Arm® CoreLink™ NI-700 Network-on-Chip Interconnect

Revision: r2p0

Technical Reference Manual

arm

101566_0200_08_en

Arm® CoreLink™ NI-700 Network-on-Chip Interconnect

Technical Reference Manual

Release Information

Document History

Issue	Date	Confidentiality	Change
0000-00	14 March 2019	Confidential	First DEV release for r0p0.
0000-01	28 June 2019	Confidential	Second DEV release for r0p0.
0000-02	05 August 2019	Confidential	First ALP release for r0p0.
0000-03	30 September 2019	Confidential	Third DEV release for r0p0.
0000-04	05 November 2019	Confidential	First BET release for r0p0.
0000-05	27 March 2020	Confidential	First LAC release for r0p0.
0001-01	17 July 2020	Confidential	First DEV release for r0p1.
0100-01	29 October 2020	Non-Confidential	First EAC release for r1p0.
0200-08	24 March 2021	Non-Confidential	First EAC release for r2p0.

Non-Confidential Proprietary Notice

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Product Status

The information in this document is Final, that is for a developed product.

Web Address

developer.arm.com

Progressive terminology commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used terms that can be offensive. Arm strives to lead the industry and create change.

This document includes terms that can be offensive. We will replace these terms in a future issue of this document.

If you find offensive terms in this document, please contact *terms@arm.com*.

Contents
Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Technical Reference Manual

Preface

This preface introduces the *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Technical Reference Manual*.

It contains the following:

About this book

This book is for the Arm® CoreLink™ NI-700 Network-on-Chip Interconnect.

Product revision status

The r*x*p*y* identifier indicates the revision status of the product described in this book, for example, r*1*p*2*, where:

r*x* Identifies the major revision of the product, for example, r1.

p*y* Identifies the minor revision or modification status of the product, for example, p2.

Intended audience

This book is written for system designers, system integrators, and programmers who are designing or programming a *System on Chip* (SoC) that uses the NI-700 Network-on-Chip Interconnect.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction

This chapter provides an overview of the CoreLink NI-700 Network-on-Chip Interconnect.

Chapter 2 Functional description

This chapter describes the functionality of NI-700.

Chapter 3 Programmers model

This chapter describes the NI-700 programmers model.

Chapter 4 Performance monitoring

This chapter describes the Performance Monitoring Unit (PMU), which enables system integrators to monitor events to optimize the design of the system.

Appendix A Signal descriptions

This appendix describes the external signals of the NI-700.

Appendix B Revisions

This appendix describes the technical changes between released issues of this book.

Glossary

The Arm® Glossary is a list of terms used in Arm documentation, together with definitions for those terms. The Arm Glossary does not contain terms that are industry standard unless the Arm meaning differs from the generally accepted meaning.

See the *Arm® Glossary* for more information.

Typographic conventions

italic

Introduces special terminology, denotes cross-references, and citations.

bold

Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

`monospace`

Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

`mono`space

Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

monospace italic
: Denotes arguments to monospace text where the argument is to be replaced by a specific value.

monospace bold
: Denotes language keywords when used outside example code.

<and>
: Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

SMALL CAPITALS
: Used in body text for a few terms that have specific technical meanings, that are defined in the *Arm® Glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Timing diagrams

The following figure explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.

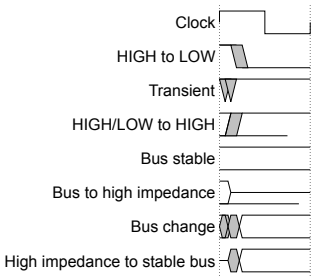

Figure 1 Key to timing diagram conventions

Signals

The signal conventions are:

Signal level
: The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means:
- HIGH for active-HIGH signals.
- LOW for active-LOW signals.

Lowercase n
: At the start or end of a signal name, n denotes an active-LOW signal.

Additional reading

Arm publications

This book contains information that is specific to this product. See the following documents for other relevant information. See https://developer.arm.com/ for Arm documents.

- *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Configuration and Integration Manual* (101567).
- *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Release Note* (PJDOC-1779577084-33002).
- *AMBA® AXI and ACE Protocol Specification* (IHI 0022H).
- *Arm® AMBA® 5 AHB Protocol Specification, AHB5, AHB-Lite* (IHI 0033B.b).
- *AMBA® APB Protocol Specification, Version: 2.0* (IHI 0024C).
- *AMBA® Low Power Interface Specification Arm® Q-Channel and P-Channel Interfaces* (IHI 0068C).
- *Arm® CoreSight™ Architecture Specification v3.0* (IHI 0029E).
- *Principles of Arm® Memory Maps White Paper* (DEN 0001).

Other publications

This section lists relevant documents that are published by third parties:

- *JEDEC Standard Manufacturer's Identification Code, JEP106* http://www.jedec.org.

Feedback

Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to *errata@arm.com*. Give:

- The title *Arm CoreLink NI-700 Network-on-Chip Interconnect Technical Reference Manual*.
- The number 101566_0200_08_en.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Note

Arm tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

Chapter 1
Introduction

This chapter provides an overview of the CoreLink NI-700 Network-on-Chip Interconnect.

It contains the following sections:

1.1 About the CoreLink NI-700 Network-on-Chip Interconnect

The CoreLink NI-700 Network-on-Chip Interconnect is a highly configurable AMBA-compliant system-level interconnect. NI-700 enables you to create a non-coherent interconnect that is optimized to the *Power, Performance, and Area* (PPA) requirements of your SoC design.

NI-700 forms the non-coherent interconnect part of the NI-700 product. It is designed to scale, making it suitable for large designs as a backplane interconnect. Using multiple routers and various topology options you can connect multiple masters and slaves that use different AMBA protocols to NI-700.

NI-700 supports the AMBA AXI5, ACE5-Lite, AHB5, APB3, and APB4 protocols. The specific AXI5 capabilities that NI-700 supports are:

AXI5

- Atomic transactions
- QoS_Accept, on the master interface side only
- Trace signals
- Loopback signals
- Wakeup signals
- NSAccess identifiers

Note

Except for wakeup signaling, all other OPTIONAL AXI5 capabilities can be disabled in NI-700 when integrated with an AXI4-based system.

ACE5-Lite

- Cache stashing transactions
- DeAllocation transactions
- Persist CMO

AXI5.F

- *Memory System Resource Partitioning and Monitoring* (MPAM)
- UniqueID
- Read data chunking
- Cache maintenance operations on write channel
- Read interleaving property

Note

ASNI cannot guarantee that data beats between transactions with different IDs do not interleave. Therefore, the property Read data interleave disabled is always False for ASNI.

AXI5.H

- MTE
- Prefetch request
- Data writes combined with cache maintenance operations

AXI3

NI-700 also supports AMBA AXI3 on the master interface connection to downstream slaves. For AXI3 supported features, see *1.6 Configurable options* on page 1-19

For more information, see the *AMBA® AXI and ACE Protocol Specification*.

NI-700 supports the following AMBA interfaces:

- AXI5

Note

For AXI5 supported features, see *1.6 Configurable options* on page 1-19.

- AHB5

  Note

  The AHB5 specification adds a set of OPTIONAL capabilities to AHB-Lite.

  Note

  To connect an AHB-Lite master or slave to NI-700, disable the OPTIONAL capabilities on the HMNI or HSNI.

- ACE5-Lite
- APB3 and APB4
- AXI3, only on NI-700 master interfaces

NI-700 contains the following functional units:

- AMBA *AXI Slave Network Interfaces* (ASNIs)
- AMBA *AXI Master Network Interfaces* (AMNIs)
- AMBA *AHB Slave Network Interfaces* (HSNIs)
- AMBA *AHB Master Network Interfaces* (HMNIs)
- AMBA *APB Master Network Interfaces* (PMNIs)
- Routers
- *Power and Clock Domain Crossing* (PCDC) units
- Interconnect link upsizing and downsizing (SERDES) units
- *Performance Monitoring Unit* (PMU)

ASNIs convert AXI and ACE-Lite transactions into NI-700 *Generic Transport* (GT) packets.

AMNIs convert packets from NI-700 GT packets to AXI or ACE-Lite protocol.

HSNIs convert AHB5 and AHB-Lite transactions into NI-700 GT packets.

HMNIs convert packets from the NI-700 GT packets to AHB5 or AHB-Lite protocol.

PMNIs convert NI-700 GT packets to APB protocol.

1.2 Key features

The NI-700 interconnect supports various features to enable you to use it at an SoC level.

