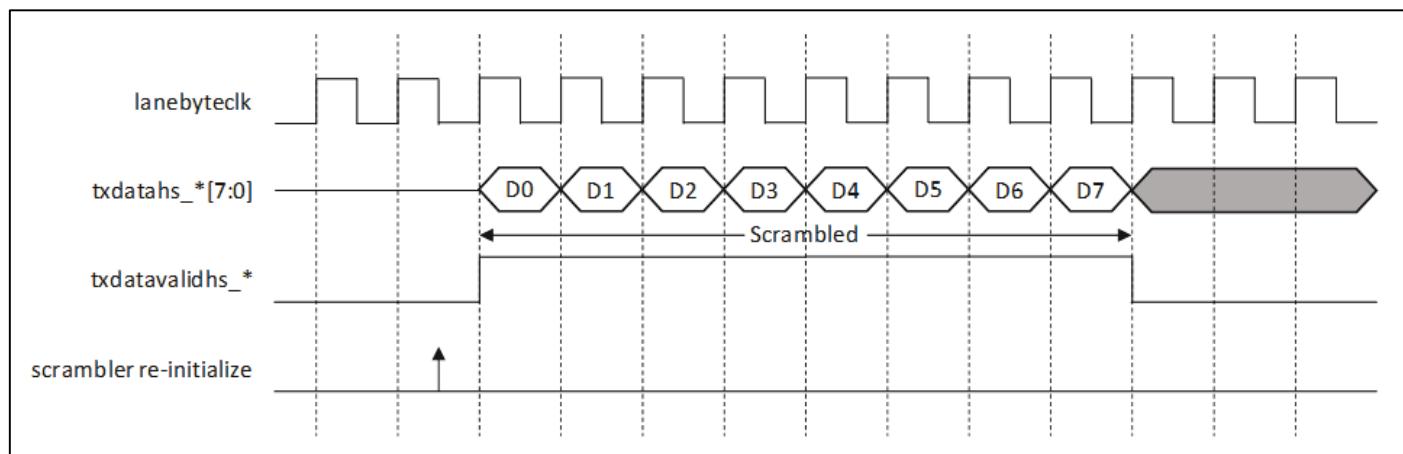
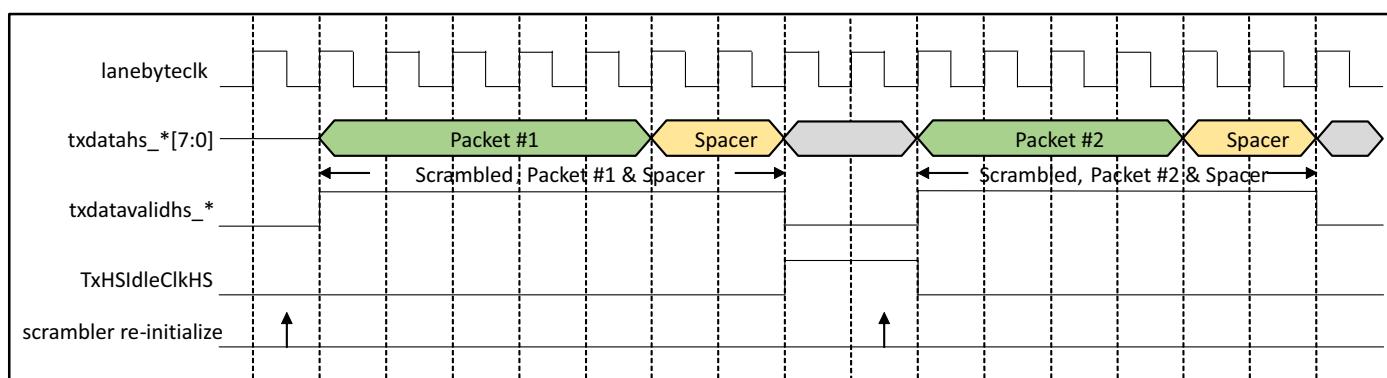


Figure 2-79 D-PHY Scrambling and Re-initialization Timing with LRTE Enabled



When D-PHY LRTE is enabled, the scrambler must be initialized under the following conditions:

1. At the beginning of the burst for both D-PHY EPD Option 1 and Option 2
 2. Prior to the first byte transmitted following high-speed-Idle for D-PHY EPD Option 1.
- The scrambler is not reinitialized between CSI-2 packets when using D-PHY EPD Option 2.

Figure 2-80 C-PHY Scrambling and Re-initialization Timing with LRTE Disabled

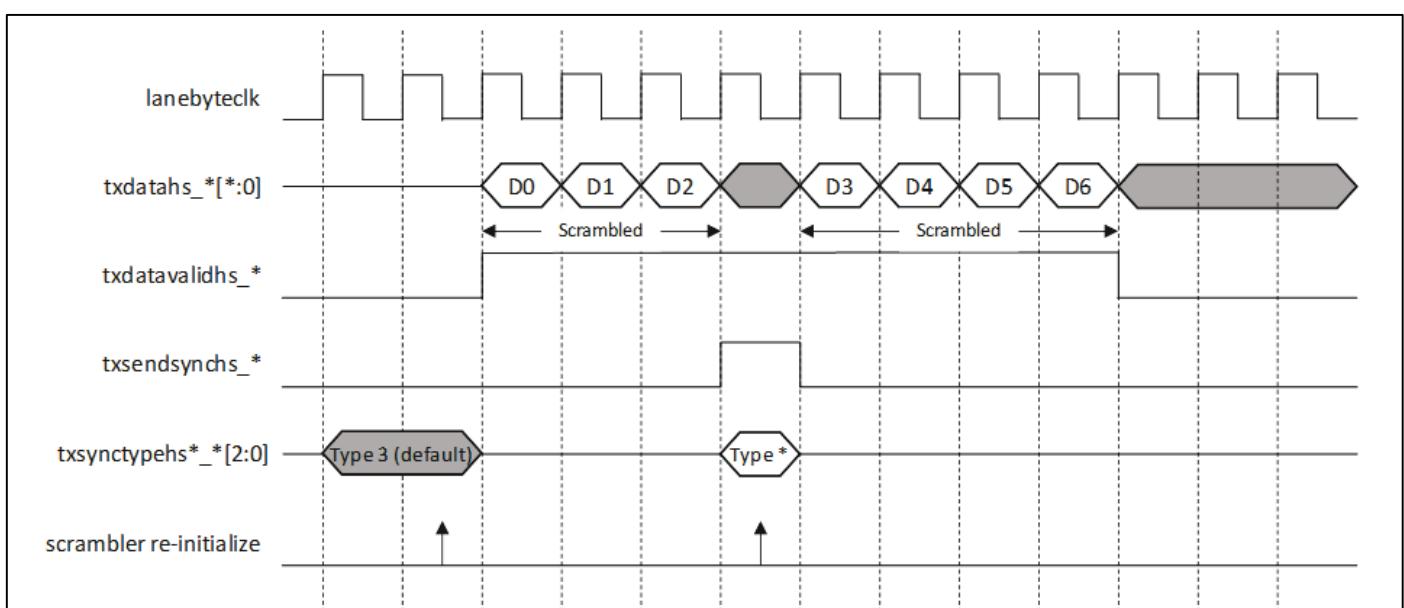
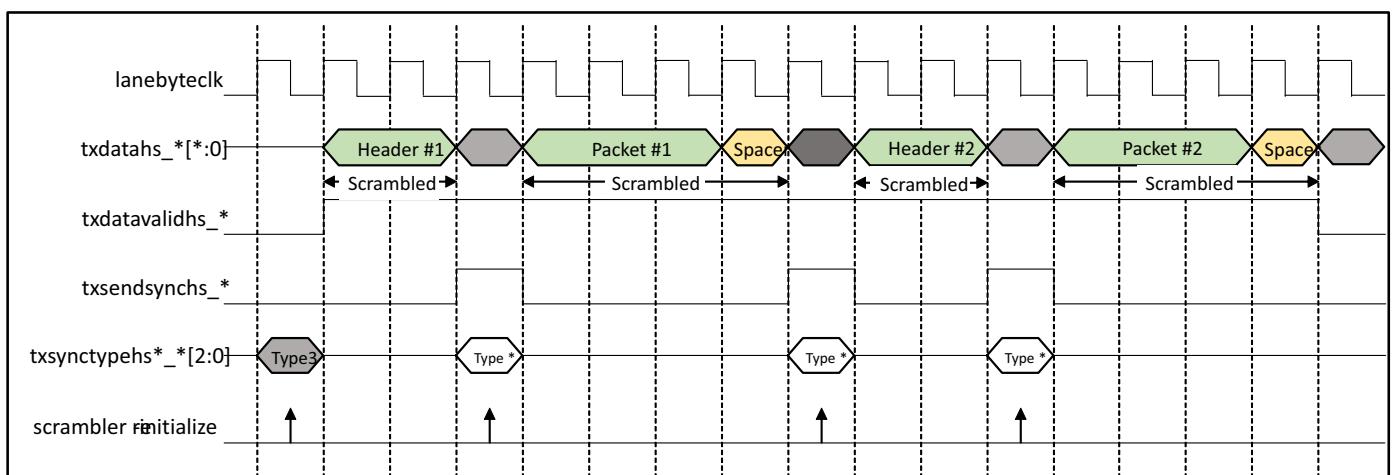


Figure 2-81 C-PHY Scrambling and Re-initialization Timing with LRTE Enabled



For C-PHY, the scrambler initialization takes place each time the Sync Word is transmitted.

2.11.2 Enabling the Scrambler

Enable this feature using the data_scrambling_en bit of the DATA_SCRAMBLING register.

For C-PHY, enable or disable single seed mode using the single_seed_en bit of the DATA_SCRAMBLING register.

2.11.3 Related Registers

Scrambler relates to only one register:

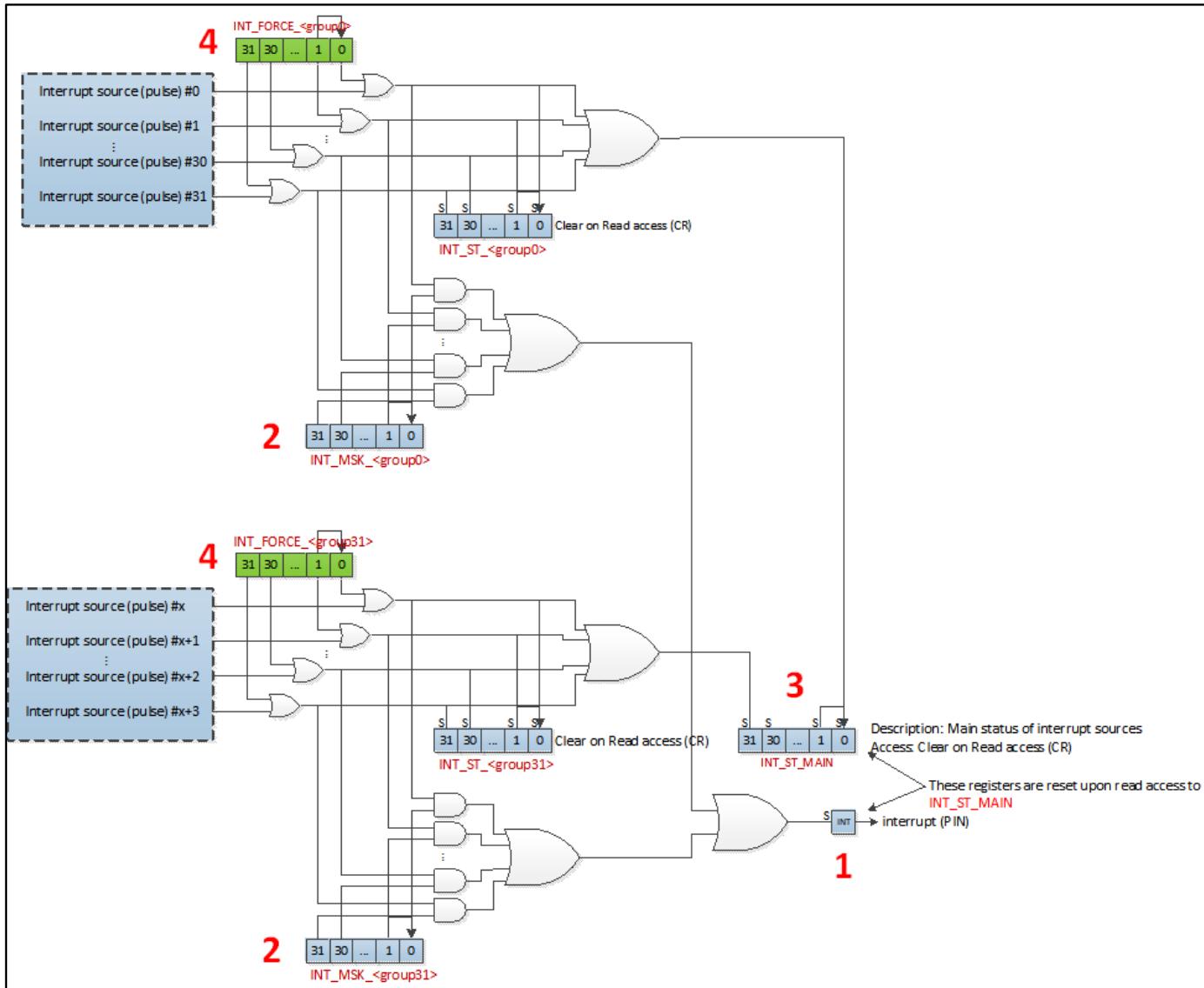
- DATA_SCRAMBLING

2.12 Interrupts

The interrupt structure in DWC_mipicsi2_device assumes that interrupt registers `INT_ST_<group>` are accessible through AMBA APB bus, and that they are running in the AMBA APB clock domain (`pclk`).

Figure 2-82 shows the interrupt structure in DWC_mipicsi2_device.

Figure 2-82 Interrupt Structure



The interrupt structure shown in Figure 2-82 is explained as follows:

1. There is a single interrupt pin registered with `pclk`, simplifying system integration. The implementation sets the interrupt pin and the applicable Interrupt Status registers at the same time.
2. The default value of the `INT_MASK_<group>` (Interrupt Mask) register is 0, and this corresponds to masking the interrupts. By default, the interrupt pin is disabled.

3. The INT_ST_MAIN register indicates which of the Interrupt Status registers have triggered the interrupt. The Interrupt Status registers contain the status of individual interrupt sources, regardless of the contents of the associated INT_MASK_<group> registers. Therefore, it is possible to service the Interrupt Status registers by polling. The Interrupt Status registers are cleared upon read access. Reading the INT_ST_MAIN register also clears the interrupt pin.
4. The INT_FORCE_<group> (Interrupt Force) registers are used for test purposes, and allows triggering interrupt events individually, without the need to activate the conditions that trigger the interrupt sources. This feature also facilitates development and testing of the software associated with the interrupt events. The INT_FORCE_<group> registers are auto-cleared. The circuitry minimizes the gate-count overhead because it avoids the need to have the Interrupt Force registers.

2.13 Error Detection

The DWC_mipicsi2_device analyzes the received packets and determines if there are protocol errors.

It is possible to monitor the following errors:

- VPG errors such as packet loss of video pattern generator
- IDI-level errors that exist if you enable the CSI2_DEVICE_IDI_IF parameter in coreConsultant
 - Frame errors such as incorrect Frame sequence, and the mismatch between Frame Start and Frame End
 - Line errors such as incorrect line sequence and mismatch between Line Start and Line End
 - Inconsistency between the IDI data configured, and the IDI data received
 - Memory errors such as header FIFO or payload FIFO overflow
- IPI-level errors that exist if you enable the CSI2_DEVICE_IPI_IF parameter in coreConsultant
 - Inconsistency between the IPI lines configured and IPI lines received
 - Inconsistency between the IPI pixels configured and the IPI pixels received
 - Inconsistency between 'ipi_embedded_wc[15:0]' and the embedded data received
 - Memory errors such as header FIFO or payload FIFO overflow and underflow
- Multiple IPI-level errors that exist if the CSI2_DEVICE_NR_IPI parameter is set to more than '1' in coreConsultant
 - Memory errors such as Multiple IPI header FIFO overflow
- Simultaneous IDI and IPI errors exist if you enable the CSI2_DEVICE_IDI_IF and CSI2_DEVICE_IPI_IF parameter in coreConsultant
 - Conflicts between IDI packets transmission and IPI packets transmission
- The SDI-level errors that exist if you enable the CSI2_DEVICE_SDI_IF parameter in coreConsultant:
 - Incorrect Frame/Line Sequence.
 - Inconsistency between the byte number of the received payload and the word count filed in packet header.
 - Inconsistency between the internal ECC/PH-CRC generation and ECC/PH-CRC of SDI bus in ECC/PH-CRC Bypass mode.
 - Inconsistency between the internal packet foot CRC generation and PF-CRC16 of SDI bus in ECC/PF-CRC16 Bypass mode.

3

Parameter Descriptions

This chapter details all the configuration parameters. You can use the coreConsultant GUI configuration reports to determine the complete configuration state of the controller. Some expressions might refer to TCL functions or procedures (sometimes identified as <functionof>) that coreConsultant uses to make calculations. The exact formula used by these TCL functions is not provided in this chapter. However, when you configure the controller in coreConsultant, all TCL functions and parameters are evaluated completely; and the resulting values are displayed where appropriate in the coreConsultant GUI reports.

The parameter descriptions in this chapter include the **Enabled:** attribute which indicates the values required to be set on other parameters before you can change the value of this parameter.

These tables define all of the configuration options for this component.

- “[Generic Configuration Parameters](#)” on page 114
- “[Data Interface FIFO Configuration Parameters](#)” on page 118
- “[Data Interface FIFO Configuration / IDI FIFO Parameters](#)” on page 119
- “[Data Interface FIFO Configuration / IPI FIFO Parameters](#)” on page 121
- “[Data Interface FIFO Configuration / IPI2 FIFO Parameters](#)” on page 123
- “[Data Interface FIFO Configuration / IPI3 FIFO Parameters](#)” on page 125
- “[Data Interface FIFO Configuration / IPI4 FIFO Parameters](#)” on page 127

3.1 Generic Configuration Parameters

Table 3-1 Generic Configuration Parameters

Label	Description
Physical Layer Configuration	
Use Synopsys D-PHY Automatic Integration	<p>This parameter enables the Synopsys D-PHY as an internal sub-module in the design.</p> <p>You need to have access to the Synopsys D-PHY package and set up the following variables:</p> <ul style="list-style-type: none"> ■ <code>setenv CSI2_DEVICE_PHY_PATH <path where D-PHY is installed></code> ■ <code>setenv CSI2_DEVICE_PLL_PATH <path where D-PHY is installed></code> ■ <code>setenv CSI2_DEVICE_PHY_LIBNAME <.lib file name></code> ■ <code>setenv CSI2_DEVICE_PLL_LIBNAME <.lib file name></code> <p>Values: 0, 1</p> <p>Default Value: 0</p> <p>Enabled: Always</p> <p>Parameter Name: <code>CSI2_DEVICE_SNPS_PHY</code></p>
Select PHY Type	<p>This parameter specifies the PHY interface used in the design.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ D-PHY (1) ■ C-PHY (2) ■ COMBO_PHY (3) <p>Default Value: D-PHY</p> <p>Enabled: <code>CSI2_DEVICE_SNPS_PHY == 0</code></p> <p>Parameter Name: <code>CSI2_DEVICE_PHY_TYPE</code></p>
Select COMBO_PHY config	<p>Values:</p> <ul style="list-style-type: none"> ■ 4L3T (1) ■ 2L2T (2) <p>Default Value: 4L3T</p> <p>Enabled: <code>CSI2_DEVICE_PHY_TYPE == 3</code></p> <p>Parameter Name: <code>CSI2_DEVICE_COMBO_PHY_CONFIG</code></p>
Select Maximum Number of D-PHY Data Lanes	<p>This parameter specifies the maximum number of data lanes supported by the D-PHY.</p> <p>Values: 1, 2, 3, 4, 5, 6, 7, 8</p> <p>Default Value: <code>(CSI2_DEVICE_COMBO_PHY_CONFIG == 2) ? 2 : 4</code></p> <p>Enabled: <code>CSI2_DEVICE_PHY_TYPE == 1</code></p> <p>Parameter Name: <code>CSI2_DEVICE_DPHY_NUM_OF_LANES</code></p>

Label	Description
Select D-PHY Data Width Mode	<p>Values:</p> <ul style="list-style-type: none"> ■ 8BIT (1) ■ 16BIT (2) ■ 32BIT (3) <p>Default Value: 8BIT</p> <p>Enabled: ((CSI2_DEVICE_DPHY_TYPE)&(CSI2_DEVICE_SNPS_PHY == 0))</p> <p>Parameter Name: CSI2_DEVICE_DPHY_WIDTH_MODE</p>
Select Maximum Number of C-PHY Lanes	<p>This parameter specifies the maximum number of data lanes supported by the C-PHY.</p> <p>Values: 1, 2, 3, 4</p> <p>Default Value: (CSI2_DEVICE_COMBO_PHY_CONFIG == 2) ? 2 : 3</p> <p>Enabled: CSI2_DEVICE_PHY_TYPE == 2</p> <p>Parameter Name: CSI2_DEVICE_CPHY_NUM_OF_LANES</p>
Select C-PHY Data Width Mode	<p>Values:</p> <ul style="list-style-type: none"> ■ 16BIT (1) ■ 32BIT (2) <p>Default Value: 16BIT</p> <p>Enabled: CSI2_DEVICE_CPHY_TYPE</p> <p>Parameter Name: CSI2_DEVICE_CPHY_WIDTH_MODE</p>
Generic Configuration	
Default Value for Number of Synchronizers	<p>This parameter selects the number of synchronization stages used for clock domain crossing.</p> <p>All stages capture the data on the rising edge of the clock.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 2 (2) ■ 3 (3) ■ 4 (4) <p>Default Value: 2</p> <p>Enabled: Always</p> <p>Parameter Name: CSI2_DEVICE_DFLT_F_SYNC_TYPE</p>
Data Scrambling	<p>Data scrambling enable, active high.</p> <p>Values: 0, 1</p> <p>Default Value: 1</p> <p>Enabled: Always</p> <p>Parameter Name: CSI2_DEVICE_DATA_SCRAMBLING</p>

Label	Description
Virtual Channel Extension	<p>Virtual Channel Extension enable:</p> <ul style="list-style-type: none"> ■ 0: Up to 4 virtual channels ■ 1: Up to 16 virtual channels for DPHY; Up to 32 virtual channels for CPHY <p>Values: 0, 1 Default Value: 1 Enabled: Always Parameter Name: CSI2_DEVICE_VC_EXTENSION</p>
Video Pattern Generation	<p>VPG enable, active high.</p> <p>Values: 0, 1 Default Value: 1 Enabled: Always Parameter Name: CSI2_DEVICE_VPG</p>
Data Interface	
Select System Interface	<p>This parameter specifies the system data interface:</p> <ul style="list-style-type: none"> ■ IDI: Image Data Interface ■ IPI: Image Pixel Interface ■ IDI&IPI: Both IDI and IPI interfaces ■ SDI: Stream Data Interface <p>Values:</p> <ul style="list-style-type: none"> ■ IDI (1) ■ IPI (2) ■ IDI&IPI (3) ■ SDI (4) <p>Default Value: IDI Enabled: Always Parameter Name: CSI2_DEVICE_DATAINTERFACE</p>
Data Width of IDI Interfaces	<p>This parameter specifies the Data Width of IDI Interfaces</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 64BIT (64) ■ 128BIT (128) <p>Default Value: 64BIT Enabled: ((CSI2_DEVICE_DATAINTERFACE == 1) ((CSI2_DEVICE_DATAINTERFACE == 3))) Parameter Name: CSI2_DEVICE_IDI_DATA_WIDTH</p>
Number of IPI Interfaces	<p>This parameter specifies the Number of IPI interfaces in the controller.</p> <p>Values: 1, 2, 3, 4 Default Value: 1 Enabled: CSI2_DEVICE_DATAINTERFACE == 2 Parameter Name: CSI2_DEVICE_NR_IPI</p>

Label	Description
Data Width of IPI Interfaces	<p>This parameter specifies the Data Width of IPI interfaces.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 48BIT (1) ■ 192BIT (4) <p>Default Value: 48BIT</p> <p>Enabled: ((CSI2_DEVICE_DATAINTERFACE == 2) ((CSI2_DEVICE_DATAINTERFACE == 3))</p> <p>Parameter Name: CSI2_DEVICE_IPI_LANE_NUM</p>
Automotive Package Configuration	
Enable Automotive Package	<p>Enable Automotive Package To generate a DWC_mipicsi2_device controller with support for automotive package, you must have an associated DWC_ap_mipicsi2_device license.</p> <p>Values: 0, 1</p> <p>Default Value: 0</p> <p>Enabled: Always</p> <p>Parameter Name: CSI2_DEVICE_AP</p>
Enable Safety Slave	<p>Enable Safety Slave</p> <p>Values: 0, 1</p> <p>Default Value: 0</p> <p>Enabled: CSI2_DEVICE_AP</p> <p>Parameter Name: CSI2_DEVICE_SS</p>

3.2 Data Interface FIFO Configuration Parameters

Table 3-2 Data Interface FIFO Configuration Parameters

Label	Description
RAM TYPE Configuration	
Select External RAM Type	<p>This parameter specifies External RAM Type:</p> <ul style="list-style-type: none">■ DPRAM: Dual-port synchronous RAMs■ SPRAM: Single-port synchronous RAMs <p>Values:</p> <ul style="list-style-type: none">■ DPRAM (1)■ SPRAM (2) <p>Default Value: DPRAM</p> <p>Enabled: ((CSI2_DEVICE_DATAINTERFACE == 1) ((CSI2_DEVICE_DATAINTERFACE == 2) ((CSI2_DEVICE_DATAINTERFACE == 3)))</p> <p>Parameter Name: CSI2_DEVICE_EXT_RAM_TYPE</p>

3.3 Data Interface FIFO Configuration / IDI FIFO Parameters

Table 3-3 Data Interface FIFO Configuration / IDI FIFO Parameters

Label	Description
IDI Header FIFO Configuration	
IDI Header FIFO Depth	<p>This parameter specifies the depth of the IDI header FIFO. This FIFO is implemented with logic gates (flip-flops).</p> <p>Values: 4, ..., 256</p> <p>Default Value: 8</p> <p>Enabled: CSI2_DEVICE_IDI_IF == 1</p> <p>Parameter Name: CSI2_DEVICE_IDI_HD_FIFO_DEPTH</p>
IDI Payload FIFO Width	
External Payload RAM Width for IDI	<p>The width of the external payload RAM for IDI</p> <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_AP CSI2_DEVICE_NAP_IDI_PL_FIFO_WIDTH CSI2_DEVICE_IDI_PL_ECC_WIDTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IDI_PLD_WIDTH</p>
IDI Payload FIFO Configuration with DPRAM	
IDI Payload FIFO Depth with DPRAM	<p>This parameter specifies the depth of the IDI payload FIFO. This FIFO requires an external asynchronous DPRAM.</p> <p>Values: 128, ..., 16384</p> <p>Default Value: 1024</p> <p>Enabled: ((CSI2_DEVICE_USE_DPRAM == 1) && (CSI2_DEVICE_IDI_IF == 1))</p> <p>Parameter Name: CSI2_DEVICE_IDI_PLD_FIFO_DEPTH</p>
Required IDI Payload DPRAM Depth	<p>This parameter specifies the depth of the RAM required for the IDI payload FIFO.</p> <ul style="list-style-type: none"> ■ If value of CSI2_DEVICE_IDI_PLD_FIFO_DEPTH parameter is odd, the value of CSI2_DEVICE_IDI_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IDI_PLD_FIFO_DEPTH + 1). ■ If value of CSI2_DEVICE_IDI_PLD_FIFO_DEPTH parameter is even but not an integer power of two, the value of CSI2_DEVICE_IDI_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IDI_PLD_FIFO_DEPTH + 2). <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_IDI_PLD_FIFO_DEPTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IDI_PLD_RAM_DEPTH</p>

Label	Description
IDI Payload FIFO Configuration with SPRAM	
IDI Payload FIFO Depth with SPRAM	<p>This parameter specifies the total depth of the IDI payload FIFO. This FIFO requires two external SPRAM. Provided value must be a multiple of 2.</p> <p>The total depth includes:</p> <ul style="list-style-type: none"> ■ External RAM depth. ■ Internal CDC FIFO depth which is based on the level of almost full and cdc sync stage. <p>Values: 128, ..., 16384 Default Value: 1024 Enabled: ((CSI2_DEVICE_USE_SPRAM == 1) && (CSI2_DEVICE_IDI_IF == 1)) Parameter Name: CSI2_DEVICE_IDI_PLD_FIFO_TOTAL_DEPTH</p>
Required IDI Payload PTP SPRAM Depth	<p>This parameter specifies the depth of RAM to the underlying PTP RAM controller of the IDI payload FIFO. This depth should be equally divided by the external odd and even ram.</p> <p>Values: -2147483648, ..., 2147483647 Default Value: [<functionof> CSI2_DEVICE_IDI_PLD_FIFO_PTP_DEPTH] Enabled: 0 Parameter Name: CSI2_DEVICE_IDI_PLD_PTP_SPRAM_DEPTH</p>

3.4 Data Interface FIFO Configuration / IPI FIFO Parameters

Table 3-4 Data Interface FIFO Configuration / IPI FIFO Parameters

Label	Description
IPI Header FIFO Configuration	
IPI Header FIFO Depth	<p>The parameter specifies the depth of the IPI header FIFO. This FIFO is implemented with logic gates (flip-flops).</p> <p>Values: 8, ..., 256</p> <p>Default Value: 8</p> <p>Enabled: CSI2_DEVICE_IPI_IF == 1</p> <p>Parameter Name: CSI2_DEVICE_IPI_HD_FIFO_DEPTH</p>
IPI Payload FIFO Width	
External Payload RAM Width for IPI	<p>The width of the external payload RAM for IPI</p> <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_AP CSI2_DEVICE_NAP_IPI_PL_FIFO_WIDTH CSI2_DEVICE_IPI_PL_ECC_WIDTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IPI_PLD_WIDTH</p>
IPI Payload FIFO Configuration with DPRAM	
IPI Payload FIFO Depth with DPRAM	<p>This parameter is the depth of the IPI payload FIFO. This FIFO requires an external asynchronous DPRAM.</p> <p>Values: 128, ..., 16384</p> <p>Default Value: 1024</p> <p>Enabled: ((CSI2_DEVICE_USE_DPRAM == 1) && (CSI2_DEVICE_IPI_IF == 1))</p> <p>Parameter Name: CSI2_DEVICE_IPI_PLD_FIFO_DEPTH</p>
Required IPI Payload DPRAM depth	<p>This parameter is the depth of the RAM required for IPI payload FIFO.</p> <ul style="list-style-type: none"> ■ If value of CSI2_DEVICE_IPI_PLD_FIFO_DEPTH parameter is odd, the value of CSI2_DEVICE_IPI_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IPI_PLD_FIFO_DEPTH + 1). ■ If value of CSI2_DEVICE_IPI_PLD_FIFO_DEPTH parameter is even but not an integer power of two, the value of CSI2_DEVICE_IPI_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IPI_PLD_FIFO_DEPTH + 2). <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_IPI_PLD_FIFO_DEPTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IPI_PLD_RAM_DEPTH</p>

Label	Description
IPI Payload FIFO Configuration with SPRAM	
IPI Payload FIFO Depth with SPRAM	<p>This parameter specifies the total depth of the IPI payload FIFO. This FIFO requires two external SPRAM. Provided value must be a multiple of 2.</p> <p>The total depth includes:</p> <ul style="list-style-type: none"> ■ External RAM depth. ■ Internal CDC FIFO depth which is based on the level of almost full and cdc sync stage. <p>Values: 128, ..., 16384 Default Value: 1024 Enabled: ((CSI2_DEVICE_USE_SPRAM == 1) && (CSI2_DEVICE_IPI_IF == 1)) Parameter Name: CSI2_DEVICE_IPI_PLD_FIFO_TOTAL_DEPTH</p>
Required IPI Payload PTP SPRAM Depth	<p>This parameter specifies the depth of RAM to the underlying PTP RAM controller of the IPI payload FIFO. This depth should be equally divided by the external odd and even ram.</p> <p>Values: -2147483648, ..., 2147483647 Default Value: [<functionof> CSI2_DEVICE_IPI_PLD_FIFO_PTP_DEPTH] Enabled: 0 Parameter Name: CSI2_DEVICE_IPI_PLD_PTP_SPRAM_DEPTH</p>

3.5 Data Interface FIFO Configuration / IPI2 FIFO Parameters

Table 3-5 Data Interface FIFO Configuration / IPI2 FIFO Parameters

Label	Description
IPI2 Header FIFO Configuration	
IPI2 Header FIFO Depth	<p>This parameter specifies the depth of the IPI2 header FIFO. This FIFO is implemented with logic gates (flip-flops).</p> <p>Values: 8, ..., 256</p> <p>Default Value: 8</p> <p>Enabled: CSI2_DEVICE_IPI2_IF == 1</p> <p>Parameter Name: CSI2_DEVICE_IPI2_HD_FIFO_DEPTH</p>
IPI Payload FIFO Width	
External Payload RAM Width for IPI	<p>The width of the external payload RAM for IPI</p> <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_AP CSI2_DEVICE_NAP_IPI_PL_FIFO_WIDTH CSI2_DEVICE_IPI_PL_ECC_WIDTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IPI_PLD_WIDTH</p>
IPI2 Payload FIFO Configuration with DPRAM	
IPI2 Payload FIFO Depth with DPRAM	<p>This parameter is the depth of the IPI2 payload FIFO. This FIFO requires an external asynchronous DPRAM.</p> <p>Values: 128, ..., 16384</p> <p>Default Value: 1024</p> <p>Enabled: ((CSI2_DEVICE_USE_DPRAM == 1) && (CSI2_DEVICE_IPI2_IF == 1))</p> <p>Parameter Name: CSI2_DEVICE_IPI2_PLD_FIFO_DEPTH</p>
Required IPI2 Payload DPRAM Depth	<p>This parameter is the depth of the RAM required for IPI2 payload FIFO.</p> <ul style="list-style-type: none"> ■ If value of CSI2_DEVICE_IPI2_PLD_FIFO_DEPTH parameter is odd, the value of CSI2_DEVICE_IPI2_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IPI2_PLD_FIFO_DEPTH + 1). ■ If value of CSI2_DEVICE_IPI2_PLD_FIFO_DEPTH parameter is even but not an integer power of two, the value of CSI2_DEVICE_IPI2_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IPI2_PLD_FIFO_DEPTH + 2). <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_IPI2_PLD_FIFO_DEPTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IPI2_PLD_RAM_DEPTH</p>

Label	Description
IPI2 Payload FIFO Configuration with SPRAM	
IPI2 Payload FIFO Depth with SPRAM	<p>This parameter specifies the total depth of the IPI2 payload FIFO. This FIFO requires two external SPRAM. Provided value must be a multiple of 2.</p> <p>The total depth includes:</p> <ul style="list-style-type: none"> ■ External RAM depth. ■ Internal CDC FIFO depth which is based on the level of almost full and cdc sync stage. <p>Values: 128, ..., 16384 Default Value: 1024 Enabled: ((CSI2_DEVICE_USE_SPRAM == 1) && (CSI2_DEVICE_IPI2_IF == 1)) Parameter Name: CSI2_DEVICE_IPI2_PLD_FIFO_TOTAL_DEPTH</p>
Required IPI2 Payload PTP SPRAM Depth	<p>This parameter specifies the depth of RAM to the underlying PTP RAM controller of the IPI2 payload FIFO. This depth should be equally divided by the external odd and even ram.</p> <p>Values: -2147483648, ..., 2147483647 Default Value: [<functionof> CSI2_DEVICE_IPI2_PLD_FIFO_PTP_DEPTH] Enabled: 0 Parameter Name: CSI2_DEVICE_IPI2_PLD_PTP_SPRAM_DEPTH</p>

3.6 Data Interface FIFO Configuration / IPI3 FIFO Parameters

Table 3-6 Data Interface FIFO Configuration / IPI3 FIFO Parameters

Label	Description
IPI3 Header FIFO Configuration	
IPI3 Header FIFO Depth	<p>This parameter is the depth of the IPI3 header FIFO. This FIFO is implemented with logic gates (flip-flops).</p> <p>Values: 8, ..., 256</p> <p>Default Value: 8</p> <p>Enabled: CSI2_DEVICE_IPI3_IF == 1</p> <p>Parameter Name: CSI2_DEVICE_IPI3_HD_FIFO_DEPTH</p>
IPI Payload FIFO Width	
External Payload RAM Width for IPI	<p>The width of the external payload RAM for IPI</p> <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_AP CSI2_DEVICE_NAP_IPI_PL_FIFO_WIDTH CSI2_DEVICE_IPI_PL_ECC_WIDTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IPI_PLD_WIDTH</p>
IPI3 Payload FIFO Configuration with DPRAM	
IPI3 Payload FIFO Depth with DPRAM	<p>This parameter specifies the depth of the IPI3 payload FIFO. This FIFO requires an external asynchronous DPRAM.</p> <p>Values: 128, ..., 16384</p> <p>Default Value: 1024</p> <p>Enabled: ((CSI2_DEVICE_USE_DPRAM == 1) && (CSI2_DEVICE_IPI3_IF == 1))</p> <p>Parameter Name: CSI2_DEVICE_IPI3_PLD_FIFO_DEPTH</p>
Required IPI3 Payload DPRAM Depth	<p>Depth of the RAM required for IPI3 payload FIFO. If value of CSI2_DEVICE_IPI3_PLD_FIFO_DEPTH parameter is odd, the value of CSI2_DEVICE_IPI3_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IPI3_PLD_FIFO_DEPTH + 1). If value of CSI2_DEVICE_IPI3_PLD_FIFO_DEPTH parameter is even value but not an integer power of two, the value of CSI2_DEVICE_IPI3_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IPI3_PLD_FIFO_DEPTH + 2).</p> <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_IPI3_PLD_FIFO_DEPTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IPI3_PLD_RAM_DEPTH</p>

Label	Description
IPI3 Payload FIFO Configuration with SPRAM	
IPI3 Payload FIFO Depth with SPRAM	<p>This parameter specifies the total depth of the IPI3 payload FIFO. This FIFO requires two external SPRAM. Provided value must be a multiple of 2.</p> <p>The total depth includes:</p> <ul style="list-style-type: none"> ■ External RAM depth. ■ Internal CDC FIFO depth which is based on the level of almost full and cdc sync stage. <p>Values: 128, ..., 16384 Default Value: 1024 Enabled: ((CSI2_DEVICE_USE_SPRAM == 1) && (CSI2_DEVICE_IPI3_IF == 1)) Parameter Name: CSI2_DEVICE_IPI3_PLD_FIFO_TOTAL_DEPTH</p>
Required IPI3 Payload PTP SPRAM Depth	<p>This parameter specifies the depth of RAM to the underlying PTP RAM controller of the IPI3 payload FIFO. This depth should be equally divided by the external odd and even ram.</p> <p>Values: -2147483648, ..., 2147483647 Default Value: [<functionof> CSI2_DEVICE_IPI3_PLD_FIFO_PTP_DEPTH] Enabled: 0 Parameter Name: CSI2_DEVICE_IPI3_PLD_PTP_SPRAM_DEPTH</p>

3.7 Data Interface FIFO Configuration / IPI4 FIFO Parameters

Table 3-7 Data Interface FIFO Configuration / IPI4 FIFO Parameters

Label	Description
IPI4 Header FIFO Configuration	
IPI4 Header FIFO Depth	<p>This parameter is the depth of the IPI4 header FIFO. This FIFO is implemented with logic gates (flip-flops).</p> <p>Values: 8, ..., 256</p> <p>Default Value: 8</p> <p>Enabled: CSI2_DEVICE_IPI4_IF == 1</p> <p>Parameter Name: CSI2_DEVICE_IPI4_HD_FIFO_DEPTH</p>
IPI Payload FIFO Width	
External Payload RAM Width for IPI	<p>The width of the external payload RAM for IPI</p> <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_AP CSI2_DEVICE_NAP_IPI_PL_FIFO_WIDTH CSI2_DEVICE_IPI_PL_ECC_WIDTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IPI_PLD_WIDTH</p>
IPI4 Payload FIFO Configuration with DPRAM	
IPI4 Payload FIFO Depth with DPRAM	<p>This parameter specifies the depth of the IPI4 payload FIFO. This FIFO requires an external asynchronous DPRAM.</p> <p>Values: 128, ..., 16384</p> <p>Default Value: 1024</p> <p>Enabled: ((CSI2_DEVICE_USE_DPRAM == 1) && (CSI2_DEVICE_IPI4_IF == 1))</p> <p>Parameter Name: CSI2_DEVICE_IPI4_PLD_FIFO_DEPTH</p>
Required IPI4 Payload DPRAM Depth	<p>This parameter is the depth of the RAM required for IPI4 payload FIFO.</p> <ul style="list-style-type: none"> ■ If value of CSI2_DEVICE_IPI4_PLD_FIFO_DEPTH parameter is odd, the value of CSI2_DEVICE_IPI4_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IPI4_PLD_FIFO_DEPTH + 1). ■ If value of CSI2_DEVICE_IPI4_PLD_FIFO_DEPTH parameter is even but not an integer power of two, the value of CSI2_DEVICE_IPI4_PLD_RAM_DEPTH is required to be (CSI2_DEVICE_IPI4_PLD_FIFO_DEPTH + 2). <p>Values: -2147483648, ..., 2147483647</p> <p>Default Value: [<functionof> CSI2_DEVICE_IPI4_PLD_FIFO_DEPTH]</p> <p>Enabled: 0</p> <p>Parameter Name: CSI2_DEVICE_IPI4_PLD_RAM_DEPTH</p>

Label	Description
IPI4 Payload FIFO Configuration with SPRAM	
IPI4 Payload FIFO Depth with SPRAM	<p>This parameter specifies the total depth of the IPI4 payload FIFO. This FIFO requires two external SPRAM. Provided value must be a multiple of 2.</p> <p>The total depth includes:</p> <ul style="list-style-type: none"> ■ External RAM depth. ■ Internal CDC FIFO depth which is based on the level of almost full and cdc sync stage. <p>Values: 128, ..., 16384 Default Value: 1024 Enabled: ((CSI2_DEVICE_USE_SPRAM == 1) && (CSI2_DEVICE_IPI4_IF == 1)) Parameter Name: CSI2_DEVICE_IPI4_PLD_FIFO_TOTAL_DEPTH</p>
Required IPI4 Payload PTP SPRAM Depth	<p>This parameter specifies the depth of RAM to the underlying PTP RAM controller of the IPI4 payload FIFO. This depth should be equally divided by the external odd and even ram.</p> <p>Values: -2147483648, ..., 2147483647 Default Value: [<functionof> CSI2_DEVICE_IPI4_PLD_FIFO_PTP_DEPTH] Enabled: 0 Parameter Name: CSI2_DEVICE_IPI4_PLD_PTP_SPRAM_DEPTH</p>

4

Signal Descriptions

This chapter details all possible I/O signals in the IP. For configurable IP titles, your actual configuration might not contain all of these signals.

Inputs are on the left of the signal diagrams; outputs are on the right.

Attention: For configurable IP titles, do not use this document to determine the exact I/O footprint of the controller. It is for reference purposes only.

When you configure the controller in coreConsultant, you must access the I/O signals for your actual configuration at workspace/report/IO.html or workspace/report/IO.xml after you have completed the report creation activity. That report comes from the exact same source as this chapter but removes all the I/O signals that are not in your actual configuration. This does not apply to non-configurable IP titles. In addition, all parameter expressions are evaluated to actual values. Therefore, the widths might change depending on your actual configuration.

Some expressions might refer to TCL functions or procedures (sometimes identified as <functionof>) that coreConsultant uses to make calculations. The exact formula used by these TCL functions is not provided in this chapter. However, when you configure the controller in coreConsultant, all TCL functions and parameters are evaluated completely; and the resulting values are displayed where appropriate in the coreConsultant GUI reports.

In addition to describing the function of each signal, the signal descriptions in this chapter include the following information:

- **Active State:** Indicates whether the signal is active high or active low. When a signal is not intended to be used in a particular application, then this signal needs to be tied or driven to the inactive state (opposite of the active state).
- **Registered:** Indicates whether or not the signal is registered directly inside the IP boundary without intervening logic (excluding simple buffers). A value of No does not imply that the signal is not synchronous, only that there is some combinatorial logic between the signal's origin or destination register and the boundary of the controller. A value of N/A indicates that this information is not provided for this IP title.
- **Synchronous to:** Indicates which clocks in the IP sample this input (drive for an output). This clock might not be the same as the clock that your application logic should use to clock (sample/drive) this pin. For more details, consult the clock section in the databook.
- **Exists:** Name of configuration parameter that populates this signal in your configuration.

The I/O signals are grouped as follows:

- “AMBA Slave Interface Signals” on page 131
- “Interrupt Signals” on page 134
- “FMEDA Interrupt Signals” on page 135
- “SSM Bus Interface Signals” on page 136
- “Diagnosis fault Signals” on page 139
- “IDI Interface Signals” on page 143
- “IDI Payload RAM Signals” on page 148
- “SDI Interface Signals” on page 153
- “IPI Interface Signals” on page 157
- “IPI Payload RAM Signals” on page 167
- “PHY Signals” on page 177
- “PHY Protocol Interface (PPI) Signals” on page 180
- “PHY Protocol Interface (PPI) Signals for D-PHY Clock Lane” on page 181
- “PHY Protocol Interface (PPI) Signals for D-PHY” on page 183
- “PHY Protocol Interface (PPI) Signals for Data Lane N” on page 184
- “Parallel Port for PHY Configuration Signals” on page 189
- “PHY Protocol Interface (PPI) Signals for Calibration Signals” on page 191
- “Scan Chain Signals” on page 192

4.1 AMBA Slave Interface Signals

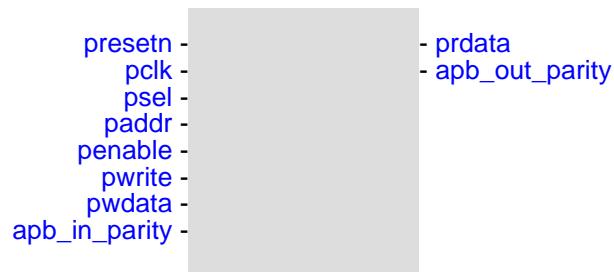


Table 4-1 AMBA Slave Interface Signals

Port Name	I/O	Description
presetn	I	<p>Peripheral reset This is the global reset of the controller, including all the registers.</p> <p>Exists: Always Synchronous To: Asynchronous Registered: N/A Power Domain: SINGLE_DOMAIN Active State: Low</p>
pclk	I	<p>APB bus clock The rising edge of pclk is used to time all transfers on the APB bus. The minimum required frequency is 15 MHz.</p> <p>Exists: Always Synchronous To: None Registered: N/A Power Domain: SINGLE_DOMAIN Active State: N/A</p>
psel	I	<p>APB select</p> <p>Exists: Always Synchronous To: pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
paddr[11:0]	I	<p>APB address bus</p> <p>Exists: Always Synchronous To: pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

Port Name	I/O	Description
penable	I	<p>APB strobe</p> <p>Exists: Always</p> <p>Synchronous To: pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
pwrite	I	<p>APB transfer direction</p> <p>When High, this signal indicates an APB write access; when Low, a read access.</p> <p>Exists: Always</p> <p>Synchronous To: pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
pwdata[31:0]	I	<p>APB write data bus</p> <p>Exists: Always</p> <p>Synchronous To: pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
prdata[31:0]	O	<p>APB read data bus</p> <p>Exists: Always</p> <p>Synchronous To: pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
apb_in_parity[5:0]	I	<p>Parity Data of Even Mode</p> <p>This signal is an input along with the APB interface, which indicates the correctness for the parity of APB input interface signals.</p> <p>The following bits correspond to the parity of the specific APB bus signal:</p> <ul style="list-style-type: none"> ■ Bit 0: pwdata[7:0] ■ Bit 1: pwdata[15:8] ■ Bit 2: pwdata[23:16] ■ Bit 3: pwdata[31:24] ■ Bit 4: paddr[7:0] ■ Bit 5: {paddr[11:8], psel, penable and pwrite} <p>Exists: CSI2_DEVICE_AP</p> <p>Synchronous To: pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
apb_out_parity[3:0]	O	<p>Parity Data of Even Mode This signal is an output along with the APB interface, which indicates the correctness for the parity prdata as follows:</p> <ul style="list-style-type: none"> ■ Bit 0: prdata[7:0] ■ Bit 1: prdata[15:8] ■ Bit 2: prdata[23:16] ■ Bit 3: prdata[31:24] <p>Exists: CSI2_DEVICE_AP Synchronous To: pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

4.2 Interrupt Signals

- interrupt
- anti_interrupt

Table 4-2 Interrupt Signals

Port Name	I/O	Description
interrupt	O	<p>Interrupt function</p> <p>Exists: Always</p> <p>Synchronous To: pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
anti_interrupt	O	<p>Antivalent Interrupt</p> <p>It is the antivalent interrupt output, which in AP mode is duplicated to implement the antivalent feature. {anti_interrupt,interrupt}</p> <ul style="list-style-type: none"> ■ 00: Invalid ■ 01: Active state ■ 10: Non-active state ■ 11: Invalid <p>Exists: CSI2_DEVICE_AP</p> <p>Synchronous To: pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

4.3 FMEDA Interrupt Signals

- diag_interrupt

Table 4-3 FMEDA Interrupt Signals

Port Name	I/O	Description
diag_interrupt[1:0]	O	<p>FMEDA Interrupt function It is the antivalent output, which is duplicated to implement the antivalent feature.</p> <ul style="list-style-type: none"> ■ 00: Invalid ■ 01: Active state ■ 10: Non-active state ■ 11: Invalid <p>Exists: CSI2_DEVICE_AP Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High</p>

4.4 SSM Bus Interface Signals

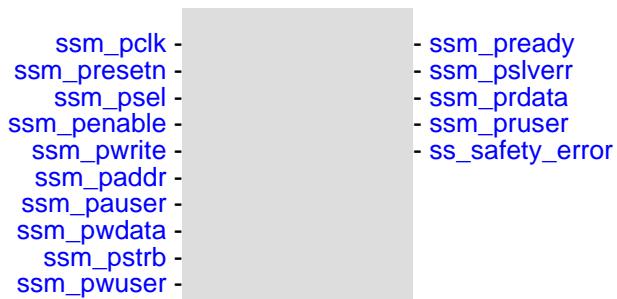


Table 4-4 SSM Bus Interface Signals

Port Name	I/O	Description
ssm_pclk	I	<p>Safety Slave clock The clock may be enabled only when accessing the safety slave. Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: None Registered: N/A Power Domain: SINGLE_DOMAIN Active State: N/A</p>
ssm_presetn	I	<p>Safety Slave reset Active low. Must be asserted along with core_resetn and de-asserted synchronously with ssm_pclk after core_resetn goes inactive. Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: Asynchronous Registered: N/A Power Domain: SINGLE_DOMAIN Active State: Low</p>
ssm_psel	I	<p>Safety Slave select Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: ssm_pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
ssm_penable	I	<p>Safety Slave Enable Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: ssm_pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>

Port Name	I/O	Description
ssm_pwrite	I	<p>Safety Slave Write/Read indicator When High, this signal indicates a write access; when Low, a read access.</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: ssm_pcclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
ssm_paddr[7:0]	I	<p>Safety Slave address</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: ssm_pcclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
ssm_pauser[4:0]	I	<p>Safety Slave Parity Data This consists of the parity bit for the address and Antivalent bits of the control signals pwrite, psel and penable. {ssm_paddr_parit, 1'b0, ssm_pwrite_av, ssm_penable_av, ssm_psel_av}.</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: ssm_pcclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
ssm_pwdata[31:0]	I	<p>Safety Slave Write Data</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: ssm_pcclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
ssm_pstrb[3:0]	I	<p>Safety Slave Write Data Strobe</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS) Synchronous To: ssm_pcclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>

Port Name	I/O	Description
ssm_pwuser[4:0]	I	<p>Safety Slave Write Parity Data of Even Mode {ssm_pwdata_parity, ssm_pstrb_parity}</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS)</p> <p>Synchronous To: ssm_pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ssm_pready	O	<p>Safety Slave Ready</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS)</p> <p>Synchronous To: ssm_pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ssm_ps verr	O	<p>Safety Slave Error</p> <p>Parity error detected if enabled on ssm_paddr or ssm_pwdata or ssm_pstrb.</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS)</p> <p>Synchronous To: ssm_pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ssm_prdata[31:0]	O	<p>Safety Slave Read data</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS)</p> <p>Synchronous To: ssm_pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ssm_pruser[5:0]	O	<p>Safety Slave Read Parity Data of Even Mode {ssm_prdata_parity, ssm_ps verr_av, ssm_pready_av}</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS)</p> <p>Synchronous To: ssm_pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ss_safety_error[1:0]	O	<p>Safety Error. Sticky until reset. No Error = 2'b10 Error = 2'b1</p> <p>Exists: (CSI2_DEVICE_AP) && (CSI2_DEVICE_SS)</p> <p>Synchronous To: pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

4.5 Diagnosis fault Signals

- diag_faults

Table 4-5 Diagnosis fault Signals

Port Name	I/O	Description
diag_faults[15:0]	O	<p>Diagnosis Fault Bus This signal is an output along within the automotive package, which indicates the correctness and provides a fast access of the faulty behavior of the controller.</p> <ul style="list-style-type: none"> ■ Bit[1:0]: internal_fault - Registers Parity, Redundancy and Single pulse <ul style="list-style-type: none"> □ AMBAAPBINTF_AP_STATUS.reg_parity_error □ PHY_IF_CTRL_AP_STATUS.reg_parity_error □ PHY_IF_CTRL_AP_STATUS.lane_mngr_reg_parity_error □ PKT_IF_AP_STATUS.idi_ipi_control_reg_parity_error □ PKT_IF_AP_STATUS.reg_parity_error □ REG_BANK_AP_STATUS.reg_parity_error □ MT_IPI_CONTROL_AP_STATUS.reg_parity_error □ MT_IPI_CONTROL_AP_STATUS.mt_ipi_field_control_reg_parity_error □ CMU_AP_STATUS.reg_parity_error □ ECF_AP_STATUS.reg_parity_error □ PKT_BUILDER_AP_STATUS.reg_parity_error □ IDI_AP_STATUS.reg_parity_error □ IDI_FIFOCTRL_AP_STATUS.reg_parity_error □ SDI_AP_STATUS.reg_parity_error □ IPI_AP_STATUS.reg_parity_error □ IPI_FIFOCTRL_AP_STATUS.reg_parity_error □ IPI2_AP_STATUS.reg_parity_error □ IPI2_FIFOCTRL_AP_STATUS.reg_parity_error □ IPI3_AP_STATUS.reg_parity_error □ IPI3_FIFOCTRL_AP_STATUS.reg_parity_error □ IPI4_AP_STATUS.reg_parity_error □ IPI4_FIFOCTRL_AP_STATUS.reg_parity_error □ ERR_HANDLER_AP_STATUS.duplicate_error □ PKT_IF_AP_STATUS.channel_sel_duplicate_error □ PKT_BUILDER_AP_STATUS.elastbuf_duplicate_error □ IDI_FIFOCTRL_AP_STATUS.elastbuf_duplicate_error □ IPI_FIFOCTRL_AP_STATUS.elastbuf_duplicate_error □ IPI2_FIFOCTRL_AP_STATUS.elastbuf_duplicate_error □ MT_IPI_CONTROL_AP_STATUS.channel_sel_duplicate_error □ IPI3_FIFOCTRL_AP_STATUS.elastbuf_duplicate_error

Port Name	I/O	Description
diag_faults[15:0]...(cont.)	O.	<ul style="list-style-type: none"> <input type="checkbox"/> IPI4_FIFOCTRL_AP_STATUS.elastbuf_duplicate_error <input type="checkbox"/> PHY_IF_CTRL_AP_STATUS.elastbuf_duplicate_error <input type="checkbox"/> PHY_IF_CTRL_AP_STATUS.scrambler_duplicate_error <input type="checkbox"/> PHY_IF_CTRL_AP_STATUS.sig_pulse_error <input type="checkbox"/> REG_BANK_AP_STATUS.sig_pulse_error <input type="checkbox"/> AMBAAPBINTF_AP_STATUS.sig_pulse_error ■ Bit[3:2]: ECC error found and corrected fault <ul style="list-style-type: none"> <input type="checkbox"/> IDI_FIFOCTRL_AP_STATUS.hddp_eccerr <input type="checkbox"/> IDI_FIFOCTRL_AP_STATUS.pldp_eccerr <input type="checkbox"/> IPI_FIFOCTRL_AP_STATUS.hddp_eccerr <input type="checkbox"/> IPI_FIFOCTRL_AP_STATUS.pldp_eccerr <input type="checkbox"/> IPI2_FIFOCTRL_AP_STATUS.hddp_eccerr <input type="checkbox"/> IPI2_FIFOCTRL_AP_STATUS.pldp_eccerr <input type="checkbox"/> MT_IPI_CONTROL_AP_STATUS.hddp_eccerr <input type="checkbox"/> IPI3_FIFOCTRL_AP_STATUS.hddp_eccerr <input type="checkbox"/> IPI3_FIFOCTRL_AP_STATUS.pldp_eccerr <input type="checkbox"/> IPI4_FIFOCTRL_AP_STATUS.hddp_eccerr <input type="checkbox"/> IPI4_FIFOCTRL_AP_STATUS.pldp_eccerr ■ Bit[5:4]: ECC error found but not corrected fault <ul style="list-style-type: none"> <input type="checkbox"/> IDI_FIFOCTRL_AP_STATUS.hddp_eccmulerr <input type="checkbox"/> IDI_FIFOCTRL_AP_STATUS.pldp_eccmulerr <input type="checkbox"/> IPI_FIFOCTRL_AP_STATUS.hddp_eccmulerr <input type="checkbox"/> IPI_FIFOCTRL_AP_STATUS.pldp_eccmulerr <input type="checkbox"/> IPI2_FIFOCTRL_AP_STATUS.hddp_eccmulerr <input type="checkbox"/> IPI2_FIFOCTRL_AP_STATUS.pldp_eccmulerr <input type="checkbox"/> MT_IPI_CONTROL_AP_STATUS.hddp_eccmulerr <input type="checkbox"/> IPI3_FIFOCTRL_AP_STATUS.hddp_eccmulerr <input type="checkbox"/> IPI3_FIFOCTRL_AP_STATUS.pldp_eccmulerr <input type="checkbox"/> IPI4_FIFOCTRL_AP_STATUS.hddp_eccmulerr <input type="checkbox"/> IPI4_FIFOCTRL_AP_STATUS.pldp_eccmulerr ■ Bit[7:6]: CRC error found on data incoming from memory fault <ul style="list-style-type: none"> <input type="checkbox"/> PKT_BUILDER_AP_STATUS.pldp_crcerr ■ Bit[9:8]: input parity check errors fault <ul style="list-style-type: none"> <input type="checkbox"/> IDI_AP_STATUS.input_parity_error

Port Name	I/O	Description
diag_faults[15:0]...(cont..)	O..	<ul style="list-style-type: none"> <input type="checkbox"/> SDI_AP_STATUS.input_parity_error <input type="checkbox"/> IPI_AP_STATUS.input_parity_error <input type="checkbox"/> IPI2_AP_STATUS.input_parity_error <input type="checkbox"/> IPI3_AP_STATUS.input_parity_error <input type="checkbox"/> IPI4_AP_STATUS.input_parity_error <input type="checkbox"/> AMBAAPBINTF_AP_STATUS.input_parity_error ■ Bit[11:10]: synchronizer check errors fault <ul style="list-style-type: none"> <input type="checkbox"/> INT_ST_DIAG0.int_st_diag_sync ■ Bit[13:12]: PHY errors fault <ul style="list-style-type: none"> <input type="checkbox"/> INT_ST_FAP_PHY.to_hs_tx <input type="checkbox"/> INT_ST_FAP_PHY.errcontentionlp0 <input type="checkbox"/> INT_ST_FAP_PHY.errcontentionlp1 ■ Bit[15:14]: protocol errors fault <ul style="list-style-type: none"> <input type="checkbox"/> INT_ST_FAP_IDI.idi_errwc <input type="checkbox"/> INT_ST_FAP_IDI.idi_vc*_err_seq <input type="checkbox"/> INT_ST_FAP_IDI.idi_fifo_overflow <input type="checkbox"/> INT_ST_FAP_IPI.ipi_errpixel <input type="checkbox"/> INT_ST_FAP_SDI.sdi_errwc <input type="checkbox"/> INT_ST_FAP_SDI.sdi_vc*_err_seq <input type="checkbox"/> INT_ST_FAP_SDI.sdi_hdr_err <input type="checkbox"/> INT_ST_FAP_SDI.sdi_pld_err <input type="checkbox"/> INT_ST_FAP_IPI.ipi_fifo_overflow <input type="checkbox"/> INT_ST_FAP_IPI.ipi_errline <input type="checkbox"/> INT_ST_FAP_IPI.ipi_fifo_underflow <input type="checkbox"/> INT_ST_FAP_IPI.ipi_trans_conflict <input type="checkbox"/> INT_ST_FAP_IPI.ipi2_errpixel <input type="checkbox"/> INT_ST_FAP_IPI.ipi2_fifo_overflow <input type="checkbox"/> INT_ST_FAP_IPI.ipi2_errline <input type="checkbox"/> INT_ST_FAP_IPI.ipi2_fifo_underflow <input type="checkbox"/> INT_ST_FAP_IPI.ipi3_errpixel <input type="checkbox"/> INT_ST_FAP_IPI.ipi3_fifo_overflow <input type="checkbox"/> INT_ST_FAP_IPI.ipi3_errline <input type="checkbox"/> INT_ST_FAP_IPI.ipi3_fifo_underflow <input type="checkbox"/> INT_ST_FAP_IPI.ipi4_errpixel <input type="checkbox"/> INT_ST_FAP_IPI.ipi4_fifo_overflow <input type="checkbox"/> INT_ST_FAP_IPI.ipi4_errline <input type="checkbox"/> INT_ST_FAP_IPI.ipi4_fifo_underflow <input type="checkbox"/> INT_ST_FAP_MT_IPI.mt_ipi_fifo_overflow <input type="checkbox"/> INT_ST_FAP_VPG.vpg_pkt_lost

Port Name	I/O	Description
diag_faults[15:0]...(cont...)	O.. .	<ul style="list-style-type: none">■ It is the antivalent output, which is duplicated to implement the anti-valent feature.<ul style="list-style-type: none"><input type="checkbox"/> 00: Invalid<input type="checkbox"/> 01: Active state<input type="checkbox"/> 10: Non-active state<input type="checkbox"/> 11: Invalid <p>Exists: (CSI2_DEVICE_AP) && (!CSI2_DEVICE_SS) Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: N/A</p>

4.6 IDI Interface Signals

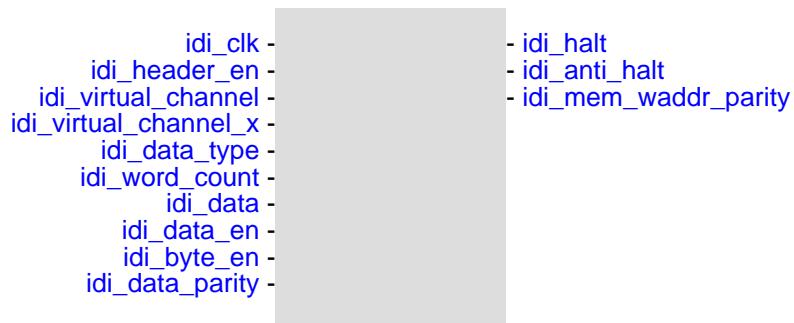


Table 4-6 IDI Interface Signals

Port Name	I/O	Description
idi_clk	I	<p>IDI Clock</p> <p>This is the Image Data Interface (IDI) Clock signal. All IDI signals are synchronous with this clock.</p> <p>Exists: CSI2_DEVICE_IDI_IF</p> <p>Synchronous To: None</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_header_en	I	<p>Packet Header Availability Indicator</p> <p>This signal informs that the header data at the input (signals virtual_channel, data_type, word_count, and ecc) is valid for the packet being transferred.</p> <p>This signal stays high during the complete packet transfer.</p> <p>Exists: CSI2_DEVICE_IDI_IF</p> <p>Synchronous To: idi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
idi_virtual_channel[1:0]	I	<p>Virtual Channel(VC) Identifier Value for the Packet Header</p> <p>This is bit[1:0] of the virtual channel, a part of the Data Identifier (DI) byte.</p> <ul style="list-style-type: none"> ■ 00: Virtual channel 0 ■ 01: Virtual channel 1 ■ 10: Virtual channel 2 ■ 11: Virtual channel 3 <p>Exists: CSI2_DEVICE_IDI_IF</p> <p>Synchronous To: idi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
idi_virtual_channel_x[(CSI2_DEVICE_VC_X_DWIDTH-1):0]	I	<p>Virtual Channel Extension(VCX) Identifier Value for the Packet Header This is bit[4:2] of the virtual channel on D-PHY mode, bit[4:2] of the virtual channel on D-PHY mode, a part of the Data Identifier (DI) byte.</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_VC_EXTENSION) Synchronous To: idi_clk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
idi_data_type[5:0]	I	<p>Data Type(DT) Value for the Packet Header This is a part of the Data Identifier (DI) byte.</p> <ul style="list-style-type: none"> ■ 0x00 - 0x07: Synchronization short packet data types ■ 0x08 - 0x0F: Generic short packet data types ■ 0x10 - 0x17: Generic long packet data types ■ 0x18 - 0x1F: YUV data ■ 0x20 - 0x27: RGB data ■ 0x28 - 0x2F: RAW data ■ 0x30 - 0x37: User defined byte-based data ■ 0x38 - 0x3F: Reserved <p>Exists: CSI2_DEVICE_IDI_IF Synchronous To: idi_clk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
idi_word_count[15:0]	I	<p>16-bit Word Count(WC) Information for the Packet Header This signal indicates the number of bytes and remains stable during the entire packet transfer.</p> <p>Exists: CSI2_DEVICE_IDI_IF Synchronous To: idi_clk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
idi_data[(CSI2_DEVICE_IDI_DATA_WIDT H-1):0]	I	<p>Payload Data This signal is formatted according to the CSI-2 recommended data storage format.</p> <p>Exists: CSI2_DEVICE_IDI_IF Synchronous To: idi_clk Registered: CSI2_DEVICE_AP ? "No" : "Yes" Power Domain: SINGLE_DOMAIN Active State: N/A</p>

Port Name	I/O	Description
idi_data_en	I	<p>New Payload data present This signal is only asserted when receiving long packets.</p> <p>Exists: CSI2_DEVICE_IDI_IF</p> <p>Synchronous To: idi_clk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
idi_byte_en[(CSI2_DEVICE_IDI_DATA_BYTEEN_WIDTH-1):0]	I	<p>Valid Bytes of idi_data This signal indicates the valid bytes of idi_data as follows:</p> <ul style="list-style-type: none"> ■ 0000: 1 valid byte in idi_data[7:0] ■ 0001: 2 valid bytes in idi_data[15:0] ■ 0010: 3 valid bytes in idi_data[23:0] ■ 0011: 4 valid bytes in idi_data[31:0] ■ 0100: 5 valid bytes in idi_data[39:0] ■ 0101: 6 valid bytes in idi_data[47:0] ■ 0110: 7 valid bytes in idi_data[55:0] ■ 0111: 8 valid bytes in idi_data[63:0] ■ 1000: 9 valid byte in idi_data[71:0] ■ 1001: 10 valid bytes in idi_data[79:0] ■ 1010: 11 valid bytes in idi_data[87:0] ■ 1011: 12 valid bytes in idi_data[95:0] ■ 1100: 13 valid bytes in idi_data[103:0] ■ 1101: 14 valid bytes in idi_data[111:0] ■ 1110: 15 valid bytes in idi_data[119:0] ■ 1111: 16 valid bytes in idi_data[127:0] <p>For YUV420_8L/YUV_422_8, the valid bytes are the reverse of the previous description.</p> <p>Exists: CSI2_DEVICE_IDI_IF</p> <p>Synchronous To: idi_clk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
idi_halt	O	<p>IDI Halt request This signal indicates IDI cannot receive the IDI input.</p> <p>Exists: CSI2_DEVICE_IDI_IF</p> <p>Synchronous To: CSI2_DEVICE_USE_SPRAM ? "idi_clk,lanebyteclk,pclk" : "idi_clk"</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

Port Name	I/O	Description
idi_anti_halt	O	<p>Antivalent IDI Halt request It is the antivalent idi_halt output, which in AP mode is duplicated to implement the antivalent feature. {idi_anti_halt,idi_halt}</p> <ul style="list-style-type: none"> ■ 00: Invalid ■ 01: Active state ■ 10: Non-active state ■ 11: Invalid <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_AP) Synchronous To: CSI2_DEVICE_USE_SRAM ? "idi_clk,lanebyteclk,pclk" : "idi_clk" Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
idi_data_parity[(CSI2_DEVICE_IDI_DATA_PARITY_WIDTH-1):0]	I	<p>Parity Data of Even Mode</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {idi_data_type,idi_data_en,idi_header_en}; ■ Bit 1 corresponding to the parity of idi_word_count[7:0] ■ Bit 2 corresponding to the parity of idi_word_count[15:8] ■ Bit 3 corresponding to the parity of {'b0,idi_byte_en}; ■ Bit 4 corresponding to the parity of {'b0,idi_virtual_channel_x,idi_virtual_channel} ■ Bit 5 corresponding to the parity of idi_data[7:0] ■ Bit 6 corresponding to the parity of idi_data[15:8] ■ Bit 7 corresponding to the parity of idi_data[23:16] ■ Bit 8 corresponding to the parity of idi_data[31:24] ■ Bit 9 corresponding to the parity of idi_data[39:32] ■ Bit 10 corresponding to the parity of idi_data[47:40] ■ Bit 11 corresponding to the parity of idi_data[55:48] ■ Bit 12 corresponding to the parity of idi_data[63:56] ■ Bit 13 corresponding to the parity of idi_data[71:64] ■ Bit 14 corresponding to the parity of idi_data[79:72] ■ Bit 15 corresponding to the parity of idi_data[87:80] ■ Bit 16 corresponding to the parity of idi_data[95:88] ■ Bit 17 corresponding to the parity of idi_data[103:96] ■ Bit 18 corresponding to the parity of idi_data[111:104] ■ Bit 19 corresponding to the parity of idi_data[119:112] ■ Bit 20 corresponding to the parity of idi_data[127:120] <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_AP) Synchronous To: idi_clk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

