DIPORT CONTROLLER Specification

VERSION Draft (16.Sept.2022)

EDITOR(S):

Gary Miller, NXP

Executive Summary

The diPort controller (diPort) is intended to provide a scalable chiplet interface controller for use with IO coherent accelerators. diPort is intended to add runtime processor communication and signaling between two die via designated serial and parallel interface protocols.

DiPort is optimized for die-to-die communications and serves as an ecosystem for Systems-in-Package. It enables automatic configuration of communication at start-up. The connected die appears to software as on-die resources using built-in memory map address translation. Error detection, corruption prevention and resiliency are supported.

This initial specification captures NXP diPort content, which supports a virtual interconnection of an AXI bus between two die, as well as signaling of discrete information between die. IO coherent support (e.g., ACE-lite) will be added in the next specification release. The diPort controller is, however, intended to interface with any on-die bus definition to provide a virtual connection between die.

About Open Compute Foundation

Contents

1 License

1.1 OCP CLA

1.2 Acknowledgements

The Contributors of this Specification would like to acknowledge the following companies for their feedback:

David Kruckemyer, Ventana Micro

David Kehlet, Intel

2 Revision Table

Revision #	Date	Author	Description
NXP definition contribution (draft)	24 Feb 2020 Draft	Gary Miller	• This initial release is only to provide a baseline • ODSA additional features will be provided in next release
ODSA initial review feedback incorporated (draft)	19 Jan 2021 Draft	Gary Miller	• Incorporated extensive OSDS feedback from David Kruckemyer and David Kehlet • NXP considering donating diPort design collateral. If approved, this specification is intended to be initial version • Architected all planned future messages with future and current implementation plans documented • Added registers and simplified process for enabling chips from different companies to establish link • Proposed BoW integration using LPIF interface
Editorial changes (draft)	16 Sept 2022 Draft	Gary Miller	• Editorial changes • Minor technical corrections

3 Glossary

Controller or diPortSD Controller	The diPortSD (SD means serial data) controller is the official IP name that appears in various sections and diagrams. The name diPort controller, a shortened version, is used as well.
Device, Die and Chiplet	The diPort controller is intended for use with homogeneous and heterogeneous devices, die and chiplets. The term chip will be used in most instances to universally represent product device, die and chiplet, whichever may be applicable.
PHY	Physical interface for chip-to-chip communication. A PHY comprises logic (PCS) and analog (PMA) portions, and one or the other may be inferred depending upon the context.
Signaling	Signaling aspects include transferring any general signaling information between the two chips (e.g., interrupt request and system information signals).
SiP	System in Package refers to a package with two or more chips.
X Bar	Bus crossbar switch that provides the hardware interconnect matrix routing transactions from bus masters to bus slaves.

4 Introduction

The diPortSD controller (diPort) is intended to provide a scalable chip-to-chip controller for use with homogeneous and heterogeneous devices, die as well as chiplets. The term chip will be used in most instances to universally represent product device, die and chiplet, whichever may be most applicable. There are myriads of applications such as:

- Non-coherent accesses between chips, where AXI transactions are transported. An example is accesses between two general chips, each containing local masters accessing memory mapped resources of the remote chip.
- IO coherent access, where AXI with ACE-Lite extensions are transported. An example is accesses between a main chip and an accelerator chip. The main chip may setup the accelerator configuration to initiate a function to be accelerated. The accelerator chip may access cached information on the main chip, and write back results when complete. When write-back is completed, the accelerator may trigger a coherent event on the main chip using barriers to assure the main chip caches are coherent.

NOTE

This version supports only non-coherent access via AXI transactions. A future specification will add ACE-Lite extensions for IO coherent accesses.

diPort is intended to add runtime processor communication and signaling between two chips. Communication aspects include ease of software access to code, data, and peripheral information between a local chip and a remote chip. Signaling aspects include access to signals between a local chip and a remote chip.

diPort resides on both chips that are interconnected via a PHY in a package. This allows for masters on either chip to access memory mapped resources on the remote chip.

As mentioned diPort supports both runtime processor communication and signaling between two chips. This arrangement works adequately unless the number of signals required or change rate of the signals becomes too high.

In a typical SiP configuration, diPort messages are transported via a PHY between two chips. Internal bus transactions on the local chip targeted to a resource on the remote chip are automatically translated and transported via diPort messages. Furthermore, the diPort bi-directionally transports signaling of necessary signal information between chips, such as system status and interrupt requests.

The diPort is latency- and bandwidth-optimized for typical master-to-peripheral access scenarios, efficiently handling single word accesses as well as burst transfers. Furthermore, another objective is to most efficiently packetize AXI transactions, using the minimum amount of AXI bus information for each of the 5 defined AXI channels, to provide the optimal latency and bandwidth between chips.

The diPort controller may be integrated directly with a PHY under certain conditions or integrated with a link layer. The diPort controller implements link layer functions intended to be used with a low error rate PHY technology. If the PHY technology has a high error rate then including FEC or integrating with a link layer with retry support is advisable.

4.1 Block Diagram

An illustration of how diPort may be integrated into SiP is shown below:

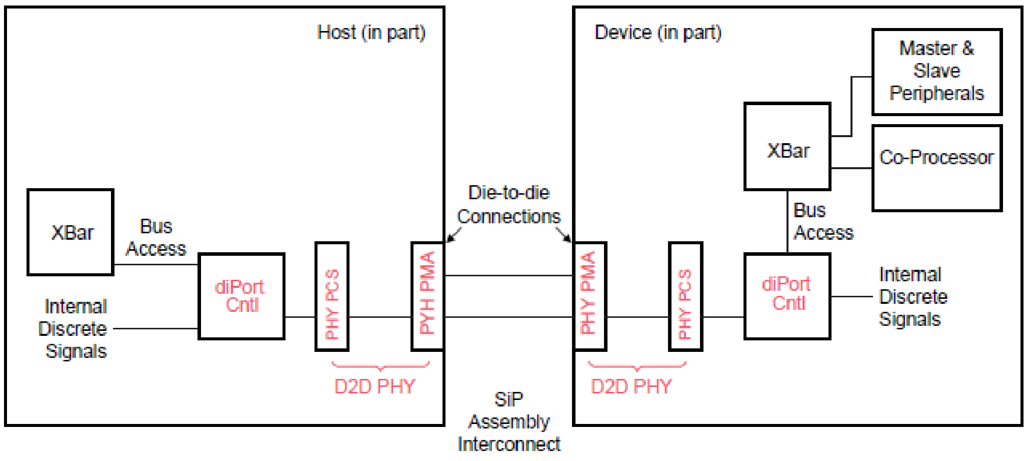


Figure 1. diPort subsystem illustration

Using parametrization diPort may be instantiated to be connected on-die to:

- AXI master
- AXI slave
- Both AXI master and AXI slave

The following block diagrams illustrate a connection to a chip that acts as both AXI master and AXI slave.

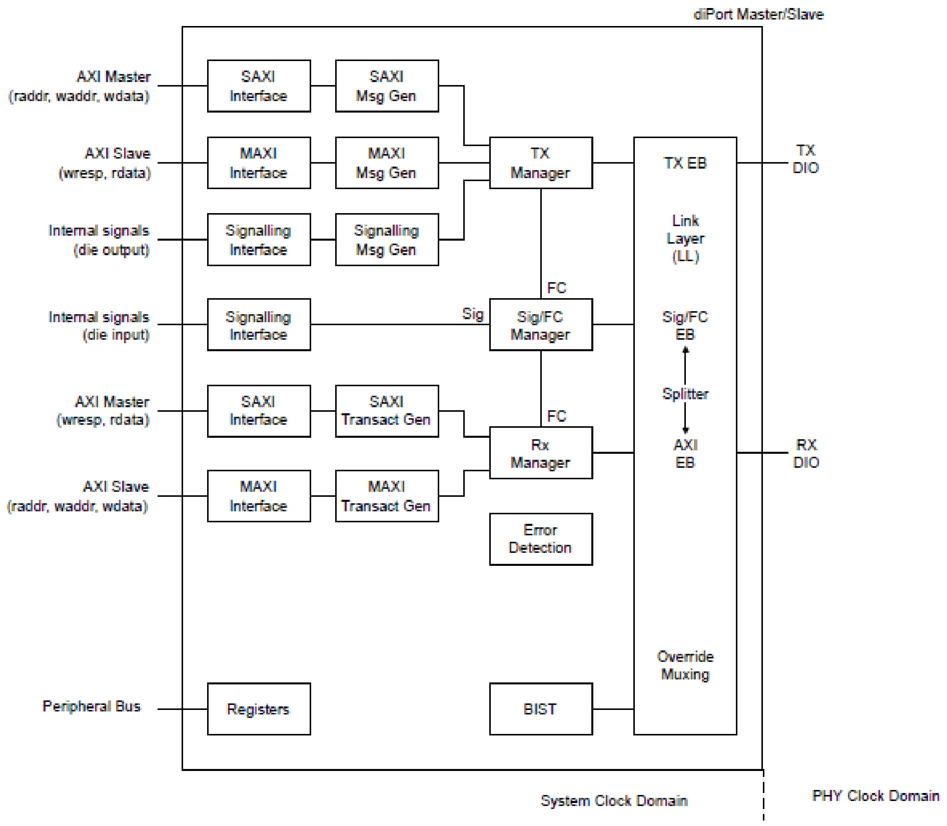


Figure 2. diPort master and slave block diagram

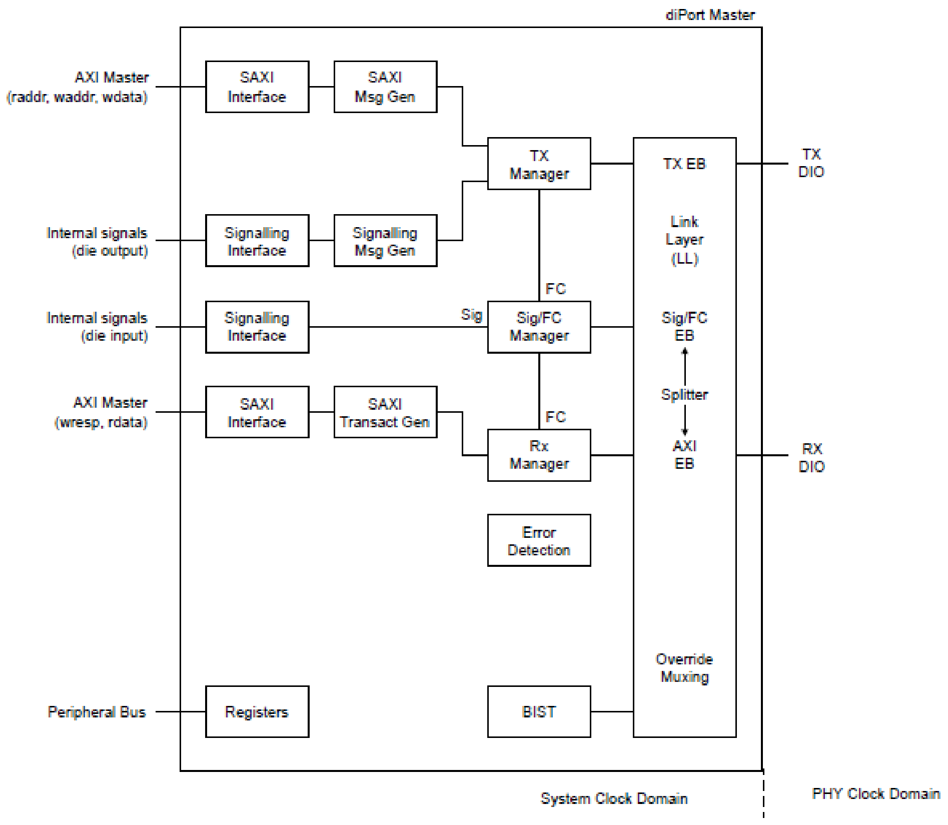


Figure 3. diPort Master block diagram

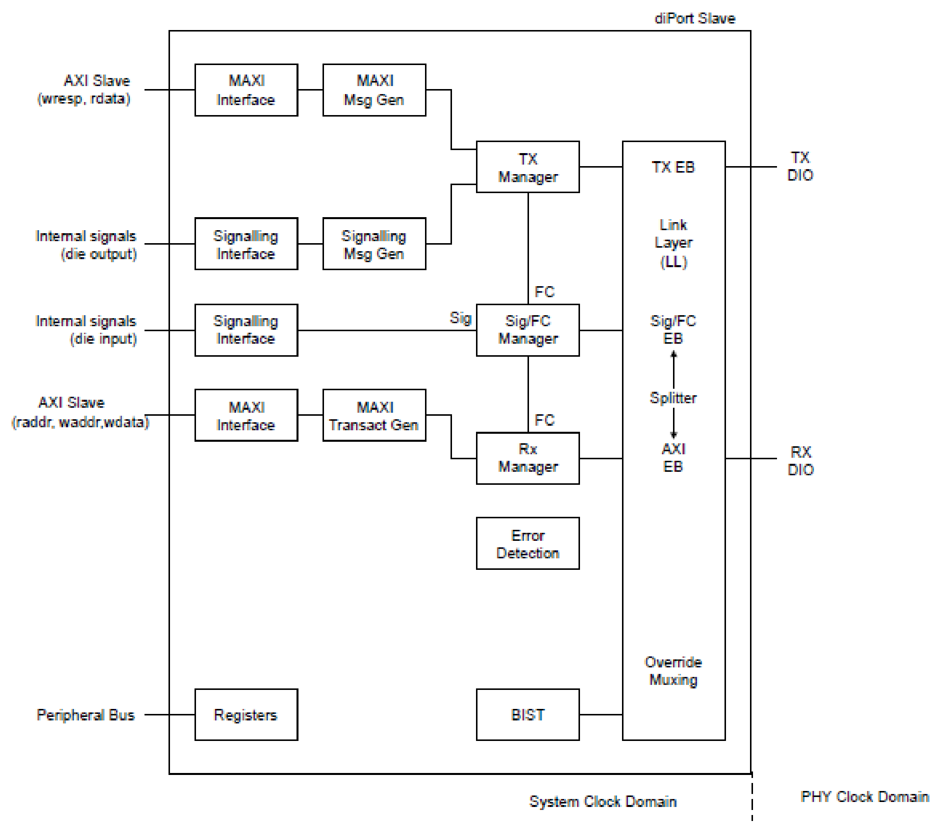


Figure 4. diPort slave block diagram

Block	**Description**
diPort	diPort controller
DIO	diPort chip-to-chip digital I/O
SAXI Interface	Slave AXI Interface is state logic to interface to a master's read address, write address, read data, write data and response AXI channels
MAXI Interface	Master AXI Interface is state logic to interface to a slave's write response read data AXI channels
SAXI Msg Gen	Slave AXI message generation logic to produce associated messages as received on read address, write address and write data AXI channels
MAXI Msg Gen	Master AXI message generation logic to produce associated messages as received on write response read data AXI channels
SAXI Transact Gen	Slave AXI message generation logic to produce associated AXI transactions as received in messages pertaining to write response and read data
MAXI Transact Gen	Master AXI message generation logic to produce associated AXI transactions as received in messages pertaining to read address, write address and write data
TX Manager	Manages which AXI message to transmit
Sig/FC Manager	Manages which signaling or flow control message to transmit
BIST	At-speed testing through the on-die PHY pads and across chip-to-chip assembly connections
Link Layer	Performs buffering via elasticity buffers (EB) as well as translation of physical data to logic data and vice versa. May also be connected with a link layer supporting retry.
RX and TX DIO	Provides the chip-to-chip optimized I/O interface
Peripheral Bus	NXP register access bus for reading and writing configuration and status registers
Sig/FC Manager	Manages which signaling or flow control message to transmit

Table 1. Block diagram legend

4.2 Features

- Optimized for chip-to-chip communication - enables ecosystem for SiPs
- Automatic configuration of communication at start-up
- — A limited amount of software configuration of register is needed for some features
- Connected chip appears to software as on-die resources using built-in memory map address translation
- Provides a virtual interconnection of AXI bus between two chips
- — Note that the current design requires the AXI bus on both chips to have the same data width
- Supports signaling of hundreds of system states, messages, etc. between two chips
- — Automatic replication on remote chip

  — Supports clock-domain crossing for scalar and vectors

  — Quality of service for decoupling signaling impact on AXI traffic
- Error detection, corruption prevention and resiliency (e.g., functional safety)
- Optimized for AXI channels protocol

  — Supports AXI4 - plan to add support for ACE-Lite for managing IO coherency

  — Tracks AXI state flow with a minimal amount of AXI attributes transferred between chips

  — Minimal packetizing delays for both single and burst transactions

  — Pipelining for many simultaneous reads and writes for improved latency and bandwidth

  — Virtual-linked/hardware-synchronized AXI bus channels between chips

  — Current design requires the AXI bus on both chips to have the same data bus width

4.3 Operating Modes

diPort module supports the following basic operating modes:

- Disabled
- BIST mode
- Normal mode.