NI-700 supports the following key features:

- Native support for the following AMBA protocols:
  - AXI5, AXI-G, and AXI-H
  - AHB5
  - APB3 and APB4
  - AXI3, only on NI-700 master interfaces
- Packet transfer over multiple clock, power, and voltage domains
- Source-based packet routing
- Worm-hole routing with support for multiple *Resource Planes* (RPs)
- Flit-level credit-based flow control
- *Quality of Service* (QoS) features to permit prioritization of information transfer
- Distributed switching mechanism to enable traffic management and protect against network saturation
- Variable, user-defined topology that is specified through Socrates™ IP tooling platform

The following features are not supported:

Table 1-1 Unsupported AMBA features

AMBA protocol	Unsupported features
AXI	AXI region identifiers (**AxREGION** signaling)
	Barrier transactions (**AxBAR** signaling)
AXI3	Not supported on NI-700 slave interfaces, only supported on NI-700 master interfaces
	Write data interleaving is not supported
	NI-700 supports normal and exclusive accesses only, not locked accesses
	Write data dependencies • For AXI4 onwards, the AXI protocol added an extra dependency for write transactions. An AXI3 slave that accepts all write data and provides a write response before accepting the address, is not compliant with AXI4 or later versions of the AMBA AXI protocol. The AXI specification strongly recommends that any new AXI3 slave implementation includes this additional dependency. • The NI-700 AMNI conforms to the AXI4 dependency requirement. If a write response is received before the write address phase is accepted its behavior is UNPREDICTABLE. Downstream AXI3 slaves must conform to the AXI4 requirement to integrate directly with NI-700. • To integrate a downstream AXI3 slave that follows the AXI3 write dependency requirement requires an external wrapper. The external wrapper ensures a returning write response is not provided until the slave has accepted the appropriate address.
AHB	Locked transfers are not supported. ——— **Note** ——— **HMASTLOCK** indication is ignored at the AHB slave network interface. ———
	Multi-copy atomicity ——— **Note** ——— Not supported if early write response is enabled. However if early write response is disabled multi-copy atomicity is supported. ———
	Multiple slaves select (**HSELx** signaling)
	Split and retry

1.3 Compliance

NI-700 complies with Arm and external specifications and standards.

NI-700 complies with the following specifications:

- *AMBA® AXI and ACE Protocol Specification*

  ———— **Note** ————

  NI-700 does not support AXI4-Lite.

  ————————————

- *Arm® AMBA® 5 AHB Protocol Specification, AHB5, AHB-Lite*
- *AMBA® APB Protocol Specification, Version: 2.0*
- *AMBA® Low Power Interface Specification Arm® Q-Channel and P-Channel Interfaces*

Read this *Technical Reference Manual* (TRM) along with the following:

- Architecture Reference Manuals
- Protocol specifications
- Relevant external standards

The TRM does not duplicate information from these sources.

1.4 Interfaces

NI-700 has slave and master interfaces.

The following definitions apply:

Slave interface
An interface that receives input from a master device.

Master interface
An interface that sends output to a slave device.

The following figure shows how AXI master and slave devices interface with NI-700.

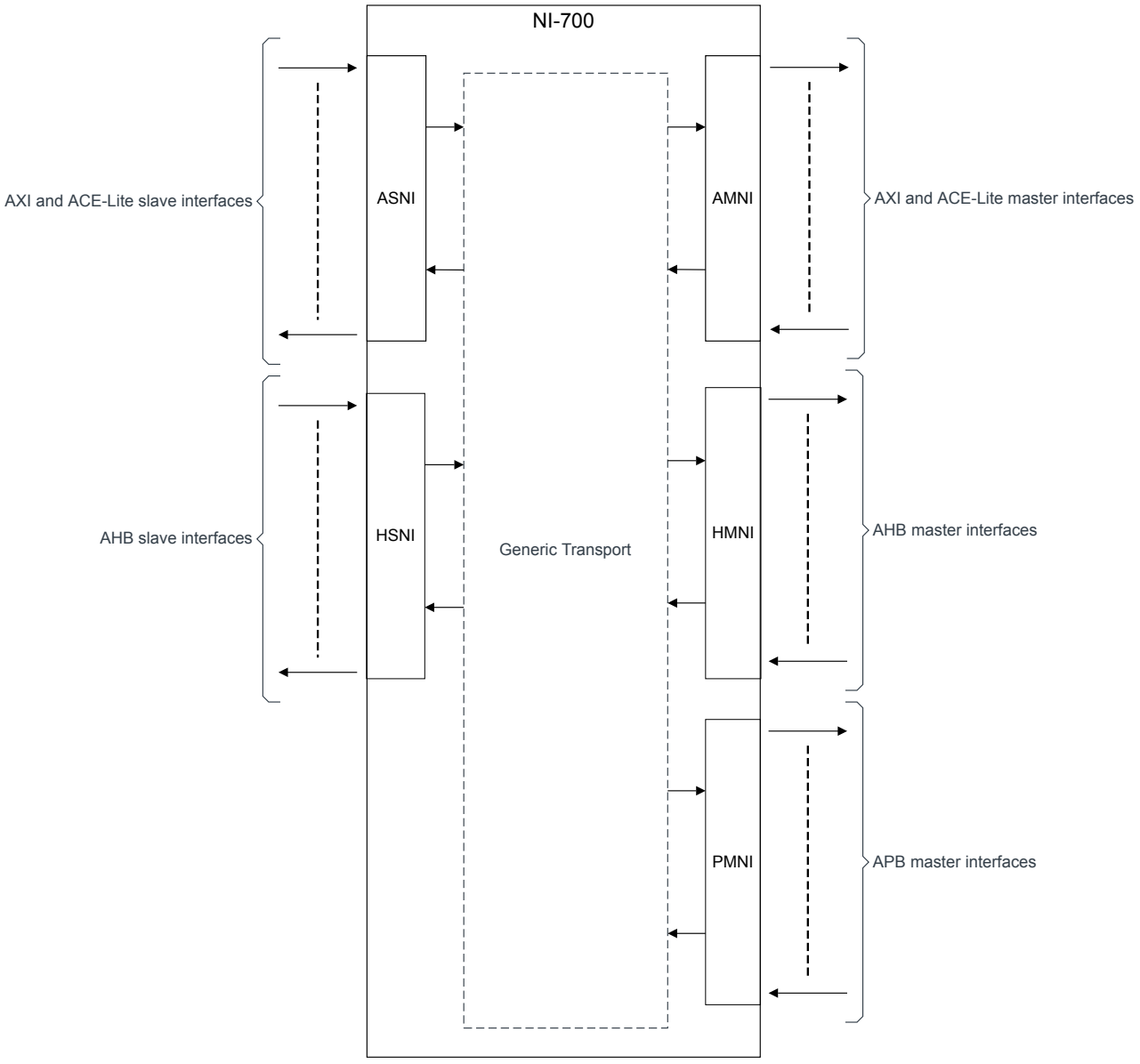

Figure 1-1 NI-700 top-level interfaces

To control the clock and power functions, NI-700 has *Low-Power Interfaces* (LPIs).

1.5 Architecture overview

NI-700 has an architecture that addresses the bandwidth and PPA considerations in your SoC design.

NI-700 is a high frequency, low latency interconnect. All endpoints and transport components have a minimum latency of one cycle per block except for HSNI requests which is two cycles. An NI-700 interconnect with a configured link width of 512-bit operating at a frequency of 1GHz provides raw bandwidth of 64GB/s.

To optimize system bandwidth and PPA, NI-700 provides the following architecture considerations:

- Multiple masters and slaves with a combination of AXI5, ACE5-Lite, ACE5-LiteACP, AHB5, APB3, and APB4 protocols
- AXI3 protocol on master interfaces only
- Packetizing mechanism that enables configurable link widths from 64-bit to 512-bit
- Independent widths of user-defined sideband signals for each channel
- *Resource Planes* (RP) to permit traffic isolation
- Non-blocking resource planes
- Configurable duplicate links between pairs of router units
- QoS regulators for improved QoS
- Address striping
- Highly flexible timing closure options
- Support for multiple clock domains and hierarchical clock gating
- Support for multiple power domains and power gating

The following figure shows an example of the NI-700 top-level architecture, with defined inputs and outputs.