Port Name	I/O	Description
idi_mem_waddr_parity[(CSI2_DEVICE_ID_I_PLD_FIFO_PWIDTH-1):0]	O	<p>IDI Memory Write Address Parity Bus of Even Mode This signal is the parity of idi_mem_waddr. One bit parity is corresponding to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0: corresponding to the parity of {idi_mem_waddr[6:0],idi_mem_wen} ■ Bit 1 corresponding to the parity of idi_mem_waddr[14:7]... <p>Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SRAM) && (CSI2_DEVICE_AP)</p> <p>Synchronous To: idi_clk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

4.7 IDI Payload RAM Signals

id_i_r_dat_evn -
 id_i_r_dat_odd -
 id_i_mem_rdata -
 - id_i_ram_clk
 - id_i_en_evn_n
 - id_i_w_evn_n
 - id_i_addr_evn
 - id_i_addr_evn_parity
 - id_i_w_dat_evn
 - id_i_en_odd_n
 - id_i_w_odd_n
 - id_i_addr_odd
 - id_i_addr_odd_parity
 - id_i_w_dat_odd
 - id_i_mem_wclk
 - id_i_mem_waddr
 - id_i_mem_wen
 - id_i_mem_wdata
 - id_i_mem_rclk
 - id_i_mem_raddr
 - id_i_mem_raddr_parity
 - id_i_mem_ren

Table 4-7 IDI Payload RAM Signals

Port Name	I/O	Description
id_i_ram_clk	O	RAM Read/Write Clock(idi_clk or lanebyteclk) Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM) Synchronous To: pclk Registered: N/A Power Domain: SINGLE_DOMAIN Active State: N/A
id_i_en_evn_n	O	IDI Even RAM Enable Signal Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM) Synchronous To: idi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: Low
id_i_w_evn_n	O	IDI Even RAM Write Enable Signal Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM) Synchronous To: idi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: Low

Port Name	I/O	Description
idi_addr_evn[(CSI2_DEVICE_IDI_PLD_PTP_SRAM_AWIDTH-2):0]	O	<p>IDI Even RAM Write/Read Address Bus The memory size can be configured.</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_addr_evn_parity[(CSI2_DEVICE_IDI_PLD_FIFO_PWIDHT-1):0]	O	<p>IDI Even RAM Write/Read Address Parity Bus of Even Mode This signal is the parity of idi_addr_evn. One bit parity is corresponding to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0: corresponding to the parity of {idi_addr_evn[5:0], idi_en_evn_n, idi_w_evn_n} ■ Bit 1: corresponding to the parity of idi_addr_evn[13:6] <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM) && (CSI2_DEVICE_AP)</p> <p>Synchronous To: idi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_w_dat_evn[(CSI2_DEVICE_IDI_PLD_WIDTH-1):0]	O	<p>IDI Even RAM Write Data Bus</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_r_dat_evn[(CSI2_DEVICE_IDI_PLD_WIDTH-1):0]	I	<p>IDI Even RAM Read Data Bus</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk, lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_en_odd_n	O	<p>IDI Odd RAM Enable Signal</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: Low</p>

Port Name	I/O	Description
idi_w_odd_n	O	<p>IDI Odd RAM Write Enable Signal</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: Low</p>
idi_addr_odd[(CSI2_DEVICE_IDI_PLD_PTP_SPRAM_AWIDTH-2):0]	O	<p>IDI Odd RAM Write/Read Address Bus The memory size can be configured.</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_addr_odd_parity[(CSI2_DEVICE_IDI_PLD_FIFO_PWIDTH-1):0]	O	<p>IDI Odd RAM Write/Read Address Parity Bus of Even Mode This signal is the parity of idi_addr_odd. One bit parity is corresponding to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0: corresponding to the parity of {idi_addr_odd[5:0], idi_en_odd_n, idi_w_odd_n} ■ Bit 1: corresponding to the parity of idi_addr_odd[13:6] <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM) && (CSI2_DEVICE_AP)</p> <p>Synchronous To: idi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_w_dat_odd[(CSI2_DEVICE_IDI_PLD_WIDTH-1):0]	O	<p>IDI Odd RAM Write Data Bus</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_r_dat_odd[(CSI2_DEVICE_IDI_PLD_WIDTH-1):0]	I	<p>IDI Odd RAM Read Data Bus</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk, lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
idi_mem_wclk	O	<p>Memory Write Clock(idi_clk)</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: None</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_mem_waddr[(CSI2_DEVICE_IDI_PLD_FIFO_AWIDTH-1):0]	O	<p>IDI Memory Write Address Bus The memory size can be configured.</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_mem_wen	O	<p>IDI Memory Write Enable Signal</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
idi_mem_wdata[(CSI2_DEVICE_IDI_PLD_WIDTH-1):0]	O	<p>IDI Memory Write Data Bus</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: idi_clk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_mem_rclk	O	<p>IDI Memory Read Clock(lanebyteclk)</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: None</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
idi_mem_raddr[(CSI2_DEVICE_IDI_PLD_FIFO_AWIDTH-1):0]	O	<p>IDI Memory Read Address Bus The memory size can be configured.</p> <p>Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
idi_mem_raddr_parity[(CSI2_DEVICE_IDI_PLD_FIFO_PWIDTH-1):0]	O	<p>IDI Memory Read Address Parity Bus of Even Mode This signal is the parity of idi_mem_raddr. One bit parity is corresponding to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {idi_mem_raddr[6:0],idi_mem_ren} ■ Bit 1 corresponding to the parity of idi_mem_raddr[14:7]... <p>Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SRAM) && (CSI2_DEVICE_AP) Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
idi_mem_ren	O	<p>IDI Memory Read Enable Signal Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SRAM) Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
idi_mem_rdata[(CSI2_DEVICE_IDI_PLD_WIDTH-1):0]	I	<p>IDI Memory Read Data Bus Exists: (CSI2_DEVICE_IDI_IF) && (!CSI2_DEVICE_USE_SRAM) Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

4.8 SDI Interface Signals



Table 4-8 SDI Interface Signals

Port Name	I/O	Description
sdi_clk	O	SDI Clock (lanebyteclk) Exists: CSI2_DEVICE_SDI_IF Synchronous To: None Registered: N/A Power Domain: SINGLE_DOMAIN Active State: N/A
sdi_clkrstz	O	SDI Clock synchronized rising edge reset (lanebyteclkrstz) Exists: CSI2_DEVICE_SDI_IF Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: Low
sdi_pkt_start	I	A new packet is starting and packet Header Availability Indicator. Exists: CSI2_DEVICE_SDI_IF Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High
sdi_data[(CSI2_DEVICE_NAP_PLD_WIDTH-1):0]	I	SDI Data Exists: CSI2_DEVICE_SDI_IF Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A
sdi_pld_byte_en[(CSI2_DEVICE_SDI_PLD_BYTEEN_WIDTH-1):0]	I	Valid Bytes of sdi_data during payload transmission Exists: CSI2_DEVICE_SDI_IF Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High

Port Name	I/O	Description
sdi_pkt_end	I	<p>Packet end event Indicates a packet ends being transferred. For long packets, this indicates that CRC bus is valid. Otherwise, for short packets, it coincides with sdi_pkt_start.</p> <p>Exists: CSI2_DEVICE_SDIF Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
sdi_pkt_data_ack	O	<p>Packet transmission acknowledge Exists: CSI2_DEVICE_SDIF Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
sdi_anti_pkt_data_ack	O	<p>Antivalent Packet transmission acknowledge It is the antivalent sdi_pkt_data_ack output, which in AP mode is duplicated to implement the antivalent feature. {sdi_anti_pkt_data_ack,sdi_pkt_data_ack}</p> <ul style="list-style-type: none"> ■ 00: Invalid ■ 01: Active state ■ 10: Non-active state ■ 11: Invalid <p>Exists: (CSI2_DEVICE_SDIF) && (CSI2_DEVICE_AP) Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>

Port Name	I/O	Description
sdi_data_parity[(CSI2_DEVICE_SDI_IN_PARITY_WIDTH-1):0]	I	<p>SDI Parity Data of Even Mode</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {'b0,sdi_pld_byte_en,sdi_pld_last,sdi_pkt_end,sdi_pkt_start}}; ■ Bit 1 corresponding to the parity of sdi_data[7:0] ■ Bit 2 corresponding to the parity of sdi_data[15:8] ■ Bit 3 corresponding to the parity of sdi_data[23:16] ■ Bit 4 corresponding to the parity of sdi_data[31:24] ■ Bit 5 corresponding to the parity of sdi_data[39:32] ■ Bit 6 corresponding to the parity of sdi_data[47:40] ■ Bit 7 corresponding to the parity of sdi_data[55:48] ■ Bit 8 corresponding to the parity of sdi_data[63:56] ■ Bit 9 corresponding to the parity of sdi_data[71:64] ■ Bit 10 corresponding to the parity of sdi_data[79:72] ■ Bit 11 corresponding to the parity of sdi_data[87:80] ■ Bit 12 corresponding to the parity of sdi_data[95:88] ■ Bit 13 corresponding to the parity of sdi_data[103:96] ■ Bit 14 corresponding to the parity of sdi_data[111:104] ■ Bit 15 corresponding to the parity of sdi_data[119:112] ■ Bit 16 corresponding to the parity of sdi_data[127:120] <p>Exists: (CSI2_DEVICE_SDI_IF) && (CSI2_DEVICE_AP) Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

4.9 IPI Interface Signals

ipi_clk	- ipi_halt
ipi_data_0	- ipi_frame_number
ipi_data_en_0	- ipi_anti_halt
ipi_data_1	- ipi_out_parity
ipi_data_en_1	- ipiN_halt (for N = 2; N <= CSI2_DEVICE_NR_IPI)
ipi_data_2	- ipiN_frame_number (for N = 2; N <= CSI2_DEVICE_NR_IPI)
ipi_data_en_2	- ipiN_anti_halt (for N = 2; N <= CSI2_DEVICE_NR_IPI)
ipi_data_3	- ipiN_out_parity (for N = 2; N <= CSI2_DEVICE_NR_IPI)
ipi_data_en_3	
ipi_data	
ipi_data_en	
ipi_embedded	
ipi_embedded_wc	
ipi_vsync	
ipi_hsync	
ipi_data_parity	
ipiN_data_0 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_en_0 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_1 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_en_1 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_2 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_en_2 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_3 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_en_3 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_en (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_embedded (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_embedded_wc (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_vsync (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_hsync (for N = 2; N <= CSI2_DEVICE_NR_IPI)	
ipiN_data_parity (for N = 2; N <= CSI2_DEVICE_NR_IPI)	

Table 4-9 IPI Interface Signals

Port Name	I/O	Description
ipi_clk	I	<p>IPI Clock This signal is the Image Pixel Interface (IPI) Clock signal.</p> <p>Exists: CSI2_DEVICE_IPI_IF</p> <p>Synchronous To: None</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_data_0[47:0]	I	<p>IPI Pixel Data 0</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_data_en_0	I	<p>IPI Pixel Data Enable 0</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_data_1[47:0]	I	<p>IPI Pixel Data 1</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_data_en_1	I	<p>IPI Pixel Data Enable 1</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_data_2[47:0]	I	<p>IPI Pixel Data 2</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipi_data_en_2	I	<p>IPI Pixel Data Enable 2</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_data_3[47:0]	I	<p>IPI Pixel Data 3</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_data_en_3	I	<p>IPI Pixel Data Enable 3</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_data[47:0]	I	<p>IPI Pixel Data</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_data_en	I	<p>IPI Pixel Data Enable</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_IPI_MULTI_LANE)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_embedded	I	<p>IPI Embedded Data Enable</p> <p>Exists: CSI2_DEVICE_IPI_IF</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

Port Name	I/O	Description
ipi_embedded_wc[15:0]	I	<p>IPI Embedded Data Word Count</p> <p>Exists: CSI2_DEVICE_IPI_IF</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_vsync	I	<p>IPI Vertical Synchronism</p> <p>Exists: CSI2_DEVICE_IPI_IF</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_hsync	I	<p>IPI Horizontal Synchronism</p> <p>Exists: CSI2_DEVICE_IPI_IF</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_halt	O	<p>IPI Halt request</p> <p>This signal indicates the IPI cannot receive an IPI input.</p> <p>Exists: CSI2_DEVICE_IPI_IF</p> <p>Synchronous To: CSI2_DEVICE_USE_SPRAM ? "ipi_clk,lanebyteclk,pclk" : "ipi_clk"</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_frame_number[15:0]	O	<p>IPI Frame Number</p> <p>This signal indicates the IPI Frame Number.</p> <p>Exists: CSI2_DEVICE_IPI_IF</p> <p>Synchronous To: ipi_clk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipi_anti_halt	O	<p>Antivalent IPI Halt request It is the antivalent ipi_halt output, which in AP mode is duplicated to implement the antivalent feature. {ipi_anti_halt,ipi_halt}</p> <ul style="list-style-type: none"> ■ 00: Invalid ■ 01: Active state ■ 10: Non-active state ■ 11: Invalid <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_AP) Synchronous To: CSI2_DEVICE_USE_SPRAM ? "ipi_clk,lanebyteclk,pclk" : "ipi_clk" Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
ipi_data_parity[(CSI2_DEVICE_IPI_DATA _PARITY_WIDTH-1):0]	I	<p>IPI Parity Data of Even Mode</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {1'b0,ipi_data_en_3/1'b0,ipi_data_en_2/1'b0,ipi_data_en_1/1'b0,ipi_data_en_0/ipi_data_en,ipi_embedded,ipi_hsync,ipi_vsync}; ■ Bit 1 corresponding to the parity of ipi_embedded_wc[7:0] ■ Bit 2 corresponding to the parity of ipi_embedded_wc[15:8] ■ Bit 3 corresponding to the parity of ipi_data_0/ipi_data[7:0] ■ Bit 4 corresponding to the parity of ipi_data_0/ipi_data[15:8] ■ Bit 5 corresponding to the parity of ipi_data_0/ipi_data[23:16] ■ Bit 6 corresponding to the parity of ipi_data_0/ipi_data[31:24] ■ Bit 7 corresponding to the parity of ipi_data_0/ipi_data[39:32] ■ Bit 8 corresponding to the parity of ipi_data_0/ipi_data[47:40] ■ Bit 9 corresponding to the parity of ipi_data_1[7:0] ■ Bit 10 corresponding to the parity of ipi_data_1[15:8] ■ Bit 11 corresponding to the parity of ipi_data_1[23:16] ■ Bit 12 corresponding to the parity of ipi_data_1[31:24] ... ■ Bit 26 corresponding to the parity of ipi_data_3[47:40] <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_AP) Synchronous To: ipi_clk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

Port Name	I/O	Description
ipi_out_parity[1:0]	O	<p>IPI Output Parity Data of Even Mode This signal is an output along with the IPI. Bit 0 corresponding to the parity of ipi_frame_number[7:0] Bit 1 corresponding to the parity of ipi_frame_number[15:8]</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_AP)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_data_0[47:0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data 0</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_data_en_0 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data Enable Signal 0.</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_data_1[47:0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data 1</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_data_en_1 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data Enable Signal 1</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_data_2[47:0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data 2</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipiN_data_en_2 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data Enable Signal 2</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_data_3[47:0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data 3</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_data_en_3 (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data Enable Signal 3</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_data[47:0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_data_en (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Pixel Data Enable Signal</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_embedded (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIIn Embedded Data Enable Signal</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

Port Name	I/O	Description
ipiN_embedded_wc[15:0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIn Embedded Data Word Count</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_vsync (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIn Vertical Synchronism</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_hsync (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIn Horizontal Synchronism</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_halt (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Halt request</p> <p>This signal indicates IPIn cannot receive an IPIn input.</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: CSI2_DEVICE_USE_SPRAM ? "ipi_clk,lanebyteclk,pclk" : "ipi_clk"</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_frame_number[15:0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Frame Number</p> <p>This signal indicates the IPIn Frame Number.</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipiN_anti_halt (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>Antivalent IPI2 Halt request It is the antivalent ipi2_halt output, which in AP mode is duplicated to implement the antivalent feature. {ipi2_anti_halt,ipi2_halt}</p> <ul style="list-style-type: none"> ■ 00: Invalid ■ 01: Active state ■ 10: Non-active state ■ 11: Invalid <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI) Synchronous To: CSI2_DEVICE_USE_SRAM ? "ipi_clk,lanebyteclk,pclk" : "ipi_clk" Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>
ipiN_data_parity[(CSI2_DEVICE_IPI_DAT_A_PARITY_WIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIn Parity Data of Even Mode</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {1'b0,ipi2_data_en_3/1'b0,ipi2_data_en_2/1'b0,ipi2_data_en_1/1'b0,ipi2_data_en_0/ipi2_data_en,ipi2_embedded,ipi2_hsync,ipi2_vsync}; ■ Bit 1 corresponding to the parity of ipi2_embedded_wc[7:0] ■ Bit 2 corresponding to the parity of ipi2_embedded_wc[15:8] ■ Bit 3 corresponding to the parity of ipi2_data_0/ipi2_data[7:0] ■ Bit 4 corresponding to the parity of ipi2_data_0/ipi2_data[15:8] ■ Bit 5 corresponding to the parity of ipi2_data_0/ipi2_data[23:16] ■ Bit 6 corresponding to the parity of ipi2_data_0/ipi2_data[31:24] ■ Bit 7 corresponding to the parity of ipi2_data_0/ipi2_data[39:32] ■ Bit 8 corresponding to the parity of ipi2_data_0/ipi2_data[47:40] ■ Bit 9 corresponding to the parity of ipi2_data_1[7:0] ■ Bit 10 corresponding to the parity of ipi2_data_1[15:8] ■ Bit 11 corresponding to the parity of ipi2_data_1[23:16] ■ Bit 12 corresponding to the parity of ipi2_data_1[31:24] ... ■ Bit 26 corresponding to the parity of ipi2_data_3[47:40] <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI) Synchronous To: ipi_clk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

Port Name	I/O	Description
ipiN_out_parity[1:0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Output Parity Data of Even Mode This signal is an output along with the IPI2.</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of ipi2_frame_number[7:0] ■ Bit 1 corresponding to the parity of ipi2_frame_number[15:8] <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI) Synchronous To: ipi_clk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>

4.10 IPI Payload RAM Signals

ipi_r_dat_evn -
ipi_r_dat_odd -
ipi_mem_rdata -
ipiN_r_dat_evn (for N = 2; N <= CSI2_DEVICE_NR_IPI) -
ipiN_r_dat_odd (for N = 2; N <= CSI2_DEVICE_NR_IPI) -
ipiN_mem_rdata (for N = 2; N <= CSI2_DEVICE_NR_IPI) -

- ipi_ram_clk
- ipi_en_evn_n
- ipi_w_evn_n
- ipi_addr_evn
- ipi_addr_evn_parity
- ipi_w_dat_evn
- ipi_en_odd_n
- ipi_w_odd_n
- ipi_addr_odd
- ipi_addr_odd_parity
- ipi_w_dat_odd
- ipi_mem_wclk
- ipi_mem_waddr
- ipi_mem_waddr_parity
- ipi_mem_wen
- ipi_mem_wdata
- ipi_mem_rclk
- ipi_mem_raddr
- ipi_mem_raddr_parity
- ipi_mem_ren
- ipiN_ram_clk (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_en_evn_n (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_w_evn_n (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_addr_evn (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_addr_evn_parity (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_w_dat_evn (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_en_odd_n (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_w_odd_n (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_addr_odd (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_addr_odd_parity (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_w_dat_odd (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_mem_wclk (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_mem_waddr (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_mem_waddr_parity (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_mem_wen (for N = 2; N <= CSI2_DEVICE_NR_IPI)
- ipiN_mem_wdata (for N = 2; N <= CSI2_DEVICE_NR_IPI)

- `ipiN_mem_rclk` (for N = 2; N <= CSI2_DEVICE_NR_IPI)
 - `ipiN_mem_raddr` (for N = 2; N <= CSI2_DEVICE_NR_IPI)
 - `ipiN_mem_raddr_parity` (for N = 2; N <= CSI2_DEVICE_NR_IPI)
 - `ipiN_mem_ren` (for N = 2; N <= CSI2_DEVICE_NR_IPI)

Table 4-10 IPI Payload RAM Signals

Port Name	I/O	Description
<code>ipi_ram_clk</code>	O	ERAM Read/Write Clock(ipi_clk or lanebyteclk) Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM) Synchronous To: pclk Registered: N/A Power Domain: SINGLE_DOMAIN Active State: N/A
<code>ipi_en_evn_n</code>	O	IPI Even RAM Enable Signal Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM) Synchronous To: ipi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: Low
<code>ipi_w_evn_n</code>	O	IPI Even RAM Write Enable Signal Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM) Synchronous To: ipi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: Low
<code>ipi_addr_evn[(CSI2_DEVICE_IPI_PLD_PTP_SPRAM_AWIDTH-2):0]</code>	O	IPI Even RAM Write/Read Address Bus The memory size can be configured. Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM) Synchronous To: ipi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A

Port Name	I/O	Description
ipi_addr_evn_parity[(CSI2_DEVICE_IPI_PLD_FIFO_PWIDTH-1):0]	O	<p>IPI Even RAM Write/Read Address Parity Bus of Even Mode This signal is the parity of ipi_addr_evn. One bit parity is corresponding to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0: corresponding to the parity of {ipi_addr_evn[5:0],ipi_en_evn_n,ipi_w_evn_n} ■ Bit 1: corresponding to the parity of ipi_addr_evn[13:6]... <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM) && (CSI2_DEVICE_AP)</p> <p>Synchronous To: ipi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
ipi_w_dat_evn[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0]	O	<p>IPI Even RAM Write Data Bus</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: ipi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
ipi_r_dat_evn[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0]	I	<p>IPI Even RAM Read Data Bus</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: CSI2_DEVICE_AP ? "ipi_ram_clk,ipi_clk,lanebyteclk" : "ipi_clk,lanebyteclk"</p> <p>Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A</p>
ipi_en_odd_n	O	<p>IPI Odd RAM Enable Signal</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: ipi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: Low</p>
ipi_w_odd_n	O	<p>IPI Odd RAM Write Enable Signal</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: ipi_clk,lanebyteclk,pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: Low</p>

Port Name	I/O	Description
ipi_addr_odd[(CSI2_DEVICE_IPI_PLD_PTP_SRAM_AWIDTH-2):0]	O	<p>IPI Odd RAM Write/Read Address Bus The memory size can be configured.</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: ipi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_addr_odd_parity[(CSI2_DEVICE_IPI_PLD_FIFO_PWIDHT-1):0]	O	<p>IPI Odd RAM Write/Read Address Parity Bus of Even Mode This signal is the parity of ipi_addr_odd. One bit parity is corresponding to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0: corresponding to the parity of {ipi_addr_odd[5:0], ipi_en_odd_n, ipi_w_odd_n} ■ Bit 1: corresponding to the parity of ipi_addr_odd[13:6]... <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM) && (CSI2_DEVICE_AP)</p> <p>Synchronous To: ipi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_w_dat_odd[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0]	O	<p>IPI Odd RAM Write Data Bus</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: ipi_clk, lanebyteclk, pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_r_dat_odd[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0]	I	<p>IPI Odd RAM Read Data Bus</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: CSI2_DEVICE_AP ? "ipi_ram_clk, ipi_clk, lanebyteclk" : "ipi_clk, lanebyteclk"</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_mem_wclk	O	<p>IPI Memory Write Clock (ipi_clk)</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: None</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipi_mem_waddr[(CSI2_DEVICE_IPI_PLD_FIFO_AWIDTH-1):0]	O	<p>IPI Memory Write Address Bus The memory size is parameterizable.</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SRAM)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_mem_waddr_parity[(CSI2_DEVICE_IP_L_PLD_FIFO_PWIDTH-1):0]	O	<p>IPI Memory Write Address Parity Bus of Even Mode This signal is the parity of ipi_mem_waddr. One bit parity corresponds to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {ipi_mem_waddr[6:0],ipi_mem_wen} ■ Bit 1 corresponding to the parity of ipi_mem_waddr[14:7]... <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SRAM) && (CSI2_DEVICE_AP)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_mem_wen	O	<p>IPI Memory Write Enable</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SRAM)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_mem_wdata[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0]	O	<p>IPI Memory Write Data Bus</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SRAM)</p> <p>Synchronous To: ipi_clk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_mem_rclk	O	<p>IPI Memory Read Clock (lanebyteclk)</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SRAM)</p> <p>Synchronous To: None</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipi_mem_raddr[(CSI2_DEVICE_IPI_PLD_FIFO_AWIDTH-1):0]	O	<p>IPI Memory Read Address Bus The memory size can be configured.</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_mem_raddr_parity[(CSI2_DEVICE_IPI_PLD_FIFO_PWIDTH-1):0]	O	<p>IPI Memory Read Address Parity Bus of Even Mode This signal is the parity of ipi_mem_waddr. One bit parity corresponds to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {ipi_mem_raddr[6:0],ipi_mem_ren} ■ Bit 1 corresponding to the parity of ipi_mem_raddr[14:7]... <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SPRAM) && (CSI2_DEVICE_AP)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipi_mem_ren	O	<p>IPI Memory Read Enable</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipi_mem_rdata[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0]	I	<p>IPI Memory Read Data Bus</p> <p>Exists: (CSI2_DEVICE_IPI_IF) && (!CSI2_DEVICE_USE_SPRAM)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_ram_clk (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn RAM Read/Write Clock (ipi_clk or lanebyteclk)</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: pclk</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipiN_en_evn_n (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Even RAM Enable Signal</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: Low</p>
ipiN_w_evn_n (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Even RAM Write Enable Signal</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: Low</p>
ipiN_addr_evn[(CSI2_DEVICE_IPI2_PLD _PTP_SPRAM_AWIDTH-2):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Even RAM Write/Read Address Bus</p> <p>The memory size can be configured.</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_addr_evn_parity[(CSI2_DEVICE_IPI 2_PLD_FIFO_PWIDHT-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Even RAM Write/Read Address Parity Bus of Even Mode</p> <p>This signal is the parity of ipiN_addr_evn. One bit parity corresponds to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {ipiN_ad-dr_evn[5:0],ipiN_en_evn_n,ipiN_w_evn_n} ■ Bit 1 corresponding to the parity of ipiN_addr_evn[13:6]... <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_w_dat_evn[(CSI2_DEVICE_IPI_PLD _WIDHT-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Even RAM Write Data Bus</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipiN_r_dat_ev[n[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)]	I	<p>IPIn Even RAM Read Data Bus</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_en_odd_n (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Odd RAM Enable Signal</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: Low</p>
ipiN_w_odd_n (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Odd RAM Write Enable Signal</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: Low</p>
ipiN_addr_odd[n[(CSI2_DEVICE_IPI2_PLD_PTP_SPRAM_AWIDTH-2):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)]	O	<p>IPIn Odd RAM Write/Read Address Bus</p> <p>The memory size can be configured.</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_addr_odd_parity[n[(CSI2_DEVICE_IPI2_PLD_FIFO_PWIDHT-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)]	O	<p>IPIn Odd RAM Write/Read Address Parity Bus of Even Mode</p> <p>This signal is the parity of ipiN_addr_odd. One bit parity corresponds to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {ipiN_addr_odd[5:0], ipiN_en_odd_n, ipiN_w_odd_n} ■ Bit 1 corresponding to the parity of ipiN_addr_odd[13:6]... <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipiN_w_dat_odd[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Odd RAM Write Data Bus</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_r_dat_odd[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPIn Odd RAM Read Data Bus</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_ram_clk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_mem_wclk (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Memory Write Clock (ipi_clk)</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: None</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_mem_waddr[(CSI2_DEVICE_IPI2_PLD_FIFO_AWIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Memory Write Address Bus</p> <p>The memory size can be configured.</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_mem_wclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_mem_waddr_parity[(CSI2_DEVICE_IPI2_PLD_FIFO_PWIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Memory Write Address Parity Bus of Even Mode</p> <p>This signal is the parity of ipiN_mem_waddr. One bit parity corresponds to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {ipiN_mem_waddr[6:0],ipiN_mem_wen} ■ Bit 1 corresponding to the parity of ipiN_mem_waddr[14:7]... <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_mem_wclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipiN_mem_wen (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Memory Write Enable</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_mem_wclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
ipiN_mem_wdata[(CSI2_DEVICE_IPI_PL_D_WIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Memory Write Data Bus</p> <p>Exists: (2<=N<=CSI2_DEVICE_N)</p> <p>Synchronous To: ipiN_mem_wclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_mem_rclk (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Memory Read Clock (lanebyteclk)</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: None</p> <p>Registered: N/A</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_mem_raddr[(CSI2_DEVICE_IPI2_PL_D_FIFO_AWIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Memory Read Address Bus</p> <p>The memory size can be configured.</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_mem_rclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_mem_raddr_parity[(CSI2_DEVICE_IPI2_PLD_FIFO_PWIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPIn Memory Read Address Parity Bus of Even Mode</p> <p>This signal is the parity of ipiN_mem_waddr. One bit parity corresponds to one byte address.</p> <ul style="list-style-type: none"> ■ Bit 0 corresponding to the parity of {ipiN_mem_raddr[6:0], ipiN_mem_ren} ■ Bit 1 corresponding to the parity of ipiN_mem_raddr[14:7]... <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_mem_rclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
ipiN_mem_ren (for N = 2; N <= CSI2_DEVICE_NR_IPI)	O	<p>IPI In Memory Read Enable</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_mem_rclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
ipiN_mem_rdata[(CSI2_DEVICE_IPI_PLD_WIDTH-1):0] (for N = 2; N <= CSI2_DEVICE_NR_IPI)	I	<p>IPI In Memory Read Data Bus</p> <p>Exists: (2<=N<=CSI2_DEVICE_NR_IPI)</p> <p>Synchronous To: ipiN_mem_rclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

4.11 PHY Signals



Table 4-11 PHY Signals

Port Name	I/O	Description
phy_mode	O	<p>PHY Mode This signal is used to select D-PHY mode or C-PHY mode.</p> <ul style="list-style-type: none"> ■ 0: D-PHY mode ■ 1: C-PHY mode <p>Exists: !(CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE) && (CSI2_DEVICE_CPHY_TYPE) Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: N/A</p>
txdatawidthhs[1:0]	O	<p>PHY Lane Data Width This signal is used to select the PHY data lane width.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: Low</p>
physhutdownz	O	<p>PHY Shutdown This signal is directly controlled by the PHY_RSTZ register.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: Low</p>

Port Name	I/O	Description
phyrstz	O	<p>PHY Reset This signal is directly controlled by the PHY_RSTZ register. Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: Low</p>
forcepll	O	<p>PHY PLL Enable This signal is directly controlled by the PHY_RSTZ register. Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High</p>
phy_forcetxstopmode	O	<p>Force PHY Return to Stop Mode This signal is directly controlled by the PHY_RSTZ register. Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High</p>
pll_lock	I	<p>PHY PLL Locked This signal indicates the PLL in the PHY has been locked and provides a stable output (lanebyteclk). Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: Asynchronous Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High</p>
txescclk	O	<p>PHY Escape Mode Clock This signal is triggered in ULPS mode. Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: lanebyteclk Registered: N/A Power Domain: SINGLE_DOMAIN Active State: N/A</p>

Port Name	I/O	Description
enable_clk	O	<p>PHY Clock Enable This signal is directly controlled by the PHY_RSTZ register. Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE) Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High</p>
enable_In (for N = 0; N <= 7)	O	<p>Data Lane N Enable Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: None Registered: No Power Domain: SINGLE_DOMAIN Active State: High</p>

4.12 PHY Protocol Interface (PPI) Signals

lanebyteclk -

Table 4-12 PHY Protocol Interface (PPI) Signals

Port Name	I/O	Description
lanebyteclk	I	<p>High-Speed Receive Clock This is used to synchronize the PPI signals in the high-speed receive clock domain.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: None Registered: N/A Power Domain: SINGLE_DOMAIN Active State: N/A</p>

4.13 PHY Protocol Interface (PPI) Signals for D-PHY Clock Lane



Table 4-13 PHY Protocol Interface (PPI) Signals for D-PHY Clock Lane

Port Name	I/O	Description
txrequesths_clk	O	<p>Clock Lane High-Speed Request This signal is used to request clock lane to enter high-speed mode.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txulpsesc_clk	O	<p>Clock Lane ULPS Request This is used to request a clock lane to enter ULPS mode.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txulpsexit_clk	O	<p>Clock Lane ULPS Exit Request This is used to request a clock lane to exit ULPS mode.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txstopstate_clk	I	<p>Clock Lane in Stop State This signal indicates that the clock lane is in a Stop state and is asynchronous to any clock in the PPI interface.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE)</p> <p>Synchronous To: Asynchronous</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

Port Name	I/O	Description
txulpsactivenot_clk	I	<p>Clock Lane in ULP State This signal indicates that the clock lane has entered the Ultra Low Power State. This signal is kept low until a Stop state is sent or detected the lane is interconnect.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE)</p> <p>Synchronous To: Asynchronous</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: Low</p>

4.14 PHY Protocol Interface (PPI) Signals for D-PHY

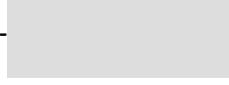
txhsidleclkreadyhs -  - txhsidleclkhs

Table 4-14 PHY Protocol Interface (PPI) Signals for D-PHY

Port Name	I/O	Description
txhsidleclkhs	O	<p>HS-Idle State Start This is an optional signal to initiate the HS-Idle State at the transmitter. A high lever on TxHSIdleClkHS directs the PHY to start HS-Idle-Post sub-state when payload transmission on all Data Lanes is completed.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txhsidleclkreadyhs	I	<p>Clock Ready to Exit HS-Idle-ClkHS0 Sub-State This optional active high signal indicates that the transmitter is ready to exit HS-Idle-ClkHS0 sub-state.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_DPHY_TYPE)</p> <p>Synchronous To: Asynchronous</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

4.15 PHY Protocol Interface (PPI) Signals for Data Lane N

txreadyhs_Io txstopstate_In (for N = 0; N <= 7) txulpactiveNot_In (for N = 0; N <= 7) errcontentionlp0_Io errcontentionlp1_Io	txrequesths_In (for N = 0; N <= 7) txdatahs_In (for N = 0; N <= 7) txdatavalidhs_In (for N = 0; N <= 3) txsendsynchs_In (for N = 0; N <= 3) txsynctypehs0_In (for N = 0; N <= 3) txsynctypehs1_In (for N = 0; N <= 3) txsendalphs_In (for N = 0; N <= 3) txalpcodehs0_In (for N = 0; N <= 3) txalpcodehs1_In (for N = 0; N <= 3) txrequestesc_In (for N = 0; N <= 7) txulpsesc_In (for N = 0; N <= 7) txulpsexit_In (for N = 0; N <= 7) txalpcodehs0_I3 txalpcodehs1_I3
---	--

Table 4-15 PHY Protocol Interface (PPI) Signals for Data Lane N

Port Name	I/O	Description
txrequesths_In (for N = 0; N <= 7)	O	Data Lane N High-Speed Request This signal is used to request Data Lane N to enter High-Speed mode. Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: lanebyteclk Registered: No Power Domain: SINGLE_DOMAIN Active State: High
txdatahs_In[(CSI2_DEVICE_PHY_LANE0 _WIDTH-1):0] (for N = 0; N <= 7)	O	Data Lane N High-Speed Transmission Data Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: lanebyteclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: N/A
txdatavalidhs_In[(CSI2_DEVICE_PHY_VA LID_WIDTH-1):0] (for N = 0; N <= 3)	O	Data Lane N High-Speed Data Valid Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_PHY_16BIT_MODE) Synchronous To: lanebyteclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: N/A

Port Name	I/O	Description
txsendsynchs_In[(CSI2_DEVICE_CPHY_SS_WIDTH-1):0] (for N = 0; N <= 3)	O	<p>Data Lane N High-Speed Send Sync Word</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_CPHY_TYPE)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
txsynctypehs0_In[(CSI2_DEVICE_CPHY_ST_WIDTH-1):0] (for N = 0; N <= 3)	O	<p>Data Lane N High-Speed Sync Word Type for C-PHY 16-bit Width</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_CPHY_TYPE) && (CSI2_DEVICE_DATA_SCRAMBLING)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
txsynctypehs1_In[(CSI2_DEVICE_CPHY_ST_WIDTH-1):0] (for N = 0; N <= 3)	O	<p>Data Lane N High-Speed Sync Word Type for C-PHY 32-bit Width</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_CPHY_TYPE) && (CSI2_DEVICE_DATA_SCRAMBLING) && (CSI2_DEVICE_CPHY_32BIT_MODE)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
txsendalpshs_In[(CSI2_DEVICE_CPHY_ALP_HS_RS-1):0] (for N = 0; N <= 3)	O	<p>Data Lane N High-Speed Command to Transmit ALPS Code</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_CPHY_TYPE) && (CSI2_DEVICE_CPHY_ALP_EN)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
txalpcodesh0_In[(CSI2_DEVICE_CPHY_ALP_CODE_RS-1):0] (for N = 0; N <= 3)	O	<p>Data Lane N High-Speed ALPS Code for Low 16-bit</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (CSI2_DEVICE_CPHY_TYPE) && (CSI2_DEVICE_CPHY_ALP_EN)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

Port Name	I/O	Description
txalpcodehs1_In[(CSI2_DEVICE_CPHY_ALP_CODE_RS-1):0] (for N = 0; N <= 3)	O	<p>Data Lane N High-Speed ALPS Code for High 16-bit</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY && (CSI2_DEVICE_CPHY_TYPE) && (CSI2_DEVICE_CPHY_ALP_EN) && (CSI2_DEVICE_CPHY_32BIT_MODE)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
txreadyhs_I0	I	<p>Data Lane N Ready For Transmission</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txrequestesc_In (for N = 0; N <= 7)	O	<p>Data Lane 0 Escape Mode Request</p> <p>This signal is used to request Data Lane 0 to enter Escape mode.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txulpsesc_In (for N = 0; N <= 7)	O	<p>Data Lane 0 ULPS Request</p> <p>This signal is used to request Data Lane N to enter ULPS mode; the request occurs when txrequestesc_In is asserted.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txulpsexit_In (for N = 0; N <= 7)	O	<p>Data Lane N Exit ULPS Request</p> <p>This signal is used to request Data Lane N to exit ULPS mode; the request occurs when txrequestesc_In de-asserted.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

Port Name	I/O	Description
txstopstate_In (for N = 0; N <= 7)	I	<p>Data Lane N in Stop State This signal indicates that the Data Lane N is in a Stop state; it is asynchronous to any clock in the PPI interface.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: lanebyteclk,pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High</p>
txulpsactivenot_In (for N = 0; N <= 7)	I	<p>Data Lane N in ULP State This signal indicates that the Data Lane N has entered Ultra Low Power State. This signal is kept low until a Stop state is sent or detected the lane is interconnect.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: Low</p>
errcontentionlp0_I0	I	<p>Data Lane 0 in LP0 Contention Error This signal indicates LP0 contention error ErrContentionLP1 from Lane 0.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High</p>
errcontentionlp1_I0	I	<p>Data Lane 0 in LP1 Contention Error This signal indicates LP1 contention error ErrContentionLP1 from Lane 0.</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High</p>

Port Name	I/O	Description
txalpcodes0_l3[(CSI2_DEVICE_CPHY_ALP_CODE_RS-1):0]	O	<p>Data Lane 3 High-Speed ALPS Code for Low 16-bit</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (((CSI2_DEVICE_DPHY_INCLUDE_LANE_4) (CSI2_DEVICE_CPHY_INCLUDE_LANE_4))) && (((CSI2_DEVICE_CPHY_TYPE) & (CSI2_DEVICE_CPHY_NUM_OF_LANES >= 4))) && (CSI2_DEVICE_CPHY_ALP_EN)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
txalpcodes1_l3[(CSI2_DEVICE_CPHY_ALP_CODE_RS-1):0]	O	<p>Data Lane 3 High-Speed ALPS Code for High 16-bit</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (((CSI2_DEVICE_DPHY_INCLUDE_LANE_4) (CSI2_DEVICE_CPHY_INCLUDE_LANE_4))) && (((CSI2_DEVICE_CPHY_TYPE) & (CSI2_DEVICE_CPHY_NUM_OF_LANES >= 4))) && (CSI2_DEVICE_CPHY_ALP_EN) && (CSI2_DEVICE_CPHY_32BIT_MODE)</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

4.16 Parallel Port for PHY Configuration Signals

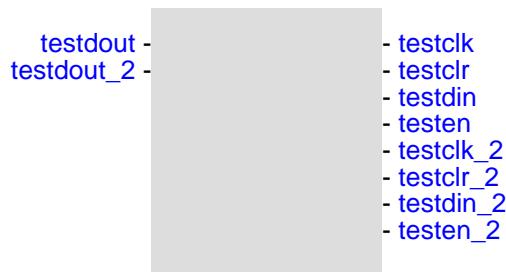
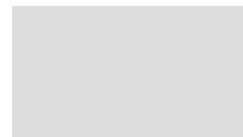


Table 4-16 Parallel Port for PHY Configuration Signals

Port Name	I/O	Description
testclk	O	PHY Test Clock Exists: !ICSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: N/A
testclr	O	PHY Test Clear Exists: !ICSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: High
testdin[7:0]	O	PHY Test Data Output Exists: !ICSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: Yes Power Domain: SINGLE_DOMAIN Active State: N/A
testdout[7:0]	I	PHY Test Data Input Exists: !ICSI2_DEVICE_SNPS_PHY Synchronous To: pclk Registered: No Power Domain: SINGLE_DOMAIN Active State: N/A

Port Name	I/O	Description
testen	O	<p>PHY Test Enable</p> <p>Exists: !CSI2_DEVICE_SNPS_PHY</p> <p>Synchronous To: pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
testclk_2	O	<p>Second PHY Test Clock</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && ((CSI2_DEVICE_DPHY_INCLUDE_LANE_5))</p> <p>Synchronous To: pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
testclr_2	O	<p>Second PHY Test Clear</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && ((CSI2_DEVICE_DPHY_INCLUDE_LANE_5))</p> <p>Synchronous To: pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
testdin_2[7:0]	O	<p>The second PHY Test Data Output.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && ((CSI2_DEVICE_DPHY_INCLUDE_LANE_5))</p> <p>Synchronous To: pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
testdout_2[7:0]	I	<p>Second PHY Test Data Input</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && ((CSI2_DEVICE_DPHY_INCLUDE_LANE_5))</p> <p>Synchronous To: pclk</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>
testen_2	O	<p>Second PHY Test Enable</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && ((CSI2_DEVICE_DPHY_INCLUDE_LANE_5))</p> <p>Synchronous To: pclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: N/A</p>

4.17 PHY Protocol Interface (PPI) Signals for Calibration Signals



- txskewcalhs
- txeqactivehs
- txeqlevelhs

Table 4-17 PHY Protocol Interface (PPI) Signals for Calibration Signals

Port Name	I/O	Description
txskewcalhs	O	<p>Calibration Output Pin to D-PHY A low-to-high transition on txskewcalhs signal causes the PHY to initiate the transmission of a skew calibration pattern.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (((CSI2_DEVICE_GEN3DPHY == 1) (CSI2_DEVICE_SNPS_PHY == 0)))</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txeqactivehs	O	<p>Equalization Active Output Pin to D-PHY A high-level on txeqactivehs signal means that the PHY has enabled equalization.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (((CSI2_DEVICE_GEN3DPHY == 1) (CSI2_DEVICE_SNPS_PHY == 0)))</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
txeqlevelhs	O	<p>Equalization Level Output Pin to D-PHY This signal indicates the PHY equalization level.</p> <p>Exists: (!CSI2_DEVICE_SNPS_PHY) && (((CSI2_DEVICE_GEN3DPHY == 1) (CSI2_DEVICE_SNPS_PHY == 0)))</p> <p>Synchronous To: lanebyteclk</p> <p>Registered: CSI2_DEVICE_AP ? "No" : "Yes"</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

4.18 Scan Chain Signals

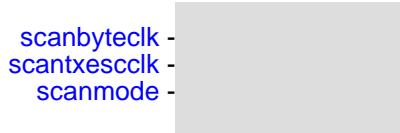


Table 4-18 Scan Chain Signals

Port Name	I/O	Description
scanbyteclk	I	<p>Scan Clock for lanebyteclk Domain</p> <p>Exists: !!CSI2_DEVICE_SNPS_PHY</p> <p>Synchronous To: None</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
scantxescclk	I	<p>Scan Clock for D-PHY ESC Domain</p> <p>Exists: (!!CSI2_DEVICE_SNPS_PHY) && ([<functionof>]>=4) && (!([<functionof>]==4))</p> <p>Synchronous To: None</p> <p>Registered: Yes</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>
scanmode	I	<p>Scan Mode Selection</p> <p>Assert high to configure the controller for Scan-Chain operation. Assert low for normal operation.</p> <p>Exists: Always</p> <p>Synchronous To: Asynchronous</p> <p>Registered: No</p> <p>Power Domain: SINGLE_DOMAIN</p> <p>Active State: High</p>

Register Descriptions

This chapter details all possible registers in the controller. They are arranged hierarchically into maps and blocks (banks). For configurable IP titles, your actual configuration might not contain all of these registers.

Attention: For configurable IP titles, do not use this document to determine the exact attributes of your register map. It is for reference purposes only.

When you configure the controller in coreConsultant, you must access the register attributes for your actual configuration at `workspace/report/ComponentRegisters.html` or `workspace/report/ComponentRegisters.xml` after you have completed the report creation activity. That report comes from the exact same source as this chapter but removes all the registers that are not in your actual configuration. This does not apply to non-configurable IP titles. In addition, all parameter expressions are evaluated to actual values. Therefore, the Offset and Memory Access values might change depending on your actual configuration.

Some expressions might refer to TCL functions or procedures (sometimes identified as `<functionof>`) that coreConsultant uses to make calculations. The exact formula used by these TCL functions is not provided in this chapter. However, when you configure the controller in coreConsultant, all TCL functions and parameters are evaluated completely; and the resulting values are displayed where appropriate in the coreConsultant GUI reports.

Exists Expressions

These expressions indicate the combination of configuration parameters required for a register, field, or block to exist in the memory map. The expression is only valid in the local context and does not indicate the conditions for existence of the parent. For example, the expression for a bit field in a register assumes that the register exists and does not include the conditions for existence of the register.

Offset

The term *Offset* is synonymous with *Address*.

Memory Access Attributes

The Memory Access attribute is defined as `<ReadBehavior>/<WriteBehavior>` which are defined in the following table.

Table 5-1 Possible Read and Write Behaviors

Read (or Write) Behavior	Description
RC	A read clears this register field.
RS	A read sets this register field.
RM	A read modifies the contents of this register field.
Wo	You can only write once to this register field.
W1C	A write of 1 clears this register field.
W1S	A write of 1 sets this register field.
W1T	A write of 1 toggles this register field.
W0C	A write of 0 clears this register field.
W0S	A write of 0 sets this register field.
W0T	A write of 0 toggles this register field.
WC	Any write clears this register field.
WS	Any write sets this register field.
WM	Any write toggles this register field.
no Read Behavior attribute	You cannot read this register. It is Write-Only.
no Write Behavior attribute	You cannot write to this register. It is Read-Only.

Table 5-2 Memory Access Examples

Memory Access	Description
R	Read-only register field.
W	Write-only register field.
R/W	Read/write register field.
R/W1C	You can read this register field. Writing 1 clears it.
RC/W1C	Reading this register field clears it. Writing 1 clears it.
R/Wo	You can read this register field. You can only write to it once.

Special Optional Attributes

Some register fields might use the following optional attributes.

Table 5-3 Optional Attributes

Attribute	Description
Volatile	As defined by the IP-XACT specification. If true, indicates in the case of a write followed by read, or in the case of two consecutive reads, there is no guarantee as to what is returned by the read on the second transaction or that this return value is consistent with the write or read of the first transaction. The element implies there is some additional mechanism by which this field can acquire new values other than by reads/writes/resets and other access methods known to IP-XACT. For example, when the controller updates the register field contents.
Testable	As defined by the IP-XACT specification. Possible values are unconstrained, untestable, readOnly, writeAsRead, restore. Untestable means that this field is untestable by a simple automated register test. For example, the read-write access of the register is controlled by a pin or another register. readOnly means that you should not write to this register; only read from it. This might apply for a register that modifies the contents of another register.
Reset Mask	As defined by the IP-XACT specification. Indicates that this register field has an unknown reset value. For example, the reset value is set by another register or an input pin; or the register is implemented using RAM.
* Varies	Indicates that the memory access (or reset) attribute (read, write behavior) is not fixed. For example, the read-write access of the register is controlled by a pin or another register. Or when the access depends on some configuration parameter; in this case the post-configuration report in coreConsultant gives the actual access value.

5.1 Component Memory Maps

The following table documents the memory maps visible in this component.

Table 5-4 DWC_mipicsi2_device Memory Maps

Memory Map	Description
"Registers for the DWC_mipicsi2_device_MemMap Memory Map" on page 197	DWC_mipicsi2_device Memory Map DWC_mipicsi2_device_MemMap Exists: Always
"Registers for the DW_safety_slave_map Memory Map" on page 638	DW Safety Slave Memory Map Exists: CSI2_DEVICE_SS==1

5.2 DWC_mipicsi2_device_MemMap Memory Map Registers

Register definitions for each component memory map.

Table 5-5 Registers for the DWC_mipicsi2_device_MemMap Memory Map

Register	Offset	Description
MAIN MAIN Exists: Always		
"VERSION" on page 208	0x0	Controller Version identification.
"CSI2_RESETN" on page 209	0x4	Soft reset.
"DATA_SCRAMBLING" on page 210	0x8	Data scrambling control.
"VER_TYPE" on page 211	0xc	Controller Version TYPE identification.
"SS_CTRL" on page 213	0x10	Safety Slave control.
"FIFO_PTP_RAM_CFG" on page 214	0x14	Dual Clock FIFO Controller with PTP RAM configuration.
INT INT Exists: Always		
"INT_ST_MAIN" on page 217	0x100	Interrupt status main.
"INT_ST_VPG" on page 219	0x104	VPG interrupt.
"INT_ST_IDI" on page 220	0x108	IDI interrupt.
"INT_ST_IPI" on page 222	0x10c	IPI interrupt.
"INT_ST_PHY" on page 226	0x110	PHY interrupt.
"INT_ST_IDI_VCX" on page 227	0x114	IDI interrupt for VCX.
"INT_ST_IDI_VCX2" on page 232	0x118	IDI interrupt for VCX2.
"INT_ST_MT_IPI" on page 238	0x11c	Multiple IPI interrupt.
"INT_ST_SDI" on page 239	0x120	SDI interrupt.
"INT_ST_SDI_VCX" on page 242	0x124	SDI interrupt for VCX.
"INT_ST_SDI_VCX2" on page 247	0x128	SDI interrupt for VCX2.
"INT_MASK_N_VPG" on page 253	0x160	VPG interrupt mask.
"INT_FORCE_VPG" on page 254	0x164	VPG interrupt force.
"INT_MASK_N_IDI" on page 255	0x168	IDI interrupt mask.
"INT_FORCE_IDI" on page 257	0x16c	IDI interrupt force.
"INT_MASK_N_IPI" on page 259	0x170	IPI interrupt mask.

Register	Offset	Description
"INT_FORCE_IPI" on page 262	0x174	IPI interrupt force.
"INT_MASK_N_PHY" on page 265	0x178	PHY interrupt mask.
"INT_FORCE_PHY" on page 266	0x17c	PHY interrupt force.
"INT_MASK_N_IDI_VCX" on page 267	0x180	IDI interrupt mask for VCX.
"INT_FORCE_IDI_VCX" on page 270	0x184	IDI interrupt force for VCX.
"INT_MASK_N_IDI_VCX2" on page 273	0x188	IDI interrupt mask for VCX2.
"INT_FORCE_IDI_VCX2" on page 276	0x18c	IDI interrupt force for VCX2.
"INT_MASK_N_MT_IPI" on page 279	0x190	Multiple IPI interrupt mask.
"INT_FORCE_MT_IPI" on page 280	0x194	Multiple IPI interrupt force.
"INT_MASK_N_SDI" on page 281	0x198	SDI interrupt mask.
"INT_FORCE_SDI" on page 283	0x19c	SDI interrupt force.
"INT_MASK_N_SDI_VCX" on page 285	0x1a0	SDI interrupt mask for VCX.
"INT_FORCE_SDI_VCX" on page 288	0x1a4	SDI interrupt force for VCX.
"INT_MASK_N_SDI_VCX2" on page 291	0x1a8	SDI interrupt mask for VCX2.
"INT_FORCE_SDI_VCX2" on page 294	0x1ac	SDI interrupt force for VCX2.
VPG VPG Exists: Always		
"VPG_CTRL" on page 298	0x200	VPG control.
"VPG_STATUS" on page 299	0x204	VPG status.
"VPG_MODE_CFG" on page 300	0x208	VPG mode configuration.
"VPG_PKT_CFG" on page 301	0x20c	VPG packet configuration.
"VPG_PKT_SIZE" on page 303	0x210	VPG packet size.
"VPG_HSA_TIME" on page 304	0x214	VPG Horizontal Synchronism Active time.
"VPG_HBP_TIME" on page 305	0x218	VPG Horizontal Back Porch time.
"VPG_HLINE_TIME" on page 306	0x21c	VPG each line time.
"VPG_VSA_LINES" on page 307	0x220	VPG Vertical Synchronism Active lines.
"VPG_VBP_LINES" on page 308	0x224	VPG Vertical Back Porch lines.
"VPG_VFP_LINES" on page 309	0x228	VPG Vertical Front Porch lines.
"VPG_ACT_LINES" on page 310	0x22c	VPG vertical resolution.

Register	Offset	Description
"VPG_MAX_FRAME_NUM" on page 311	0x230	VPG Maximun Frame Number.
"VPG_START_LINE_NUM" on page 312	0x234	VPG Start Line Number.
"VPG_STEP_LINE_NUM" on page 313	0x238	VPG Step Line Number.
"VPG_BK_LINES" on page 314	0x23c	VPG blanking lines.
PHY PHY Exists: Always		
"PHY_RSTZ" on page 316	0x300	D-PHY reset and PLL control.
"PHY_IF_CFG" on page 318	0x304	PHY active lanes number configuration.
"LPCLK_CTRL" on page 320	0x308	Non continuous clock control.
"PHY_ULPS_CTRL" on page 321	0x30c	PHY ULPS control.
"CLKMGR_CFG" on page 323	0x310	Lanebyteclk divide configuration.
"PHY_CAL" on page 324	0x314	D-PHY calibration.
"PHY_CAL_STATUS" on page 326	0x318	D-PHY calibration status.
"TO_CNT_CFG" on page 327	0x31c	Timeout counter configuration.
"PHY_MODE" on page 328	0x320	PHY mode.
"PHY_SWITCH_TIME" on page 329	0x324	PHY switch time.
"PHY_TIMING_CALC_CTRL" on page 330	0x328	PHY timing calculation control.
"PHY_TIMING_CALC_STATUS" on page 331	0x32c	PHY timing calculation status.
"PHY_SWITCH_TIME_AUTO" on page 332	0x330	Automatically calculated PHY switch time.
"PHY_STATUS" on page 333	0x334	PHY status.
"PHY0_TST_CTRL0" on page 338	0x338	D-PHY0 clock and clear pins control.
"PHY0_TST_CTRL1" on page 339	0x33c	D-PHY0 data and enable pins control.
"PHY1_TST_CTRL0" on page 341	0x340	D-PHY1 clock and clear pins control.
"PHY1_TST_CTRL1" on page 342	0x344	D-PHY1 data and enable pins control.
"TX_REG_CSI_EPD_EN_SSP" on page 344	0x348	EPD option control for short packet.
"TX_REG_CSI_EPD_OP_SLP" on page 345	0x34c	EPD option control for long packet.
"TX_REG_CSI_EPD_MISC_OPTIONS" on page 346	0x350	EPD misc options control.
"TX_REG_CSI_EPD_MAX" on page 348	0x354	EPD spacer maximum number configuration.

Register	Offset	Description
"TX_HS_IDLE_TIME" on page 349	0x358	HS_IDLE time.
"PHY_HS_IDLE_TIME_AUTO" on page 350	0x35c	Automatically calculated HS_IDLE time.
		IDI IDI Exists: Always
"IDI_FIFO_STATUS" on page 352	0x400	IDI fifo status.
		SDI SDI Exists: Always
"SDI_CFG" on page 355	0x500	SDI Control
		IPI IPI Exists: Always
"IPI_PKT_CFG" on page 357	0x600	IPI packet configuration.
"IPI_PIXELS" on page 360	0x604	IPI pixels.
"IPI_MAX_FRAME_NUM" on page 361	0x608	IPI max frame number.
"IPI_START_LINE_NUM" on page 362	0x60c	IPI start line number.
"IPI_STEP_LINE_NUM" on page 363	0x610	IPI step line number.
"IPI_LINES" on page 364	0x614	IPI lines.
"IPI_DATA_SEND_START" on page 365	0x618	IPI data send start.
"IPI_FIFO_STATUS" on page 366	0x61c	IPI fifo status.
"IPI_TRANS_STATUS" on page 368	0x640	IPI transmission status.
"IPI_HSA_HBP_PPI_TIME" on page 369	0x644	IPI HSA and HBP PPI time.
"IPI_HLINE_PPI_TIME" on page 370	0x648	IPI each line PPI time.
"IPI_VSA_LINES" on page 371	0x64c	IPI Vertical Synchronism Active lines.
"IPI_VBP_LINES" on page 372	0x650	IPI Vertical Back Porch lines.
"IPI_VFP_LINES" on page 373	0x654	IPI Vertical Front Porch lines.
"IPI_ACT_LINES" on page 374	0x658	IPI Vertical resolution lines.
"IPI_FB_LINES" on page 375	0x65c	IPI frame blanking lines.
"IPI_INSERT_CTRL" on page 376	0x660	IPI insertion control.
"IPI2_PKT_CFG" on page 377	0x680	IPI2 packet configuration.
"IPI2_PIXELS" on page 380	0x684	IPI2 pixels.

Register	Offset	Description
"IPI2_MAX_FRAME_NUM" on page 381	0x688	IPI2 max frame number.
"IPI2_START_LINE_NUM" on page 382	0x68c	IPI2 start line number
"IPI2_STEP_LINE_NUM" on page 383	0x690	IPI2 step line number
"IPI2_LINES" on page 384	0x694	IPI2 lines.
"IPI2_DATA_SEND_START" on page 385	0x698	IPI2 Data Send Start.
"IPI2_FIFO_STATUS" on page 386	0x69c	IPI2 fifo status.
"IPI3_PKT_CFG" on page 388	0x6c0	IPI3 packet configuration.
"IPI3_PIXELS" on page 391	0x6c4	IPI3 pixels.
"IPI3_MAX_FRAME_NUM" on page 392	0x6c8	IPI3 max frame number.
"IPI3_START_LINE_NUM" on page 393	0x6cc	IPI3 start line number.
"IPI3_STEP_LINE_NUM" on page 394	0x6d0	IPI3 Step Line Number.
"IPI3_LINES" on page 395	0x6d4	IPI3 lines.
"IPI3_DATA_SEND_START" on page 396	0x6d8	IPI3 data send start.
"IPI3_FIFO_STATUS" on page 397	0x6dc	IPI3 fifo status.
"IPI4_PKT_CFG" on page 399	0x700	IPI4 packet configuration.
"IPI4_PIXELS" on page 402	0x704	IPI4 pixels.
"IPI4_MAX_FRAME_NUM" on page 403	0x708	IPI4 max frame number.
"IPI4_START_LINE_NUM" on page 404	0x70c	IPI4 start line number.
"IPI4_STEP_LINE_NUM" on page 405	0x710	IPI4 Step Line Number.
"IPI4_LINES" on page 406	0x714	IPI4 lines.
"IPI4_DATA_SEND_START" on page 407	0x718	IPI4 data send start.
"IPI4_FIFO_STATUS" on page 408	0x71c	IPI4 fifo status.
"MT_IPI_CFG" on page 410	0x740	Mutiple IPI configuration.
"MT_IPI_DF_TIME" on page 411	0x744	Mutiple IPI divided field time.
"MT_IPI_FIFO_STATUS" on page 412	0x748	Multiple IPI fifo status.
"MT_IPI1_TRANS_CFG" on page 413	0x74c	Multiple IPI1 transmission configuration.
"MT_IPI2_TRANS_CFG" on page 415	0x750	Multiple IPI2 transmission configuration.
"MT_IPI3_TRANS_CFG" on page 417	0x754	Multiple IPI3 transmission configuration.
"MT_IPI4_TRANS_CFG" on page 419	0x758	Multiple IPI4 transmission configuration.

Register	Offset	Description
"IPI1_HSA_HBP_TIME" on page 421	0x75c	IPI1 HSA and HBP time.
"IPI1_LP_TIME" on page 422	0x760	IPI1 data packet transmission time.
"IPI2_HSA_HBP_TIME" on page 423	0x764	IPI2 HSA and HBP time.
"IPI2_LP_TIME" on page 424	0x768	IPI2 data packet transmission time.
"IPI3_HSA_HBP_TIME" on page 425	0x76c	IPI3 HSA and HBP time.
"IPI3_LP_TIME" on page 426	0x770	IPI3 data packet transmission time.
"IPI4_HSA_HBP_TIME" on page 427	0x774	IPI4 HSA and HBP time.
"IPI4_LP_TIME" on page 428	0x778	IPI4 data packet transmission time.
AP_INT AP_INT Exists: Always		
"INT_ST_DIAG_MAIN" on page 430	0x800	Interrupt status diagnosis main.
"INT_ST_DIAG0" on page 432	0x804	Interrupt status diagnosis group 0.
"INT_ST_FAP_VPG" on page 436	0x808	VPG functional AP interrupt.
"INT_ST_FAP_IDI" on page 437	0x80c	IDI functional AP interrupt.
"INT_ST_FAP_IPI" on page 439	0x810	IPI functional AP interrupt.
"INT_ST_FAP_PHY" on page 443	0x814	PHY functional AP interrupt.
"INT_ST_FAP_IDI_VCX" on page 444	0x818	IDI VCX functional AP interrupt.
"INT_ST_FAP_IDI_VCX2" on page 449	0x81c	IDI VCX2 functional AP interrupt.
"INT_ST_FAP_MT_IPI" on page 455	0x820	Multiple IPI functional AP interrupt.
"INT_ST_FAP_SDI" on page 456	0x824	SDI functional AP interrupt.
"INT_ST_FAP_SDI_VCX" on page 459	0x828	SDI VCX functional AP interrupt.
"INT_ST_FAP_SDI_VCX2" on page 464	0x82c	SDI VCX2 functional AP interrupt.
"INT_MASK_N_DIAG0" on page 470	0x860	Interrupt status diagnosis group 0 mask.
"INT_FORCE_DIAG0" on page 474	0x864	Interrupt status diagnosis group 0 force.
"INT_MASK_N_FAP_VPG" on page 478	0x868	INT_ST_FAP_VPG interrupt mask.
"INT_FORCE_FAP_VPG" on page 479	0x86c	INT_ST_FAP_VPG interrupt sources force.
"INT_MASK_N_FAP_IDI" on page 480	0x870	INT_ST_FAP_IDI Interrupt mask.
"INT_FORCE_FAP_IDI" on page 482	0x874	INT_ST_FAP_IDI interrupt sources force.
"INT_MASK_N_FAP_IPI" on page 484	0x878	INT_ST_FAP_IPI interrupt mask.

Register	Offset	Description
"INT_FORCE_FAP_IPI" on page 487	0x87c	INT_ST_FAP_IPI interrupt sources force.
"INT_MASK_N_FAP_PHY" on page 490	0x880	INT_ST_FAP_PHY interrupt mask.
"INT_FORCE_FAP_PHY" on page 491	0x884	INT_ST_FAP_PHY interrupt sources force.
"INT_MASK_N_FAP_IDI_VCX" on page 492	0x888	INT_ST_FAP_IDI_VCX interrupt mask.
"INT_FORCE_FAP_IDI_VCX" on page 495	0x88c	INT_ST_FAP_IDI_VCX interrupt sources force.
"INT_MASK_N_FAP_IDI_VCX2" on page 498	0x890	INT_ST_FAP_IDI_VCX2 interrupt mask.
"INT_FORCE_FAP_IDI_VCX2" on page 502	0x894	INT_ST_FAP_IDI_VCX2 interrupt sources force.
"INT_MASK_N_FAP_MT_IPI" on page 506	0x898	INT_ST_FAP_MT_IPI interrupt mask.
"INT_FORCE_FAP_MT_IPI" on page 507	0x89c	INT_ST_FAP_MT_IPI interrupt sources force.
"INT_MASK_N_FAP_SDI" on page 508	0x8a0	INT_ST_FAP_SDI Interrupt mask.
"INT_FORCE_FAP_SDI" on page 510	0x8a4	INT_ST_FAP_SDI interrupt sources force.
"INT_MASK_N_FAP_SDI_VCX" on page 512	0x8a8	INT_ST_FAP_SDI_VCX interrupt mask.
"INT_FORCE_FAP_SDI_VCX" on page 515	0x8ac	INT_ST_FAP_SDI_VCX interrupt sources force.
"INT_MASK_N_FAP_SDI_VCX2" on page 518	0x8b0	INT_ST_FAP_SDI_VCX2 interrupt mask.
"INT_FORCE_FAP_SDI_VCX2" on page 522	0x8b4	INT_ST_FAP_SDI_VCX2 interrupt sources force.
"IPI_AP_STATUS" on page 526	0x900	IPI AP check status.
"IDI_AP_STATUS" on page 527	0x904	IDI AP check status.
"AMBAAPPBINTF_AP_STATUS" on page 528	0x908	Amba apb interface AP check status.
"PHY_IF_CTRL_AP_STATUS" on page 529	0x90c	PHY interface control AP check status.
"REG_BANK_AP_STATUS" on page 531	0x910	Register bank AP check status.
"IPI_FIFOCTRL_AP_STATUS" on page 532	0x914	IPI FIFO control AP check status.
"IDI_FIFOCTRL_AP_STATUS" on page 534	0x918	IDI FIFO control AP check status.
"PKT_BUILDER_AP_STATUS" on page 536	0x91c	Packet builder AP check status.
"ERR_HANDLER_AP_STATUS" on page 537	0x920	Error handler AP check status.
"SYNC_AP_STATUS" on page 538	0x924	Synchronizer AP check status.
"PKT_IF_AP_STATUS" on page 539	0x928	Packet interface AP check status.
"ECF_AP_STATUS" on page 541	0x92c	Ecf AP check status.
"CMU_AP_STATUS" on page 542	0x930	Cmu AP check status.

Register	Offset	Description
"MT_IPI_CONTROL_AP_STATUS" on page 543	0x934	Multiple IPI Controller AP check status.
"IPI2_AP_STATUS" on page 545	0x938	IPI2 AP check status.
"IPI2_FIFOCTRL_AP_STATUS" on page 546	0x93c	IPI2 FIFO control AP check status.
"IPI3_AP_STATUS" on page 548	0x940	IPI3 AP check status.
"IPI3_FIFOCTRL_AP_STATUS" on page 549	0x944	IPI3 FIFO control AP check status.
"IPI4_AP_STATUS" on page 551	0x948	IPI4 AP check status.
"IPI4_FIFOCTRL_AP_STATUS" on page 552	0x94c	IPI4 FIFO control AP check status.
"SDI_AP_STATUS" on page 554	0x950	SDI AP check status.
"MASK_N_IPI_AP_STATUS" on page 555	0x9c0	IPI_AP_STATUS interrupt mask.
"FORCE_IPI_AP_STATUS" on page 556	0x9c4	IPI_AP_STATUS sources force.
"MASK_N_IDI_AP_STATUS" on page 557	0x9c8	IDI_AP_STATUS interrupt mask
"FORCE_IDI_AP_STATUS" on page 558	0x9cc	IDI_AP_STATUS sources force.
"MASK_N_AMBAAPBINTF_AP_STATUS" on page 559	0x9d0	AMBAAPBINTF_AP_STATUS interrupt mask.
"FORCE_AMBAAPBINTF_AP_STATUS" on page 561	0x9d4	AMBAAPBINTF_AP_STATUS sources FORCE.
"MASK_N_PHY_IF_CTRL_AP_STATUS" on page 562	0x9d8	PHY_IF_CTRL_AP_STATUS interrupt mask
"FORCE_PHY_IF_CTRL_AP_STATUS" on page 564	0x9dc	PHY_IF_CTRL_AP_STATUS sources force.
"MASK_N_REG_BANK_AP_STATUS" on page 566	0x9e0	REG_BANK_AP_STATUS interrupt mask.
"FORCE_REG_BANK_AP_STATUS" on page 567	0x9e4	REG_BANK_AP_STATUS sources force.
"MASK_N_IPI_FIFOCTRL_AP_STATUS" on page 568	0x9e8	IPI_FIFOCTRL_AP_STATUS interrupt mask.
"FORCE_IPI_FIFOCTRL_AP_STATUS" on page 570	0x9ec	IPI_FIFOCTRL_AP_STATUS sources force.
"MASK_N_IDI_FIFOCTRL_AP_STATUS" on page 572	0x9f0	IDI_FIFOCTRL_AP_STATUS interrupt mask.
"FORCE_IDI_FIFOCTRL_AP_STATUS" on page 574	0x9f4	IDI_FIFOCTRL_AP_STATUS sources force.