4.3.1 Disabled Mode

Disabled mode is used when a die mask set is not intended for use in a SiP. diPort provides for configuration via a top-level enable input, that when negated disables diPort IP. To enable diPort IP the top-level enable input must be asserted.

4.3.2 BIST Mode

BIST mode is entered only when test mode is enabled and a diPort BIST test mode is selected. BIST mode enables at-speed testing through the on-die micro-pads and across die-to-die assembly connections. BIST mode has a higher priority than Normal mode.

Requirements to be able to use BIST mode and BIST capabilities are:

- To use on-die loopback BIST testing, the on-die PHY must support loop back from TX to RX
- To use chip-to-chip loopback BIST testing, the remote PHY must support loop back from RX to TX

4.3.3 Normal Mode

AXI transactions and signaling of information between each die is fully functional.

When transitioning into Normal mode, customer software running on each of the two interconnected chips may have to provide software configuration steps to initialize diPort features, especially if the chips are from different companies. Refer to the section Memory Map Integration and Initialization

Also, refer to Normal Operation section for a detailed description of the various aspects of normal mode operation.

5 diPortSD register descriptions

NOTE

1. Write accesses other than 32-bit accesses are not supported. When attempted, no change is made to the register value and no transfer error is produced.
2. Generally, any access to unimplemented register space results in a transfer error. This is not the case for the unimplemented register at offset 188h, which does not generate a transfer error.

5.1 diPortSD memory map

d_ip_diPort_sd_syn base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
100h	Error and Control Register (ECR)	32	RW	0000_0000h
104h	Error Status Register (SESR)	32	W1C	0000_0000h
108h	Quality Of Service (QOS)	32	RW	0000_1801h
10Ch	Signaling Enable (SIGEN)	32	RW	0000_0000h
118h	Signaling Delay (SIGD)	32	RW	0000_0002h
11Ch	Elasticity Buffer Configuration (EBCFG)	32	RW	0000_0000h
128h	Logical Address Region 0 (LAR0)	32	RW	5500_0000h
12Ch	Physical Address Region 0 (PAR0)	32	RW	0000_0000h
130h	Region Size 0 (RS0)	32	RW	0000_000Bh
134h	Logical Address Region 1 (LAR1)	32	RW	5540_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
138h	Physical Address Region 1 (PAR1)	32	RW	0040_0000h
13Ch	Region Size 1 (RS1)	32	RW	0000_000Ah
140h	Logical Address Region 2 (LAR2)	32	RW	5560_0000h
144h	Physical Address Region 2 (PAR2)	32	RW	0060_0000h
148h	Region Size 2 (RS2)	32	RW	0000_0009h
14Ch	Logical Address Region 3 (LAR3)	32	RW	5570_0000h
150h	Physical Address Region 3 (PAR3)	32	RW	0070_0000h
154h	Region Size 3 (RS3)	32	RW	0000_0009h
158h	Logical Address Region 4 (LAR4)	32	RW	5580_0000h
15Ch	Physical Address Region 4 (PAR4)	32	RW	0080_0000h
160h	Region Size 4 (RS4)	32	RW	0000_000Ch
164h	Logical Address Region 5 (LAR5)	32	RW	5600_0000h
168h	Physical Address Region 5 (PAR5)	32	RW	2000_0000h
16Ch	Region Size 5 (RS5)	32	RW	0000_000Ch
170h	Logical Address Region 6 (LAR6)	32	RW	5680_0000h
174h	Physical Address Region 6 (PAR6)	32	RW	4000_0000h
178h	Region Size 6 (RS6)	32	RW	0000_000Ch
17Ch	Logical Address Region 7 (LAR7)	32	RW	0000_0000h
180h	Physical Address Region 7 (PAR7)	32	RW	0000_0000h
184h	Region Size 7 (RS7)	32	RW	0000_0000h
188h	Logical Address Region Upper 0 (LARU0)	32	RW	5500_0000h
18Ch	Physical Address Region Upper 0 (PARU0)	32	RW	5500_0000h
190h	Logical Address Region Upper 1 (LARU1)	32	RW	5540_0000h
194h	Physical Address Region Upper 1 (PARU1)	32	RW	5540_0000h
198h	Logical Address Region Upper 2 (LARU2)	32	RW	5560_0000h
19Ch	Physical Address Region Upper 2 (PARU2)	32	RW	5560_0000h
1A0h	Logical Address Region Upper 3 (LARU3)	32	RW	5570_0000h
1A4h	Physical Address Region Upper 3 (PARU3)	32	RW	5570_0000h
1A8h	Logical Address Region Upper 4 (LARU4)	32	RW	5580_0000h
1ACh	Physical Address Region Upper 4 (PARU4)	32	RW	5580_0000h
1B0h	Logical Address Region Upper 5 (LARU5)	32	RW	5600_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1B4h	Physical Address Region Upper 5 (PARU5)	32	RW	5600_0000h
1B8h	Logical Address Region Upper 6 (LARU6)	32	RW	5680_0000h
1BCh	Physical Address Region Upper 6 (PARU6)	32	RW	5680_0000h
1C0h	Logical Address Region Upper 7 (LARU7)	32	RW	0000_0000h
1C4h	Physical Address Region Upper 7 (PARU7)	32	RW	0000_0000h
200h	BIST Control and Pattern (BIST_CP)	32	RW	0000_0000h
204h	BIST Status (BIST_ST)	32	RO	0000_0000h
208h	BIST Failures (BIST_FLS)	32	RO	0000_0000h
20Ch	BIST Upper Failing Data (BIST_UFD)	32	RO	0000_0000h
210h	BIST Lower Failing Data (BIST_LFD)	32	RO	0000_0000h

5.2 Error and Control Register (ECR)

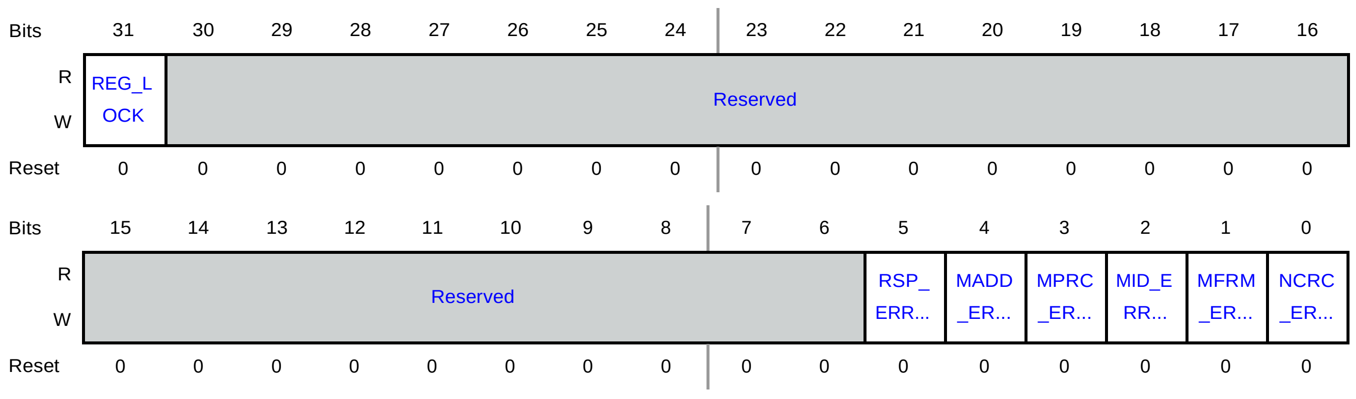
Offset

Register	Offset
ECR	100h

Function

The ECR is used for general control bits as well as to enable the error signal outputs of the diPortSD module. This register is initialized by resetting the controller.

Diagram



Fields

Field	Function
31 REG_LOCK	Register Lock The set-once control bit locks the contents of selected configuration registers when set. The control bit is cleared by reseting the controller. 0b - When cleared at reset, writes can update contents of LAR, PAR, SIGD and BIST registers. 1b - When set, locks the contents of LAR, PAR, SIGD and BIST registers.
30-6 —	Reserved
5 RSP_ERR_EN	Response Error Enable 0b - No error signal is generated 1b - Detection of response error generates an error signal
4 MADD_ERR_EN	Message Address Error Enable 0b - No error signal is generated 1b - Detection of response error generates an error signal
3 MPRC_ERR_EN	Message Processing Error Enable 0b - No error signal is generated 1b - Detection of message processing error generates an error signal
2 MID_ERR_EN	Message ID Error Enable 0b - No error signal is generated 1b - Detection of message ID error generates an error signal
1 MFRM_ERR_EN	Message Frame Error Enable 0b - No error signal is generated 1b - Detection of message frame error generates an error signal
0 NCRC_ERR_EN	Normal Mode CRC Error Enable 0b - No error signal is generated 1b - Detection of Normal mode CRC error generates an error signal

5.3 Error Status Register (SESR)

Offset

Register	Offset
SESR	104h

Function

The SESR indicates the occurrence of an error due to a random hardware fault. This register is initialized by reseting the controller.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										RSP_ ERR	MADD _ERR	MPRC _ERR	MID_E RR	MFRM _ERR	NCRC _ERR
W											W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5 RSP_ERR	Response Error Status If, due to a random hardware fault, the remote chiplet does not respond as expected to an AXI message (i.e., not sending write acknowledge back), this error condition is detected. 0b - Cleared by writing logic 1 1b - Set when a response error is detected during Normal mode
4 MADD_ERR	Message Address Error Status A message address error may occur due to a random hardware fault. This pertains to the ARADDR and AWADDR fields in read and write address messages. These fields are checked at the remote chiplet to assure that after transport they properly reside within the physical address space defined. 0b - Cleared by writing logic 1 1b - Set when a message address error is detected during Normal mode
3 MPRC_ERR	Message Processing Error Status A message processing error may occur due to a random hardware fault. A message processing error is detected if anywhere along the data path from the physical to logic there is a FIFO overrun. 0b - Cleared by writing logic 1 1b - Set when a message error is reported during Normal mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 MID_ERR	Message ID Error Status A message identification error may occur due to a random hardware fault. This pertains to the MID field in each message used to identify message type and form. A message ID error occurs if the message ID is not registered as a valid ID. 0b - Cleared by writing logic 1 1b - Set when a message ID error is reported during Normal mode
1 MFRM_ERR	Message Frame Error Status A message frame error may occur due to a random hardware fault. A framing error is detected if the message form is not correct due to an incomplete number of message bytes received. 0b - Cleared by writing logic 1 1b - Set when a message frame error is reported during Normal mode
0 NCRC_ERR	Normal Mode CRC Error Status A CRC error may be detected due to a random hardware fault. 0b - Cleared by writing logic 1 1b - Set when a CRC error is reported during Normal mode

5.3 Quality Of Service (QOS)

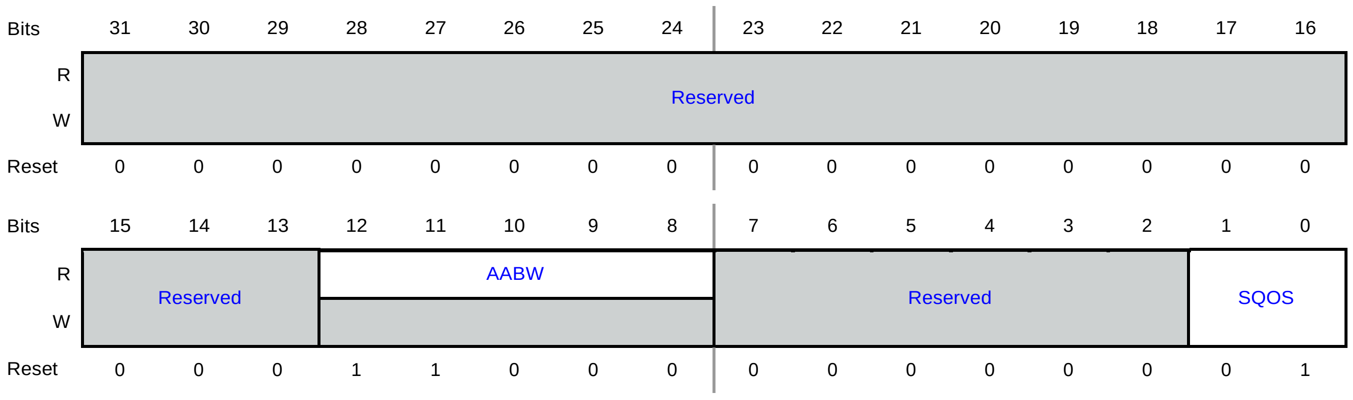
Offset

Register	Offset
QOS	108h

Function

This register is used for over-riding the QOS selected parameter setting. This register is initialized by reseting the controller.

Diagram



Field	Function
31-13 —	Reserved
12-8 AABW	Active AXI Bandwidth The AABW field provides the active or instantaneous calculation for AXI bandwidth over the last 24 double bytes of controller message generation. The range is 0 to 24 where each count represents 4.166% of AXI bandwidth.
7-2 —	Reserved
1-0 SQOS	Quality of Service The SQOS field allows for software to override the SQoS setting described in the Signaling section. 00b - Change to 50% bandwidth to both AXI and Signaling messages 01b - Change to 75% bandwidth to AXI messages and 25% bandwidth to Signaling messages 10b - Change to 87.5% bandwidth to AXI messages and 12.5% bandwidth to Signaling messages 11b - Change to 100% bandwidth to AXI messages

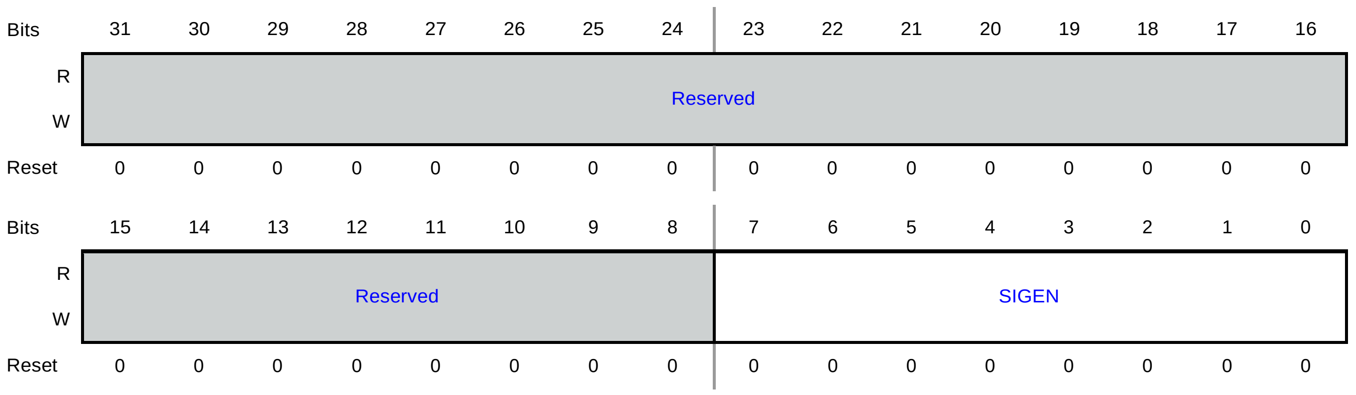
5.4 Signaling Enable (SIGEN)

Offset

Register	Offset
SIGEN	10Ch

Function
This register is used for overriding the settings. This register is initialized by reseting the controller. When set, it enables signaling events (for example, signal state transition) input to the diPortSD to be detected on the local chiplet where SIGEN was set. Once detected, the resultant signal state is mirrored on the remote chiplet via a signaling message that is transported between chiplets. An initialization process is initiated when first set, in order to initialize states on the remote chiplet. To enable signaling events on both chiplets, the SIGEN register must be set on both chiplets. Note that signaling should only be enabled after all diPort related clock frequencies on both chiplets are fully ramped.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 SIGEN	Signaling Enable This field is used for over-riding the default SIGEN setting. This register is initialized by reseting the controller. If written to 0xFF all signaling is enabled.