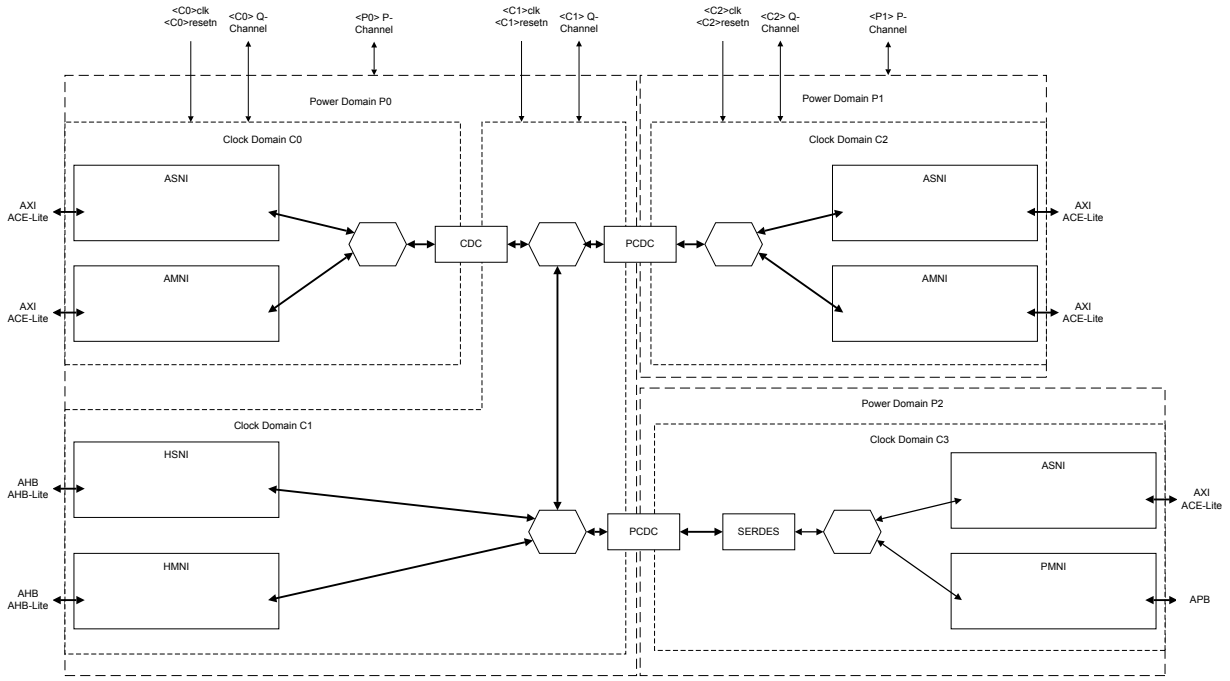

Figure 1-2 Example NI-700 top-level architecture with defined inputs and outputs

Note

The NI-700 PCDC unit is responsible for bridging between power domains and clock domains. The PCDC unit can be configured to provide only clock domain crossings or both power and clock domain crossings.

1.6 Configurable options

To meet specific design requirements, you can customize the top-level topology and the individual functional units of NI-700.

NI-700 has the following configurable options:

- The NI-700 microarchitecture is scalable up to a maximum of 255 master and slave interfaces. The following numbers apply to this release and any subsequent release includes any relevant updates:
  - Up to 128 slave network interfaces, ASNIs and HSNIs
  - Up to 127 master network interfaces, AMNIs, HMNIs, and PMNIs
- Voltage, power, and clock domains:
  - 1-32 voltage domains
  - 1-32 power domains where each power domain is in one voltage domain
  - 1-32 clock domains where each clock domain is in one power domain
  - Power and clock domain crossing within the network supporting synchronous, asynchronous, and ratio clock domain crossings
  - RTL hierarchy by voltage, power then clock domain
  - RTL hierarchy according to user-specified grouping of components
- Address map:
  - Address-based routing from each slave interface to corresponding master interfaces based on user specified address map
  - Each slave interface can have a separate address map.
  - Each address map supports multiple address regions with each region aligned and sized based on a 4KB granularity.
  - Each region in the address map can target one master interface or can be hashed across two or four master interfaces.
- Cache line size:
  - NI-700 only supports the following cache line sizes:

Table 1-2 Supported cache line sizes

Data width	Cache line size
32-bit	64-Bytes
64-bit, 128-bit, 256-bit, 512-bit	64-Bytes or 128-Bytes
1024-bit	128-Bytes

- Topologies:
  - Flexible topology choices using routers with up to eight inputs and up to eight outputs
  - Up to four *Resource Planes* (RPs) to reduce *Head-of-line* (HoL) blocking
  - Configurable link sizes and link crediting with ability to resize flits within the network by using SERDES components
  - Bridging between different power and clock domains using PCDC components
  - Ability to merge read and write channels to reduce wire count and area
  - Ability to duplicate channels for more bandwidth
- Unit-level configuration options:
  - Flexible timing closure options
  - Configurable transaction tracker depths
  - Burst splitting logic. This option can be included when transactions are to be split or excluded to save area on designs where the feature is not required.
  - QoS regulators which can update the QoS value on a transaction according to latency targets.

— *Interconnect Device Management* (IDM)
— Configurable FIFO sizes when crossing clock and power domains

This section contains the following subsections:

1.6.1 ASNI configuration options

You can configure the ASNI unit to meet your specific design requirements.

You can configure the following options:

- Address width between 32-bit and 64-bit
- Data width of 32-bit, 64-bit, 128-bit, 256-bit, 512-bit, or 1024-bit
- User sideband signal width, see *2.13.8 User signals* on page 2-112
- Write acceptance capability of 1-256 transactions

  Note

  Sometimes the ASNI might accept more transactions than specified in the write acceptance capability. For example, configuring a register slice at the slave interface position increases the acceptance capability.

- Read acceptance capability of 1-256 transactions

  Note

  Sometimes the ASNI might accept more transactions than specified in the read acceptance capability. For example, configuring a register slice at the slave interface position increases the acceptance capability.

- Minimum atomic acceptance
  — Minimum atomic acceptance only applies if the Atomic_Transactions property is enabled on the ASNI. If enabled, then the total read tracker size is read acceptance + minimum atomic acceptance.
  — While atomic transactions are received on the write channel, three of the atomic variants, load, compare, and swap, require a read response also. This process uses a tracker entry in the read tracker.
  — The minimum atomic acceptance parameter provides a guarantee on the minimum number of read tracker entries reserved for atomics.
- Timing isolation:
  — From the external master
  — From the network
- Read reorder depth of 1-64 entries

  Note

  Permitted read reorder depth values are 1, 2, and multiples of 4 including 64.

- Write data FIFO depth of 0-32 entries
- ID width of 1-24-bit
- Ordered write observation, select one of enabled, disabled, or pin

  Note

  The ordered write observation feature is an AXI4 property. For more information, see the *AMBA® AXI and ACE Protocol Specification*.

- IDM enable
- IDM device ID
- Whether Burst splitting logic is included
- Presence of QoS regulators of several types:

Read regulator present	Enables QoS regulator for AR channel
Write regulator present	Enables QoS regulator for AW channel
Combined regulator present	Enables QoS regulator that regulates traffic according to combined bandwidth across both AR and AW channel

The following table shows the features that the ASNI supports for a specific interface type.

Table 1-3 Features supported by the ASNI for a specific interface type

Interface type	Parameter name	Support
AXI5 and ACE-Lite	`Wakeup_Signals`	Required **Note** Devices attached to NI-700 must support Wakeup_Signals.
	`Check_Type`	Not supported
	`Poison`	Not supported
	`Trace_Signals`	OPTIONAL **Note** User configurable.
	`Unique_ID_Support`	OPTIONAL
	`QoS_Accept`	Not supported
	`Loopback_Signals`	OPTIONAL **Note** However, if enabled, set to 8-bits only.
	`Untranslated_Transactions`	Not supported
	`NSAccess_Identifiers`	OPTIONAL
	`MPAM_Support`	
	`Read_Interleaving_Disabled`	Always set to `FALSE`
	`Read_Data_Chunking`	OPTIONAL
	`Atomic_Transactions`	
	`MTE_Support`	

Table 1-3 Features supported by the ASNI for a specific interface type (continued)

Interface type	Parameter name	Support
ACE-Lite	CMO_On_Read	OPTIONAL
	CMO_On_Write	
	Persist_CMO	
	Write_Plus_CMO	
	Cache_Stash_Transactions	
	DeAllocation_Transactions	
	Prefetch_Transaction	

1.6.2 AMNI configuration options

You can configure the AMNI unit to meet your specific design requirements.

You can configure the following options:

- Address width of 32-64 bit
- Data width of 32-bit, 64-bit, 128-bit, 256-bit, 512-bit, or 1024-bit
- User sideband signal width, see *2.13.8 User signals* on page 2-112. User signals are applicable to all AMNI interface types including AXI3.
- Number of RPs present in each channel (read request, write request, read response, and write response)
- Write issuing capability of 1-256 transactions
- Read issuing capability of 1-256 transactions
- Minimum atomic issue:
  - Minimum atomic issue only applies if the Atomic_Transactions property is enabled on the AMNI. If enabled, then the total read tracker size is read issue + minimum atomic issue.
  - While atomic transactions are received on the write channel, three of the atomic variants, load, compare, and swap, also require a read response. This process uses a tracker entry in the read tracker.
  - The minimum atomic issue parameter provides a guarantee on the minimum number of read tracker entries reserved for atomics
- Timing isolation:
  - From the external slave
  - From the network
- IDM enable
- IDM device ID
- AXI ID width of 1-32 bit. To form the outgoing AXI ID, the AMNI appends the SRCID of the incoming request to the request AXI ID. The SRCID of the incoming request is captured in the node_id field of the *ASNI_NODE_TYPE, Node type register for ASNI registers* on page 3-172.
- Ordered write observation as one of enabled, disabled, or pin.

  Note

  The ordered write observation feature is an AXI4 property. For more information, see the *AMBA® AXI and ACE Protocol Specification*.

Note

The **AxREGION** signal is not supported.

You can configure AMNI to have AXI5, ACE5-Lite, ACE5-LiteACP, or AXI3 as the master interface type.

ACE5-LiteACP has many constraints, some of which are transaction constraints and some of which are interface constraints. These constraints are specified in the AMBA AXI specification.

The following table shows the features that the AMNI supports for a specific interface type.