Register	Offset	Description
"MASK_N_PKT_BUILDER_AP_STATUS" on page 576	0x9f8	PKT_BUILDER_AP_STATUS interrupt mask.
"FORCE_PKT_BUILDER_AP_STATUS" on page 578	0x9fc	PKT_BUILDER_AP_STATUS sources force.
"MASK_N_ERR_HANDLER_AP_STATUS" on page 580	0xa00	ERR_HANDLER_AP_STATUS interrupt mask.
"FORCE_ERR_HANDLER_AP_STATUS" on page 581	0xa04	ERR_HANDLER_AP_STATUS sources force.
"MASK_N_SYNC_AP_STATUS" on page 582	0xa08	SYNC_AP_STATUS interrupt mask.
"FORCE_SYNC_AP_STATUS" on page 583	0xa0c	SYNC_AP_STATUS sources force.
"MASK_N_PKT_IF_AP_STATUS" on page 584	0xa10	PKT_IF_AP_STATUS interrupt mask.
"FORCE_PKT_IF_AP_STATUS" on page 586	0xa14	PKT_IF_AP_STATUS sources force.
"MASK_N_ECF_AP_STATUS" on page 588	0xa18	ECF_AP_STATUS interrupt mask.
"FORCE_ECF_AP_STATUS" on page 589	0xa1c	ECF_AP_STATUS sources force.
"MASK_N_CMU_AP_STATUS" on page 590	0xa20	CMU_AP_STATUS interrupt mask.
"FORCE_CMU_AP_STATUS" on page 591	0xa24	CMU_AP_STATUS sources force.
"MASK_N_MT_IPI_CONTROL_AP_STATUS" on page 592	0xa28	MT_IPI_CONTROL_AP_STATUS interrupt mask.
"FORCE_MT_IPI_CONTROL_AP_STATUS" on page 594	0xa2c	MT_IPI_CONTROL_AP_STATUS sources force.
"MASK_N_IPI2_AP_STATUS" on page 596	0xa30	IPI2_AP_STATUS interrupt mask.
"FORCE_IPI2_AP_STATUS" on page 597	0xa34	IPI2_AP_STATUS sources force.
"MASK_N_IPI2_FIFOCTRL_AP_STATUS" on page 598	0xa38	IPI2_FIFOCTRL_AP_STATUS interrupt mask.
"FORCE_IPI2_FIFOCTRL_AP_STATUS" on page 600	0xa3c	IPI2_FIFOCTRL_AP_STATUS sources force.
"MASK_N_IPI3_AP_STATUS" on page 602	0xa40	IPI3_AP_STATUS interrupt mask.
"FORCE_IPI3_AP_STATUS" on page 603	0xa44	IPI3_AP_STATUS sources force.
"MASK_N_IPI3_FIFOCTRL_AP_STATUS" on page 604	0xa48	IPI3_FIFOCTRL_AP_STATUS interrupt mask.
"FORCE_IPI3_FIFOCTRL_AP_STATUS" on page 606	0xa4c	IPI3_FIFOCTRL_AP_STATUS sources force.
"MASK_N_IPI4_AP_STATUS" on page 608	0xa50	IPI4_AP_STATUS interrupt mask.

Register	Offset	Description
"FORCE_IPI4_AP_STATUS" on page 609	0xa54	IPI4_AP_STATUS sources force.
"MASK_N_IPI4_FIFOCTRL_AP_STATUS" on page 610	0xa58	IPI4_FIFOCTRL_AP_STATUS interrupt mask.
"FORCE_IPI4_FIFOCTRL_AP_STATUS" on page 612	0xa5c	IPI4_FIFOCTRL_AP_STATUS sources force.
"MASK_N_SDI_AP_STATUS" on page 614	0xa60	SDI_AP_STATUS interrupt mask.
"FORCE_SDI_AP_STATUS" on page 615	0xa64	SDI_AP_STATUS sources force.
AP AP Exists: Always		
"ERR_INJ_CTRL" on page 617	0xb00	Error injection control.
"ERR_INJ_STATUS" on page 618	0xb04	Error injection status.
"ERR_INJ_CHK_MASK" on page 619	0xb08	Check error injection mask.
"ERR_INJ_D31_0_MASK" on page 620	0xb0c	Data 0-31 bit error injection mask.
"ERR_INJ_D63_32_MASK" on page 621	0xb10	Data 32-63bit error injection mask.
"ERR_INJ_D95_64_MASK" on page 622	0xb14	Data 64-95bit error injection mask.
"ERR_INJ_D127_96_MASK" on page 623	0xb18	Data 96-127bit error injection mask.
"ERR_INJ_D159_128_MASK" on page 624	0xb1c	Data 128-159bit error injection mask.
"ERR_INJ_D191_160_MASK" on page 625	0xb20	Data 160-191bit error injection mask.
"ERR_INJ_D223_192_MASK" on page 626	0xb24	Data 192-223bit error injection mask.
"ERR_INJ_D255_224_MASK" on page 627	0xb28	Data 224-255bit error injection mask.
"IDI_RAM_ERR_LOG_AP" on page 628	0xb2c	IDI logged error RAM addresses number.
"IDI_RAM_ERR_ADDR_AP" on page 629	0xb30	IDI first error RAM address.
"IPI_RAM_ERR_LOG_AP" on page 630	0xb34	IPI logged error RAM addresses number.
"IPI_RAM_ERR_ADDR_AP" on page 631	0xb38	IPI first error RAM address.
"IPI2_RAM_ERR_LOG_AP" on page 632	0xb3c	IPI2 logged error RAM addresses number.
"IPI2_RAM_ERR_ADDR_AP" on page 633	0xb40	IPI2 first error RAM address.
"IPI3_RAM_ERR_LOG_AP" on page 634	0xb44	IPI3 logged error RAM addresses number.
"IPI3_RAM_ERR_ADDR_AP" on page 635	0xb48	IPI3 first error RAM address.
"IPI4_RAM_ERR_LOG_AP" on page 636	0xb4c	IPI4 logged error RAM addresses number.
"IPI4_RAM_ERR_ADDR_AP" on page 637	0xb50	IPI4 first error RAM address.

5.2.1 DWC_mipicsi2_device_MemMap/MAIN Registers

5.2.1.1 VERSION

- **Name:** Controller Version identification.
- **Description:** Controller Version identification represented in ASCII.
- **Size:** 32 bits
- **Offset:** 0x0
- **Exists:** Always

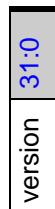


Table 5-6 Fields for Register: VERSION

Bits	Name	Memory Access	Description
31:0	version	R	<p>This field indicates the version of the DWC_mipicsi2_device.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (VERSION_version) <p>Value After Reset: CSI2_DEVICE_VERSION_ID</p> <p>Exists: Always</p>

5.2.1.2 CSI2_RESETN

- **Name:** Soft reset.
- **Description:** Controls the DWC_mipicsi2_device logic reset state. When activated, the internal logic of the controller goes into the reset state. The configuration is not reset to default values with this register, instead, only the internal controller logic is affected.
- **Size:** 32 bits
- **Offset:** 0x4
- **Exists:** Always



Table 5-7 Fields for Register: CSI2_RESETN

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	ss_swreset	R/W	<p>Safety Slave reset output. Active High. Can reset all the logic module of safety slave in core clock domain.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (CSI2_RESETN_ss_swreset) <p>Value After Reset: 0x1</p> <p>Exists: CSI2_DEVICE_SS==1</p>
0	csi2_resetn_rw	R/W	<p>CSI-2 device controller reset output. Active Low. Can reset all the logic module except APB.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (CSI2_RESETN_csi2_resetn_rw) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.1.3 DATA_SCRAMBLING

- **Name:** Data scrambling control.
- **Description:** DWC_mipicsi2_device data scrambling control.
- **Size:** 32 bits
- **Offset:** 0x8
- **Exists:** CSI2_DEVICE_DATA_SCRAMBLING==1

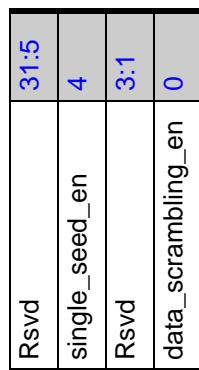


Table 5-8 Fields for Register: DATA_SCRAMBLING

Bits	Name	Memory Access	Description
31:5			Reserved Field: Yes
4	single_seed_en	R/W	<p>To control whether to use single seed mode with C-PHY.</p> <ul style="list-style-type: none"> ■ 1: Enable single seed mode. Only sync type 3 is transmitted. ■ 0: Disable single seed mode. Randomly choosing sync type 0,1,2,3 by internal PRBS generator <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (DATA_SCRAMBLING_single_seed_en) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_CPHY_TYPE==1</p>
3:1			Reserved Field: Yes
0	data_scrambling_en	R/W	<p>To enable CSI-2 device controller data scrambling, Active High.</p> <ul style="list-style-type: none"> ■ 1: Enable data scrambling. ■ 0: Disable data scrambling. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (DATA_SCRAMBLING_data_scrambling_en) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.1.4 VER_TYPE

- **Name:** Controller Version TYPE identification.
- **Description:** Controller Version TYPE identification represented in ASCII.
- **Size:** 32 bits
- **Offset:** 0xc
- **Exists:** Always



Table 5-9 Fields for Register: VER_TYPE

Bits	Name	Memory Access	Description
31:28	type_enum	R	<p>coreKit release type:</p> <ul style="list-style-type: none"> ■ 0x0: GA release ■ 0x1: LCA release ■ 0x2: EA release ■ 0x3: LP release ■ 0x4: Reserved ■ 0x5: SOW release ■ 0x6: EC release ■ Others: Reserved <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VER_TYPE_type_enum) <p>Value After Reset: CSI2_DEVICE_VER_TYPE_ENUM</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
27:24	pkg_num	R	<p>Alphabetical packaging number of the coreKit ID:</p> <ul style="list-style-type: none"> ■ 4'b1010: 'a' ■ 4'b1011: 'b' ■ 4'b1100: 'c' ■ 4'b1101: 'd' ■ 4'b1110: 'e' ■ 4'b1111: 'f' ■ 4'b0000: 'g' ■ ... ■ 4'b1000: 'o' ■ 4'b1001: 'p' <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VER_TYPE_pkg_num) <p>Value After Reset: CSI2_DEVICE_VER_TYPE_PKG</p> <p>Exists: Always</p>
23:16	type_num	R	<p>coreKit release version in BCD format. For example, a release number 1.70a-lp04 contains 0x4.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VER_TYPE_type_num) <p>Value After Reset: CSI2_DEVICE_VER_TYPE_NUM</p> <p>Exists: Always</p>
15:0	ver_number	R	<p>IP version number. Nibble-wise representation of BCD values containing the coreKit release version number. For example, 0x0100 means 1.00.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VER_TYPE_ver_number) <p>Value After Reset: CSI2_DEVICE_VER_NUMBER</p> <p>Exists: Always</p>

5.2.1.5 SS_CTRL

- **Name:** Safety Slave control.
- **Description:** DWC_mipicsi2_device Safety Slave control.
- **Size:** 32 bits
- **Offset:** 0x10
- **Exists:** CSI2_DEVICE_SS==1

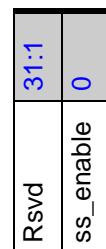


Table 5-10 Fields for Register: SS_CTRL

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	ss_enable	R/W	<p>Enable core monitoring by SS. This bit is used to enable monitoring of safety errors by Safety Slave. The monitoring can be disabled, while 2 or more cores monitored by Safety Manager and application want to disable monitoring of some cores.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (SS_CTRL_ss_enable) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.1.6 FIFO_PTP_RAM_CFG

- Name:** Dual Clock FIFO Controller with PTP RAM configuration.
- Description:** When using the Dual Clock FIFO Controller with PTP RAM(BCM79), push_dom_faster fields need to be programmed. And reset release needs to wait at least 2 cycles of slower clock domain after any change on push_dom_faster. For more details, refer to the description of RESET_RELEASED_AFTER_PDF_CHANGED_MET assertion in BCM79 databook.
- Size:** 32 bits
- Offset:** 0x14
- Exists:** CSI2_DEVICE_USE_SPRAM==1

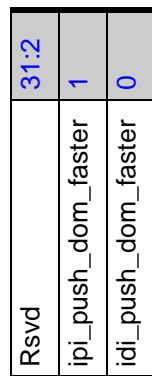


Table 5-11 Fields for Register: FIFO_PTP_RAM_CFG

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	ipi_push_dom_faster	R/W	<p>Selects which clock domain is faster for IPI Dual Clock FIFO Controller with PTP RAM(BCM79), impacts the location of the underlying FIFO controller (BCM78) as follows: 0: Pop domain is faster and places DWbb_bcm78 in the lanebyteclk domain. 1: Push domain is faster and places DWbb_bcm78 in the ipi clock domain</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FIFO_PTP_RAM_CFG_ipi_push_dom_faster) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>

Bits	Name	Memory Access	Description
0	idi_push_dom_faster	R/W	<p>Selects which clock domain is faster for IDI Dual Clock FIFO Controller with PTP RAM(BCM79), impacts the location of the underlying FIFO controller (BCM78) as follows: 0: Pop domain is faster and places DWbb_bcm78 in the lanebyteclk domain. 1: Push domain is faster and places DWbb_bcm78 in the idi clock domain</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FIFO_PTP_RAM_CFG_idi_push_dom_faster) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>

5.2.2 DWC_mipicsi2_device_MemMap/INT Registers

5.2.2.1 INT_ST_MAIN

- **Name:** Interrupt status main.
- **Description:** Clear on read register. Contains the status of individual interrupt sources, regardless of the contents of the associated interrupt mask registers, so it is possible to service the interrupt status registers by polling. Reading this register clears it and deasserts the interrupt pin.
- **Size:** 32 bits
- **Offset:** 0x100
- **Exists:** Always

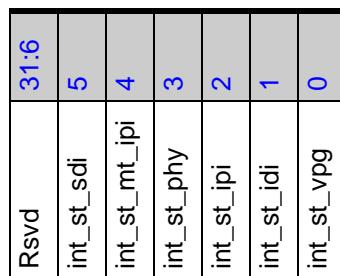


Table 5-12 Fields for Register: INT_ST_MAIN

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	int_st_sdi	RC	<p>Status of INT_ST_SDI interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_MAIN_int_st_sdi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_SDIF==1</p>
4	int_st_mt_ipi	RC	<p>Status of INT_ST_MT_IPI interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_MAIN_int_st_mt_ipi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>

Bits	Name	Memory Access	Description
3	int_st_phy	RC	<p>Status of INT_ST_PHY interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_MAIN_int_st_phy) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	int_st_ipi	RC	<p>Status of INT_ST_IPI interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_MAIN_int_st_ipi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>
1	int_st_idi	RC	<p>Status of INT_ST_IDI interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_MAIN_int_st_idi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>
0	int_st_vpg	RC	<p>Status of INT_ST_VPG interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_MAIN_int_st_vpg) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_VPG==1</p>

5.2.2.2 INT_ST_VPG

- **Name:** VPG interrupt.
- **Description:** Interrupt group caused by the Video pattern generator.
- **Size:** 32 bits
- **Offset:** 0x104
- **Exists:** CSI2_DEVICE_VPG==1

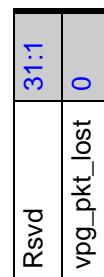


Table 5-13 Fields for Register: INT_ST_VPG

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	vpg_pkt_lost	RC	<p>Packet lost of video pattern generator.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_VPG_vpg_pkt_lost) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.3 INT_ST_IDI

- **Name:** IDI interrupt.
- **Description:** Interrupt group caused by the IDI interface. Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x108
- **Exists:** CSI2_DEVICE_IDI_IF==1

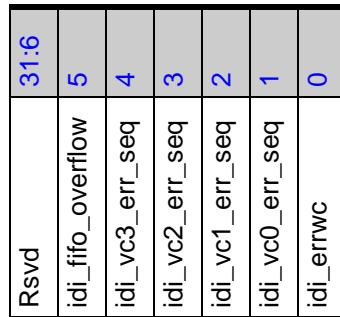


Table 5-14 Fields for Register: INT_ST_IDI

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	idi_fifo_overflow	RC	<p>The IDI Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_idi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	idi_vc3_err_seq	RC	<p>Indicator for sequence error in virtual channel 3:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_idi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
3	idi_vc2_err_seq	RC	<p>Indicator for sequence error in virtual channel 2:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_idi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	idi_vc1_err_seq	RC	<p>Indicator for sequence error in virtual channel 1:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_idi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	idi_vc0_err_seq	RC	<p>Indicator for sequence error in virtual channel 0:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_idi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	idi_errwc	RC	<p>Indicator for inconsistent between IDI Word Count and packet payload length.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_idi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.4 INT_ST_IPI

- **Name:** IPI interrupt.
 - **Description:** Interrupt group caused by the IPI interface. Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
 - **Size:** 32 bits
 - **Offset:** 0x10c
 - **Exists:** CSI2_DEVICE_IPI_IF==1

Rsvd	31:28
ipi4_fifo_underflow	27
ipi4_errline	26
ipi4_fifo_overflow	25
ipi4_errpixel	24
Rsvd	23:20
ipi3_fifo_underflow	19
ipi3_errline	18
ipi3_fifo_overflow	17
ipi3_errpixel	16
Rsvd	15:12
ipi2_fifo_underflow	11
ipi2_errline	10
ipi2_fifo_overflow	9
ipi2_errpixel	8
Rsvd	7:5
ipi_trans_conflict	4
ipi_fifo_underflow	3
ipi_errline	2
ipi_fifo_overflow	1
ipi_errpixel	0

Table 5-15 Fields for Register: INT_ST_IPI

Bits	Name	Memory Access	Description
31:28			Reserved Field: Yes
27	ipi4_fifo_underflow	RC	<p>The IPI4 Payload FIFO has lost information because read data when it was already empty. This is used for Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi4_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
26	ipi4_errline	RC	<p>Indicator for inconsistent between IPI4 line configuration and the number of input lines.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi4_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
25	ipi4_fifo_overflow	RC	<p>The IPI4 Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi4_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>

Bits	Name	Memory Access	Description
24	ipi4_errpixel	RC	<p>Indicator for inconsistent between IPI4 Pixel configuration and the number of input pixels or inconsistency between ipi4_embedded_wc and the embedded data received.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi4_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
23:20			Reserved Field: Yes
19	ipi3_fifo_underflow	RC	<p>The IPI3 Payload FIFO has lost information because read data when it was already empty. This is used for Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi3_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
18	ipi3_errline	RC	<p>Indicator for inconsistent between IPI3 line configuration and the number of input lines.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi3_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
17	ipi3_fifo_overflow	RC	<p>The IPI3 Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi3_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	ipi3_errpixel	RC	<p>Indicator for inconsistent between IPI3 Pixel configuration and the number of input pixels or inconsistency between ipi3_embedded_wc and the embedded data received.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi3_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15:12			Reserved Field: Yes

Bits	Name	Memory Access	Description
11	ipi2_fifo_underflow	RC	<p>The IPI2 Payload FIFO has lost information because read data when it was already empty. This is used for Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi2_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
10	ipi2_errline	RC	<p>Indicator for inconsistent between IPI2 line configuration and the number of input lines.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi2_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
9	ipi2_fifo_overflow	RC	<p>The IPI2 Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi2_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
8	ipi2_errpixel	RC	<p>Indicator for inconsistent between IPI2 Pixel configuration and the number of input pixels or inconsistency between ipi2_embedded_wc and the embedded data received.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi2_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
7:5			Reserved Field: Yes
4	ipi_trans_conflict	RC	<p>This bit indicates that IPI packet transmission is in conflict with IDI interface.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi_trans_conflict) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IPI_IF==1</p>
3	ipi_fifo_underflow	RC	<p>The IPI Payload FIFO has lost information because read data when it was already empty. This is used for Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	ipi_errline	RC	<p>Indicator for inconsistent between IPI line configuration and the number of input lines</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi_errline) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	ipi_fifo_overflow	RC	<p>The IPI Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	ipi_errpixel	RC	<p>Indicator for inconsistent between IPI Pixel configuration and the number of input pixels or inconsistency between ipi_embedded_wc and the embedded data received</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IPI_ipi_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.5 INT_ST_PHY

- **Name:** PHY interrupt.
- **Description:** Interrupt group caused by the PHY.Groups and notifies which interrupt bits caused the interruption.Stores the source of the error.Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x110
- **Exists:** Always

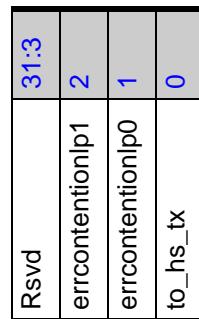


Table 5-16 Fields for Register: INT_ST_PHY

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	errcontentionlp1	RC	<p>This bit indicates LP1 contention error ErrContentionLP1 from Lane 0.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_PHY_errcontentionlp1) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	errcontentionlp0	RC	<p>This bit indicates LP0 contention error ErrContentionLP0 from Lane 0.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_PHY_errcontentionlp0) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	to_hs_tx	RC	<p>This bit indicates that the high-speed transmission timeout counter reached the end and contention has been detected.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_PHY_to_hs_tx) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.6 INT_ST_IDI_VCX

- **Name:** IDI interrupt for VCX.
- **Description:** Interrupt group caused by the IDI interface for virtual channel extension. (vc4~vc15)Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x114
- **Exists:** CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

Rsvd	31:12	idi_vc15_err_seq	11	idi_vc14_err_seq	10	idi_vc13_err_seq	9	idi_vc12_err_seq	8	idi_vc11_err_seq	7	idi_vc10_err_seq	6	idi_vc9_err_seq	5	idi_vc8_err_seq	4	idi_vc7_err_seq	3	idi_vc6_err_seq	2	idi_vc5_err_seq	1	idi_vc4_err_seq	0
------	-------	------------------	----	------------------	----	------------------	---	------------------	---	------------------	---	------------------	---	-----------------	---	-----------------	---	-----------------	---	-----------------	---	-----------------	---	-----------------	---

Table 5-17 Fields for Register: INT_ST_IDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	idi_vc15_err_seq	RC	<p>Indicator for sequence error in virtual channel 15:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc15_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
10	idi_vc14_err_seq	RC	<p>Indicator for sequence error in virtual channel 14:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc14_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	idi_vc13_err_seq	RC	<p>Indicator for sequence error in virtual channel 13:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc13_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	idi_vc12_err_seq	RC	<p>Indicator for sequence error in virtual channel 12:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7	idi_vc11_err_seq	RC	<p>Indicator for sequence error in virtual channel 11:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	idi_vc10_err_seq	RC	<p>Indicator for sequence error in virtual channel 10:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	idi_vc9_err_seq	RC	<p>Indicator for sequence error in virtual channel 9:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
4	idi_vc8_err_seq	RC	<p>Indicator for sequence error in virtual channel 8:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	idi_vc7_err_seq	RC	<p>Indicator for sequence error in virtual channel 7:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	idi_vc6_err_seq	RC	<p>Indicator for sequence error in virtual channel 6:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	idi_vc5_err_seq	RC	<p>Indicator for sequence error in virtual channel 5:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	idi_vc4_err_seq	RC	<p>Indicator for sequence error in virtual channel 4:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX_idi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.7 INT_ST_IDI_VCX2

- **Name:** IDI interrupt for VCX2.
- **Description:** Interrupt group caused by the IDI interface for virtual channel extension. (vc16~vc31)Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x118
- **Exists:** CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

31:16	Rsvd	idi_vc31_err_seq	15	idi_vc30_err_seq	14	idi_vc29_err_seq	13	idi_vc28_err_seq	12	idi_vc27_err_seq	11	idi_vc26_err_seq	10	idi_vc25_err_seq	9	idi_vc24_err_seq	8	idi_vc23_err_seq	7	idi_vc22_err_seq	6	idi_vc21_err_seq	5	idi_vc20_err_seq	4	idi_vc19_err_seq	3	idi_vc18_err_seq	2	idi_vc17_err_seq	1	idi_vc16_err_seq	0
-------	------	------------------	----	------------------	----	------------------	----	------------------	----	------------------	----	------------------	----	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---

Table 5-18 Fields for Register: INT_ST_IDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	idi_vc31_err_seq	RC	<p>Indicator for sequence error in virtual channel 31:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc31_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
14	idi_vc30_err_seq	RC	<p>Indicator for sequence error in virtual channel 30:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc30_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
13	idi_vc29_err_seq	RC	<p>Indicator for sequence error in virtual channel 29:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc29_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
12	idi_vc28_err_seq	RC	<p>Indicator for sequence error in virtual channel 28:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
11	idi_vc27_err_seq	RC	<p>Indicator for sequence error in virtual channel 27:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	idi_vc26_err_seq	RC	<p>Indicator for sequence error in virtual channel 26:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	idi_vc25_err_seq	RC	<p>Indicator for sequence error in virtual channel 25:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
8	idi_vc24_err_seq	RC	<p>Indicator for sequence error in virtual channel 24:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	idi_vc23_err_seq	RC	<p>Indicator for sequence error in virtual channel 23:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	idi_vc22_err_seq	RC	<p>Indicator for sequence error in virtual channel 22:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	idi_vc21_err_seq	RC	<p>Indicator for sequence error in virtual channel 21:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	idi_vc20_err_seq	RC	<p>Indicator for sequence error in virtual channel 20:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	idi_vc19_err_seq	RC	<p>Indicator for sequence error in virtual channel 19:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	idi_vc18_err_seq	RC	<p>Indicator for sequence error in virtual channel 18:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	idi_vc17_err_seq	RC	<p>Indicator for sequence error in virtual channel 17:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	idi_vc16_err_seq	RC	<p>Indicator for sequence error in virtual channel 16:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_IDI_VCX2_idi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.8 INT_ST_MT_IPI

- **Name:** Multiple IPI interrupt.
- **Description:** Interrupt group caused by the Multiple IPI transmission.
- **Size:** 32 bits
- **Offset:** 0x11c
- **Exists:** CSI2_DEVICE_IPI2_IF==1



Table 5-19 Fields for Register: INT_ST_MT_IPI

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	mt_ipi_fifo_overflow	RC	<p>The MT IPI Header FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_MT_IPI_mt_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.9 INT_ST_SDI

- **Name:** SDI interrupt.
- **Description:** Interrupt group caused by the SDI interface. Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x120
- **Exists:** CSI2_DEVICE_SDI_IF==1

Rsvd	31:7
sdi_pld_err	6
sdi_hdr_err	5
sdi_vc3_err_seq	4
sdi_vc2_err_seq	3
sdi_vc1_err_seq	2
sdi_vc0_err_seq	1
sdi_errwc	0

Table 5-20 Fields for Register: INT_ST_SDI

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	sdi_pld_err	RC	<p>Indicator for sdi payload error in PF_CRC Bypass mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDI_sdi_pld_err) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	sdi_hdr_err	RC	<p>Indicator for sdi header error in ECC/PH-CRC Bypass mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDI_sdi_hdr_err) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
4	sdi_vc3_err_seq	RC	<p>Indicator for sequence error in virtual channel 3:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SD1_sdi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	sdi_vc2_err_seq	RC	<p>Indicator for sequence error in virtual channel 2:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SD1_sdi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	sdi_vc1_err_seq	RC	<p>Indicator for sequence error in virtual channel 1:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SD1_sdi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	sdi_vc0_err_seq	RC	<p>Indicator for sequence error in virtual channel 0:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDI_sdi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	sdi_errwc	RC	<p>Indicator for inconsistent between SDI Word Count and packet payload length.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDI_sdi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.10 INT_ST_SDIVCX

- **Name:** SDI interrupt for VCX.
- **Description:** Interrupt group caused by the SDI interface for virtual channel extension. (vc4~vc15)Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x124
- **Exists:** CSI2_DEVICE_SDIF==1&&CSI2_DEVICE_VC_EXTENSION==1

31:12	Rsvd	sdi_vc15_err_seq	11
		sdi_vc14_err_seq	10
		sdi_vc13_err_seq	9
		sdi_vc12_err_seq	8
		sdi_vc11_err_seq	7
		sdi_vc10_err_seq	6
		sdi_vc9_err_seq	5
		sdi_vc8_err_seq	4
		sdi_vc7_err_seq	3
		sdi_vc6_err_seq	2
		sdi_vc5_err_seq	1
		sdi_vc4_err_seq	0

Table 5-21 Fields for Register: INT_ST_SDIVCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	sdi_vc15_err_seq	RC	<p>Indicator for sequence error in virtual channel 15:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIVCX_sdi_vc15_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
10	sdi_vc14_err_seq	RC	<p>Indicator for sequence error in virtual channel 14:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc14_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	sdi_vc13_err_seq	RC	<p>Indicator for sequence error in virtual channel 13:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc13_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	sdi_vc12_err_seq	RC	<p>Indicator for sequence error in virtual channel 12:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7	sdi_vc11_err_seq	RC	<p>Indicator for sequence error in virtual channel 11:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	sdi_vc10_err_seq	RC	<p>Indicator for sequence error in virtual channel 10:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	sdi_vc9_err_seq	RC	<p>Indicator for sequence error in virtual channel 9:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
4	sdi_vc8_err_seq	RC	<p>Indicator for sequence error in virtual channel 8:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	sdi_vc7_err_seq	RC	<p>Indicator for sequence error in virtual channel 7:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	sdi_vc6_err_seq	RC	<p>Indicator for sequence error in virtual channel 6:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	sdi_vc5_err_seq	RC	<p>Indicator for sequence error in virtual channel 5:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	sdi_vc4_err_seq	RC	<p>Indicator for sequence error in virtual channel 4:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX_VCX_sdi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.11 INT_ST_SDI_VCX2

- **Name:** SDI interrupt for VCX2.
- **Description:** Interrupt group caused by the SDI interface for virtual channel extension. (vc16~vc31)Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x128
- **Exists:** CSI2_DEVICE_SDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_C-PHY_TYPE==1

Rsvd	31:16
sdi_vc31_err_seq	15
sdi_vc30_err_seq	14
sdi_vc29_err_seq	13
sdi_vc28_err_seq	12
sdi_vc27_err_seq	11
sdi_vc26_err_seq	10
sdi_vc25_err_seq	9
sdi_vc24_err_seq	8
sdi_vc23_err_seq	7
sdi_vc22_err_seq	6
sdi_vc21_err_seq	5
sdi_vc20_err_seq	4
sdi_vc19_err_seq	3
sdi_vc18_err_seq	2
sdi_vc17_err_seq	1
sdi_vc16_err_seq	0

Table 5-22 Fields for Register: INT_ST_SDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	sdi_vc31_err_seq	RC	<p>Indicator for sequence error in virtual channel 31:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDI_VCX2_sdi_vc31_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
14	sdi_vc30_err_seq	RC	<p>Indicator for sequence error in virtual channel 30:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc30_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
13	sdi_vc29_err_seq	RC	<p>Indicator for sequence error in virtual channel 29:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc29_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
12	sdi_vc28_err_seq	RC	<p>Indicator for sequence error in virtual channel 28:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
11	sdi_vc27_err_seq	RC	<p>Indicator for sequence error in virtual channel 27:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	sdi_vc26_err_seq	RC	<p>Indicator for sequence error in virtual channel 26:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	sdi_vc25_err_seq	RC	<p>Indicator for sequence error in virtual channel 25:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
8	sdi_vc24_err_seq	RC	<p>Indicator for sequence error in virtual channel 24:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_sdi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	sdi_vc23_err_seq	RC	<p>Indicator for sequence error in virtual channel 23:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_sdi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	sdi_vc22_err_seq	RC	<p>Indicator for sequence error in virtual channel 22:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_sdi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	sdi_vc21_err_seq	RC	<p>Indicator for sequence error in virtual channel 21:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	sdi_vc20_err_seq	RC	<p>Indicator for sequence error in virtual channel 20:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	sdi_vc19_err_seq	RC	<p>Indicator for sequence error in virtual channel 19:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_SDI_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	sdi_vc18_err_seq	RC	<p>Indicator for sequence error in virtual channel 18:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_sdi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	sdi_vc17_err_seq	RC	<p>Indicator for sequence error in virtual channel 17:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_sdi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	sdi_vc16_err_seq	RC	<p>Indicator for sequence error in virtual channel 16:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_SDIX2_sdi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.12 INT_MASK_N_VPG

- **Name:** VPG interrupt mask.
- **Description:** Interrupt mask for INT_ST_VPG; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source. 0: Interrupt source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x160
- **Exists:** CSI2_DEVICE_VPG==1



Table 5-23 Fields for Register: INT_MASK_N_VPG

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	mask_vpg_pkt_lost	R/W	<p>Mask for vpg_pkt_lost.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_VPG_mask_vpg_pkt_lost) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.13 INT_FORCE_VPG

- **Name:** VPG interrupt force.
- **Description:** Test register to force activation of INT_ST_VPG interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x164
- **Exists:** CSI2_DEVICE_VPG==1



Table 5-24 Fields for Register: INT_FORCE_VPG

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	force_vpg_pkt_lost	R/WC	<p>Force for vpg_pkt_lost.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_VPG_force_vpg_pkt_lost) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.14 INT_MASK_N_IDI

- **Name:** IDI interrupt mask.
 - **Description:** Interrupt mask for INT_ST_IDI; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked. Static read and write register.
 - **Size:** 32 bits
 - **Offset:** 0x168
 - **Exists:** CSI2_DEVICE_IDI_IF==1

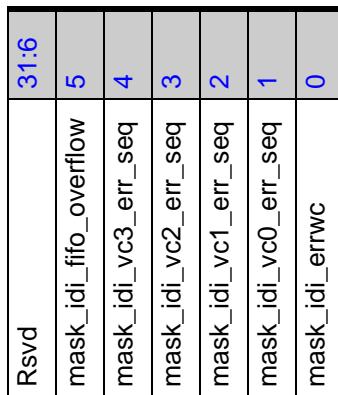


Table 5-25 Fields for Register: INT_MASK_N_IDI

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	mask_idi_fifo_overflow	R/W	Mask for idi_fifo_overflow. Values:
			<ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_mask_idi_fifo_overflow) Value After Reset: 0x0
			Exists: Always
4	mask_idi_vc3_err_seq	R/W	Mask for idi_vc3_err_seq. Values:
			<ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_mask_idi_vc3_err_seq) Value After Reset: 0x0
			Exists: Always
3	mask_idi_vc2_err_seq	R/W	Mask for idi_vc2_err_seq. Values:
			<ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_mask_idi_vc2_err_seq) Value After Reset: 0x0
			Exists: Always

Bits	Name	Memory Access	Description
2	mask_idi_vc1_err_seq	R/W	<p>Mask for idi_vc1_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_mask_idi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_idi_vc0_err_seq	R/W	<p>Mask for idi_vc0_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_mask_idi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_idi_errwc	R/W	<p>Mask for idi_errwc.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_mask_idi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.15 INT_FORCE_IDI

- **Name:** IDI interrupt force.
- **Description:** Test register to force activation of INT_ST_IDI interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x16c
- **Exists:** CSI2_DEVICE_IDI_IF==1

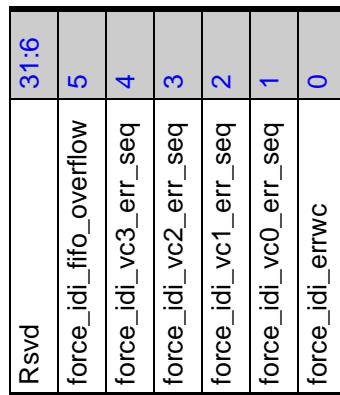


Table 5-26 Fields for Register: INT_FORCE_IDI

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	force_idi_fifo_overflow	R/WC	<p>Force for idi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_force_idi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_idi_vc3_err_seq	R/WC	<p>Force for idi_vc3_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_force_idi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_idi_vc2_err_seq	R/WC	<p>Force for idi_vc2_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_force_idi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	force_idi_vc1_err_seq	R/WC	<p>Force for idi_vc1_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_force_idi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_idi_vc0_err_seq	R/WC	<p>Force for idi_vc0_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_force_idi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_idi_errwc	R/WC	<p>Force for idi_errwc.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_force_idi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.16 INT_MASK_N_IPI

- **Name:** IPI interrupt mask.
 - **Description:** Interrupt mask for INT_ST_IPI; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked. Static read and write register.
 - **Size:** 32 bits
 - **Offset:** 0x170
 - **Exists:** CSI2_DEVICE_IPI_IF==1

Rsvd		31:28
mask_ipi4_fifo_underflow	27	26
mask_ipi4_errline	25	25
mask_ipi4_overflow	24	24
mask_ipi4_errpixel		23:20
Rsvd		
mask_ipi3_fifo_underflow	19	19
mask_ipi3_errline	18	18
mask_ipi3_fifo_overflow	17	17
mask_ipi3_errpixel		16
Rsvd		15:12
mask_ipi2_fifo_underflow	11	11
mask_ipi2_errline	10	10
mask_ipi2_fifo_overflow	9	9
mask_ipi2_errpixel	8	8
Rsvd		7:5
mask_ipi_trans_conflict		4
mask_ipi_fifo_underflow	3	3
mask_ipi_errline		2
mask_ipi_fifo_overflow	1	1
mask_ipi_errpixel		0

Table 5-27 Fields for Register: INT_MASK_N_IPI

Bits	Name	Memory Access	Description
31:28			Reserved Field: Yes
27	mask_ipi4_fifo_underflow	R/W	Mask for ipi4_fifo_underflow Values: <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi4_fifo_underflow) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1
26	mask_ipi4_errline	R/W	Mask for ipi4_errline Values: <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi4_errline) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1
25	mask_ipi4_fifo_overflow	R/W	Mask for ipi4_fifo_overflow Values: <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi4_fifo_overflow) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1

Bits	Name	Memory Access	Description
24	mask_ipi4_errpixel	R/W	<p>Mask for ipi4_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi4_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
23:20			Reserved Field: Yes
19	mask_ipi3_fifo_underflow	R/W	<p>Mask for ipi3_fifo_underflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi3_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
18	mask_ipi3_errline	R/W	<p>Mask for ipi3_errline</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi3_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
17	mask_ipi3_fifo_overflow	R/W	<p>Mask for ipi3_fifo_overflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi3_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	mask_ipi3_errpixel	R/W	<p>Mask for ipi3_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi3_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15:12			Reserved Field: Yes
11	mask_ipi2_fifo_underflow	R/W	<p>Mask for ipi2_fifo_underflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi2_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
10	mask_ipi2_errline	R/W	<p>Mask for ipi2_errline</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi2_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>

Bits	Name	Memory Access	Description
9	mask_ipi2_fifo_overflow	R/W	<p>Mask for ipi2_fifo_overflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi2_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
8	mask_ipi2_errpixel	R/W	<p>Mask for ipi2_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi2_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
7:5			Reserved Field: Yes
4	mask_ipi_trans_conflict	R/W	<p>Mask for ipi_trans_conflict.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi_trans_conflict) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IPI_IF==1</p>
3	mask_ipi_fifo_underflow	R/W	<p>Mask for ipi_fifo_underflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_ipi_errline	R/W	<p>Mask for ipi_errline.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi_errline) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_ipi_fifo_overflow	R/W	<p>Mask for ipi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_ipi_errpixel	R/W	<p>Mask for ipi_errpixel.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IPI_mask_ipi_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.17 INT_FORCE_IPI

- **Name:** IPI interrupt force.
- **Description:** Test register to force activation of INT_ST_IPI interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x174
- **Exists:** CSI2_DEVICE_IPI_IF==1

Rsvd	31:28
force_ipi4_fifo_underflow	27
force_ipi4_errline	26
force_ipi4_fifo_overflow	25
force_ipi4_errpixel	24
Rsvd	23:20
force_ipi3_fifo_underflow	19
force_ipi3_errline	18
force_ipi3_fifo_overflow	17
force_ipi3_errpixel	16
Rsvd	15:12
force_ipi2_fifo_underflow	11
force_ipi2_errline	10
force_ipi2_fifo_overflow	9
force_ipi2_errpixel	8
Rsvd	7:5
force_ipi_trans_conflict	4
force_ipi_fifo_underflow	3
force_ipi_errline	2
force_ipi_fifo_overflow	1
force_ipi_errpixel	0

Table 5-28 Fields for Register: INT_FORCE_IPI

Bits	Name	Memory Access	Description
31:28			Reserved Field: Yes
27	force_ipi4_fifo_underflow	R/WC	Force for ipi4_fifo_underflow Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_IPI_force_ipi4_fifo_underflow) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1
26	force_ipi4_errline	R/WC	Force for ipi4_errline Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_IPI_force_ipi4_errline) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1
25	force_ipi4_fifo_overflow	R/WC	Force for ipi4_fifo_overflow Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_IPI_force_ipi4_fifo_overflow) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1

Bits	Name	Memory Access	Description
24	force_ipi4_errpixel	R/WC	<p>Force for ipi4_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi4_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
23:20			Reserved Field: Yes
19	force_ipi3_fifo_underflow	R/WC	<p>Force for ipi3_fifo_underflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi3_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
18	force_ipi3_errline	R/WC	<p>Force for ipi3_errline</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi3_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
17	force_ipi3_fifo_overflow	R/WC	<p>Force for ipi3_fifo_overflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi3_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	force_ipi3_errpixel	R/WC	<p>Force for ipi3_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi3_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15:12			Reserved Field: Yes
11	force_ipi2_fifo_underflow	R/WC	<p>Force for ipi2_fifo_underflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi2_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
10	force_ipi2_errline	R/WC	<p>Force for ipi2_errline</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi2_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>

Bits	Name	Memory Access	Description
9	force_ipi2_fifo_overflow	R/WC	<p>Force for ipi2_fifo_overflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi2_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
8	force_ipi2_errpixel	R/WC	<p>Force for ipi2_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi2_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
7:5			Reserved Field: Yes
4	force_ipi_trans_conflict	R/WC	<p>Force for ipi_trans_conflict.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi_trans_conflict) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IPI_IF==1</p>
3	force_ipi_fifo_underflow	R/WC	<p>Force for ipi_fifo_underflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_ipi_errline	R/WC	<p>Force for ipi_errline.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi_errline) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_ipi_fifo_overflow	R/WC	<p>Force for ipi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_ipi_errpixel	R/WC	<p>Force for ipi_errpixel.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IPI_force_ipi_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.18 INT_MASK_N_PHY

- **Name:** PHY interrupt mask.
- **Description:** Interrupt mask for INT_ST_PHY; controls which interrupt status bit triggers the interrupt pin.
1: Enable the interrupt source.
0: Interrupt source is masked.
- **Size:** 32 bits
- **Offset:** 0x178
- **Exists:** Always

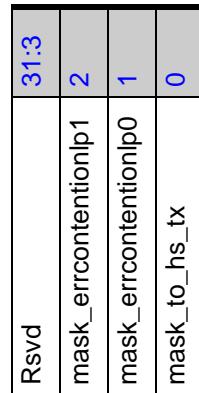


Table 5-29 Fields for Register: INT_MASK_N_PHY

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_errcontentionlp1	R/W	<p>Mask for errcontentionlp1</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_PHY_mask_errcontentionlp1) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_errcontentionlp0	R/W	<p>Mask for errcontentionlp0</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_PHY_mask_errcontentionlp0) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_to_hs_tx	R/W	<p>Mask for to_hs_tx.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_PHY_mask_to_hs_tx) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.19 INT_FORCE_PHY

- **Name:** PHY interrupt force.
- **Description:** Test register to force activation of INT_ST_PHY interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x17c
- **Exists:** Always

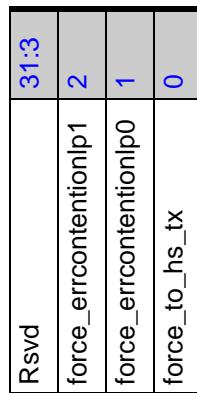


Table 5-30 Fields for Register: INT_FORCE_PHY

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_errcontentionlp1	R/WC	Force for errcontentionlp1 Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_PHY_force_errcontentionlp1) Value After Reset: 0x0 Exists: Always
1	force_errcontentionlp0	R/WC	Force for errcontentionlp0 Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_PHY_force_errcontentionlp0) Value After Reset: 0x0 Exists: Always
0	force_to_hs_tx	R/WC	Force for to_hs_tx. Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_PHY_force_to_hs_tx) Value After Reset: 0x0 Exists: Always

5.2.2.20 INT_MASK_N_IDI_VCX

- Name:** IDI interrupt mask for VCX.
- Description:** Interrupt mask for INT_ST_IDI_VCX; controls which interrupt status bit triggers the interrupt pin. 1: Enable the interrupt source.0: Interrupt source is masked. Static read and write register.
- Size:** 32 bits
- Offset:** 0x180
- Exists:** CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

31:12	Rsvd	mask_idi_vc15_err_seq	11
	mask_idi_vc14_err_seq	10	
	mask_idi_vc13_err_seq	9	
	mask_idi_vc12_err_seq	8	
	mask_idi_vc11_err_seq	7	
	mask_idi_vc10_err_seq	6	
	mask_idi_vc9_err_seq	5	
	mask_idi_vc8_err_seq	4	
	mask_idi_vc7_err_seq	3	
	mask_idi_vc6_err_seq	2	
	mask_idi_vc5_err_seq	1	
	mask_idi_vc4_err_seq	0	

Table 5-31 Fields for Register: INT_MASK_N_IDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	mask_idi_vc15_err_seq	R/W	<p>Mask for idi_vc15_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc15_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
10	mask_idi_vc14_err_seq	R/W	<p>Mask for idi_vc14_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc14_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
9	mask_idi_vc13_err_seq	R/W	<p>Mask for idi_vc13_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc13_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
8	mask_idi_vc12_err_seq	R/W	<p>Mask for idi_vc12_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	mask_idi_vc11_err_seq	R/W	<p>Mask for idi_vc11_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_idi_vc10_err_seq	R/W	<p>Mask for idi_vc10_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_idi_vc9_err_seq	R/W	<p>Mask for idi_vc9_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_idi_vc8_err_seq	R/W	<p>Mask for idi_vc8_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	mask_idi_vc7_err_seq	R/W	<p>Mask for idi_vc7_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_idi_vc6_err_seq	R/W	<p>Mask for idi_vc6_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	mask_idi_vc5_err_seq	R/W	<p>Mask for idi_vc5_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_idi_vc4_err_seq	R/W	<p>Mask for idi_vc4_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX_mask_idi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.21 INT_FORCE_IDI_VCX

- **Name:** IDI interrupt force for VCX.
- **Description:** Test register to force activation of INT_ST_IDI_VCX interrupt sources.1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x184
- **Exists:** CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

Rsvd	31:12	force_idi_vc15_err_seq	11	force_idi_vc14_err_seq	10	force_idi_vc13_err_seq	9	force_idi_vc12_err_seq	8	force_idi_vc11_err_seq	7	force_idi_vc10_err_seq	6	force_idi_vc9_err_seq	5	force_idi_vc8_err_seq	4	force_idi_vc7_err_seq	3	force_idi_vc6_err_seq	2	force_idi_vc5_err_seq	1	force_idi_vc4_err_seq	0
------	-------	------------------------	----	------------------------	----	------------------------	---	------------------------	---	------------------------	---	------------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---

Table 5-32 Fields for Register: INT_FORCE_IDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	force_idi_vc15_err_seq	R/WC	<p>Force for idi_vc15_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc15_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
10	force_idi_vc14_err_seq	R/WC	<p>Force for idi_vc14_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc14_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
9	force_idi_vc13_err_seq	R/WC	<p>Force for idi_vc13_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc13_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
8	force_idi_vc12_err_seq	R/WC	<p>Force for idi_vc12_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	force_idi_vc11_err_seq	R/WC	<p>Force for idi_vc11_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_idi_vc10_err_seq	R/WC	<p>Force for idi_vc10_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_idi_vc9_err_seq	R/WC	<p>Force for idi_vc9_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_idi_vc8_err_seq	R/WC	<p>Force for idi_vc8_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_idi_vc7_err_seq	R/WC	<p>Force for idi_vc7_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_idi_vc6_err_seq	R/WC	<p>Force for idi_vc6_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	force_idi_vc5_err_seq	R/WC	<p>Force for idi_vc5_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_idi_vc4_err_seq	R/WC	<p>Force for idi_vc4_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (INT_FORCE_IDI_VCX_force_idi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.22 INT_MASK_N_IDI_VCX2

- Name:** IDI interrupt mask for VCX2.
- Description:** Interrupt mask for INT_ST_IDI_VCX2; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
- Size:** 32 bits
- Offset:** 0x188
- Exists:** CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

31:16	Rsvd	mask_idi_vc31_err_seq	15
	mask_idi_vc30_err_seq	14	
	mask_idi_vc29_err_seq	13	
	mask_idi_vc28_err_seq	12	
	mask_idi_vc27_err_seq	11	
	mask_idi_vc26_err_seq	10	
	mask_idi_vc25_err_seq	9	
	mask_idi_vc24_err_seq	8	
	mask_idi_vc23_err_seq	7	
	mask_idi_vc22_err_seq	6	
	mask_idi_vc21_err_seq	5	
	mask_idi_vc20_err_seq	4	
	mask_idi_vc19_err_seq	3	
	mask_idi_vc18_err_seq	2	
	mask_idi_vc17_err_seq	1	
	mask_idi_vc16_err_seq	0	

Table 5-33 Fields for Register: INT_MASK_N_IDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	mask_idi_vc31_err_seq	R/W	<p>Mask for idi_vc31_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc31_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
14	mask_idi_vc30_err_seq	R/W	<p>Mask for idi_vc30_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc30_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
13	mask_idi_vc29_err_seq	R/W	<p>Mask for idi_vc29_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc29_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
12	mask_idi_vc28_err_seq	R/W	<p>Mask for idi_vc28_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	mask_idi_vc27_err_seq	R/W	<p>Mask for idi_vc27_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	mask_idi_vc26_err_seq	R/W	<p>Mask for idi_vc26_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	mask_idi_vc25_err_seq	R/W	<p>Mask for idi_vc25_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	mask_idi_vc24_err_seq	R/W	<p>Mask for idi_vc24_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	mask_idi_vc23_err_seq	R/W	<p>Mask for idi_vc23_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_idi_vc22_err_seq	R/W	<p>Mask for idi_vc22_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	mask_idi_vc21_err_seq	R/W	<p>Mask for idi_vc21_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_idi_vc20_err_seq	R/W	<p>Mask for idi_vc20_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	mask_idi_vc19_err_seq	R/W	<p>Mask for idi_vc19_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_idi_vc18_err_seq	R/W	<p>Mask for idi_vc18_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_idi_vc17_err_seq	R/W	<p>Mask for idi_vc17_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_idi_vc16_err_seq	R/W	<p>Mask for idi_vc16_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_IDI_VCX2_mask_idi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.23 INT_FORCE_IDI_VCX2

- **Name:** IDI interrupt force for VCX2.
 - **Description:** Test register to force activation of INT_ST_IDI_VCX2 interrupt sources.1: Force internal interrupt source.0: No action. Writing this register clears it.
 - **Size:** 32 bits
 - **Offset:** 0x18c
 - **Exists:** CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

Rsvd		31:16
force_idi_vc31_err_seq	15	
force_idi_vc30_err_seq	14	
force_idi_vc29_err_seq	13	
force_idi_vc28_err_seq	12	
force_idi_vc27_err_seq	11	
force_idi_vc26_err_seq	10	
force_idi_vc25_err_seq	9	
force_idi_vc24_err_seq	8	
force_idi_vc23_err_seq	7	
force_idi_vc22_err_seq	6	
force_idi_vc21_err_seq	5	
force_idi_vc20_err_seq	4	
force_idi_vc19_err_seq	3	
force_idi_vc18_err_seq	2	
force_idi_vc17_err_seq	1	
force_idi_vc16_err_seq	0	

Table 5-34 Fields for Register: INT_FORCE_IDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	force_idi_vc31_err_seq	R/WC	Force for idi_vc31_err_seq.
			Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc31_err_seq)
			Value After Reset: 0x0
			Exists: Always
14	force_idi_vc30_err_seq	R/WC	Force for idi_vc30_err_seq.
			Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc30_err_seq)
			Value After Reset: 0x0
			Exists: Always
13	force_idi_vc29_err_seq	R/WC	Force for idi_vc29_err_seq.
			Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc29_err_seq)
			Value After Reset: 0x0
			Exists: Always

Bits	Name	Memory Access	Description
12	force_idi_vc28_err_seq	R/WC	<p>Force for idi_vc28_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	force_idi_vc27_err_seq	R/WC	<p>Force for idi_vc27_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	force_idi_vc26_err_seq	R/WC	<p>Force for idi_vc26_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	force_idi_vc25_err_seq	R/WC	<p>Force for idi_vc25_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	force_idi_vc24_err_seq	R/WC	<p>Force for idi_vc24_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	force_idi_vc23_err_seq	R/WC	<p>Force for idi_vc23_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_idi_vc22_err_seq	R/WC	<p>Force for idi_vc22_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	force_idi_vc21_err_seq	R/WC	<p>Force for idi_vc21_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_idi_vc20_err_seq	R/WC	<p>Force for idi_vc20_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_idi_vc19_err_seq	R/WC	<p>Force for idi_vc19_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_idi_vc18_err_seq	R/WC	<p>Force for idi_vc18_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_idi_vc17_err_seq	R/WC	<p>Force for idi_vc17_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_idi_vc16_err_seq	R/WC	<p>Force for idi_vc16_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_IDI_VCX2_force_idi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.24 INT_MASK_N_MT_IPI

- **Name:** Multiple IPI interrupt mask.
- **Description:** Interrupt mask for INT_ST_MT_IPI; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x190
- **Exists:** CSI2_DEVICE_IPI2_IF==1



Table 5-35 Fields for Register: INT_MASK_N_MT_IPI

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	mask_mt_ipi_fifo_overflow	R/W	<p>Mask for mt_ipi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_MT_IPI_mask_mt_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.25 INT_FORCE_MT_IPI

- **Name:** Multiple IPI interrupt force.
- **Description:** Test register to force activation of INT_ST_MT_IPI interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x194
- **Exists:** CSI2_DEVICE_IPI2_IF==1

	31:1	
Rsvd		0
	force_mt_ipi_fifo_overflow	

Table 5-36 Fields for Register: INT_FORCE_MT_IPI

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	force_mt_ipi_fifo_overflow	R/WC	<p>Force for mt_ipi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_MT_IPI_force_mt_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.26 INT MASK N SDI

- **Name:** SDI interrupt mask.
 - **Description:** Interrupt mask for INT_ST_SDI; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
 - **Size:** 32 bits
 - **Offset:** 0x198
 - **Exists:** CSI2_DEVICE SDI_IF==1

Rsvd	31:7
mask_sdi_pld_err	6
mask_sdi_hdr_err	5
mask_sdi_vc3_err_seq	4
mask_sdi_vc2_err_seq	3
mask_sdi_vc1_err_seq	2
mask_sdi_vc0_err_seq	1
mask_sdi_errwc	0

Table 5-37 Fields for Register: INT MASK N SDI

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	mask_sdi_pld_err	R/W	<p>Mask for sdi_pld_err.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1I_mask_sdi_pld_err) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_sdi_hdr_err	R/W	<p>Mask for sdi_hdr_err.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1I_mask_sdi_hdr_err) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_sdi_vc3_err_seq	R/W	<p>Mask for sdi_vc3_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1I_mask_sdi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
3	mask_sdi_vc2_err_seq	R/W	<p>Mask for sdi_vc2_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_mask_sdi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_sdi_vc1_err_seq	R/W	<p>Mask for sdi_vc1_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_mask_sdi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_sdi_vc0_err_seq	R/W	<p>Mask for sdi_vc0_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_mask_sdi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_sdi_errwc	R/W	<p>Mask for sdi_errwc.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_mask_sdi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.27 INT_FORCE_SDI

- **Name:** SDI interrupt force.
- **Description:** Test register to force activation of INT_ST_SDI interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x19c
- **Exists:** CSI2_DEVICE_SDI_IF==1

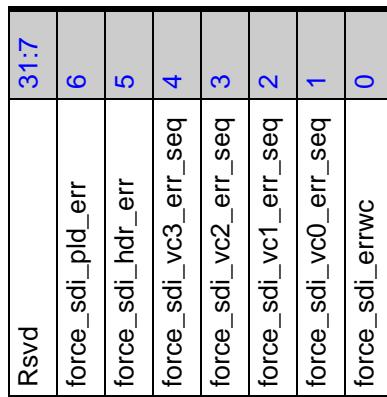


Table 5-38 Fields for Register: INT_FORCE_SDI

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	force_sdi_pld_err	R/WC	<p>Force for sdi_pld_err.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_force_sdi_pld_err) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_sdi_hdr_err	R/WC	<p>Force for sdi_hdr_err.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_force_sdi_hdr_err) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_sdi_vc3_err_seq	R/WC	<p>Force for sdi_vc3_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_force_sdi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
3	force_sdi_vc2_err_seq	R/WC	<p>Force for sdi_vc2_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_force_sdi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_sdi_vc1_err_seq	R/WC	<p>Force for sdi_vc1_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_force_sdi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_sdi_vc0_err_seq	R/WC	<p>Force for sdi_vc0_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_force_sdi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_sdi_errwc	R/WC	<p>Force for sdi_errwc.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_force_sdi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.28 INT_MASK_N_SD1_VCX

- **Name:** SDI interrupt mask for VCX.
- **Description:** Interrupt mask for INT_ST_SD1_VCX; controls which interrupt status bit triggers the interrupt pin. 1: Enable the interrupt source.0: Interrupt source is masked. Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x1a0
- **Exists:** CSI2_DEVICE_SD1_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

31:12	Rsvd	mask_sd1_vc15_err_seq	11	mask_sd1_vc14_err_seq	10	mask_sd1_vc13_err_seq	9	mask_sd1_vc12_err_seq	8	mask_sd1_vc11_err_seq	7	mask_sd1_vc10_err_seq	6	mask_sd1_vc9_err_seq	5	mask_sd1_vc8_err_seq	4	mask_sd1_vc7_err_seq	3	mask_sd1_vc6_err_seq	2	mask_sd1_vc5_err_seq	1	mask_sd1_vc4_err_seq	0
-------	------	-----------------------	----	-----------------------	----	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	----------------------	---	----------------------	---	----------------------	---	----------------------	---	----------------------	---	----------------------	---

Table 5-39 Fields for Register: INT_MASK_N_SD1_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	mask_sd1_vc15_err_seq	R/W	<p>Mask for sd1_vc15_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sd1_vc15_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
10	mask_sd1_vc14_err_seq	R/W	<p>Mask for sd1_vc14_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sd1_vc14_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
9	mask_sd1_vc13_err_seq	R/W	<p>Mask for sd1_vc13_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sd1_vc13_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
8	mask_sdi_vc12_err_seq	R/W	<p>Mask for sdi_vc12_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	mask_sdi_vc11_err_seq	R/W	<p>Mask for sdi_vc11_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_sdi_vc10_err_seq	R/W	<p>Mask for sdi_vc10_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_sdi_vc9_err_seq	R/W	<p>Mask for sdi_vc9_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_sdi_vc8_err_seq	R/W	<p>Mask for sdi_vc8_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	mask_sdi_vc7_err_seq	R/W	<p>Mask for sdi_vc7_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_sdi_vc6_err_seq	R/W	<p>Mask for sdi_vc6_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	mask_sdi_vc5_err_seq	R/W	<p>Mask for sdi_vc5_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_sdi_vc4_err_seq	R/W	<p>Mask for sdi_vc4_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX_mask_sdi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.29 INT_FORCE_SDI_VCX

- **Name:** SDI interrupt force for VCX.
- **Description:** Test register to force activation of INT_ST_SDI_VCX interrupt sources.1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x1a4
- **Exists:** CSI2_DEVICE_SDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

31:12	Rsvd	force_sdi_vc15_err_seq	11	force_sdi_vc14_err_seq	10	force_sdi_vc13_err_seq	9	force_sdi_vc12_err_seq	8	force_sdi_vc11_err_seq	7	force_sdi_vc10_err_seq	6	force_sdi_vc9_err_seq	5	force_sdi_vc8_err_seq	4	force_sdi_vc7_err_seq	3	force_sdi_vc6_err_seq	2	force_sdi_vc5_err_seq	1	force_sdi_vc4_err_seq	0
-------	------	------------------------	----	------------------------	----	------------------------	---	------------------------	---	------------------------	---	------------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---

Table 5-40 Fields for Register: INT_FORCE_SDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	force_sdi_vc15_err_seq	R/WC	<p>Force for sdi_vc15_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc15_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
10	force_sdi_vc14_err_seq	R/WC	<p>Force for sdi_vc14_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc14_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
9	force_sdi_vc13_err_seq	R/WC	<p>Force for sdi_vc13_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc13_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
8	force_sdi_vc12_err_seq	R/WC	<p>Force for sdi_vc12_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	force_sdi_vc11_err_seq	R/WC	<p>Force for sdi_vc11_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_sdi_vc10_err_seq	R/WC	<p>Force for sdi_vc10_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_sdi_vc9_err_seq	R/WC	<p>Force for sdi_vc9_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_sdi_vc8_err_seq	R/WC	<p>Force for sdi_vc8_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_sdi_vc7_err_seq	R/WC	<p>Force for sdi_vc7_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_sdi_vc6_err_seq	R/WC	<p>Force for sdi_vc6_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	force_sdi_vc5_err_seq	R/WC	<p>Force for sdi_vc5_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_sdi_vc4_err_seq	R/WC	<p>Force for sdi_vc4_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (INT_FORCE_SDI_VCX_force_sdi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.30 INT_MASK_N_SD1_VCX2

- **Name:** SDI interrupt mask for VCX2.
- **Description:** Interrupt mask for INT_ST_SD1_VCX2; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x1a8
- **Exists:** CSI2_DEVICE_SD1_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

	31:16
Rsvd	
mask_sd1_vc31_err_seq	15
mask_sd1_vc30_err_seq	14
mask_sd1_vc29_err_seq	13
mask_sd1_vc28_err_seq	12
mask_sd1_vc27_err_seq	11
mask_sd1_vc26_err_seq	10
mask_sd1_vc25_err_seq	9
mask_sd1_vc24_err_seq	8
mask_sd1_vc23_err_seq	7
mask_sd1_vc22_err_seq	6
mask_sd1_vc21_err_seq	5
mask_sd1_vc20_err_seq	4
mask_sd1_vc19_err_seq	3
mask_sd1_vc18_err_seq	2
mask_sd1_vc17_err_seq	1
mask_sd1_vc16_err_seq	0

Table 5-41 Fields for Register: INT_MASK_N_SD1_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	mask_sd1_vc31_err_seq	R/W	<p>Mask for sd1_vc31_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sd1_vc31_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
14	mask_sd1_vc30_err_seq	R/W	<p>Mask for sd1_vc30_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sd1_vc30_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
13	mask_sd1_vc29_err_seq	R/W	<p>Mask for sd1_vc29_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sd1_vc29_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
12	mask_sdi_vc28_err_seq	R/W	<p>Mask for sdi_vc28_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	mask_sdi_vc27_err_seq	R/W	<p>Mask for sdi_vc27_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	mask_sdi_vc26_err_seq	R/W	<p>Mask for sdi_vc26_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	mask_sdi_vc25_err_seq	R/W	<p>Mask for sdi_vc25_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	mask_sdi_vc24_err_seq	R/W	<p>Mask for sdi_vc24_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	mask_sdi_vc23_err_seq	R/W	<p>Mask for sdi_vc23_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_sdi_vc22_err_seq	R/W	<p>Mask for sdi_vc22_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	mask_sdi_vc21_err_seq	R/W	<p>Mask for sdi_vc21_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_sdi_vc20_err_seq	R/W	<p>Mask for sdi_vc20_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	mask_sdi_vc19_err_seq	R/W	<p>Mask for sdi_vc19_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_sdi_vc18_err_seq	R/W	<p>Mask for sdi_vc18_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_sdi_vc17_err_seq	R/W	<p>Mask for sdi_vc17_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_sdi_vc16_err_seq	R/W	<p>Mask for sdi_vc16_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_SD1_VCX2_mask_sdi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.2.31 INT_FORCE_SDI_VCX2

- **Name:** SDI interrupt force for VCX2.
- **Description:** Test register to force activation of INT_ST_SDI_VCX2 interrupt sources.1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x1ac
- **Exists:** CSI2_DEVICE_SDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_C-PHY_TYPE==1

	31:16
Rsvd	
force_sdi_vc31_err_seq	15
force_sdi_vc30_err_seq	14
force_sdi_vc29_err_seq	13
force_sdi_vc28_err_seq	12
force_sdi_vc27_err_seq	11
force_sdi_vc26_err_seq	10
force_sdi_vc25_err_seq	9
force_sdi_vc24_err_seq	8
force_sdi_vc23_err_seq	7
force_sdi_vc22_err_seq	6
force_sdi_vc21_err_seq	5
force_sdi_vc20_err_seq	4
force_sdi_vc19_err_seq	3
force_sdi_vc18_err_seq	2
force_sdi_vc17_err_seq	1
force_sdi_vc16_err_seq	0

Table 5-42 Fields for Register: INT_FORCE_SDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	force_sdi_vc31_err_seq	R/WC	Force for sdi_vc31_err_seq. Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc31_err_seq) Value After Reset: 0x0 Exists: Always
14	force_sdi_vc30_err_seq	R/WC	Force for sdi_vc30_err_seq. Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc30_err_seq) Value After Reset: 0x0 Exists: Always
13	force_sdi_vc29_err_seq	R/WC	Force for sdi_vc29_err_seq. Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc29_err_seq) Value After Reset: 0x0 Exists: Always

Bits	Name	Memory Access	Description
12	force_sdi_vc28_err_seq	R/WC	<p>Force for sdi_vc28_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	force_sdi_vc27_err_seq	R/WC	<p>Force for sdi_vc27_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	force_sdi_vc26_err_seq	R/WC	<p>Force for sdi_vc26_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	force_sdi_vc25_err_seq	R/WC	<p>Force for sdi_vc25_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	force_sdi_vc24_err_seq	R/WC	<p>Force for sdi_vc24_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	force_sdi_vc23_err_seq	R/WC	<p>Force for sdi_vc23_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_sdi_vc22_err_seq	R/WC	<p>Force for sdi_vc22_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	force_sdi_vc21_err_seq	R/WC	<p>Force for sdi_vc21_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_sdi_vc20_err_seq	R/WC	<p>Force for sdi_vc20_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_sdi_vc19_err_seq	R/WC	<p>Force for sdi_vc19_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_sdi_vc18_err_seq	R/WC	<p>Force for sdi_vc18_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_sdi_vc17_err_seq	R/WC	<p>Force for sdi_vc17_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_sdi_vc16_err_seq	R/WC	<p>Force for sdi_vc16_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_SDI_VCX2_force_sdi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3 DWC_mipicsi2_device_MemMap/VPG Registers

5.2.3.1 VPG_CTRL

- **Name:** VPG control.
- **Description:** This register contains the enable signal of video pattern generator.
- **Size:** 32 bits
- **Offset:** 0x200
- **Exists:** CSI2_DEVICE_VPG==1

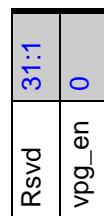


Table 5-43 Fields for Register: VPG_CTRL

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	vpg_en	R/W	<p>Video pattern generator enable signal. Active High.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_CTRL_vpg_en) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.2 VPG_STATUS

- **Name:** VPG status.
- **Description:** This register contains the status of video pattern generator.
- **Size:** 32 bits
- **Offset:** 0x204
- **Exists:** CSI2_DEVICE_VPG==1

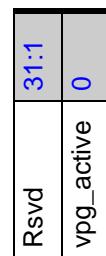


Table 5-44 Fields for Register: VPG_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	vpg_active	R	<p>video pattern status:</p> <ul style="list-style-type: none"> ■ 1: Video pattern generation is running and vpg_en should not be set to 1 again. ■ 0: One video pattern transmission finished and the next transmission can be started. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_STATUS_vpg_active) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.3 VPG_MODE_CFG

- **Name:** VPG mode configuration.
- **Description:** Test Mode Configuration of video pattern generator
- **Size:** 32 bits
- **Offset:** 0x208
- **Exists:** CSI2_DEVICE_VPG==1

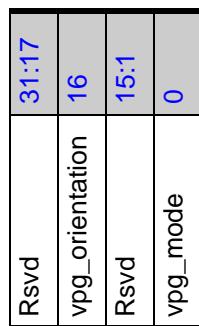


Table 5-45 Fields for Register: VPG_MODE_CFG

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16	vpg_orientation	R/W	<p>This field indicates the color bar orientation:</p> <ul style="list-style-type: none"> ■ 1: Horizontal Mode ■ 0: Vertical Mode <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_MODE_CFG_vpg_orientation) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15:1			Reserved Field: Yes
0	vpg_mode	R/W	<p>Pattern Type:</p> <ul style="list-style-type: none"> ■ 1: BER pattern, vertical only. ■ 0: Color bar, vertical or horizontal. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_MODE_CFG_vpg_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.4 VPG_PKT_CFG

- **Name:** VPG packet configuration.
- **Description:** This register indicates the packet configuration of video pattern generator.
- **Size:** 32 bits
- **Offset:** 0x20c
- **Exists:** CSI2_DEVICE_VPG==1

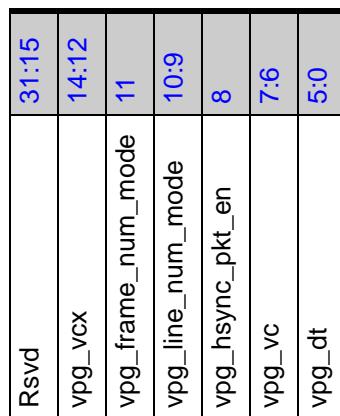


Table 5-46 Fields for Register: VPG_PKT_CFG

Bits	Name	Memory Access	Description
31:15			Reserved Field: Yes
14:12	vpg_vcx	R/W	<p>The Virtual Channel bit[4:2] of video pattern packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_PKT_CFG_vpg_vcx) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_VC_EXTENSION==1</p>
11	vpg_frame_num_mode	R/W	<p>This field indicates the frame number mode:</p> <ul style="list-style-type: none"> ■ 1: Frame Number Increments One mode. ■ 0: Frame Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_PKT_CFG_vpg_frame_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
10:9	vpg_line_num_mode	R/W	<p>This field indicates line number mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Line Number Increments Arbitrary Value mode. ■ 01: Line Number Increments One mode. ■ 00: Line Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_PKT_CFG_vpg_line_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	vpg_hsync_pkt_en	R/W	<p>This field indicates the line synchronization packets mode:</p> <ul style="list-style-type: none"> ■ 1: Transmit line synchronization packets. ■ 0: Don't transmit line synchronization packets. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_PKT_CFG_vpg_hsync_pkt_en) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7:6	vpg_vc	R/W	<p>The Virtual Channel bit[1:0] of video pattern packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_PKT_CFG_vpg_vc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5:0	vpg_dt	R/W	<p>The Data Type of video pattern packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_PKT_CFG_vpg_dt) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.5 VPG_PKT_SIZE