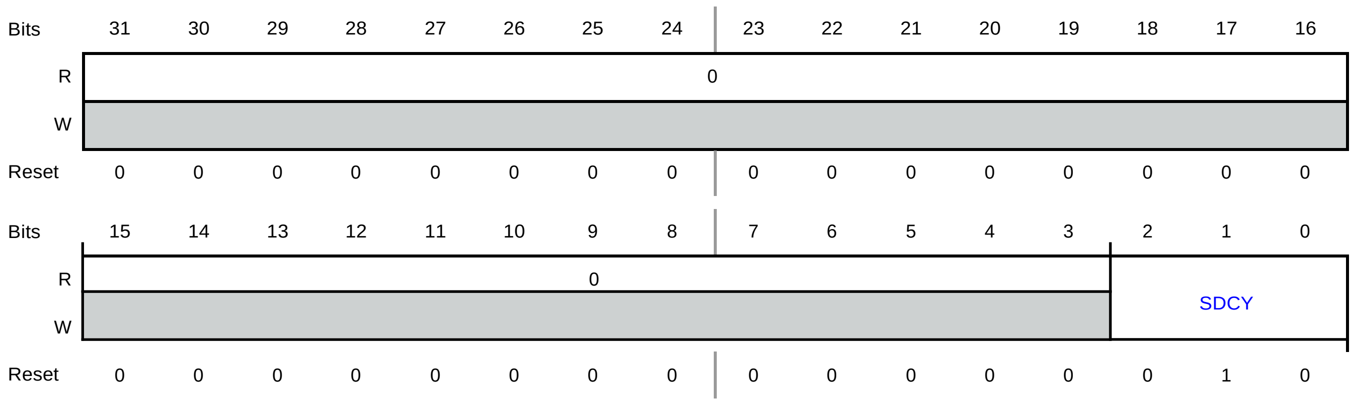
5.5 Signaling Delay (SIGD)

Offset

Register	Offset
SIGD	118h

Function
Use this register to specify the number of AXI bus clocks between signaling messages.

Diagram



Field	Function
31-3 —	Reserved
2-0 SDCY	Signaling Delay Cycles. This field defines the number of AXI bus clocks between signaling messages. The setting provides a bandwidth limit to signaling messages and is applied when there are no competing requests for AXI messages. It assures signaling messages do not exceed 50 % of total diPortSD bus bandwidth.

5.6 Elasticity Buffer Configuration (EBCFG)

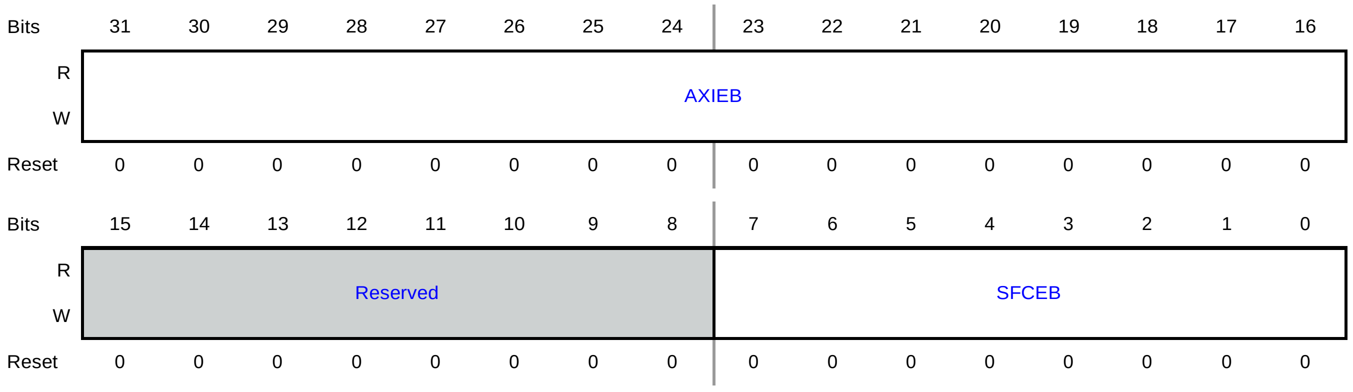
Offset

Register	Offset
EBCFG	11Ch

Function

Use this register to read the implemented size for the AXI and Signaling/FC Elasticity Buffers and re-configure if needed. The reset values indicate the implemented size for each elasticity buffer. The size for each buffer may be reduced as needed to match the size for the corresponding elasticity buffer on the remote chiplet. The AXI and Signaling/FC Elasticity Buffer sizes for both chiplets must be set to match to assure proper operation.

Diagram



Fields

Field	Function
31-16 AXIEB	AXI Elasticity Buffer This field is used for reading the implemented size for the AXI Elasticity Buffer and for re-configuring the size if needed to match the remote chiplets buffer size. The reset value indicates the implemented size, which may be reduced as needed to match the size for the corresponding elasticity buffer on the remote chiplet. The size may not be increased beyond the reset value. The field represents the size in double bytes.
15-8 —	Reserved
7-0 SFCEB	Signaling/FC Elasticity Buffer This field is used for reading the implemented size for the Signaling/FC Elasticity Buffer and for re-configuring the size if needed to match the remote chiplets buffer size. The reset value indicates the implemented size, which may be reduced as needed to match the size for the corresponding elasticity buffer on the remote chiplet. The size may not be increased beyond the reset value. The field represents the size in double bytes.

5.7 Logical Address Region 0 (LAR0)

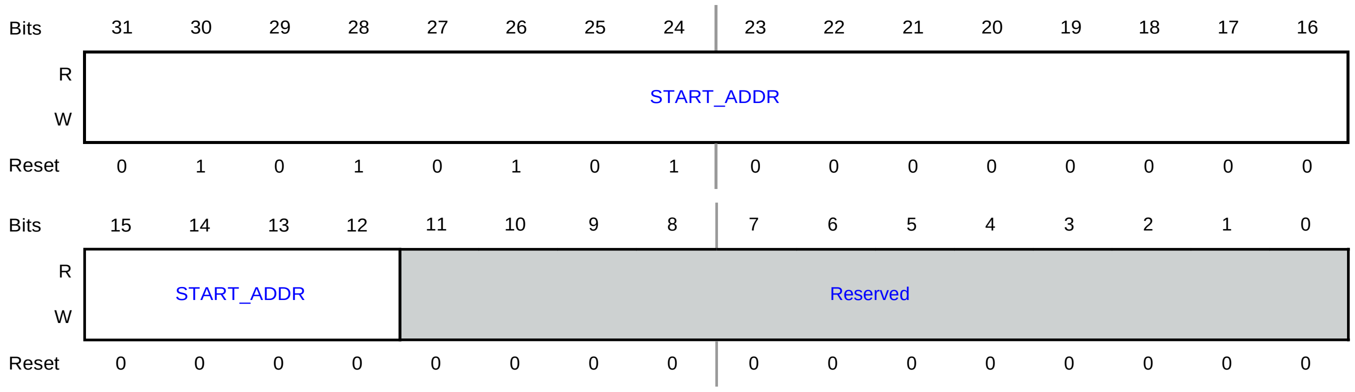
Offset

Register	Offset
LAR0	128h

Function

The LAR registers define base address bits [31:12] of the logical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Diagram



Field	Function
31-12 START_ADDR	Starting Logical Address for Expansion Region 0 The START_ADDR field defines the logical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.8 Physical Address Region 0 (PAR0)

Offset

Register	Offset
PAR0	12Ch

Function
The PAR registers define base address bits [31:12] of the physical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of PAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability. Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Physical Address for Expansion Region 0 The START_ADDR field defines the physical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.9 Region Size 0 (RS0)

Offset

Register	Offset
RS0	130h

Function

Normally there is no need to write the Region Size registers, unless the reset value is not adequate. The reset values for the Region Size registers are defined at design instance using the RDSZ0 to RDSZ7 parameters. Note that the values for the Region Size registers must match the remote chiplet's parameters for region sizes. If they do not, address errors are generated and the transaction dropped. The BLOCK_SIZE field defines the block size of the associated expansion region. This register is only accessible when device parameterization enables AXI master capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											BLOCK_SIZE				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 BLOCK_SIZE	Block Size of the Expansion Address Region 0 BLOCK_SIZE determines the size of the region. Depending upon the setting, as described below, a specific number of lower order bits of LAR and PAR must be set to zero. 00000b - Zero size region 00001b - 4 KB block size. Lower 12 bits of associated START_ADDR must be zero. 00010b - 8 KB block size. Lower 13 bits of associated START_ADDR must be zero. 00011b - 16 KB block size. Lower 14 bits of associated START_ADDR must be zero. 00100b - 32 KB block size. Lower 15 bits of associated START_ADDR must be zero. 00101b - 64 KB block size. Lower 16 bits of associated START_ADDR must be zero. 00110b - 128 KB block size. Lower 17 bits of associated START_ADDR must be zero. 00111b - 256 KB block size. Lower 18 bits of associated START_ADDR must be zero. 01000b - 512 KB block size. Lower 19 bits of associated START_ADDR must be zero. 01001b - 1 MB block size. Lower 20 bits of associated START_ADDR must be zero. 01010b - 2 MB block size. Lower 21 bits of associated START_ADDR must be zero. 01011b - 4 MB block size. Lower 22 bits of associated START_ADDR must be zero. 01100b - 8 MB block size. Lower 23 bits of associated START_ADDR must be zero. 01101b - 16 MB block size. Lower 24 bits of associated START_ADDR must be zero. 01110b - 32 MB block size. Lower 25 bits of associated START_ADDR must be zero. 01111b - 64 MB block size. Lower 26 bits of associated START_ADDR must be zero. 10000b - 128 MB block size. Lower 27 bits of associated START_ADDR must be zero. 10001b - 256 MB block size. Lower 28 bits of associated START_ADDR must be zero. 10010b - 512 MB block size. Lower 29 bits of associated START_ADDR must be zero. 10011b - 1 GB block size. Lower 30 bits of associated START_ADDR must be zero. 10100b - 2 GB block size. Lower 31 bits of associated START_ADDR must be zero.

5.10 Logical Address Region 1 (LAR1)

Offset

Register	Offset
LAR1	134h

Function
The LAR registers define base address bits [31:12] of the logical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	1	0	1	0	1	0	1	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Logical Address for Expansion Region 1 The START_ADDR field defines the logical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.11 Physical Address Region 1 (PAR1)

Offset

Register	Offset
PAR1	138h

Function

The PAR registers define base address bits [31:12] of the physical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of PAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability. Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R / W	START_ADDR															
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R / W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Physical Address for Expansion Region 1 The START_ADDR field defines the physical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.12 Region Size 1 (RS1)

Offset

Register	Offset
RS1	13Ch

Function
Normally there is no need to write the Region Size registers, unless the reset value is not adequate. The reset values for the Region Size registers are defined at design instance using the RDSZ0 to RDSZ7 parameters. Note that the values for the Region Size registers must match the remote chiplet's parameters for region sizes. If they do not, address errors are generated and the transaction dropped. The BLOCK_SIZE field defines the block size of the associated expansion region. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											BLOCK_SIZE				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 BLOCK_SIZE	Block Size of the Expansion Address Region 1 BLOCK_SIZE determines the size of the region. Depending upon the setting, as described below, a specific number of lower order bits of LAR and PAR must be set to zero. 00000b - Zero size region 00001b - 4 KB block size. Lower 12 bits of associated START_ADDR must be zero. 00010b - 8 KB block size. Lower 13 bits of associated START_ADDR must be zero. 00011b - 16 KB block size. Lower 14 bits of associated START_ADDR must be zero. 00100b - 32 KB block size. Lower 15 bits of associated START_ADDR must be zero. 00101b - 64 KB block size. Lower 16 bits of associated START_ADDR must be zero. 00110b - 128 KB block size. Lower 17 bits of associated START_ADDR must be zero. 00111b - 256 KB block size. Lower 18 bits of associated START_ADDR must be zero. 01000b - 512 KB block size. Lower 19 bits of associated START_ADDR must be zero. 01001b - 1 MB block size. Lower 20 bits of associated START_ADDR must be zero. 01010b - 2 MB block size. Lower 21 bits of associated START_ADDR must be zero. 01011b - 4 MB block size. Lower 22 bits of associated START_ADDR must be zero. 01100b - 8 MB block size. Lower 23 bits of associated START_ADDR must be zero. 01101b - 16 MB block size. Lower 24 bits of associated START_ADDR must be zero. 01110b - 32 MB block size. Lower 25 bits of associated START_ADDR must be zero. 01111b - 64 MB block size. Lower 26 bits of associated START_ADDR must be zero. 10000b - 128 MB block size. Lower 27 bits of associated START_ADDR must be zero. 10001b - 256 MB block size. Lower 28 bits of associated START_ADDR must be zero. 10010b - 512 MB block size. Lower 29 bits of associated START_ADDR must be zero. 10011b - 1 GB block size. Lower 30 bits of associated START_ADDR must be zero. 10100b - 2 GB block size. Lower 31 bits of associated START_ADDR must be zero.

5.13 Logical Address Region 2 (LAR2)

Offset

Register	Offset
LAR2	140h

Function

The LAR registers define base address bits [31:12] of the logical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	1	0	1	0	1	0	1	0	1	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Logical Address for Expansion Region 2 The START_ADDR field defines the logical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.14 Physical Address Region 2 (PAR2)

Offset

Register	Offset
PAR2	144h

Function

The PAR registers define base address bits [31:12] of the physical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of PAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability. Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Physical Address for Expansion Region 2 The START_ADDR field defines the physical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.15 Region Size 2 (RS2)

Offset

Register	Offset
RS2	148h

Function
Normally there is no need to write the Region Size registers, unless the reset value is not adequate. The reset values for the Region Size registers are defined at design instance using the RDSZ0 to RDSZ7 parameters. Note that the values for the Region Size registers must match the remote chiplet's parameters for region sizes. If they do not, address errors are generated and the transaction dropped. The BLOCK_SIZE field defines the block size of the associated expansion region. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											BLOCK_SIZE				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 BLOCK_SIZE	Block Size of the Expansion Address Region 2 BLOCK_SIZE determines the size of the region. Depending upon the setting, as described below, a specific number of lower order bits of LAR and PAR must be set to zero. 00000b - Zero size region 00001b - 4 KB block size. Lower 12 bits of associated START_ADDR must be zero. 00010b - 8 KB block size. Lower 13 bits of associated START_ADDR must be zero. 00011b - 16 KB block size. Lower 14 bits of associated START_ADDR must be zero. 00100b - 32 KB block size. Lower 15 bits of associated START_ADDR must be zero. 00101b - 64 KB block size. Lower 16 bits of associated START_ADDR must be zero. 00110b - 128 KB block size. Lower 17 bits of associated START_ADDR must be zero. 00111b - 256 KB block size. Lower 18 bits of associated START_ADDR must be zero. 01000b - 512 KB block size. Lower 19 bits of associated START_ADDR must be zero. 01001b - 1 MB block size. Lower 20 bits of associated START_ADDR must be zero. 01010b - 2 MB block size. Lower 21 bits of associated START_ADDR must be zero. 01011b - 4 MB block size. Lower 22 bits of associated START_ADDR must be zero. 01100b - 8 MB block size. Lower 23 bits of associated START_ADDR must be zero. 01101b - 16 MB block size. Lower 24 bits of associated START_ADDR must be zero. 01110b - 32 MB block size. Lower 25 bits of associated START_ADDR must be zero. 01111b - 64 MB block size. Lower 26 bits of associated START_ADDR must be zero. 10000b - 128 MB block size. Lower 27 bits of associated START_ADDR must be zero. 10001b - 256 MB block size. Lower 28 bits of associated START_ADDR must be zero. 10010b - 512 MB block size. Lower 29 bits of associated START_ADDR must be zero. 10011b - 1 GB block size. Lower 30 bits of associated START_ADDR must be zero. 10100b - 2 GB block size. Lower 31 bits of associated START_ADDR must be zero.

5.16 Logical Address Region 3 (LAR3)

Offset

Register	Offset
LAR3	14Ch

Function
The LAR registers define base address bits [31:12] of the logical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Logical Address for Expansion Region 3 The START_ADDR field defines the logical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.17 Physical Address Region 3 (PAR3)

Offset

Register	Offset
PAR3	150h

Function
The PAR registers define base address bits [31:12] of the physical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of PAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability. Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R / W	START_ADDR															
Reset	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R / W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Physical Address for Expansion Region 3 The START_ADDR field defines the physical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.18 Region Size 3 (RS3)

Offset

Register	Offset
RS3	154h

Function

Normally there is no need to write the Region Size registers, unless the reset value is not adequate. The reset values for the Region Size registers are defined at design instance using the RDSZ0 to RDSZ7 parameters. Note that the values for the Region Size registers must match the remote chiplet's parameters for region sizes. If they do not, address errors are generated and the transaction dropped. The BLOCK_SIZE field defines the block size of the associated expansion region. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											BLOCK_SIZE				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 BLOCK_SIZE	Block Size of the Expansion Address Region 3 BLOCK_SIZE determines the size of the region. Depending upon the setting, as described below, a specific number of lower order bits of LAR and PAR must be set to zero. 00000b - Zero size region 00001b - 4 KB block size. Lower 12 bits of associated START_ADDR must be zero. 00010b - 8 KB block size. Lower 13 bits of associated START_ADDR must be zero. 00011b - 16 KB block size. Lower 14 bits of associated START_ADDR must be zero. 00100b - 32 KB block size. Lower 15 bits of associated START_ADDR must be zero. 00101b - 64 KB block size. Lower 16 bits of associated START_ADDR must be zero. 00110b - 128 KB block size. Lower 17 bits of associated START_ADDR must be zero. 00111b - 256 KB block size. Lower 18 bits of associated START_ADDR must be zero. 01000b - 512 KB block size. Lower 19 bits of associated START_ADDR must be zero. 01001b - 1 MB block size. Lower 20 bits of associated START_ADDR must be zero. 01010b - 2 MB block size. Lower 21 bits of associated START_ADDR must be zero. 01011b - 4 MB block size. Lower 22 bits of associated START_ADDR must be zero. 01100b - 8 MB block size. Lower 23 bits of associated START_ADDR must be zero. 01101b - 16 MB block size. Lower 24 bits of associated START_ADDR must be zero. 01110b - 32 MB block size. Lower 25 bits of associated START_ADDR must be zero. 01111b - 64 MB block size. Lower 26 bits of associated START_ADDR must be zero. 10000b - 128 MB block size. Lower 27 bits of associated START_ADDR must be zero. 10001b - 256 MB block size. Lower 28 bits of associated START_ADDR must be zero. 10010b - 512 MB block size. Lower 29 bits of associated START_ADDR must be zero. 10011b - 1 GB block size. Lower 30 bits of associated START_ADDR must be zero. 10100b - 2 GB block size. Lower 31 bits of associated START_ADDR must be zero.