Table 1-4 Features that the AMNI supports for a specific interface type

Interface type	Parameter name	Support
AXI5 and ACE-Lite	Wakeup_Signals	Required **Note** AMNI has an output AWAKEUP signal. The downstream slave can choose to use it or ignore it.
	Check_Type	Not supported
	Poison	
	Trace_Signals	OPTIONAL
	Unique_ID_Support	OPTIONAL
	Qos_Accept	
	Loopback_Signals	OPTIONAL **Note** However, if enabled, it is set to 8-bit only.
	Untranslated_Transactions	Not supported
	NSAccess_Identifiers	OPTIONAL
	MPAM_Support	
	Read_Interleaving_Disabled	
	Read_Data_Chunking	
	Atomic_Transactions	
	MTE_Support	
ACE-Lite	CMO_On_Read	OPTIONAL
	CMO_On_Write	
	Persist_CMO	
	Write_Plus_CMO	
	Cache_Stash_Transactions	
	DeAllocation_Transactions	
	Prefetch_Transaction	

Table 1-4 Features that the AMNI supports for a specific interface type (continued)

Interface type	Parameter name	Support
ACE5-LiteACP	Atomic_Transactions	Not supported as per ACE5-LiteACP protocol
	Write_Plus_CMO	
	Prefetch_Transaction	
	DeAllocation_Transactions	
	Cache_Stash_Transactions	OPTIONAL
	CMO_On_Read	Not supported as per ACE5-LiteACP protocol
	CMO_On_Write	
	Persist_CMO	
	Trace_Signals	OPTIONAL
	NSAccess_Identifiers	Not supported as per ACE5-LiteACP protocol
	MPAM_Support	OPTIONAL
	Unique_ID_Support	
	Read_Data_Chunking	
	Loopback_Signals	Not supported as per ACE5-LiteACP protocol
	MTE_Support	
	Qos_Accept	
	Read_Interleaving_Disabled	OPTIONAL
	Untranslated_Transactions	Not supported
	Check_Type	
	Poison	

Table 1-4 Features that the AMNI supports for a specific interface type (continued)

Interface type	Parameter name	Support
AXI3	Atomic_Transactions	Not supported
	Trace_Signals	
	NSAccess_Identifiers	
	MPAM_Support	
	Unique_ID_Support	
	Read_Data_Chunking	
	Loopback_Signals	
	MTE_Support	
	QoS_Accept	
	Read_Interleaving_Disabled	
	DeAllocation_Transactions	
	Cache_Stash_Transactions	
	CMO_On_Read	
	CMO_On_Write	
	Persist_CMO	
	Write_Plus_CMO	
	Prefetch_Transaction	
	Untranslated_Transactions	
	Check_Type	
	Poison	

1.6.3 HSNI configuration options

You can configure various options for each HSNI instance to meet your specific design requirements. Some HSNI properties are fixed in NI-700.

You can configure the following options for each HSNI instance:

- Interface type

  ——— **Note** ———

  HSNI supports AHB5 and mirrored interfaces.

  ———————————

- Read and write data widths of 32-bit, 64-bit, 128-bit, 256-bit

  ——— **Note** ———

  Read and write data widths must be the same.

  ———————————

- ID width of 1-8
- User sideband signal width, see *2.13.8 User signals* on page 2-112
- Write acceptance capability of 1-16 transactions

Note

There are occasions when the HSNI might accept more transactions than specified in the write acceptance capability. For example, configuring a register slice at the slave interface position increases the acceptance capability.

- Configurable **HMASTER** width of 1-24 bit

Table 1-5 HSNI configuration options

Parameter Name	Support
`Extended_Memory_Types`	OPTIONAL enabled or disabled
`Secure_Transfers`	OPTIONAL pin, programmable, Secure, or Non-secure
`Endianness`	BE32 only **Note** Only supports word-invariant little-endianness.
`Exclusive_Transfers`	OPTIONAL enabled or disabled
`Mirror_Interface`	
`Multi_Copy_Atomicity`	OPTIONAL **Note** `False` if early write response is enabled, otherwise `True`.
`Stable_Between_Clock`	`False`

Note

HSNI does not support multiple slaves select (HSELx signaling), split and retry, or locked transfers.

- *Extended Memory Type* (EMT) support
- Secure transfer support
- Exclusive transfer support
- Early Burst termination acceptance
- Burst conversion support
- Early write response support. When early write response is enabled, you can configure HSNI to support between 1, 2, or 4-64 outstanding writes, in increments of 4.
- Write data buffer FIFO depth of 0-16 data beats
- Presence of programmable QoS regulators, either present or absent
- Timing isolation:
  - From the external master
  - From the network
- IDM enable
- IDM device ID

The following HSNI properties are not configurable:

- Address width, this value is fixed at 32-bit
- Endianness, HSNI only supports word-invariant little-endianness

1.6.4 HMNI configuration options

You can configure various options for each HMNI instance to meet your specific design requirements. Some HMNI properties are fixed in NI-700.

You can configure the following options for each HMNI instance:

- HMNI supports AHB5 standard and mirrored interface types
- Read and write data widths of 32, 64, 128, or 256

  Note

  Read and write data widths must be the same.

- ID width of 1-8
- User sideband signal width, see *2.13.8 User signals* on page 2-112

Table 1-6 HMNI configuration options

Parameter Name	Support
`Extended_Memory_Types`	OPTIONAL enabled or disabled
`Secure_Transfers`	OPTIONAL select PIN, Register, Secure or Non-secure
`Endianness`	BE32 only **Note** Only supports word-invariant little-endianness.
`Exclusive_Transfers`	OPTIONAL enabled or disabled
`Mirror_Interface`	Enabled or disabled
`Stable_Between_Clock`	`False`

Note

HMNI uses **HMASTLOCK** when splitting any non-modifiable Burst. HMNI asserts **HMASTLOCK** to prevent other masters accessing the same memory location during a non-modifiable Burst split sequence.

Note

HMNI does not support multiple slave select (HSELx signaling) and split and retry.

- *Extended Memory Type* (EMT) support
- Secure transfer support
- Secure access support
- Exclusive transfer support
- Timing isolation:
  - From the external master
  - From the network
- IDM enable
- IDM device ID

The following HMNI properties are not configurable:

- Address width, this value is fixed at 32-bit
- Endianness, HMNI only supports word-invariant little-endianness

1.6.5 PMNI configuration options

You can configure various options for each PMNI instance to meet your specific design requirements. Some PMNI properties are fixed in NI-700.

You can configure the following options for each PMNI instance:

- APB protocol type from APB3 or APB4
- Secure access support which a register can control or is determined from a pin
- Timing isolation:
  - From the external master
  - From the network
- IDM enable
- IDM device ID

The following PMNI properties are not configurable:

- Address width, this value is fixed at 32-bit
- Read and write data widths, these values are fixed at 32-bit

1.7 Test features

NI-700 supports a scan cell insertion methodology for your SoC *Design for Test* (DFT) strategy. DFT control signals provide high coverage for your test strategy for NI-700 design.

The DFT control signals provide the following capabilities:

- Disabling internal resets
- Controlling architectural clock gating
- Clock disable pin. Use the **DFT<CLKNAME>DISABLE** inputs to disable specific clock regions to reduce power consumption during testing.

For more information about the test features of NI-700, see the *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Configuration and Integration Manual*.

1.8 Product design flow and documentation

SoC design flow has several processes: implementation, integration, and programming. NI-700 documentation supports the efficient and effective achievement of design flow tasks.

This section contains the following subsections:

1.8.1 Product design flow

You are required to perform several processes before usingNI-700. To obtain the best performance, Arm recommends that you perform some of the implementation stages before integrating NI-700 into your wider SoC.

Product design flow has the following processes:

Implementation
The implementer configures and synthesizes the *Register Transfer Level* (RTL).

Integration
The integrator connects the implemented design into an SoC. Integration includes connecting the design to:

- A memory system
- Processors
- Peripherals

Final SoC implementation
The process of implementing the final, fully integrated SoC in silicon. Arm can only provide guidance relevant to its own products for this process. If Arm provides guidance on this process for your product, then a separate document is included in the implementation bundle for that product.

Programming
Programming is the last process. The system programmer develops the software that is required to configure and initialize NI-700, and tests the required application software.

For information on NI-700 documents that provide information on these processes, see *1.8.2 Product documentation* on page 1-31.

Each process:

- Is separate, and a different person can complete it
- Can include implementation and integration choices that affect the behavior and features of NI-700, and therefore the other tasks in the flow

When configuring NI-700, the Socrates IP tooling platform provides a physically aware tooling canvas with an integrated performance feedback to optimize the selected path for faster timing closure.

The operation of the final device depends on the following:

Build configuration
The implementer chooses the configuration options that affect the preprocessing of the RTL source files. These options usually include or exclude the logic that affects one or more of the features. Features can be:

- Area
- Maximum frequency
- Performance of the resulting macrocell

For example, the implementer can control the number of outstanding transactions that each master and slave interface supports.