- **Name:** VPG packet size.
- **Description:** Test packet size.
- **Size:** 32 bits
- **Offset:** 0x210
- **Exists:** CSI2_DEVICE_VPG==1

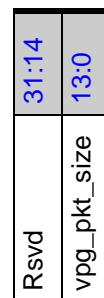


Table 5-47 Fields for Register: VPG_PKT_SIZE

Bits	Name	Memory Access	Description
31:14			Reserved Field: Yes
13:0	vpg_pkt_size	R/W	<p>The number of pixels in a single video pattern packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_PKT_SIZE_vpg_pkt_size) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.6 VPG_HSA_TIME

- **Name:** VPG Horizontal Synchronism Active time.
- **Description:** Horizontal Synchronism Active(HSA) time.
- **Size:** 32 bits
- **Offset:** 0x214
- **Exists:** CSI2_DEVICE_VPG==1

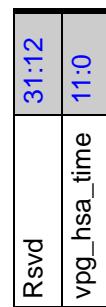


Table 5-48 Fields for Register: VPG_HSA_TIME

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11:0	vpg_hsa_time	R/W	<p>The configuration of horizontal synchronism active period in lane byte clock domain. The smallest value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_HSA_TIME_vpg_hsa_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.7 VPG_HBP_TIME

- **Name:** VPG Horizontal Back Porch time.
- **Description:** Horizontal Back Porch(HBP) time.
- **Size:** 32 bits
- **Offset:** 0x218
- **Exists:** CSI2_DEVICE_VPG==1

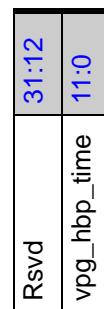


Table 5-49 Fields for Register: VPG_HBP_TIME

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11:0	vpg_hbp_time	R/W	<p>The configuration of horizontal back porch period in lane byte clock domain. The smallest value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_HBP_TIME_vpg_hbp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.8 VPG_HLINE_TIME

- **Name:** VPG each line time.
- **Description:** Overall time for each video line.
- **Size:** 32 bits
- **Offset:** 0x21c
- **Exists:** CSI2_DEVICE_VPG==1

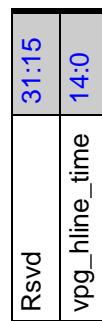


Table 5-50 Fields for Register: VPG_HLINE_TIME

Bits	Name	Memory Access	Description
31:15			Reserved Field: Yes
14:0	vpg_hline_time	R/W	<p>The size of total line time(HSA+HBP+HACT+HFP) counted in lane byte clock domain.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_HLINE_TIME_vpg_hline_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.9 VPG_VSA_LINES

- **Name:** VPG Vertical Synchronism Active lines.
- **Description:** Vertical Synchronism Active(VSA) period.
- **Size:** 32 bits
- **Offset:** 0x220
- **Exists:** CSI2_DEVICE_VPG==1

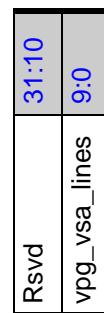


Table 5-51 Fields for Register: VPG_VSA_LINES

Bits	Name	Memory Access	Description
31:10			Reserved Field: Yes
9:0	vpg_vsa_lines	R/W	<p>The vertical synchronism active period measured in number of horizontal lines. The smallest value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_VSA_LINES_vpg_vsa_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.10 VPG_VBP_LINES

- **Name:** VPG Vertical Back Porch lines.
- **Description:** Vertical Back Porch(VBP) period.
- **Size:** 32 bits
- **Offset:** 0x224
- **Exists:** CSI2_DEVICE_VPG==1

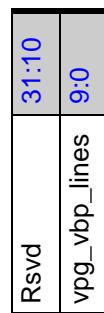


Table 5-52 Fields for Register: VPG_VBP_LINES

Bits	Name	Memory Access	Description
31:10			Reserved Field: Yes
9:0	vpg_vbp_lines	R/W	<p>The vertical back porch period measured in number of horizontal lines. The smallest value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_VBP_LINES_vpg_vbp_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.11 VPG_VFP_LINES

- **Name:** VPG Vertical Front Porch lines.
- **Description:** Vertical Front Porch(VFP) period.
- **Size:** 32 bits
- **Offset:** 0x228
- **Exists:** CSI2_DEVICE_VPG==1

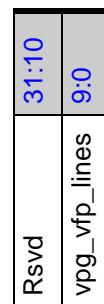


Table 5-53 Fields for Register: VPG_VFP_LINES

Bits	Name	Memory Access	Description
31:10			Reserved Field: Yes
9:0	vpg_vfp_lines	R/W	<p>The vertical front porch period measured in number of horizontal lines. The smallest value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_VFP_LINES_vpg_vfp_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.12 VPG_ACT_LINES

- **Name:** VPG vertical resolution.
- **Description:** Vertical resolution of video pattern.
- **Size:** 32 bits
- **Offset:** 0x22c
- **Exists:** CSI2_DEVICE_VPG==1

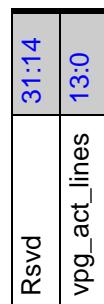


Table 5-54 Fields for Register: VPG_ACT_LINES

Bits	Name	Memory Access	Description
31:14			Reserved Field: Yes
13:0	vpg_act_lines	R/W	<p>The vertical active period measured in number of horizontal lines. The smallest value is 8.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_ACT_LINES_vpg_act_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.13 VPG_MAX_FRAME_NUM

- **Name:** VPG Maximum Frame Number.
- **Description:** Maximum Frame Number.
- **Size:** 32 bits
- **Offset:** 0x230
- **Exists:** CSI2_DEVICE_VPG==1



Table 5-55 Fields for Register: VPG_MAX_FRAME_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	vpg_max_frame_num	R/W	<p>Define the max frame number under Frame Number Increments one mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_MAX_FRAME_NUM_vpg_max_frame_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.14 VPG_START_LINE_NUM

- **Name:** VPG Start Line Number.
- **Description:** Start Line Number.
- **Size:** 32 bits
- **Offset:** 0x234
- **Exists:** CSI2_DEVICE_VPG==1



Table 5-56 Fields for Register: VPG_START_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	vpg_start_line_num	R/W	<p>Define the start line number under Line Number Increments Arbitrary Value mode. The value must be a non-zero value.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_START_LINE_NUM_vpg_start_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.15 VPG_STEP_LINE_NUM

- **Name:** VPG Step Line Number.
- **Description:** Step Line Number.
- **Size:** 32 bits
- **Offset:** 0x238
- **Exists:** CSI2_DEVICE_VPG==1



Table 5-57 Fields for Register: VPG_STEP_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	vpg_step_line_num	R/W	<p>Define the step value for line number under Line Number Increments Arbitrary Value mode. The value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_STEP_LINE_NUM_vpg_step_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.3.16 VPG_BK_LINES

- **Name:** VPG blanking lines.
- **Description:** Frame blanking period.
- **Size:** 32 bits
- **Offset:** 0x23c
- **Exists:** CSI2_DEVICE_VPG==1

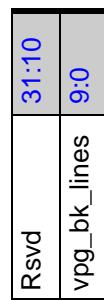


Table 5-58 Fields for Register: VPG_BK_LINES

Bits	Name	Memory Access	Description
31:10			Reserved Field: Yes
9:0	vpg_bk_lines	R/W	<p>The Frame blanking period measured in number of horizontal lines. The smallest value is 0.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (VPG_BK_LINES_vpg_bk_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4 DWC_mipicsi2_device_MemMap/PHY Registers

5.2.4.1 PHY_RSTZ

- **Name:** D-PHY reset and PLL control.
- **Description:** This register controls resets and the PLL of the D-PHY.
- **Size:** 32 bits
- **Offset:** 0x300
- **Exists:** Always

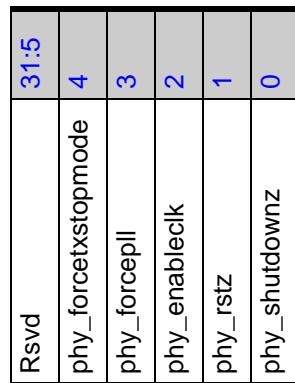


Table 5-59 Fields for Register: PHY_RSTZ

Bits	Name	Memory Access	Description
31:5			Reserved Field: Yes
4	phy_forcetxstopmode	R/W	<p>When set to 1, this bit enables the PHY return to stopstate mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_RSTZ_phy_forcetxstopmode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	phy_forcepll	R/W	<p>When the D-PHY is in ULPS, this bit enables the D-PHY PLL.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_RSTZ_phy_forcepll) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	phy_enableclk	R/W	<p>When set to 1, this bit enables the D-PHY Clock Lane Module.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_RSTZ_phy_enableclk) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	phy_rstz	R/W	<p>When set to 0, this bit places the digital section of the D-PHY in the reset state.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_RSTZ_phy_rstz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	phy_shutdownz	R/W	<p>When set to 0, this bit places the complete D-PHY macro in power-down state.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_RSTZ_phy_shutdownz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.2 PHY_IF_CFG

- **Name:** PHY active lanes number configuration.
- **Description:** This register configures the number of active lanes.
- **Size:** 32 bits
- **Offset:** 0x304
- **Exists:** Always

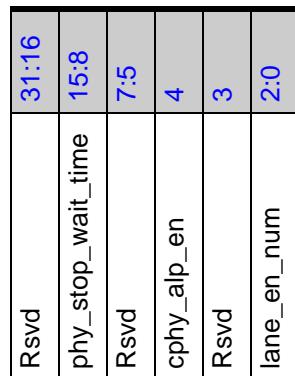


Table 5-60 Fields for Register: PHY_IF_CFG

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:8	phy_stop_wait_time	R/W	<p>This field configures the minimum wait period to request a high-speed transmission after all the active data lanes stopstate signal are asserted.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_IF_CFG_phy_stop_wait_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7:5			Reserved Field: Yes
4	cphy_alp_en	R/W	<p>This field configures C-PHY Alternative Low Power state is enabled</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_IF_CFG_cphy_alp_en) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_CPHY_TYPE==1</p>
3			Reserved Field: Yes

Bits	Name	Memory Access	Description
2:0	lane_en_num	R/W	<p>This field configures the number of active data lanes:</p> <ul style="list-style-type: none"> ■ 111: 8 data lanes, lanes 0 - 7 ■ 110: 7 data lanes, lanes 0 - 6 ■ 101: 6 data lanes, lanes 0 - 5 ■ 100: 5 data lanes, lanes 0 - 4 ■ 011: 4 data lanes, lanes 0 - 3 ■ 010: 3 data lanes, lanes 0 - 2 ■ 001: 2 data lanes, lanes 0 - 1 ■ 000: 1 data lane, lane 0 <p>Can only be updated when phy_shutdownz and phy_rstz are both low.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_IF_CFG_lane_en_num) <p>Value After Reset: CSI2_DEVICE_LANE_NUM_QST_RST_VAL</p> <p>Exists: Always</p>

5.2.4.3 LPCLK_CTRL

- **Name:** Non continuous clock control.
- **Description:** This register configures the possibility for using non continuous clock for RX PHY.
- **Size:** 32 bits
- **Offset:** 0x308
- **Exists:** Always

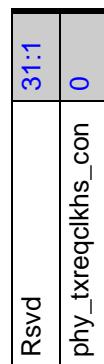


Table 5-61 Fields for Register: LPCLK_CTRL

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	phy_txreqclkhs_con	R/W	<p>This bit controls the PHY PPI txrequestclkhs signal:</p> <ul style="list-style-type: none"> ■ 1: continuous clock mode. ■ 0: non-continuous clock mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (LPCLK_CTRL_phy_txreqclkhs_con) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.4 PHY_ULPS_CTRL

- **Name:** PHY ULPS control.
- **Description:** This register configures entering and leaving ULPS in the PHY.
- **Size:** 32 bits
- **Offset:** 0x30c
- **Exists:** Always

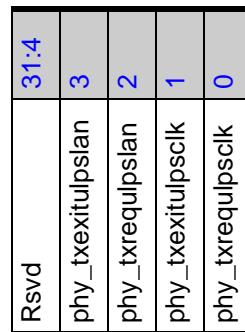


Table 5-62 Fields for Register: PHY_ULPS_CTRL

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	phy_txexitulpslan	R/W	<p>ULPS mode Exit on all active data lanes. (when asserted, duration need exceed 1 tx_esc_clk cycle).</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_ULPS_CTRL_phy_txexitulpslan) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	phy_txrequlpslan	R/W	<p>ULPS mode Request on all active data lanes. (when asserted, duration need exceed 1 tx_esc_clk cycle).</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_ULPS_CTRL_phy_txrequlpslan) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	phy_txexitulpsclk	R/W	<p>ULPS mode Exit on clock lane. (when asserted, duration need exceed 1 tx_esc_clk cycle).</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_ULPS_CTRL_phy_txexitulpsclk) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_DPHY_TYPE==1</p>

Bits	Name	Memory Access	Description
0	phy_txrequlpsclk	R/W	<p>ULPS mode Request on clock lane. (when asserted, the duration need exceed 1 tx_esc_clk cycle).</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (PHY_ULPS_CTRL_phy_txrequlpsclk) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_DPHY_TYPE==1</p>

5.2.4.5 CLKMGR_CFG

- **Name:** Lanebyteclk divide configuration.
- **Description:** This register configures the factor for internal dividers to divide lanebyteclk for timeout purposes.
- **Size:** 32 bits
- **Offset:** 0x310
- **Exists:** Always

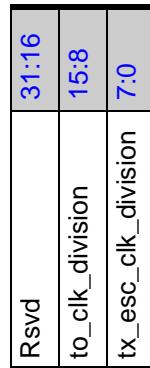


Table 5-63 Fields for Register: CLKMGR_CFG

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:8	to_clk_division	R/W	<p>This field indicates the division factor for the Time Out clock used as the timing unit in the configuration of high-speed to low-power and low-power to high-speed transition error.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (CLKMGR_CFG_to_clk_division) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7:0	tx_esc_clk_division	R/W	<p>This field indicates the division factor for the TX Escape clock source (lanebyteclk). The values 0 and 1 stop the TX_ESC clock generation.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (CLKMGR_CFG_tx_esc_clk_division) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.6 PHY_CAL

- **Name:** D-PHY calibration.
- **Description:** This register configures the calibration of D-PHY.
- **Size:** 32 bits
- **Offset:** 0x314
- **Exists:** ((CSI2_DEVICE_GEN3DPHY == 1) || (CSI2_DEVICE_SNPS_PHY == 0)) == 1

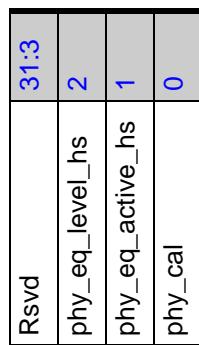


Table 5-64 Fields for Register: PHY_CAL

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	phy_eq_level_hs	R/W	<p>This bit controls equalization level pin of D-PHY:</p> <ul style="list-style-type: none"> ■ 1: high-level, 7dB +/- 1dB. ■ 0: low-level, 3.5dB +/- 1dB. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_CAL_phy_eq_level_hs) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	phy_eq_active_hs	R/W	<p>This bit controls equalization pin of D-PHY:</p> <ul style="list-style-type: none"> ■ 1: equalization is activated. ■ 0: No equalization. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_CAL_phy_eq_active_hs) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
0	phy_cal	R/W	<p>This bit controls calibration pin of D-PHY:</p> <ul style="list-style-type: none"> ■ 1: High-speed transmit skew calibration is activated. ■ 0: No calibration. <p>For initial calibration, controller will assert calibration pin immediately when D-PHY enters stopstate. For periodic calibration, controller will not response to this configuration until D-PHY finishes current HS transmission and enters stopstate.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_CAL_phy_cal) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.7 PHY_CAL_STATUS

- **Name:** D-PHY calibration status.
- **Description:** This register indicates the calibration status of D-PHY.
- **Size:** 32 bits
- **Offset:** 0x318
- **Exists:** ((CSI2_DEVICE_GEN3DPHY == 1) || (CSI2_DEVICE_SNPS_PHY == 0)) == 1

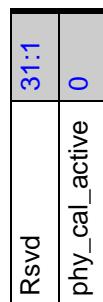


Table 5-65 Fields for Register: PHY_CAL_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	phy_cal_active	R	<p>Initial or periodic calibration status:</p> <ul style="list-style-type: none"> ■ 1'b1: Pin txskewcalhs is asserted, causes PHY to generate skew calibration patterns. ■ 1'b0: Pin txskewcalhs is de-asserted, causes PHY to end the calibration operation. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_CAL_STATUS_phy_cal_active) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.8 TO_CNT_CFG

- **Name:** Timeout counter configuration.
- **Description:** This register configures counters that trigger timeout errors. These are used to warn the system of a failure, through an interrupt, and restart the core in case of unexpected situations that cause deadlock conditions.
- **Size:** 32 bits
- **Offset:** 0x31c
- **Exists:** Always

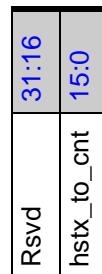


Table 5-66 Fields for Register: TO_CNT_CFG

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	hstx_to_cnt	R/W	<p>This field configures the timeout counter that triggers a high-speed transmission timeout contention detection, measured in tx_esc_clk cycles. For CSI2 device, the CSI link returns the low-power state once per packet, then you should configure the TO_CLK_DIVISION and hstx_to_cnt to be in accordance with: hstx_to_cnt * lanebyteclkperiod * TO_CLK_DIVISION >= the time of one data packet transmission in PPI * 110%</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (TO_CNT_CFG_hstx_to_cnt) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.9 PHY_MODE

- **Name:** PHY mode.
- **Description:** This register configures PHY mode.
- **Size:** 32 bits
- **Offset:** 0x320
- **Exists:** Always

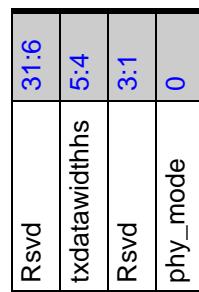


Table 5-67 Fields for Register: PHY_MODE

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5:4	txdatawidthhs	R/W	<p>This field configures PHY data width mode for high-speed:</p> <ul style="list-style-type: none"> ■ 11: reserved ■ 10: 32-bit mode; ■ 01: 16-bit mode; ■ 00: 8-bit mode, only for D-PHY, not support C-PHY <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_MODE_txdatawidthhs) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	phy_mode	R/W	<p>This field configures PHY mode:</p> <ul style="list-style-type: none"> ■ 1: C-PHY mode; ■ 0: D-PHY mode; <p>PHY mode can be switched only when all lanes are at stopstate. After switching, PHY needs to be re-initialized.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_MODE_phy_mode) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_PHY_TYPE==3</p>

5.2.4.10 PHY_SWITCH_TIME

- **Name:** PHY switch time.
- **Description:** This register configures PHY switch time.
- **Size:** 32 bits
- **Offset:** 0x324
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1

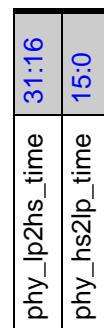


Table 5-68 Fields for Register: PHY_SWITCH_TIME

Bits	Name	Memory Access	Description
31:16	phy_lp2hs_time	R/W	<p>phy_lp2hs_time: The time from txrequest_l0 is asserted to txreadyhs_l0 is asserted.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_SWITCH_TIME_phy_lp2hs_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15:0	phy_hs2lp_time	R/W	<p>phy_hs2lp_time: The time from txrequest_l0 is de-asserted to all the active lanes stopstate signals are asserted.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_SWITCH_TIME_phy_hs2lp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.11 PHY_TIMING_CALC_CTRL

- **Name:** PHY timing calculation control.
- **Description:** This register controls the enable trigger for PHY timing calculation function.
- **Size:** 32 bits
- **Offset:** 0x328
- **Exists:** Always

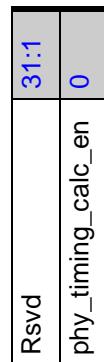


Table 5-69 Fields for Register: PHY_TIMING_CALC_CTRL

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	phy_timing_calc_en	R/WC	<p>PHY timing calculation enable signal. Write "1" to trigger the calculation once, and writing the bit clears it.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_TIMING_CALC_CTRL_phy_timing_calc_en) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.12 PHY_TIMING_CALC_STATUS

- **Name:** PHY timing calculation status.
- **Description:** This register contains the status of PHY timing calculation function.
- **Size:** 32 bits
- **Offset:** 0x32c
- **Exists:** Always

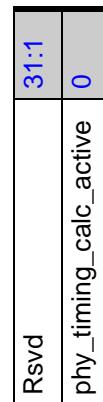


Table 5-70 Fields for Register: PHY_TIMING_CALC_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	phy_timing_calc_active	R	<p>PHY timing calculation status:</p> <ul style="list-style-type: none"> ■ 1'b1: PHY timing is under calculation and phy_timing_calc_en should not be set to 1 again. ■ 1'b0: PHY timing calculation is done, normal packet transmission or a second trigger for calculation can be launched. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_TIMING_CALC_STATUS_phy_timing_calc_active) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.13 PHY_SWITCH_TIME_AUTO

- **Name:** Automatically calculated PHY switch time.
- **Description:** This register shows PHY switch time which is calculated automatically by controller.
- **Size:** 32 bits
- **Offset:** 0x330
- **Exists:** Always

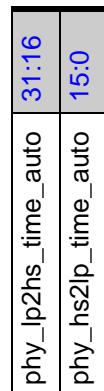


Table 5-71 Fields for Register: PHY_SWITCH_TIME_AUTO

Bits	Name	Memory Access	Description
31:16	phy_lp2hs_time_auto	R	<p>Automatically calculated value for phy_lp2hs_time: The time from txrequest_I0 is asserted to txreadyhs_I0 is asserted.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_SWITCH_TIME_AUTO_phy_lp2hs_time_auto) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15:0	phy_hs2lp_time_auto	R	<p>Automatically calculated value for phy_hs2lp_time: The time from txrequest_I0 is de-asserted to all the active lanes stopstate signals are asserted.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_SWITCH_TIME_AUTO_phy_hs2lp_time_auto) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.14 PHY_STATUS

- **Name:** PHY status.
- **Description:** This register contains information about the status of the PHY.
- **Size:** 32 bits
- **Offset:** 0x334
- **Exists:** Always

Rsvd	31:22
TxUlpsActiveNot_L7	21
TxStopState_L7	20
TxUlpsActiveNot_L6	19
TxStopState_L6	18
TxUlpsActiveNot_L5	17
TxStopState_L5	16
TxUlpsActiveNot_L4	15
TxStopState_L4	14
TxUlpsActiveNot_L3	13
TxStopState_L3	12
TxUlpsActiveNot_L2	11
TxStopState_L2	10
TxUlpsActiveNot_L1	9
TxStopState_L1	8
TxUlpsActiveNot_L0	7
TxStopState_L0	6
TxUlpsActiveNot_Clk	5
TxStopState_Clk	4
PLL_Lock	3
lane_max_num	2:0

Table 5-72 Fields for Register: PHY_STATUS

Bits	Name	Memory Access	Description
31:22			Reserved Field: Yes
21	TxUlpsActiveNot_L7	R	This bit indicates the status of TxUlpsActiveNot_L7 PHY signal, the reset value refers to PHY Databook. Values: <ul style="list-style-type: none">■ 0x0 (PHY_STATUS_TxUlpsActiveNot_L7) Value After Reset: 0x0 Exists: (CSI2_DEVICE_DPHY_INCLUDE_LANE_7)==1
20	TxStopState_L7	R	This bit indicates the status of TxStopState_L7 PHY signal, the reset value refers to PHY Databook. Values: <ul style="list-style-type: none">■ 0x0 (PHY_STATUS_TxStopState_L7) Value After Reset: 0x0 Exists: (CSI2_DEVICE_DPHY_INCLUDE_LANE_7)==1
19	TxUlpsActiveNot_L6	R	This bit indicates the status of TxUlpsActiveNot_L6 D-PHY signal, the reset value refers to PHY Databook. Values: <ul style="list-style-type: none">■ 0x0 (PHY_STATUS_TxUlpsActiveNot_L6) Value After Reset: 0x0 Exists: (CSI2_DEVICE_DPHY_INCLUDE_LANE_6)==1

Bits	Name	Memory Access	Description
18	TxStopState_L6	R	<p>This bit indicates the status of TxStopState_L6 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxStopState_L6) <p>Value After Reset: 0x0</p> <p>Exists: (CSI2_DEVICE_DPHY_INCLUDE_LANE_6)==1</p>
17	TxUlpsActiveNot_L5	R	<p>This bit indicates the status of TxUlpsActiveNot_L5 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxUlpsActiveNot_L5) <p>Value After Reset: 0x0</p> <p>Exists: (CSI2_DEVICE_DPHY_INCLUDE_LANE_5)==1</p>
16	TxStopState_L5	R	<p>This bit indicates the status of TxStopState_L5 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxStopState_L5) <p>Value After Reset: 0x0</p> <p>Exists: (CSI2_DEVICE_DPHY_INCLUDE_LANE_5)==1</p>
15	TxUlpsActiveNot_L4	R	<p>This bit indicates the status of TxUlpsActiveNot_L4 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxUlpsActiveNot_L4) <p>Value After Reset: 0x0</p> <p>Exists: ((CSI2_DEVICE_DPHY_INCLUDE_LANE_4) (CSI2_DEVICE_CPHY_INCLUDE_LANE_4))==1</p>
14	TxStopState_L4	R	<p>This bit indicates the status of TxStopState_L4 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxStopState_L4) <p>Value After Reset: 0x0</p> <p>Exists: ((CSI2_DEVICE_DPHY_INCLUDE_LANE_4) (CSI2_DEVICE_CPHY_INCLUDE_LANE_4))==1</p>
13	TxUlpsActiveNot_L3	R	<p>This bit indicates the status of TxUlpsActiveNot_L3 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxUlpsActiveNot_L3) <p>Value After Reset: 0x0</p> <p>Exists: ((CSI2_DEVICE_DPHY_INCLUDE_LANE_3) (CSI2_DEVICE_CPHY_INCLUDE_LANE_3))==1</p>

Bits	Name	Memory Access	Description
12	TxStopState_L3	R	<p>This bit indicates the status of TxStopState_L3 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxStopState_L3) <p>Value After Reset: 0x0</p> <p>Exists: ((CSI2_DEVICE_DPHY_INCLUDE_LANE_3) (CSI2_DEVICE_CPHY_INCLUDE_LANE_3))==1</p>
11	TxUlpsActiveNot_L2	R	<p>This bit indicates the status of TxUlpsActiveNot_L2 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxUlpsActiveNot_L2) <p>Value After Reset: 0x0</p> <p>Exists: ((CSI2_DEVICE_DPHY_INCLUDE_LANE_2) (CSI2_DEVICE_CPHY_INCLUDE_LANE_2))==1</p>
10	TxStopState_L2	R	<p>This bit indicates the status of TxStopState_L2 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxStopState_L2) <p>Value After Reset: 0x0</p> <p>Exists: ((CSI2_DEVICE_DPHY_INCLUDE_LANE_2) (CSI2_DEVICE_CPHY_INCLUDE_LANE_2))==1</p>
9	TxUlpsActiveNot_L1	R	<p>This bit indicates the status of TxUlpsActiveNot_L1 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxUlpsActiveNot_L1) <p>Value After Reset: 0x0</p> <p>Exists: ((CSI2_DEVICE_DPHY_INCLUDE_LANE_1) (CSI2_DEVICE_CPHY_INCLUDE_LANE_1))==1</p>
8	TxStopState_L1	R	<p>This bit indicates the status of TxStopState_L1 PHY signal, the reset value refers to PHY Databook.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxStopState_L1) <p>Value After Reset: 0x0</p> <p>Exists: ((CSI2_DEVICE_DPHY_INCLUDE_LANE_1) (CSI2_DEVICE_CPHY_INCLUDE_LANE_1))==1</p>

Bits	Name	Memory Access	Description
7	TxUlpsActiveNot_L0	R	<p>This bit indicates the status of TxUlpsActiveNot_L0 PHY signal:</p> <ul style="list-style-type: none"> ■ 1: Not ulps state of data lane0; ■ 0: ULPS state of data lane0. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxUlpsActiveNot_L0) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	TxStopState_L0	R	<p>This bit indicates the status of TxStopState_L0 PHY signal:</p> <ul style="list-style-type: none"> ■ 1: Stop state of data lane0; ■ 0: Not stop state of data lane0. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxStopState_L0) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	TxUlpsActiveNot_Clk	R	<p>This bit indicates the status of TxUlpsActiveNot_Clk D-PHY signal:</p> <ul style="list-style-type: none"> ■ 1: Not ulps state of clock lane; ■ 0: ULPS state of clock lane. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxUlpsActiveNot_Clk) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_DPHY_TYPE==1</p>
4	TxStopState_Clk	R	<p>This bit indicates the status of TxStopState_Clk D-PHY signal:</p> <ul style="list-style-type: none"> ■ 1: Stop state of clock lane. ■ 0: Not stop state of clock lane. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_TxStopState_Clk) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_DPHY_TYPE==1</p>

Bits	Name	Memory Access	Description
3	PLL_Lock	R	<p>This bit indicates the status of phylock D-PHY signal:</p> <ul style="list-style-type: none"> ■ 1: PLL locked; ■ 0: PLL unlocked. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_PLL_Lock) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2:0	lane_max_num	R	<p>This 3-bit indicates the maximum number of D-PHY lanes.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_STATUS_lane_max_num) <p>Value After Reset: CSI2_DEVICE_LANE_NUM_QST_RST_VAL</p> <p>Exists: Always</p>

5.2.4.15 PHY0_TST_CTRL0

- **Name:** D-PHY0 clock and clear pins control.
- **Description:** This register controls clock and clear pins of the D-PHY0 vendor specific interface.
- **Size:** 32 bits
- **Offset:** 0x338
- **Exists:** Always

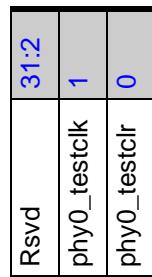


Table 5-73 Fields for Register: PHY0_TST_CTRL0

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	phy0_testclk	R/W	<p>This bit is used to clock the TESTDIN bus into the D-PHY0.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY0_TST_CTRL0_phy0_testclk) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	phy0_testclr	R/W	<p>PHY0 test interface clear. Active High.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY0_TST_CTRL0_phy0_testclr) <p>Value After Reset: 0x1</p> <p>Exists: Always</p>

5.2.4.16 PHY0_TST_CTRL1

- **Name:** D-PHY0 data and enable pins control.
- **Description:** This register controls data and enable pins of the D-PHY0 vendor specific interface.
- **Size:** 32 bits
- **Offset:** 0x33c
- **Exists:** Always



Table 5-74 Fields for Register: PHY0_TST_CTRL1

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16	phy0_testen	R/W	<p>PHY0 test interface operation selector:</p> <ul style="list-style-type: none"> ■ 1: the address write operation is set on the falling edge of the testclk signal. ■ 0: the data write operation is set on the rising edge of the testclk signal. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY0_TST_CTRL1_phy0_testen) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15:8	phy0_testdout	R	<p>PHY0 output 8-bit data bus for read-back and internal probing functionalities.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY0_TST_CTRL1_phy0_testdout) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7:0	phy0_testdin	R/W	<p>PHY0 test interface input 8-bit data bus for internal register programming and test functionalities access.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (PHY0_TST_CTRL1_phy0_testdin) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.17 PHY1_TST_CTRL0

- **Name:** D-PHY1 clock and clear pins control.
- **Description:** This register controls clock and clear pins of the D-PHY1 vendor specific interface.
- **Size:** 32 bits
- **Offset:** 0x340
- **Exists:** CSI2_DEVICE_DPHY_NUM_OF_LANES>4

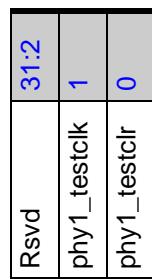


Table 5-75 Fields for Register: PHY1_TST_CTRL0

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	phy1_testclk	R/W	<p>This bit is used to clock the TESTDIN bus into the D-PHY1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY1_TST_CTRL0_phy1_testclk) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	phy1_testclr	R/W	<p>PHY1 test interface clear. Active High.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY1_TST_CTRL0_phy1_testclr) <p>Value After Reset: 0x1</p> <p>Exists: Always</p>

5.2.4.18 PHY1_TST_CTRL1

- **Name:** D-PHY1 data and enable pins control.
- **Description:** This register controls data and enable pins of the D-PHY1 vendor specific interface.
- **Size:** 32 bits
- **Offset:** 0x344
- **Exists:** CSI2_DEVICE_DPHY_NUM_OF_LANES>4

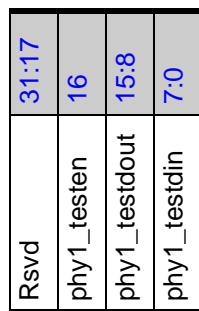


Table 5-76 Fields for Register: PHY1_TST_CTRL1

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16	phy1_testen	R/W	<p>PHY1 test interface operation selector:</p> <ul style="list-style-type: none"> ■ 1: the address write operation is set on the falling edge of the testclk signal. ■ 0: the data write operation is set on the rising edge of the testclk signal. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY1_TST_CTRL1_phy1_testen) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15:8	phy1_testdout	R	<p>PHY1 output 8-bit data bus for read-back and internal probing functionalities.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY1_TST_CTRL1_phy1_testdout) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7:0	phy1_testdin	R/W	<p>PHY1 test interface input 8-bit data bus for internal register programming and test functionalities access.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (PHY1_TST_CTRL1_phy1_testdin) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.19 TX_REG_CSI_EPD_EN_SSP

- **Name:** EPD option control for short packet.
- **Description:** This register controls EPD enable and the minimum or fixed spacer number following short packet.
- **Size:** 32 bits
- **Offset:** 0x348
- **Exists:** Always

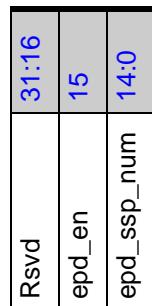


Table 5-77 Fields for Register: TX_REG_CSI_EPD_EN_SSP

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	epd_en	R/W	<ul style="list-style-type: none"> ■ 1'b0: Disable EPD. ■ 1'b1: Enable EPD. Values: <ul style="list-style-type: none"> ■ 0x0 (TX_REG_CSI_EPD_EN_SSP_epd_en) Value After Reset: 0x0 Exists: Always
14:0	epd_ssp_num	R/W	<p>This field indicates the minimum or fixed spacer number depend on different EPD modes.</p> <ul style="list-style-type: none"> ■ For D-PHY Option 1, Option 2 with Variable Spacers and C-PHY: the minimum spacer number per lane following a short packet. ■ Otherwise, for D-PHY Option 2: the fixed number per lane following a short packet. Values: <ul style="list-style-type: none"> ■ 0x0 (TX_REG_CSI_EPD_EN_SSP_epd_ssp_num) Value After Reset: 0x0 Exists: Always

5.2.4.20 TX_REG_CSI_EPD_OP_SLP

- **Name:** EPD option control for long packet.
- **Description:** This register controls EPD Option and the minimum or fixed spacer number following long packet.
- **Size:** 32 bits
- **Offset:** 0x34c
- **Exists:** Always



Table 5-78 Fields for Register: TX_REG_CSI_EPD_OP_SLP

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	epd_op_dphy	R/W	<p>When D-PHY:</p> <ul style="list-style-type: none"> ■ 1'b0: D-PHY EPD Option 1. ■ 1'b1: D-PHY EPD Option 2. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (TX_REG_CSI_EPD_OP_SLP_epd_op_dphy) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_DPHY_TYPE==1</p>
14:0	epd_slp_num	R/W	<p>This field indicates the minimum or fixed spacer number depend on different EPD modes.</p> <ul style="list-style-type: none"> ■ For D-PHY Option 1, Option 2 with Variable Spacers and C-PHY: the minimum spacer number per lane following a long packet. ■ Otherwise, for D-PHY Option 2: the fixed number per lane following a long packet. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (TX_REG_CSI_EPD_OP_SLP_epd_slp_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.21 TX_REG_CSI_EPD_MISC_OPTIONS

- **Name:** EPD misc options control.
- **Description:** This register controls EPD misc options like spacers-without-pdq, eotp and etc.
- **Size:** 32 bits
- **Offset:** 0x350
- **Exists:** Always

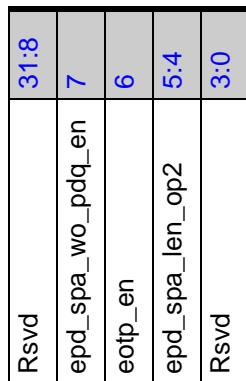


Table 5-79 Fields for Register: TX_REG_CSI_EPD_MISC_OPTIONS

Bits	Name	Memory Access	Description
31:8			Reserved Field: Yes
7	epd_spa_wo_pdq_en	R/W	<p>For C-PHY or D-PHY EPD Option1, enable insertion of Spacers-without-PDQ after CSI-2 packets just prior to C/D-PHY EoT.</p> <ul style="list-style-type: none"> ■ 1'b0: Disable Spacers-without-PDQ. ■ 1'b1: Enable Spacers-without-PDQ. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (TX_REG_CSI_EPD_MISC_OPTIONS_epd_spa_wo_pdq_en) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	eotp_en	R/W	<p>For D-PHY EPD Option 2:</p> <ul style="list-style-type: none"> ■ 1'b0: Disable EoTp. ■ 1'b1: Enable EoTp. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (TX_REG_CSI_EPD_MISC_OPTIONS_eotp_en) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_DPHY_TYPE==1</p>

Bits	Name	Memory Access	Description
5:4	epd_spa_len_op2	R/W	<p>For D-PHY EPD Option 2, enable variable-length Spacer insertions with a multiple of n Spacer bytes per Lane.</p> <ul style="list-style-type: none"> ■ 2'b00: Enable fixed-length Spacers ■ 2'b01: Enable variable-length Spacers with n=1 ■ 2'b10: Enable variable-length Spacers with n=2 ■ 2'b11: Enable variable-length Spacers with n=4 <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (TX_REG_CSI_EPD_MISC_OPTIONS_epd_spa_len_op2) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_DPHY_TYPE==1</p>
3:0			Reserved Field: Yes

5.2.4.22 TX_REG_CSI_EPD_MAX

- **Name:** EPD spacer maximum number configuration.
- **Description:** This register configures EPD spacer maximum number.
- **Size:** 32 bits
- **Offset:** 0x354
- **Exists:** Always



Table 5-80 Fields for Register: TX_REG_CSI_EPD_MAX

Bits	Name	Memory Access	Description
31:0	epd_spacer_max	R/W	<p>The maximum spacer number per lane following a short or long packet. ONLY used by following mode:</p> <ul style="list-style-type: none"> ■ D-PHY EPD Option 1 when enable Spacers-without-PDQ. ■ D-PHY EPD Option 2 when enable EoTp and variable-length Spacers. ■ C-PHY EPD when enable Spacers-without-PDQ. <p>Should be greater than minimum spacer number configuration following a short or long packet. A whole hline time converted to maximum spacer number is recommended if want to deliver timing in LRTE mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (TX_REG_CSI_EPD_MAX_epd_spacer_max) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.23 TX_HS_IDLE_TIME

- **Name:** HS_IDLE time.
- **Description:** This register configures HS_IDLE time for D-PHY option1 on LRTE mode.
- **Size:** 32 bits
- **Offset:** 0x358
- **Exists:** CSI2_DEVICE_DPHY_TYPE==1&&CSI2_DEVICE_IDI_IPI_IF==1

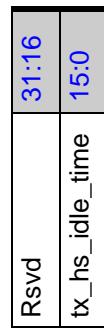


Table 5-81 Fields for Register: TX_HS_IDLE_TIME

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	tx_hs_idle_time	R/W	<p>The PHY from entry to exit HS_IDLE time when EPD D-PHY Option 1 mode, including the SoT sequence time following HS_IDLE. Not used by EPD D-PHY Option 2 and C-PHY mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (TX_HS_IDLE_TIME_tx_hs_idle_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.4.24 PHY_HS_IDLE_TIME_AUTO

- **Name:** Automatically calculated HS_IDLE time.
- **Description:** This register shows HS_IDLE time for D-PHY option1 on LRTE mode which is calculated automatically by controller.
- **Size:** 32 bits
- **Offset:** 0x35c
- **Exists:** CSI2_DEVICE_DPHY_TYPE==1

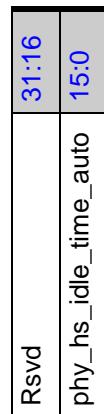


Table 5-82 Fields for Register: PHY_HS_IDLE_TIME_AUTO

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	phy_hs_idle_time_auto	R	<p>Automatically calculated value for hs_idle time: The PHY from entry to exit HS_IDLE time when EPD D-PHY Option 1 mode, including the SoT sequence time following HS_IDLE. Not used by EPD D-PHY Option 2 and C-PHY mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_HS_IDLE_TIME_AUTO_phy_hs_idle_time_auto) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.5 DWC_mipicsi2_device_MemMap/IDI Registers

5.2.5.1 IDI_FIFO_STATUS

- **Name:** IDI fifo status.
- **Description:** IDI channel fifo status.
- **Size:** 32 bits
- **Offset:** 0x400
- **Exists:** CSI2_DEVICE_IDI_IF==1

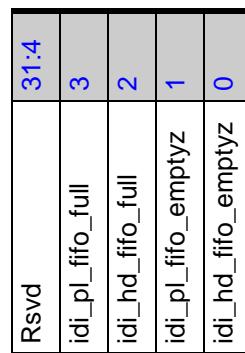


Table 5-83 Fields for Register: IDI_FIFO_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	idi_pl_fifo_full	R	<p>This field indicates the payload FIFO full status of IDI interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFO_STATUS_idi_pl_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	idi_hd_fifo_full	R	<p>This field indicates the header FIFO full status of IDI interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFO_STATUS_idi_hd_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	idi_pl_fifo_emptyz	R	<p>This field indicates the payload FIFO empty status of IDI interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFO_STATUS_idi_pl_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
0	idi_hd_fifo_emptyz	R	<p>This field indicates the header FIFO empty status of IDI interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IDI_FIFO_STATUS_idi_hd_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.6 DWC_mipicsi2_device_MemMap/SDI Registers

5.2.6.1 SDI_CFG

- **Name:** SDI Control
- **Description:** Control SDI mode
- **Size:** 32 bits
- **Offset:** 0x500
- **Exists:** CSI2_DEVICE_SDI_IF==1

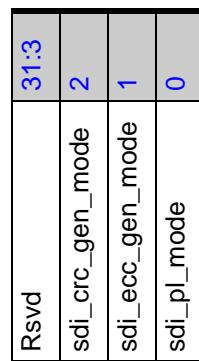


Table 5-84 Fields for Register: SDI_CFG

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	sdi_crc_gen_mode	R/W	<p>Configure CRC mode: - 1: Bypass mode - 0: Generation mode</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (SDI_CFG_sdi_crc_gen_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	sdi_ecc_gen_mode	R/W	<p>Configure ECC/PH-CRC mode: - 1: Bypass mode - 0: Generation mode</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (SDI_CFG_sdi_ecc_gen_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	sdi_pl_mode	R/W	<p>Configure Payload format: - 1: Packet Format - 0: Memory storage format</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (SDI_CFG_sdi_pl_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7 DWC_mipicsi2_device_MemMap/IPI Registers

5.2.7.1 IPI_PKT_CFG

- **Name:** IPI packet configuration.
- **Description:** This register indicates the packet configuration of IPI.
- **Size:** 32 bits
- **Offset:** 0x600
- **Exists:** CSI2_DEVICE_IPI_IF==1

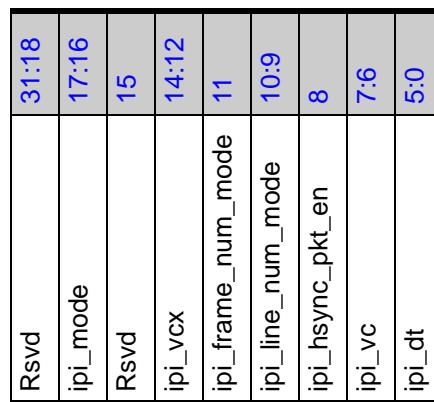


Table 5-85 Fields for Register: IPI_PKT_CFG

Bits	Name	Memory Access	Description
31:18			Reserved Field: Yes
17:16	ipi_mode	R/W	<p>This field is to select the IPI mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Backpressure Mode - Provides a backpressure mechanism to inform the IPI driver that the memory is almost full. In this mode, IPI timing can't be maintained. ■ 01: Cut through Mode - Initiates the HS transmission to the PHY after the configured number of ipi_clk cycles. Makes use of very shallow memory. ■ 00: Store and Forward Mode - Stores the full pixel packet before forwarding. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_PKT_CFG_ipi_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15			Reserved Field: Yes

Bits	Name	Memory Access	Description
14:12	ipi_vc _x	R/W	<p>The Virtual Channel bit[4:2] of IPI packet</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_PKT_CFG_ipi_vc_x) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_VC_EXTENSION==1</p>
11	ipi_frame_num_mode	R/W	<p>This field indicates the IPI frame number mode:</p> <ul style="list-style-type: none"> ■ 1: Frame Number Increments One mode. ■ 0: Frame Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_PKT_CFG_ipi_frame_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10:9	ipi_line_num_mode	R/W	<p>This field indicates IPI line number mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Line Number Increments Arbitrary Value mode. ■ 01: Line Number Increments One mode. ■ 00: Line Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_PKT_CFG_ipi_line_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	ipi_hsync_pkt_en	R/W	<p>This field indicates the line synchronization packets mode:</p> <ul style="list-style-type: none"> ■ 1: Transmit line synchronization packets. ■ 0: Don't transmit line synchronization packets. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_PKT_CFG_ipi_hsync_pkt_en) <p>Value After Reset: 0x1</p> <p>Exists: Always</p>
7:6	ipi_vc	R/W	<p>The Virtual Channel bit[1:0] of IPI packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_PKT_CFG_ipi_vc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5:0	ipi_dt	R/W	<p>The Data Type of IPI packet.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IPI_PKT_CFG_ipi_dt) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.2 IPI_PIXELS

- **Name:** IPI pixels.
- **Description:** IPI Horizontal resolution.
- **Size:** 32 bits
- **Offset:** 0x604
- **Exists:** CSI2_DEVICE_IPI_IF==1



Table 5-86 Fields for Register: IPI_PIXELS

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi_pixels	R/W	<p>This field configures horizontal resolution for IPI. For YUV,RGB and RAW, this field configures pixel number of data packet for IPI; For User Defined Byte-based Data, this field configures byte number of data packet for IPI.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_PIXELS_ipi_pixels) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.3 IPI_MAX_FRAME_NUM

- **Name:** IPI max frame number.
- **Description:** Max frame number of IPI frame synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x608
- **Exists:** CSI2_DEVICE_IPI_IF==1



Table 5-87 Fields for Register: IPI_MAX_FRAME_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi_max_frame_num	R/W	<p>If Frame Number Increments One mode is selected, this field is to define the max frame number of IPI frame synchronization packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_MAX_FRAME_NUM_ipi_max_frame_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.4 IPI_START_LINE_NUM

- **Name:** IPI start line number.
- **Description:** Start line number of IPI line synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x60c
- **Exists:** CSI2_DEVICE_IPI_IF==1



Table 5-88 Fields for Register: IPI_START_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi_start_line_num	R/W	<p>If Line Number Increments Arbitrary Value mode is selected, this field is to define the start line number of line synchronization packet. The start value must be a non-zero.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_START_LINE_NUM_ipi_start_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.5 IPI_STEP_LINE_NUM

- **Name:** IPI step line number.
- **Description:** Step Line Number of IPI line synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x610
- **Exists:** CSI2_DEVICE_IPI_IF==1



Table 5-89 Fields for Register: IPI_STEP_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi_step_line_num	R/W	<p>If Line Number Increments Arbitrary Value mode is selected, this field is to define the step value for Line number increments. The step value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_STEP_LINE_NUM_ipi_step_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.6 IPI_LINES

- **Name:** IPI lines.
- **Description:** IPI Vertical resolution.
- **Size:** 32 bits
- **Offset:** 0x614
- **Exists:** CSI2_DEVICE_IPI_IF==1

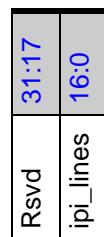


Table 5-90 Fields for Register: IPI_LINES

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi_lines	R/W	<p>This field configures line number of a frame for IPI(VSA+VBP+VACT+VFP) The step value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_LINES_ipi_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.7 IPI_DATA_SEND_START

- **Name:** IPI data send start.
- **Description:** IPI Data Send Start in Cut through Mode.
- **Size:** 32 bits
- **Offset:** 0x618
- **Exists:** CSI2_DEVICE_IPI_IF==1

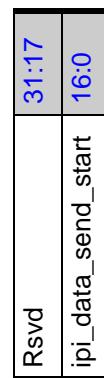


Table 5-91 Fields for Register: IPI_DATA_SEND_START

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi_data_send_start	R/W	<p>This field configures the number of ipi_clk cycles for start to send data packet after the ipi_data_en is asserted in Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_DATA_SEND_START_ipi_data_send_start) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.8 IPI_FIFO_STATUS

- **Name:** IPI fifo status.
- **Description:** IPI channel fifo status.
- **Size:** 32 bits
- **Offset:** 0x61c
- **Exists:** CSI2_DEVICE_IPI_IF==1

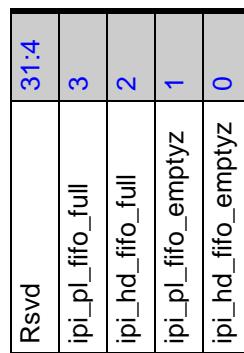


Table 5-92 Fields for Register: IPI_FIFO_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	ipi_pl_fifo_full	R	<p>This field indicates the payload FIFO full status of IPI interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFO_STATUS_ipi_pl_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	ipi_hd_fifo_full	R	<p>This field indicates the header FIFO full status of IPI interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFO_STATUS_ipi_hd_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	ipi_pl_fifo_emptyz	R	<p>This field indicates the payload FIFO empty status of IPI interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFO_STATUS_ipi_pl_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
0	ipi_hd_fifo_emptyz	R	<p>This field indicates the header FIFO empty status of IPI interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IPI_FIFO_STATUS_ipi_hd_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.9 IPI_TRANS_STATUS

- **Name:** IPI transmission status.
- **Description:** IPI transmission status.
- **Size:** 32 bits
- **Offset:** 0x640
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1



Table 5-93 Fields for Register: IPI_TRANS_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	ipi_trans_active	R	<p>This bit indicates the status of IPI data transmission:</p> <ul style="list-style-type: none"> ■ 1: IPI packets are under transmission. ■ 0: No IPI packets are under transmission. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_TRANS_STATUS_ipi_trans_active) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.10 IPI_HSA_HBP_PPI_TIME

- **Name:** IPI HSA and HBP PPI time.
- **Description:** Horizontal Synchronism Active(HSA) and Horizontal Back Porch(HBP) time.
- **Size:** 32 bits
- **Offset:** 0x644
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1

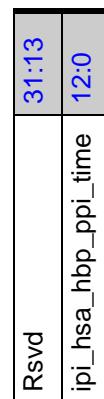


Table 5-94 Fields for Register: IPI_HSA_HBP_PPI_TIME

Bits	Name	Memory Access	Description
31:13			Reserved Field: Yes
12:0	ipi_hsa_hbp_ppi_time	R/W	<p>This bit configures the total time of horizontal synchronism active period and horizontal back porch period(HSA+HBP). Measured in lane byte clock cycles. The minimum value is 2*CSI2_DEVICE_DFLT_F_SYNC_TYPE.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_HSA_HBP_PPI_TIME_ipi_hsa_hbp_ppi_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.11 IPI_HLINE_PPI_TIME

- **Name:** IPI each line PPI time.
- **Description:** Overall time for each video line in PPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x648
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1



Table 5-95 Fields for Register: IPI_HLINE_PPI_TIME

Bits	Name	Memory Access	Description
31:19			Reserved Field: Yes
18:0	ipi_hline_ppi_time	R/W	<p>The size of total line time(HSA+HBP+HACT+HFP). Measured in lane byte clock cycles. The configured value should be less than actual IPI input hline time, but also need be greater than half of IPI input hline time. When configure to 0, means simultaneous IDI & IPI is not supported.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_HLINE_PPI_TIME_ipi_hline_ppi_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.12 IPI_VSA_LINES

- **Name:** IPI Vertical Synchronism Active lines.
- **Description:** Vertical Synchronism Active(VSA) period of IPI interface format.
- **Size:** 32 bits
- **Offset:** 0x64c
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1

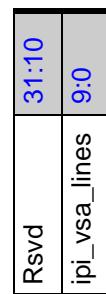


Table 5-96 Fields for Register: IPI_VSA_LINES

Bits	Name	Memory Access	Description
31:10			Reserved Field: Yes
9:0	ipi_vsa_lines	R/W	<p>The vertical synchronism active period measured in number of horizontal lines. The minimum value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_VSA_LINES_ipi_vsa_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.13 IPI_VBP_LINES

- **Name:** IPI Vertical Back Porch lines.
- **Description:** Vertical Back Porch(VBP) period of IPI interface format.
- **Size:** 32 bits
- **Offset:** 0x650
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1

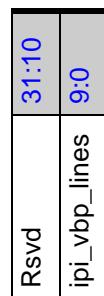


Table 5-97 Fields for Register: IPI_VBP_LINES

Bits	Name	Memory Access	Description
31:10			Reserved Field: Yes
9:0	ipi_vbp_lines	R/W	<p>The vertical back porch period measured in number of horizontal lines. The minimum value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_VBP_LINES_ipi_vbp_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.14 IPI_VFP_LINES

- **Name:** IPI Vertical Front Porch lines.
- **Description:** Vertical Front Porch(VFP) period of IPI interface format.
- **Size:** 32 bits
- **Offset:** 0x654
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1

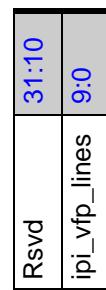


Table 5-98 Fields for Register: IPI_VFP_LINES

Bits	Name	Memory Access	Description
31:10			Reserved Field: Yes
9:0	ipi_vfp_lines	R/W	<p>The vertical front porch period measured in number of horizontal lines. The minimum value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_VFP_LINES_ipi_vfp_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.15 IPI_ACT_LINES

- **Name:** IPI Vertical resolution lines.
- **Description:** Vertical resolution of IPI interface format.
- **Size:** 32 bits
- **Offset:** 0x658
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1

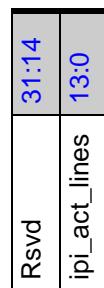


Table 5-99 Fields for Register: IPI_ACT_LINES

Bits	Name	Memory Access	Description
31:14			Reserved Field: Yes
13:0	ipi_act_lines	R/W	<p>The vertical active period measured in number of horizontal lines. The minimum value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_ACT_LINES_ipi_act_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.16 IPI_FB_LINES

- **Name:** IPI frame blanking lines.
- **Description:** Frame blanking of IPI interface format.
- **Size:** 32 bits
- **Offset:** 0x65c
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1

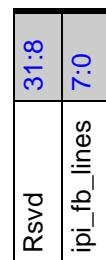


Table 5-100 Fields for Register: IPI_FB_LINES

Bits	Name	Memory Access	Description
31:8			Reserved Field: Yes
7:0	ipi_fb_lines	R/W	<p>The frame blanking period measured in number of horizontal lines. The minimum value is 0.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FB_LINES_ipi_fb_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.17 IPI_INSERT_CTRL

- **Name:** IPI insertion control.
- **Description:** Insertion field control when IPI and IDI can work simultaneously.
- **Size:** 32 bits
- **Offset:** 0x660
- **Exists:** CSI2_DEVICE_IDL_IPI_IF==1

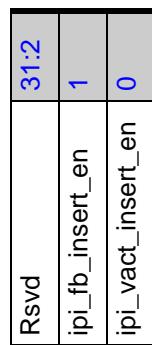


Table 5-101 Fields for Register: IPI_INSERT_CTRL

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	ipi_fb_insert_en	R/W	<p>Whether to insert IDI packet in IPI Frame Blanking field:</p> <ul style="list-style-type: none"> ■ 1: Enable ■ 0: Disable <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_INSERT_CTRL_ipi_fb_insert_en) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	ipi_vact_insert_en	R/W	<p>Whether to insert IDI packet in IPI VACT field:</p> <ul style="list-style-type: none"> ■ 1: Enable ■ 0: Disable <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_INSERT_CTRL_ipi_vact_insert_en) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.18 IPI2_PKT_CFG

- **Name:** IPI2 packet configuration.
- **Description:** This register indicates the packet configuration of IPI2.
- **Size:** 32 bits
- **Offset:** 0x680
- **Exists:** CSI2_DEVICE_IPI2_IF==1

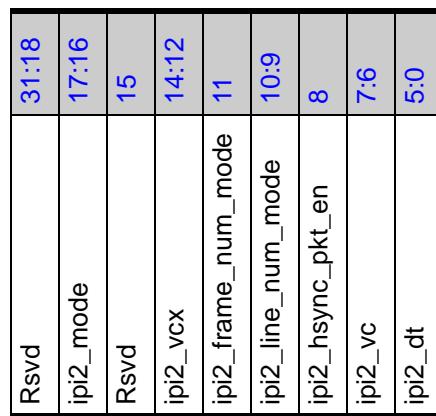


Table 5-102 Fields for Register: IPI2_PKT_CFG

Bits	Name	Memory Access	Description
31:18			Reserved Field: Yes
17:16	ipi2_mode	R/W	<p>This field is to select the IPI mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Backpressure Mode - Provides a backpressure mechanism to inform the IPI driver that the memory is almost full. In this mode, IPI timing can't be maintained. ■ 01: Cut through Mode - Initiates the HS transmission to the PHY after the configured number of ipi_clk cycles. Makes use of very shallow memory. ■ 00: Store and Forward Mode - Stores the full pixel packet before forwarding. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_PKT_CFG_ipi2_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15			Reserved Field: Yes

Bits	Name	Memory Access	Description
14:12	ipi2_vcx	R/W	<p>The Virtual Channel bit[4:2] of IPI packet</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_PKT_CFG_ipi2_vcx) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_VC_EXTENSION==1</p>
11	ipi2_frame_num_mode	R/W	<p>This field indicates the IPI2 frame number mode:</p> <ul style="list-style-type: none"> ■ 1: Frame Number Increments One mode. ■ 0: Frame Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_PKT_CFG_ipi2_frame_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10:9	ipi2_line_num_mode	R/W	<p>This field indicates IPI2 line number mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Line Number Increments Arbitrary Value mode. ■ 01: Line Number Increments One mode. ■ 00: Line Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_PKT_CFG_ipi2_line_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	ipi2_hsync_pkt_en	R/W	<p>This field indicates the line synchronization packets mode:</p> <ul style="list-style-type: none"> ■ 1: Transmit line synchronization packets. ■ 0: Don't transmit line synchronization packets. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_PKT_CFG_ipi2_hsync_pkt_en) <p>Value After Reset: 0x1</p> <p>Exists: Always</p>
7:6	ipi2_vc	R/W	<p>The Virtual Channel of IPI2 packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_PKT_CFG_ipi2_vc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5:0	ipi2_dt	R/W	<p>The Data Type of IPI2 packet.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IPI2_PKT_CFG_ipi2_dt) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.19 IPI2_PIXELS

- **Name:** IPI2 pixels.
- **Description:** IPI2 Horizontal resolution.
- **Size:** 32 bits
- **Offset:** 0x684
- **Exists:** CSI2_DEVICE_IPI2_IF==1

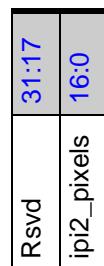


Table 5-103 Fields for Register: IPI2_PIXELS

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi2_pixels	R/W	<p>This field configures horizontal resolution for IPI2. For YUV,RGB and RAW, this field configures pixel number of data packet for IPI2; For User Defined Byte-based Data, this field configures byte number of data packet for IPI2.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_PIXELS_ipi2_pixels) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.20 IPI2_MAX_FRAME_NUM

- **Name:** IPI2 max frame number.
- **Description:** Max frame number of IPI2 frame synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x688
- **Exists:** CSI2_DEVICE_IPI2_IF==1



Table 5-104 Fields for Register: IPI2_MAX_FRAME_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi2_max_frame_num	R/W	<p>If Frame Number Increments One mode is selected, this field is to define the max frame number of IPI2 frame synchronization packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_MAX_FRAME_NUM_ipi2_max_frame_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.21 IPI2_START_LINE_NUM

- **Name:** IPI2 start line number
- **Description:** Start line number of IPI2 line synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x68c
- **Exists:** CSI2_DEVICE_IPI2_IF==1



Table 5-105 Fields for Register: IPI2_START_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi2_start_line_num	R/W	If Line Number Increments Arbitrary Value mode is selected, this field is to define the start line number of line synchronization packet. The start value must be a non-zero. Values: <ul style="list-style-type: none">■ 0x0 (IPI2_START_LINE_NUM_ipi2_start_line_num) Value After Reset: 0x0 Exists: Always

5.2.7.22 IPI2_STEP_LINE_NUM

- **Name:** IPI2 step line number
- **Description:** Step Line Number of IPI2 line synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x690
- **Exists:** CSI2_DEVICE_IPI2_IF==1



Table 5-106 Fields for Register: IPI2_STEP_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi2_step_line_num	R/W	<p>If Line Number Increments Arbitrary Value mode is selected, this field is to define the step value for Line number increments. The step value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_STEP_LINE_NUM_ipi2_step_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.23 IPI2_LINES

- **Name:** IPI2 lines.
- **Description:** Overall line for a frame for IPI2.
- **Size:** 32 bits
- **Offset:** 0x694
- **Exists:** CSI2_DEVICE_IPI2_IF==1

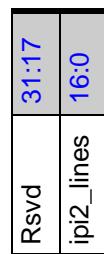


Table 5-107 Fields for Register: IPI2_LINES

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi2_lines	R/W	<p>This field configures line number of a frame for IPI2(VSA+VBP+VACT+VFP). The step value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_LINES_ipi2_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.24 IPI2_DATA_SEND_START

- **Name:** IPI2 Data Send Start.
- **Description:** IPI2 Data Send Start in Cut Through Mode.
- **Size:** 32 bits
- **Offset:** 0x698
- **Exists:** CSI2_DEVICE_IPI2_IF==1



Table 5-108 Fields for Register: IPI2_DATA_SEND_START

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi2_data_send_start	R/W	<p>This field configures the number of ipi2_clk cycles for start to send data packet after the ipi2_data_en is asserted in Cut Through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_DATA_SEND_START_ipi2_data_send_start) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.25 IPI2_FIFO_STATUS

- **Name:** IPI2 fifo status.
- **Description:** IPI2 channel fifo status.
- **Size:** 32 bits
- **Offset:** 0x69c
- **Exists:** CSI2_DEVICE_IPI2_IF==1

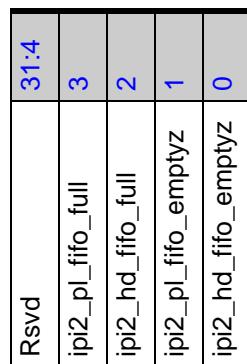


Table 5-109 Fields for Register: IPI2_FIFO_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	ipi2_pl_fifo_full	R	<p>This field indicates the payload FIFO full status of IPI2 interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_FIFO_STATUS_ipi2_pl_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	ipi2_hd_fifo_full	R	<p>This field indicates the header FIFO full status of IPI2 interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_FIFO_STATUS_ipi2_hd_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	ipi2_pl_fifo_emptyz	R	<p>This field indicates the payload FIFO empty status of IPI2 interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_FIFO_STATUS_ipi2_pl_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
0	ipi2_hd_fifo_emptyz	R	<p>This field indicates the header FIFO empty status of IPI2 interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IPI2_FIFO_STATUS_ipi2_hd_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.26 IPI3_PKT_CFG

- **Name:** IPI3 packet configuration.
- **Description:** This register indicates the packet configuration of IPI3.
- **Size:** 32 bits
- **Offset:** 0x6c0
- **Exists:** CSI2_DEVICE_IPI3_IF==1

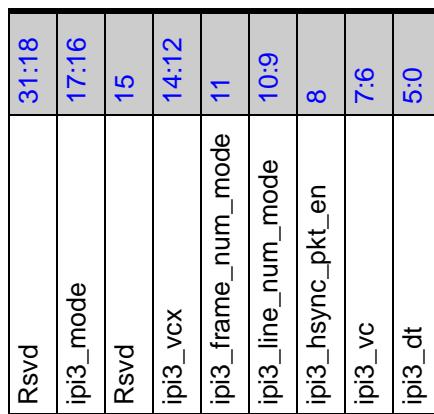


Table 5-110 Fields for Register: IPI3_PKT_CFG

Bits	Name	Memory Access	Description
31:18			Reserved Field: Yes
17:16	ipi3_mode	R/W	<p>This field is to select the IPI mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Backpressure Mode - Provides a backpressure mechanism to inform the IPI driver that the memory is almost full. In this mode, IPI timing can't be maintained. ■ 01: Cut through Mode - Initiates the HS transmission to the PHY after the configured number of ipi_clk cycles. Makes use of very shallow memory. ■ 00: Store and Forward Mode - Stores the full pixel packet before forwarding. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_PKT_CFG_ipi3_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15			Reserved Field: Yes

Bits	Name	Memory Access	Description
14:12	ipi3_vcx	R/W	<p>The Virtual Channel bit[4:2] of IPI packet</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_PKT_CFG_ipi3_vcx) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_VC_EXTENSION==1</p>
11	ipi3_frame_num_mode	R/W	<p>This field indicates the IPI3 frame number mode:</p> <ul style="list-style-type: none"> ■ 1: Frame Number Increments One mode. ■ 0: Frame Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_PKT_CFG_ipi3_frame_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10:9	ipi3_line_num_mode	R/W	<p>This field indicates IPI3 line number mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Line Number Increments Arbitrary Value mode. ■ 01: Line Number Increments One mode. ■ 00: Line Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_PKT_CFG_ipi3_line_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	ipi3_hsync_pkt_en	R/W	<p>This field indicates the line synchronization packets mode:</p> <ul style="list-style-type: none"> ■ 1: Transmit line synchronization packets. ■ 0: Don't transmit line synchronization packets. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_PKT_CFG_ipi3_hsync_pkt_en) <p>Value After Reset: 0x1</p> <p>Exists: Always</p>
7:6	ipi3_vc	R/W	<p>The Virtual Channel of IPI3 packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_PKT_CFG_ipi3_vc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5:0	ipi3_dt	R/W	<p>The Data Type of IPI3 packet.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IPI3_PKT_CFG_ipi3_dt) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.27 IPI3_PIXELS

- **Name:** IPI3 pixels.
- **Description:** IPI3 Horizontal resolution.
- **Size:** 32 bits
- **Offset:** 0x6c4
- **Exists:** CSI2_DEVICE_IPI3_IF==1

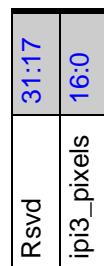


Table 5-111 Fields for Register: IPI3_PIXELS

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi3_pixels	R/W	<p>This field configures horizontal resolution for IPI3. For YUV,RGB and RAW, this field configures pixel number of data packet for IPI 3; For User Defined Byte-based Data, this field configures byte number of data packet for IPI3.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_PIXELS_ipi3_pixels) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.28 IPI3_MAX_FRAME_NUM

- **Name:** IPI3 max frame number.
- **Description:** Max frame number of IPI3 frame synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x6c8
- **Exists:** CSI2_DEVICE_IPI3_IF==1



Table 5-112 Fields for Register: IPI3_MAX_FRAME_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi3_max_frame_num	R/W	<p>If Frame Number Increments One mode is selected, this field is to define the max frame number of IPI3 frame synchronization packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_MAX_FRAME_NUM_ipi3_max_frame_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.29 IPI3_START_LINE_NUM

- **Name:** IPI3 start line number.
- **Description:** Start Line number of IPI3 line synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x6cc
- **Exists:** CSI2_DEVICE_IPI3_IF==1



Table 5-113 Fields for Register: IPI3_START_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi3_start_line_num	R/W	<p>If Line Number Increments Arbitrary Value mode is selected, this field is to define the start line number of line synchronization packet. The start value must be a non-zero.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_START_LINE_NUM_ipi3_start_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.30 IPI3_STEP_LINE_NUM

- **Name:** IPI3 Step Line Number.
- **Description:** Step Line Number of IPI3 line synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x6d0
- **Exists:** CSI2_DEVICE_IPI3_IF==1



Table 5-114 Fields for Register: IPI3_STEP_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi3_step_line_num	R/W	<p>If Line Number Increments Arbitrary Value mode is selected, this field is to define the step value for Line number increments. The step value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_STEP_LINE_NUM_ipi3_step_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.31 IPI3_LINES

- **Name:** IPI3 lines.
- **Description:** Overall line for a frame for IPI3.
- **Size:** 32 bits
- **Offset:** 0x6d4
- **Exists:** CSI2_DEVICE_IPI3_IF==1

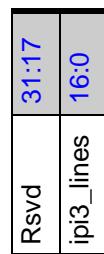


Table 5-115 Fields for Register: IPI3_LINES

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi3_lines	R/W	<p>This field configures line number of a frame for IPI3(VSA+VBP+VACT+VFP). The step value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_LINES_ipi3_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.32 IPI3_DATA_SEND_START