5.19 Logical Address Region 4 (LAR4)

Offset

Register	Offset
LAR4	158h

Function
The LAR registers define base address bits [31:12] of the logical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	1	0	1	0	1	0	1	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Logical Address for Expansion Region 4 The START_ADDR field defines the logical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.20 Physical Address Region 4 (PAR4)

Offset

Register	Offset
PAR4	15Ch

Function

The PAR registers define base address bits [31:12] of the physical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of PAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability. Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Physical Address for Expansion Region 4 The START_ADDR field defines the physical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.21 Region Size 4 (RS4)

Offset

Register	Offset
RS4	160h

Function
Normally there is no need to write the Region Size registers, unless the reset value is not adequate. The reset values for the Region Size registers are defined at design instance using the RDSZ0 to RDSZ7 parameters. Note that the values for the Region Size registers must match the remote chiplet's parameters for region sizes. If they do not, address errors are generated and the transaction dropped. The BLOCK_SIZE field defines the block size of the associated expansion region. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											BLOCK_SIZE				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 BLOCK_SIZE	Block Size of the Expansion Address Region 4 BLOCK_SIZE determines the size of the region. Depending upon the setting, as described below, a specific number of lower order bits of LAR and PAR must be set to zero. 00000b - Zero size region 00001b - 4 KB block size. Lower 12 bits of associated START_ADDR must be zero. 00010b - 8 KB block size. Lower 13 bits of associated START_ADDR must be zero. 00011b - 16 KB block size. Lower 14 bits of associated START_ADDR must be zero. 00100b - 32 KB block size. Lower 15 bits of associated START_ADDR must be zero. 00101b - 64 KB block size. Lower 16 bits of associated START_ADDR must be zero. 00110b - 128 KB block size. Lower 17 bits of associated START_ADDR must be zero. 00111b - 256 KB block size. Lower 18 bits of associated START_ADDR must be zero. 01000b - 512 KB block size. Lower 19 bits of associated START_ADDR must be zero. 01001b - 1 MB block size. Lower 20 bits of associated START_ADDR must be zero. 01010b - 2 MB block size. Lower 21 bits of associated START_ADDR must be zero. 01011b - 4 MB block size. Lower 22 bits of associated START_ADDR must be zero. 01100b - 8 MB block size. Lower 23 bits of associated START_ADDR must be zero. 01101b - 16 MB block size. Lower 24 bits of associated START_ADDR must be zero. 01110b - 32 MB block size. Lower 25 bits of associated START_ADDR must be zero. 01111b - 64 MB block size. Lower 26 bits of associated START_ADDR must be zero. 10000b - 128 MB block size. Lower 27 bits of associated START_ADDR must be zero. 10001b - 256 MB block size. Lower 28 bits of associated START_ADDR must be zero. 10010b - 512 MB block size. Lower 29 bits of associated START_ADDR must be zero. 10011b - 1 GB block size. Lower 30 bits of associated START_ADDR must be zero. 10100b - 2 GB block size. Lower 31 bits of associated START_ADDR must be zero.

5.22 Logical Address Region 5 (LAR5)

Offset

Register	Offset
LAR5	164h

Function
The LAR registers define base address bits [31:12] of the logical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Logical Address for Expansion Region 5 The START_ADDR field defines the logical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.23 Physical Address Region 5 (PAR5)

Offset

Register	Offset
PAR5	168h

Function

The PAR registers define base address bits [31:12] of the physical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of PAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability. Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Physical Address for Expansion Region 5 The START_ADDR field defines the physical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.24 Region Size 5 (RS5)

Offset

Register	Offset
RS5	16Ch

Function
Normally there is no need to write the Region Size registers, unless the reset value is not adequate. The reset values for the Region Size registers are defined at design instance using the RDSZ0 to RDSZ7 parameters. Note that the values for the Region Size registers must match the remote chiplet's parameters for region sizes. If they do not, address errors are generated and the transaction dropped. The BLOCK_SIZE field defines the block size of the associated expansion region. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											BLOCK_SIZE				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 BLOCK_SIZE	Block Size of the Expansion Address Region 5 BLOCK_SIZE determines the size of the region. Depending upon the setting, as described below, a specific number of lower order bits of LAR and PAR must be set to zero. 00000b - Zero size region 00001b - 4 KB block size. Lower 12 bits of associated START_ADDR must be zero. 00010b - 8 KB block size. Lower 13 bits of associated START_ADDR must be zero. 00011b - 16 KB block size. Lower 14 bits of associated START_ADDR must be zero. 00100b - 32 KB block size. Lower 15 bits of associated START_ADDR must be zero. 00101b - 64 KB block size. Lower 16 bits of associated START_ADDR must be zero. 00110b - 128 KB block size. Lower 17 bits of associated START_ADDR must be zero. 00111b - 256 KB block size. Lower 18 bits of associated START_ADDR must be zero. 01000b - 512 KB block size. Lower 19 bits of associated START_ADDR must be zero. 01001b - 1 MB block size. Lower 20 bits of associated START_ADDR must be zero. 01010b - 2 MB block size. Lower 21 bits of associated START_ADDR must be zero. 01011b - 4 MB block size. Lower 22 bits of associated START_ADDR must be zero. 01100b - 8 MB block size. Lower 23 bits of associated START_ADDR must be zero. 01101b - 16 MB block size. Lower 24 bits of associated START_ADDR must be zero. 01110b - 32 MB block size. Lower 25 bits of associated START_ADDR must be zero. 01111b - 64 MB block size. Lower 26 bits of associated START_ADDR must be zero. 10000b - 128 MB block size. Lower 27 bits of associated START_ADDR must be zero. 10001b - 256 MB block size. Lower 28 bits of associated START_ADDR must be zero. 10010b - 512 MB block size. Lower 29 bits of associated START_ADDR must be zero. 10011b - 1 GB block size. Lower 30 bits of associated START_ADDR must be zero. 10100b - 2 GB block size. Lower 31 bits of associated START_ADDR must be zero.

5.25 Logical Address Region 6 (LAR6)

Offset

Register	Offset
LAR6	170h

Function
The LAR registers define base address bits [31:12] of the logical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	1	0	1	0	1	1	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Logical Address for Expansion Region 6 The START_ADDR field defines the logical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.26 Physical Address Region 6 (PAR6)

Offset

Register	Offset
PAR6	174h

Function

The PAR registers define base address bits [31:12] of the physical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of PAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability. Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Physical Address for Expansion Region 6 The START_ADDR field defines the physical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.27 Region Size 6 (RS6)

Offset

Register	Offset
RS6	178h

Function

Normally there is no need to write the Region Size registers, unless the reset value is not adequate. The reset values for the Region Size registers are defined at design instance using the RDSZ0 to RDSZ7 parameters. Note that the values for the Region Size registers must match the remote chiplet's parameters for region sizes. If they do not, address errors are generated and the transaction dropped. The BLOCK_SIZE field defines the block size of the associated expansion region. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											BLOCK_SIZE				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 BLOCK_SIZE	Block Size of the Expansion Address Region 6 BLOCK_SIZE determines the size of the region. Depending upon the setting, as described below, a specific number of lower order bits of LAR and PAR must be set to zero. 00000b - Zero size region 00001b - 4 KB block size. Lower 12 bits of associated START_ADDR must be zero. 00010b - 8 KB block size. Lower 13 bits of associated START_ADDR must be zero. 00011b - 16 KB block size. Lower 14 bits of associated START_ADDR must be zero. 00100b - 32 KB block size. Lower 15 bits of associated START_ADDR must be zero. 00101b - 64 KB block size. Lower 16 bits of associated START_ADDR must be zero. 00110b - 128 KB block size. Lower 17 bits of associated START_ADDR must be zero. 00111b - 256 KB block size. Lower 18 bits of associated START_ADDR must be zero. 01000b - 512 KB block size. Lower 19 bits of associated START_ADDR must be zero. 01001b - 1 MB block size. Lower 20 bits of associated START_ADDR must be zero. 01010b - 2 MB block size. Lower 21 bits of associated START_ADDR must be zero. 01011b - 4 MB block size. Lower 22 bits of associated START_ADDR must be zero. 01100b - 8 MB block size. Lower 23 bits of associated START_ADDR must be zero. 01101b - 16 MB block size. Lower 24 bits of associated START_ADDR must be zero. 01110b - 32 MB block size. Lower 25 bits of associated START_ADDR must be zero. 01111b - 64 MB block size. Lower 26 bits of associated START_ADDR must be zero. 10000b - 128 MB block size. Lower 27 bits of associated START_ADDR must be zero. 10001b - 256 MB block size. Lower 28 bits of associated START_ADDR must be zero. 10010b - 512 MB block size. Lower 29 bits of associated START_ADDR must be zero. 10011b - 1 GB block size. Lower 30 bits of associated START_ADDR must be zero. 10100b - 2 GB block size. Lower 31 bits of associated START_ADDR must be zero.

5.28 Logical Address Region 7 (LAR7)

Offset

Register	Offset
LAR7	17Ch

Function
The LAR registers define base address bits [31:12] of the logical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R/W	START_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Logical Address for Expansion Region 7 The START_ADDR field defines the logical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.29 Physical Address Region 7 (PAR7)

Offset

Register	Offset
PAR7	180h

Function

The PAR registers define base address bits [31:12] of the physical address of the expansion regions. Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of PAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability. Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR				Reserved											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 START_ADDR	Starting Physical Address for Expansion Region 7 The START_ADDR field defines the physical start address bits [31:12] of the expansion region.
11-0 —	Reserved

5.30 Region Size 7 (RS7)

Offset

Register	Offset
RS7	184h

Function

Normally there is no need to write the Region Size registers, unless the reset value is not adequate. The reset values for the Region Size registers are defined at design instance using the RDSZ0 to RDSZ7 parameters. Note that the values for the Region Size registers must match the remote chiplet's parameters for region sizes. If they do not, address errors are generated and the transaction dropped. The BLOCK_SIZE field defines the block size of the associated expansion region. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved											BLOCK_SIZE				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 BLOCK_SIZE	Block Size of the Expansion Address Region 7 BLOCK_SIZE determines the size of the region. Depending upon the setting, as described below, a specific number of lower order bits of LAR and PAR must be set to zero. 00000b - Zero size region 00001b - 4 KB block size. Lower 12 bits of associated START_ADDR must be zero. 00010b - 8 KB block size. Lower 13 bits of associated START_ADDR must be zero. 00011b - 16 KB block size. Lower 14 bits of associated START_ADDR must be zero. 00100b - 32 KB block size. Lower 15 bits of associated START_ADDR must be zero. 00101b - 64 KB block size. Lower 16 bits of associated START_ADDR must be zero. 00110b - 128 KB block size. Lower 17 bits of associated START_ADDR must be zero. 00111b - 256 KB block size. Lower 18 bits of associated START_ADDR must be zero. 01000b - 512 KB block size. Lower 19 bits of associated START_ADDR must be zero. 01001b - 1 MB block size. Lower 20 bits of associated START_ADDR must be zero. 01010b - 2 MB block size. Lower 21 bits of associated START_ADDR must be zero. 01011b - 4 MB block size. Lower 22 bits of associated START_ADDR must be zero. 01100b - 8 MB block size. Lower 23 bits of associated START_ADDR must be zero. 01101b - 16 MB block size. Lower 24 bits of associated START_ADDR must be zero. 01110b - 32 MB block size. Lower 25 bits of associated START_ADDR must be zero. 01111b - 64 MB block size. Lower 26 bits of associated START_ADDR must be zero. 10000b - 128 MB block size. Lower 27 bits of associated START_ADDR must be zero. 10001b - 256 MB block size. Lower 28 bits of associated START_ADDR must be zero. 10010b - 512 MB block size. Lower 29 bits of associated START_ADDR must be zero. 10011b - 1 GB block size. Lower 30 bits of associated START_ADDR must be zero. 10100b - 2 GB block size. Lower 31 bits of associated START_ADDR must be zero.

5.31 Logical Address Region Upper 0 (LARU0)

Offset

Register	Offset
LARU0	188h

Function

The LARU registers define the upper 32 bits of the base address bits of the logical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Logical Address for Expansion Region 0 The START_ADDR_U field defines the upper 32 bits of the logical start address of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.32 Physical Address Region Upper 0 (PARU0)

Offset

Register	Offset
PARU0	18Ch

Function
The PARU registers define the upper 32 bits of the base address bits of the physical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Physical Address for Expansion Region 0 The START_ADDR_U field defines the physical start address upper bits of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.33 Logical Address Region Upper 1 (LARU1)

Offset

Register	Offset
LARU1	190h

Function
The LARU registers define the upper 32 bits of the base address bits of the logical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Logical Address for Expansion Region 1 The START_ADDR_U field defines the upper 32 bits of the logical start address of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.34 Physical Address Region Upper 1 (PARU1)

Offset

Register	Offset
PARU1	194h

Function
The PARU registers define the upper 32 bits of the base address bits of the physical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Physical Address for Expansion Region 1 The START_ADDR_U field defines the physical start address upper bits of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.35 Logical Address Region Upper 2 (LARU2)

Offset

Register	Offset
LARU2	198h

Function
The LARU registers define the upper 32 bits of the base address bits of the logical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	0	1	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Logical Address for Expansion Region 2 The START_ADDR_U field defines the upper 32 bits of the logical start address of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.36 Physical Address Region Upper 2 (PARU2)

Offset

Register	Offset
PARU2	19Ch

Function
The PARU registers define the upper 32 bits of the base address bits of the physical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	0	1	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Physical Address for Expansion Region 2 The START_ADDR_U field defines the physical start address upper bits of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.37 Logical Address Region Upper 3 (LARU3)

Offset

Register	Offset
LARU3	1A0h

Function

The LARU registers define the upper 32 bits of the base address bits of the logical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Logical Address for Expansion Region 3 The START_ADDR_U field defines the upper 32 bits of the logical start address of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.38 Physical Address Region Upper 3 (PARU3)

Offset

Register	Offset
PARU3	1A4h

Function
The PARU registers define the upper 32 bits of the base address bits of the physical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	0	1	1	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Physical Address for Expansion Region 3 The START_ADDR_U field defines the physical start address upper bits of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.39 Logical Address Region Upper 4 (LARU4)

Offset

Register	Offset
LARU4	1A8h

Function
The LARU registers define the upper 32 bits of the base address bits of the logical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Logical Address for Expansion Region 4 The START_ADDR_U field defines the upper 32 bits of the logical start address of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.40 Physical Address Region Upper 4 (PARU4)

Offset

Register	Offset
PARU4	1ACh

Function

The PARU registers define the upper 32 bits of the base address bits of the physical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	0	1	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Physical Address for Expansion Region 4 The START_ADDR_U field defines the physical start address upper bits of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.41 Logical Address Region Upper 5 (LARU5)

Offset

Register	Offset
LARU5	1B0h

Function

The LARU registers define the upper 32 bits of the base address bits of the logical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Logical Address for Expansion Region 5 The START_ADDR_U field defines the upper 32 bits of the logical start address of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.42 Physical Address Region Upper 5 (PARU5)

Offset

Register	Offset
PARU5	1B4h

Function
The PARU registers define the upper 32 bits of the base address bits of the physical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Physical Address for Expansion Region 5 The START_ADDR_U field defines the physical start address upper bits of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.43 Logical Address Region Upper 6 (LARU6)

Offset

Register	Offset
LARU6	1B8h

Function
The LARU registers define the upper 32 bits of the base address bits of the logical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	1	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Logical Address for Expansion Region 6 The START_ADDR_U field defines the upper 32 bits of the logical start address of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.44 Physical Address Region Upper 6 (PARU6)

Offset

Register	Offset
PARU6	1BCh

Function

The PARU registers define the upper 32 bits of the base address bits of the physical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	1	0	1	0	1	1	0	1	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Physical Address for Expansion Region 6 The START_ADDR_U field defines the physical start address upper bits of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.45 Logical Address Region Upper 7 (LARU7)

Offset

Register	Offset
LARU7	1C0h

Function

The LARU registers define the upper 32 bits of the base address bits of the logical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Depending upon the setting of BLOCK_SIZE, a specific number of lower order bits of LAR must be set to zero. This register is only accessible when device parameterization enables AXI master capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Logical Address for Expansion Region 7 The START_ADDR_U field defines the upper 32 bits of the logical start address of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.46 Physical Address Region Upper 7 (PARU7)

Offset

Register	Offset
PARU7	1C4h

Function

The PARU registers define the upper 32 bits of the base address bits of the physical address of the expansion regions when 64-bit address message types are enabled (i.e., RA64, WA64). Note that the values for the Physical Address Region registers must match the remote chiplet's parameters for physical address regions. If they do not, address errors are generated and the transactions dropped.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	START_ADDR_U															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 START_ADDR_U	Starting Physical Address for Expansion Region 7 The START_ADDR_U field defines the physical start address upper bits of the expansion region when 64-bit address message types are enabled (i.e., RA64, WA64).