Configuration inputs
The integrator configures some features of NI-700 by tying inputs to specific values. These configurations affect the start-up behavior before you specify the software configuration. They can also limit the options available to the software.

Software configuration
The programmer configures NI-700 by programming values into registers. These values affect the behavior of NI-700, for example, by enabling QoS features.

1.8.2 Product documentation

Each NI-700 document has an intended audience and is associated with specific tasks in the design flow. These documents do not reproduce Arm architecture and protocol information.

For relevant protocol and architectural information that relates to this product, see *Additional reading* on page 9.

The NI-700 documentation is as follows:

Technical Reference Manual
The *Technical Reference Manual* (TRM) describes the functionality and the effects of functional options on the behavior of the NI-700. It is required at all stages of the design flow. The choices that are made in the design flow can mean that some behaviors that the TRM describes are not relevant. If you are programming NI-700, then contact:

- The implementer to determine:
  - The build configuration of the implementation
  - The integration, if any, that was performed before implementing the NI-700
- The integrator to determine the pin configuration of the device that you use

Configuration and Integration Manual
The *Configuration and Integration Manual* (CIM) contains:

- A description of the NI-700 features
- A list of the design-time configuration options
- A list of the reset-time configuration options
- The available build configuration options and related issues in selecting them
- How to configure the RTL with the build configuration options
- How to run test vectors
- The processes to sign off on the configured design.
- Considerations when integrating the NI-700 into your system

The Arm product deliverables include reference scripts and information about using them to implement your design. Reference methodology flows that Arm supplies are example reference implementations. Contact your EDA vendor for EDA tool support.

The CIM is a confidential book that is only available to licensees.

Chapter 2
Functional description

This chapter describes the functionality of NI-700.

It contains the following sections:

2.1 About the functional units

The NI-700 is built from functional units. Each functional unit has its own transfer function.

You can use the Socrates IP tooling platform to create network topologies that are built from the functional units.

The functional units process and route network traffic across the NI-700 network layer by:
- Converting between AXI, AHB, or APB transactions and NI-700 GT protocol flits
- Routing flits across the network between any slave interface and any master interface
- Arbitrating flits according to QoS ordering and resource plane allocation
- Handling the passage of flits across different power and clock domains and across areas of the network with different flit widths
- Monitoring the performance of the network

Note

All functional units have the following configurable options:
- Number of credits available for each channel
- Flit width for each channel

This section contains the following subsections:

2.1.1 AXI Slave Network Interface

The NI-700 ASNI unit receives and processes requests from AXI master devices. The ASNI unit packetizes transactions into flits according to the NI-700 GT protocol and depacketizes GT response flits into AXI responses.

The ASNI unit performs the following functions:
- Conversion of requests, data, and response transactions between AXI and GT protocol
- Transaction address decode into:
  - Target ID
  - Route vector
  - *Decode Error* (DECERR) indication for requests to out of range memory regions
  - Data width resizing indication
  - Stripe indication
- Burst splitting of incoming transactions. The ASNI splits Bursts if a transaction crosses a stripe boundary or if the transaction Burst size is larger than the programmed ASNI Burst split size.
- Reordering of read data and write response transactions through internal buffering
- Hard and soft QoS bandwidth regulation
- Timing isolation from the external master and the network
- Low-wire mode, where GT request and response channels are shared between reads and writes
- High-wire mode, where GT request and response channels are independent for reads and write

Configurable pipeline slices

The ASNI includes OPTIONAL pipeline slices that are enabled through build time parameters and set from the tooling. This pipeline provides flexibility in trading off latency for higher frequency.

If there is an expectation I/O timing will be tight, you can enable a pipeline slice at the AXI I/O interface for write requests and responses. Similarly, you can enable a pipeline slice at the ASNI write response interface to the internal packetized network. In addition to these pipeline slices at the I/O interfaces, the ASNI also has an OPTIONAL internal pipeline slice in the write request and response path.

If there is an expectation I/O timing will be tight, you can enable a pipeline slice at the AXI I/O interface for read requests and responses. Similarly, you can enable a pipeline slice at the ASNI interface to the internal packetized network for both the read request and response path. In addition to these pipeline slices at the I/O interfaces, the ASNI also has an OPTIONAL internal pipeline slice in the read request and response path.

2.1.2 AXI Master Network Interface

The NI-700 AMNI unit receives and processes GT packets from the NI-700 network layer.

The AMNI unit depacketizes GT packets, converts them to AXI request transactions, and forwards them to connected AXI slave devices. The AMNI unit receives AXI responses from slave devices and packetizes them into GT response flits.

The AMNI unit performs the following functions:

- Conversion between network GT requests and AXI transactions
- Routes read and write response channel traffic back to request initiators
- Burst splitting of transactions. The AMNI splits Bursts if the size of the original transaction is greater than the maximum Burst size which the AMNI can issue.
- Data width resizing
- Memory controller bandwidth regulation through **VAxQOSACCEPT**
- Timing isolation from the external slave and the network
- Low-wire mode where GT request and response channels are shared between reads and writes
- High-wire mode where GT request and response channels are independent for reads and write

Support for AXI3 interface types

You can configure an AMNI to have an AXI3 interface. However, there are several constraints that the downstream AXI3 slave must be aware of and sometimes obey to integrate with an AMNI:

- When configured as AXI3, the AMNI has a WID pin on the interface that the downstream slave can use to connect to the WID input.
- When configured as AXI3, the AMNI observes the maximum Burst length of 16 supported by AXI3.
- AXI3 locked accesses are not supported and the AMNI does not generate locked accesses.
- For write data dependency:
  - From AXI4 onwards, the AXI protocol added an extra dependency for write transactions. An AXI3 slave that accepts all write data and provides a write response before accepting the address, is not compliant with AXI4 or later versions of the AMBA AXI protocol. The AXI specification strongly recommends that any new AXI3 slave implementation includes this additional dependency.
  - The NI-700 AMNI conforms to the AXI4 dependency requirement. If a write response is received before the write address phase is accepted its behavior is UNPREDICTABLE. Downstream AXI3 slaves must conform to the AXI4 requirement to integrate directly with NI-700.
  - To integrate a downstream AXI3 slave that follows the AXI3 write dependency requirement the ASNI requires an external wrapper. The external wrapper ensures a returning write response is not provided until the slave has accepted the appropriate address.

Configurable pipeline slices

The AMNI includes OPTIONAL pipeline slices that are enabled through build time parameters and set from the tooling. This pipeline provides flexibility in trading off latency for higher frequency.

If you expect I/O timing will be tight, you can enable a pipeline slice at the AXI I/O interface for write requests and responses. Similarly, you can enable a pipeline slice at the AMNI write request and response interface to the internal packetized network. In addition to these pipeline slices at the I/O

interfaces, the AMNI has multiple OPTIONAL internal pipeline slices in the write request and response path.

If you expect I/O timing will be tight, you can enable a pipeline slice at the AXI I/O interface for read requests and responses. Similarly, you can enable a pipeline slice at the AMNI interface to the internal packetized network for the read request path. In addition to these pipeline slices at the I/O interfaces, the AMNI also has two internal pipeline slices in the read request path.

2.1.3 AHB Slave Network Interface

The NI-700 HSNI unit receives and processes requests from AHB and AHB-Lite master devices. The HSNI unit converts AHB and AHB-Lite transactions into GT packets and decodes GT read and write response packets into AHB responses.

The HSNI external interface can be configured as an AHB or AHB-Lite slave interface or an AHB or AHB-Lite mirrored master interface. The following is a short description of each type of external interface.

AHB slave interface
The AHB slave interface configures the interface with input signals **HSEL**, **HREADYOUT**, and **HREADY**.

AHB mirrored master interface
The AHB interface does not have **HSEL** or **HREADY** input signals. It is designed to connect directly to an AHB master.

The HSNI unit carries out the following functions:

- Conversion of requests, data, and response transactions between AHB and internal NOC protocol
- Address decoding
- Hazarding. If there are any outstanding writes, a new read transaction is stopped.
- If Burst promotion is enabled, the HSNI converts AHB INCR Bursts to INCR4 Bursts where possible.
- Burst splitting of incoming transactions. The HSNI splits Bursts if a transaction crosses a stripe boundary or if the transaction Burst size is larger than the programmed HSNI Burst split size.
- Early write response generation and hazarding on subsequent read requests against the writes until write response is received from downstream
- Hard and soft QoS bandwidth regulation. The HSNI has programmable registers for setting these regulators in different modes.
- Timing isolation from the external master and the network
- Low-wire mode, where GT request and response channels are shared between reads and writes
- HSNI can receive responses from master NIs which have data widths that are the same size, a smaller size, or a larger size. When HSNI receives data from a target with a smaller data width, the read data beats can arrive as fragments. HSNI collates the fragmented responses to create a data beat of a size that corresponds to its data width.
- It is possible that an HSNI receives different error responses when combining responses for individual data fragments. In this case, HSNI uses the following priority order to create the final response that is sent to the AHB master:
  1. DECERR or SLVERR (highest)
  2. OK
  3. EXOK (lowest)

HSNI does not support the following features:

- Data width of 512 bits or 1024 bits
- Locked transfers: **HMASTLOCK** signal not supported
- Multi-copy atomicity

  ——— **Note** ———

  If early write response is disabled, then HSNI does support multi-copy atomicity.