- **Name:** IPI3 data send start.
- **Description:** IPI3 Data Send Start in Cut through Mode.
- **Size:** 32 bits
- **Offset:** 0x6d8
- **Exists:** CSI2_DEVICE_IPI3_IF==1



Table 5-116 Fields for Register: IPI3_DATA_SEND_START

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi3_data_send_start	R/W	<p>This field configures the number of ipi3_clk cycles for start to send data packet after the ipi3_data_en is asserted in Cut Through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_DATA_SEND_START_ipi3_data_send_start) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.33 IPI3_FIFO_STATUS

- **Name:** IPI3 fifo status.
- **Description:** IPI3 channel fifo status.
- **Size:** 32 bits
- **Offset:** 0x6dc
- **Exists:** CSI2_DEVICE_IPI3_IF==1

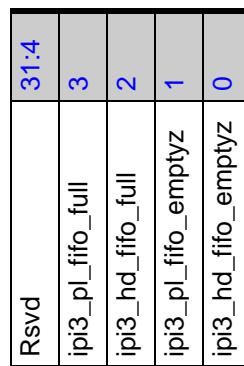


Table 5-117 Fields for Register: IPI3_FIFO_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	ipi3_pl_fifo_full	R	<p>This field indicates the payload FIFO full status of IPI3 interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFO_STATUS_ipi3_pl_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	ipi3_hd_fifo_full	R	<p>This field indicates the header FIFO full status of IPI3 interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFO_STATUS_ipi3_hd_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	ipi3_pl_fifo_emptyz	R	<p>This field indicates the payload FIFO empty status of IPI3 interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFO_STATUS_ipi3_pl_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
0	ipi3_hd_fifo_emptyz	R	<p>This field indicates the header FIFO empty status of IPI3 interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IPI3_FIFO_STATUS_ipi3_hd_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.34 IPI4_PKT_CFG

- **Name:** IPI4 packet configuration.
 - **Description:** This register indicates the packet configuration of IPI4.
 - **Size:** 32 bits
 - **Offset:** 0x700
 - **Exists:** CSI2_DEVICE_IPI4_IF==1

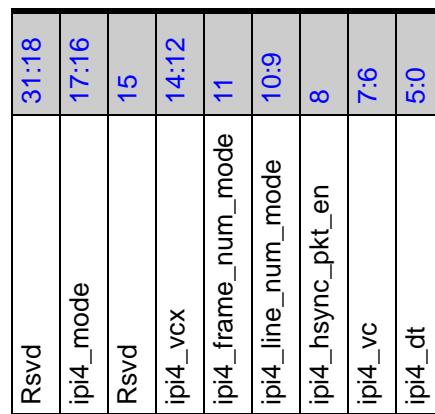


Table 5-118 Fields for Register: IPI4_PKT_CFG

Bits	Name	Memory Access	Description
31:18			Reserved Field: Yes
17:16	ipi4_mode	R/W	<p>This field is to select the IPI mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Backpressure Mode - Provides a backpressure mechanism to inform the IPI driver that the memory is almost full. In this mode, IPI timing can't be maintained. ■ 01: Cut through Mode - Initiates the HS transmission to the PHY after the configured number of ipi_clk cycles. Makes use of very shallow memory. ■ 00: Store and Forward Mode - Stores the full pixel packet before forwarding. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_PKT_CFG_ipi4_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
15			Reserved Field: Yes

Bits	Name	Memory Access	Description
14:12	ipi4_vcx	R/W	<p>The Virtual Channel bit[4:2] of IPI packet</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_PKT_CFG_ipi4_vcx) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_VC_EXTENSION==1</p>
11	ipi4_frame_num_mode	R/W	<p>This field indicates the IPI4 frame number mode:</p> <ul style="list-style-type: none"> ■ 1: Frame Number Increments One mode. ■ 0: Frame Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_PKT_CFG_ipi4_frame_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10:9	ipi4_line_num_mode	R/W	<p>This field indicates IPI4 line number mode:</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Line Number Increments Arbitrary Value mode. ■ 01: Line Number Increments One mode. ■ 00: Line Number Zero mode. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_PKT_CFG_ipi4_line_num_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	ipi4_hsync_pkt_en	R/W	<p>This field indicates the line synchronization packets mode:</p> <ul style="list-style-type: none"> ■ 1: Transmit line synchronization packets. ■ 0: Don't transmit line synchronization packets. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_PKT_CFG_ipi4_hsync_pkt_en) <p>Value After Reset: 0x1</p> <p>Exists: Always</p>
7:6	ipi4_vc	R/W	<p>The Virtual Channel of IPI4 packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_PKT_CFG_ipi4_vc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5:0	ipi4_dt	R/W	<p>The Data Type of IPI4 packet.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IPI4_PKT_CFG_ipi4_dt) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.35 IPI4_PIXELS

- **Name:** IPI4 pixels.
- **Description:** IPI4 Horizontal resolution.
- **Size:** 32 bits
- **Offset:** 0x704
- **Exists:** CSI2_DEVICE_IPI4_IF==1

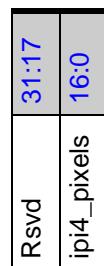


Table 5-119 Fields for Register: IPI4_PIXELS

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi4_pixels	R/W	<p>This field configures horizontal resolution for IPI4. For YUV,RGB and RAW, this field configures pixel number of data packet for IPI4; For User Defined Byte-based Data, this field configures byte number of data packet for IPI4.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_PIXELS_ipi4_pixels) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.36 IPI4_MAX_FRAME_NUM

- **Name:** IPI4 max frame number.
- **Description:** Max frame number of IPI4 frame synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x708
- **Exists:** CSI2_DEVICE_IPI4_IF==1



Table 5-120 Fields for Register: IPI4_MAX_FRAME_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi4_max_frame_num	R/W	<p>If Frame Number Increments One mode is selected, this field is to define the max frame number of IPI4 frame synchronization packet.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_MAX_FRAME_NUM_ipi4_max_frame_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.37 IPI4_START_LINE_NUM

- **Name:** IPI4 start line number.
- **Description:** Start line number of IPI4 line synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x70c
- **Exists:** CSI2_DEVICE_IPI4_IF==1



Table 5-121 Fields for Register: IPI4_START_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi4_start_line_num	R/W	<p>If Line Number Increments Arbitrary Value mode is selected, this field is to define the start line number of line synchronization packet. The start value must be a non-zero.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_START_LINE_NUM_ipi4_start_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.38 IPI4_STEP_LINE_NUM

- **Name:** IPI4 Step Line Number.
- **Description:** Step Line Number of IPI4 line synchronization packet.
- **Size:** 32 bits
- **Offset:** 0x710
- **Exists:** CSI2_DEVICE_IPI4_IF==1



Table 5-122 Fields for Register: IPI4_STEP_LINE_NUM

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:0	ipi4_step_line_num	R/W	<p>If Line Number Increments Arbitrary Value mode is selected, this field is to define the step value for Line number increments. The step value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_STEP_LINE_NUM_ipi4_step_line_num) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.39 IPI4_LINES

- **Name:** IPI4 lines.
- **Description:** Overall line for a frame for IPI4.
- **Size:** 32 bits
- **Offset:** 0x714
- **Exists:** CSI2_DEVICE_IPI4_IF==1

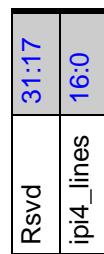


Table 5-123 Fields for Register: IPI4_LINES

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi4_lines	R/W	<p>This field configures line number of a frame for IPI4(VSA+VBP+VACT+VFP). The step value must be greater than one.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_LINES_ipi4_lines) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.40 IPI4_DATA_SEND_START

- **Name:** IPI4 data send start.
- **Description:** IPI4 Data Send Start in Cut through Mode.
- **Size:** 32 bits
- **Offset:** 0x718
- **Exists:** CSI2_DEVICE_IPI4_IF==1



Table 5-124 Fields for Register: IPI4_DATA_SEND_START

Bits	Name	Memory Access	Description
31:17			Reserved Field: Yes
16:0	ipi4_data_send_start	R/W	<p>This field configures the number of ipi4_clk cycles for start to send data packet after the ipi4_data_en is asserted in Cut Through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_DATA_SEND_START_ipi4_data_send_start) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.41 IPI4_FIFO_STATUS

- **Name:** IPI4 fifo status.
- **Description:** IPI4 channel fifo status.
- **Size:** 32 bits
- **Offset:** 0x71c
- **Exists:** CSI2_DEVICE_IPI4_IF==1

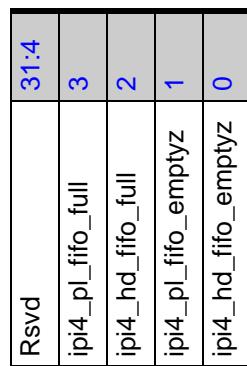


Table 5-125 Fields for Register: IPI4_FIFO_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	ipi4_pl_fifo_full	R	<p>This field indicates the payload FIFO full status of IPI4 interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_FIFO_STATUS_ipi4_pl_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	ipi4_hd_fifo_full	R	<p>This field indicates the header FIFO full status of IPI4 interface, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_FIFO_STATUS_ipi4_hd_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	ipi4_pl_fifo_emptyz	R	<p>This field indicates the payload FIFO empty status of IPI4 interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_FIFO_STATUS_ipi4_pl_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
0	ipi4_hd_fifo_emptyz	R	<p>This field indicates the header FIFO empty status of IPI4 interface, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (IPI4_FIFO_STATUS_ipi4_hd_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.42 MT_IPI_CFG

- **Name:** Mutiple IPI configuration.
- **Description:** This register indicates the configuration of Multiple IPI transmission.
- **Size:** 32 bits
- **Offset:** 0x740
- **Exists:** CSI2_DEVICE_IPI2_IF==1

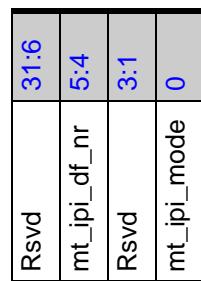


Table 5-126 Fields for Register: MT_IPI_CFG

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5:4	mt_ipi_df_nr	R/W	<p>The number of the divided field for Multiple IPI transmission</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_CFG_mt_ipi_df_nr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	mt_ipi_mode	R/W	<p>This field indicates Multiple IPI transmissionin mode.</p> <ul style="list-style-type: none"> ■ 1: Timing deliver mode ■ 0: No Timing deliver mode <p>In Timing deliver mode, all IPIs should be in Store and Forward Mode. In No Timing deliver mode, all IPIs should be in Backpressure Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_CFG_mt_ipi_mode) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.43 MT_IPI_DF_TIME

- **Name:** Multiple IPI divided field time.
- **Description:** The divided field time of Multiple IPI transmission in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x744
- **Exists:** CSI2_DEVICE_IPI2_IF==1

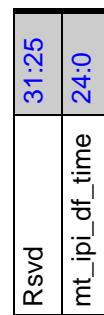


Table 5-127 Fields for Register: MT_IPI_DF_TIME

Bits	Name	Memory Access	Description
31:25			Reserved Field: Yes
24:0	mt_ipi_df_time	R/W	<p>The size of the divided field time for Multiple IPI transmission in IPI clock domain. Measured in ipi clock cycles.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_DF_TIME_mt_ipi_df_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.44 MT_IPI_FIFO_STATUS

- **Name:** Multiple IPI fifo status.
- **Description:** Multiple IPI fifo status.
- **Size:** 32 bits
- **Offset:** 0x748
- **Exists:** CSI2_DEVICE_IPI2_IF==1

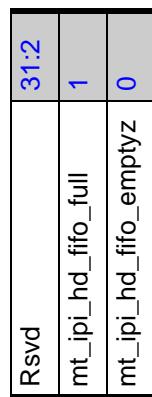


Table 5-128 Fields for Register: MT_IPI_FIFO_STATUS

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	mt_ipi_hd_fifo_full	R	<p>This field indicates the header FIFO full status of Multiple IPI, active HIGH.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_FIFO_STATUS_mt_ipi_hd_fifo_full) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mt_ipi_hd_fifo_emptyz	R	<p>This field indicates the header FIFO empty status of Multiple IPI, active LOW.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_FIFO_STATUS_mt_ipi_hd_fifo_emptyz) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.45 MT_IPI1_TRANS_CFG

- **Name:** Multiple IPI1 transmission configuration.
- **Description:** The setting of IPI1 transmission for Multiple IPI transmission.
- **Size:** 32 bits
- **Offset:** 0x74c
- **Exists:** CSI2_DEVICE_IPI2_IF==1

Rsvd	31:15
mt_ipi1_field4_cfg	14:12
Rsvd	11
mt_ipi1_field3_cfg	10:8
Rsvd	7
mt_ipi1_field2_cfg	6:4
Rsvd	3
mt_ipi1_field1_cfg	2:0

Table 5-129 Fields for Register: MT_IPI1_TRANS_CFG

Bits	Name	Memory Access	Description
31:15			Reserved Field: Yes
14:12	mt_ipi1_field4_cfg	R/W	<p>Control the IPI1 transmission order in the field4.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI1_TRANS_CFG_mt_ipi1_field4_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11			Reserved Field: Yes

Bits	Name	Memory Access	Description
10:8	mt_ipi1_field3_cfg	R/W	<p>Control the IPI1 transmission order in the field3.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI1_TRANS_CFG_mt_ipi1_field3_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7			Reserved Field: Yes
6:4	mt_ipi1_field2_cfg	R/W	<p>Control the IPI1 transmission order in the field2.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI1_TRANS_CFG_mt_ipi1_field2_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3			Reserved Field: Yes
2:0	mt_ipi1_field1_cfg	R/W	<p>Control the IPI1 transmission order in the field1.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI1_TRANS_CFG_mt_ipi1_field1_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.46 MT_IPI2_TRANS_CFG

- **Name:** Multiple IPI2 transmission configuration.
- **Description:** The setting of IPI2 transmission for Multiple IPI transmission.
- **Size:** 32 bits
- **Offset:** 0x750
- **Exists:** CSI2_DEVICE_IPI2_IF==1

Rsvd	31:15
mt_ipi2_field4_cfg	14:12
Rsvd	11
mt_ipi2_field3_cfg	10:8
Rsvd	7
mt_ipi2_field2_cfg	6:4
Rsvd	3
mt_ipi2_field1_cfg	2:0

Table 5-130 Fields for Register: MT_IPI2_TRANS_CFG

Bits	Name	Memory Access	Description
31:15			Reserved Field: Yes
14:12	mt_ipi2_field4_cfg	R/W	<p>Control the IPI2 transmission order in the field4.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI2_TRANS_CFG_mt_ipi2_field4_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11			Reserved Field: Yes

Bits	Name	Memory Access	Description
10:8	mt_ipi2_field3_cfg	R/W	<p>Control the IPI2 transmission order in the field3.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI2_TRANS_CFG_mt_ipi2_field3_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7			Reserved Field: Yes
6:4	mt_ipi2_field2_cfg	R/W	<p>Control the IPI2 transmission order in the field2.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI2_TRANS_CFG_mt_ipi2_field2_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3			Reserved Field: Yes
2:0	mt_ipi2_field1_cfg	R/W	<p>Control the IPI2 transmission order in the field1.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI2_TRANS_CFG_mt_ipi2_field1_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.47 MT_IPI3_TRANS_CFG

- **Name:** Multiple IPI3 transmission configuration.
- **Description:** The setting of IPI3 transmission for Multiple IPI transmission.
- **Size:** 32 bits
- **Offset:** 0x754
- **Exists:** CSI2_DEVICE_IPI3_IF==1

Rsvd	31:15
mt_ipi3_field4_cfg	14:12
Rsvd	11
mt_ipi3_field3_cfg	10:8
Rsvd	7
mt_ipi3_field2_cfg	6:4
Rsvd	3
mt_ipi3_field1_cfg	2:0

Table 5-131 Fields for Register: MT_IPI3_TRANS_CFG

Bits	Name	Memory Access	Description
31:15			Reserved Field: Yes
14:12	mt_ipi3_field4_cfg	R/W	<p>Control the IPI3 transmission order in the field4.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI3_TRANS_CFG_mt_ipi3_field4_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11			Reserved Field: Yes

Bits	Name	Memory Access	Description
10:8	mt_ipi3_field3_cfg	R/W	<p>Control the IPI3 transmission order in the field3.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI3_TRANS_CFG_mt_ipi3_field3_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7			Reserved Field: Yes
6:4	mt_ipi3_field2_cfg	R/W	<p>Control the IPI3 transmission order in the field2.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI3_TRANS_CFG_mt_ipi3_field2_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3			Reserved Field: Yes
2:0	mt_ipi3_field1_cfg	R/W	<p>Control the IPI3 transmission order in the field1.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI3_TRANS_CFG_mt_ipi3_field1_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.48 MT_IPI4_TRANS_CFG

- **Name:** Multiple IPI4 transmission configuration.
- **Description:** The setting of IPI4 transmission for Multiple IPI transmission.
- **Size:** 32 bits
- **Offset:** 0x758
- **Exists:** CSI2_DEVICE_IPI4_IF==1

Rsvd	31:15
mt_ipi4_field4_cfg	14:12
Rsvd	11
mt_ipi4_field3_cfg	10:8
Rsvd	7
mt_ipi4_field2_cfg	6:4
Rsvd	3
mt_ipi4_field1_cfg	2:0

Table 5-132 Fields for Register: MT_IPI4_TRANS_CFG

Bits	Name	Memory Access	Description
31:15			Reserved Field: Yes
14:12	mt_ipi4_field4_cfg	R/W	<p>Control the IPI4 transmission order in the field4.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI4_TRANS_CFG_mt_ipi4_field4_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11			Reserved Field: Yes

Bits	Name	Memory Access	Description
10:8	mt_ipi4_field3_cfg	R/W	<p>Control the IPI4 transmission order in the field3.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI4_TRANS_CFG_mt_ipi4_field3_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7			Reserved Field: Yes
6:4	mt_ipi4_field2_cfg	R/W	<p>Control the IPI4 transmission order in the field2.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI4_TRANS_CFG_mt_ipi4_field2_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3			Reserved Field: Yes
2:0	mt_ipi4_field1_cfg	R/W	<p>Control the IPI4 transmission order in the field1.</p> <ul style="list-style-type: none"> ■ 3'b000: No transmission. ■ 3'b001: The 1st transmission. ■ 3'b010: The 2nd transmission. ■ 3'b011: The 3rd transmission. ■ 3'b100: The 4th transmission. ■ Others: Reserved. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI4_TRANS_CFG_mt_ipi4_field1_cfg) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.49 IPI1_HSA_HBP_TIME

- **Name:** IPI1 HSA and HBP time.
- **Description:** IPI1 Horizontal Synchronism Active(HSA) and Horizontal Back Porch(HBP) time in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x75c
- **Exists:** CSI2_DEVICE_IPI2_IF==1



Table 5-133 Fields for Register: IPI1_HSA_HBP_TIME

Bits	Name	Memory Access	Description
31:19			Reserved Field: Yes
18:0	ipi1_hsa_hbp_time	R/W	<p>This bit configures the total time of horizontal synchronism active period and horizontal back porch period in IPI clock domain (HSA+HBP). The minimum value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI1_HSA_HBP_TIME_ipi1_hsa_hbp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.50 IPI1_LP_TIME

- **Name:** IPI1 data packet transmission time.
- **Description:** IPI1 Data packet transmission time in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x760
- **Exists:** CSI2_DEVICE_IPI2_IF==1

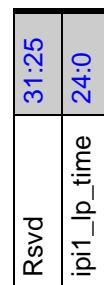


Table 5-134 Fields for Register: IPI1_LP_TIME

Bits	Name	Memory Access	Description
31:25			Reserved Field: Yes
24:0	ipi1_lp_time	R/W	<p>Data packet transmission time in IPI clock domain. The configured value should be greater than actual IPI input Data packet transmission time in IPI clock domain.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI1_LP_TIME_ipi1_lp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.51 IPI2_HSA_HBP_TIME

- **Name:** IPI2 HSA and HBP time.
- **Description:** IPI2 Horizontal Synchronism Active(HSA) and Horizontal Back Porch(HBP) time in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x764
- **Exists:** CSI2_DEVICE_IPI2_IF==1



Table 5-135 Fields for Register: IPI2_HSA_HBP_TIME

Bits	Name	Memory Access	Description
31:19			Reserved Field: Yes
18:0	ipi2_hsa_hbp_time	R/W	<p>This bit configures the total time of horizontal synchronism active period and horizontal back porch period in IPI clock domain (HSA+HBP). The minimum value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_HSA_HBP_TIME_ipi2_hsa_hbp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.52 IPI2_LP_TIME

- **Name:** IPI2 data packet transmission time.
- **Description:** IPI2 Data packet transmission time in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x768
- **Exists:** CSI2_DEVICE_IPI2_IF==1

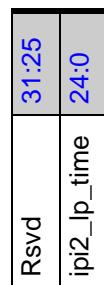


Table 5-136 Fields for Register: IPI2_LP_TIME

Bits	Name	Memory Access	Description
31:25			Reserved Field: Yes
24:0	ipi2_lp_time	R/W	<p>Data packet transmission time in IPI clock domain. The configured value should be greater than actual IPI input Data packet transmission time in IPI clock domain.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_LP_TIME_ipi2_lp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.53 IPI3_HSA_HBP_TIME

- **Name:** IPI3 HSA and HBP time.
- **Description:** IPI3 Horizontal Synchronism Active(HSA) and Horizontal Back Porch(HBP) time in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x76c
- **Exists:** CSI2_DEVICE_IPI3_IF==1



Table 5-137 Fields for Register: IPI3_HSA_HBP_TIME

Bits	Name	Memory Access	Description
31:19			Reserved Field: Yes
18:0	ipi3_hsa_hbp_time	R/W	<p>This bit configures the total time of horizontal synchronism active period and horizontal back porch period in IPI clock domain (HSA+HBP). The minimum value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_HSA_HBP_TIME_ipi3_hsa_hbp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.54 IPI3_LP_TIME

- **Name:** IPI3 data packet transmission time.
- **Description:** IPI3 Data packet transmission time in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x770
- **Exists:** CSI2_DEVICE_IPI3_IF==1

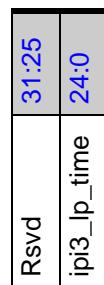


Table 5-138 Fields for Register: IPI3_LP_TIME

Bits	Name	Memory Access	Description
31:25			Reserved Field: Yes
24:0	ipi3_lp_time	R/W	<p>Data packet transmission time in IPI clock domain. The configured value should be greater than actual IPI input Data packet transmission time in IPI clock domain.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_LP_TIME_ipi3_lp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.55 IPI4_HSA_HBP_TIME

- **Name:** IPI4 HSA and HBP time.
- **Description:** IPI4 Horizontal Synchronism Active(HSA) and Horizontal Back Porch(HBP) time in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x774
- **Exists:** CSI2_DEVICE_IPI4_IF==1



Table 5-139 Fields for Register: IPI4_HSA_HBP_TIME

Bits	Name	Memory Access	Description
31:19			Reserved Field: Yes
18:0	ipi4_hsa_hbp_time	R/W	<p>This bit configures the total time of horizontal synchronism active period and horizontal back porch period in IPI clock domain (HSA+HBP). The minimum value is 1.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_HSA_HBP_TIME_ipi4_hsa_hbp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.7.56 IPI4_LP_TIME

- **Name:** IPI4 data packet transmission time.
- **Description:** IPI4 Data packet transmission time in IPI clock domain.
- **Size:** 32 bits
- **Offset:** 0x778
- **Exists:** CSI2_DEVICE_IPI4_IF==1

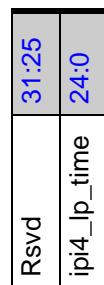


Table 5-140 Fields for Register: IPI4_LP_TIME

Bits	Name	Memory Access	Description
31:25			Reserved Field: Yes
24:0	ipi4_lp_time	R/W	<p>Data packet transmission time in IPI clock domain. The configured value should be greater than actual IPI input Data packet transmission time in IPI clock domain.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_LP_TIME_ipi4_lp_time) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8 DWC_mipicsi2_device_MemMap/AP_INT Registers

5.2.8.1 INT_ST_DIAG_MAIN

- **Name:** Interrupt status diagnosis main.
- **Description:** Clear on read register. Contains the status of individual internal diagnosis interrupt sources group, regardless of the contents of the associated interrupt mask registers, so it is possible to service the interrupt status registers by polling. Reading this register clears it and deasserts the interrupt pin.
- **Size:** 32 bits
- **Offset:** 0x800
- **Exists:** CSI2_DEVICE_AP==1

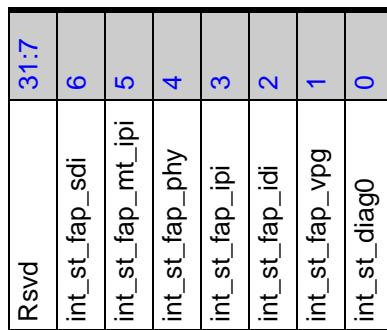


Table 5-141 Fields for Register: INT_ST_DIAG_MAIN

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	int_st_fap_sdi	RC	Status of INT_ST_FAP_SDI Values: <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG_MAIN_int_st_fap_sdi) Value After Reset: 0x0 Exists: CSI2_DEVICE_SD1_IF==1
5	int_st_fap_mt_ipi	RC	Status of INT_ST_FAP_MT_IPI interrupt source group: <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive Values: <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG_MAIN_int_st_fap_mt_ipi) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI2_IF==1

Bits	Name	Memory Access	Description
4	int_st_fap_phy	RC	<p>Status of INT_ST_FAP_PHY interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG_MAIN_int_st_fap_phy) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	int_st_fap_ipi	RC	<p>Status of INT_ST_FAP_IPI interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG_MAIN_int_st_fap_ipi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>
2	int_st_fap_idi	RC	<p>Status of INT_ST_FAP_IDI</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG_MAIN_int_st_fap_idi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>
1	int_st_fap_vpg	RC	<p>Status of INT_ST_FAP_VPG interrupt source group:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG_MAIN_int_st_fap_vpg) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_VPG==1</p>
0	int_st_diag0	RC	<p>Status of Internal Diagnosis interrupt source group 0:</p> <ul style="list-style-type: none"> ■ 1: Interrupt source group is active. ■ 0: Interrupt source group is inactive. <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG_MAIN_int_st_diag0) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.2 INT_ST_DIAG0

- **Name:** Interrupt status diagnosis group 0.
- **Description:** Interrupt group caused by Internal Diagnosis group 0. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x804
- **Exists:** CSI2_DEVICE_AP==1

Rsvd	31:21
int_st_diag_sdi	20
int_st_diag_ipi4_ch_fifo_control	19
int_st_diag_ipi4	18
int_st_diag_ipi3_ch_fifo_control	17
int_st_diag_ipi3	16
int_st_diag_ipi2_ch_fifo_control	15
int_st_diag_ipi2	14
int_st_diag_mt_ipi_control	13
int_st_diag_cmu	12
int_st_diag_ecf	11
int_st_diag_pkt_if	10
int_st_diag_sync	9
int_st_diag_err_handler	8
int_st_diag_pkt_builder	7
int_st_diag_idi_ch_fifo_control	6
int_st_diag_ipi_ch_fifo_control	5
int_st_diag_reg_bank	4
int_st_diag_phy_if_ctrl	3
int_st_diag_ambaapbintf	2
int_st_diag_idi	1
int_st_diag_ipi	0

Table 5-142 Fields for Register: INT_ST_DIAG0

Bits	Name	Memory Access	Description
31:21			Reserved Field: Yes
20	int_st_diag_sdi	RC	Internal Diagnosis error for SD Values: <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_sdi) Value After Reset: 0x0 Exists: CSI2_DEVICE_SDIF==1
19	int_st_diag_ipi4_ch_fifo_control	RC	Internal Diagnosis error for IPI4 Channel FIFO control Values: <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ipi4_ch_fifo_control) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1
18	int_st_diag_ipi4	RC	Internal Diagnosis error for IPI4 interface Values: <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ipi4) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1

Bits	Name	Memory Access	Description
17	int_st_diag_ipi3_ch_fifo_control	RC	<p>Internal Diagnosis error for IPI3 Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ipi3_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	int_st_diag_ipi3	RC	<p>Internal Diagnosis error for IPI3 interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ipi3) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15	int_st_diag_ipi2_ch_fifo_control	RC	<p>Internal Diagnosis error for IPI2 Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ipi2_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
14	int_st_diag_ipi2	RC	<p>Internal Diagnosis error for IPI2 interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ipi2) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
13	int_st_diag_mt_ipi_control	RC	<p>Internal Diagnosis error for Multiple IPI Contrille</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_mt_ipi_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
12	int_st_diag_cmu	RC	<p>Internal Diagnosis error for Clock Manager Unit</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_cmu) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	int_st_diag_ecf	RC	<p>Internal Diagnosis error for ERR correction functional block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ecf) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
10	int_st_diag_pkt_if	RC	<p>Internal Diagnosis error for packet interface block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_pkt_if) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	int_st_diag_sync	RC	<p>Internal Diagnosis error for synchronizer</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_sync) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	int_st_diag_err_handler	RC	<p>Internal Diagnosis error for Error handler</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_err_handler) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	int_st_diag_pkt_builder	RC	<p>Internal Diagnosis error for Packet builder</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_pkt_builder) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	int_st_diag_idi_ch_fifo_control	RC	<p>Internal Diagnosis error for IDI Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_idi_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>
5	int_st_diag_ipi_ch_fifo_control	RC	<p>Internal Diagnosis error for IPI Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ipi_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>
4	int_st_diag_reg_bank	RC	<p>Internal Diagnosis error for Register Bank</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_reg_bank) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
3	int_st_diag_phy_if_ctrl	RC	<p>Internal Diagnosis error for PHY interface control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_phy_if_ctrl) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	int_st_diag_ambaapbintf	RC	<p>Internal Diagnosis error for APB interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ambaapbintf) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	int_st_diag_idi	RC	<p>Internal Diagnosis error for IDI interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_idi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>
0	int_st_diag_ipi	RC	<p>Internal Diagnosis error for IPI interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_DIAG0_int_st_diag_ipi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>

5.2.8.3 INT_ST_FAP_VPG

- **Name:** VPG functional AP interrupt.
- **Description:** Functional AP interrupt group caused by the Video pattern generator.
- **Size:** 32 bits
- **Offset:** 0x808
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_VPG==1

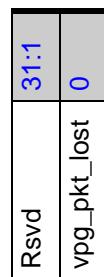


Table 5-143 Fields for Register: INT_ST_FAP_VPG

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	vpg_pkt_lost	RC	<p>Packet lost of video pattern generator.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_VPG_vpg_pkt_lost) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.4 INT_ST_FAP_IDI

- **Name:** IDI functional AP interrupt.
- **Description:** Functional AP interrupt group caused by the IDI interface. Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x80c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

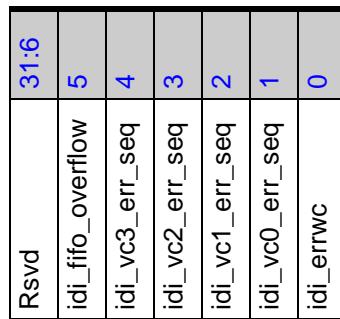


Table 5-144 Fields for Register: INT_ST_FAP_IDI

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	idi_fifo_overflow	RC	<p>The IDI Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_idi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	idi_vc3_err_seq	RC	<p>Indicator for sequence error in virtual channel 3:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_idi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
3	idi_vc2_err_seq	RC	<p>Indicator for sequence error in virtual channel 2:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_idi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	idi_vc1_err_seq	RC	<p>Indicator for sequence error in virtual channel 1:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_idi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	idi_vc0_err_seq	RC	<p>Indicator for sequence error in virtual channel 0:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_idi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	idi_errwc	RC	<p>Indicator for inconsistent between IDI Word Count and packet payload length.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_idi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.5 INT_ST_FAP_IPI

- **Name:** IPI functional AP interrupt.
 - **Description:** Functional AP interrupt group caused by the IPI interface. Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
 - **Size:** 32 bits
 - **Offset:** 0x810
 - **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

Rsvd	31:28
ipi4_fifo_underflow	27
ipi4_errline	26
ipi4_fifo_overflow	25
ipi4_errpixel	24
Rsvd	23:20
ipi3_fifo_underflow	19
ipi3_errline	18
ipi3_fifo_overflow	17
ipi3_errpixel	16
Rsvd	15:12
ipi2_fifo_underflow	11
ipi2_errline	10
ipi2_fifo_overflow	9
ipi2_errpixel	8
Rsvd	7:5
ipi_trans_conflict	4
ipi_fifo_underflow	3
ipi_errline	2
ipi_fifo_overflow	1
ipi_errpixel	0

Table 5-145 Fields for Register: INT_ST_FAP_IPI

Bits	Name	Memory Access	Description
31:28			Reserved Field: Yes
27	ipi4_fifo_underflow	RC	<p>The IPI4 Payload FIFO has lost information because read data when it was already empty. This is used for Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi4_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
26	ipi4_errline	RC	<p>Indicator for inconsistent between IPI4 line configuration and the number of input lines.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi4_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
25	ipi4_fifo_overflow	RC	<p>The IPI4 Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi4_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>

Bits	Name	Memory Access	Description
24	ipi4_errpixel	RC	<p>Indicator for inconsistent between IPI4 Pixel configuration and the number of input pixels or inconsistency between ipi4_embedded_wc and the embedded data received.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi4_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
23:20			Reserved Field: Yes
19	ipi3_fifo_underflow	RC	<p>The IPI3 Payload FIFO has lost information because read data when it was already empty. This is used for Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi3_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
18	ipi3_errline	RC	<p>Indicator for inconsistent between IPI3 line configuration and the number of input lines.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi3_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
17	ipi3_fifo_overflow	RC	<p>The IPI3 Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi3_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	ipi3_errpixel	RC	<p>Indicator for inconsistent between IPI3 Pixel configuration and the number of input pixels or inconsistency between ipi3_embedded_wc and the embedded data received.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi3_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15:12			Reserved Field: Yes

Bits	Name	Memory Access	Description
11	ipi2_fifo_underflow	RC	<p>The IPI2 Payload FIFO has lost information because read data when it was already empty. This is used for Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi2_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
10	ipi2_errline	RC	<p>Indicator for inconsistent between IPI2 line configuration and the number of input lines.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi2_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
9	ipi2_fifo_overflow	RC	<p>The IPI2 Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi2_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
8	ipi2_errpixel	RC	<p>Indicator for inconsistent between IPI2 Pixel configuration and the number of input pixels or inconsistency between ipi2_embedded_wc and the embedded data received.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi2_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
7:5			Reserved Field: Yes
4	ipi_trans_conflict	RC	<p>This bit indicates that IPI packet transmission is in conflict with IDI interface.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi_trans_conflict) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IPI_IF==1</p>
3	ipi_fifo_underflow	RC	<p>The IPI Payload FIFO has lost information because read data when it was already empty. This is used for Cut through Mode.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	ipi_errline	RC	<p>Indicator for inconsistent between IPI line configuration and the number of input lines</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi_errline) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	ipi_fifo_overflow	RC	<p>The IPI Header FIFO or Payload FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	ipi_errpixel	RC	<p>Indicator for inconsistent between IPI Pixel configuration and the number of input pixels or inconsistency between ipi_embedded_wc and the embedded data received</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IPI_ipi_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.6 INT_ST_FAP_PHY

- **Name:** PHY functional AP interrupt.
- **Description:** Functional AP interrupt group caused by the PHY.Groups and notifies which interrupt bits caused the interruption.Stores the source of the error.Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x814
- **Exists:** CSI2_DEVICE_AP==1

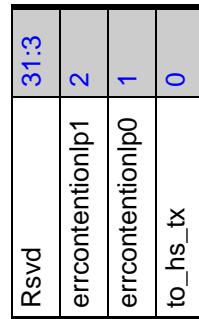


Table 5-146 Fields for Register: INT_ST_FAP_PHY

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	errcontentionlp1	RC	<p>This bit indicates LP1 contention error ErrContentionLP1 from Lane 0.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_PHY_errcontentionlp1) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	errcontentionlp0	RC	<p>This bit indicates LP0 contention error ErrContentionLP0 from Lane 0.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_PHY_errcontentionlp0) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	to_hs_tx	RC	<p>This bit indicates that the high-speed transmission timeout counter reached the end and contention has been detected.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_PHY_to_hs_tx) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.7 INT_ST_FAP_IDI_VCX

- **Name:** IDI VCX functional AP interrupt.
 - **Description:** Functional AP interrupt group caused by the IDI interface for virtual channel extension. (vc4~vc15)Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
 - **Size:** 32 bits
 - **Offset:** 0x818
 - **Exists:** CSI2 DEVICE AP==1&&CSI2 DEVICE IDI IF==1&&CSI2 DEVICE VC EXTENSION==1

Rsvd		31:12
idi_vc15_err_seq	11	
idi_vc14_err_seq	10	
idi_vc13_err_seq	9	
idi_vc12_err_seq	8	
idi_vc11_err_seq	7	
idi_vc10_err_seq	6	
idi_vc9_err_seq	5	
idi_vc8_err_seq	4	
idi_vc7_err_seq	3	
idi_vc6_err_seq	2	
idi_vc5_err_seq	1	
idi_vc4_err_seq	0	

Table 5-147 Fields for Register: INT_ST_FAP_IDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	idi_vc15_err_seq	RC	<p>Indicator for sequence error in virtual channel 15:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc15_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
10	idi_vc14_err_seq	RC	<p>Indicator for sequence error in virtual channel 14:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc14_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	idi_vc13_err_seq	RC	<p>Indicator for sequence error in virtual channel 13:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc13_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	idi_vc12_err_seq	RC	<p>Indicator for sequence error in virtual channel 12:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7	idi_vc11_err_seq	RC	<p>Indicator for sequence error in virtual channel 11:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	idi_vc10_err_seq	RC	<p>Indicator for sequence error in virtual channel 10:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	idi_vc9_err_seq	RC	<p>Indicator for sequence error in virtual channel 9:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
4	idi_vc8_err_seq	RC	<p>Indicator for sequence error in virtual channel 8:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	idi_vc7_err_seq	RC	<p>Indicator for sequence error in virtual channel 7:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	idi_vc6_err_seq	RC	<p>Indicator for sequence error in virtual channel 6:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	idi_vc5_err_seq	RC	<p>Indicator for sequence error in virtual channel 5:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	idi_vc4_err_seq	RC	<p>Indicator for sequence error in virtual channel 4:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX_idi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.8 INT_ST_FAP_IDI_VCX2

- **Name:** IDI VCX2 functional AP interrupt.
- **Description:** Functional AP interrupt group caused by the IDI interface for virtual channel extension. (vc16~vc31)Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x81c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

31:16	Rsvd	15	idi_vc31_err_seq	14	idi_vc30_err_seq	13	idi_vc29_err_seq	12	idi_vc28_err_seq	11	idi_vc27_err_seq	10	idi_vc26_err_seq	9	idi_vc25_err_seq	8	idi_vc24_err_seq	7	idi_vc23_err_seq	6	idi_vc22_err_seq	5	idi_vc21_err_seq	4	idi_vc20_err_seq	3	idi_vc19_err_seq	2	idi_vc18_err_seq	1	idi_vc17_err_seq	0	idi_vc16_err_seq
-------	------	----	------------------	----	------------------	----	------------------	----	------------------	----	------------------	----	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------	---	------------------

Table 5-148 Fields for Register: INT_ST_FAP_IDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	idi_vc31_err_seq	RC	<p>Indicator for sequence error in virtual channel 31:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc31_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
14	idi_vc30_err_seq	RC	<p>Indicator for sequence error in virtual channel 30:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc30_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
13	idi_vc29_err_seq	RC	<p>Indicator for sequence error in virtual channel 29:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc29_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
12	idi_vc28_err_seq	RC	<p>Indicator for sequence error in virtual channel 28:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
11	idi_vc27_err_seq	RC	<p>Indicator for sequence error in virtual channel 27:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	idi_vc26_err_seq	RC	<p>Indicator for sequence error in virtual channel 26:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	idi_vc25_err_seq	RC	<p>Indicator for sequence error in virtual channel 25:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
8	idi_vc24_err_seq	RC	<p>Indicator for sequence error in virtual channel 24:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	idi_vc23_err_seq	RC	<p>Indicator for sequence error in virtual channel 23:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	idi_vc22_err_seq	RC	<p>Indicator for sequence error in virtual channel 22:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	idi_vc21_err_seq	RC	<p>Indicator for sequence error in virtual channel 21:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	idi_vc20_err_seq	RC	<p>Indicator for sequence error in virtual channel 20:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	idi_vc19_err_seq	RC	<p>Indicator for sequence error in virtual channel 19:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	idi_vc18_err_seq	RC	<p>Indicator for sequence error in virtual channel 18:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	idi_vc17_err_seq	RC	<p>Indicator for sequence error in virtual channel 17:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	idi_vc16_err_seq	RC	<p>Indicator for sequence error in virtual channel 16:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_IDI_VCX2_idi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.9 INT_ST_FAP_MT_IPI

- **Name:** Multiple IPI functional AP interrupt.
- **Description:** Functional AP interrupt group caused by the Multiple IPI transmission.
- **Size:** 32 bits
- **Offset:** 0x820
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1



Table 5-149 Fields for Register: INT_ST_FAP_MT_IPI

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	mt_ipi_fifo_overflow	RC	<p>The MT IPI Header FIFO has lost information because data arrived when it was already full.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_MT_IPI_mt_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.10 INT_ST_FAP_SDI

- **Name:** SDI functional AP interrupt.
- **Description:** Functional AP interrupt group caused by the SDI interface. Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x824
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==1

Rsvd	31:7	sdi_pld_err	6	sdi_hdr_err	5	sdi_vc3_err_seq	4	sdi_vc2_err_seq	3	sdi_vc1_err_seq	2	sdi_vc0_err_seq	1	sdi_errwc	0
------	------	-------------	---	-------------	---	-----------------	---	-----------------	---	-----------------	---	-----------------	---	-----------	---

Table 5-150 Fields for Register: INT_ST_FAP_SDI

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	sdi_pld_err	RC	<p>Indicator for sdi payload error in PF_CRC Bypass mode. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_sdi_pld_err) <p>Value After Reset: 0x0 Exists: Always</p>
5	sdi_hdr_err	RC	<p>Indicator for sdi header error in ECC/PH-CRC Bypass mode. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_sdi_hdr_err) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
4	sdi_vc3_err_seq	RC	<p>Indicator for sequence error in virtual channel 3:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_sdi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	sdi_vc2_err_seq	RC	<p>Indicator for sequence error in virtual channel 2:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_sdi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	sdi_vc1_err_seq	RC	<p>Indicator for sequence error in virtual channel 1:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_sdi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	sdi_vc0_err_seq	RC	<p>Indicator for sequence error in virtual channel 0:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_sdi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	sdi_errwc	RC	<p>Indicator for inconsistent between SDI Word Count and packet payload length.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_sdi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.11 INT_ST_FAP_SD1_VCX

- **Name:** SDI VCX functional AP interrupt.
- **Description:** Functional AP interrupt group caused by the SDI interface for virtual channel extension. (vc4~vc15)Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x828
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SD1_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

Rsvd	31:12	sd1_vc15_err_seq	11	sd1_vc14_err_seq	10	sd1_vc13_err_seq	9	sd1_vc12_err_seq	8	sd1_vc11_err_seq	7	sd1_vc10_err_seq	6	sd1_vc9_err_seq	5	sd1_vc8_err_seq	4	sd1_vc7_err_seq	3	sd1_vc6_err_seq	2	sd1_vc5_err_seq	1	sd1_vc4_err_seq	0
------	-------	------------------	----	------------------	----	------------------	---	------------------	---	------------------	---	------------------	---	-----------------	---	-----------------	---	-----------------	---	-----------------	---	-----------------	---	-----------------	---

Table 5-151 Fields for Register: INT_ST_FAP_SD1_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	sd1_vc15_err_seq	RC	<p>Indicator for sequence error in virtual channel 15:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SD1_VCX_sd1_vc15_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
10	sdi_vc14_err_seq	RC	<p>Indicator for sequence error in virtual channel 14:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc14_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	sdi_vc13_err_seq	RC	<p>Indicator for sequence error in virtual channel 13:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc13_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	sdi_vc12_err_seq	RC	<p>Indicator for sequence error in virtual channel 12:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7	sdi_vc11_err_seq	RC	<p>Indicator for sequence error in virtual channel 11:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	sdi_vc10_err_seq	RC	<p>Indicator for sequence error in virtual channel 10:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	sdi_vc9_err_seq	RC	<p>Indicator for sequence error in virtual channel 9:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
4	sdi_vc8_err_seq	RC	<p>Indicator for sequence error in virtual channel 8:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	sdi_vc7_err_seq	RC	<p>Indicator for sequence error in virtual channel 7:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	sdi_vc6_err_seq	RC	<p>Indicator for sequence error in virtual channel 6:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	sdi_vc5_err_seq	RC	<p>Indicator for sequence error in virtual channel 5:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	sdi_vc4_err_seq	RC	<p>Indicator for sequence error in virtual channel 4:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet ■ Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX_sdi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.12 INT_ST_FAP_SDI_VCX2

- **Name:** SDI VCX2 functional AP interrupt.
- **Description:** Functional AP interrupt group caused by the SDI interface for virtual channel extension. (vc16~vc31)Groups and notifies which interrupt bits caused the interruption. Stores the source of the error. Reading this register clears it.
- **Size:** 32 bits
- **Offset:** 0x82c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

Rsvd	31:16
sdi_vc31_errf_seq	15
sdi_vc30_errf_seq	14
sdi_vc29_errf_seq	13
sdi_vc28_errf_seq	12
sdi_vc27_errf_seq	11
sdi_vc26_errf_seq	10
sdi_vc25_errf_seq	9
sdi_vc24_errf_seq	8
sdi_vc23_errf_seq	7
sdi_vc22_errf_seq	6
sdi_vc21_errf_seq	5
sdi_vc20_errf_seq	4
sdi_vc19_errf_seq	3
sdi_vc18_errf_seq	2
sdi_vc17_errf_seq	1
sdi_vc16_errf_seq	0

Table 5-152 Fields for Register: INT_ST_FAP_SDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	sdi_vc31_errf_seq	RC	<p>Indicator for sequence error in virtual channel 31:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc31_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
14	sdi_vc30_errf_seq	RC	<p>Indicator for sequence error in virtual channel 30:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc30_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
13	sdi_vc29_errf_seq	RC	<p>Indicator for sequence error in virtual channel 29:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc29_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
12	sdi_vc28_errf_seq	RC	<p>Indicator for sequence error in virtual channel 28:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc28_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
11	sdi_vc27_errf_seq	RC	<p>Indicator for sequence error in virtual channel 27:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc27_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	sdi_vc26_errf_seq	RC	<p>Indicator for sequence error in virtual channel 26:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc26_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	sdi_vc25_errf_seq	RC	<p>Indicator for sequence error in virtual channel 25:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc25_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
8	sdi_vc24_errf_seq	RC	<p>Indicator for sequence error in virtual channel 24:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc24_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	sdi_vc23_errf_seq	RC	<p>Indicator for sequence error in virtual channel 23:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc23_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	sdi_vc22_errf_seq	RC	<p>Indicator for sequence error in virtual channel 22:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc22_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	sdi_vc21_errf_seq	RC	<p>Indicator for sequence error in virtual channel 21:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc21_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	sdi_vc20_errf_seq	RC	<p>Indicator for sequence error in virtual channel 20:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc20_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	sdi_vc19_errf_seq	RC	<p>Indicator for sequence error in virtual channel 19:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc19_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	sdi_vc18_errf_seq	RC	<p>Indicator for sequence error in virtual channel 18:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc18_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	sdi_vc17_errf_seq	RC	<p>Indicator for sequence error in virtual channel 17:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc17_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	sdi_vc16_errf_seq	RC	<p>Indicator for sequence error in virtual channel 16:</p> <ul style="list-style-type: none"> ■ Frame Sequence Incorrect ■ The Frame Number of a Frame End mismatch the Frame Number from previous Frame Start packet - Line Sequence Incorrect ■ The Line Number of current Line End mismatch the Line Number from previous Line Start packet <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_ST_FAP_SDI_VCX2_sdi_vc16_errf_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.13 INT_MASK_N_DIAG0

- **Name:** Interrupt status diagnosis group 0 mask.
- **Description:** Interrupt mask for INT_ST_DIAG0; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x860
- **Exists:** CSI2_DEVICE_AP==1

Rsvd	31:21
mask_n_diag_sdi	20
mask_n_diag_ipi4_ch_fifo_control	19
mask_n_diag_ipi4	18
mask_n_diag_ipi3_ch_fifo_control	17
mask_n_diag_ipi3	16
mask_n_diag_ipi2_ch_fifo_control	15
mask_n_diag_ipi2	14
mask_n_diag_mt_ipi_control	13
mask_n_diag_cmu	12
mask_n_diag_ecf	11
mask_n_diag_pkt_if	10
mask_n_diag_sync	9
mask_n_diag_err_handler	8
mask_n_diag_pkt_builder	7
mask_n_diag_idi_ch_fifo_control	6
mask_n_diag_ipi_ch_fifo_control	5
mask_n_diag_reg_bank	4
mask_n_diag_phy_if_ctrl	3
mask_n_diag_ambaapbintf	2
mask_n_diag_idi	1
mask_n_diag_ipi	0

Table 5-153 Fields for Register: INT_MASK_N_DIAG0

Bits	Name	Memory Access	Description
31:21			Reserved Field: Yes
20	mask_n_diag_sdi	R/W	<p>Mask for Internal Diagnosis error for SDI Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_sdi) <p>Value After Reset: 0x0 Exists: CSI2_DEVICE_SDIF==1</p>
19	mask_n_diag_ipi4_ch_fifo_control	R/W	<p>Mask for Internal Diagnosis error for IPI4 Channel FIFO control Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ipi4_ch_fifo_control) <p>Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1</p>

Bits	Name	Memory Access	Description
18	mask_n_diag_ipi4	R/W	<p>Mask for Internal Diagnosis error for IPI4 interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ipi4) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
17	mask_n_diag_ipi3_ch_fifo_control	R/W	<p>Mask for Internal Diagnosis error for IPI3 Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ipi3_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	mask_n_diag_ipi3	R/W	<p>Mask for Internal Diagnosis error for IPI3 interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ipi3) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15	mask_n_diag_ipi2_ch_fifo_control	R/W	<p>Mask for Internal Diagnosis error for IPI2 Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ipi2_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
14	mask_n_diag_ipi2	R/W	<p>Mask for Internal Diagnosis error for IPI2 interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ipi2) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
13	mask_n_diag_mt_ipi_control	R/W	<p>Mask for Internal Diagnosis error for Multiple IPI Contrille</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_mt_ipi_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
12	mask_n_diag_cmu	R/W	<p>Mask for Internal Diagnosis error for Clock Manager Unit</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_cmu) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
11	mask_n_diag_ecf	R/W	<p>Mask for Internal Diagnosis error for ERR correction functional block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ecf) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	mask_n_diag_pkt_if	R/W	<p>Mask for Internal Diagnosis error for packet interface block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_pkt_if) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	mask_n_diag_sync	R/W	<p>Mask for Internal Diagnosis error for synchronizer</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_sync) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	mask_n_diag_err_handler	R/W	<p>Mask for Internal Diagnosis error for Error handler</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_err_handler) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	mask_n_diag_pkt_builder	R/W	<p>Mask for Internal Diagnosis error for Packet builder</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_pkt_builder) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_n_diag_idi_ch_fifo_control	R/W	<p>Mask for Internal Diagnosis error for IDI Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_idi_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>
5	mask_n_diag_ipi_ch_fifo_control	R/W	<p>Mask for Internal Diagnosis error for IPI Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ipi_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>

Bits	Name	Memory Access	Description
4	mask_n_diag_reg_bank	R/W	<p>Mask for Internal Diagnosis error for Register Bank</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_reg_bank) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	mask_n_diag_phy_if_ctrl	R/W	<p>Mask for Internal Diagnosis error for PHY interface control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_phy_if_ctrl) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_n_diag_ambaapbintf	R/W	<p>Mask for Internal Diagnosis error for APB interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ambaapbintf) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_n_diag_idi	R/W	<p>Mask for Internal Diagnosis error for IDI interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_idi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>
0	mask_n_diag_ipi	R/W	<p>Mask for Internal Diagnosis error for IPI interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_DIAG0_mask_n_diag_ipi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>

5.2.8.14 INT_FORCE_DIAG0

- **Name:** Interrupt status diagnosis group 0 force.
- **Description:** Test register to force activation of INT_ST_DIAG0 interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x864
- **Exists:** CSI2_DEVICE_AP==1

Rsvd	31:21
force_diag_sdi	20
force_diag_ipi4_ch_fifo_control	19
force_diag_ipi4	18
force_diag_ipi3_ch_fifo_control	17
force_diag_ipi3	16
force_diag_ipi2_ch_fifo_control	15
force_diag_ipi2	14
force_diag_mt_ipi_control	13
force_diag_cmu	12
force_diag_ecf	11
force_diag_pkt_if	10
force_diag_sync	9
force_diag_err_handler	8
force_diag_pkt_builder	7
force_diag_idi_ch_fifo_control	6
force_diag_ipi_ch_fifo_control	5
force_diag_reg_bank	4
force_diag_phy_if_ctrl	3
force_diag_ambaapbintf	2
force_diag_idi	1
force_diag_ipi	0

Table 5-154 Fields for Register: INT_FORCE_DIAG0

Bits	Name	Memory Access	Description
31:21			Reserved Field: Yes
20	force_diag_sdi	R/WC	<p>Force for Internal Diagnosis error for SDI Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_sdi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_SDIF==1</p>
19	force_diag_ipi4_ch_fifo_control	R/WC	<p>Force for Internal Diagnosis error for IPI4 Channel FIFO control Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ipi4_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>

Bits	Name	Memory Access	Description
18	force_diag_ipi4	R/WC	<p>Force for Internal Diagnosis error for IPI4 interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ipi4) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
17	force_diag_ipi3_ch_fifo_control	R/WC	<p>Force for Internal Diagnosis error for IPI3 Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ipi3_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	force_diag_ipi3	R/WC	<p>Force for Internal Diagnosis error for IPI3 interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ipi3) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15	force_diag_ipi2_ch_fifo_control	R/WC	<p>Force for Internal Diagnosis error for IPI2 Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ipi2_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
14	force_diag_ipi2	R/WC	<p>Force for Internal Diagnosis error for IPI2 interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ipi2) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
13	force_diag_mt_ipi_control	R/WC	<p>Force for Internal Diagnosis error for Multiple IPI Control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_mt_ipi_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
12	force_diag_cmu	R/WC	<p>Force for Internal Diagnosis error for Clock Manager Unit</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_cmu) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
11	force_diag_ecf	R/WC	<p>Force for Internal Diagnosis error for ERR correction functional block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ecf) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	force_diag_pkt_if	R/WC	<p>Force for Internal Diagnosis error for packet interface block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_pkt_if) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	force_diag_sync	R/WC	<p>Force for Internal Diagnosis error for synchronizer</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_sync) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	force_diag_err_handler	R/WC	<p>Force for Internal Diagnosis error for Error handler</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_err_handler) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	force_diag_pkt_builder	R/WC	<p>Force for Internal Diagnosis error for Packet builder</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_pkt_builder) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_diag_idi_ch_fifo_control	R/WC	<p>Force for Internal Diagnosis error for IDI Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_idi_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>
5	force_diag_ipi_ch_fifo_control	R/WC	<p>Force for Internal Diagnosis error for IPI Channel FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ipi_ch_fifo_control) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>

Bits	Name	Memory Access	Description
4	force_diag_reg_bank	R/WC	<p>Force for Internal Diagnosis error for Register Bank</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_reg_bank) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_diag_phy_if_ctrl	R/WC	<p>Force for Internal Diagnosis error for PHY interface control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_phy_if_ctrl) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_diag_ambaapbintf	R/WC	<p>Force for Internal Diagnosis error for APB interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ambaapbintf) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_diag_idi	R/WC	<p>Force for Internal Diagnosis error for IDI interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_idi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IF==1</p>
0	force_diag_ipi	R/WC	<p>Force for Internal Diagnosis error for IPI interface</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_DIAG0_force_diag_ipi) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI_IF==1</p>

5.2.8.15 INT_MASK_N_FAP_VPG

- **Name:** INT_ST_FAP_VPG interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_VPG; controls which interrupt status bits triggers the interrupt pin.1 - Enable the interrupt source.0 - Interrupt source is masked. Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x868
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_VPG==1



Table 5-155 Fields for Register: INT_MASK_N_FAP_VPG

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	mask_vpg_pkt_lost	R/W	<p>Mask for vpg_pkt_lost.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_VPG_mask_vpg_pkt_lost) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.16 INT_FORCE_FAP_VPG

- **Name:** INT_ST_FAP_VPG interrupt sources force.
- **Description:** Test register to force activation of INT_ST_FAP_VPG interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x86c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_VPG==1



Table 5-156 Fields for Register: INT_FORCE_FAP_VPG

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	force_vpg_pkt_lost	R/WC	<p>Force for vpg_pkt_lost. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_VPG_force_vpg_pkt_lost) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.17 INT_MASK_N_FAP_IDI

- **Name:** INT_ST_FAP_IDI Interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_IDI, controls which interrupt status bits triggers the interrupt pin.
1: - Enable the interrupt source.
0: - Interrupt source is masked.
Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x870
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

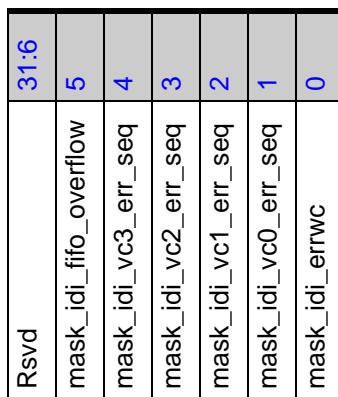


Table 5-157 Fields for Register: INT_MASK_N_FAP_IDI

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	mask_idi_fifo_overflow	R/W	<p>Mask for idi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_mask_idi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_idi_vc3_err_seq	R/W	<p>Mask for idi_vc3_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_mask_idi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	mask_idi_vc2_err_seq	R/W	<p>Mask for idi_vc2_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_mask_idi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	mask_idi_vc1_err_seq	R/W	<p>Mask for idi_vc1_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_mask_idi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_idi_vc0_err_seq	R/W	<p>Mask for idi_vc0_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_mask_idi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_idi_errwc	R/W	<p>Mask for idi_errwc.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_mask_idi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.18 INT_FORCE_FAP_IDI

- **Name:** INT_ST_FAP_IDI interrupt sources force.
- **Description:** Test register to force activation of INT_ST_FAP_IDI interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x874
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

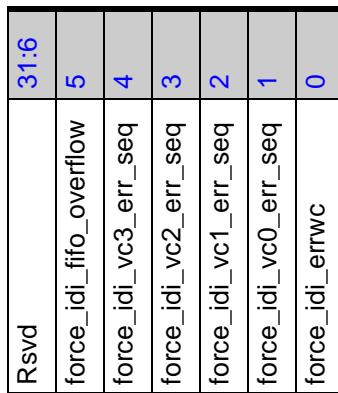


Table 5-158 Fields for Register: INT_FORCE_FAP_IDI

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	force_idi_fifo_overflow	R/WC	Force for idi_fifo_overflow. Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_IDI_force_idi_fifo_overflow) Value After Reset: 0x0 Exists: Always
4	force_idi_vc3_err_seq	R/WC	Force for idi_vc3_err_seq. Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_IDI_force_idi_vc3_err_seq) Value After Reset: 0x0 Exists: Always
3	force_idi_vc2_err_seq	R/WC	Force for idi_vc2_err_seq. Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_IDI_force_idi_vc2_err_seq) Value After Reset: 0x0 Exists: Always

Bits	Name	Memory Access	Description
2	force_idi_vc1_err_seq	R/WC	<p>Force for idi_vc1_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_force_idi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_idi_vc0_err_seq	R/WC	<p>Force for idi_vc0_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_force_idi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_idi_errwc	R/WC	<p>Force for idi_errwc.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_force_idi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.19 INT_MASK_N_FAP_IPI

- **Name:** INT_ST_FAP_IPI interrupt mask.
 - **Description:** Interrupt mask for INT_ST_FAP_IPI; controls which interrupt status bits triggers the interrupt pin.
1: Enable the interrupt source.
0: Interrupt source is masked.
Static read and write register.
 - **Size:** 32 bits
 - **Offset:** 0x878
 - **Exists:** CSI2 DEVICE AP==1&&CSI2 DEVICE IPI IF==1

Rsvd	31:28
mask_ipi4_fifo_underflow	27
mask_ipi4_errline	26
mask_ipi4_fifo_overflow	25
mask_ipi4_errpixel	24
Rsvd	23:20
mask_ipi3_fifo_underflow	19
mask_ipi3_errline	18
mask_ipi3_fifo_overflow	17
mask_ipi3_errpixel	16
Rsvd	15:12
mask_ipi2_fifo_underflow	11
mask_ipi2_errline	10
mask_ipi2_fifo_overflow	9
mask_ipi2_errpixel	8
Rsvd	7:5
mask_ipi_trans_conflict	4
mask_ipi_fifo_underflow	3
mask_ipi_errline	2
mask_ipi_fifo_overflow	1
mask_ipi_errpixel	0

Table 5-159 Fields for Register: INT_MASK_N_FAP_IPI

Bits	Name	Memory Access	Description
31:28			Reserved Field: Yes
27	mask_ipi4_fifo_underflow	R/W	Mask for ipi4_fifo_underflow
	Values:		
	■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi4_fifo_underflow)		
	Value After Reset: 0x0		
	Exists: CSI2_DEVICE_IPI4_IF==1		
26	mask_ipi4_errline	R/W	Mask for ipi4_errline
	Values:		
	■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi4_errline)		
	Value After Reset: 0x0		
	Exists: CSI2_DEVICE_IPI4_IF==1		
25	mask_ipi4_fifo_overflow	R/W	Mask for ipi4_fifo_overflow
	Values:		
	■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi4_fifo_overflow)		
	Value After Reset: 0x0		
	Exists: CSI2_DEVICE_IPI4_IF==1		

Bits	Name	Memory Access	Description
24	mask_ipi4_errpixel	R/W	<p>Mask for ipi4_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi4_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
23:20			Reserved Field: Yes
19	mask_ipi3_fifo_underflow	R/W	<p>Mask for ipi3_fifo_underflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi3_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
18	mask_ipi3_errline	R/W	<p>Mask for ipi3_errline</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi3_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
17	mask_ipi3_fifo_overflow	R/W	<p>Mask for ipi3_fifo_overflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi3_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	mask_ipi3_errpixel	R/W	<p>Mask for ipi3_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi3_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15:12			Reserved Field: Yes
11	mask_ipi2_fifo_underflow	R/W	<p>Mask for ipi2_fifo_underflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi2_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
10	mask_ipi2_errline	R/W	<p>Mask for ipi2_errline</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi2_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>

Bits	Name	Memory Access	Description
9	mask_ipi2_fifo_overflow	R/W	<p>Mask for ipi2_fifo_overflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi2_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
8	mask_ipi2_errpixel	R/W	<p>Mask for ipi2_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi2_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
7:5			Reserved Field: Yes
4	mask_ipi_trans_conflict	R/W	<p>Mask for ipi_trans_conflict.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi_trans_conflict) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IPI_IF==1</p>
3	mask_ipi_fifo_underflow	R/W	<p>Mask for ipi_fifo_underflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_ipi_errline	R/W	<p>Mask for ipi_errline.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi_errline) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_ipi_fifo_overflow	R/W	<p>Mask for ipi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_ipi_errpixel	R/W	<p>Mask for ipi_errpixel.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IPI_mask_ipi_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.20 INT_FORCE_FAP_IPI

- **Name:** INT_ST_FAP_IPI interrupt sources force.
- **Description:** Test register to force activation of INT_ST_FAP_IPI interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x87c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

Rsvd	31:28
force_ipi4_fifo_underflow	27
force_ipi4_errline	26
force_ipi4_fifo_overflow	25
force_ipi4_errpixel	24
Rsvd	23:20
force_ipi3_fifo_underflow	19
force_ipi3_errline	18
force_ipi3_fifo_overflow	17
force_ipi3_errpixel	16
Rsvd	15:12
force_ipi2_fifo_underflow	11
force_ipi2_errline	10
force_ipi2_fifo_overflow	9
force_ipi2_errpixel	8
Rsvd	7:5
force_ipi_trans_conflict	4
force_ipi_fifo_underflow	3
force_ipi_errline	2
force_ipi_fifo_overflow	1
force_ipi_errpixel	0

Table 5-160 Fields for Register: INT_FORCE_FAP_IPI

Bits	Name	Memory Access	Description
31:28			Reserved Field: Yes
27	force_ipi4_fifo_underflow	R/WC	Force for ipi4_fifo_underflow Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_IPI_force_ipi4_fifo_underflow) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1
26	force_ipi4_errline	R/WC	Force for ipi4_errline Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_IPI_force_ipi4_errline) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1
25	force_ipi4_fifo_overflow	R/WC	Force for ipi4_fifo_overflow Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_IPI_force_ipi4_fifo_overflow) Value After Reset: 0x0 Exists: CSI2_DEVICE_IPI4_IF==1

Bits	Name	Memory Access	Description
24	force_ipi4_errpixel	R/WC	<p>Force for ipi4_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi4_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI4_IF==1</p>
23:20			Reserved Field: Yes
19	force_ipi3_fifo_underflow	R/WC	<p>Force for ipi3_fifo_underflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi3_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
18	force_ipi3_errline	R/WC	<p>Force for ipi3_errline</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi3_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
17	force_ipi3_fifo_overflow	R/WC	<p>Force for ipi3_fifo_overflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi3_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
16	force_ipi3_errpixel	R/WC	<p>Force for ipi3_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi3_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI3_IF==1</p>
15:12			Reserved Field: Yes
11	force_ipi2_fifo_underflow	R/WC	<p>Force for ipi2_fifo_underflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi2_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
10	force_ipi2_errline	R/WC	<p>Force for ipi2_errline</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi2_errline) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>

Bits	Name	Memory Access	Description
9	force_ipi2_fifo_overflow	R/WC	<p>Force for ipi2_fifo_overflow</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi2_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
8	force_ipi2_errpixel	R/WC	<p>Force for ipi2_errpixel</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi2_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IPI2_IF==1</p>
7:5			Reserved Field: Yes
4	force_ipi_trans_conflict	R/WC	<p>Force for ipi_trans_conflict.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi_trans_conflict) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IPI_IF==1</p>
3	force_ipi_fifo_underflow	R/WC	<p>Force for ipi_fifo_underflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi_fifo_underflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_ipi_errline	R/WC	<p>Force for ipi_errline.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi_errline) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_ipi_fifo_overflow	R/WC	<p>Force for ipi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_ipi_errpixel	R/WC	<p>Force for ipi_errpixel.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IPI_force_ipi_errpixel) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.21 INT_MASK_N_FAP_PHY

- **Name:** INT_ST_FAP_PHY interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_PHY; controls which interrupt status bits triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x880
- **Exists:** CSI2_DEVICE_AP==1

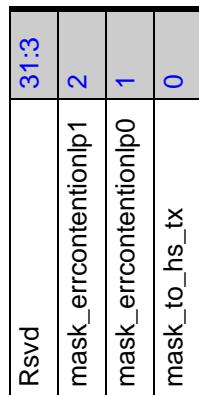


Table 5-161 Fields for Register: INT_MASK_N_FAP_PHY

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_errcontentionlp1	R/W	<p>Mask for errcontentionlp1 Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_PHY_mask_errcontentionlp1) <p>Value After Reset: 0x0 Exists: Always</p>
1	mask_errcontentionlp0	R/W	<p>Mask for errcontentionlp0 Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_PHY_mask_errcontentionlp0) <p>Value After Reset: 0x0 Exists: Always</p>
0	mask_to_hs_tx	R/W	<p>Mask for to_hs_tx. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_PHY_mask_to_hs_tx) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.22 INT_FORCE_FAP_PHY

- **Name:** INT_ST_FAP_PHY interrupt sources force.
- **Description:** Test register to force activation of INT_ST_FAP_PHY interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x884
- **Exists:** CSI2_DEVICE_AP==1

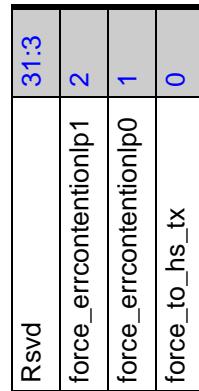


Table 5-162 Fields for Register: INT_FORCE_FAP_PHY

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_errcontentionlp1	R/WC	Force for errcontentionlp1 Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_PHY_force_errcontentionlp1) Value After Reset: 0x0 Exists: Always
1	force_errcontentionlp0	R/WC	Force for errcontentionlp0 Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_PHY_force_errcontentionlp0) Value After Reset: 0x0 Exists: Always
0	force_to_hs_tx	R/WC	Force for to_hs_tx. Values: <ul style="list-style-type: none">■ 0x0 (INT_FORCE_FAP_PHY_force_to_hs_tx) Value After Reset: 0x0 Exists: Always

5.2.8.23 INT_MASK_N_FAP_IDI_VCX

- **Name:** INT_ST_FAP_IDI_VCX interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_IDI_VCX; controls which interrupt status bits triggers the interrupt pin. 1: Enable the interrupt source.0: Interrupt source is masked. Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x888
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

31:12	Rsvd	mask_idi_vc15_err_seq	11
	mask_idi_vc14_err_seq	10	
	mask_idi_vc13_err_seq	9	
	mask_idi_vc12_err_seq	8	
	mask_idi_vc11_err_seq	7	
	mask_idi_vc10_err_seq	6	
	mask_idi_vc9_err_seq	5	
	mask_idi_vc8_err_seq	4	
	mask_idi_vc7_err_seq	3	
	mask_idi_vc6_err_seq	2	
	mask_idi_vc5_err_seq	1	
	mask_idi_vc4_err_seq	0	