5.47 BIST Control and Pattern (BIST_CP)

Offset

Register	Offset
BIST_CP	200h

Function

The BIST_CP register is used to define diPortSD BIST engine configurations or attributes, control operation, and provide an optional programmable test pattern for use by the diPortSD BIST engine.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TESTMODE		0						PTRN_SRC	CONT	ACCMLT	INJ_ERR	STATIC	0		
W			▨	▨	▨	▨	▨	▨						▨	▨	▨
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TST_PTRN							
W	▨	▨	▨	▨	▨	▨	▨	▨								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 TESTMODE	Selects the type of loop back test connection used and starts loop back testing operation of the diPortSD. 00b - BIST engine is disabled. 01b - On-die loop back test mode is enabled. 10b - Die-to-die loop back test mode is configured for other die. 11b - Die-to-die loop back test mode is enabled (in other die this field must be set to 10).
29-24 —	Reserved
23 PTRN_SRC	This field specifies the test pattern source, either the LFSR or TST_PTRN data. 0b - The LFSR generates pseudo-random test data pattern data. 1b - TST_PTRN data is used to generate test data.
22 CONT	When set to 1, BIST operates in a continuous test mode that repeats the test sequence indefinitely. 0b - Stops the test after one full data cycle sequence has been run. 1b - Repeats the test sequence indefinitely.
21 ACCMLT	When set to 1, this field causes all fail information to be accumulated. 0b - Capture first fail information in status registers 1b - Accumulate all fail information.
20 INJ_ERR	This field, when set to 1, forces an error in the BIST test data pattern to verify that the BIST engine properly reports an error. Note that Static Pattern mode overrides INJ_ERR = 1. No errors can be injected when Static Pattern mode (STATIC = 1). The STATIC field value must be 0 for error injection. 0b - No error is injected in the test data pattern 1b - An error is injected using the TST_PTRN data.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 STATIC	This field specifies whether to swizzle the test data from the LFSR or the TST_PTRN field or to replicate and statically drive out the TST_PTRN data without swizzling. Note that to swizzle TST_PTRN data, the PTRN_SRC field value must be set to 1. 0b - Swizzle test data pattern 1b - Replicate and statically drive out TST_PTRN data without swizzling the data
18-8 —	Reserved
7-0 TST_PTRN	This field specifies an alternate programmable test data pattern used when the PTRN_SRC bit is set to 1. The test data pattern is toggled between the non-inverted and inverted pattern in Toggle Pattern mode or continuously driven out in Static Pattern mode.

5.48 BIST Status (BIST_ST)

Offset

Register	Offset
BIST_ST	204h

Function
Provides pass/fail status and the CSI, PCSI, and data valid (DV) bits that indicate errors during BIST loop back testing.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FAIL	DONE	ACTIV E	0					FAILCSI							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FAILCSI								FAILPCSI				FAILDV			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FAIL	Indicates whether the BIST loop back test passed or failed. 0b - Test passed 1b - Test failed.
30 DONE	Indicates whether the BIST loop back test has completed. 0b - BIST test is not complete. 1b - BIST test is complete.
29 ACTIVE	Indicates whether the BIST engine is actively running tests. 0b - BIST engine is not actively running tests. 1b - BIST engine is actively running tests
28-24 —	Reserved
23-8 FAILCSI	Indicates which CSI bits, if any, failed the loop back BIST test. A logic 0 in bit[i] indicates CSI bit[i] failed. A logic 1 in bit[i] indicates CSI bit[i] passed.
7-4 FAILPCSI	Indicates which PCSI bits, if any, failed the loop back BIST test. A logic 0 in bit[i] indicates PCSI bit[i] failed. A logic 1 in bit[i] indicates PCSI bit[i] passed.
3-0 FAILDV	Indicates which data valid bits, if any, failed the loop back BIST test. A logic 0 in bit[i] indicates DV bit[i] failed. A logic 1 in bit[i] indicates DV bit[i] passed.

5.49 BIST Failures (BIST_FLS)

Offset

Register	Offset
BIST_FLS	208h

Function
This register provides a failure count and fail code for the data pattern(s) that failed during BIST loop back testing.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FAILCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								FAILCODE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 FAILCNT	This field provides a cumulative count of the number of failures.
15-8 —	Reserved
7-0 FAILCODE	If an error occurs, this field captures the first failing 8 bit pseudo-random value as an index of when the failure occurred in the sequence.

5.50 BIST Upper Failing Data (BIST_UFD)

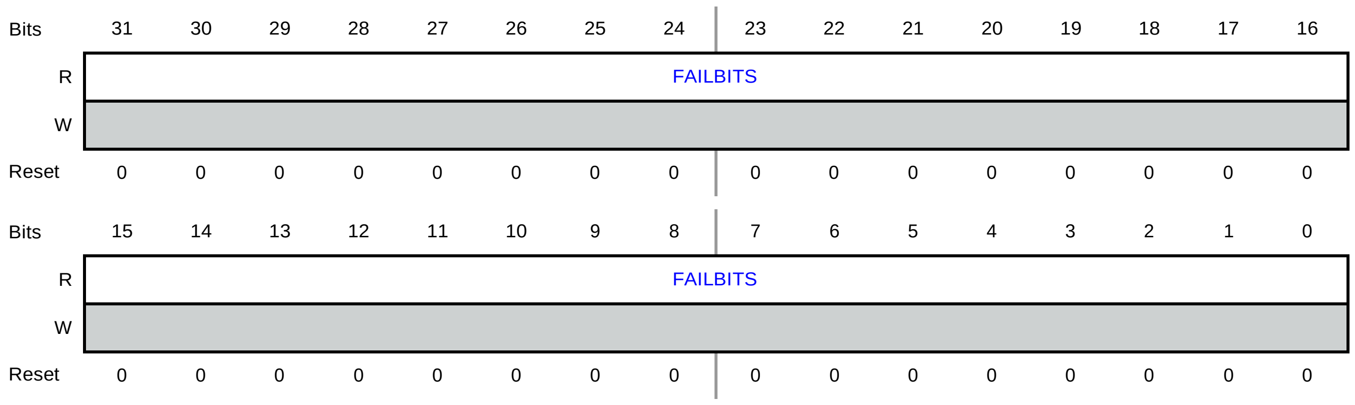
Offset

Register	Offset
BIST_UFD	20Ch

Function

This register provides the upper data bits that failed during BIST loop back testing.

Diagram



Fields

Field	Function
31-0 FAILBITS	[failing_bits[0:31] This field provides an indication of which data bits failed during loop back BIST testing

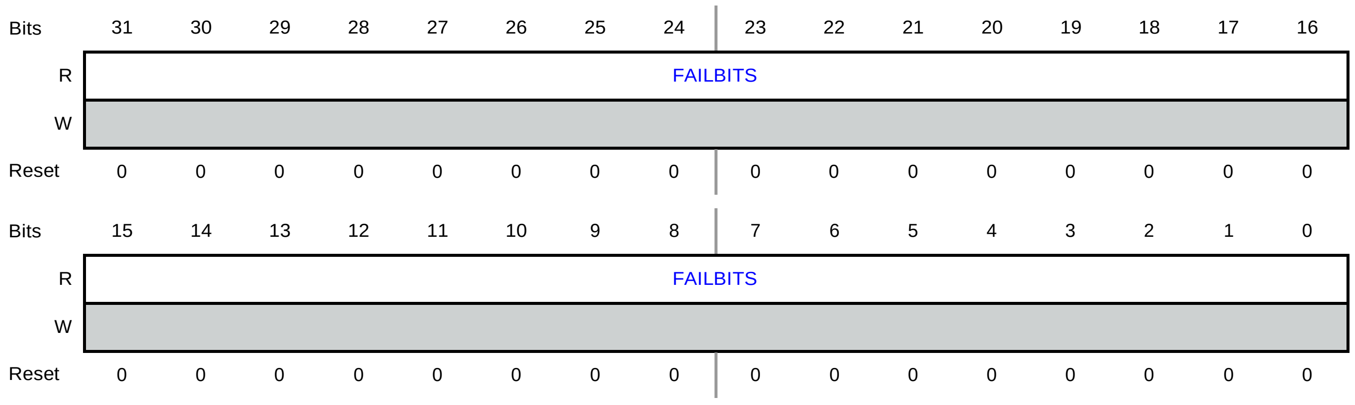
5.51 BIST Lower Failing Data (BIST_LFD)

Offset

Register	Offset
BIST_LFD	210h

Function
This register provides the lower data bits that failed during BIST loop back testing.

Diagram



Fields

Field	Function
31-0 FAILBITS	[failing_bits[32:63] This field provides an indication of which data bits failed during loop back BIST testing

6 Functional Operation

The following sections describe key aspects of diPort functional operation.

6.1 Normal Mode

AXI transactions and signaling of information between each die is fully functional.

When transitioning into Normal mode, customer software running on each of the two interconnected chips may have to provide software configuration steps to initialize diPort features, expecially if the chips are from different companies. Refer to the section Memory Map Integration and Initialization

Also, refer to Normal Operation section for a detailed description of the various aspects of normal mode operation.

6.1.1 Memory Map Integration and Initialization

diPort logic supports 8 address translation regions (depending upon design parameterization). This capability enables integration of a unified memory map across multiple chips. That is, address regions provide for memory mapped regions from remote chip's AXI slave interface to be translated automatically into the local chip's AXI master interface memory mapped regions. If both chips have internal master and slave AXI interfaces connected to diPort logic, then each will have 8 address translation regions. Note that in the context, AXI master interface initiates AXI transactions while AXI slave interface responds accordingly.

Since it is likely that physical addresses on the remote chip conflict with physical addresses on the local chip's physical address, the physical address on the remote chip is translated to non-conflicting logical addresses on the local chip. Each address region is defined by the following registers:

- Logical Address Register (LAR) - defines base address bits [31:12] of the logical address for the AXI master interface. Depending upon the setting of region size, a specific number of lower-order bits of LAR must be set to zero.
- Region Size (RS) - determines the size of the region. Region size options are binary from 4 KB to 2 GB
- Physical Address Register (PAR) - defines base address bits [31:12] of the physical address for the AXI slave interface. Depending upon the setting of region size, a specific number of lower-order bits of PAR must be set to zero.

There are strict constraints regarding the RS and PAR setting for each region. These must match exactly the physical region base address and region size for the AXI slave interface of the remote chip. If they do not, then error detection logic will view this as a message address error, while dropping the erroneous message.

The address translation functions like typical memory address translation regions. That is, if there is a hit to one of the logical regions, it is translated to the associated physical region and sent to the remote chip. If there is no hit, the logical address is passed on to the remote chip without address translation.

A message address error condition is detected when a logical address hit occurs on the local chip but the translated address received on the remote chip does not hit one of the physical address regions defined via design parametrization. Refer to Functional Safety and Error Reporting section for more information.

diPort logic support 32-bit and 64-bit addresses, where only one is enabled at design instance. When 32-bit address are enabled, the RA32 and WA32 messages types are utilized.

When 64-bit address are enabled, the RA64 and WA64 messages types are utilized. Furthermore, the upper 32-bit address for each address region is defined by the following registers:

- Logical Address Register Upper (LARU) - defines the upper 32 bits of the base address bits of the logical address for the AXI master interface.
- Physical Address Register Upper (PARU) - defines the upper 32 bits of the base address bits of the physical address for the AXI slave interface.

NOTE

The current diPort design requires the AXI bus on both chips to have the same data bus size. Initial support is provided for a 64-bit AXI data bus. Subsequently support will be added for a 256-bit AXI data bus and others that are most common.

When transitioning into Normal mode, customer software running on each of the two interconnected chips may have to provide software configuration steps to initialize diPort features, expecially if the chips are from different companies. The following recommendations assume that the high-speed link is already fully functional:

- Initialize LAR registers (and LARU registers if applicable) to define on-chip address regions that require address translation to correctly map into the remote chip. All AXI transations directed to these programmed address regions will be translated based on PAR and RS. Otherwise, no address translation will occur.
- Initialize PAR registers (and PARU registers if applicable) to correlate with the associated physical address regions on the remote chip. If these register values do not correlate with the remote chip memory map, address errors are generated.
- Initialize RS registers to define the region sizes.
- Initialize QoS register to change as needed.
- Initialize EBCFG register to configure the local Elasticity Buffers to match the size for the corresponding Elasticity Buffers on the remote chip.
- If the Signaling feature is used, initialize SIGD register to assure that Signaling messages are limited to 50% or less of the chip-to-chip link bandwidth. For example, if the AXI clock is 1 GHz and the link bandwidth allows for a SIGWR message to be transmitted in 1 ns, then SIGD[SDYC] should be set to 2 or larger.
- If the Signaling feature is used, enable it via SIGEN register.
- Enable interrupts for errors conditions as needed.

After these initialization steps are completed, then AXI messages may be safely transmitted between chips.

6.1.2 Frame Types

There are two basic message frame types used in Normal mode. These are:

- AXI frame type
- Signaling/flow control frame type

The purpose for these two frame types is that diPort is optimized for AXI message throughput. Signaling and flow control messages are provided limited bandwidth, however they may have lower latency.

AXI messages within the AXI frame should be packed back-to-back for as long a sequence as possible. Likewise, packing back-to-back signaling and/or flow control messages in this frame type may also be done.

There are some important differences between each frame type. AXI messages require a larger number of bits to be transmitted. Also, AXI messages require longer (variable) processing time, since AXI bus transactions are generated, and can take many bus clocks to complete. AXI messages received are placed into a substantial AXI Elasticity Buffer (AXI EB), to accommodate peaks in message transmission and delays in returning transactions. Since each of the 5 AXI channels are fully pipeline-able, the AXI EB is beneficial only, however, if AXI masters and slaves also support pipelining.

Signaling and flow control messages are short and require only one bus clock processing time. Signaling and flow control messages received are placed into a minimal Signaling/Flow Control EB (Sig/FC EB) associated with these message types.

Consequently, there are effectively message streams funneled within each of the two frame types. Changing frame types requires one frame type to be ended, and the other frame type to be started. The first message type for a new frame identifies the frame type.

6.1.3 Rules for Messaging

The following paragraphs describe various rules for diPort messaging.

Message creation:

To reduce overall chip-to-chip transaction latency, it is critical to minimize the conversion time when receiving an AXI transaction to the creation of associated Serial diPort messages described in in Messages and Message Fields. Since AXI-related messages reuse AXI fields directly, it is possible to keep the AXI transaction to message conversion time to the minimum number of clocks. Conversely it is possible to keep the conversion time for a message to AXI transaction minimal as well.

Message framing:

A new message frame may contain only messages compatible to the frame type, as indicated by the initial message type. A message frame indicator from the diPort controller:

- May remain asserted for back-to-back AXI messages or Idle messages - ideally AXI messages are packed
- May remain asserted for back-to-back signaling and/or FC messages or Idle messages - ideally signaling and/or FC messages are packed
- Must be negated (i.e., no framed message) and re-asserted to indicate transiting frame types, from signaling and flow control frame type to AXI frame type, and visa verse. This is required for the splitter to be able to easily redirect incoming messages to either the Sig/FC EB or AXI EB based on the first message header bit.
- When frame is negated (i.e., no framed message), all message data must also be negated.

Header and CRC:

Each of the AXI and flow control messages listed in Messages and Message Fields will consist of the following header and CRC fields:

- Header field of 6 bits to define the type of message
- CRC field of 8 or 12 bits to contain CRC information based in message size

Each of the Signaling messages listed in Messages and Message Fields will consist of the following header and CRC fields:

- Header field of 6 bits to define the type of message
- CRC field of 6 bits to contain CRC information

Flow control for AXI messages:

As mentioned previously, flow control messages and message fields are used in Normal mode to prevent an bus master on one chip from over-running an bus slave. For example, if an AXI master on one chip is performing a write burst with 4 64-bit beats, this could potentially over-run the AXI EB in diPort on the remote chip, due primarily to AXI slave delays.