  ———————————

- Multiple slave selects (**HSELx** signaling)
- Split and retry

Note

A shareable exclusive transaction is downgraded to a Non-shareable exclusive transaction.

Configurable pipeline slices

The HSNI includes OPTIONAL pipeline slices that are enabled through build time parameters and set from the tooling. These pipeline slices provide flexibility in trading off latency for higher frequency.

At the AHB I/O interface, the address channel is registered to align with the data phase before packetization. The HSNI always contains a buffer at the interface to the internal packetized network. There are other OPTIONAL pipeline slices available to register the **HWDATA** and input to the packetizer.

On the response path, the output of the depacketizer is always registered. There are other OPTIONAL pipeline slices available at the AHB I/O interface, and the HSNI I/O interface to the internal packetized network.

2.1.4 AHB Master Network Interface

The NI-700 HMNI unit receives and processes GT packets from the network layer. The HMNI unit converts GT packets into AHB and AHB-Lite transactions and decodes AHB read and write response packets into GT packets.

The HMNI can be configured as either an AHB master interface or a mirrored AHB slave interface.

AHB master interface
This option provides all the expected AHB signals on an AHB master, so it does not have **HSEL** or **HREADY** output signals. The input AHB ready signal is named **HREADY** instead of **HREADYOUT**.

AHB mirrored slave interface
This option provides all the AHB signals for a slave, which includes **HSEL**, **HREADY** input, and **HREADY** output signals. This option then enables the direct connection of an AHB slave to the HMNI.

The HMNI unit carries out the following functions:

- Conversion of requests, data, and response transactions between AHB and GT protocol
- Transaction address decode into route vector
- Timing isolation from the external master and the network
- Low-wire mode, where GT request and response channels are shared between reads and writes
- Support for multiple incoming RPs to permit non-blocking flow control of concurrent traffic
- Burst handling of incoming WRAP and INCR Bursts
- Burst conversion and splitting to handle sparse writes and unaligned accesses
- Handling error responses from downstream slave

Configurable pipeline slices

The HMNI includes OPTIONAL pipeline slices that are enabled through build time parameters and set from the tooling. This option provides flexibility in trading off latency for higher frequency.

If you expect I/O timing to be tight, you can enable a pipeline slice at the AHB I/O interface for both the request and response path. Similarly, you can enable a pipeline slice at the HMNI request interface to the internal packetized network. The HMNI interface to the internal packetized network on the response path is always registered. In addition to these pipeline slices, the HMNI has an OPTIONAL internal pipeline slice in the request path.

2.1.5 APB Master Network Interface

The NI-700 PMNI unit receives and processes GT packets from the network layer. The PMNI unit converts GT packets into APB transactions and decodes APB read and write response packets into GT packets.

NI-700 is compliant with the APB3 and APB4 protocols.

The PMNI unit carries out the following functions:

- Size conversions from GT to a fixed data width of 32 bits
- Burst splitting to split incoming Bursts into multiple individual APB beats
- Works in low-wire mode to handle multiplexed read and write traffic on a single channel
- Supports multiple RPs to permit non-blocking flow control of concurrent traffic
- Uses address decoder to route read and write responses back to initiators
- Supports up to 16 APB interfaces on a single PMNI. Each interface can be individually specified to be APB3 or APB4. Target decoder is used to generate the APB **PSELx** signal for selecting a specific APB master interface.
- PMNI supports WriteNoSnoop and ReadNoSnoop opcodes only. All unsupported opcodes are processed as follows:

  Write requests PMNI drains write data instead of forwarding the data onto the APB bus. PMNI issues write response with an error

  Read requests PMNI forwards all zero read data beats and issues read response with an error

Configurable pipeline options

The PMNI includes OPTIONAL pipeline slices that are enabled through build time parameters and set from the tooling. The pipeline slices provide flexibility in trading off latency for higher frequency.

If there is an expectation I/O timing will be tight, you can enable a pipeline slice at the APB I/O interface for write requests and responses. Similarly, you can enable a pipeline slice at the PMNI request interface to the internal packetized network. The PMNI interface to the internal packetized network on the response path and the PMNI APB I/O interface on the request path are always registered. In addition to these pipeline slices, the PMNI has an OPTIONAL internal pipeline slice in the request path.

2.1.6 Power and Clock Domain Crossing

The NI-700 PCDC unit forms a bridge between different clock domains, power domains, or both clock and power domains. As GT flits are transferred between domains operating at different clock speeds, the PCDC synchronizes the passing flit to the new clock speed.

If your design contains multiple clock domains, power domains, or clock and power domains, the PCDC unit is used to control power and clock domain crossing.

To permit entry and exit of flits, the PCDC unit has one GT input port and one GT output port.

The PCDC unit has the following Q-Channel LPIs:

- Q-Channel for each configured power domain, permitting power domain control
- Q-Channel for each configured clock domain, permitting clock domain control

  ———— **Note** ————

  The preceding Q-Channel LPIs are combined at the NI-700 top level to provide a single Q-Channel and P-Channel per clock and power domain respectively.

  ————————————

The PCDC unit performs the following functions:

- Power and clock domain crossing
- Reordering of flits according to resource plane
- Control of power domain quiescence
- Control of clock domain quiescence

Note

The PCDC unit does not alter flits as they traverse the block.

Configuration options

You can configure the PCDC unit to meet your specific design requirements.

You can configure the following options:

- Type of synchronization that occurs within the PCDC, choose from the following options:
  - Asynchronous
  - 1:1
  - 1:N
  - M:1
- Maximum number of credits per resource plane that can be accepted at the input and output ports
- Flit width for each channel
- When configured as asynchronous:
  - Number of synchronizer register stages from 2-4
  - Buffer depth for data and credit FIFO

2.1.7 Router

The NI-700 router unit routes GT flits within the network layer of the interconnect.

The router performs the following functions:

- Transportation of GT flits between a configurable number of input ports and output ports according to the flit routing field
- Routing of flits according to *Resource Plane* (RP)

Note

If the router has more than one output port, it updates the flit routing field. Other than this update, the router unit does not alter flits as they are routed through the unit.

Configuration options

You can configure the router unit to meet your specific design requirements.

You can configure the following options:

- Number of router inputs between 1-8
- Number of router outputs between 1-8
- Channel credits can be specified for each source, destination, and RP
- Frequency of arbitration decisions that disregard QoS

2.1.8 SERDES

The NI-700 SERDES unit resizes GT flits within the network layer of the interconnect.

The SERDES unit has the following connections:

- One GT input port and one GT output port
- A threshold control input

The SERDES unit performs the following functions:

- Converting the width of flits
- Collating multiple sequential input flits into a single output flit when carrying out the upsizing function
- Splitting a single input flit into a sequence of output flits when carrying out the downsizing function
- Reordering of flits according to RP

2.1.9 Performance Monitoring Unit

The NI-700 PMU counts performance events generated by the interconnect functional units. Performance events are used to monitor various behaviors of your SoC.

The PMU is distributed across all the clock domains in NI-700. Within each clock domain, there are the following PMU components:

- Eight 32-bit software-visible event counters
- One 64-bit cycle counter (split across two 32-bit registers)
- One programmable crossbar to select a particular event for a counter to monitor
- Control network interface for programming and read access requests from NI-700 configuration memory space

The functional units within a clock domain in NI-700, such as ASNIs, can generate performance events. Generated performance events are multiplexed onto an 8-bit event bus and routed to the event counter for that clock domain.

Each event counter has shadow snapshot registers, so that all event counters can be sampled simultaneously. The event counters also have overflow functionality.

If an event or cycle counter overflows, an interrupt is triggered. This interrupt is connected to the top-level interrupt, **<CLKNAME>_nPMUINTERRUPT**. The counter that has overflowed can be determined through the PMU control and configuration registers. These registers can also be used to clear any counter overflow flags so that the interrupt can be cleared.

You can configure the functional crossbar within a component using the local event programming registers. By configuring the crossbar, you indicate an event type to forward to one of the eight available clock domain counters.

For more information, see *Chapter 4 Performance monitoring* on page 4-292.

2.2 Power, clock, and reset management

NI-700 supports a configurable number of power, voltage, and clock domains, with reset signals for each clock domain. Because NI-700 is highly configurable, it can occupy various power states and operating modes.

An external P-Channel controls each power domain and defines the power state that the power domain can enter into. An external Q-Channel connects to each clock domain, and indicates whether the clock can be externally gated.

The following clock, power, and voltage domain restrictions apply to NI-700:

- A clock domain must only be associated with a single power domain
- A power domain must only be associated with a single voltage domain
- A power domain must support one or more clock domains
- A voltage domain must support one or more power domains

This section contains the following subsections:

2.2.1 Power overview

NI-700 supports configuration of multiple power and voltage domains. Each power domain can be separately gated.