Table 5-163 Fields for Register: INT_MASK_N_FAP_IDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	mask_idi_vc15_err_seq	R/W	<p>Mask for idi_vc15_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc15_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
10	mask_idi_vc14_err_seq	R/W	<p>Mask for idi_vc14_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc14_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
9	mask_idi_vc13_err_seq	R/W	<p>Mask for idi_vc13_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc13_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	mask_idi_vc12_err_seq	R/W	<p>Mask for idi_vc12_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	mask_idi_vc11_err_seq	R/W	<p>Mask for idi_vc11_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_idi_vc10_err_seq	R/W	<p>Mask for idi_vc10_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_idi_vc9_err_seq	R/W	<p>Mask for idi_vc9_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_idi_vc8_err_seq	R/W	<p>Mask for idi_vc8_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
3	mask_idi_vc7_err_seq	R/W	<p>Mask for idi_vc7_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_idi_vc6_err_seq	R/W	<p>Mask for idi_vc6_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_idi_vc5_err_seq	R/W	<p>Mask for idi_vc5_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_idi_vc4_err_seq	R/W	<p>Mask for idi_vc4_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX_mask_idi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.24 INT_FORCE_FAP_IDI_VCX

- **Name:** INT_ST_FAP_IDI_VCX interrupt sources force.
- **Description:** Test register to force activation of INT_ST_FAP_IDI_VCX interrupt sources.1: Force internal interrupt source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x88c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

Rsvd	31:12	11	10	9	8	7	6	5	4	3	2	1	0
force_idi_vc15_err_seq													
force_idi_vc14_err_seq													
force_idi_vc13_err_seq													
force_idi_vc12_err_seq													
force_idi_vc11_err_seq													
force_idi_vc10_err_seq													
force_idi_vc9_err_seq													
force_idi_vc8_err_seq													
force_idi_vc7_err_seq													
force_idi_vc6_err_seq													
force_idi_vc5_err_seq													
force_idi_vc4_err_seq													

Table 5-164 Fields for Register: INT_FORCE_FAP_IDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	force_idi_vc15_err_seq	R/WC	<p>Force for idi_vc15_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc15_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
10	force_idi_vc14_err_seq	R/WC	<p>Force for idi_vc14_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc14_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
9	force_idi_vc13_err_seq	R/WC	<p>Force for idi_vc13_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc13_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
8	force_idi_vc12_err_seq	R/WC	<p>Force for idi_vc12_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	force_idi_vc11_err_seq	R/WC	<p>Force for idi_vc11_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_idi_vc10_err_seq	R/WC	<p>Force for idi_vc10_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_idi_vc9_err_seq	R/WC	<p>Force for idi_vc9_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_idi_vc8_err_seq	R/WC	<p>Force for idi_vc8_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_idi_vc7_err_seq	R/WC	<p>Force for idi_vc7_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_idi_vc6_err_seq	R/WC	<p>Force for idi_vc6_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	force_idi_vc5_err_seq	R/WC	<p>Force for idi_vc5_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_idi_vc4_err_seq	R/WC	<p>Force for idi_vc4_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX_force_idi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.25 INT_MASK_N_FAP_IDI_VCX2

- **Name:** INT_ST_FAP_IDI_VCX2 interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_IDI_VCX2; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x890
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

Rsvd	31:16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
mask_idi_vc31_err_seq																	
mask_idi_vc30_err_seq																	
mask_idi_vc29_err_seq																	
mask_idi_vc28_err_seq																	
mask_idi_vc27_err_seq																	
mask_idi_vc26_err_seq																	
mask_idi_vc25_err_seq																	
mask_idi_vc24_err_seq																	
mask_idi_vc23_err_seq																	
mask_idi_vc22_err_seq																	
mask_idi_vc21_err_seq																	
mask_idi_vc20_err_seq																	
mask_idi_vc19_err_seq																	
mask_idi_vc18_err_seq																	
mask_idi_vc17_err_seq																	
mask_idi_vc16_err_seq																	

Table 5-165 Fields for Register: INT_MASK_N_FAP_IDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	mask_idi_vc31_err_seq	R/W	<p>Mask for idi_vc31_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc31_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
14	mask_idi_vc30_err_seq	R/W	<p>Mask for idi_vc30_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc30_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
13	mask_idi_vc29_err_seq	R/W	<p>Mask for idi_vc29_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc29_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
12	mask_idi_vc28_err_seq	R/W	<p>Mask for idi_vc28_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	mask_idi_vc27_err_seq	R/W	<p>Mask for idi_vc27_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	mask_idi_vc26_err_seq	R/W	<p>Mask for idi_vc26_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	mask_idi_vc25_err_seq	R/W	<p>Mask for idi_vc25_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	mask_idi_vc24_err_seq	R/W	<p>Mask for idi_vc24_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7	mask_idi_vc23_err_seq	R/W	<p>Mask for idi_vc23_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_idi_vc22_err_seq	R/W	<p>Mask for idi_vc22_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_idi_vc21_err_seq	R/W	<p>Mask for idi_vc21_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_idi_vc20_err_seq	R/W	<p>Mask for idi_vc20_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	mask_idi_vc19_err_seq	R/W	<p>Mask for idi_vc19_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_idi_vc18_err_seq	R/W	<p>Mask for idi_vc18_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	mask_idi_vc17_err_seq	R/W	<p>Mask for idi_vc17_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_idi_vc16_err_seq	R/W	<p>Mask for idi_vc16_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_IDI_VCX2_mask_idi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.26 INT_FORCE_FAP_IDI_VCX2

- **Name:** INT_ST_FAP_IDI_VCX2 interrupt sources force.
 - **Description:** Test register to force activation of INT_ST_FAP_IDI_VCX2 interrupt sources.1: Force internal interrupt source.0: No action. Writing this register clears it.
 - **Size:** 32 bits
 - **Offset:** 0x894
 - **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

Rsvd		31:16
	force_idi_vc31_err_seq	15
	force_idi_vc30_err_seq	14
	force_idi_vc29_err_seq	13
	force_idi_vc28_err_seq	12
	force_idi_vc27_err_seq	11
	force_idi_vc26_err_seq	10
	force_idi_vc25_err_seq	9
	force_idi_vc24_err_seq	8
	force_idi_vc23_err_seq	7
	force_idi_vc22_err_seq	6
	force_idi_vc21_err_seq	5
	force_idi_vc20_err_seq	4
	force_idi_vc19_err_seq	3
	force_idi_vc18_err_seq	2
	force_idi_vc17_err_seq	1
	force_idi_vc16_err_seq	0

Table 5-166 Fields for Register: INT_FORCE_FAP_IDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	force_idi_vc31_err_seq	R/WC	<p>Force for idi_vc31_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc31_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
14	force_idi_vc30_err_seq	R/WC	<p>Force for idi_vc30_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc30_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
13	force_idi_vc29_err_seq	R/WC	<p>Force for idi_vc29_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc29_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
12	force_idi_vc28_err_seq	R/WC	<p>Force for idi_vc28_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	force_idi_vc27_err_seq	R/WC	<p>Force for idi_vc27_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	force_idi_vc26_err_seq	R/WC	<p>Force for idi_vc26_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	force_idi_vc25_err_seq	R/WC	<p>Force for idi_vc25_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	force_idi_vc24_err_seq	R/WC	<p>Force for idi_vc24_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7	force_idi_vc23_err_seq	R/WC	<p>Force for idi_vc23_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_idi_vc22_err_seq	R/WC	<p>Force for idi_vc22_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_idi_vc21_err_seq	R/WC	<p>Force for idi_vc21_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_idi_vc20_err_seq	R/WC	<p>Force for idi_vc20_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_idi_vc19_err_seq	R/WC	<p>Force for idi_vc19_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_idi_vc18_err_seq	R/WC	<p>Force for idi_vc18_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	force_idi_vc17_err_seq	R/WC	<p>Force for idi_vc17_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_idi_vc16_err_seq	R/WC	<p>Force for idi_vc16_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_IDI_VCX2_force_idi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.27 INT_MASK_N_FAP_MT_IPI

- **Name:** INT_ST_FAP_MT_IPI interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_MT_IPI; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x898
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1



Table 5-167 Fields for Register: INT_MASK_N_FAP_MT_IPI

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	mask_mt_ipi_fifo_overflow	R/W	<p>Mask for mt_ipi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_MT_IPI_mask_mt_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.28 INT_FORCE_FAP_MT_IPI

- **Name:** INT_ST_FAP_MT_IPI interrupt sources force.
- **Description:** Test register to force activation of INT_ST_FAP_MT_IPI interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x89c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1



Table 5-168 Fields for Register: INT_FORCE_FAP_MT_IPI

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	force_mt_ipi_fifo_overflow	R/WC	<p>Force for mt_ipi_fifo_overflow.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_MT_IPI_force_mt_ipi_fifo_overflow) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.29 INT_MASK_N_FAP_SDI

- **Name:** INT_ST_FAP_SDI Interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_SDI, controls which interrupt status bits triggers the interrupt pin.
1: - Enable the interrupt source.
0: - Interrupt source is masked.
Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x8a0
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==1

Rsvd	31:7	mask_sdi_pld_err	6	mask_sdi_hdr_err	5	mask_sdi_vc3_err_seq	4	mask_sdi_vc2_err_seq	3	mask_sdi_vc1_err_seq	2	mask_sdi_vc0_err_seq	1	mask_sdi_errwc	0
------	------	------------------	---	------------------	---	----------------------	---	----------------------	---	----------------------	---	----------------------	---	----------------	---

Table 5-169 Fields for Register: INT_MASK_N_FAP_SDI

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	mask_sdi_pld_err	R/W	<p>Mask for sdi_pld_err.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_mask_sdi_pld_err) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_sdi_hdr_err	R/W	<p>Mask for sdi_hdr_err.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_mask_sdi_hdr_err) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_sdi_vc3_err_seq	R/W	<p>Mask for sdi_vc3_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_mask_sdi_vc3_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
3	mask_sdi_vc2_err_seq	R/W	<p>Mask for sdi_vc2_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_mask_sdi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_sdi_vc1_err_seq	R/W	<p>Mask for sdi_vc1_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_mask_sdi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_sdi_vc0_err_seq	R/W	<p>Mask for sdi_vc0_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_mask_sdi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_sdi_errwc	R/W	<p>Mask for sdi_errwc.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_mask_sdi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.30 INT_FORCE_FAP_SDI

- **Name:** INT_ST_FAP_SDI interrupt sources force.
- **Description:** Test register to force activation of INT_ST_FAP_SDI interrupt sources. 1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x8a4
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==1

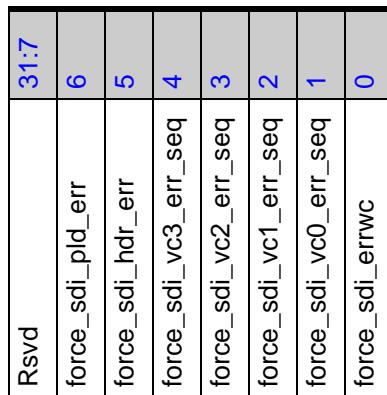


Table 5-170 Fields for Register: INT_FORCE_FAP_SDI

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	force_sdi_pld_err	R/WC	Force for sdi_pld_err. Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_force_sdi_pld_err) Value After Reset: 0x0 Exists: Always
5	force_sdi_hdr_err	R/WC	Force for sdi_hdr_err. Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_force_sdi_hdr_err) Value After Reset: 0x0 Exists: Always
4	force_sdi_vc3_err_seq	R/WC	Force for sdi_vc3_err_seq. Values: <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_force_sdi_vc3_err_seq) Value After Reset: 0x0 Exists: Always

Bits	Name	Memory Access	Description
3	force_sdi_vc2_err_seq	R/WC	<p>Force for sdi_vc2_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_force_sdi_vc2_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_sdi_vc1_err_seq	R/WC	<p>Force for sdi_vc1_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_force_sdi_vc1_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_sdi_vc0_err_seq	R/WC	<p>Force for sdi_vc0_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_force_sdi_vc0_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_sdi_errwc	R/WC	<p>Force for sdi_errwc.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_force_sdi_errwc) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.31 INT_MASK_N_FAP_SDI_VCX

- **Name:** INT_ST_FAP_SDI_VCX interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_SDI_VCX; controls which interrupt status bits triggers the interrupt pin. 1: Enable the interrupt source.0: Interrupt source is masked. Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x8a8
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

31:12	Rsvd	mask_sdi_vc15_err_seq	11
	mask_sdi_vc14_err_seq	10	
	mask_sdi_vc13_err_seq	9	
	mask_sdi_vc12_err_seq	8	
	mask_sdi_vc11_err_seq	7	
	mask_sdi_vc10_err_seq	6	
	mask_sdi_vc9_err_seq	5	
	mask_sdi_vc8_err_seq	4	
	mask_sdi_vc7_err_seq	3	
	mask_sdi_vc6_err_seq	2	
	mask_sdi_vc5_err_seq	1	
	mask_sdi_vc4_err_seq	0	

Table 5-171 Fields for Register: INT_MASK_N_FAP_SDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	mask_sdi_vc15_err_seq	R/W	<p>Mask for sdi_vc15_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc15_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
10	mask_sdi_vc14_err_seq	R/W	<p>Mask for sdi_vc14_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc14_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
9	mask_sdi_vc13_err_seq	R/W	<p>Mask for sdi_vc13_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc13_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	mask_sdi_vc12_err_seq	R/W	<p>Mask for sdi_vc12_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	mask_sdi_vc11_err_seq	R/W	<p>Mask for sdi_vc11_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_sdi_vc10_err_seq	R/W	<p>Mask for sdi_vc10_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_sdi_vc9_err_seq	R/W	<p>Mask for sdi_vc9_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_sdi_vc8_err_seq	R/W	<p>Mask for sdi_vc8_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
3	mask_sdi_vc7_err_seq	R/W	<p>Mask for sdi_vc7_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_sdi_vc6_err_seq	R/W	<p>Mask for sdi_vc6_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	mask_sdi_vc5_err_seq	R/W	<p>Mask for sdi_vc5_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_sdi_vc4_err_seq	R/W	<p>Mask for sdi_vc4_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX_mask_sdi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.32 INT_FORCE_FAP_SDI_VCX

- Name:** INT_ST_FAP_SDI_VCX interrupt sources force.
- Description:** Test register to force activation of INT_ST_FAP_SDI_VCX interrupt sources.1: Force internal interrupt source.0: No action.Writing this register clears it.
- Size:** 32 bits
- Offset:** 0x8ac
- Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==1&&CSI2_DEVICE_VC_EXTENSION==1

31:12	Rsvd	force_sdi_vc15_err_seq	11	force_sdi_vc14_err_seq	10	force_sdi_vc13_err_seq	9	force_sdi_vc12_err_seq	8	force_sdi_vc11_err_seq	7	force_sdi_vc10_err_seq	6	force_sdi_vc9_err_seq	5	force_sdi_vc8_err_seq	4	force_sdi_vc7_err_seq	3	force_sdi_vc6_err_seq	2	force_sdi_vc5_err_seq	1	force_sdi_vc4_err_seq	0
-------	------	------------------------	----	------------------------	----	------------------------	---	------------------------	---	------------------------	---	------------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---	-----------------------	---

Table 5-172 Fields for Register: INT_FORCE_FAP_SDI_VCX

Bits	Name	Memory Access	Description
31:12			Reserved Field: Yes
11	force_sdi_vc15_err_seq	R/WC	<p>Force for sdi_vc15_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc15_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
10	force_sdi_vc14_err_seq	R/WC	<p>Force for sdi_vc14_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc14_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
9	force_sdi_vc13_err_seq	R/WC	<p>Force for sdi_vc13_err_seq. Values:</p> <ul style="list-style-type: none"> 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc13_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
8	force_sdi_vc12_err_seq	R/WC	<p>Force for sdi_vc12_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc12_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	force_sdi_vc11_err_seq	R/WC	<p>Force for sdi_vc11_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc11_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_sdi_vc10_err_seq	R/WC	<p>Force for sdi_vc10_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc10_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_sdi_vc9_err_seq	R/WC	<p>Force for sdi_vc9_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc9_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_sdi_vc8_err_seq	R/WC	<p>Force for sdi_vc8_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc8_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_sdi_vc7_err_seq	R/WC	<p>Force for sdi_vc7_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc7_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
2	force_sdi_vc6_err_seq	R/WC	<p>Force for sdi_vc6_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc6_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_sdi_vc5_err_seq	R/WC	<p>Force for sdi_vc5_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc5_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_sdi_vc4_err_seq	R/WC	<p>Force for sdi_vc4_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX_force_sdi_vc4_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.33 INT_MASK_N_FAP_SDI_VCX2

- **Name:** INT_ST_FAP_SDI_VCX2 interrupt mask.
- **Description:** Interrupt mask for INT_ST_FAP_SDI_VCX2; controls which interrupt status bit triggers the interrupt pin.1: Enable the interrupt source.0: Interrupt source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x8b0
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDIF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

31:16	Rsvd	mask_sdi_vc31_err_seq	15
	mask_sdi_vc30_err_seq	14	
	mask_sdi_vc29_err_seq	13	
	mask_sdi_vc28_err_seq	12	
	mask_sdi_vc27_err_seq	11	
	mask_sdi_vc26_err_seq	10	
	mask_sdi_vc25_err_seq	9	
	mask_sdi_vc24_err_seq	8	
	mask_sdi_vc23_err_seq	7	
	mask_sdi_vc22_err_seq	6	
	mask_sdi_vc21_err_seq	5	
	mask_sdi_vc20_err_seq	4	
	mask_sdi_vc19_err_seq	3	
	mask_sdi_vc18_err_seq	2	
	mask_sdi_vc17_err_seq	1	
	mask_sdi_vc16_err_seq	0	

Table 5-173 Fields for Register: INT_MASK_N_FAP_SDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	mask_sdi_vc31_err_seq	R/W	<p>Mask for sdi_vc31_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc31_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
14	mask_sdi_vc30_err_seq	R/W	<p>Mask for sdi_vc30_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc30_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
13	mask_sdi_vc29_err_seq	R/W	<p>Mask for sdi_vc29_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc29_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
12	mask_sdi_vc28_err_seq	R/W	<p>Mask for sdi_vc28_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	mask_sdi_vc27_err_seq	R/W	<p>Mask for sdi_vc27_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	mask_sdi_vc26_err_seq	R/W	<p>Mask for sdi_vc26_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	mask_sdi_vc25_err_seq	R/W	<p>Mask for sdi_vc25_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	mask_sdi_vc24_err_seq	R/W	<p>Mask for sdi_vc24_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7	mask_sdi_vc23_err_seq	R/W	<p>Mask for sdi_vc23_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	mask_sdi_vc22_err_seq	R/W	<p>Mask for sdi_vc22_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_sdi_vc21_err_seq	R/W	<p>Mask for sdi_vc21_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_sdi_vc20_err_seq	R/W	<p>Mask for sdi_vc20_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	mask_sdi_vc19_err_seq	R/W	<p>Mask for sdi_vc19_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	mask_sdi_vc18_err_seq	R/W	<p>Mask for sdi_vc18_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	mask_sdi_vc17_err_seq	R/W	<p>Mask for sdi_vc17_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_sdi_vc16_err_seq	R/W	<p>Mask for sdi_vc16_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_MASK_N_FAP_SDI_VCX2_mask_sdi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.34 INT_FORCE_FAP_SDI_VCX2

- **Name:** INT_ST_FAP_SDI_VCX2 interrupt sources force.
- **Description:** Test register to force activation of INT_ST_FAP_SDI_VCX2 interrupt sources.1: Force internal interrupt source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x8b4
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDIF==1&&CSI2_DEVICE_VC_EXTENSION==1&&CSI2_DEVICE_CPHY_TYPE==1

	31:16
Rsvd	
force_sdi_vc31_err_seq	15
force_sdi_vc30_err_seq	14
force_sdi_vc29_err_seq	13
force_sdi_vc28_err_seq	12
force_sdi_vc27_err_seq	11
force_sdi_vc26_err_seq	10
force_sdi_vc25_err_seq	9
force_sdi_vc24_err_seq	8
force_sdi_vc23_err_seq	7
force_sdi_vc22_err_seq	6
force_sdi_vc21_err_seq	5
force_sdi_vc20_err_seq	4
force_sdi_vc19_err_seq	3
force_sdi_vc18_err_seq	2
force_sdi_vc17_err_seq	1
force_sdi_vc16_err_seq	0

Table 5-174 Fields for Register: INT_FORCE_FAP_SDI_VCX2

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15	force_sdi_vc31_err_seq	R/WC	<p>Force for sdi_vc31_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc31_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>
14	force_sdi_vc30_err_seq	R/WC	<p>Force for sdi_vc30_err_seq. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc30_err_seq) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
13	force_sdi_vc29_err_seq	R/WC	<p>Force for sdi_vc29_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc29_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
12	force_sdi_vc28_err_seq	R/WC	<p>Force for sdi_vc28_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc28_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
11	force_sdi_vc27_err_seq	R/WC	<p>Force for sdi_vc27_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc27_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
10	force_sdi_vc26_err_seq	R/WC	<p>Force for sdi_vc26_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc26_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
9	force_sdi_vc25_err_seq	R/WC	<p>Force for sdi_vc25_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc25_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
8	force_sdi_vc24_err_seq	R/WC	<p>Force for sdi_vc24_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc24_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
7	force_sdi_vc23_err_seq	R/WC	<p>Force for sdi_vc23_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc23_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	force_sdi_vc22_err_seq	R/WC	<p>Force for sdi_vc22_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc22_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_sdi_vc21_err_seq	R/WC	<p>Force for sdi_vc21_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc21_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_sdi_vc20_err_seq	R/WC	<p>Force for sdi_vc20_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc20_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	force_sdi_vc19_err_seq	R/WC	<p>Force for sdi_vc19_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc19_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2	force_sdi_vc18_err_seq	R/WC	<p>Force for sdi_vc18_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc18_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
1	force_sdi_vc17_err_seq	R/WC	<p>Force for sdi_vc17_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc17_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_sdi_vc16_err_seq	R/WC	<p>Force for sdi_vc16_err_seq.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (INT_FORCE_FAP_SDI_VCX2_force_sdi_vc16_err_seq) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.35 IPI_AP_STATUS

- **Name:** IPI AP check status.
- **Description:** AP check status for IPI block.
- **Size:** 32 bits
- **Offset:** 0x900
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

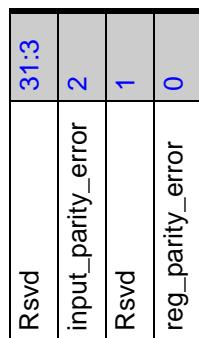


Table 5-175 Fields for Register: IPI_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	input_parity_error	RC	Input parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IPI_AP_STATUS_input_parity_error) Value After Reset: 0x0 Exists: Always
1			Reserved Field: Yes
0	reg_parity_error	RC	Register parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IPI_AP_STATUS_reg_parity_error) Value After Reset: 0x0 Exists: Always

5.2.8.36 IDI_AP_STATUS

- **Name:** IDI AP check status.
- **Description:** AP check status for IDI block.
- **Size:** 32 bits
- **Offset:** 0x904
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1



Table 5-176 Fields for Register: IDI_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	input_parity_error	RC	Input parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IDI_AP_STATUS_input_parity_error) Value After Reset: 0x0 Exists: Always
1			Reserved Field: Yes
0	reg_parity_error	RC	Register parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IDI_AP_STATUS_reg_parity_error) Value After Reset: 0x0 Exists: Always

5.2.8.37 AMBAAPBINTF_AP_STATUS

- **Name:** Amba apb interface AP check status.
- **Description:** AP check status for amba apb interface block.
- **Size:** 32 bits
- **Offset:** 0x908
- **Exists:** CSI2_DEVICE_AP==1

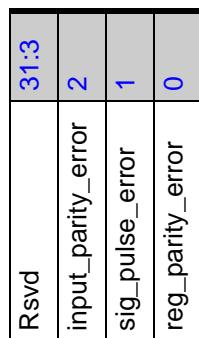


Table 5-177 Fields for Register: AMBAAPBINTF_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	input_parity_error	RC	<p>Input parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (AMBAAPBINTF_AP_STATUS_input_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	sig_pulse_error	RC	<p>Single pulse check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (AMBAAPBINTF_AP_STATUS_sig_pulse_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (AMBAAPBINTF_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.38 PHY_IF_CTRL_AP_STATUS

- **Name:** PHY interface control AP check status.
- **Description:** AP check status for PHY interface control block.
- **Size:** 32 bits
- **Offset:** 0x90c
- **Exists:** CSI2_DEVICE_AP==1

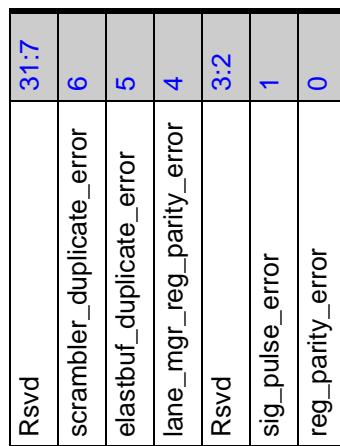


Table 5-178 Fields for Register: PHY_IF_CTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	scrambler_duplicate_error	RC	<p>Duplicate module check error for the scrambler wrapper inside the lane mg</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_IF_CTRL_AP_STATUS_scrambler_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_DATA_SCRAMBLING==1</p>
5	elastbuf_duplicate_error	RC	<p>Duplicate module check error for the Elastbuf module inside the lane mgr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_IF_CTRL_AP_STATUS_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
4	lane_mgr_reg_parity_error	RC	<p>Register parity check error for the lane mgr block inside the phy interface control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_IF_CTRL_AP_STATUS_lane_mgr_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:2			Reserved Field: Yes
1	sig_pulse_error	RC	<p>Single pulse check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_IF_CTRL_AP_STATUS_sig_pulse_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PHY_IF_CTRL_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.39 REG_BANK_AP_STATUS

- **Name:** Register bank AP check status.
- **Description:** AP check status for register bank block.
- **Size:** 32 bits
- **Offset:** 0x910
- **Exists:** CSI2_DEVICE_AP==1

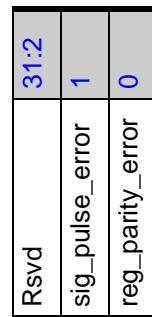


Table 5-179 Fields for Register: REG_BANK_AP_STATUS

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	sig_pulse_error	RC	<p>Single pulse check error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (REG_BANK_AP_STATUS_sig_pulse_error) <p>Value After Reset: 0x0 Exists: Always</p>
0	reg_parity_error	RC	<p>Register parity check error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (REG_BANK_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.40 IPI_FIFOCTRL_AP_STATUS

- **Name:** IPI FIFO control AP check status.
- **Description:** AP check status for IPI FIFO control block.
- **Size:** 32 bits
- **Offset:** 0x914
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

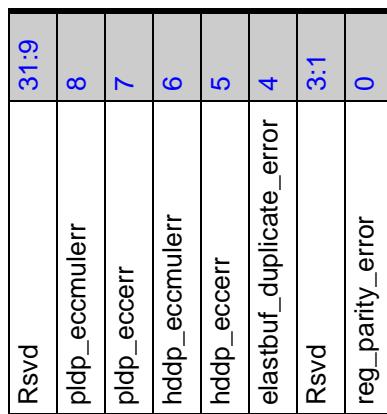


Table 5-180 Fields for Register: IPI_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	pldp_eccmulerr	RC	<p>multiple-bit error for IPI payload datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFOCTRL_AP_STATUS_pldp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
7	pldp_eccerr	RC	<p>single-bit error IPI payload datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFOCTRL_AP_STATUS_pldp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	hddp_eccmulerr	RC	<p>multiple-bit error for IPI header datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFOCTRL_AP_STATUS_hddp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	hddp_eccerr	RC	<p>single-bit error IPI header datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFOCTRL_AP_STATUS_hddp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	elastbuf_duplicate_error	RC	<p>Duplicate module check error for the Elastbuf module inside the IPI FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFOCTRL_AP_STATUS_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_FIFOCTRL_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.41 IDI_FIFOCTRL_AP_STATUS

- **Name:** IDI FIFO control AP check status.
- **Description:** AP check status for IDI FIFO control block.
- **Size:** 32 bits
- **Offset:** 0x918
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

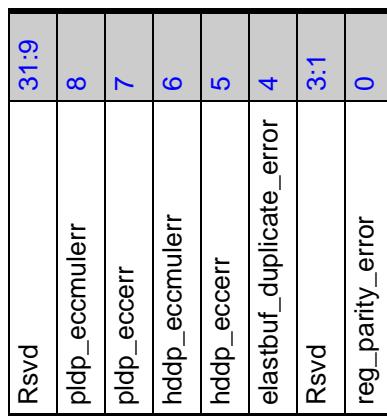


Table 5-181 Fields for Register: IDI_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	pldp_eccmulerr	RC	multiple-bit error for IDI payload datapath with ecc Values: <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFOCTRL_AP_STATUS_pldp_eccmulerr) Value After Reset: 0x0 Exists: Always
7	pldp_eccerr	RC	single-bit error IDI payload datapath with ecc Values: <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFOCTRL_AP_STATUS_pldp_eccerr) Value After Reset: 0x0 Exists: Always
6	hddp_eccmulerr	RC	multiple-bit error for IDI header datapath with ecc Values: <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFOCTRL_AP_STATUS_hddp_eccmulerr) Value After Reset: 0x0 Exists: Always

Bits	Name	Memory Access	Description
5	hddp_eccerr	RC	<p>single-bit error IDI header datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFOCTRL_AP_STATUS_hddp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	elastbuf_duplicate_error	RC	<p>Duplicate module check error for the Elastbuf module inside the IDI FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFOCTRL_AP_STATUS_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IDI_FIFOCTRL_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.42 PKT_BUILDER_AP_STATUS

- **Name:** Packet builder AP check status.
- **Description:** AP check status for packet builder block.
- **Size:** 32 bits
- **Offset:** 0x91c
- **Exists:** CSI2_DEVICE_AP==1

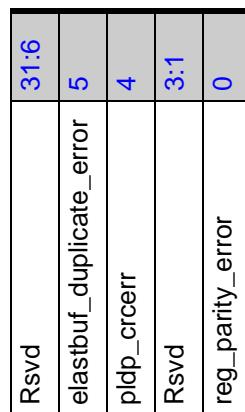


Table 5-182 Fields for Register: PKT_BUILDER_AP_STATUS

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	elastbuf_duplicate_error	RC	<p>Duplicate module check error for the Elastbuf module inside the PKT BUILDER</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PKT_BUILDER_AP_STATUS_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	pldp_crcerr	RC	<p>error for payload datapath with crc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PKT_BUILDER_AP_STATUS_pldp_crcerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PKT_BUILDER_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.43 ERR_HANDLER_AP_STATUS

- **Name:** Error handler AP check status.
- **Description:** AP check status for error handler block.
- **Size:** 32 bits
- **Offset:** 0x920
- **Exists:** CSI2_DEVICE_AP==1

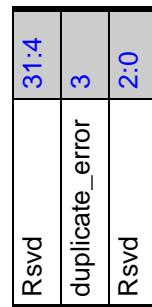


Table 5-183 Fields for Register: ERR_HANDLER_AP_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	duplicate_error	RC	<p>Duplicate mode check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_HANDLER_AP_STATUS_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2:0			Reserved Field: Yes

5.2.8.44 SYNC_AP_STATUS

- **Name:** Synchronizer AP check status.
- **Description:** AP check status for synchronizer block.
- **Size:** 32 bits
- **Offset:** 0x924
- **Exists:** CSI2_DEVICE_AP==1

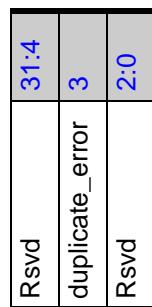


Table 5-184 Fields for Register: SYNC_AP_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	duplicate_error	RC	<p>Duplicate module check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (SYNC_AP_STATUS_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2:0			Reserved Field: Yes

5.2.8.45 PKT_IF_AP_STATUS

- **Name:** Packet interface AP check status.
- **Description:** AP check status for packet interface block.
- **Size:** 32 bits
- **Offset:** 0x928
- **Exists:** CSI2_DEVICE_AP==1

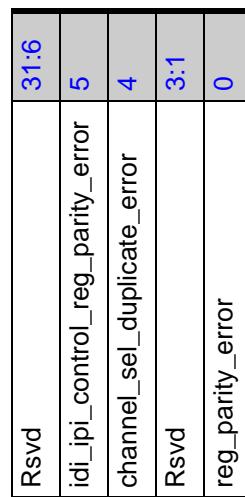


Table 5-185 Fields for Register: PKT_IF_AP_STATUS

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	idi_ipi_control_reg_parity_error	RC	<p>Register parity check error for IDI and IPI control for working simultaneously</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PKT_IF_AP_STATUS_idi_ipi_control_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IPI_IF == 1</p>
4	channel_sel_duplicate_error	RC	<p>Duplicate mode check error for the Channel Select module inside the block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (PKT_IF_AP_STATUS_channel_sel_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes

Bits	Name	Memory Access	Description
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (PKT_IF_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.46 ECF_AP_STATUS

- **Name:** Ecf AP check status.
- **Description:** AP check status for ecf block.
- **Size:** 32 bits
- **Offset:** 0x92c
- **Exists:** CSI2_DEVICE_AP==1



Table 5-186 Fields for Register: ECF_AP_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	reg_parity_error	RC	Register parity check error Values: ■ 0x0 (ECF_AP_STATUS_reg_parity_error) Value After Reset: 0x0 Exists: Always

5.2.8.47 CMU_AP_STATUS

- **Name:** Cmu AP check status.
- **Description:** AP check status for cmu block.
- **Size:** 32 bits
- **Offset:** 0x930
- **Exists:** CSI2_DEVICE_AP==1



Table 5-187 Fields for Register: CMU_AP_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	reg_parity_error	RC	Register parity check error Values: ■ 0x0 (CMU_AP_STATUS_reg_parity_error) Value After Reset: 0x0 Exists: Always

5.2.8.48 MT_IPI_CONTROL_AP_STATUS

- **Name:** Multiple IPI Controller AP check status.
- **Description:** AP check status for Multiple IPI Controller block.
- **Size:** 32 bits
- **Offset:** 0x934
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

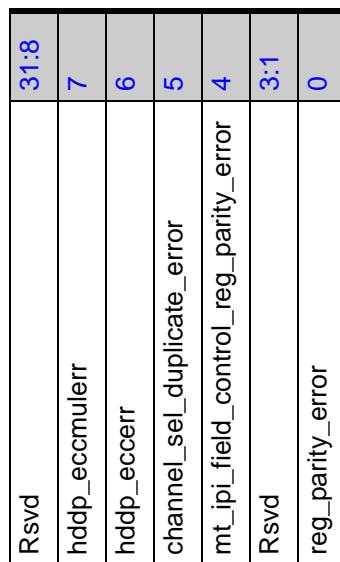


Table 5-188 Fields for Register: MT_IPI_CONTROL_AP_STATUS

Bits	Name	Memory Access	Description
31:8			Reserved Field: Yes
7	hddp_eccmulerr	RC	<p>multiple-bit error for MT IPI header datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_CONTROL_AP_STATUS_hddp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
6	hddp_eccerr	RC	<p>single-bit error MT IPI header datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_CONTROL_AP_STATUS_hddp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
5	channel_sel_duplicate_error	RC	<p>Duplicate mode check error for the Channel Select module inside the block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_CONTROL_AP_STATUS_channel_sel_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mt_ipi_field_control_reg_parity_err or	RC	<p>Register parity check error for the mt ipi field control module inside the block</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_CONTROL_AP_STATUS_mt_ipi_field_control_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MT_IPI_CONTROL_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.49 IPI2_AP_STATUS

- **Name:** IPI2 AP check status.
- **Description:** AP check status for IPI2 block.
- **Size:** 32 bits
- **Offset:** 0x938
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1



Table 5-189 Fields for Register: IPI2_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	input_parity_error	RC	Input parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IPI2_AP_STATUS_input_parity_error) Value After Reset: 0x0 Exists: Always
1			Reserved Field: Yes
0	reg_parity_error	RC	Register parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IPI2_AP_STATUS_reg_parity_error) Value After Reset: 0x0 Exists: Always

5.2.8.50 IPI2_FIFOCTRL_AP_STATUS

- **Name:** IPI2 FIFO control AP check status.
- **Description:** AP check status for IPI2 FIFO control block.
- **Size:** 32 bits
- **Offset:** 0x93c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

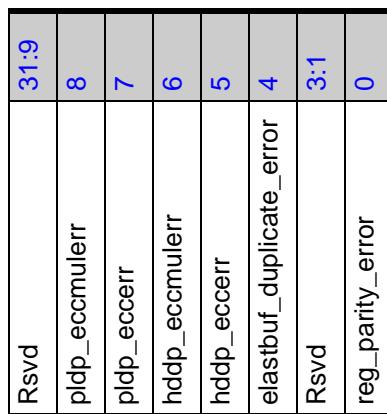


Table 5-190 Fields for Register: IPI2_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	pldp_eccmulerr	RC	multiple-bit error for IPI2 payload datapath with ecc Values: ■ 0x0 (IPI2_FIFOCTRL_AP_STATUS_pldp_eccmulerr) Value After Reset: 0x0 Exists: Always
7	pldp_eccerr	RC	single-bit error IPI2 payload datapath with ecc Values: ■ 0x0 (IPI2_FIFOCTRL_AP_STATUS_pldp_eccerr) Value After Reset: 0x0 Exists: Always
6	hddp_eccmulerr	RC	multiple-bit error for IPI2 header datapath with ecc Values: ■ 0x0 (IPI2_FIFOCTRL_AP_STATUS_hddp_eccmulerr) Value After Reset: 0x0 Exists: Always

Bits	Name	Memory Access	Description
5	hddp_eccerr	RC	<p>single-bit error IPI2 header datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_FIFOCTRL_AP_STATUS_hddp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	elastbuf_duplicate_error	RC	<p>Duplicate module check error for the Elastbuf module inside the IPI2 FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_FIFOCTRL_AP_STATUS_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_FIFOCTRL_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.51 IPI3_AP_STATUS

- **Name:** IPI3 AP check status.
- **Description:** AP check status for IPI3 block.
- **Size:** 32 bits
- **Offset:** 0x940
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI3_IF==1

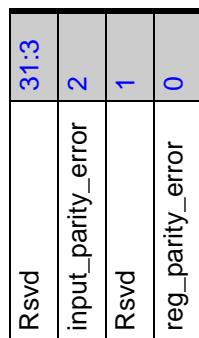


Table 5-191 Fields for Register: IPI3_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	input_parity_error	RC	Input parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IPI3_AP_STATUS_input_parity_error) Value After Reset: 0x0 Exists: Always
1			Reserved Field: Yes
0	reg_parity_error	RC	Register parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IPI3_AP_STATUS_reg_parity_error) Value After Reset: 0x0 Exists: Always

5.2.8.52 IPI3_FIFOCTRL_AP_STATUS

- **Name:** IPI3 FIFO control AP check status.
- **Description:** AP check status for IPI3 FIFO control block.
- **Size:** 32 bits
- **Offset:** 0x944
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI3_IF==1

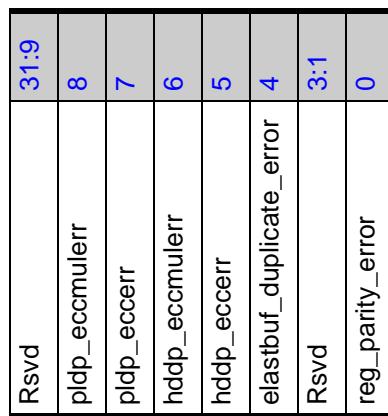


Table 5-192 Fields for Register: IPI3_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	pldp_eccmulerr	RC	multiple-bit error for IPI3 payload datapath with ecc Values: <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFOCTRL_AP_STATUS_pldp_eccmulerr) Value After Reset: 0x0 Exists: Always
7	pldp_eccerr	RC	single-bit error IPI3 payload datapath with ecc Values: <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFOCTRL_AP_STATUS_pldp_eccerr) Value After Reset: 0x0 Exists: Always
6	hddp_eccmulerr	RC	multiple-bit error for IPI3 header datapath with ecc Values: <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFOCTRL_AP_STATUS_hddp_eccmulerr) Value After Reset: 0x0 Exists: Always

Bits	Name	Memory Access	Description
5	hddp_eccerr	RC	<p>single-bit error IPI3 header datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFOCTRL_AP_STATUS_hddp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	elastbuf_duplicate_error	RC	<p>Duplicate module check error for the Elastbuf module inside the IPI3 FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFOCTRL_AP_STATUS_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_FIFOCTRL_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.53 IPI4_AP_STATUS

- **Name:** IPI4 AP check status.
- **Description:** AP check status for IPI4 block.
- **Size:** 32 bits
- **Offset:** 0x948
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI4_IF==1



Table 5-193 Fields for Register: IPI4_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	input_parity_error	RC	Input parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IPI4_AP_STATUS_input_parity_error) Value After Reset: 0x0 Exists: Always
1			Reserved Field: Yes
0	reg_parity_error	RC	Register parity check error Values: <ul style="list-style-type: none"> ■ 0x0 (IPI4_AP_STATUS_reg_parity_error) Value After Reset: 0x0 Exists: Always

5.2.8.54 IPI4_FIFOCTRL_AP_STATUS

- **Name:** IPI4 FIFO control AP check status.
- **Description:** AP check status for IPI4 FIFO control block.
- **Size:** 32 bits
- **Offset:** 0x94c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI4_IF==1

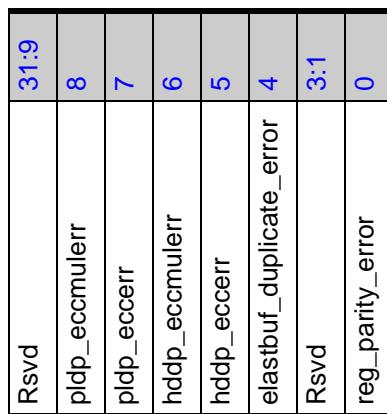


Table 5-194 Fields for Register: IPI4_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	pldp_eccmulerr	RC	multiple-bit error for IPI4 payload datapath with ecc Values: ■ 0x0 (IPI4_FIFOCTRL_AP_STATUS_pldp_eccmulerr) Value After Reset: 0x0 Exists: Always
7	pldp_eccerr	RC	single-bit error IPI4 payload datapath with ecc Values: ■ 0x0 (IPI4_FIFOCTRL_AP_STATUS_pldp_eccerr) Value After Reset: 0x0 Exists: Always
6	hddp_eccmulerr	RC	multiple-bit error for IPI4 header datapath with ecc Values: ■ 0x0 (IPI4_FIFOCTRL_AP_STATUS_hddp_eccmulerr) Value After Reset: 0x0 Exists: Always

Bits	Name	Memory Access	Description
5	hddp_eccerr	RC	<p>single-bit error IPI4 header datapath with ecc</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_FIFOCTRL_AP_STATUS_hddp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	elastbuf_duplicate_error	RC	<p>Duplicate module check error for the Elastbuf module inside the IPI4 FIFO control</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_FIFOCTRL_AP_STATUS_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_FIFOCTRL_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.55 SDI_AP_STATUS

- **Name:** SDI AP check status.
- **Description:** AP check status for SDI
- **Size:** 32 bits
- **Offset:** 0x950
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==1



Table 5-195 Fields for Register: SDI_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	input_parity_error	RC	<p>Input parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (SDI_AP_STATUS_input_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1			Reserved Field: Yes
0	reg_parity_error	RC	<p>Register parity check error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (SDI_AP_STATUS_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.56 MASK_N_IPI_AP_STATUS

- **Name:** IPI_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for IPI_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x9c0
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

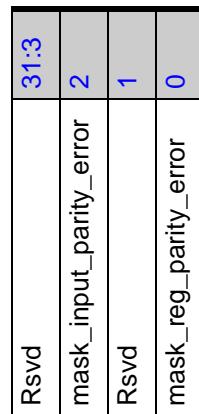


Table 5-196 Fields for Register: MASK_N_IPI_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_input_parity_error	R/W	<p>Mask for input_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI_AP_STATUS_mask_input_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>
1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.57 FORCE_IPI_AP_STATUS

- **Name:** IPI_AP_STATUS sources force.
- **Description:** Test register to force activation of IPI_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x9c4
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

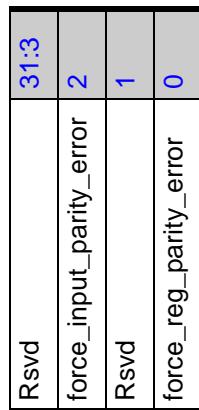


Table 5-197 Fields for Register: FORCE_IPI_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_input_parity_error	R/WC	<p>Force for input_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI_AP_STATUS_force_input_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>
1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.58 MASK_N_IDI_AP_STATUS

- **Name:** IDI_AP_STATUS interrupt mask
- **Description:** Interrupt mask for IDI_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x9c8
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

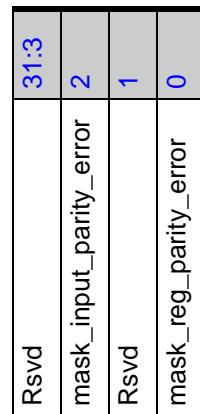


Table 5-198 Fields for Register: MASK_N_IDI_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_input_parity_error	R/W	<p>Mask for input_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IDI_AP_STATUS_mask_input_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>
1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IDI_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.59 FORCE_IDI_AP_STATUS

- **Name:** IDI_AP_STATUS sources force.
- **Description:** Test register to force activation of IDI_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x9cc
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

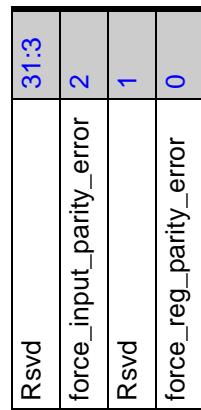


Table 5-199 Fields for Register: FORCE_IDI_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_input_parity_error	R/WC	<p>Force for input_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IDI_AP_STATUS_force_input_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IDI_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.60 MASK_N_AMBAAPBINTF_AP_STATUS

- **Name:** AMBAAPBINTF_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for AMBAAPBINTF_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x9d0
- **Exists:** CSI2_DEVICE_AP==1

31:3	Rsvd	mask_input_parity_error	mask_sig_pulse_error	mask_reg_parity_error
		2	1	0

Table 5-200 Fields for Register: MASK_N_AMBAAPBINTF_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_input_parity_error	R/W	<p>Mask for input_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_AMBAAPBINTF_AP_STATUS_mask_input_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>
1	mask_sig_pulse_error	R/W	<p>Mask for sig_pulse_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_AMBAAPBINTF_AP_STATUS_mask_sig_pulse_error) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (MASK_N_AMBAAPPINTF_AP_STA-TUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.61 FORCE_AMBAAPBINTF_AP_STATUS

- **Name:** AMBAAPBINTF_AP_STATUS sources FORCE.
- **Description:** Test register to force activation of AMBAAPBINTF_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x9d4
- **Exists:** CSI2_DEVICE_AP==1

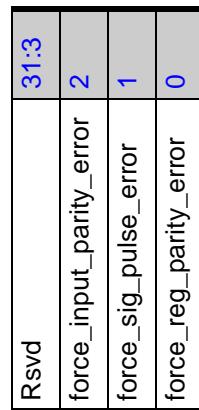


Table 5-201 Fields for Register: FORCE_AMBAAPBINTF_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_input_parity_error	R/WC	<p>Force for input_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_AMBAAPBINTF_AP_STATUS_force_input_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1	force_sig_pulse_error	R/WC	<p>Force for sig_pulse_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_AMBAAPBINTF_AP_STATUS_force_sig_pulse_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_AMBAAPBINTF_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.62 MASK_N_PHY_IF_CTRL_AP_STATUS

- **Name:** PHY_IF_CTRL_AP_STATUS interrupt mask
- **Description:** Interrupt mask for PHY_IF_CTRL_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x9d8
- **Exists:** CSI2_DEVICE_AP==1

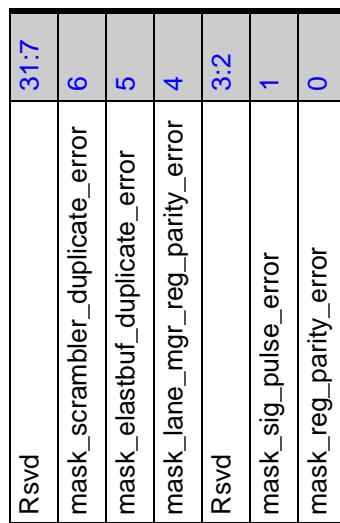


Table 5-202 Fields for Register: MASK_N_PHY_IF_CTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	mask_scrambler_duplicate_error	R/W	<p>Mask for scrambler_duplicate_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PHY_IF_CTRL_AP_STATUS_mask_scrambler_duplicate_error) <p>Value After Reset: 0x0 Exists: CSI2_DEVICE_DATA_SCRAMBLING==1</p>
5	mask_elastbuf_duplicate_error	R/W	<p>Mask for elastbuf_duplicate_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PHY_IF_CTRL_AP_STATUS_mask_elastbuf_duplicate_error) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
4	mask_lane_mgr_reg_parity_error	R/W	<p>Mask for lane_mgr_reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PHY_IF_CTRL_AP_STATUS_mask_lane_mgr_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:2			Reserved Field: Yes
1	mask_sig_pulse_error	R/W	<p>Mask for sig_pulse_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PHY_IF_CTRL_AP_STATUS_mask_sig_pulse_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PHY_IF_CTRL_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.63 FORCE_PHY_IF_CTRL_AP_STATUS

- **Name:** PHY_IF_CTRL_AP_STATUS sources force.
- **Description:** Test register to force activation of PHY_IF_CTRL_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x9dc
- **Exists:** CSI2_DEVICE_AP==1

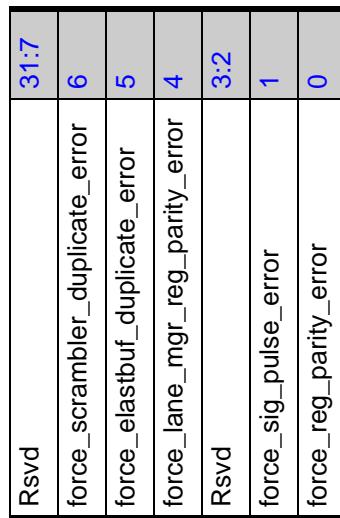


Table 5-203 Fields for Register: FORCE_PHY_IF_CTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:7			Reserved Field: Yes
6	force_scrambler_duplicate_error	R/WC	<p>Force for scrambler_duplicate_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PHY_IF_CTRL_AP_STATUS_force_scrambler_duplicate_error) <p>Value After Reset: 0x0 Exists: CSI2_DEVICE_DATA_SCRAMBLING==1</p>
5	force_elastbuf_duplicate_error	R/WC	<p>Force for elastbuf_duplicate_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PHY_IF_CTRL_AP_STATUS_force_elastbuf_duplicate_error) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
4	force_lane_mgr_reg_parity_error	R/WC	<p>Force for lane_mgr_reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PHY_IF_CTRL_AP_STATUS_force_lane_mgr_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:2			Reserved Field: Yes
1	force_sig_pulse_error	R/WC	<p>Force for sig_pulse_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PHY_IF_CTRL_AP_STATUS_force_sig_pulse_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PHY_IF_CTRL_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.64 MASK_N_REG_BANK_AP_STATUS

- **Name:** REG_BANK_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for REG_BANK_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x9e0
- **Exists:** CSI2_DEVICE_AP==1

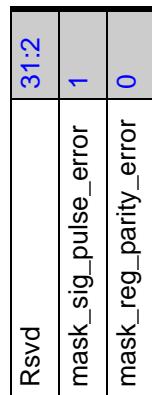


Table 5-204 Fields for Register: MASK_N_REG_BANK_AP_STATUS

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	mask_sig_pulse_error	R/W	<p>Mask for sig_pulse_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_REG_BANK_AP_STATUS_mask_sig_pulse_error) <p>Value After Reset: 0x0 Exists: Always</p>
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_REG_BANK_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.65 FORCE_REG_BANK_AP_STATUS

- **Name:** REG_BANK_AP_STATUS sources force.
- **Description:** Test register to force activation of REG_BANK_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x9e4
- **Exists:** CSI2_DEVICE_AP==1

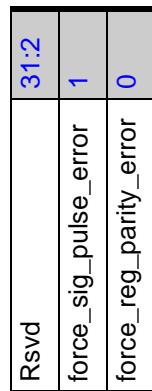


Table 5-205 Fields for Register: FORCE_REG_BANK_AP_STATUS

Bits	Name	Memory Access	Description
31:2			Reserved Field: Yes
1	force_sig_pulse_error	R/WC	<p>Force for sig_pulse_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_REG_BANK_AP_STATUS_force_sig_pulse_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_REG_BANK_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.66 MASK_N_IPI_FIFOCTRL_AP_STATUS

- **Name:** IPI_FIFOCTRL_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for IPI_FIFOCTRL_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x9e8
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

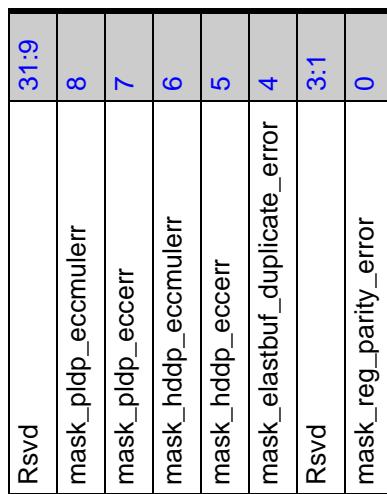


Table 5-206 Fields for Register: MASK_N_IPI_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	mask_pldp_eccmulerr	R/W	<p>Mask for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI_FIFOCTRL_AP_STATUS_mask_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	mask_pldp_eccerr	R/W	<p>Mask for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI_FIFOCTRL_AP_STATUS_mask_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	mask_hddp_eccmulerr	R/W	<p>Mask for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI_FIFOCTRL_AP_STATUS_mask_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_hddp_eccerr	R/W	<p>Mask for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI_FIFOCTRL_AP_STATUS_mask_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_elastbuf_duplicate_error	R/W	<p>Mask for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI_FIFOCTRL_AP_STA-TUS_mask_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI_FIFOCTRL_AP_STA-TUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.67 FORCE_IPI_FIFOCTRL_AP_STATUS

- **Name:** IPI_FIFOCTRL_AP_STATUS sources force.
- **Description:** Test register to force activation of IPI_FIFOCTRL_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x9ec
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

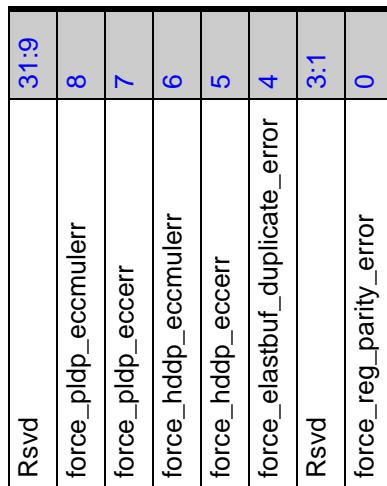


Table 5-207 Fields for Register: FORCE_IPI_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	force_pldp_eccmulerr	R/WC	<p>Force for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI_FIFOCTRL_AP_STATUS_force_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	force_pldp_eccerr	R/WC	<p>Force for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI_FIFOCTRL_AP_STATUS_force_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	force_hddp_eccmulerr	R/WC	<p>Force for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI_FIFOCTRL_AP_STATUS_force_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_hddp_eccerr	R/WC	<p>Force for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI_FIFOCTRL_AP_STATUS_force_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_elastbuf_duplicate_error	R/WC	<p>Force for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI_FIFOCTRL_AP_STATUS_force_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI_FIFOCTRL_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.68 MASK_N_IDI_FIFOCTRL_AP_STATUS

- **Name:** IDI_FIFOCTRL_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for IDI_FIFOCTRL_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x9f0
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

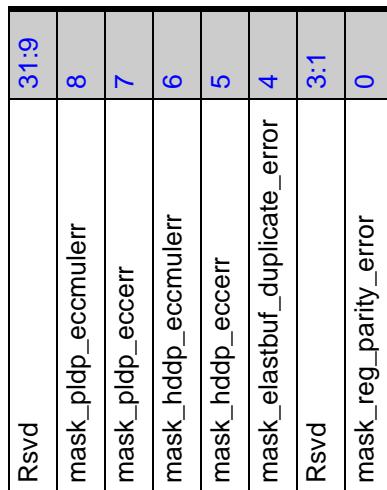


Table 5-208 Fields for Register: MASK_N_IDI_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	mask_pldp_eccmulerr	R/W	<p>Mask for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IDI_FIFOCTRL_AP_STATUS_mask_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	mask_pldp_eccerr	R/W	<p>Mask for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IDI_FIFOCTRL_AP_STATUS_mask_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	mask_hddp_eccmulerr	R/W	<p>Mask for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IDI_FIFOCTRL_AP_STATUS_mask_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_hddp_eccerr	R/W	<p>Mask for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IDI_FIFOCTRL_AP_STATUS_mask_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_elastbuf_duplicate_error	R/W	<p>Mask for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IDI_FIFOCTRL_AP_STA-TUS_mask_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IDI_FIFOCTRL_AP_STA-TUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.69 FORCE_IDI_FIFOCTRL_AP_STATUS

- **Name:** IDI_FIFOCTRL_AP_STATUS sources force.
- **Description:** Test register to force activation of IDI_FIFOCTRL_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x9f4
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

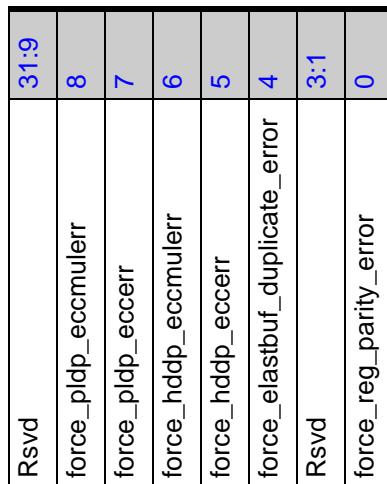


Table 5-209 Fields for Register: FORCE_IDI_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	force_pldp_eccmulerr	R/WC	<p>Force for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IDI_FIFOCTRL_AP_STATUS_force_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	force_pldp_eccerr	R/WC	<p>Force for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IDI_FIFOCTRL_AP_STATUS_force_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	force_hddp_eccmulerr	R/WC	<p>Force for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IDI_FIFOCTRL_AP_STATUS_force_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_hddp_eccerr	R/WC	<p>Force for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IDI_FIFOCTRL_AP_STATUS_force_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_elastbuf_duplicate_error	R/WC	<p>Force for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IDI_FIFOCTRL_AP_STATUS_force_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IDI_FIFOCTRL_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.70 MASK_N_PKT_BUILDER_AP_STATUS

- **Name:** PKT_BUILDER_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for PKT_BUILDER_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0x9f8
- **Exists:** CSI2_DEVICE_AP==1

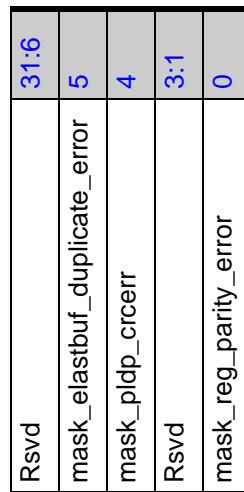


Table 5-210 Fields for Register: MASK_N_PKT_BUILDER_AP_STATUS

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	mask_elastbuf_duplicate_error	R/W	<p>Mask for elastbuf_duplicate_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PKT_BUILDER_AP_STATUS_mask_elastbuf_duplicate_error) <p>Value After Reset: 0x0 Exists: Always</p>
4	mask_pldp_crcerr	R/W	<p>Mask for pldp_crcerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PKT_BUILDER_AP_STATUS_mask_pldp_crcerr) <p>Value After Reset: 0x0 Exists: Always</p>
3:1			Reserved Field: Yes

Bits	Name	Memory Access	Description
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (MASK_N_PKT_BUILDER_AP_STA-TUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.71 FORCE_PKT_BUILDER_AP_STATUS

- **Name:** PKT_BUILDER_AP_STATUS sources force.
- **Description:** Test register to force activation of PKT_BUILDER_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0x9fc
- **Exists:** CSI2_DEVICE_AP==1

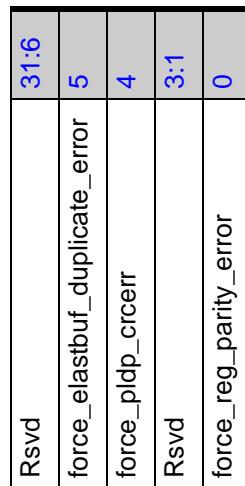


Table 5-211 Fields for Register: FORCE_PKT_BUILDER_AP_STATUS

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	force_elastbuf_duplicate_error	R/WC	<p>Force for elastbuf_duplicate_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PKT_BUILDER_AP_STATUS_force_elastbuf_duplicate_error) <p>Value After Reset: 0x0 Exists: Always</p>
4	force_pldp_crcerr	R/WC	<p>Force for pldp_crcerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PKT_BUILDER_AP_STATUS_force_pldp_crcerr) <p>Value After Reset: 0x0 Exists: Always</p>
3:1			Reserved Field: Yes

Bits	Name	Memory Access	Description
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (FORCE_PKT_BUILDER_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.72 MASK_N_ERR_HANDLER_AP_STATUS

- **Name:** ERR_HANDLER_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for ERR_HANDLER_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked. Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa00
- **Exists:** CSI2_DEVICE_AP==1

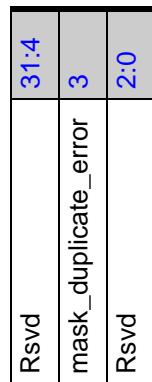


Table 5-212 Fields for Register: MASK_N_ERR_HANDLER_AP_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	mask_duplicate_error	R/W	<p>Mask for duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_ERR_HANDLER_AP_STATUS_mask_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2:0			Reserved Field: Yes

5.2.8.73 FORCE_ERR_HANDLER_AP_STATUS

- **Name:** ERR_HANDLER_AP_STATUS sources force.
- **Description:** Test register to force activation of ERR_HANDLER_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa04
- **Exists:** CSI2_DEVICE_AP==1

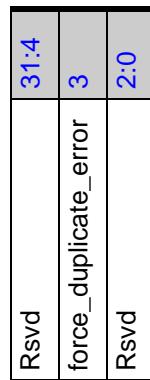


Table 5-213 Fields for Register: FORCE_ERR_HANDLER_AP_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	force_duplicate_error	R/WC	<p>Force for duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_ERR_HANDLER_AP_STATUS_force_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2:0			Reserved Field: Yes

5.2.8.74 MASK_N_SYNC_AP_STATUS

- **Name:** SYNC_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for SYNC_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa08
- **Exists:** CSI2_DEVICE_AP==1

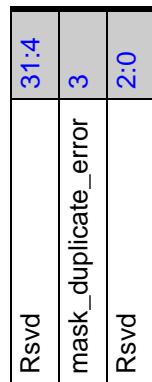


Table 5-214 Fields for Register: MASK_N_SYNC_AP_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	mask_duplicate_error	R/W	<p>Mask for duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_SYNC_AP_STATUS_mask_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2:0			Reserved Field: Yes

5.2.8.75 FORCE_SYNC_AP_STATUS

- **Name:** SYNC_AP_STATUS sources force.
- **Description:** Test register to force activation of SYNC_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa0c
- **Exists:** CSI2_DEVICE_AP==1

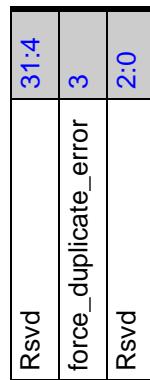


Table 5-215 Fields for Register: FORCE_SYNC_AP_STATUS

Bits	Name	Memory Access	Description
31:4			Reserved Field: Yes
3	force_duplicate_error	R/WC	<p>Force for duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_SYNC_AP_STATUS_force_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
2:0			Reserved Field: Yes

5.2.8.76 MASK_N_PKT_IF_AP_STATUS

- **Name:** PKT_IF_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for PKT_IF_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa10
- **Exists:** CSI2_DEVICE_AP==1

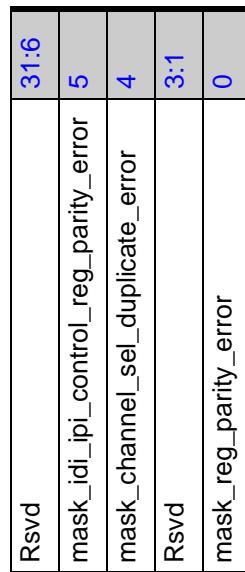


Table 5-216 Fields for Register: MASK_N_PKT_IF_AP_STATUS

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	mask_idi_ipi_control_reg_parity_error	R/W	<p>Mask for idi_ipi_control_reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PKT_IF_AP_STATUS_mask_idi_ipi_control_reg_parity_error) <p>Value After Reset: 0x0 Exists: CSI2_DEVICE_IDI_IPI_IF == 1</p>
4	mask_channel_sel_duplicate_error	R/W	<p>Mask for channel_sel_duplicate_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_PKT_IF_AP_STATUS_mask_channel_sel_duplicate_error) <p>Value After Reset: 0x0 Exists: Always</p>
3:1			Reserved Field: Yes

Bits	Name	Memory Access	Description
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (MASK_N_PKT_IF_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.77 FORCE_PKT_IF_AP_STATUS

- **Name:** PKT_IF_AP_STATUS sources force.
- **Description:** Test register to force activation of PKT_IF_AP_STATUS sources.1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa14
- **Exists:** CSI2_DEVICE_AP==1

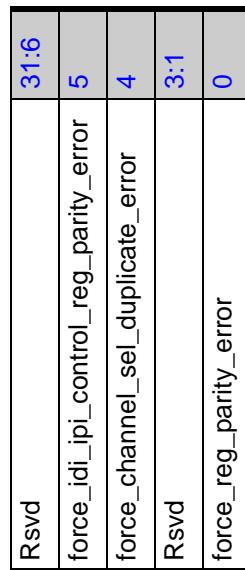


Table 5-217 Fields for Register: FORCE_PKT_IF_AP_STATUS

Bits	Name	Memory Access	Description
31:6			Reserved Field: Yes
5	force_idi_ipi_control_reg_parity_error	R/WC	<p>Force for idi_ipi_control_reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PKT_IF_AP_STATUS_force_idi_ipi_control_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_IDI_IPI_IF == 1</p>
4	force_channel_sel_duplicate_error	R/WC	<p>Force for channel_sel_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_PKT_IF_AP_STATUS_force_channel_sel_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes

Bits	Name	Memory Access	Description
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none">■ 0x0 (FORCE_PKT_IF_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.78 MASK_N_ECF_AP_STATUS

- **Name:** ECF_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for ECF_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa18
- **Exists:** CSI2_DEVICE_AP==1



Table 5-218 Fields for Register: MASK_N_ECF_AP_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_ECF_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.79 FORCE_ECF_AP_STATUS

- **Name:** ECF_AP_STATUS sources force.
- **Description:** Test register to force activation of ECF_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa1c
- **Exists:** CSI2_DEVICE_AP==1



Table 5-219 Fields for Register: FORCE_ECF_AP_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_ECF_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.80 MASK_N_CMU_AP_STATUS

- **Name:** CMU_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for CMU_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa20
- **Exists:** CSI2_DEVICE_AP==1



Table 5-220 Fields for Register: MASK_N_CMU_AP_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_CMU_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.81 FORCE_CMU_AP_STATUS

- **Name:** CMU_AP_STATUS sources force.
- **Description:** Test register to force activation of CMU_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa24
- **Exists:** CSI2_DEVICE_AP==1

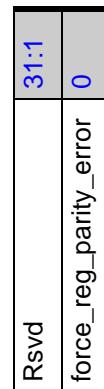


Table 5-221 Fields for Register: FORCE_CMU_AP_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_CMU_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.82 MASK_N_MT_IPI_CONTROL_AP_STATUS

- **Name:** MT_IPI_CONTROL_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for MT_IPI_CONTROL_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa28
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

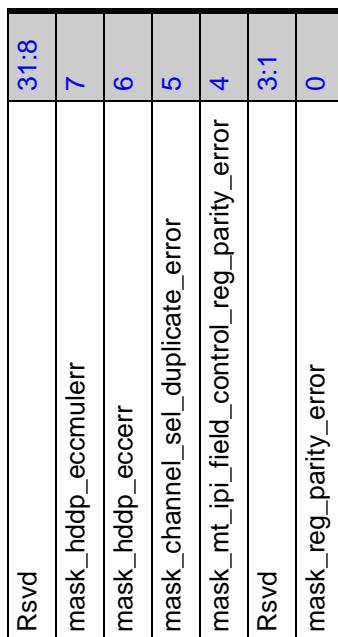


Table 5-222 Fields for Register: MASK_N_MT_IPI_CONTROL_AP_STATUS

Bits	Name	Memory Access	Description
31:8			Reserved Field: Yes
7	mask_hddp_eccmulerr	R/W	<p>Mask for hddp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_MT_IPI_CONTROL_AP_STATUS_mask_hddp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	mask_hddp_eccerr	R/W	<p>Mask for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_MT_IPI_CONTROL_AP_STATUS_mask_hddp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_channel_sel_duplicate_error	R/W	<p>Mask for channel_sel_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_MT_IPI_CONTROL_AP_STATUS_mask_channel_sel_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_mt_ipi_field_control_reg_parity_error	R/W	<p>Mask for mt_ipi_field_control_reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_MT_IPI_CONTROL_AP_STATUS_mask_mt_ipi_field_control_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_MT_IPI_CONTROL_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.83 FORCE_MT_IPI_CONTROL_AP_STATUS

- **Name:** MT_IPI_CONTROL_AP_STATUS sources force.
- **Description:** Test register to force activation of MT_IPI_CONTROL_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa2c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