Flow control messages and message fields are used to formulate a debit-credit system for the AXI EB. The AXI EB debit-credit system enables each diPort to be aware of the state of the remote chip's AXI EB status.

The AXI bus also uses 'VALID' and 'READY' signals on each of the 5 channels to determine when a request can and will be accepted. To reduce the number of bits and also handle the delays between the two chips a few rules are needed to comply with the AXI protocol. For example, after acknowledging via Write addr ack message an write address request, it will delay sending the Write data ack message until the data write to the destination has completed.

With unused bits in AXI messages, flow control message fields are included in all AXI message types. The intent is to prevent the need from taking up transmission bandwidth for flow control messages.

The following 3 figures illustrate how flow control works. In the following examples the AXI EB is made to be unreasonably small to more effectively illustrate flow control. Normally the AXI EB would be much larger (i.e., hundreds of bytes) to avoid message transmission pauses as much as possible. Properly sized AXI EB minimizes allows for efficiencies from many outstanding and out of order AXI transactions

In the figures the flow control field EBC will indicate the precise number of double-bytes that have been cleared from the remote chip's AXI EB. That is, values of 0x000 to 0x3FE indicate the number of double bytes that have been cleared (e.g., 001 means 2 bytes cleared). The value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty.

Other normal AXI fields such as BID are also used for flow control purposes, as they indicate when address and data channel transactions related to a specific read ID and write ID are completed.

Table 2. Chip 1 Performing 4 beat burst write to chip 2

Chip 1 Transmission	Chip 2 Transmission	Chip 1 AXI EB Byte Status	Chip 2 AXI EB Byte Status	Other Activity
		16	16	Both chips have full 16 byte credit
Write Address 32		16	6	Chip 1 D2D controller pauses for write address acknowledge due to limited AXI EB size
	Flow Control	16	16	Chip 1 credits 10 bytes for write address acknowledge
Write Data 16 (1st beat)		16	8	Chip 1 debits 8 bytes for 1st beat write
Write Data 16 (2nd beat)		16	0	Chip 1 debits 8 bytes for 2nd beat write - Chip 1 D2D controller pauses for write data acknowledge due to limited AXI EB size
	Flow Control	16	16	Chip 2 completes 1st and 2nd write - Chip 1 credits 16 bytes for write data acknowledge
Write Data 16 (3rd beat)		16	8	Chip 2 performs 3rd write - Chip 1 debits 8 bytes
Write Data 16 (4th beat)		16	0	Chip 2 performs 4th write - Chip 1 debits 8 bytes
	Flow Control	16	16	Chip 2 completes 3rd and 4th write - Chip 1 credits 16 bytes for write data acknowledge

Table 3. Chip 1 Performing 4 beat burst read to chip 2

Chip 1 Transmission	Chip 2 Transmission	Chip 1 AXI EB Byte Status	Chip 2 AXI EB Byte Status	Other Activity
		16	16	Both chips have full 16 byte credit
Read Address 32		16	6	Chip 1 D2D controller pauses for 1st read beat due to limited AXI EB size
	Read Data 16 (1st beat)	8	16	Chip 2 debits 10 bytes for 1st read beat
	Read Data 16 (2nd beat)	0	16	Chip 2 debits 8 bytes for 2nd read beat - Chip 2 D2D controller pauses for read address ack due to limited AXI EB size

Table continues on the next page...

Table 3. Chip 1 Performing 4 beat burst read to chip 2 (continued)

Chip 1 Transmission	Chip 2 Transmission	Chip 1 AXI EB Byte Status	Chip 2 AXI EB Byte Status	Other Activity
Flow Control		16	16	Chip 2 credits 16 bytes for read data acknowledge
	Read Data 16 (3rd beat)	8	16	Chip 1 debits 8 bytes for read data (3rd beat)
	Read Data 16 (4th beat)	0	16	Chip 1 debits 8 bytes for read data (4th beat)
Flow Control		16	16	Chip 2 credits 16 bytes for read data acknowledge

Table 4. Both chips performing 4 beat burst write

Chip 1 Transmission	Chip 2 Transmission	Chip 1 AXI EB Byte Status	Chip 2 AXI EB Byte Status	Other Activity
		16	16	Both chips have full 16 byte credit
Write Address 32	Write Address 32	6	6	Both chips D2D controllers pause for write address acknowledge due to limited AXI EB size
Flow Control Write Addr Ack	Flow Control Write Addr Ack	16	16	Both chips credit 10 bytes for write address acknowledge
Write Data 16 (1st beat)	Write Data 16 (1st beat)	8	8	Both chips debit 8 bytes for 1st beat write
Write Data 16 (2nd beat)	Write Data 16 (2nd beat)	0	0	Both chips debit 8 bytes for 2nd beat write - Both chips stall waiting for write data ack
Flow Control Write Data Ack	Flow Control Write Data Ack	16	16	Both chips complete 1st and 2nd write - Both chips credit 16 bytes for write data ack
Write Data 16 (3rd beat)	Write Data 16 (3rd beat)	8	8	Both chips perform 3rd write - Both chips debit 8 bytes
Write Data 16 (4th beat)	Write Data 16 (4th beat)	0	0	Both chips perform 4th write - Both chips debit 8 bytes
Flow Control Write Data Ack	Flow Control Write Data Ack	16	16	Both chips complete 3rd and 4th write - Both chips credit 16 bytes

Rules for transmitting and receiving signaling and flow control messages:

To prevent signaling and flow control messages from interfering with through-put and latency of AXI messages, there are some special rules for transmission of these message types. Furthermore, to not consume any space in AXI EB buffers, upon reception of these messages they must be split off into a Sig/FC EB. Therefore upon reception of messages all AXI messages are directed into the AXI EB while all signaling and flow control messages are directed to the Sig/FC EB.

Hence signaling and flow control messages do not impact in any way the debit/credit system employed for AXI messages, since they are stored in a separate Sig/FC EB with designated rules to prevent over-running it. For simplicity in preventing over-running the Sig/FC EB the following transmission rules are considered:

- Signaling and FC messages only require one AXI bus clock on the destination chip for processing (unlike AXI messages which require many). But since the source chip may have a higher AXI bus frequency, Signaling and FC messages must be limited.
- When AXI messages are competing with Signaling and FC messages for bus bandwidth, the QoS logic will assure that bandwidth is limited to the QoS settings. But specifically when there are no competing AXI messages, signaling messages must be limited so as to not overrun the destination chip's Sig/FC EB.
- At design instantiation, use the SIGD parameter to restrain the maximum number of back-to-back signaling messages. The value of this parameter must be set to assure that when there is no AXI traffic, the signal message bandwidth is limited to below 50%. The parameter sets the minimum number of AXI clock that must expire before another signal message can be generated. The default is 5 AXI clocks but may be reduced with detailed investigation for chip with slower AXI clock frequencies. Additionally, this parameter may be change by software by writing to the SIGD register.

As mentioned upon receipt of signaling and flow control messages, logic must process these messages in one AXI bus clock. That is, the logic must be able to continuously accept back-to-back signaling and FC messages each AXI bus clock.

6.1.4 Normal Operation

The following sections describe how the various blocks operate in Normal mode.

6.1.4.1 Link Layer - Overview

As mentioned earlier, the diPort controller may be integrated directly with a PHY under certain conditions or integrated with another link layer. The diPort controller currently implements link layer functions, described in sub-sections following, intended to be used with a low error rate PHY technology. A future release of this specification will incorporate FEC within the diPort link layer.

Referring back to Block Diagram the LL provides the following functions:

- Translate between logical or message data and physical data
- Control outputs to PHY or link layer supporting retry
- Control starting and ending message frames
- Provide CRC generation and checking functions
- Perform a splitting function for incoming messages into AXI and Sig/FC EBs

6.1.4.2 Link Layer - Outgoing Messages

For outgoing messages, LL receives coherent messages with framing information from the TX manager. That is, on any AXI bus clock the TX Manager presents all message fields coherently along with frame start and end, and valid signals per double byte. If there are no pending outgoing messages then all valid signals are negated or idle messages are transmitted.

The LL may accept all message fields coherently along with framing information into the TX EB if there is enough available bytes to store the complete message. If all message fields coherently along with framing information does not fit into the available TX EB space, the LL must not accept it until all message fields can be simultaneously stored into the TX EB.

The LL does not synchronize outgoing message fields from the TX Manager, received in the AXI bus clock domain. Rather the PHY (or link layer supporting retry) will provide the critical function of synchronizing output message fields from the LL to the PHY clock domain. The AXI bus clock and PHY clock domain are asynchronous clock domains.

If the TX Manager provides valid messages of the same frame type at a fast enough pace, the LL should pack messages back-to-back while keeping the frame open for all of these messages. This is the best situation for achieving maximum bandwidth. But if the TX manager signals a new frame is needed, the LL should end the current frame and start a new frame. This would occur, for example, when there are no more valid AXI messages, and the TX Manager is starting a new frame for signaling or flow control messages.

When a frame end is included with a message, the LL must indicate the frame end.

6.1.4.3 Link Layer - Incoming Messages

For incoming messages the LL performs a splitting function to fill the Sig/FC EB with signaling and FC messages, and to fill the AXI EB with AXI messages. The LL does not synchronized incoming messages from the PHY (or link layer supporting retry). The PHY (or link layer supporting retry) provides the critical function of synchronizing input message fields to the AXI clock domain. The AXI bus clock and PHY clock domain are asynchronous clock domains.

6.1.4.4 Link Layer - Elasticity Buffers

The data size of the AXI EB will be N x 2 bytes where N is implementation dependent. A minimum for N should allow for at least for the largest AXI message size plus some additional space. A larger N value will enable markedly better performance capability.

The data size of the Sig/FC EB will be M x 2 bytes where M is implementation dependent. A practical setting for M will be dependent on the number of Signaling Virtual Registers utilized as well as Signaling configuration such as synchronization parameterization.

From incoming message data the LL performs the splitting function to form two message streams: AXI and signaling/flow control. For a new frame, the following checking should be performed:

- Check bit [0:1] of MID field in first message of new frame
- If 11b then all messages of current frame are directed to Sig/FC EB
- If 0Xb then all messages of current frame are directed to AXI EB

6.1.4.5 TX Manager

To accommodate two chips that may be operating operating at vastly different AXI bus clock rates, the TX manager must pass all double-bytes of a new message to the TX EB simultaneously to assure coherency. Additionally the PHY (or link layer supporting retry) must continuously transmit all back-to-back bytes received from the TX EB.

For outgoing messages, the TX Manager submits an even number of bytes (e.g., up to 14 currently but it depends on data size) to the TX EB. Message coherency must be maintained so no message may be split across two submissions to the LL. Multiple messages, however, may be submitted in one submission as long as the following conditions are met:

- All messages completely fit in the same submission
- All messages are of the same frame type (e.g., multiple AXI messages)

6.1.4.6 RX Manager

The RX Manager receives incoming AXI message data from the AXI EB. After receiving an incoming message the RX Manager then directs message information to either the master or slave AXI interface, assuming both are enabled. The following criteria is used to direct the incoming message information:

- Messages pertaining to AXI read address, write address and write data channels are directed to the master interface
- Messages pertaining to AXI read data and write response channels are directed to the slave interface

The RX Manager must also re-construct messages from the message data byte-pairs received from the LL. This process is accomplished by starting at the beginning of a frame. The MID field of the first message in the frame is decoded to determine how many byte-pairs are associated with the first message. The first message is then converted into the appropriate AXI channel transaction. This process is repeated until all message byte-pairs are converted into the appropriate AXI channel transactions.

If there is a framing error detected, such as an incorrect number of byte-pairs received for a specific message as indicated by the MID field, then a framing error should be generated by the RX manager.

6.1.4.7 Sig/FC Manager

The Sig/FC Manager receives incoming signal input message data from the Sig/FC EB. After receiving an incoming message the Sig/FC Manager then directs message information to either the Signaling Interface associated with chip input or output signals, or AXI flow control logic.

Regarding signaling message data the following criteria is used to direct the incoming message information:

- Signal Write messages are directed to the Signal Interface to write the new signal information received from the remote chip. The SWID field determines the destination logical register.

Similarly, the Sig/FC Manager must reconstruct messages from the message data byte-pairs received from the LL, and perform checking for framing errors.

6.1.5 Functional Safety and Error Reporting

Above all else diPort must provide a safe and robust ecosystem for two or more chips to operate in a SiP. Primary is to assure there are no systematic faults. Systematic faults impact each instance on a device, e.g., EMI, non-robust design, design bugs, problems at process or temperature corners. Systematic defects must be avoided completely by applying the necessary diligence in specification, development, verification, validation, qualification, and production test.

Furthermore, random hardware faults are expected to happen in-field over the lifetime of one of more devices. A distinction is made between transient faults (radiation induced single event upset usually causing no physical damage and can be "removed") and permanent faults (wear and aging, physical damage which cannot be "removed"). The strategy to address random faults when they occur, is to identify how random faults are mapped into diPort failure modes and measures.

Below is a list of various types of error conditions that can detect random hardware faults:

- CRC message error
- Message frame error
- Message ID error
- Message address error
- Message response error
- Message processing error

Furthermore, a good practice is to implement robust error signals between chips, to cover if per chance a random hardware fault may cause the diPort interface to be inoperable. In this case implementing common signals between, such as ERROR and ERROR_B, can indicate if serious system failure occurs.

The following paragraphs describe error detection for the various types of unexpected or error conditions that can potentially occur in a SiP.

CRC error during normal mode

As previously mentioned, each message has CRC protection. If a CRC error is detected by either chip the status is captured in the ESR[NCRC_ERR] bit, which can trigger assertion of an error signal. A CRC error may be generated by a random fault affecting chip-to-chip signals as well as internal logic. The offending message is dropped.

Message frame error

diPort logic checks for framing errors during Normal mode. A framing error occurs if the message form is not correct due to an incomplete number of message bytes received. If a frame error is detected by either chip the status is captured in the ESR[MFRM_ERR] bit, which can trigger assertion of an error signal. The offending message is dropped.

Message ID error

diPort logic checks for message ID errors during Normal mode. This pertains to the MID field in each message that identifies the message type and form. A message ID error occurs if the message ID is not registered as a valid ID. If a message ID error is detected by either chip the status is captured in the ESR[MID_ERR] bit, which can trigger assertion of an error signal.

The offending message is dropped, however, due to corrupted message ID information an incorrect number of double bytes may be dropped. Thus, this is a serious error likely requiring a reset to be asserted for recovery of diPort IP.

Message address error

diPort logic checks for message address errors during Normal mode. This pertains to the ARADDR and AWADDR fields in read and write address messages. These fields are checked at the destination chip to assure they reside within the physical address space defined by instance parameters. Specifically, a message address error is detected when a logical address is translated on the source chip but the translated address received on the destination chip does not match one of its physical address regions. If a message address error is detected by either chip the status is captured in the ESR[MADD_ERR] bit, which can trigger assertion of an error signal. The offending message is dropped.

Message processing error

diPort logic checks for message processing errors during Normal mode. A message processing error occurs if the message anywhere along the data path from the physical to logic there is a FIFO overrun. If a message processing error is detected by either chip the status is captured in the ESR[MPRC_ERR] bit, which can trigger assertion of an error signal. A message processing error is not expected during normal operation, however, it may be generated by a random fault affecting internal logic.

Message response error

A response error may occur for a variety of reasons:

- Remote chip not responding
- Dropped message

If due to a random hardware fault the remote chip stops responding to messages (i.e., not sending back anything), this error condition will be detected. If either condition is detected and the status is captured in the ESR[RSP_ERR] bit, which can trigger assertion of an error signal.

6.1.6 Resiliency to Error Conditions

Providing comprehensive resiliency is a huge challenge simply because of the myriads of possible random hardware faults. Given that, diPort provides resiliency logic to attempt to handle a large percentage (though not all) of possible random hardware faults.

Strategy for resiliency will be discussed below with regard to AXI messages and Signaling and Flow Control messages.

AXI messages

The following table illustrates the strategy for random hardware faults that manifest themselves as Message CRC, Address, and Frame errors. The term destination refers to the chip that receives the message and detects an error.

Table 5. Resiliency strategy for message CRC, address, and frame errors

Result of Random Hardware Fault	Read Address Channel (AR)	Write Address Channel (AW)	Read Data Channel (R)	Write Data Channel (W)	Response Channel (B)
Message Address error	Destination diPort IP drops erroneous AR message and drops subsequent related R channel messages. When the data read is not returned this will cause the source diPort IP to generate a response error.	Destination diPort IP drops erroneous AW message and drops subsequent related W channel messages. When the data write response is not returned this will cause the source diPort IP to generate a response error.	N/A	N/A	N/A

Table continues on the next page...