You can configure 1-32 separate power domains and 1-32 separate voltage domains within your NI-700 design. For a design with multiple power domains, each power domain exists at the same level as the other power domains within NI-700. Hierarchical power domains are not supported.

Each power domain can be separately powered down or placed into retention. An external P-Channel LPI requests changes to the power domain state through the power domain controller. The highest power state that the power domain can be in is indicated by asserting the following P-Channel **PSTATE** bits:

PACTIVE[16]
CONFIG

PACTIVE[8]
ON

PACTIVE[5]
FULL_RET

PACTIVE[0]
OFF

Power state requirements and characteristics

The P-Channel manages the transition between several power states.

The following table shows the valid and supported power states for each power domain and their requirements.

Out of reset, the PSTATE presented to NI-700 must only be one of the supported values in the following table. If not, the behavior is UNPREDICTABLE.

Table 2-1 Valid power states for each power domain and their requirements

Power mode	DEVPACTIVE bit	PSTATE [7:4]	PSTATE [3:0]
CONFIG	[16]	0b0001	0b1000
ON	[8]	0b0000	0b1000
FULL_RET	[5]	0b0000	0b0101
OFF	[0]	0b0000	0b0000

CONFIG

CONFIG has the following characteristics:

- CONFIG is the power state to enable restricted slave interface access for this power domain. In this state, only slave interfaces with their **<SLAVE_INTERFACE>_CONFIG_ACCESS** sample at reset input pins set HIGH permit ingress of external transactions.
- **PACTIVE[16]** HIGH indicates that the CONFIG power state is the lowest power state that the system must be in. For example, this scenario can occur when only a transaction from a CONFIG defined interface requires access to fully powered logic. When **PACTIVE[16]** is LOW, it is necessary to check **PACTIVE[8]** to determine the required power state. In this situation, PACTIVE[8] determines whether the system must transition to ON or if it is possible to enter FULL_RET or OFF to save power.
- State transitions from CONFIG to ON, FULL_RET, or OFF are permissible, as determined by the lowest PACTIVE bit that is HIGH.

ON

ON has the following characteristics:

- ON is the fully powered state for all logic in the power domain.
- Power domain must be in ON for all interfaces to progress. **PACTIVE[8]** HIGH indicates that ON is required, for example, when a transaction requires access to fully powered logic. When **PACTIVE[8]** is LOW, it might be possible to transition to the FULL_RET state to save power.
- State transitions from ON to CONFIG, FULL_RET, or OFF are permissible, as determined by the lowest PACTIVE bit that is HIGH. However, we recommend only transitioning to CONFIG if system reconfiguration is required. Otherwise, a transition to OFF is recommended.

FULL_RET

FULL_RET has the following characteristics:

- FULL_RET is the static retention state for all logic instances within the power domain.
- In FULL_RET, all external flow control signals are held in a state that prevents any transaction propagation.
- The only permitted state transitions from FULL_RET are to ON or CONFIG.

OFF

OFF has the following characteristics:

- OFF is the fully off state for the controlled power domain.
- In OFF, all external flow control signals are held in a state that prevents any transaction propagation.
- The only permitted state transitions from OFF are to ON or CONFIG.

P-Channel Low-Power Interface

Each power domain in NI-700 is connected to a standard LPI P-Channel that communicates external power state information.

The P-Channel that is connected to each power domain determines whether the interconnect can be powered off or placed into retention.

The **PACTIVE** signal indicates the permitted highest power state of the power domain.

Each P-Channel LPI must have an associated NUM_SYNC_STAGES parameter specified. This parameter indicates the number of clock cycles that are required for synchronization.

P-Channel signals

The P-Channel uses signals to communicate information about the external power state and indicate the power state into which NI-700 is required to transition.

The following table shows the P-Channel LPI signals.

Table 2-2 P-Channel LPI signals

Name	Direction	Width	Purpose
PACTIVE	Output	17	Vector indicator of power states NI-700 is eligible to enter
PSTATE	Input	8	Binary value of power state into which external controller requires NI-700 to transition
PREQ	Input	1	Request signal to initiate power state transition
PACCEPT	Output	1	Handshake signal to indicate that power state transition is complete
PDENY	Output	1	Handshake signal to indicate that power state transition cannot be completed

2.2.2 Clocks overview

To improve power and performance of your design, NI-700 provides configurable clock domains and supports clock gating.

You can configure 1-32 separate clock domains within your NI-700 design. NI-700 supports hierarchical clock gating, which means that each of the configured clock domains can be separately gated.

Each clock domain has a single clock pin input, which is known as **<CLKNAME>_CLK**.

Levels of clock gating

NI-700 contains several clock types that are arranged in a hierarchy, from the clock supplying a clock domain through to local clocks that the RTL creates.

NI-700 contains the following clock types:

Top-level clock
The clock input to the clock domain **<CLKNAME>_CLK**.

Regional clocks
Created as an output of regional clock gaters that include a coarse enable for coarse-grained clock gating under idle or mostly idle conditions. Regional clock gaters can shut down the clock network between regional and local gaters. Therefore, this level of hierarchy enables greater power reduction than is possible using local clock gating. The regional clock gaters are instantiated in and controlled by NI-700 RTL.

Local clocks
Created according to the following hierarchy:

1. RTL creates fine grained enable signals
2. Fine grained enable signals control local clock gaters
3. Local clock gaters output local clock signals

Local clock signals are used to directly clock sequential elements in the NI-700. The exact set of local clocks is internal to NI-700 and is not described in this book.

Hierarchical clock gating

During low activity scenarios, the system can use hierarchical clock gating to transition to a low-power state. This transition permits the system to save power that the active clock tree normally consumes.

Control over individual clock domains permits flexible system design and therefore flexible power state design.

NI-700 supports hierarchical clock gating. This feature enables an external clock controller to use the Q-Channel LPI to individually request clock domains to be gated in the interconnect.

The interconnect blocks new transactions from entering the interconnect when there are no outstanding transactions within the clock domain. The domain acknowledges that this process is complete and the clock controller can remove the clock.

Hierarchical clock gating can gate the following regions:
- Slave *Network Interfaces* (NIs), for example ASNIs
- Master NIs, for example AMNIs
- Routers
- PCDC block
- SERDES block
- Register block

The Q-Channel LPI enables hierarchical clock gating by communicating with the clock domain controller to request that the clock domain becomes quiescent.

Q-Channel Low-Power Interface

Each clock domain in NI-700 is connected to a standard Q-Channel *Low-Power Interface* (LPI) that gates the clock domain.

A Q-Channel is present for each clock domain.

Note

Hierarchical clock gating is always present in the NI-700 configuration.

Q-Channel signals

The Q-Channel LPI contains low-power signals to control hierarchical clock gating in the NI-700.

The following table shows the Q-Channel LPI signals.

Table 2-3 Q-Channel LPI signals

Signal	Direction	Source, destination	Description
QACTIVE	Output, input	Interconnect, controller	Interconnect active
QREQn	Output, input	Controller, interconnect	System low-power request
QACKn	Output, input	Interconnect, controller	Low-power request acknowledgment
QDENY	Output, input	Interconnect, controller	Negative acknowledgment after receiving a **QREQn** assertion, indicating the NI-700 has refused the request from the controller to prepare to stop the clocks

For additional information on the function of these signals, see *AMBA® Low Power Interface Specification Arm® Q-Channel and P-Channel Interfaces*.

External Clock Controller

The *External Clock Controller* (ExtCC) controls the clock gating flow.

The following figure shows an example of how the ExtCC controls the clock gating flow.