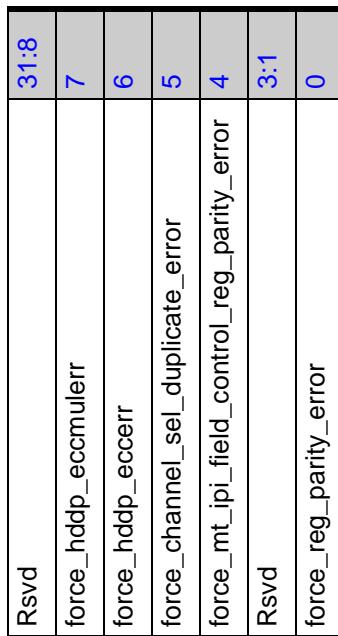


Table 5-223 Fields for Register: FORCE_MT_IPI_CONTROL_AP_STATUS

Bits	Name	Memory Access	Description
31:8			Reserved Field: Yes
7	force_hddp_eccmulerr	R/WC	<p>Force for hddp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_MT_IPI_CONTROL_AP_STATUS_force_hddp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
6	force_hddp_eccerr	R/WC	<p>Force for hddp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_MT_IPI_CONTROL_AP_STATUS_force_hddp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
5	force_channel_sel_duplicate_error	R/WC	<p>Force for channel_sel_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_MT_IPI_CONTROL_AP_STATUS_force_channel_sel_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_mt_ipi_field_control_reg_parity_error	R/WC	<p>Force for mt_ipi_field_control_reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_MT_IPI_CONTROL_AP_STATUS_force_mt_ipi_field_control_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_MT_IPI_CONTROL_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.84 MASK_N_IPI2_AP_STATUS

- **Name:** IPI2_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for IPI2_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa30
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

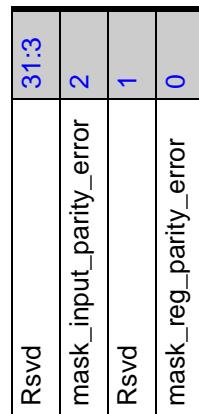


Table 5-224 Fields for Register: MASK_N_IPI2_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_input_parity_error	R/W	<p>Mask for input_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI2_AP_STATUS_mask_input_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI2_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.85 FORCE_IPI2_AP_STATUS

- **Name:** IPI2_AP_STATUS sources force.
- **Description:** Test register to force activation of IPI2_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa34
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

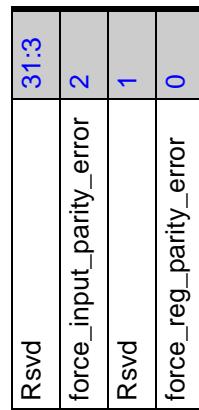


Table 5-225 Fields for Register: FORCE_IPI2_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_input_parity_error	R/WC	<p>Force for input_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI2_AP_STATUS_force_input_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>
1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI2_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.86 MASK_N_IPI2_FIFOCTRL_AP_STATUS

- **Name:** IPI2_FIFOCTRL_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for IPI2_FIFOCTRL_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa38
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

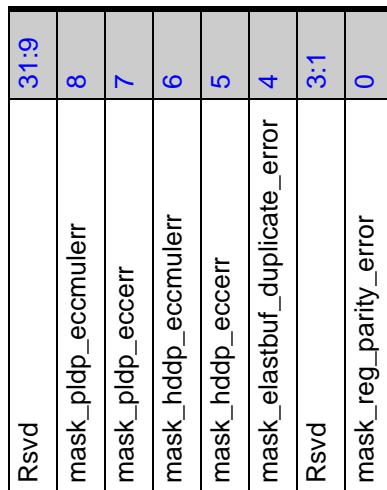


Table 5-226 Fields for Register: MASK_N_IPI2_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	mask_pldp_eccmulerr	R/W	<p>Mask for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI2_FIFOCTRL_AP_STATUS_mask_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	mask_pldp_eccerr	R/W	<p>Mask for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI2_FIFOCTRL_AP_STATUS_mask_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	mask_hddp_eccmulerr	R/W	<p>Mask for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI2_FIFOCTRL_AP_STATUS_mask_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_hddp_eccerr	R/W	<p>Mask for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI2_FIFOCTRL_AP_STATUS_mask_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_elastbuf_duplicate_error	R/W	<p>Mask for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI2_FIFOCTRL_AP_STA-TUS_mask_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI2_FIFOCTRL_AP_STA-TUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.87 FORCE_IPI2_FIFOCTRL_AP_STATUS

- **Name:** IPI2_FIFOCTRL_AP_STATUS sources force.
- **Description:** Test register to force activation of IPI2_FIFOCTRL_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa3c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

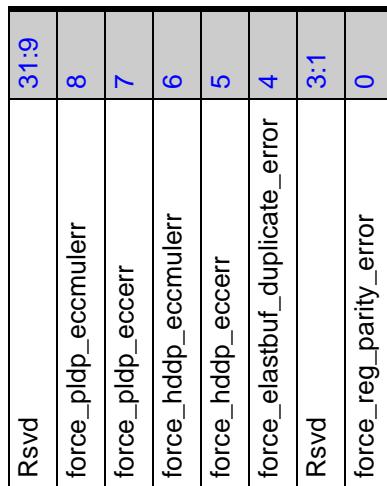


Table 5-227 Fields for Register: FORCE_IPI2_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	force_pldp_eccmulerr	R/WC	<p>Force for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI2_FIFOCTRL_AP_STATUS_force_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	force_pldp_eccerr	R/WC	<p>Force for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI2_FIFOCTRL_AP_STATUS_force_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	force_hddp_eccmulerr	R/WC	<p>Force for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI2_FIFOCTRL_AP_STATUS_force_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_hddp_eccerr	R/WC	<p>Force for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI2_FIFOCTRL_AP_STATUS_force_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_elastbuf_duplicate_error	R/WC	<p>Force for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI2_FIFOCTRL_AP_STATUS_force_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI2_FIFOCTRL_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.88 MASK_N_IPI3_AP_STATUS

- **Name:** IPI3_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for IPI3_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa40
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI3_IF==1

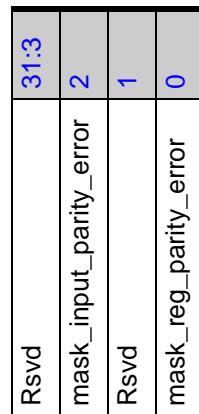


Table 5-228 Fields for Register: MASK_N_IPI3_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_input_parity_error	R/W	<p>Mask for input_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI3_AP_STATUS_mask_input_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>
1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI3_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.89 FORCE_IPI3_AP_STATUS

- **Name:** IPI3_AP_STATUS sources force.
- **Description:** Test register to force activation of IPI3_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa44
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI3_IF==1

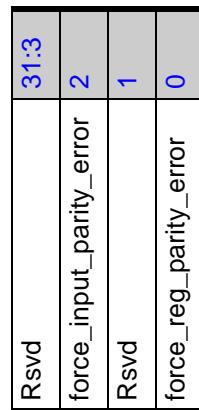


Table 5-229 Fields for Register: FORCE_IPI3_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_input_parity_error	R/WC	<p>Force for input_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI3_AP_STATUS_force_input_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI3_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.90 MASK_N_IPI3_FIFOCTRL_AP_STATUS

- **Name:** IPI3_FIFOCTRL_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for IPI3_FIFOCTRL_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa48
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI3_IF==1

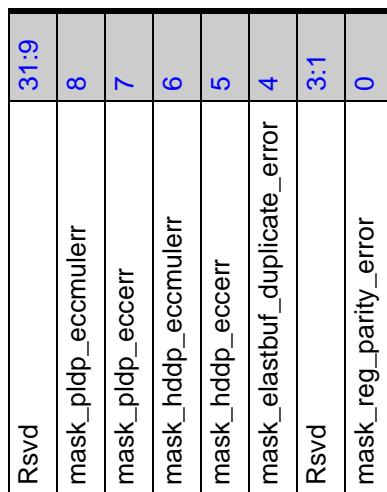


Table 5-230 Fields for Register: MASK_N_IPI3_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	mask_pldp_eccmulerr	R/W	<p>Mask for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI3_FIFOCTRL_AP_STATUS_mask_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	mask_pldp_eccerr	R/W	<p>Mask for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI3_FIFOCTRL_AP_STATUS_mask_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	mask_hddp_eccmulerr	R/W	<p>Mask for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI3_FIFOCTRL_AP_STATUS_mask_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_hddp_eccerr	R/W	<p>Mask for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI3_FIFOCTRL_AP_STATUS_mask_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_elastbuf_duplicate_error	R/W	<p>Mask for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI3_FIFOCTRL_AP_STA-TUS_mask_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI3_FIFOCTRL_AP_STA-TUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.91 FORCE_IPI3_FIFOCTRL_AP_STATUS

- **Name:** IPI3_FIFOCTRL_AP_STATUS sources force.
- **Description:** Test register to force activation of IPI3_FIFOCTRL_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa4c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI3_IF==1

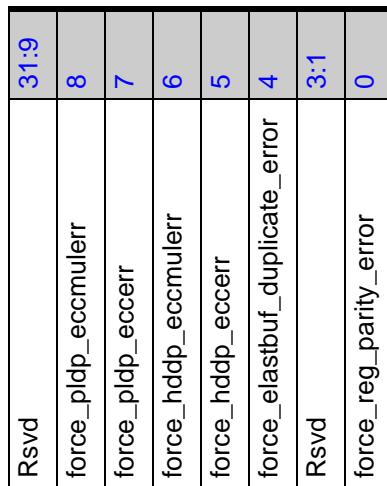


Table 5-231 Fields for Register: FORCE_IPI3_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	force_pldp_eccmulerr	R/WC	<p>Force for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI3_FIFOCTRL_AP_STATUS_force_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	force_pldp_eccerr	R/WC	<p>Force for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI3_FIFOCTRL_AP_STATUS_force_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	force_hddp_eccmulerr	R/WC	<p>Force for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI3_FIFOCTRL_AP_STATUS_force_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_hddp_eccerr	R/WC	<p>Force for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI3_FIFOCTRL_AP_STATUS_force_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_elastbuf_duplicate_error	R/WC	<p>Force for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI3_FIFOCTRL_AP_STATUS_force_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI3_FIFOCTRL_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.92 MASK_N_IPI4_AP_STATUS

- **Name:** IPI4_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for IPI4_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa50
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI4_IF==1

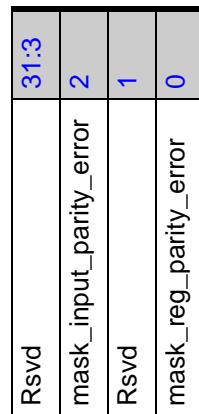


Table 5-232 Fields for Register: MASK_N_IPI4_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_input_parity_error	R/W	<p>Mask for input_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI4_AP_STATUS_mask_input_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI4_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.93 FORCE_IPI4_AP_STATUS

- **Name:** IPI4_AP_STATUS sources force.
- **Description:** Test register to force activation of IPI4_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa54
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI4_IF==1

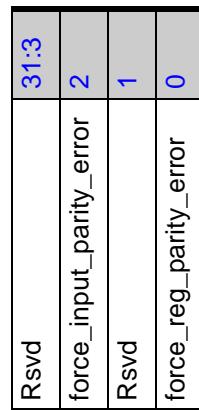


Table 5-233 Fields for Register: FORCE_IPI4_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_input_parity_error	R/WC	<p>Force for input_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI4_AP_STATUS_force_input_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI4_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.94 MASK_N_IPI4_FIFOCTRL_AP_STATUS

- **Name:** IPI4_FIFOCTRL_AP_STATUS interrupt mask.
 - **Description:** Interrupt mask for IPI4_FIFOCTRL_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
 - **Size:** 32 bits
 - **Offset:** 0xa58
 - **Exists:** CSI2 DEVICE AP==1&&CSI2 DEVICE IPI4 IF==1

Rsvd	31:9
mask_pldp_eccmulerr	8
mask_pldp_eccerr	7
mask_hdcp_eccmulerr	6
mask_hdcp_eccerr	5
mask_elastbuf_duplicate_error	4
Rsvd	3:1
mask_reg_parity_error	0

Table 5-234 Fields for Register: MASK_N_IPI4_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	mask_pldp_eccmulerr	R/W	Mask for pldp_eccmulerr
			Values: <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI4_FIFOCTRL_AP_STATUS_mask_pldp_eccmulerr)
			Value After Reset: 0x0
			Exists: Always
7	mask_pldp_eccerr	R/W	Mask for pldp_eccerr
			Values: <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI4_FIFOCTRL_AP_STATUS_mask_pldp_eccerr)
			Value After Reset: 0x0
			Exists: Always

Bits	Name	Memory Access	Description
6	mask_hddp_eccmulerr	R/W	<p>Mask for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI4_FIFOCTRL_AP_STATUS_mask_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	mask_hddp_eccerr	R/W	<p>Mask for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI4_FIFOCTRL_AP_STATUS_mask_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	mask_elastbuf_duplicate_error	R/W	<p>Mask for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI4_FIFOCTRL_AP_STA-TUS_mask_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_IPI4_FIFOCTRL_AP_STA-TUS_mask_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.95 FORCE_IPI4_FIFOCTRL_AP_STATUS

- **Name:** IPI4_FIFOCTRL_AP_STATUS sources force.
- **Description:** Test register to force activation of IPI4_FIFOCTRL_AP_STATUS sources. 1: Force internal AP error source.0: No action. Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa5c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI4_IF==1

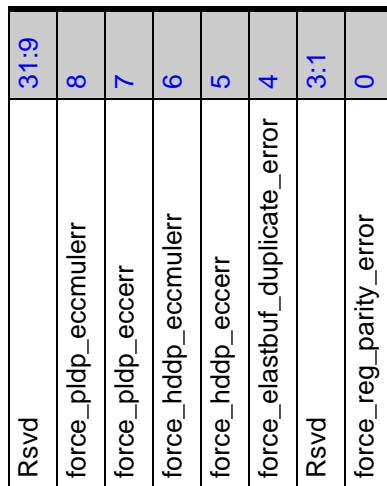


Table 5-235 Fields for Register: FORCE_IPI4_FIFOCTRL_AP_STATUS

Bits	Name	Memory Access	Description
31:9			Reserved Field: Yes
8	force_pldp_eccmulerr	R/WC	<p>Force for pldp_eccmulerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI4_FIFOCTRL_AP_STATUS_force_pldp_eccmulerr) <p>Value After Reset: 0x0 Exists: Always</p>
7	force_pldp_eccerr	R/WC	<p>Force for pldp_eccerr Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI4_FIFOCTRL_AP_STATUS_force_pldp_eccerr) <p>Value After Reset: 0x0 Exists: Always</p>

Bits	Name	Memory Access	Description
6	force_hddp_eccmulerr	R/WC	<p>Force for hddp_eccmulerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI4_FIFOCTRL_AP_STATUS_force_hd-dp_eccmulerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
5	force_hddp_eccerr	R/WC	<p>Force for hddp_eccerr</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI4_FIFOCTRL_AP_STATUS_force_hd-dp_eccerr) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
4	force_elastbuf_duplicate_error	R/WC	<p>Force for elastbuf_duplicate_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI4_FIFOCTRL_AP_STATUS_force_elastbuf_duplicate_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3:1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_IPI4_FIFOCTRL_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.8.96 MASK_N_SD1_AP_STATUS

- **Name:** SDI_AP_STATUS interrupt mask.
- **Description:** Interrupt mask for SDI_AP_STATUS; controls which AP error status bit triggers the interrupt pin.1: Enable the AP error source.0: AP error source is masked.Static read and write register.
- **Size:** 32 bits
- **Offset:** 0xa60
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SD1_IF==1

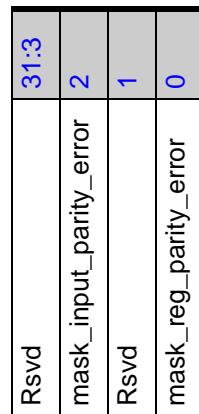


Table 5-236 Fields for Register: MASK_N_SD1_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	mask_input_parity_error	R/W	<p>Mask for input_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_SD1_AP_STATUS_mask_input_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>
1			Reserved Field: Yes
0	mask_reg_parity_error	R/W	<p>Mask for reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (MASK_N_SD1_AP_STATUS_mask_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.8.97 FORCE_SDI_AP_STATUS

- **Name:** SDI_AP_STATUS sources force.
- **Description:** Test register to force activation of SDI_AP_STATUS sources. 1: Force internal AP error source.0: No action.Writing this register clears it.
- **Size:** 32 bits
- **Offset:** 0xa64
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==1

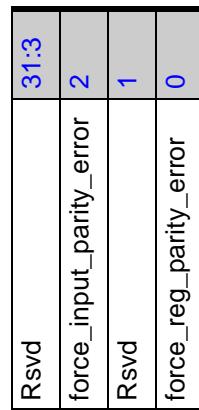


Table 5-237 Fields for Register: FORCE_SDI_AP_STATUS

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2	force_input_parity_error	R/WC	<p>Force for input_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_SDI_AP_STATUS_force_input_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>
1			Reserved Field: Yes
0	force_reg_parity_error	R/WC	<p>Force for reg_parity_error Values:</p> <ul style="list-style-type: none"> ■ 0x0 (FORCE_SDI_AP_STATUS_force_reg_parity_error) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9 DWC_mipicsi2_device_MemMap/AP Registers

5.2.9.1 ERR_INJ_CTRL

- **Name:** Error injection control.
- **Description:** Controls the memory error injection.
- **Size:** 32 bits
- **Offset:** 0xb00
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0

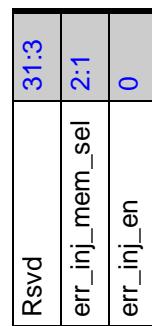


Table 5-238 Fields for Register: ERR_INJ_CTRL

Bits	Name	Memory Access	Description
31:3			Reserved Field: Yes
2:1	err_inj_mem_sel	R/W	<p>This field indicates the selected memory to inject the error: When IDI and IPI are both present</p> <ul style="list-style-type: none"> ■ 11: Reserved ■ 10: Reserved ■ 01: Select the ipi memory to inject the error. ■ 00: Select the idi memory to inject the error. <p>When multiple IPIs are present.</p> <ul style="list-style-type: none"> ■ 11: Select the ipi4 memory to inject the error. ■ 10: Select the ipi3 memory to inject the error. ■ 01: Select the ipi2 memory to inject the error. ■ 00: Select the ipi memory to inject the error <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_CTRL_err_inj_mem_sel) <p>Value After Reset: 0x0</p> <p>Exists: CSI2_DEVICE_MULTI_IF==1</p>
0	err_inj_en	R/WC	<p>Memory error Injection enable</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_CTRL_err_inj_en) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.2 ERR_INJ_STATUS

- **Name:** Error injection status.
- **Description:** Indicates the error injection status.
- **Size:** 32 bits
- **Offset:** 0xb04
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0

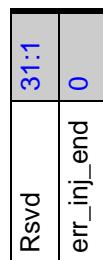


Table 5-239 Fields for Register: ERR_INJ_STATUS

Bits	Name	Memory Access	Description
31:1			Reserved Field: Yes
0	err_inj_end	RC	<p>Memory error Inject start status (autoclear): Bit-0 indicates the memory injection hasnt ended or err_inj_en is low. Bit-1 indicates the memory injection has ended.</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_STATUS_err_inj_end) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.3 ERR_INJ_CHK_MASK

- **Name:** Check error injection mask.
- **Description:** The injection mask of the error in Check bits.
- **Size:** 32 bits
- **Offset:** 0xb08
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0



Table 5-240 Fields for Register: ERR_INJ_CHK_MASK

Bits	Name	Memory Access	Description
31:10			Reserved Field: Yes
9:0	err_inj_chk_mask	R/W	<p>Error injection mask in check bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_CHK_MASK_err_inj_chk_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.4 ERR_INJ_D31_0_MASK

- **Name:** Data 0-31 bit error injection mask.
- **Description:** The injection mask of the error in data 0-31bits.
- **Size:** 32 bits
- **Offset:** 0xb0c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0



Table 5-241 Fields for Register: ERR_INJ_D31_0_MASK

Bits	Name	Memory Access	Description
31:0	err_inj_d31_0_mask	R/W	<p>Error injection mask in data 0-31bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_D31_0_MASK_err_inj_d31_0_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.5 ERR_INJ_D63_32_MASK

- **Name:** Data 32-63bit error injection mask.
- **Description:** The injection mask of the error in data 32-63bits.
- **Size:** 32 bits
- **Offset:** 0xb10
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0



Table 5-242 Fields for Register: ERR_INJ_D63_32_MASK

Bits	Name	Memory Access	Description
31:0	err_inj_d63_32_mask	R/W	<p>Error injection mask in data 32-63bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_D63_32_MASK_err_inj_d63_32_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.6 ERR_INJ_D95_64_MASK

- **Name:** Data 64-95bit error injection mask.
- **Description:** The injection mask of the error in data 64-95bits.
- **Size:** 32 bits
- **Offset:** 0xb14
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0&&CSI2_DEVICE_PL_ERR_INJ_WIDTH_GT_64==1



Table 5-243 Fields for Register: ERR_INJ_D95_64_MASK

Bits	Name	Memory Access	Description
31:0	err_inj_d95_64_mask	R/W	<p>Error injection mask in data 64-95bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_D95_64_MASK_err_inj_d95_64_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.7 ERR_INJ_D127_96_MASK

- **Name:** Data 96-127bit error injection mask.
- **Description:** The injection mask of the error in data 96-127bits.
- **Size:** 32 bits
- **Offset:** 0xb18
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0&&CSI2_DEVICE_PL_ERR_INJ_WIDTH_GT_64==1



Table 5-244 Fields for Register: ERR_INJ_D127_96_MASK

Bits	Name	Memory Access	Description
31:0	err_inj_d127_96_mask	R/W	<p>Error injection mask in data 96-127bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_D127_96_MASK_err_inj_d127_96_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.8 ERR_INJ_D159_128_MASK

- **Name:** Data 128-159bit error injection mask.
- **Description:** The injection mask of the error in data 128-159bits.
- **Size:** 32 bits
- **Offset:** 0xb1c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0&&CSI2_DEVICE_PL_ERR_INJ_WIDTH_GT_128==1



Table 5-245 Fields for Register: ERR_INJ_D159_128_MASK

Bits	Name	Memory Access	Description
31:0	err_inj_d159_128_mask	R/W	<p>Error injection mask in data 128-159bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_D159_128_MASK_err_inj_d159_128_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.9 ERR_INJ_D191_160_MASK

- **Name:** Data 160-191bit error injection mask.
- **Description:** The injection mask of the error in data 160-191bits.
- **Size:** 32 bits
- **Offset:** 0xb20
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0&&CSI2_DEVICE_PL_ERR_INJ_WIDTH_GT_128==1



Table 5-246 Fields for Register: ERR_INJ_D191_160_MASK

Bits	Name	Memory Access	Description
31:0	err_inj_d191_160_mask	R/W	<p>Error injection mask in data 160-191bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_D191_160_MASK_err_inj_d191_160_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.10 ERR_INJ_D223_192_MASK

- **Name:** Data 192-223bit error injection mask.
- **Description:** The injection mask of the error in data 192-223bits.
- **Size:** 32 bits
- **Offset:** 0xb24
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0&&CSI2_DEVICE_PL_ERR_INJ_WIDTH_GT_128==1



Table 5-247 Fields for Register: ERR_INJ_D223_192_MASK

Bits	Name	Memory Access	Description
31:0	err_inj_d223_192_mask	R/W	<p>Error injection mask in data 192-223bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_D223_192_MASK_err_inj_d223_192_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.11 ERR_INJ_D255_224_MASK

- **Name:** Data 224-255bit error injection mask.
- **Description:** The injection mask of the error in data 224-255bits.
- **Size:** 32 bits
- **Offset:** 0xb28
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_SDI_IF==0&&CSI2_DEVICE_PL_ERR_INJ_WIDTH_GT_128==1



Table 5-248 Fields for Register: ERR_INJ_D255_224_MASK

Bits	Name	Memory Access	Description
31:0	err_inj_d255_224_mask	R/W	<p>Error injection mask in data 224-255bits</p> <p>Values:</p> <ul style="list-style-type: none"> ■ 0x0 (ERR_INJ_D255_224_MASK_err_inj_d255_224_mask) <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

5.2.9.12 IDI_RAM_ERR_LOG_AP

- **Name:** IDI logged error RAM addresses number.
- **Description:** This register stores the number of faulty RAM addresses in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb2c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

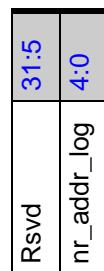


Table 5-249 Fields for Register: IDI_RAM_ERR_LOG_AP

Bits	Name	Memory Access	Description
31:5			Reserved Field: Yes
4:0	nr_addr_log	R	<p>Number of logged addresses with errors. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IDI_RAM_ERR_LOG_AP_nr_addr_log) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.13 IDI_RAM_ERR_ADDR_AP

- **Name:** IDI first error RAM address.
- **Description:** This register stores the first faulty Ram address in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb30
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IDI_IF==1

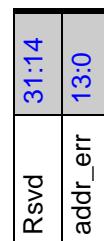


Table 5-250 Fields for Register: IDI_RAM_ERR_ADDR_AP

Bits	Name	Memory Access	Description
31:14			Reserved Field: Yes
13:0	addr_err	R	<p>Address of the first error. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IDI_RAM_ERR_ADDR_AP_addr_err) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.14 IPI_RAM_ERR_LOG_AP

- **Name:** IPI logged error RAM addresses number.
- **Description:** This register stores the number of faulty RAM addresses in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb34
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1

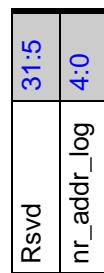


Table 5-251 Fields for Register: IPI_RAM_ERR_LOG_AP

Bits	Name	Memory Access	Description
31:5			Reserved Field: Yes
4:0	nr_addr_log	R	<p>Number of logged addresses with errors. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_RAM_ERR_LOG_AP_nr_addr_log) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.15 IPI_RAM_ERR_ADDR_AP

- **Name:** IPI first error RAM address.
- **Description:** This register stores the first faulty Ram address in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb38
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI_IF==1



Table 5-252 Fields for Register: IPI_RAM_ERR_ADDR_AP

Bits	Name	Memory Access	Description
31:14			Reserved Field: Yes
13:0	addr_err	R	<p>Address of the first error. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI_RAM_ERR_ADDR_AP_addr_err) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.16 IPI2_RAM_ERR_LOG_AP

- **Name:** IPI2 logged error RAM addresses number.
- **Description:** This register stores the number of faulty RAM addresses in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb3c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

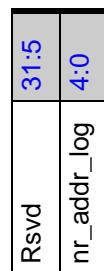


Table 5-253 Fields for Register: IPI2_RAM_ERR_LOG_AP

Bits	Name	Memory Access	Description
31:5			Reserved Field: Yes
4:0	nr_addr_log	R	<p>Number of logged addresses with errors. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_RAM_ERR_LOG_AP_nr_addr_log) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.17 IPI2_RAM_ERR_ADDR_AP

- **Name:** IPI2 first error RAM address.
- **Description:** This register stores the first faulty Ram address in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb40
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI2_IF==1

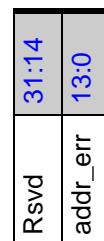


Table 5-254 Fields for Register: IPI2_RAM_ERR_ADDR_AP

Bits	Name	Memory Access	Description
31:14			Reserved Field: Yes
13:0	addr_err	R	<p>Address of the first error. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI2_RAM_ERR_ADDR_AP_addr_err) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.18 IPI3_RAM_ERR_LOG_AP

- **Name:** IPI3 logged error RAM addresses number.
- **Description:** This register stores the number of faulty RAM addresses in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb44
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI3_IF==1

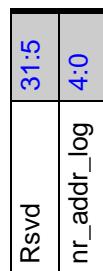


Table 5-255 Fields for Register: IPI3_RAM_ERR_LOG_AP

Bits	Name	Memory Access	Description
31:5			Reserved Field: Yes
4:0	nr_addr_log	R	<p>Number of logged addresses with errors. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_RAM_ERR_LOG_AP_nr_addr_log) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.19 IPI3_RAM_ERR_ADDR_AP

- **Name:** IPI3 first error RAM address.
- **Description:** This register stores the first faulty Ram address in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb48
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI3_IF==1

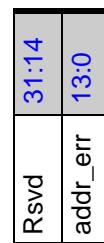


Table 5-256 Fields for Register: IPI3_RAM_ERR_ADDR_AP

Bits	Name	Memory Access	Description
31:14			Reserved Field: Yes
13:0	addr_err	R	<p>Address of the first error. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI3_RAM_ERR_ADDR_AP_addr_err) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.20 IPI4_RAM_ERR_LOG_AP

- **Name:** IPI4 logged error RAM addresses number.
- **Description:** This register stores the number of faulty RAM addresses in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb4c
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI4_IF==1

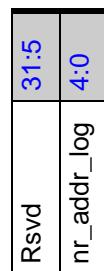


Table 5-257 Fields for Register: IPI4_RAM_ERR_LOG_AP

Bits	Name	Memory Access	Description
31:5			Reserved Field: Yes
4:0	nr_addr_log	R	<p>Number of logged addresses with errors. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_RAM_ERR_LOG_AP_nr_addr_log) <p>Value After Reset: 0x0 Exists: Always</p>

5.2.9.21 IPI4_RAM_ERR_ADDR_AP

- **Name:** IPI4 first error RAM address.
- **Description:** This register stores the first faulty Ram address in FMEDA mode.
- **Size:** 32 bits
- **Offset:** 0xb50
- **Exists:** CSI2_DEVICE_AP==1&&CSI2_DEVICE_IPI4_IF==1

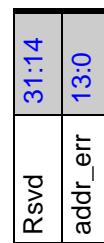


Table 5-258 Fields for Register: IPI4_RAM_ERR_ADDR_AP

Bits	Name	Memory Access	Description
31:14			Reserved Field: Yes
13:0	addr_err	R	<p>Address of the first error. Values:</p> <ul style="list-style-type: none"> ■ 0x0 (IPI4_RAM_ERR_ADDR_AP_addr_err) <p>Value After Reset: 0x0 Exists: Always</p>

5.3 DW_safety_slave_map Memory Map Registers

Register definitions for each component memory map.

Table 5-259 Registers for the DW_safety_slave_map Memory Map

Register	Offset	Description
This register block defines the Safety Slave register set DW_safety_slave_block Exists: Always		
"SS_STRT_COREOFF_REG" on page 642	0x0	Core Register Start Offset Register
"SS_CNTRL_REG" on page 643	0x4	Safety Slave Control Register
"SS_RESET_REG" on page 646	0x8	Safety Slave Reset Register
"SS_ERRSRC_REG" on page 647	0xc	Safety Error Source Register
"SS_ERRFORCE_REG" on page 649	0x10	Safety Error Force Register
"SS_DIAG0_RERRSTS_REG" on page 650	0x14	Safety Slave Recoverable Error Status Register
"SS_DIAG1_RERRSTS_REG" on page 651	0x18	Safety Slave Recoverable Error Status Register
"SS_CORE_VERNUM_REG" on page 652	0x20	Safety Slave Core Version Number Register
"SS_CORE_VERTYPE_REG" on page 653	0x24	Safety Slave Core Version Type Register
"SS_CORE_URERRSTS_REG" on page 654	0x28	Safety Slave Core Unrecoverable Error Status Register
"SS_CORE_RERRSTS_REG" on page 655	0x2c	Safety Slave Core Recoverable Error Status Register
"SS_CORE_DIAG_REG0" on page 656	0x30	Core Diagnostic Register0
"SS_CORE_DIAG_REG1" on page 657	0x34	Core Diagnostic Register1
"SS_CORE_DIAG_REG2" on page 658	0x38	Core Diagnostic Register2
"SS_CORE_DIAG_REG3" on page 659	0x3c	Core Diagnostic Register3
"SS_CORE_DIAG_REG4" on page 660	0x40	Core Diagnostic Register4
"SS_CORE_DIAG_REG5" on page 661	0x44	Core Diagnostic Register5
"SS_CORE_DIAG_REG6" on page 662	0x48	Core Diagnostic Register6
"SS_CORE_DIAG_REG7" on page 663	0x4c	Core Diagnostic Register7
"SS_CORE_DIAG_REG8" on page 664	0x50	Core Diagnostic Register8
"SS_CORE_DIAG_REG9" on page 665	0x54	Core Diagnostic Register9
"SS_CORE_DIAG_REG10" on page 666	0x58	Core Diagnostic Register10
"SS_CORE_DIAG_REG11" on page 667	0x5c	Core Diagnostic Register11
"SS_CORE_DIAG_REG12" on page 668	0x60	Core Diagnostic Register12

Register	Offset	Description
"SS_CORE_DIAG_REG13" on page 669	0x64	Core Diagnostic Register13
"SS_CORE_DIAG_REG14" on page 670	0x68	Core Diagnostic Register14
"SS_CORE_DIAG_REG15" on page 671	0x6c	Core Diagnostic Register15
"SS_CORE_DIAG_REG16" on page 672	0x70	Core Diagnostic Register16
"SS_CORE_DIAG_REG17" on page 673	0x74	Core Diagnostic Register17
"SS_CORE_DIAG_REG18" on page 674	0x78	Core Diagnostic Register18
"SS_CORE_DIAG_REG19" on page 675	0x7c	Core Diagnostic Register19
"SS_CORE_DIAG_REG20" on page 676	0x80	Core Diagnostic Register20
"SS_CORE_DIAG_REG21" on page 677	0x84	Core Diagnostic Register21
"SS_CORE_DIAG_REG22" on page 678	0x88	Core Diagnostic Register22
"SS_CORE_DIAG_REG23" on page 679	0x8c	Core Diagnostic Register23
"SS_CORE_DIAG_REG24" on page 680	0x90	Core Diagnostic Register24
"SS_CORE_DIAG_REG25" on page 681	0x94	Core Diagnostic Register25
"SS_CORE_DIAG_REG26" on page 682	0x98	Core Diagnostic Register26
"SS_CORE_DIAG_REG27" on page 683	0x9c	Core Diagnostic Register27
"SS_CORE_DIAG_REG28" on page 684	0xa0	Core Diagnostic Register28
"SS_CORE_DIAG_REG29" on page 685	0xa4	Core Diagnostic Register29
"SS_CORE_DIAG_REG30" on page 686	0xa8	Core Diagnostic Register30
"SS_CORE_DIAG_REG31" on page 687	0xac	Core Diagnostic Register31
"SS_CORE_DIAG_REG32" on page 688	0xb0	Core Diagnostic Register32
"SS_CORE_DIAG_REG33" on page 689	0xb4	Core Diagnostic Register33
"SS_CORE_DIAG_REG34" on page 690	0xb8	Core Diagnostic Register34
"SS_CORE_DIAG_REG35" on page 691	0xbc	Core Diagnostic Register35
"SS_CORE_DIAG_REG36" on page 692	0xc0	Core Diagnostic Register36
"SS_CORE_DIAG_REG37" on page 693	0xc4	Core Diagnostic Register37
"SS_CORE_DIAG_REG38" on page 694	0xc8	Core Diagnostic Register38
"SS_CORE_DIAG_REG39" on page 695	0xcc	Core Diagnostic Register39
"SS_CORE_DIAG_REG40" on page 696	0xd0	Core Diagnostic Register40
"SS_CORE_DIAG_REG41" on page 697	0xd4	Core Diagnostic Register41

Register	Offset	Description
"SS_CORE_DIAG_REG42" on page 698	0xd8	Core Diagnostic Register42

5.3.1 DW_safety_slave_map/DW_safety_slave_block Registers

5.3.1.1 SS_STRT_COREOFF_REG

- **Name:** Core Register Start Offset Register
- **Description:** This register is used to indicate the size of the core registers and start address of the core register map.
- **Size:** 32 bits
- **Offset:** 0x0
- **Exists:** Always

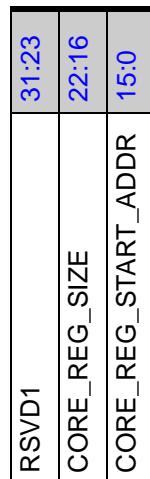


Table 5-260 Fields for Register: SS_STRT_COREOFF_REG

Bits	Name	Memory Access	Description
31:23	RSVD1	R	Reserved and read as zero. Value After Reset: 0x0 Exists: Always
22:16	CORE_REG_SIZE	R	Core Register Size, this field returns the configured value of the core register depth. The SS ignores all writes to this register Value After Reset: SS_CORE_REG_DEPTH Exists: Always Volatile: true
15:0	CORE_REG_START_ADDR	R	Core Register Start Pointer, this field returns the configured value of the core register start offset. The SS ignores all writes to this register Value After Reset: SS_CORE_REG_START_OFFSET Exists: Always Volatile: true

5.3.1.2 SS_CNTRL_REG

- **Name:** Safety Slave Control Register
- **Description:** This register is the safety slave control register. It has control signals to enable error checks, and controls for injecting different errors.
- **Size:** 32 bits
- **Offset:** 0x4
- **Exists:** Always

SS_INT_RERR_SET	31:30
RSVD	29:14
SS_DIAG1_RERR_SET	13:12
SS_DIAG0_RERR_SET	11:10
CORE_URERR_SET	9:8
CORE_RERR_SET	7:6
CORE_URMON_EN	5:4
CORE_RMON_EN	3:2
SS_EMON_EN	1:0

Table 5-261 Fields for Register: SS_CNTRL_REG

Bits	Name	Memory Access	Description
31:30	SS_INT_RERR_SET	R/W	<p>Inject SS Internal Error When enabled by the application, this will facilitate in injecting ss recoverable error to test the functionality 2'b10 - Disabled 2'b01 - Enabled 2'b00, 2'b11 - Reserved On setting this field with either 2'b00 or 2'b11 no write to the register will happen, but the transfer will be completed on APB side. When this field flips to any value from the enabled state or disabled state (when the application is not writing), then "safety error" will get generated, this error will get reflected in the SS_ERRSRC_R[SS_INT_ERR_SRC] bit.</p> <p>Value After Reset: 0x2</p> <p>Exists: Always</p> <p>Volatile: true</p>
29:14	RSVD	R	<p>Reserved and read as zero.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p>

Bits	Name	Memory Access	Description
13:12	SS_DIAG1_RERR_SET	R/W	<p>Inject SS Diagnostic1 Recoverable Error When enabled by the application, this will facilitate in injecting ss recoverable diagnostic1 error status (SS_DIAG1_RERRSTS_REG) to test the functionality 2'b10 - Disabled 2'b01 - Enabled 2'b00, 2'b11 - Reserved On setting this field with either 2'b00 or 2'b11 no write to the register will happen, but the transfer will be completed on APB side. When this field flips to any value from the enabled state or disabled state (when the application is not writing), then "safety error" will get generated, this error will get reflected in the SS_ERRSRC_R[SS_INT_ERR_SRC] bit.</p> <p>Value After Reset: 0x2</p> <p>Exists: Always</p> <p>Volatile: true</p>
11:10	SS_DIAG0_RERR_SET	R/W	<p>Inject SS Diagnostic0 Recoverable Error When enabled by the application, this will facilitate in injecting ss recoverable diagnostic0 error status (SS_DIAG0_RERRSTS_REG) to test the functionality 2'b10 - Disabled 2'b01 - Enabled 2'b00, 2'b11 - Reserved On setting this field with either 2'b00 or 2'b11 no write to the register will happen, but the transfer will be completed on APB side. When this field flips to any value from the enabled state or disabled state (when the application is not writing), then "safety error" will get generated, this error will get reflected in the SS_ERRSRC_R[SS_INT_ERR_SRC] bit.</p> <p>Value After Reset: 0x2</p> <p>Exists: Always</p> <p>Volatile: true</p>
9:8	CORE_URERR_SET	R/W	<p>Inject Unrecoverable Error When enabled by the application, this will facilitate in injecting unrecoverable error to test the functionality 2'b10 - Disabled 2'b01 - Enabled 2'b00, 2'b11 - Reserved When this field flips to any value from the enabled state or disabled state (when the application is not writing), then "safety error" will get generated.</p> <p>Value After Reset: 0x2</p> <p>Exists: Always</p> <p>Volatile: true</p>
7:6	CORE_RERR_SET	R/W	<p>Inject Recoverable Error When enabled by the application, this will facilitate in injecting recoverable error to test the functionality 2'b10 - Disabled 2'b01 - Enabled 2'b00, 2'b11 - Reserved When this field flips to any value from the enabled state or disabled state (when the application is not writing), then "safety error" will get generated.</p> <p>Value After Reset: 0x2</p> <p>Exists: Always</p> <p>Volatile: true</p>

Bits	Name	Memory Access	Description
5:4	CORE_URMON_EN	R/W	<p>Unrecoverable Error Monitor Enable When enabled by the application, SS monitors for unrecoverable errors generated by the core. Current design of the Safety Slave does not generate unrecoverable errors. 2'b10 - Disabled 2'b01 - Enabled 2'b00, 2'b11 - Reserved When this field flips to any value from the enabled state or disabled state (when the application is not writing), then "safety error" will get generated.</p> <p>Value After Reset: 0x2</p> <p>Exists: Always</p> <p>Volatile: true</p>
3:2	CORE_RMON_EN	R/W	<p>Recoverable Error Monitor Enable When enabled by the application, SS monitors for recoverable errors generated both by the Safety Slave and the core. 2'b10 - Disabled 2'b01 - Enabled 2'b00, 2'b11 - Reserved When this field flips to any value from the enabled state or disabled state (when the application is not writing), then "ss_safety_error" will get generated.</p> <p>Value After Reset: 0x2</p> <p>Exists: Always</p> <p>Volatile: true</p>
1:0	SS_EMON_EN	R/W	<p>Safety Slave Error Monitor Enable When enabled by the safety manager, the safety slave will start monitoring for errors in the SS_CNTRL_REG register and the following port inputs signals.</p> <ul style="list-style-type: none"> ■ core_swreset ■ core_enable ■ core_version_num_reg ■ core_version_type_reg ■ core_diagnostic_reg0-63 <p>2'b10 - Disabled 2'b01 - Enabled 2'b00, 2'b11 - Reserved Writing 2'b00 or 2'b11 from SSM bus will be ignored. When this field flips to any value from the enabled state or disabled state due to any internal fault conditions, then "ss_safety_error" will get generated.</p> <p>Value After Reset: 0x2</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.3 SS_RESET_REG

- **Name:** Safety Slave Reset Register
- **Description:** This register is the safety slave reset register. It has control signals to clear the error status registers.
- **Size:** 32 bits
- **Offset:** 0x8
- **Exists:** Always

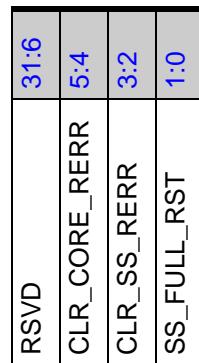


Table 5-262 Fields for Register: SS_RESET_REG

Bits	Name	Memory Access	Description
31:6	RSVD	R	Reserved and read as zero. Value After Reset: 0x0 Exists: Always
5:4	CLR_CORE_RERR	R/W	Clear Core Recoverable Error Clear the core recoverable error set in the SS Error Status register Value After Reset: 0x2 Exists: Always Volatile: true
3:2	CLR_SS_RERR	R/W	Clear Safety Slave Recoverable Error Clear the SS recoverable error set in the SS Error Status register Value After Reset: 0x2 Exists: Always Volatile: true
1:0	SS_FULL_RST	R/W	Safety Slave Full Reset When enabled the Safety Slave is reset. Value After Reset: 0x2 Exists: Always Volatile: true

5.3.1.4 SS_ERRSRC_REG

- **Name:** Safety Error Source Register
- **Description:** This register is the safety slave error source register. It shows the source of the safety slave error.
- **Size:** 32 bits
- **Offset:** 0xc
- **Exists:** Always

Rsvd	31:26
CORE_ENABLE_ERR_SRC	25
CORE_SWRST_ERR_SRC	24
RSVD1	23:9
SS_INT_RERR_SRC	8
RSVD	7:4
SS_DIAG1_RERR_SRC	3
SS_DIAG0_RERR_SRC	2
CORE_UERR_MON_SRC	1
CORE_RERR_MON_SRC	0

Table 5-263 Fields for Register: SS_ERRSRC_REG

Bits	Name	Memory Access	Description
31:26			Reserved Field: Yes
25	CORE_ENABLE_ERR_SRC	R	Core Enable Port Error This bit will get set when the SS encounters an error in the core_enable port input. Value After Reset: 0x0 Exists: Always Volatile: true
24	CORE_SWRST_ERR_SRC	R	Core SW Reset Port Error This bit will get set when the SS encounters an error in the core_swreset port input. Value After Reset: 0x0 Exists: Always Volatile: true
23:9	RSVD1	R	Reserved and read as zero. Value After Reset: 0x0 Exists: Always

Bits	Name	Memory Access	Description
8	SS_INT_RERR_SRC	R	<p>Safety Slave Internal Recoverable Error This bit will get set when the SS encounters an error in any one of the following register or port input signals. 1. core_version_num_reg 2. core_version_type_reg 3. SS_CNTRL_REG</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>
7:4	RSVD	R	<p>Reserved and read as zero.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p>
3	SS_DIAG1_RERR_SRC	R	<p>SS Recoverable Diagnostic1 Error Source This bit is set when the SS encounters ss_safety_err due to SS recoverable error i.e. parity mismatch on diagnostic regisetrs 32-64</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>
2	SS_DIAG0_RERR_SRC	R	<p>SS Recoverable Diagnostic0 Error Source This bit is set when the SS encounters ss_safety_err due to SS recoverable error i.e. parity mismatch on diagnostic regisetrs 0-31</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>
1	CORE_URERR_MON_SRC	R	<p>Core Unrecoverable Monitor Error Source This bit will get set when the SS encounters an error in the core_urerr_mon port input.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>
0	CORE_RERR_MON_SRC	R	<p>Core Recoverable Monitor Error Source This bit will get set when the SS encounters an error in the core_rerr_mon port input.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.5 SS_ERRFORCE_REG

- **Name:** Safety Error Force Register
- **Description:** This register is the safety slave error force register.
- **Size:** 32 bits
- **Offset:** 0x10
- **Exists:** Always



Table 5-264 Fields for Register: SS_ERRFORCE_REG

Bits	Name	Memory Access	Description
31:0	SS_ERR_FORCE_VAL	W	<p>Safety Slave Error Force Write the register with "1" (multiple 1s are allowed) for the supported recoverable error monitor size (SS_CORE_RERR_MON_SIZE) when register field SS_CTRN_CTRL_REG[CORE_RERR_SET] is to `SAFETY_HI for forcing Core Recoverable Error Or Write the register with "1" (multiple 1s are allowed) for the supported unrecoverable error monitor size (SS_CORE_URERR_MON_SIZE) when the register field SS_CTRN_CTRL_REG[CORE_URERR_SET] to `SAFETY_HI for forcing Core Unrecoverable Error Or Write the register SS_ERRFORCE_REG with 0x00000001 when the register field SS_CTRN_CTRL_REG[SS_RERR_SET] to `SAFETY_HI for forcing Safety slave Recoverable Error This register always returns 0 when read.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.6 SS_DIAG0_RERRSTS_REG

- **Name:** Safety Slave Recoverable Error Status Register
- **Description:** This register is the safety slave recoverable error status register. It shows the status of parity error on Diagnostic register 0-31
- **Size:** 32 bits
- **Offset:** 0x14
- **Exists:** Always



Table 5-265 Fields for Register: SS_DIAG0_RERRSTS_REG

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG0_ERR_STS	R	<p>Core Diagnostic0 Error Status If the SS encounters any parity error (if configured for) in the diagnostic register port input set from core_diagnostic_reg0 to core_diagnostic_reg31, then this register captures the error status. For example if any parity error is encountered in core_diagnostic_reg15, then bit15 of this register will get set.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.7 SS_DIAG1_RERRSTS_REG

- **Name:** Safety Slave Recoverable Error Status Register
- **Description:** This register is the safety slave recoverable error status register. It shows the status of parity error on Diagnostic register 32-63
- **Size:** 32 bits
- **Offset:** 0x18
- **Exists:** Always



Table 5-266 Fields for Register: SS_DIAG1_RERRSTS_REG

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG1_ERR_STS	R	<p>Core Diagnostic1 Error Status If the SS encounters any parity error (if configured for) in the diagnostic register port input set from core_diagnostic_reg32 to core_diagnostic_reg63, then this register captures the error status. For example if any parity error is encountered in core_diagnostic_reg47, then bit15 of this register will get set.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.8 SS_CORE_VERNUM_REG

- **Name:** Safety Slave Core Version Number Register
- **Description:** This register is the core version number register.
- **Size:** 32 bits
- **Offset:** 0x20
- **Exists:** Always



Table 5-267 Fields for Register: SS_CORE_VERNUM_REG

Bits	Name	Memory Access	Description
31:0	SS_CORE_VERNUM	R	<p>Core i/p core_version_num_reg is captured in this register</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.9 SS_CORE_VERTYPE_REG

- **Name:** Safety Slave Core Version Type Register
- **Description:** This register is the core version type register.
- **Size:** 32 bits
- **Offset:** 0x24
- **Exists:** Always



Table 5-268 Fields for Register: SS_CORE_VERTYPE_REG

Bits	Name	Memory Access	Description
31:0	SS_CORE_VERTYPE	R	<p>Core i/p core_version_type_reg is captured in this register</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.10 SS_CORE_URERRSTS_REG

- **Name:** Safety Slave Core Unrecoverable Error Status Register
- **Description:** This register reflects the Unrecoverable error status
- **Size:** 32 bits
- **Offset:** 0x28
- **Exists:** Always



Table 5-269 Fields for Register: SS_CORE_URERRSTS_REG

Bits	Name	Memory Access	Description
31:0	SS_CORE_URERR_STS	R	<p>If the SS encounters either `SAFETY_HI or any invalid value on any one of the anti-valent pair of the core_urerr_mon (only up to configured value SS_CORE_URERR_MON_SIZE) input port, then it sets corresponding bit of the SS_CORE_URERRSTS_REG register.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.11 SS_CORE_RERRSTS_REG

- **Name:** Safety Slave Core Recoverable Error Status Register
- **Description:** This register reflects the Recoverable error status
- **Size:** 32 bits
- **Offset:** 0x2c
- **Exists:** Always



Table 5-270 Fields for Register: SS_CORE_RERRSTS_REG

Bits	Name	Memory Access	Description
31:0	SS_CORE_RERR_STS	R	<p>If the SS encounters either `SAFETY_HI or any invalid value on any one of the anti-valent pair of the core_rerr_mon (only up to configured value SS_CORE_RERR_MON_SIZE) input port, then it sets corresponding bit of the SS_CORE_RERRSTS_REG register.</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.12 SS_CORE_DIAG_REG0

- **Name:** Core Diagnostic Register0
- **Description:** This is Core Diagnostic Register0
- **Size:** 32 bits
- **Offset:** 0x30
- **Exists:** Always



Table 5-271 Fields for Register: SS_CORE_DIAG_REG0

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG0	R	<p>This register stores the diagnostic register0 from the core Value After Reset: 0x0 Exists: Always Volatile: true</p>

5.3.1.13 SS_CORE_DIAG_REG1

- **Name:** Core Diagnostic Register1
- **Description:** This is Core Diagnostic Register1
- **Size:** 32 bits
- **Offset:** 0x34
- **Exists:** Always



Table 5-272 Fields for Register: SS_CORE_DIAG_REG1

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG1	R	<p>This register stores the diagnostic register1 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.14 SS_CORE_DIAG_REG2

- **Name:** Core Diagnostic Register2
- **Description:** This is Core Diagnostic Register2
- **Size:** 32 bits
- **Offset:** 0x38
- **Exists:** Always

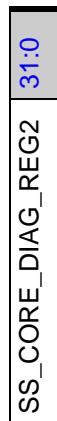


Table 5-273 Fields for Register: SS_CORE_DIAG_REG2

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG2	R	<p>This register stores the diagnostic register2 from the core Value After Reset: 0x0 Exists: Always Volatile: true</p>

5.3.1.15 SS_CORE_DIAG_REG3

- **Name:** Core Diagnostic Register3
- **Description:** This is Core Diagnostic Register3
- **Size:** 32 bits
- **Offset:** 0x3c
- **Exists:** Always



Table 5-274 Fields for Register: SS_CORE_DIAG_REG3

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG3	R	<p>This register stores the diagnostic register3 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.16 SS_CORE_DIAG_REG4

- **Name:** Core Diagnostic Register4
- **Description:** This is Core Diagnostic Register4
- **Size:** 32 bits
- **Offset:** 0x40
- **Exists:** Always

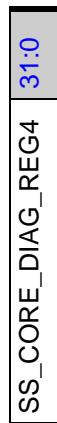


Table 5-275 Fields for Register: SS_CORE_DIAG_REG4

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG4	R	<p>This register stores the diagnostic register4 from the core Value After Reset: 0x0 Exists: Always Volatile: true</p>

5.3.1.17 SS_CORE_DIAG_REG5

- **Name:** Core Diagnostic Register5
- **Description:** This is Core Diagnostic Register5
- **Size:** 32 bits
- **Offset:** 0x44
- **Exists:** Always



Table 5-276 Fields for Register: SS_CORE_DIAG_REG5

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG5	R	<p>This register stores the diagnostic register5 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.18 SS_CORE_DIAG_REG6

- **Name:** Core Diagnostic Register6
- **Description:** This is Core Diagnostic Register6
- **Size:** 32 bits
- **Offset:** 0x48
- **Exists:** Always

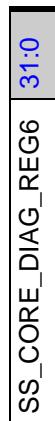


Table 5-277 Fields for Register: SS_CORE_DIAG_REG6

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG6	R	This register stores the diagnostic register6 from the core Value After Reset: 0x0 Exists: Always Volatile: true

5.3.1.19 SS_CORE_DIAG_REG7

- **Name:** Core Diagnostic Register7
- **Description:** This is Core Diagnostic Register7
- **Size:** 32 bits
- **Offset:** 0x4c
- **Exists:** Always



Table 5-278 Fields for Register: SS_CORE_DIAG_REG7

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG7	R	<p>This register stores the diagnostic register7 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.20 SS_CORE_DIAG_REG8

- **Name:** Core Diagnostic Register8
- **Description:** This is Core Diagnostic Register8
- **Size:** 32 bits
- **Offset:** 0x50
- **Exists:** Always

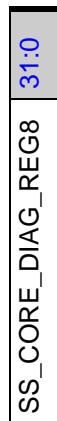


Table 5-279 Fields for Register: SS_CORE_DIAG_REG8

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG8	R	<p>This register stores the diagnostic register8 from the core Value After Reset: 0x0 Exists: Always Volatile: true</p>

5.3.1.21 SS_CORE_DIAG_REG9

- **Name:** Core Diagnostic Register9
- **Description:** This is Core Diagnostic Register9
- **Size:** 32 bits
- **Offset:** 0x54
- **Exists:** Always



Table 5-280 Fields for Register: SS_CORE_DIAG_REG9

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG9	R	<p>This register stores the diagnostic register9 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.22 SS_CORE_DIAG_REG10

- **Name:** Core Diagnostic Register10
- **Description:** This is Core Diagnostic Register10
- **Size:** 32 bits
- **Offset:** 0x58
- **Exists:** Always



Table 5-281 Fields for Register: SS_CORE_DIAG_REG10

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG10	R	<p>This register stores the diagnostic register10 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.23 SS_CORE_DIAG_REG11

- **Name:** Core Diagnostic Register11
- **Description:** This is Core Diagnostic Register11
- **Size:** 32 bits
- **Offset:** 0x5c
- **Exists:** Always



Table 5-282 Fields for Register: SS_CORE_DIAG_REG11

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG11	R	<p>This register stores the diagnostic register11 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.24 SS_CORE_DIAG_REG12

- **Name:** Core Diagnostic Register12
- **Description:** This is Core Diagnostic Register12
- **Size:** 32 bits
- **Offset:** 0x60
- **Exists:** Always

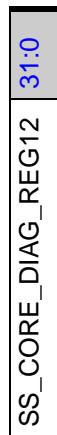


Table 5-283 Fields for Register: SS_CORE_DIAG_REG12

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG12	R	<p>This register stores the diagnostic register12 from the core Value After Reset: 0x0 Exists: Always Volatile: true</p>

5.3.1.25 SS_CORE_DIAG_REG13

- **Name:** Core Diagnostic Register13
- **Description:** This is Core Diagnostic Register13
- **Size:** 32 bits
- **Offset:** 0x64
- **Exists:** Always



Table 5-284 Fields for Register: SS_CORE_DIAG_REG13

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG13	R	<p>This register stores the diagnostic register13 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.26 SS_CORE_DIAG_REG14

- **Name:** Core Diagnostic Register14
- **Description:** This is Core Diagnostic Register14
- **Size:** 32 bits
- **Offset:** 0x68
- **Exists:** Always



Table 5-285 Fields for Register: SS_CORE_DIAG_REG14

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG14	R	<p>This register stores the diagnostic register14 from the core Value After Reset: 0x0 Exists: Always Volatile: true</p>

5.3.1.27 SS_CORE_DIAG_REG15

- **Name:** Core Diagnostic Register15
- **Description:** This is Core Diagnostic Register15
- **Size:** 32 bits
- **Offset:** 0x6c
- **Exists:** Always



Table 5-286 Fields for Register: SS_CORE_DIAG_REG15

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG15	R	<p>This register stores the diagnostic register15 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.28 SS_CORE_DIAG_REG16

- **Name:** Core Diagnostic Register16
- **Description:** This is Core Diagnostic Register16
- **Size:** 32 bits
- **Offset:** 0x70
- **Exists:** Always



Table 5-287 Fields for Register: SS_CORE_DIAG_REG16

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG16	R	<p>This register stores the diagnostic register16 from the core Value After Reset: 0x0 Exists: Always Volatile: true</p>

5.3.1.29 SS_CORE_DIAG_REG17

- **Name:** Core Diagnostic Register17
- **Description:** This is Core Diagnostic Register17
- **Size:** 32 bits
- **Offset:** 0x74
- **Exists:** Always



Table 5-288 Fields for Register: SS_CORE_DIAG_REG17

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG17	R	<p>This register stores the diagnostic register17 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.30 SS_CORE_DIAG_REG18

- **Name:** Core Diagnostic Register18
- **Description:** This is Core Diagnostic Register18
- **Size:** 32 bits
- **Offset:** 0x78
- **Exists:** Always



Table 5-289 Fields for Register: SS_CORE_DIAG_REG18

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG18	R	<p>This register stores the diagnostic register18 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.31 SS_CORE_DIAG_REG19

- **Name:** Core Diagnostic Register19
- **Description:** This is Core Diagnostic Register19
- **Size:** 32 bits
- **Offset:** 0x7c
- **Exists:** Always



Table 5-290 Fields for Register: SS_CORE_DIAG_REG19

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG19	R	<p>This register stores the diagnostic register19 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.32 SS_CORE_DIAG_REG20

- **Name:** Core Diagnostic Register20
- **Description:** This is Core Diagnostic Register20
- **Size:** 32 bits
- **Offset:** 0x80
- **Exists:** Always



Table 5-291 Fields for Register: SS_CORE_DIAG_REG20

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG20	R	<p>This register stores the diagnostic register20 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.33 SS_CORE_DIAG_REG21

- **Name:** Core Diagnostic Register21
- **Description:** This is Core Diagnostic Register21
- **Size:** 32 bits
- **Offset:** 0x84
- **Exists:** Always



Table 5-292 Fields for Register: SS_CORE_DIAG_REG21

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG21	R	<p>This register stores the diagnostic register21 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.34 SS_CORE_DIAG_REG22

- **Name:** Core Diagnostic Register22
- **Description:** This is Core Diagnostic Register22
- **Size:** 32 bits
- **Offset:** 0x88
- **Exists:** Always



Table 5-293 Fields for Register: SS_CORE_DIAG_REG22

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG22	R	<p>This register stores the diagnostic register22 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.35 SS_CORE_DIAG_REG23

- **Name:** Core Diagnostic Register23
- **Description:** This is Core Diagnostic Register23
- **Size:** 32 bits
- **Offset:** 0x8c
- **Exists:** Always



Table 5-294 Fields for Register: SS_CORE_DIAG_REG23

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG23	R	<p>This register stores the diagnostic register23 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.36 SS_CORE_DIAG_REG24

- **Name:** Core Diagnostic Register24
- **Description:** This is Core Diagnostic Register24
- **Size:** 32 bits
- **Offset:** 0x90
- **Exists:** Always



Table 5-295 Fields for Register: SS_CORE_DIAG_REG24

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG24	R	<p>This register stores the diagnostic register24 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.37 SS_CORE_DIAG_REG25

- **Name:** Core Diagnostic Register25
- **Description:** This is Core Diagnostic Register25
- **Size:** 32 bits
- **Offset:** 0x94
- **Exists:** Always



Table 5-296 Fields for Register: SS_CORE_DIAG_REG25

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG25	R	<p>This register stores the diagnostic register25 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.38 SS_CORE_DIAG_REG26

- **Name:** Core Diagnostic Register26
- **Description:** This is Core Diagnostic Register26
- **Size:** 32 bits
- **Offset:** 0x98
- **Exists:** Always



Table 5-297 Fields for Register: SS_CORE_DIAG_REG26

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG26	R	<p>This register stores the diagnostic register26 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.39 SS_CORE_DIAG_REG27

- **Name:** Core Diagnostic Register27
- **Description:** This is Core Diagnostic Register27
- **Size:** 32 bits
- **Offset:** 0x9c
- **Exists:** Always



Table 5-298 Fields for Register: SS_CORE_DIAG_REG27

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG27	R	<p>This register stores the diagnostic register27 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.40 SS_CORE_DIAG_REG28

- **Name:** Core Diagnostic Register28
- **Description:** This is Core Diagnostic Register28
- **Size:** 32 bits
- **Offset:** 0xa0
- **Exists:** Always



Table 5-299 Fields for Register: SS_CORE_DIAG_REG28

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG28	R	<p>This register stores the diagnostic register28 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.41 SS_CORE_DIAG_REG29

- **Name:** Core Diagnostic Register29
- **Description:** This is Core Diagnostic Register29
- **Size:** 32 bits
- **Offset:** 0xa4
- **Exists:** Always



Table 5-300 Fields for Register: SS_CORE_DIAG_REG29

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG29	R	<p>This register stores the diagnostic register29 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.42 SS_CORE_DIAG_REG30

- **Name:** Core Diagnostic Register30
- **Description:** This is Core Diagnostic Register30
- **Size:** 32 bits
- **Offset:** 0xa8
- **Exists:** Always



Table 5-301 Fields for Register: SS_CORE_DIAG_REG30

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG30	R	<p>This register stores the diagnostic register30 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.43 SS_CORE_DIAG_REG31

- **Name:** Core Diagnostic Register31
- **Description:** This is Core Diagnostic Register31
- **Size:** 32 bits
- **Offset:** 0xac
- **Exists:** Always



Table 5-302 Fields for Register: SS_CORE_DIAG_REG31

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG31	R	<p>This register stores the diagnostic register31 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.44 SS_CORE_DIAG_REG32

- **Name:** Core Diagnostic Register32
- **Description:** This is Core Diagnostic Register32
- **Size:** 32 bits
- **Offset:** 0xb0
- **Exists:** Always

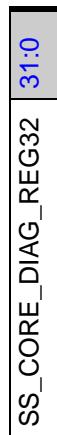


Table 5-303 Fields for Register: SS_CORE_DIAG_REG32

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG32	R	<p>This register stores the diagnostic register32 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.45 SS_CORE_DIAG_REG33

- **Name:** Core Diagnostic Register33
- **Description:** This is Core Diagnostic Register33
- **Size:** 32 bits
- **Offset:** 0xb4
- **Exists:** Always



Table 5-304 Fields for Register: SS_CORE_DIAG_REG33

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG33	R	<p>This register stores the diagnostic register33 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.46 SS_CORE_DIAG_REG34

- **Name:** Core Diagnostic Register34
- **Description:** This is Core Diagnostic Register34
- **Size:** 32 bits
- **Offset:** 0xb8
- **Exists:** Always



Table 5-305 Fields for Register: SS_CORE_DIAG_REG34

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG34	R	<p>This register stores the diagnostic register34 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.47 SS_CORE_DIAG_REG35

- **Name:** Core Diagnostic Register35
- **Description:** This is Core Diagnostic Register35
- **Size:** 32 bits
- **Offset:** 0xbc
- **Exists:** Always



Table 5-306 Fields for Register: SS_CORE_DIAG_REG35

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG35	R	<p>This register stores the diagnostic register35 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.48 SS_CORE_DIAG_REG36

- **Name:** Core Diagnostic Register36
- **Description:** This is Core Diagnostic Register36
- **Size:** 32 bits
- **Offset:** 0xc0
- **Exists:** Always



Table 5-307 Fields for Register: SS_CORE_DIAG_REG36

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG36	R	<p>This register stores the diagnostic register36 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.49 SS_CORE_DIAG_REG37

- **Name:** Core Diagnostic Register37
- **Description:** This is Core Diagnostic Register37
- **Size:** 32 bits
- **Offset:** 0xc4
- **Exists:** Always



Table 5-308 Fields for Register: SS_CORE_DIAG_REG37

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG37	R	<p>This register stores the diagnostic register37 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.50 SS_CORE_DIAG_REG38

- **Name:** Core Diagnostic Register38
- **Description:** This is Core Diagnostic Register38
- **Size:** 32 bits
- **Offset:** 0xc8
- **Exists:** Always



Table 5-309 Fields for Register: SS_CORE_DIAG_REG38

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG38	R	<p>This register stores the diagnostic register38 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.51 SS_CORE_DIAG_REG39

- **Name:** Core Diagnostic Register39
- **Description:** This is Core Diagnostic Register39
- **Size:** 32 bits
- **Offset:** 0xcc
- **Exists:** Always



Table 5-310 Fields for Register: SS_CORE_DIAG_REG39

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG39	R	<p>This register stores the diagnostic register39 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.52 SS_CORE_DIAG_REG40

- **Name:** Core Diagnostic Register40
- **Description:** This is Core Diagnostic Register40
- **Size:** 32 bits
- **Offset:** 0xd0
- **Exists:** Always



Table 5-311 Fields for Register: SS_CORE_DIAG_REG40

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG40	R	<p>This register stores the diagnostic register40 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.53 SS_CORE_DIAG_REG41

- **Name:** Core Diagnostic Register41
- **Description:** This is Core Diagnostic Register41
- **Size:** 32 bits
- **Offset:** 0xd4
- **Exists:** Always

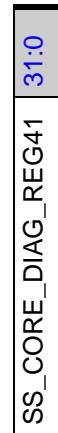


Table 5-312 Fields for Register: SS_CORE_DIAG_REG41

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG41	R	<p>This register stores the diagnostic register41 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

5.3.1.54 SS_CORE_DIAG_REG42

- **Name:** Core Diagnostic Register42
- **Description:** This is Core Diagnostic Register42
- **Size:** 32 bits
- **Offset:** 0xd8
- **Exists:** Always



Table 5-313 Fields for Register: SS_CORE_DIAG_REG42

Bits	Name	Memory Access	Description
31:0	SS_CORE_DIAG_REG42	R	<p>This register stores the diagnostic register42 from the core</p> <p>Value After Reset: 0x0</p> <p>Exists: Always</p> <p>Volatile: true</p>

A

Area and Power

The below tables show the area and power numbers of the DWC_mipicsi2_device for different configurations. The area and power numbers shown in these tables are sample values obtained from synthesizing the DWC_mipicsi2_device for industry standard 7nm, industry standard 16nm and industry standard 28nm technology. The PHY or RAM areas are not considered in these numbers. The area is represented by two-input NAND gates.