Table 5. Resiliency strategy for message CRC, address, and frame errors (continued)

Result of Random Hardware Fault	**Read Address Channel (AR)**	**Write Address Channel (AW)**	**Read Data Channel (R)**	**Write Data Channel (W)**	**Response Channel (B)**
Message Frame error	Destination diPort IP drops erroneous AR message and drops subsequent related R channel messages. When the data read is not returned this will cause the source diPort IP to generate a response error.	Destination diPort IP drops erroneous AW message and drops subsequent related W channel messages. When the data write response is not returned this will cause the source diPort IP to generate a response error.	Destination diPort IP drops erroneous R message. When the data read is not returned this will cause the source diPort IP to generate a response error.	Destination diPort IP drops erroneous W message. When the data write response is not returned this will cause the source diPort IP to generate a response error.	Destination diPort IP drops erroneous B message. When the data write response is not returned this will cause the source diPort IP to generate a response error.
Message CRC error	Destination diPort IP drops erroneous AR message and drops subsequent related R channel messages. When the data read is not returned this will cause the source diPort IP to generate a response error.	Destination diPort IP drops erroneous AW message and drops subsequent related W channel messages. When the data write response is not returned this will cause the source diPort IP to generate a response error.	If message MID and RID fields are valid, destination diPort IP completes erroneous AXI read transaction with an error indication (RRESP=2). If not valid, destination diPort IP drops erroneous R message. When the data read is not returned this will cause the source diPort IP to generate a response error.	If message MID field is valid, destination diPort IP completes erroneous AXI write transaction with no actual write occurring (WSTRB=0). If not valid, destination diPort IP drops erroneous W message. When the data write response is not returned this will cause the source diPort IP to generate a response error.	If message MID and BID fields are valid, destination diPort IP completes erroneous AXI response transaction with an error indication (BRESP=2). If not valid, destination diPort IP drops erroneous B message. When the data write response is not returned this will cause the source diPort IP to generate a response error.

When random hardware faults manifest themselves as Message ID errors, this is a very challenging area for resiliency success. The problem is when the message ID field (6 bits) is transformed by a random hardware fault into a different but valid message ID, or to an invalid ID. The challenge lies in how to cleanly drop the unidentified message. Given this information, diPort provides resiliency logic to attempt to handle Message ID errors in a similar manner as Message Frame errors.

Signaling and flow control messages

When random hardware faults manifest themselves as Message Frame and Message CRC errors in a Signaling or Flow Control message, the offending message is dropped. This will allow for a reasonable level of resiliency for these types of messages. With regard to Message ID errors in a Signaling or Flow Control message, the same challenges are present as described above.

6.2 Signaling

As explained the objectives for the signaling feature are:

- Critical internal signals (i.e., hundreds) are automatically shared across chips using standard conventions
- Only supported signal protocol is level-sensitive

- Signaling information shared between chips may include level-sensitive signals such as interrupt requests, safety information, system information, triggers, and debug information.
- DMA hand-shake signaling is not supported by diPort

And over-view of signaling implementation and features are:

- 512 actual signals transmitted in each direction, organized into 16 virtual signaling registers of 32 signals.
- Quality of Service (QoS) controls to assure signaling messages do not exceed specified bandwidth settings
- No native support for triple redundancy. Triple-redundant logic may be implemented outside diPort if needed.

For local signals that are replicated on the remote chip, the local chip's diPort controller has the following interface:

- Signaling inputs - 32 inputs x SR were SR is the number of virtual signaling registers (SREGS parameter) x 32
- — Local discrete signals or buses that are to be replicated on the remote chip
- Signaling trigger - 1 input x SREGS
- — Used for low jitter QoS setting and vector synchronization

For remote signals that are replicated on the local chip, the local chip's diPort controller has the following interface:

- Signaling outputs - 32 outputs x SR were SR is the number of virtual signaling registers (SREGS parameter) x 32
- — Remote discrete signals or buses that are to be replicated on the local chip
- Signaling clocks - 1 input x SREGS
- — Used for synchronization of signaling outputs
- Signaling resets - 1 input x SREGS
- — Use for clearing state of signaling outputs

If the signaling feature is required, it must be integrated to interconnect module to module signals that happen to span across chips. And to enable signaling, refer to the SIGEN register definition. If the signaling feature is not required, it can be left disabled.

When signaling is enabled via SIGEN register, an initialization is performed to replicate initial signal states on the other chip.

6.2.1 QoS Controls

As related to diPort signaling, QoS refers to the ability to provide different priority characteristics to each signaling logical register. These priority characteristics, with a default setting configured at tape-out, include the ability to configure each logical register to have one of these specific QoS settings:

- Lowest Latency QoS (LLQoS)
- Lowest Jitter QoS (LJQoS)

In addition to specific QoS settings, there is also a general priority characteristic

- Signaling QoS (SQoS)

The SQoS setting, with a default setting configured at tape-out, allocates a percentage of bandwidth to Signaling messages versus AXI messages. Settings are:

- 50% bandwidth to both AXI and Signaling messages (1 to 1 ratio)
- 75% bandwidth to AXI messages and 25% bandwidth to Signaling messages (4 to 1 ratio)
- 87.5% bandwidth to AXI messages and 12.5% bandwidth to Signaling messages (8 to 1 ratio)
- 100% bandwidth to AXI messages. Signaling messages may only be transmitted one at a time when there are no pending AXI messages

Note that when there are no AXI messages competing with signaling messages, signaling messages are allowed to consume unused diPort bus bandwidth. As explained in Rules for Messaging signaling messages bandwidth must be limited to less than 50% using design parametrization.

For LLQoS configuration, any signal change results in a high priority Signal Write message being queued for transmission. The Signal Write message will be transmitted as quickly as possible, given that the SQoS is maintained, so as not to over use allotted Signaling bandwidth.

For LJQoS configuration, an external trigger (e.g., periodic timer) will result in a high priority Signal Write message being queued for transmission. The Signal Write message will be transmitted as quickly as possible, given that the SQoS is maintained, so as not to over use allotted Signaling bandwidth.

Some signaling requirements that must be met, using either these QoS configurations or other methods, include:

- To prevent spurious interrupts the QoS or another method ensure the cleared source interrupt bit is signaled to the interrupt controller input prior to the exit of the associated interrupt handler.

6.2.2 CDC Synchronization Scheme Supported

The signaling logic natively supports only scalar CDC synchronization from the source clock domain to the destination clock domain. A scalar synchronization scheme implies that each of the signals transported from a source domain on one chip to a destination domain on remote chip, are scalars and therefore not recombined in the destination clock domain logic (e.g., not used together in a state machine).

To support scalar synchronization, diPort logic optionally synchronizes incoming signals to the AXI clock domain, transports them to the remote chip, then optionally synchronizes them to the destination domain, although only on a virtual register basis only. Design parameters SIG_IN_SYNC and SIG_OUT_SYNC are used to enable input and output synchronization. The default for these parameters is to enable both input and output parametrization.

Regarding vector CDC synchronization, a variety of common CDC methods may be added external to the diPort logic to assure synchronization. These methods include qualifier-based synchronization for vectors, as well as gray coded vector.

6.2.3 Frequency Limitations for Signals

The requirement for signals that are transported between chips via the diPort is that they are quasi-static or low frequency. What is classified as "low frequency" is dependent upon the following factors:

- SQoS setting determines what is the percentage of diPort bus bandwidth allocated to signaling when competing against AXI messages. When there are no AXI messages competing with signaling messages, signaling messages are required to consume less than 50% of unused diPort bus bandwidth.
- The number of signals that are transported in one direction
- The composite frequency of all signals that are transported in one direction
- Signals with the maximum frequency and minimum time between edges
- Frequency of the diPort chip-to-chip PHY bit clock
- Number of PHY lanes

6.3 Messages and Message Fields

The following sections describe diPort messages and message fields.

The objectives for the messages are:

- In Normal mode provide for the most efficient packetization of the AXI protocol, to balance achieving the lowest latency and highest bandwidth
- Provide for flow control in Normal mode to prevent a master from over-running a slave

To meet Normal mode objectives and minimize packetization delays, the minimum number of AXI channel fields are directly loaded into message fields without any manipulation.

For help in understanding tables below, definitions of columns are as follows:

- Message ID - 6-bit field to identify message type
- Message Size (Bytes) - Even number of bytes length the framed message requires.

The Message ID field encodings are partitioned into the following categories:

- Encoding 0 is idle
- Encodings 1 to 47 are for AXI messages or reserved
- Encodings 48 to 63 are for flow control and signaling or reserved

6.3.1 diPort Messages for Signaling

The table below lists the one Signaling message and associated fields.

Table 6. diPortSD message for Signaling

Message Name (Acronym)	Message ID (6 bits)	Message Size (Bytes)
Signal write (SIGWR)	0x38	6

Signaling Message

The following section describes the Signaling message and message fields.

Signal write

Table 7. Signal write message

Field Name	Field Size	Description
MID	6	Message ID
SWID	4	Signal write ID identifies one of 16 virtual registers for signal information
SWDATA	32	32-bits of signal information to be written to the identified signal storage register
CRC	6	CRC information
	48	Total bits

Signal Write messages are transmitted to export the state of key internal signals on a chiplet. Signal information that is shared between chiplets is organized into 16 virtual registers that may be written as a result of this message. Examples of shared signals between chiplets include interrupt requests, mode of operation, etc.

There are various triggers that may be used to determine when Signal Write messages are transmitted. For example, when any bit in one of the virtual registers changes. Or conversely, periodically in a round robin fashion. For more information, refer to Signaling section.

6.3.2 diPort Messages for AXI including Flow Control

The table below provides information about AXI messages.

Table 8. diPortSD messages for AXI including flow control

Message Name	Message ID (6 bits)	Message Size (Bytes)
Idle	0x00	2

Table continues on the next page...

Table 8. diPortSD messages for AXI including flow control (continued)

Message Name	Message ID (6 bits)	Message Size (Bytes)
Read Address 32 (RA32)	0x01	14
Read Address 64 (RA64)	0x02	20
Read Data 8 (RA8)	0x03	8
Read Data 16 (RA16)	0x04	8
Read Data 32 (RA32)	0x05	10
Read Data 64 (RA64)	0x06	14
Read Data 128 (RA128)	0x07	22
Read Data 256 (RA256)	0x08	38
Read Data 512 (RA512)	0x09	70
Read Data 1024 (RA1024)	0x0a	134
Write Address 32 (WA32)	0x11	14
Write Address 64 (WA64)	0x12	20
Write Data 8 bits (WD8)	0x13	8
Write Data 16 bits (WD16)	0x14	8
Write Data 32 bits (WD32)	0x15	10
Write Data 64 bits (WD64)	0x16	14
Write Data 128 bits (WD128)	0x17	24
Write Data 256 bits (WD256)	0x18	42
Write Data 512 bits (WD512)	0x19	78
Write Data 1024 bits (WD1024)	0x1a	150
Write Data Response (WDR)	0x1b	4
Flow Control (FC)	0x30	4

AXI and flow control messages

The following sections describe AXI messages and message fields. The full definition for all AXI related fields are defined in the AXI version 3.

RA32

Table 9. AXI read address 32 message

Field Name	Field Size	Description
MID	6	Message ID

Table continues on the next page...

Field Name	Field Size	Description
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
ARID	8	Identification tag for a read transaction
ARADDR	32	The address of the first transfer in a read transaction
ARSIZE	3	Size, the number of bytes in each data transfer in a read transaction
ARLEN	8	Length, the exact number of data transfers in a read transaction
ARBURST	2	Burst type, indicates how address changes between each transfer in a read transaction
ARLOCK	1	Provides information about the atomic characteristics of a read transaction
ARCACHE	4	Indicates how a read transaction is required to progress through a system
ARPROT	3	Protection attributes of a read transaction: privilege, security level, and access type
ARQOS	4	Quality of Service identifier for a read transaction
ARREGION	4	Region indicator for a read transaction
ARUSER	8	User-defined extension for the read address channel
Reserved	3	Reserved for ARSNOOP to indicate the transaction type for shareable read transactions (ACE-Lite)
Reserved	2	Reserved for ARDOMAIN to indicate the shareability domain of a read transaction (ACE-Lite)
Reserved	2	Reserved for ARBAR to indicate a transaction is a read barrier domain (ACE-Lite)
REGION_HIT	1	Logical address hit in region
Reserved	3	
CRC	8	Cyclic redundancy code

Table continues on the next page...

Table 9. AXI read address 32 message (continued)

Field Name	Field Size	Description
	112 (14 bytes)	Total implemented bits

The AXI Read Address 32 message is the initial message transmitted to initiate a read transaction. The ARUSER field is an extension to the message, which allows for secure access extensions.

RA64

Table 10. AXI read address 64 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
ARID	8	Identification tag for a read transaction
ARADDR	64	The address of the first transfer in a read transaction
ARSIZE	3	Size, the number of bytes in each data transfer in a read transaction
ARLEN	8	Length, the exact number of data transfers in a read transaction
ARBURST	2	Burst type, indicates how address changes between each transfer in a read transaction
ARLOCK	1	Provides information about the atomic characteristics of a read transaction
ARCACHE	4	Indicates how a read transaction is required to progress through a system
ARPROT	3	Protection attributes of a read transaction: privilege, security level, and access type
ARQOS	4	Quality of Service identifier for a read transaction
ARREGION	4	Region indicator for a read transaction
ARUSER	8	User-defined extension for the read address channel
Reserved	3	Reserved for ARSNOOP (ACE-Lite)

Table continues on the next page...

Table 10. AXI read address 64 message (continued)

Field Name	Field Size	Description
Reserved	2	Reserved for ARDOMAIN (ACE-Lite)
Reserved	2	Reserved for ARBAR (ACE-Lite)
REGION_HIT	1	Logical address hit in region
Reserved	15	
CRC	12	Cyclic redundancy code
	160 (20 bytes)	Total implemented bits

The AXI Read Address 64 message is the initial message transmitted to initiate a read transaction. The ARUSER field is an extension to the message, which allows for secure access extensions.

RD8

Table 11. AXI read data 8 Message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
RID	8	Read address ID
SHIFT	7	Lane shift
RDATA	8	Read data
RRESP	2	Read response
RLAST	1	Read last
Reserved	14	
CRC	8	Cyclic redundancy code
	64 (8 bytes)	Total implemented bits

AXI Read Data 8 messages are transmitted to provide read data of size 8 associated with a Read Address message.

RD16

Table 12. AXI read data 16 message

Field Name	Field Size	Description
MID	6	Message ID

Table continues on the next page...

Field Name	Field Size	Description
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
RID	8	Read address ID
SHIFT	6	Lane shift
RDATA	16	Read data
RRESP	2	Read response
RLAST	1	Read last
Reserved	7	
CRC	8	Cyclic redundancy code
	64 (8 bytes)	Total implemented bits

AXI Read Data 16 messages are transmitted to provide read data of size 16 associated with a Read Address message.

RD32

Table 13. AXI read data 32 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
RID	8	Read address ID
SHIFT	5	Lane shift
RDATA	32	Read data
RRESP	2	Read response
RLAST	1	Read last
Reserved	8	
CRC	8	Cyclic redundancy code
	80 (10 bytes)	Total implemented bits

AXI Read Data 32 messages are transmitted to provide read data of size 32 associated with a Read Address message.

RD64

Table 14. AXI read data 64 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
RID	8	Read address ID
SHIFT	4	Lane shift
RDATA	64	Read data
RRESP	2	Read response
RLAST	1	Read last
Reserved	9	
CRC	8	Cyclic redundancy code
	112 (14 bytes)	Total implemented bits

AXI Read Data 64 messages are transmitted to provide read data of size 64 associated with a Read Address message.

RD128

Table 15. AXI read data 128 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
RID	8	Read address ID
SHIFT	3	Lane shift
RDATA	128	Read data
RRESP	2	Read response
RLAST	1	Read last
Reserved	6	

Table continues on the next page...

Table 15. AXI read data 128 message (continued)

Field Name	Field Size	Description
CRC	12	Cyclic redundancy code
	176 (22 bytes)	Total implemented bits

AXI Read Data 128 messages are transmitted to provide read data of size 128 associated with a Read Address message.

RD256

Table 16. AXI read data 256 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
RID	8	Read address ID
SHIFT	2	Lane shift
RDATA	256	Read data
RRESP	2	Read response
RLAST	1	Read last
Reserved	7	
CRC	12	Cyclic redundancy code
	304 (38 bytes)	Total implemented bits

AXI Read Data 256 messages are transmitted to provide read data of size 256 associated with a Read Address message.

RD512

Table 17. AXI read data 512 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty

Table continues on the next page...

Table 17. AXI read data 512 message (continued)

Field Name	Field Size	Description
RID	8	Read address ID
SHIFT	1	Lane shift
RDATA	512	Read data
RRESP	2	Read response
RLAST	1	Read last
Reserved	8	
CRC	12	Cyclic redundancy code
	560 (70 bytes)	Total implemented bits

AXI Read Data 512 messages are transmitted to provide read data of size 512 associated with a Read Address message.

RD1024

Table 18. AXI read data 1024 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
RID	8	Read address ID
Reserved	1	
RDATA	1024	Read data
RRESP	2	Read response
RLAST	1	Read last
Reserved	8	
CRC	12	Cyclic redundancy code
	1072 (134 bytes)	Total implemented bits

AXI Read Data 1024 messages are transmitted to provide read data of size 1024 associated with a Read Address message.

WA32

Table 19. AXI write address 32 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
AWID	8	Identification tag for a write transaction
AWADDR	32	The address of the first transfer in a write transaction
AWSIZE	3	Size, the number of bytes in each data transfer in a write transaction
AWLEN	8	Bength, the exact number of data transfers in a write transaction
AWBURST	2	Burst type, indicates how address changes between each transfer in a write transaction
AWLOCK	1	Provides information about the atomic characteristics of a write transaction
AWCACHE	4	Indicates how a write transaction is required to progress through a system
AWPROT	3	Protection attributes of a write transaction: privilege, security level, and access type
AWQOS	4	Quality of Service identifier for a write transaction
AWREGION	4	Region indicator for a write transaction
AWUSER	8	User-defined extension for the write address channel
Reserved	3	Reserved for AWSNOOP to indicate the transaction type for shareable write transactions (ACE-Lite)
Reserved	2	Reserved for AWDOMAIN to indicate the shareability domain of a write transaction (ACE-Lite)
Reserved	2	Reserved for AWBAR to indicate whether a transaction is a write barrier domain (ACE-Lite)

Table continues on the next page...

Table 19. AXI write address 32 message (continued)

Field Name	Field Size	Description
Reserved	1	Reserved for AWUNIQUE to indicate the data in this transaction is permitted to be held in a Unique cache state (ACE-Lite)
REGION_HIT	1	Logical address hit in region
Reserved	2	
CRC	8	Cyclic redundancy code
Total bits	112 (14 bytes)	Total implemented bits

The AXI Write Address 32 message is the initial message transmitted to initiate a write transaction. The AWUSER field is an extension to the message, which allows for secure access extensions.

WA64

Table 20. AXI write address 64 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
AWID	8	Identification tag for a write transaction
AWADDR	64	The address of the first transfer in a write transaction
AWSIZE	3	Size, the number of bytes in each data transfer in a write transaction
AWLEN	8	Length, the exact number of data transfers in a write transaction
AWBURST	2	Burst type, indicates how address changes between each transfer in a write transaction
AWLOCK	1	Provides information about the atomic characteristics of a write transaction
AWCACHE	4	Indicates how a write transaction is required to progress through a system
AWPROT	3	Protection attributes of a write transaction: privilege, security level, and access type
AWQOS	4	Quality of Service identifier for a write transaction

Table continues on the next page...

Table 20. AXI write address 64 message (continued)

Field Name	Field Size	Description
AWREGION	4	Region indicator for a write transaction
AWUSER	8	User-defined extension for the write address channel
Reserved	3	Reserved for AWSNOOP to indicate the transaction type for shareable write transactions (ACE-Lite)
Reserved	2	Reserved for AWDOMAIN to indicate the shareability domain of a write transaction (ACE-Lite)
Reserved	2	Reserved for AWBAR to indicate whether a transaction is a write barrier domain (ACE-Lite)
Reserved	1	Reserved for AWUNIQUE to indicate the data in this transaction is permitted to be held in a Unique cache state (ACE-Lite)
REGION_HIT	1	Logical address hit in one of the address translation regions
Reserved	14	
CRC	12	Cyclic redundancy code
	160 (20 bytes)	Total implemented bits

The AXI Write Address 64 message is the initial message transmitted to initiate a write transaction. The AWUSER field is an extension to the message, which allows for secure access extensions.

WD8

Table 21. AXI write data 8 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	8	Write address ID (AXI3 only)
BSTRB	2	Byte strobes
SHIFT	7	Lane shift
WDATA	8	Write data
WLAST	1	Write last

Table continues on the next page...

Field Name	Field Size	Description
Reserved	14	
CRC	8	Cyclic redundancy code
	64 (8 bytes)	Total implemented bits

AXI Write Data 8 messages are transmitted to provide write data of size 8 associated with a Write Address message.

WD16

Table 22. AXI write data 16 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	8	Write address ID (AXI3 only)
BSTRB	2	Byte strobes
SHIFT	6	Lane shift
WDATA	16	Write data
WLAST	1	Write last
Reserved	7	
CRC	8	Cyclic redundancy code
	64 (8 bytes)	Total implemented bits

AXI Write Data 16 messages are transmitted to provide write data of size 16 associated with a Write Address message.

WD32

Table 23. AXI write data 32 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	8	Write address ID (AXI3 only)

Table continues on the next page...

Table 23. AXI write data 32 message (continued)

Field Name	Field Size	Description
BSTRB	4	Byte strobes
SHIFT	5	Lane shift
WDATA	32	Write data
WLAST	1	Write last
Reserved	6	
CRC	8	Cyclic redundancy code
	80 (10 bytes)	Total implemented bits

AXI Write Data 32 messages are transmitted to provide write data of size 32 associated with a Write Address message.

WD64

Table 24. AXI write data 64 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	8	Write address ID (AXI3 only)
BSTRB	8	Byte strobes
SHIFT	4	Lane shift
WDATA	64	Write data
WLAST	1	Write last
Reserved	3	
CRC	8	Cyclic redundancy code
	112 (14 bytes)	Total implemented bits

AXI Write Data 64 messages are transmitted to provide write data of size 64 associated with a Write Address message.

WD128

Table 25. AXI write data 128 message

Field Name	Field Size	Description
MID	6	Message ID

Table continues on the next page...

Table 25. AXI write data 128 message (continued)

Field Name	Field Size	Description
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	8	Write address ID (AXI3 only)
BSTRB	16	Byte strobes
SHIFT	4	Lane shift
WDATA	128	Write data
WLAST	1	Write last
Reserved	7	
CRC	12	Cyclic redundancy code
	192 (24 bytes)	Total implemented bits

AXI Write Data 64 messages are transmitted to provide write data of size 64 associated with a Write Address message.

WD256

Table 26. AXI write data 256 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	8	Write address ID (AXI3 only)
BSTRB	32	Byte strobes
SHIFT	2	Lane shift
WDATA	256	Write data
WLAST	1	Write last
Reserved	9	
CRC	12	Cyclic redundancy code
	336 (42 bytes)	Total implemented bits

AXI Write Data 256 messages are transmitted to provide write data of size 256 associated with a Write Address message.

WD512

Table 27. AXI write data 512 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	8	Write address ID (AXI3 only)
BSTRB	64	Byte strobes
SHIFT	1	Lane shift
WDATA	512	Write data
WLAST	1	Write last
Reserved	10	
CRC	12	Cyclic redundancy code
	624 (78 bytes)	Total implemented bits

AXI Write Data 512 messages are transmitted to provide write data of size 512 associated with a Write Address message.

WD1024

Table 28. AXI write data 1024 message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	8	Write address ID (AXI3 only)
BSTRB	128	Byte strobes
Reserved	1	
WDATA	1024	Write data
WLAST	1	Write last
Reserved	10	
CRC	12	Cyclic redundancy code
	1200 (150 bytes)	Total implemented bits

AXI Write Data 1024 messages are transmitted to provide write data of size 1024 associated with a Write Address message.

Write data response

Table 29. Write data response message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
BID	6	Write address ID
BRESP	2	Write response
CRC	8	Cyclic redundancy code
	32 (4 bytes)	Total implemented bits

Write data response messages are transmitted to both signal write response information as well as burst beat information.

FC

Table 30. Flow control message

Field Name	Field Size	Description
MID	6	Message ID
EBC	10	AXI Elasticity buffer credit. Values of 0x000 to 0x3FE indicate the number of double-bytes that have been cleared (e.g., 001 means 2 bytes cleared) Value of 0x3FF indicates that the remote chip's Elasticity Buffer is completely empty
Reserved	10	
CRC	6	Cyclic redundancy code
	32 (4 bytes)	Total implemented bits

Flow Control Ack messages are transmitted as a last resort to communicate elasticity buffer credit information. It is preferable to include elasticity buffer credit information in a scheduled AXI message.

Flow control

Flow control logic is necessary to prevent messages from being transmitted when the remote chip is not ready yet. That is, the link logic has elasticity buffers to hold numerous messages until the associated transactions are completed. So flow control logic in the TX Manager assures there are no scenarios where an EB over-flow can occur.

The intent of flow control for AXI messages is to assure the highest throughput for AXI transactions between chips, while consuming the least number of flow control message bits transmitted between chips. The function of flow control message bits is to support a debit/credit system to indicate the status of the AXI EB.

Elasticity Buffer Credit (EBC)

EBC Indicates how many double bytes are cleared from the other die's AXI EB, or if it is empty. The empty status should be communicated periodically so that the die can remain in sync when operating over long durations.

7 Appendix

The following sections describe additional information not currently available in the RTL version associated with this specification version.

7.1 AXI Messages - supported and future plans

The table below provides information about currently supported AXI messages as well as those architected for future implementation.

Table 31. diPortSD messages for AXI including flow control

Message Name	Message ID (6 bits)	Current or Future
Idle	0x00	Supported
Read Address 32	0x01	Supported
Read Address 64	0x02	Future
Read Data 8	0x03	Supported
Read Data 16	0x04	Supported
Read Data 32	0x05	Supported
Read Data 64	0x06	Supported
Read Data 128	0x07	Future
Read Data 256	0x08	Supported
Read Data 512	0x09	Future
Read Data 1024	0x0a	Future
Write Address 32	0x11	Supported
Write Address 64	0x12	Future
Write Data 8 bits	0x13	Supported
Write Data 16 bits	0x14	Supported
Write Data 32 bits	0x15	Supported
Write Data 64 bits	0x16	Supported
Write Data 128 bits	0x17	Future
Write Data 256 bits	0x18	Supported
Write Data 512 bits	0x19	Future
Write Data 1024 bits	0x1a	Future
Write Data Response	0x1b	Supported
Flow Control	0x30	Supported

7.2 Support for BoW PHY

The diPort controller TX and RX flits map naturally to the BoW PHY due to strong definition similarities. As illustrated below, an initial gasket is added to the diPort controller to convert to and from the LPIF interface, and to support up to 4 TX and 4 RX BoW PHYs. It is configurable to support 1, 2 and 4.

Other gaskets can be designed to support more BoW PHYs (e.g., 8 TX and 8 RX PHYs).

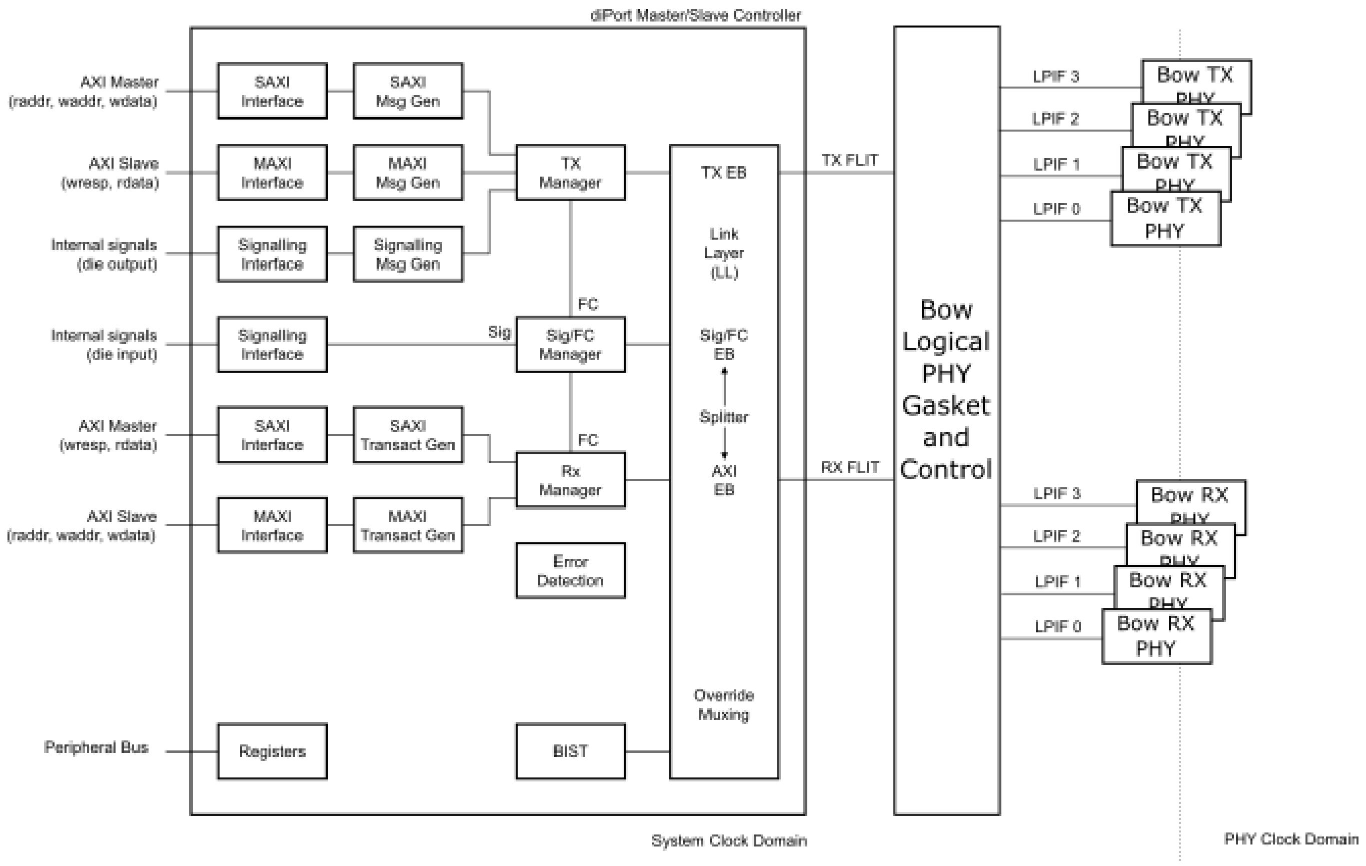


Figure 5. BoW Gasket

The following table lists the implemented LPIF interface signals. For more information refer to the LPIF specification version 1.0 available on-line.

Based on the number of BoW PHYs configured, 1 TX and 1 RX, 2 TX and 2 RX, or 4 TX and 4 RX, there will be a similar number of LPIF interfaces identified as illustrated in the figure.

Table 32. LPIF Signal List

Signal	Width (Bits)	Description
lclk	1	The clock frequency at which LPIF operates. This is an input to both the Link & the Phy.
pl_trdy	1	Indicates Phy is ready to accept data
pl_data	Configurable - 144, 288 and 576	Phy to LL data
pl_valid	1	Valid associated w/ pl_data
pl_state_sts	4	Phy to LL status indication of interface
pl_stallreq	1	Phy to LL to flush all messages for state transitions
pl_wake_ack	1	Ack from the PHY that its clocks are ungated.
pl_exit_cg_req	1	Phy req to LL to remove clock gating

Table continues on the next page...

Table 32. LPIF Signal List (continued)

Signal	Width (Bits)	Description
lp_irdy	1	Indicates the LL is ready to accept data
lp_pri	1	Priority signaling from a given LL
lp_data	Configurable - 144, 288 and 576	LL to Phy data
lp_valid	1	Valid associated w/ lp_data
lp_stallack	1	LL to Phy to indicate that messages are aligned
lp_wake_req	1	LL to Phy req to remove clock gating – asynch signal
lp_exit_cg_ack	1	Ack from the LL that its clocks are ungated
lp_state_req	4	LL to Phy to request state change
lp_rcvd_crc_err	1	LL to Phy indication of CRC error for aggregation

The configurable width of the pl_data and lp_data signals are provided to enable implementation trade-off to balance latency as well as enable lower system clock frequency demands for the diPort controller.

The repeating pattern that composes the format for pl_data and lp_data signals for BoW PHY are:

- 1 Aux bit
- 16 Data bits
- 1 FEC bit

NOTE

FEC is not defined at this time. Soliciting a light-weight FEC that minimizes latency for inclusion in the message types

This pattern will be repeated sequentially some number of times to represent each sequential BoW PHY transmit (lp_data) and receive (pl_data). This pattern is repeated 8 times (144 bit width), 16 times (244 bit width) or 32 times (576 bit width) based on the configuration selected.

Accordingly the TX and RX Flits that travel between the diPort controller and the Gasket have a similar format but of a much larger flit size. A much larger flit size is needed to accommodate the configurable number of BoW PHYs as well as the configurable width of lp_data and pl_data. Thus the following TX/RX flit widths are supported:

- Flit widths of 144, 288 or 576 for 1 BoW PHY
- Flit widths of 288, 576 or 1152 for 2 BoW PHYs
- Flit widths of 576, 1152 or 2304 for 4 BoW PHYs

The diPort controller natively uses a chip-select signal (CS) to indicate both framing and valid. This signal is naturally implemented as the BoW PHY Aux signal.