A.1 The standard controller (DWC_mipicsi2_device)

Table A-1 DWC_mipicsi2_device Gate Count and Power Numbers for Industry Standard 7nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	none	58.8	0.464
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	1	none	61.6	0.46
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	4	none	138.8	1.12
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	none	99.2	0.757
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	none	1	52.6	0.376
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	none	60.6	0.492
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	1	none	63.4	0.489
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	4	none	140.9	1.15
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	none	101.1	0.786
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	none	1	54.4	0.405
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	none	65	0.551
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	1	none	67.3	0.547
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	4	none	144.9	1.21
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	none	105.1	0.845
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	none	1	58.3	0.464

Table A-2 DWC_mipicsi2_device Gate Count and Power Numbers for Industry Standard 16nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	none	51.4	0.789
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	1	none	55.2	0.79
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	4	none	125.6	1.9
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	none	89.3	1.28
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	none	1	46.4	0.645
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	none	54.1	0.837

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	1	none	56.9	0.832
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	4	none	127.4	1.95
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	none	91.1	1.34
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	none	1	48.1	0.695
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	none	56.7	0.94
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	1	none	60.5	0.94
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	4	none	130.8	2.05
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	none	94.8	1.44
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	none	1	51.5	0.793

Table A-3 DWC_mipicsi2_device Gate Count and Power Numbers for Industry Standard 28nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	none	52.6	1.08
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	1	none	56.5	1.1
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	4	none	128.7	2.69
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	none	90.8	1.78
DPHY	2	8	2.5Gbps/312.5Mhz	1	none	none	1	47.2	0.895
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	none	54.3	1.16
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	1	none	58.1	1.18
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	4	none	130.1	2.76
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	none	92.4	1.85
DPHY	4	8	2.5Gbps/312.5Mhz	1	none	none	1	48.9	0.968
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	none	58.1	1.3
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	1	none	61.8	1.32
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	4	none	133.9	2.91
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	none	96.1	1.99

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	8	8	2.5Gbps/312.5Mhz	1	none	none	1	52.4	1.12

A.2 The standard controller with automotive (DWC_ap_mipicsi2_device)

Table A-4 DWC_ap_mipicsi2_device Gate Count and Power Numbers for Industry Standard 7nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	none	none	103.2	0.89
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	1	none	104.4	0.863
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	4	none	246.1	2.12
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	1	none	167.1	1.4
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	none	1	83.2	0.669
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	none	none	106.4	0.95
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	1	none	107.9	0.922
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	4	none	249.8	2.18
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	1	none	170.6	1.46
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	none	1	86.9	0.73
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	none	none	114	1.07
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	1	none	115.4	1.04
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	4	none	257.4	2.21
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	1	none	177.7	1.56
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	none	1	94	0.85

Table A-5 DWC_ap_mipicsi2_device Gate Count and Power Numbers for Industry Standard 16nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	none	none	92.2	1.51
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	1	none	94.9	1.49
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	4	none	228.4	3.6
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	1	none	154	2.5
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	none	1	75.8	1.15
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	none	none	95.4	1.65

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	1	none	98	1.59
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	4	none	231.4	3.7
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	1	none	156.8	2.6
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	none	1	78.9	1.25
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	none	none	101.8	1.82
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	1	none	104.4	1.8
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	4	none	238	3.91
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	1	none	163.3	2.81
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	none	1	85.1	1.46

Table A-6 DWC_ap_mipicsi2_device Gate Count and Power Numbers for Industry Standard 28nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	none	none	93.8	2.12
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	1	none	96.8	2.07
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	4	none	230.6	5.03
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	1	1	none	154.1	3.29
DPHY	2	8	2.5Gbps/312.5Mhz	1	1	none	none	1	76.5	1.63
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	none	none	96.8	2.28
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	1	none	99.7	2.22
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	4	none	232.5	5.18
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	1	1	none	158.1	3.44
DPHY	4	8	2.5Gbps/312.5Mhz	1	1	none	none	1	79.5	1.78
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	none	none	103.8	2.58
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	1	none	106.4	2.52
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	4	none	240.3	5.48
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	1	1	none	164.8	3.75

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
DPHY	8	8	2.5Gbps/312.5Mhz	1	1	none	none	1	86.2	2.08

A.3 The combo controller (DWC_mipicsi2_device_combo)

Table A-7 DWC_mipicsi2_device_combo Gate Count and Power Numbers for Industry Standard 7nm Library

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	72	0.607
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	1	none	64.4	0.513
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	4	none	142.2	1.18
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	112.5	0.897
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	none	1	66.5	0.51
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	74.9	0.643
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	1	none	67.1	0.549
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	4	none	144.8	1.22
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	115.4	0.933
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	none	1	69.7	0.538

Table A-8 DWC_mipicsi2_device_combo Gate Count and Power Numbers for Industry Standard 16nm Library

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	62.8	1.04
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	1	none	57.8	0.875
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	4	none	128.6	2
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	100.9	1.52
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	none	1	58.5	0.902
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	65.1	1.09
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	1	none	60.3	0.935
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	4	none	131.1	2.06
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	103.4	1.58
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	none	1	61.1	0.923

Table A-9 DWC_mipicsi2_device_combo Gate Count and Power Numbers for Industry Standard 28nm Library

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	64.3	1.42
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	1	none	58.8	1.24
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	4	none	131.1	2.84
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	102.4	2.13
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	none	1	59.9	1.21
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	66.7	1.51
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	1	none	61.4	1.33
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	4	none	133.6	2.93
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	104.9	2.22
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	none	none	1	62.5	1.3

A.4 The combo controller with automotive (DWC_ap_mipicsi2_device_combo)

Table A-10 DWC_ap_mipicsi2_device_combo Gate Count and Power Numbers for Industry Standard 7nm Library

PHY TYPE	DPHY/CPHY Lanes	DPHY/CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	none	124	1.13
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	1	none	109	0.948
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	4	none	251	2.24
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	1	none	187.4	1.62
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	1	103.6	0.891
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	none	128.9	1.21
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	1	none	113.9	1.02
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	4	none	256	2.37
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	1	none	192.2	1.7
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	1	109	0.966

Table A-11 DWC_ap_mipicsi2_device_combo Gate Count and Power Numbers for Industry Standard 16nm Library

PHY TYPE	DPHY/CPHY Lanes	DPHY/CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	none	111.5	1.91
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	1	none	98.9	1.64

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	4	none	232.1	3.92
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	1	none	174.1	2.97
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	1	94.2	1.53
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	none	116.2	2.04
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	1	none	103.5	1.76
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	4	none	237.4	3.93
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	1	none	178.1	2.92
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	1	98.8	1.66

Table A-12 DWC_ap_mipicsi2_device_combo Gate Count and Power Numbers for Industry Standard 28nm Library

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	none	113.4	2.68
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	1	none	100.8	2.29
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	4	none	235.6	5.33
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	1	none	175	3.89
Combo PHY	2L/2T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	1	95.1	2.17
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	none	none	117.9	2.87

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	64-bit IDI 400Mhz	48-bit IPI 400Mhz	SDI		
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	1	none	105.2	2.47
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	4	none	239.9	5.52
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	1	1	none	178.3	4.1
Combo PHY	4L/3T	8/16	2.5Gbps/2.5Gsps/357.2Mhz	1	1	none	none	1	99.6	2.35

A.5 The high-performance controller (DWC_mipicsi2_hp_device)

Table A-13 DWC_mipicsi2_hp_device Gate Count and Power Numbers for Industry Standard 7nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	none	none	65.4	0.6
DPHY	2	16	6.5Gbps/406.3Mhz	1	none	1	none	90.1	0.773
DPHY	2	16	6.5Gbps/406.3Mhz	1	none	4	none	246.1	2.08
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	1	none	131.9	1.14
DPHY	2	16	6.5Gbps/406.3Mhz	1	none	none	1	54.6	0.478
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	none	none	68.8	0.642
DPHY	4	16	6.5Gbps/406.3Mhz	1	none	1	none	93.6	0.813
DPHY	4	16	6.5Gbps/406.3Mhz	1	none	4	none	250.2	2.2
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	1	none	135.1	1.16
DPHY	4	16	6.5Gbps/406.3Mhz	1	none	none	1	57.8	0.523
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	none	none	90.9	0.74
DPHY	8	16	6.5Gbps/406.3Mhz	1	none	1	none	116.3	0.91
DPHY	8	16	6.5Gbps/406.3Mhz	1	none	4	none	272	2.25
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	1	none	157.6	1.29
DPHY	8	16	6.5Gbps/406.3Mhz	1	none	none	1	82.4	0.67
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	none	none	68.7	0.444
DPHY	2	32	6.5Gbps/203.2Mhz	1	none	1	none	93.2	0.609
DPHY	2	32	6.5Gbps/203.2Mhz	1	none	4	none	248.4	1.88
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	1	none	135	0.944
DPHY	2	32	6.5Gbps/203.2Mhz	1	none	none	1	58.1	0.278
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	none	none	89.9	0.477
DPHY	4	32	6.5Gbps/203.2Mhz	1	none	1	none	114.8	0.644
DPHY	4	32	6.5Gbps/203.2Mhz	1	none	4	none	271.4	1.96
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	1	none	156.5	0.978
DPHY	4	32	6.5Gbps/203.2Mhz	1	none	none	1	81.4	0.339

Table A-14 DWC_mipicsi2_hp_device Gate Count and Power Numbers for Industry Standard 16nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	none	none	56.8	1.02
DPHY	2	16	6.5Gbps/406.3Mhz	1	none	1	none	80.5	1.26
DPHY	2	16	6.5Gbps/406.3Mhz	1	none	4	none	221.3	3.47
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	1	none	118.6	1.91
DPHY	2	16	6.5Gbps/406.3Mhz	1	none	none	1	48.6	0.876
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	none	none	60	1.08
DPHY	4	16	6.5Gbps/406.3Mhz	1	none	1	none	83.5	1.34
DPHY	4	16	6.5Gbps/406.3Mhz	1	none	4	none	224.3	3.52
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	1	none	121.7	1.96
DPHY	4	16	6.5Gbps/406.3Mhz	1	none	none	1	51.3	0.901
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	none	none	81.3	1.25
DPHY	8	16	6.5Gbps/406.3Mhz	1	none	1	none	105	1.54
DPHY	8	16	6.5Gbps/406.3Mhz	1	none	4	none	246.4	3.78
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	1	none	143.7	2.15
DPHY	8	16	6.5Gbps/406.3Mhz	1	none	none	1	74.2	1.15
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	none	none	60	0.746
DPHY	2	32	6.5Gbps/203.2Mhz	1	none	1	none	83.7	0.991
DPHY	2	32	6.5Gbps/203.2Mhz	1	none	4	none	224.5	3.09
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	1	none	121.4	1.55
DPHY	2	32	6.5Gbps/203.2Mhz	1	none	none	1	51.6	0.476
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	none	none	80.1	0.795
DPHY	4	32	6.5Gbps/203.2Mhz	1	none	1	none	103.9	1.05
DPHY	4	32	6.5Gbps/203.2Mhz	1	none	4	none	245	3.15
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	1	none	142.1	1.61
DPHY	4	32	6.5Gbps/203.2Mhz	1	none	none	1	73.2	0.563

A.6 The high-performance controller with automotive (DWC_ap_mipicsi2_hp_device)

Table A-15 DWC_ap_mipicsi2_hp_device Gate Count and Power Numbers for Industry Standard 7nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	1	none	none	120.2	1.13
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	none	1	none	160.5	1.35
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	none	4	none	456.7	3.75
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	1	1	none	236	2.01
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	none	none	1	86.3	0.824
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	1	none	none	126	1.22
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	none	1	none	167.4	1.44
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	none	4	none	462.7	3.74
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	1	1	none	241.9	2.1
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	none	none	1	92.5	0.915
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	1	none	none	156.7	1.41
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	none	1	none	199.1	1.64
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	none	4	none	495.2	4.01
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	1	1	none	274.2	2.32
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	none	none	1	126.6	1.16
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	1	none	none	125.6	0.853
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	none	1	none	166.7	1.03
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	none	4	none	463.1	3.09
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	1	1	none	242.6	1.6
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	none	none	1	91.9	0.532
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	1	none	none	155.1	0.922
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	none	1	none	196.1	1.1
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	none	4	none	491	3.17
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	1	1	none	271.7	1.67

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	none	none	1	124.1	0.634

Table A-16 DWC_ap_mipicsi2_hp_device Gate Count and Power Numbers for Industry Standard 16nm Library

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	1	none	none	108.5	1.92
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	none	1	none	150.1	2.28
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	none	4	none	434.5	6.28
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	1	1	none	221.5	3.48
DPHY	2	16	6.5Gbps/406.3Mhz	1	1	none	none	1	78.8	1.42
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	1	none	none	113.7	2.12
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	none	1	none	155.8	2.49
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	none	4	none	440	6.36
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	1	1	none	227	3.63
DPHY	4	16	6.5Gbps/406.3Mhz	1	1	none	none	1	84.4	1.64
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	1	none	none	143	2.48
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	none	1	none	185.2	2.77
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	none	4	none	469.2	6.8
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	1	1	none	258.8	3.93
DPHY	8	16	6.5Gbps/406.3Mhz	1	1	none	none	1	115.4	1.99
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	1	none	none	113.5	1.44
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	none	1	none	155	1.75
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	none	4	none	438.5	5.21
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	1	1	none	225.4	2.71
DPHY	2	32	6.5Gbps/203.2Mhz	1	1	none	none	1	84	0.917
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	1	none	none	141	1.56
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	none	1	none	182.4	1.86

PHY TYPE	Lanes	PPI Width	Max Physical Interface speed per lane/ PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	none	4	none	466.2	5.39
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	1	1	none	254.6	2.83
DPHY	4	32	6.5Gbps/203.2Mhz	1	1	none	none	1	113	1.07

A.7 The high-performance and combo controller (DWC_mipicsi2_hp_device_combo)

Table A-17 DWC_mipicsi2_hp_device_combo Gate Count and Power Numbers for Industry Standard 7nm Library

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	none	77.4	0.79
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	1	none	92	0.87
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	4	none	248.4	2.23
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	none	144.5	1.35
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	none	1	67.6	0.686
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	none	81	0.832
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	1	none	95.8	0.925
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	4	none	251.6	2.26
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	none	147.4	1.36
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	none	1	71.2	0.731
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	none	80.8	0.775
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	1	none	95.4	0.852
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	4	none	252.2	2.18
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	none	148	1.29
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	none	1	71.4	0.65
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	none	102.3	0.85
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	1	none	117.3	0.93
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	4	none	274.2	2.26
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	none	169.7	1.39
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	none	1	94.6	0.819

Table A-18 DWC_mipicsi2_hp_device_combo Gate Count and Power Numbers for Industry Standard 16nm Library

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface				Area (K Gates)	Power (mW)
				APB 250Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	none	67.3	1.34
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	1	none	81.8	1.47
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	4	none	222.1	3.66
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	none	128.9	2.2
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	none	1	59.8	1.27
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	none	70.3	1.43
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	1	none	85	1.56
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	4	none	225.1	3.78
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	none	132	2.33
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	none	none	1	62.9	1.26
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	none	70.8	1.32
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	1	none	85.6	1.44
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	4	none	225.5	3.64
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	none	132.4	2.17
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	none	1	63.4	1.12
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	none	91.2	1.44
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	1	none	106.4	1.54
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	4	none	246.7	3.78
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	none	153.8	2.29
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	none	none	1	85.3	1.42

A.8 The high performance and combo controller with automotive (DWC_ap_mipicsi2_hp_device_combo)

Table A-19 DWC_ap_mipicsi2_hp_device_combo Gate Count and Power Numbers for Industry Standard 7nm Library

PHY TYPE	DPHY/CPHY Lanes	DPHY/CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	none	none	138.9	1.47
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	1	none	164.3	1.53
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	4	none	458.5	4.07
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	1	none	255.8	2.43
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	none	1	105.2	1.15
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	none	none	144.9	1.64
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	1	none	169.4	1.68
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	4	none	463.7	4.32
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	1	none	262.3	2.49
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	none	1	110.6	1.27
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	none	none	145.1	1.45
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	1	none	169.7	1.57
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	4	none	464.3	3.99
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	1	none	261.1	2.34
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	none	1	111.2	1.19

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	none	none	174.4	1.6
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	1	none	200.5	1.66
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	4	none	496.4	4.14
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	1	none	292.7	2.48
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	none	1	143.5	1.4

Table A-20 DWC_ap_mipicsi2_hp_device_combo Gate Count and Power Numbers for Industry Standard 16nm Library

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	none	none	126.2	2.51
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	1	none	153.8	2.64
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	4	none	440.8	6.93
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	1	none	241.6	4.14
Combo PHY	2L/2T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	none	1	96	2.1
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	none	none	131.9	2.66
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	1	none	159.5	2.87
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	4	none	446.3	7.12
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	1	1	none	247.1	4.32

PHY TYPE	DPHY/ CPHY Lanes	DPHY/ CPHY PPI Width	DPHY/CPHY Max Physical Interface speed per lane / PPI clk frequency	System Interface					Area (K Gates)	Power (mW)
				APB 250Mhz	SSM 400Mhz	128-bit IDI 400Mhz	192-bit IPI 400Mhz	SDI		
Combo PHY	4L/3T	16/16	6.5Gbps/3.5Gsps/500Mhz	1	1	none	none	1	101.3	2.17
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	none	none	131.8	2.47
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	1	none	159.4	2.63
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	4	none	445.3	6.72
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	1	none	246.7	4.01
Combo PHY	2L/2T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	none	1	101.4	1.93
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	none	none	160.1	2.78
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	1	none	187.9	2.88
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	4	none	474.8	7.18
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	1	1	none	277.5	4.31
Combo PHY	4L/3T	32/32	6.5Gbps/6.5Gsps/464.2Mhz	1	1	none	none	1	132.1	2.32

B

DWC_mipicsi2_device Automotive Safety

DWC_mipicsi2_device is ISO 26262 ASIL B ready. This section describes the automotive features supported by the controller and enabled by the CSI2_DEVICE_AP parameter when you have the corresponding automotive license. The following topics are discussed:

- “Overview of Automotive Safety Feature” on page [722](#)
- “Automotive Safety Package Documents” on page [723](#)
- “Signals Related to Automotive Safety Feature” on page [724](#)
- “Programming Automotive Safety Feature” on page [725](#)
- “Configuring Automotive Safety Feature” on page [726](#)
- “Automotive Package Safety Memory Requirements” on page [727](#)
- “Memory Error Injection” on page [728](#)
- “Faulty Address Logger” on page [729](#)

B.1 Overview of Automotive Safety Feature

To make an IP better suited for applications in the automotive space, functional safety needs to be considered throughout the IP development process. ISO 26262 addresses functional safety and requires the hardware components to be analyzed with respect to their ability to detect and/or control the effects of random hardware faults. Random hardware faults can occur at any point in time during the life-cycle of the hardware component. They can take the form of permanent faults, or transient faults, and require safety mechanisms to be in place to ensure that severe consequences do not occur as a result of those faults.

The DWC_mipicsi2_device includes a broad range of internal safety mechanisms which assist in handling random hardware faults. These complement the mechanisms already defined in the DWC_mipicsi2_device protocol.

B.1.1 ASIL B Certification

The DWC_mipicsi2_device has been analyzed in the scope of ISO 26262 targeting ASIL B for a particular reference configuration.

Reference Configuration

A reference configuration containing the major features such as; Image Pixel Interface, Image Data Interface, and 4 Lanes has been used for ASIL B analysis.

The reference configuration internal safety mechanisms monitor the IP logic and report any detected errors to the application. The application should take an action to handle the error conditions and ensure safe operation of the hardware component.

FMEDA

A Failure Mode, Effects, and Diagnostic Analysis (FMEDA) has also been performed for this reference configuration. The analysis relies on a per-transistor failure rate derived in accordance to IEC TR 62380. This, in conjunction with area information from design synthesis trials, allows to determine the failure rate for the IP which can then be distributed between the modules of the design according to their relative area.

The list of failure modes and their effects is derived for each module and each of the failure modes is assigned a portion of the failure rate of the module they belong to. If the failure mode has the potential to violate a top level safety requirement (error at a top level output pin) there is a search for a safety mechanism that can prevent that violation, totally or partially. If a safety mechanism exists, for example, ECC protection for RAMs, a diagnostic coverage (DC) value is assigned representing the percentage of the failure rate associated with the specific failure mode that is prevented from violating a top level safety requirement. The process is repeated for all failure modes and finally results are accumulated and IP safety metrics are collected in the FMEDA.

B.2 Automotive Safety Package Documents

Table B-1 describes the documents included in the DWC_mipicsi2_device safety package. You enable the safety package by setting the CSI2_DEVICE_AP configuration parameter. The controller safety package contains the L10W certificate, L10W0002 certification report, IP safety manual which includes all safety relevant information (such as the Failure Mode and Effects), and Diagnostic Analysis (FMEDA) document which shows a probabilistic evaluation of the hardware. After you have configured the controller in coreConsultant, you can access the automotive documents in your workspace directory at workspace/doc/automotive.

Table B-1 Description of Documents in Automotive Safety Package

File Name	Description	Contents
DWC_mipicsi2_device_dm_safety_manual.pdf	Safety Manual	Lists all Synopsys quality process documents; explains IP design flow and tools, verification flows (high-level), and code coverage concepts; describes FMEDA worksheet
FMEDA.IC.DWC_mipicsi2_device	FMEDA Worksheet	Calculates FIT rates for each module; enumerates diagnostic capabilities for failure modes
DFMEA_DWC_mipicsi2_device_DM_exploration.xlsx	DFMEA Worksheet	DFMEA analysis.

B.3 Signals Related to Automotive Safety Feature

For more information, see “FMEDA Interrupt Signals” and “Diagnosis fault Signals” in *Signal Descriptions* chapter.

B.4 Programming Automotive Safety Feature

For programming sequences, see the following sections in the “Programming DWC_mipicsi2_device” chapter of the MIPI CSI-2 Device Controller User Guide:

- Automotive Package Detecting Errors
- Automotive package Error Injection
- Automotive Package Synchronizer Check
- Automotive Package Faulty Address Logging

B.5 Configuring Automotive Safety Feature

If you do not have a valid license, this configuration parameter is disabled (grayed out) in coreConsultant. To generate a DWC_mipicsi2_device controller with support for automotive package, you must have an associated DWC_ap_mipicsi2_device license. You can enable the feature using CSI2_DEVICE_AP parameter.

You can then access this feature by selecting Automotive Package option > Core Configuration pane from the Specify Configuration activity dialog box, while configuring the controller in the coreConsultant GUI.

B.6 Automotive Package Safety Memory Requirements

Due to the increased storage of cyclic redundancy check, 2-port IDI RAM requirement is shown in [Table B-2](#).

Table B-2 Automotive Packet Safety 2-Port IDI RAM Requirements

Feature	Timing Requirement	2-port RAMs Size
Backpressure feature is enabled	No	<ul style="list-style-type: none"> ■ When the width of the payload FIFO is 72bits with Automotive Package function <ul style="list-style-type: none"> □ Long packet size + 4*64bits*(72/64) ■ When the width of the payload FIFO is 144bits without Automotive Package function <ul style="list-style-type: none"> □ Long packet size + 4*128bits)*(144/128)
Backpressure feature is disabled	Yes	<ul style="list-style-type: none"> ■ When the width of the payload FIFO is 72bits with Automotive Package function <ul style="list-style-type: none"> □ 2*(Long packet size + 1*64bits)*(72/64) ■ When the width of the payload FIFO is 144bits without Automotive Package function <ul style="list-style-type: none"> □ 2*(Long packet size + 1*128bits)*(144/128)

B.6.1 IPI RAM requirement

[Table B-3](#) shows 2-Port IPI RAM requirement.

Table B-3 Automotive Packet Safety 2-Port IDI RAM Requirements

Mode	Timing Requirement	2-port RAMs Size
Store and Forward Mode	Yes	<ul style="list-style-type: none"> ■ When the width of the payload FIFO is 72bits with Automotive Package function <ul style="list-style-type: none"> □ 2*(Long packet size + 1*64bits)*(72/64) ■ When the width of the payload FIFO is 144bits without Automotive Package function <ul style="list-style-type: none"> □ 2*(Long packet size + 1*128bits)*(144/128)
Cut Through mode	Yes	<ul style="list-style-type: none"> ■ When the width of the payload FIFO is 72bits with Automotive Package function <ul style="list-style-type: none"> □ Long packet size + 1*64bits)*(72/64) ■ When the width of the payload FIFO is 144bits without Automotive Package function <ul style="list-style-type: none"> □ Long packet size + 1*128bits)*(144/128)
Backpressure Mode	Yes	<ul style="list-style-type: none"> ■ When the width of the payload FIFO is 72bits with Automotive Package function <ul style="list-style-type: none"> □ Long packet size + 6*64bits*(72/64) ■ When the width of the payload FIFO is 144bits without Automotive Package function <ul style="list-style-type: none"> □ Long packet size + 6*128bits)*(144/128)

B.7 Memory Error Injection

This feature allows error injection in IDI/IPI RAM memory for testing purposes. You can provide an error mask to select which bits to flip in payload and ECC 8-bit protection as relevant.

The error masks can be configured with the following registers:

- ERR_INJ_CHK_MASK (ECC bits)
- ERR_INJ_DH32_MASK (data high 32 bits)
- ERR_INJ_DL32_MASK (data low 32 bits)

The following two registers are used for control and status:

- ERR_INJ_CTRL (err_inj_en, err_inj_mem_sel)
- ERR_INJ_STATUS (err_inj_end)

The injected errors will originate N interrupt pulses that should be read by polling single and multiple ECC error bits as follows.

- For idi: INT_ST_DIAG0[15:14]
- For ipi: INT_ST_DIAG0[23:22]
- For ipi2: INT_ST_DIAG1[4:3]
- For ipi3: INT_ST_DIAG1[11:10]
- For ipi4: INT_ST_DIAG1[18:17]

Detailed information about this procedure can be found in MIPI CSI-2 Device Controller User Guide.

B.8 Faulty Address Logger

When a single or multiple ECC error occurs in IDI/IPI/IPI2/IPI3/IPI4 RAM, it is possible to access the faulty addresses using the Faulty Address Logger.

DWC_mipicsi2_device controller stores up to 15 faulty RAM addresses in an internal FIFO. When new address needs to be logged and FIFO is full, the address is dropped.

The following two registers are provided to allow access to this information:

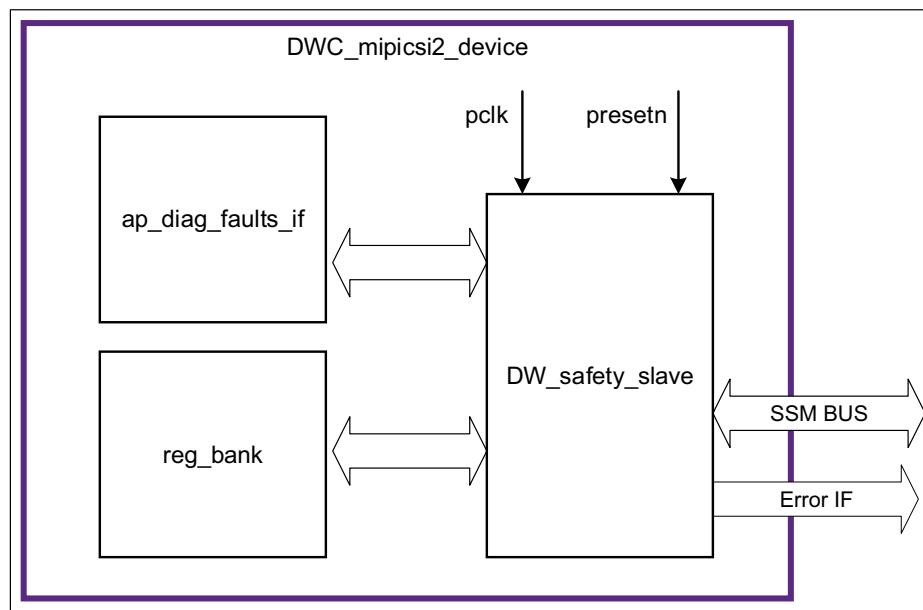
- IDI_RAM_ERR_LOG_AP/ IPI*_RAM_ERR_LOG_AP: A read-only status register stores how many errors are logged.
- IDI_RAM_ADDR_LOG_AP/IPI*_RAM_ERR_LOG_AP: A read-only register that stores the first faulty address, changing to the following upon an APB read operation. This read operations is destructive, which is to say the previous address value is overwritten by the next faulty address. Upon read operation IDI_RAM_ERR_LOG_AP/IPI_RAM_ERR_LOG_AP is updated.

B.9 Safety Slave

DWC_mipicsi2_device support safety slave to enhance the Automotive Safety mechanism. Safety slave will be controlled by safety manager application. This usage of safety slave is optional. It can be disabled by parameters CSI2_DEVICE_AP and CSI2_DEVICE_SS.

B.9.1 Overview of Safety Slave

Figure B-1 Safety Slave



New interfaces will be added to the top module of DWC_mipicsi2_device

- SSM bus interface – Interface between safety slave and safety manager
- 2bit Error signal – This is for notifying safety manager that error detected

B.9.2 Description of Safety Slave

The Safety Slave (SS) is a standalone module which provide the necessary hooks for connecting DWC_mipicsi2_device to an external safety manager module through a dedicated safety bus (SSM). SSM is the modified APB4. Core diagnostic register reads are allowed through SSM bus. Core diagnostic register writes are not allowed and not supported. Write transfers are allowed only to programming the control registers of SS. The following functions are supported in the safety slave module.

- SSM slave state machine with parity and anti-valence bit support for address/data/control signals. Detection and reporting of any SSM bus errors.
- Address decoder to access core diagnostic registers.
- Synchronizers for CDC signals (SSM and core clocks)
- Safety Slave reset mechanism with Core reset (hw and sw) and SSM bus resets.
- Generation of “safety error” state to central Safety Manager.

Safety Bus Parity

The SS supports an optional parity check for the safety bus signals. The Safety master generates either an odd or an even parity for each byte of ssm_pwdata, ssm_pstrb (1 per nibble) and ssm_paddr. The generated parity bits are passed on to the SS through the respective p*user signals. SS recomputes the parity on ssm_pwdata, ssm_pstrb and ssm_paddr signals and checks for validity before accepting the write transfer. If invalid, SS ignores the writes and registers the error and reports through pslverr.

Safety Bus Wait State

The core registers (monitor and diagnostic) are maintained in core clock domain. Any core register read transfer initiated by the SM will get synchronized to the core clock domain before the SS responds with the read data. The SS puts a wait state on the safety bus until the read data is ready.

Safety Diagnostic Registers

Safety Diagnostic Registers consist of safety slave registers with fixed offset and core registers with relocatable offset. Only read operations are allowed from the SSM bus to the core register space. The following figure shows the register map of the safety slave register space.

Figure B-2 Safety Slave Register Map

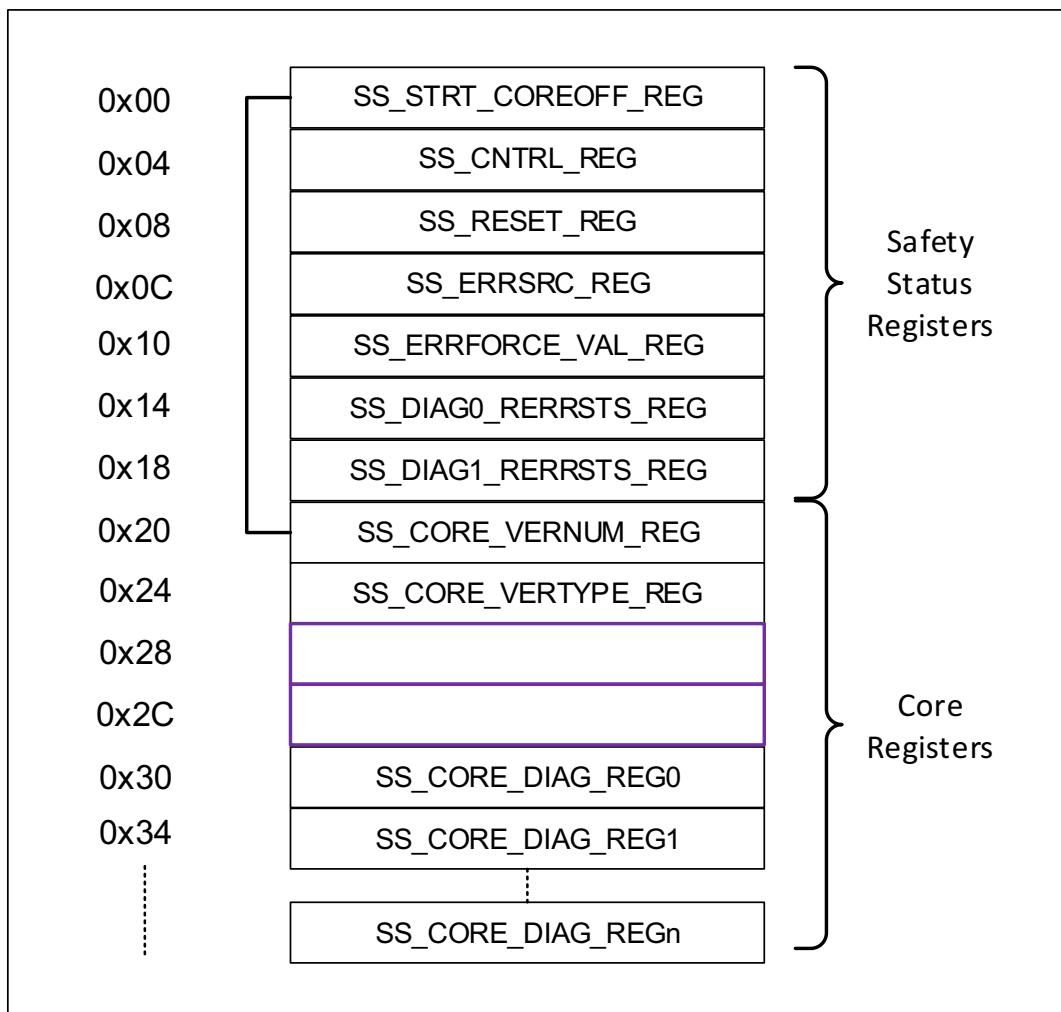


Table B-4 Fatal and Unrecoverable Errors from the Controller

Field Name	Core Unrecoverable Error Status
SS_CORE_URERR_STS[11:10]	Protocol errors fault
SS_CORE_URERR_STS[9:8]	PHY errors fault
SS_CORE_URERR_STS[7:6]	Input parity check errors fault
SS_CORE_URERR_STS[5:4]	CRC error found on data incoming from memory fault
SS_CORE_URERR_STS[3:2]	ECC error found but not corrected fault
SS_CORE_URERR_STS[1:0]	Registers Parity, Redundancy and Single pulse

Table B-5 No Fatal and Recoverable Errors from Controller

Field Name	Core Unrecoverable Error Status
SS_CORE_RERR_STS[3:2]	Synchronizer check errors fault
SS_CORE_RERR_STS[1:0]	ECC error found and corrected fault

Table B-6 Mapping of Core registers and DWC_mipicsi2_device Diagnostic Registers

Core Register	DWC_mipicsi2_device Diagnostic registers	Core register	DWC_mipicsi2_device Diagnostic registers
SS_CORE_VERNUM	VERSION	SS_CORE_VERTYPE	VER_TYPE
core_diagnostic_reg0	INT_ST_DIAG_MAIN	core_diagnostic_reg1	INT_ST_DIAG0
core_diagnostic_reg2	IPI_AP_STATUS	core_diagnostic_reg3	IDI_AP_STATUS
core_diagnostic_reg4	AMBAAPBINTF_AP_STATUS	core_diagnostic_reg5	PHY_IF_CTRL_AP_STATUS
core_diagnostic_reg6	REG_BANK_AP_STATUS	core_diagnostic_reg7	IPI_FIFOCTRL_AP_STATUS
core_diagnostic_reg8	IDI_FIFOCTRL_AP_STATUS	core_diagnostic_reg9	PKT_BUILDER_AP_STATUS
core_diagnostic_reg10	ERR_HANDLER_AP_STATUS	core_diagnostic_reg11	SYNC_AP_STATUS
core_diagnostic_reg12	PKT_IF_AP_STATUS	core_diagnostic_reg13	ECF_AP_STATUS
core_diagnostic_reg14	CMU_AP_STATUS	core_diagnostic_reg15	MT_IPI_CONTROL_AP_STATUS
core_diagnostic_reg16	IPI2_AP_STATUS	core_diagnostic_reg17	IPI2_FIFOCTRL_AP_STATUS
core_diagnostic_reg18	IPI3_AP_STATUS	core_diagnostic_reg19	IPI3_FIFOCTRL_AP_STATUS
core_diagnostic_reg20	IPI4_AP_STATUS	core_diagnostic_reg21	IPI4_FIFOCTRL_AP_STATUS

Core Register	DWC_mipicsi2_device Diagnostic registers	Core register	DWC_mipicsi2_device Diagnostic registers
core_diagnostic_reg22	SDI_AP_STATUS	core_diagnostic_reg23	INT_ST_FAP_VPG
core_diagnostic_reg24	INT_ST_FAP_IDI	core_diagnostic_reg25	INT_ST_FAP_IPI
core_diagnostic_reg26	INT_ST_FAP_PHY	core_diagnostic_reg27	INT_ST_FAP_IDI_VCX
core_diagnostic_reg28	INT_ST_FAP_IDI_VCX2	core_diagnostic_reg29	INT_ST_FAP_MT_IPI
core_diagnostic_reg30	INT_ST_FAP_SDI	core_diagnostic_reg31	INT_ST_FAP_SDI_VCX
core_diagnostic_reg32	INT_ST_FAP_SDI_VCX2	core_diagnostic_reg33	IDI_RAM_ERR_LOG_AP
core_diagnostic_reg34	IDI_RAM_ERR_ADDR_AP	core_diagnostic_reg35	IPI_RAM_ERR_LOG_AP
core_diagnostic_reg36	IPI_RAM_ERR_ADDR_AP	core_diagnostic_reg37	IPI2_RAM_ERR_LOG_AP
core_diagnostic_reg38	IPI2_RAM_ERR_ADDR_AP	core_diagnostic_reg39	IPI3_RAM_ERR_LOG_AP
core_diagnostic_reg40	IPI3_RAM_ERR_ADDR_AP	core_diagnostic_reg41	IPI4_RAM_ERR_LOG_AP
core_diagnostic_reg42	IPI4_RAM_ERR_ADDR_AP		

Sample transfers

Figure B-3 SSM Bus Write Transfer with Wait State and Error

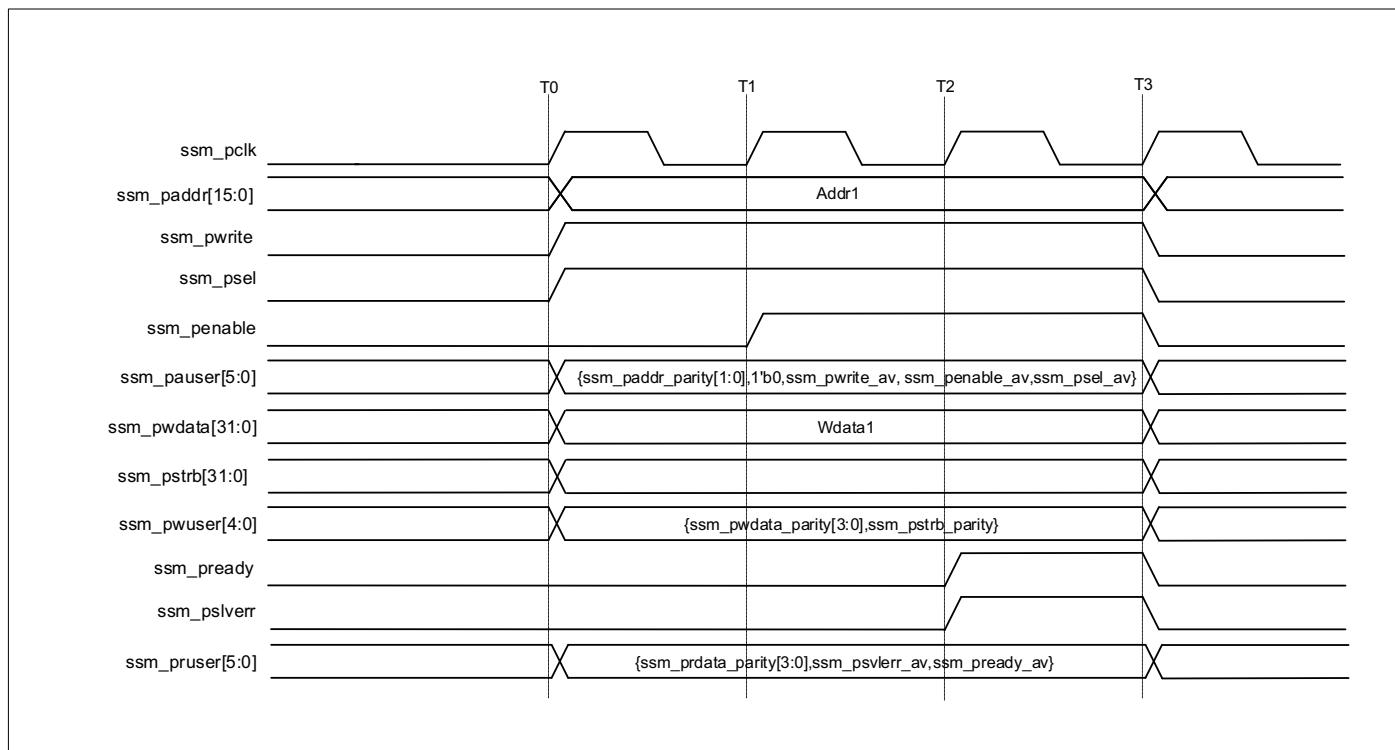
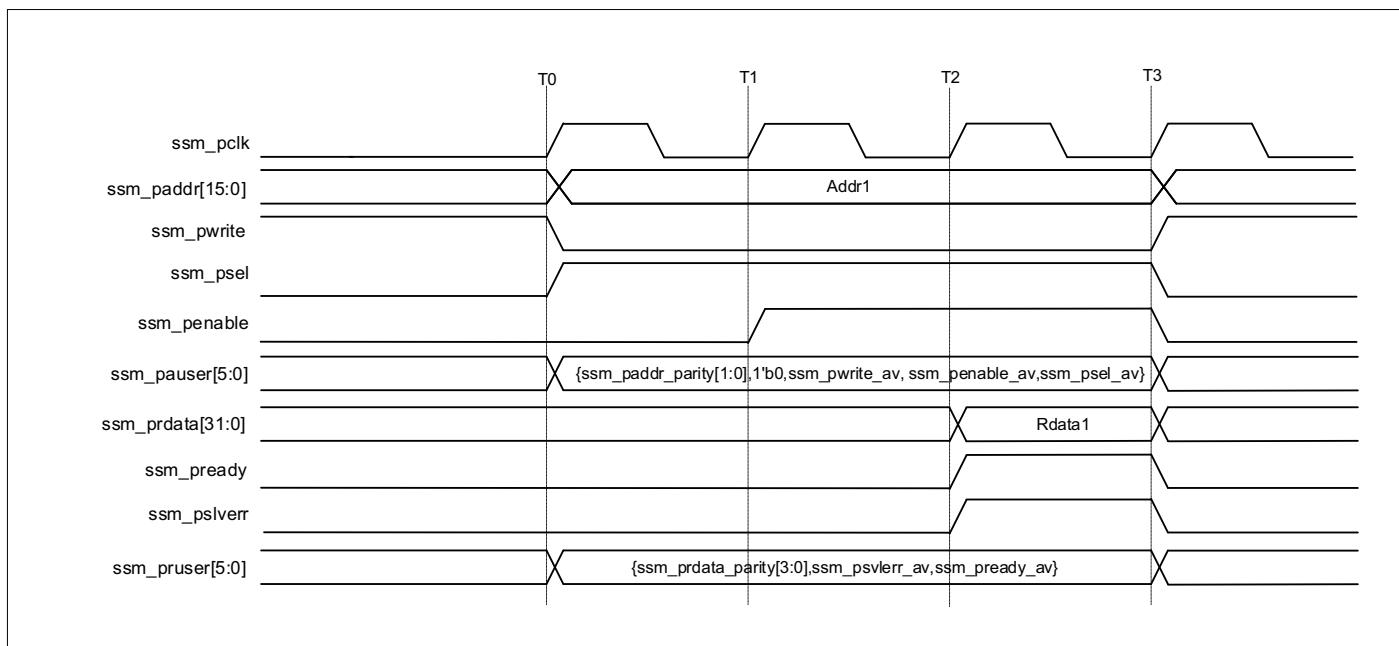


Figure B-4 SSM Bus Read Transfer with Wait State and Error

C

DWC_mipicsi2_device Packet Transmission Calculation

This section describes the packet transmission time calculation in PPI for each mode.

- “LRTE Disable” on page [736](#)
- “LRTE Enable” on page [738](#)

C.1 LRTE Disable

This section describes LRTE disable for SP_time and LP_time.

Figure C-1 SP_Time

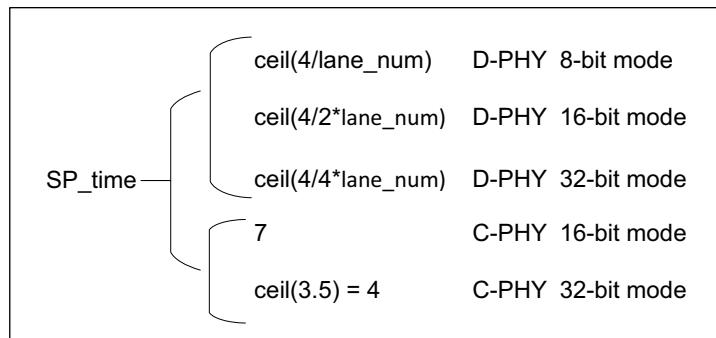
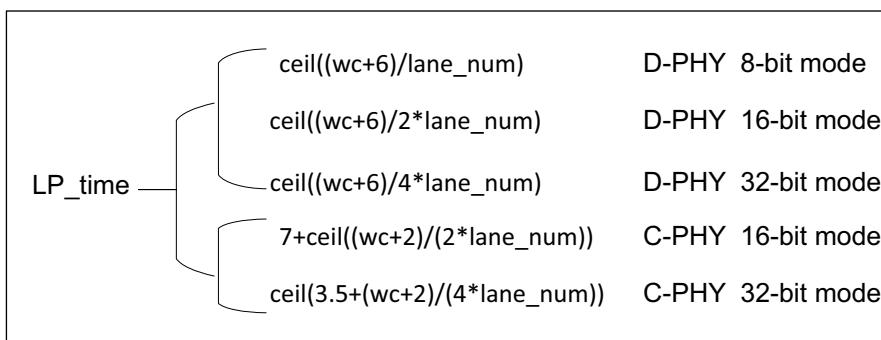


Figure C-2 LP_Time



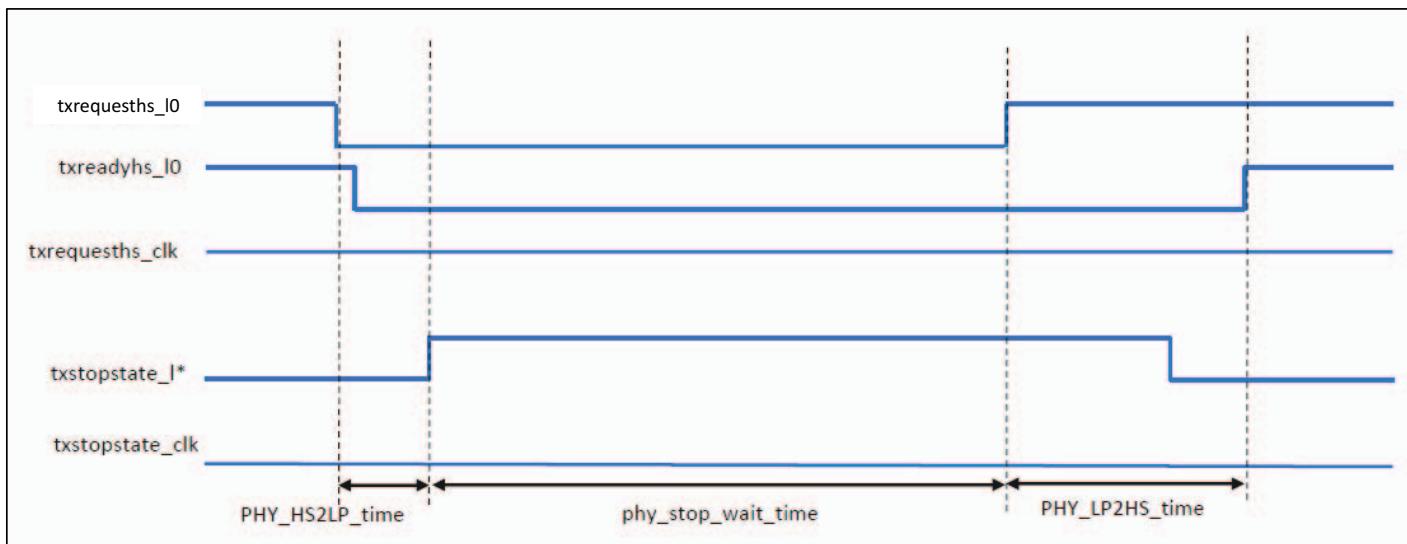
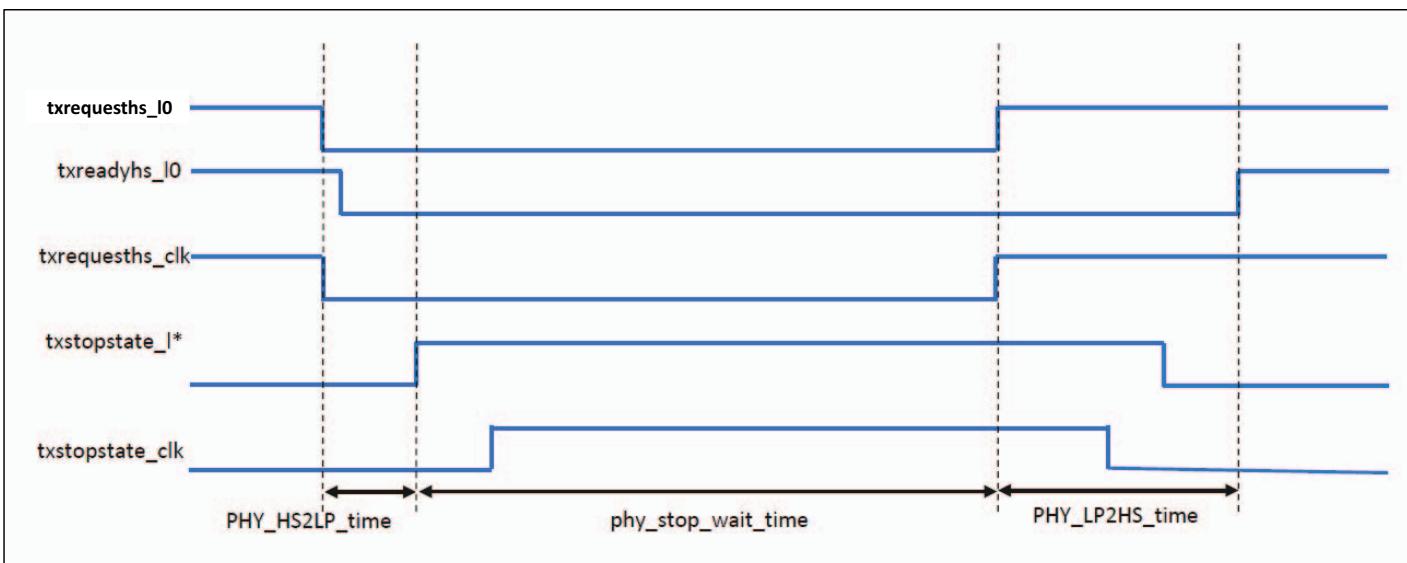
Switch_time = PHY_HS2LP_time + PHY_LP2HS_time + phy_stop_wait_time

Where,

- wc = word count
- SP_trans_time = SP_time + Switch_time
- LP_trans_time = LP_time + Switch_time
- phy_stop_wait_time is the value configured in the register with same name in PHY_IF_CFG. It is the minimum wait period to request a high-speed transmission after all the active lanes stopstate signal are asserted.
- PHY_HS2LP_time is the time from txrequesths_l0 is de-asserted to all the active lanes stopstate signals are asserted.
- PHY_LP2HS_time is the time from txrequesths_l0 is asserted to txreadyhs_l0 is asserted.



For SNPS D-PHY values, refer to the SNPS PHY databook.

Figure C-3 Switch_time in Continuous Clock Lane**Figure C-4 Switch_time in Non Continuous Clock Lane**

C.2 LRTE Enable

This section describes LRTE enable for SP_time and LP-time.

C.2.1 D-PHY EPD Options 1 and 2

Figure C-5 SP_Time

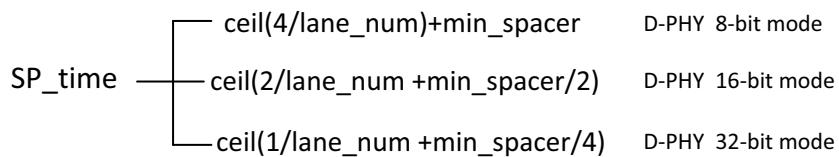
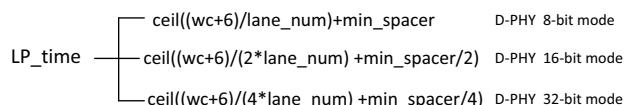


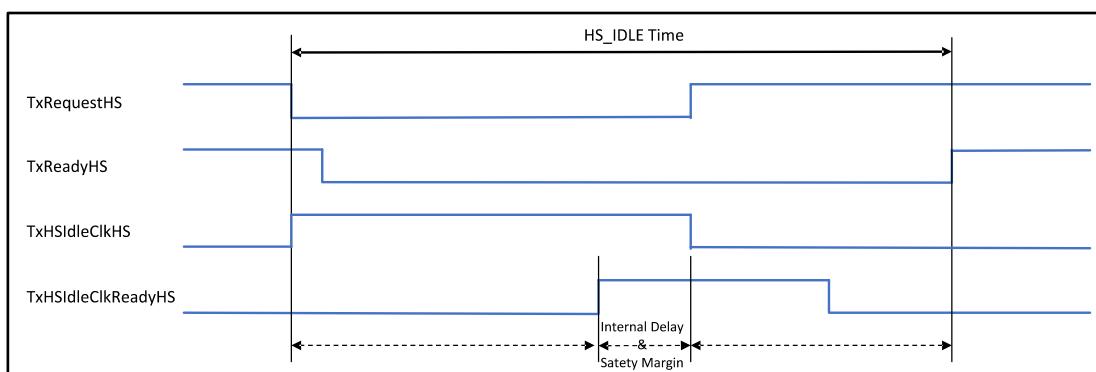
Figure C-6 LP_Time



For D-PHY EPD option 1:

- wc = word count
- SP_trans_time = SP_time + HS_IDLE
- LP_trans_time = LP_time + HS_IDLE
- HS_IDLE time consists of THS-IDLE-POST, THS-IDLE-CLKHS0, THS-IDLE-PRE, THS-ZERO and 10 PPI cycle safety margin.

Figure C-7 HS_IDLE Time Definition



For D-PHY EPD option 2:

- wc = word count

- $SP_{trans_time} = SP_time$
- $LP_{trans_time} = LP_time$

For more details about HS_IDLE description, refer to *MIPI D-PHY Specification*.

C.2.2 C-PHY Option

Figure C-8 SP_Time

$$SP_time \left[\begin{array}{ll} 7 + \text{ceil}(\text{min_spacer}/2) & \text{C-PHY 16-bit mode} \\ \text{ceil}(3.5+\text{min_spacer}/4) & \text{C-PHY 32-bit mode} \end{array} \right.$$

Figure C-9 LP_Time

$$LP_time \left[\begin{array}{ll} 7 + \text{ceil}((wc+2)/(2*\text{lane_num})+\text{min_spacer}/2) & \text{C-PHY 16-bit mode} \\ \text{ceil}(3.5+(wc+2)/4*\text{lane_num}+\text{min_spacer}/4) & \text{C-PHY 32-bit mode} \end{array} \right.$$

Where:

- $wc = \text{word count}$
- $SP_{trans_time} = SP_time$
- $LP_{trans_time} = LP_time$

D

Internal Parameter Descriptions

Provides a description of the internal parameters that might be indirectly referenced in expressions in the Signals, Parameters, or Registers chapters. These parameters are not visible in the coreConsultant GUI and most of them are derived automatically from visible parameters. **You must not set any of these parameters directly.**

Some expressions might refer to TCL functions or procedures (sometimes identified as **function_of**) that coreConsultant uses to make calculations. The exact formula used by these TCL functions is not provided in this chapter. However, when you configure the core in coreConsultant, all TCL functions and parameters are evaluated completely; and the resulting values are displayed where appropriate in the coreConsultant GUI reports.

Table D-1 Internal Parameters

Parameter Name	Equals To
CSI2_DEVICE_CPHY_16BIT_MODE	((CSI2_DEVICE_CPHY_TYPE) & (CSI2_DEVICE_CPHY_WIDTH_MODE >= 1))
CSI2_DEVICE_CPHY_32BIT_MODE	((CSI2_DEVICE_CPHY_TYPE) & (CSI2_DEVICE_CPHY_WIDTH_MODE == 2))
CSI2_DEVICE_CPHY_ALP_CODE_RS	4
CSI2_DEVICE_CPHY_ALP_EN	CSI2_DEVICE_CPHY_TYPE
CSI2_DEVICE_CPHY_ALP_HS_RS	((CSI2_DEVICE_CPHY_32BIT_MODE) ? 2 : 1)
CSI2_DEVICE_CPHY_INCLUDE_LANE_1	((CSI2_DEVICE_CPHY_TYPE) & (CSI2_DEVICE_CPHY_NUM_OF_LANES >= 1))
CSI2_DEVICE_CPHY_INCLUDE_LANE_2	((CSI2_DEVICE_CPHY_TYPE) & (CSI2_DEVICE_CPHY_NUM_OF_LANES >= 2))
CSI2_DEVICE_CPHY_INCLUDE_LANE_3	((CSI2_DEVICE_CPHY_TYPE) & (CSI2_DEVICE_CPHY_NUM_OF_LANES >= 3))
CSI2_DEVICE_CPHY_INCLUDE_LANE_4	((CSI2_DEVICE_CPHY_TYPE) & (CSI2_DEVICE_CPHY_NUM_OF_LANES >= 4))
CSI2_DEVICE_CPHY_LANE_WIDTH	((CSI2_DEVICE_CPHY_32BIT_MODE) ? 32 : 16)

Parameter Name	Equals To
CSI2_DEVICE_CPHY_SS_WIDTH	((CSI2_DEVICE_CPHY_32BIT_MODE) ? 2 : 1)
CSI2_DEVICE_CPHY_ST_WIDTH	3
CSI2_DEVICE_CPHY_TYPE	((CSI2_DEVICE_PHY_TYPE == 2) (CSI2_DEVICE_PHY_TYPE == 3))
CSI2_DEVICE_DPHY_16BIT_MODE	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_WIDTH_MODE >= 2))
CSI2_DEVICE_DPHY_16BIT_OR_CPHY_32BIT_MODE	((CSI2_DEVICE_DPHY_16BIT_MODE) (CSI2_DEVICE_CPHY_32BIT_MODE))
CSI2_DEVICE_DPHY_32BIT_MODE	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_WIDTH_MODE == 3))
CSI2_DEVICE_DPHY_INCLUDE_LANE_1	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_NUM_OF_LANES >= 1))
CSI2_DEVICE_DPHY_INCLUDE_LANE_2	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_NUM_OF_LANES >= 2))
CSI2_DEVICE_DPHY_INCLUDE_LANE_3	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_NUM_OF_LANES >= 3))
CSI2_DEVICE_DPHY_INCLUDE_LANE_4	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_NUM_OF_LANES >= 4))
CSI2_DEVICE_DPHY_INCLUDE_LANE_5	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_NUM_OF_LANES >= 5))
CSI2_DEVICE_DPHY_INCLUDE_LANE_6	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_NUM_OF_LANES >= 6))
CSI2_DEVICE_DPHY_INCLUDE_LANE_7	((CSI2_DEVICE_DPHY_TYPE) & (CSI2_DEVICE_DPHY_NUM_OF_LANES >= 7))
CSI2_DEVICE_DPHY_LANE_WIDTH	((CSI2_DEVICE_DPHY_32BIT_MODE) ? 32 : (CSI2_DEVICE_DPHY_16BIT_MODE) ? 16 : 8)
CSI2_DEVICE_DPHY_TYPE	((CSI2_DEVICE_PHY_TYPE == 1) (CSI2_DEVICE_PHY_TYPE == 3))
CSI2_DEVICE_GEN3DPHY	[function_of:>=4]
CSI2_DEVICE_IDI_DATA_BYTEEN_WIDTH	(CSI2_DEVICE_IDI_DATA_WIDTH == 128 ? 4 : 3)
CSI2_DEVICE_IDI_DATA_PARITY_WIDTH	(CSI2_DEVICE_IDI_DATA_WIDTH == 128 ? 21 : 13)
CSI2_DEVICE_IDI_IF	((CSI2_DEVICE_DATAINTERFACE == 1) (CSI2_DEVICE_DATAINTERFACE == 3))
CSI2_DEVICE_IDI_IPI_IF	(CSI2_DEVICE_DATAINTERFACE == 3)
CSI2_DEVICE_IDI_PLD_FIFO_AWIDTH	[function_of: CSI2_DEVICE_IDI_PLD_RAM_DEPTH]

Parameter Name	Equals To
CSI2_DEVICE_IDI_PLD_FIFO_CDC_FIFO_DEPTH	[function_of: CSI2_DEVICE_IDI_PLD_PUSH_AF_LVL CSI2_DEVICE_DFLT_F_SYNC_TYPE]
CSI2_DEVICE_IDI_PLD_FIFO_PTP_DEPTH	(CSI2_DEVICE_IDI_PLD_FIFO_TOTAL_DEPTH - CSI2_DEVICE_IDI_PLD_FIFO_CDC_FIFO_DEPTH)
CSI2_DEVICE_IDI_PLD_FIFO_PWIDTH	[function_of: CSI2_DEVICE_USE_DRAM CSI2_DEVICE_IDI_PLD_FIFO_AWIDTH CSI2_DEVICE_IDI_PLD_PTP_SRAM_AWIDTH]
CSI2_DEVICE_IDI_PLD_PTP_SRAM_AWIDTH	[function_of: CSI2_DEVICE_IDI_PLD_PTP_SRAM_DEPTH]
CSI2_DEVICE_IDI_PLD_PUSH_AF_LVL	(CSI2_DEVICE_AP+3)
CSI2_DEVICE_IDI_PL_ECC_WIDTH	(CSI2_DEVICE_MAX_PPI_WIDTH > 128 ? 10 : (CSI2_DEVICE_MAX_PPI_WIDTH > 64 CSI2_DEVICE_IDI_DATA_WIDTH > 64) ? 9 : 8)
CSI2_DEVICE_IPI2_IF	(CSI2_DEVICE_NR_IPI >= 2)
CSI2_DEVICE_IPI2_PLD_FIFO_AWIDTH	[function_of: CSI2_DEVICE_IPI2_PLD_RAM_DEPTH]
CSI2_DEVICE_IPI2_PLD_FIFO_CDC_FIFO_DEPTH	[function_of: CSI2_DEVICE_IPI2_PLD_PUSH_AF_LVL CSI2_DEVICE_DFLT_F_SYNC_TYPE]
CSI2_DEVICE_IPI2_PLD_FIFO_PTP_DEPTH	(CSI2_DEVICE_IPI2_PLD_FIFO_TOTAL_DEPTH - CSI2_DEVICE_IPI2_PLD_FIFO_CDC_FIFO_DEPTH)
CSI2_DEVICE_IPI2_PLD_FIFO_PWIDTH	[function_of: CSI2_DEVICE_USE_DRAM CSI2_DEVICE_IPI2_PLD_FIFO_AWIDTH CSI2_DEVICE_IPI2_PLD_PTP_SRAM_AWIDTH]
CSI2_DEVICE_IPI2_PLD_PTP_SRAM_AWIDTH	[function_of: CSI2_DEVICE_IPI2_PLD_PTP_SRAM_DEPTH]
CSI2_DEVICE_IPI2_PLD_PUSH_AF_LVL	(CSI2_DEVICE_AP+5)
CSI2_DEVICE_IPI3_IF	(CSI2_DEVICE_NR_IPI >= 3)
CSI2_DEVICE_IPI3_PLD_FIFO_CDC_FIFO_DEPTH	[function_of: CSI2_DEVICE_IPI3_PLD_PUSH_AF_LVL CSI2_DEVICE_DFLT_F_SYNC_TYPE]
CSI2_DEVICE_IPI3_PLD_FIFO_PTP_DEPTH	(CSI2_DEVICE_IPI3_PLD_FIFO_TOTAL_DEPTH - CSI2_DEVICE_IPI3_PLD_FIFO_CDC_FIFO_DEPTH)
CSI2_DEVICE_IPI3_PLD_PUSH_AF_LVL	(CSI2_DEVICE_AP+5)
CSI2_DEVICE_IPI4_IF	(CSI2_DEVICE_NR_IPI >= 4)
CSI2_DEVICE_IPI4_PLD_FIFO_CDC_FIFO_DEPTH	[function_of: CSI2_DEVICE_IPI4_PLD_PUSH_AF_LVL CSI2_DEVICE_DFLT_F_SYNC_TYPE]
CSI2_DEVICE_IPI4_PLD_FIFO_PTP_DEPTH	(CSI2_DEVICE_IPI4_PLD_FIFO_TOTAL_DEPTH - CSI2_DEVICE_IPI4_PLD_FIFO_CDC_FIFO_DEPTH)

Parameter Name	Equals To
CSI2_DEVICE_IPI4_PLD_PUSH_AF_LVL	(CSI2_DEVICE_AP+5)
CSI2_DEVICE_IPI_DATA_PARITY_WIDTH	(CSI2_DEVICE_IPI_LANE_NUM > 1 ? 27 : 9)
CSI2_DEVICE_IPI_IF	((CSI2_DEVICE_DATAINTERFACE == 2) (CSI2_DEVICE_DATAINTERFACE == 3))
CSI2_DEVICE_IPI_MULTI_LANE	(CSI2_DEVICE_IPI_LANE_NUM > 1)
CSI2_DEVICE_IPI_PLD_FIFO_AWIDTH	[function_of: CSI2_DEVICE_IPI_PLD_RAM_DEPTH]
CSI2_DEVICE_IPI_PLD_FIFO_CDC_FIFO_DEPTH	[function_of: CSI2_DEVICE_IPI_PLD_PUSH_AF_LVL CSI2_DEVICE_DFLT_F_SYNC_TYPE]
CSI2_DEVICE_IPI_PLD_FIFO_PTP_DEPTH	(CSI2_DEVICE_IPI_PLD_FIFO_TOTAL_DEPTH - CSI2_DEVICE_IPI_PLD_FIFO_CDC_FIFO_DEPTH)
CSI2_DEVICE_IPI_PLD_FIFO_PWIDTH	[function_of: CSI2_DEVICE_USE_DRAM CSI2_DEVICE_IPI_PLD_FIFO_AWIDTH CSI2_DEVICE_IPI_PLD_PTP_SRAM_AWIDTH]
CSI2_DEVICE_IPI_PLD_PTP_SRAM_AWIDTH	[function_of: CSI2_DEVICE_IPI_PLD_PTP_SRAM_DEPTH]
CSI2_DEVICE_IPI_PLD_PUSH_AF_LVL	(CSI2_DEVICE_AP+5)
CSI2_DEVICE_IPI_PL_ECC_WIDTH	((CSI2_DEVICE_IPI_LANE_NUM > 1) (CSI2_DEVICE_MAX_PPI_WIDTH > 128)) ? 10 : CSI2_DEVICE_MAX_PPI_WIDTH > 64 ? 9 : 8)
CSI2_DEVICE_LANE_NUM_QST_RST_VAL	(CSI2_DEVICE_DPHY_TYPE ? (CSI2_DEVICE_DPHY_NUM_OF_LANES-1) : (CSI2_DEVICE_CPHY_NUM_OF_LANES-1))
CSI2_DEVICE_MAX_PPI_WIDTH	[function_of: CSI2_DEVICE_PHY_TYPE CSI2_DEVICE_DPHY_WIDTH_MODE CSI2_DEVICE_DPHY_NUM_OF_LANES CSI2_DEVICE_CPHY_WIDTH_MODE CSI2_DEVICE_CPHY_NUM_OF_LANES]
CSI2_DEVICE_MULTI_IF	((CSI2_DEVICE_IDI_IPI_IF == 1) (CSI2_DEVICE_NR_IPI >= 2))
CSI2_DEVICE_NAP_IDI_PL_FIFO_WIDTH	(CSI2_DEVICE_MAX_PPI_WIDTH > 128 ? 256 : (CSI2_DEVICE_MAX_PPI_WIDTH > 64 CSI2_DEVICE_IDI_DATA_WIDTH > 64) ? 128 : 64)
CSI2_DEVICE_NAP_IPI_PL_FIFO_WIDTH	(CSI2_DEVICE_IPI_LANE_NUM > 1 ? 256 : CSI2_DEVICE_MAX_PPI_WIDTH > 64 ? 128 : 64)
CSI2_DEVICE_NAP_PLD_WIDTH	(CSI2_DEVICE_MAX_PPI_WIDTH > 128 ? 256 : CSI2_DEVICE_MAX_PPI_WIDTH > 64 ? 128 : 64)
CSI2_DEVICE_PHY_16BIT_MODE	((CSI2_DEVICE_DPHY_16BIT_MODE) (CSI2_DEVICE_CPHY_16BIT_MODE))

Parameter Name	Equals To
CSI2_DEVICE_PHY_LANE0_WIDTH	[function_of: CSI2_DEVICE_PHY_TYPE CSI2_DEVICE_DPHY_LANE_WIDTH CSI2_DEVICE_CPHY_LANE_WIDTH]
CSI2_DEVICE_PHY_VALID_WIDTH	((CSI2_DEVICE_DPHY_32BIT_MODE) ? 4 : CSI2_DEVICE_DPHY_16BIT_OR_CPHY_32BIT_MODE ? 2 : 1)
CSI2_DEVICE_PL_ERR_INJ_WIDTH_GT_128	((CSI2_DEVICE_IPI_LANE_NUM > 1) (CSI2_DEVICE_MAX_PPI_WIDTH > 128))
CSI2_DEVICE_PL_ERR_INJ_WIDTH_GT_64	((CSI2_DEVICE_IPI_LANE_NUM > 1) (CSI2_DEVICE_MAX_PPI_WIDTH > 64) (CSI2_DEVICE_IDI_DATA_WIDTH > 64))
CSI2_DEVICE_SDI_IF	(CSI2_DEVICE_DATAINTERFACE == 4)
CSI2_DEVICE_SDI_IN_PARITY_WIDTH	(CSI2_DEVICE_MAX_PPI_WIDTH > 128 ? 33 : (CSI2_DEVICE_MAX_PPI_WIDTH > 64 ? 17 : 9))
CSI2_DEVICE_SDI_PLD_BYTEEN_WIDTH	(CSI2_DEVICE_MAX_PPI_WIDTH > 128 ? 6 : (CSI2_DEVICE_MAX_PPI_WIDTH > 64 ? 5 : 4))
CSI2_DEVICE_USE_DPRAM	(CSI2_DEVICE_EXT_RAM_TYPE == 1)
CSI2_DEVICE_USE_SPRAM	(CSI2_DEVICE_EXT_RAM_TYPE == 2)
CSI2_DEVICE_VCX_DWIDTH	((CSI2_DEVICE_CPHY_TYPE) ? 3 : 2)
CSI2_DEVICE_VER_NUMBER	0x132
CSI2_DEVICE_VERSION_ID	0x3133322a
CSI2_DEVICE_VER_TYPE_ENUM	0x0
CSI2_DEVICE_VER_TYPE_NUM	0x0
CSI2_DEVICE_VER_TYPE_PKG	0xa
SS_CORE_REG_DEPTH	47
SS_CORE_REG_START_OFFSET	32