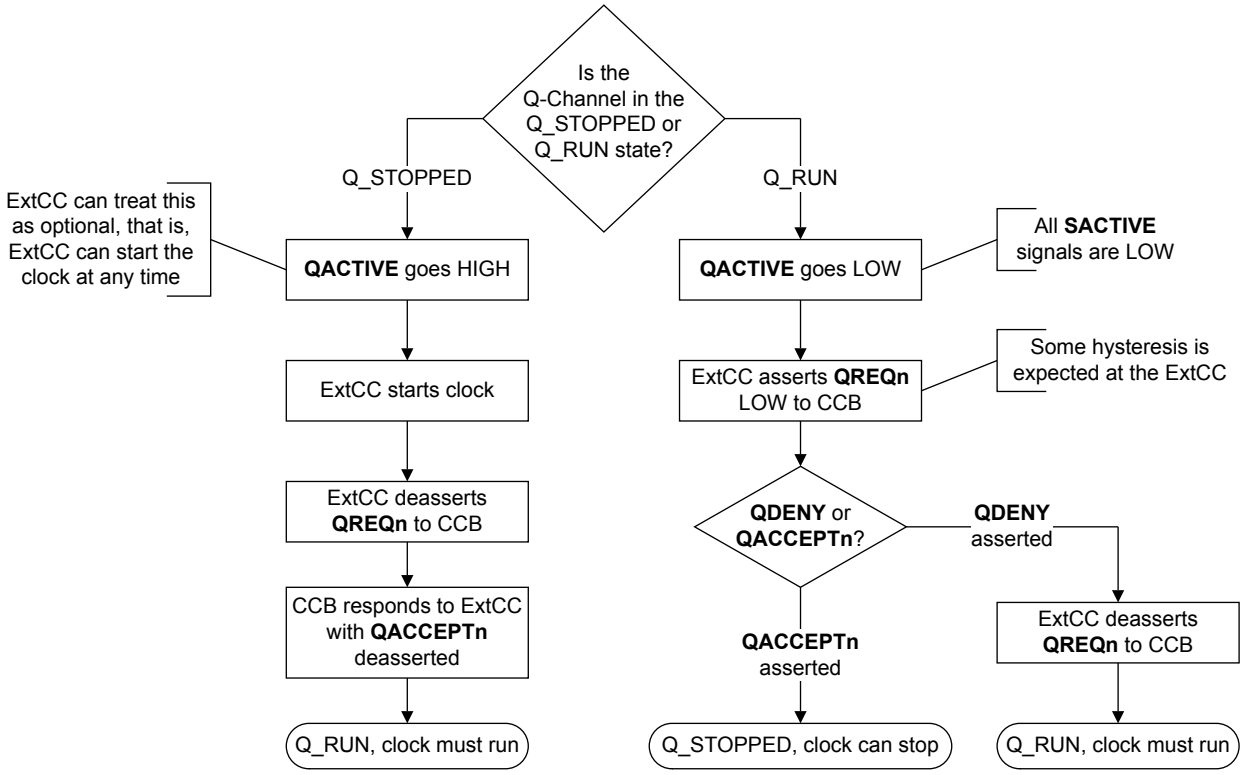

Figure 2-1 Clock gating control using ExtCC

This example clock gating sequence begins and ends with the Q-Channel in either of the following states:

Q_STOPPED
Quiescent state, where **QREQn** and **QACCEPTn** are asserted.

Q_RUN
Active state, where **QREQn** and **QACCEPTn** are deasserted.erna.

The following requirements apply to the ExtCC:

- It must supply a clock to the NI-700 when the Q-Channel is in any state other than Q_STOPPED.
- The ExtCC can either:
  - Choose to gate the clock to NI-700 when the Q-Channel is in the Q_STOPPED state
  - Choose to run the clock at any time
- The ExtCC is responsible for bringing the Q-Channel to Q_RUN state after reset deassertion.
- This manual does not describe the exact behavior of the ExtCC and its usage of **QREQn** in response to **QACTIVE** deassertion. However, the design of the ExtCC is likely to include a control loop with some hysteresis so that *High-level Clock Gating* (HCG) is enabled when the system is inactive for long periods. HCG is not enabled for short periods of inactivity. If the clocks are stopped in response to short periods of inactivity, performance of the NI-700 can be negatively affected.
- It is the responsibility of the SoC designer to fully control the clock management Q-Channel. If there is a requirement for a control or configuration bit to completely enable or disable HCG functionality, that register or bit must exist outside of NI-700. More specifically, NI-700 has no internal means of disabling HCG.

Clock domain wakeup

Wakeup signals are present on the master device side of the ASNIs and the slave device side of the AMNIs. These signals indicate incoming or outgoing network traffic, so that the relevant system components are woken and available to receive traffic.

NI-700 requires upstream masters to support **AWAKEUP** when connecting to it. Similarly, NI-700 drives **AWAKEUP** from its AXI master interfaces. AXI4 masters can connect to NI-700 as long as it supports **AWAKEUP**.

Each ASNI has an input signal, **AWAKEUP**, which must be asserted when the AXI or ACE-Lite **AxVALID** signal is HIGH. **AWAKEUP** must remain asserted until the associated **ARVALID**, **ARREADY** handshake, or the **AWVALID**, **AWREADY** handshake completes. When the address handshake is completed, NI-700 keeps the clock active until the transaction completes. When **AWAKEUP** is asserted, NI-700 drives the **QACTIVE** signal of the corresponding clock domain HIGH, to request activation of the clock signal.

2.2.3 Power control

The NI-700 power control network consists of *Power Control* (PC) blocks, *Clock Control* (CC) blocks, and several power control signals.

The NI-700 power controller for any power domain must be in a *Relatively Always ON* (RAON) power domain with regards to its corresponding power domain. This requirement enables the internal wakeup signal and the **PACTIVE** signal on the external P-Channel to be asserted as they are in the *_RAON power domain. When asserted, they indicate that the corresponding power domain must be turned ON.

In the following diagram, P1_RAON and P1; and P2_RAON and P2 are corresponding power domains. For example, if P2_RAON asserts the external **PACTIVE** signal, then the SoC power controller is expected to turn on the power for the corresponding P2 power domain. Similarly, before the SoC power controller requests and the power state transition through the *_AON power controller, it must also ensure that the corresponding power domain either P1 or P2, is already powered ON.

The clock and reset to the power controller comes from the clock controller in the corresponding power domain. For example, P1 possibly contains multiple clock domains. However, the power controller in P1_RAON is considered to be in the same clock domain as one of the clock domains in P1. The clock and reset to the P1_RAON power controller therefore comes from its corresponding clock controller in the same clock domain in P1. Before the corresponding P1 or P2 power domain powers down, the signals crossing between the power domain and its corresponding P1_RAON or P2_RAON power domain are all isolated.

The following figure shows the NI-700 power control network.

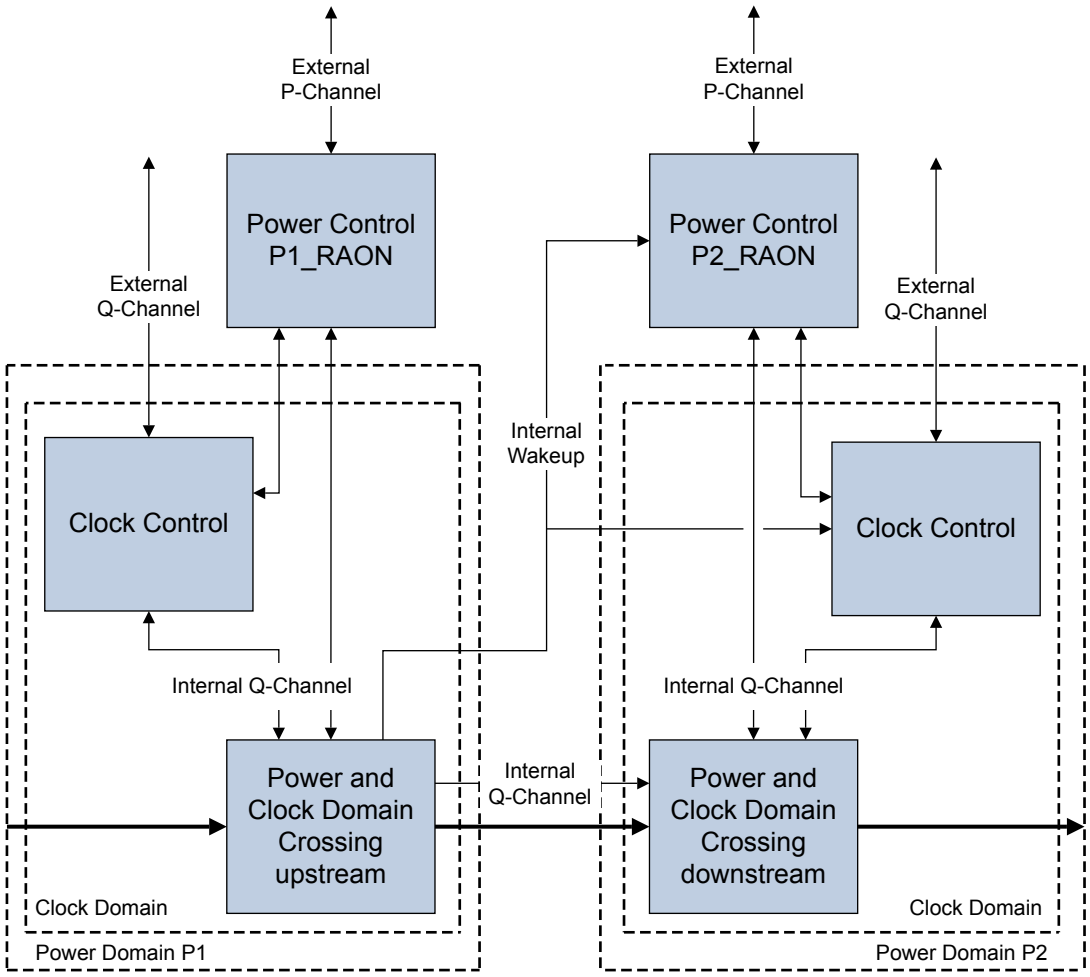

Figure 2-2 Power control network

Power control sequences

To permit downstream power domains to transition between power states, the NI-700 power control network must perform specific sequences of actions.

Upstream power domain ON, downstream power domain ON→OFF

The following sequence describes how a downstream power domain transitions from ON→OFF when the upstream power domain is ON:

1. The downstream external **PACTIVE[16:1]** signal is driven LOW, indicating that all activity within the power domain is complete.
2. External P-Channel requests power domain to enter P_OFF state.
3. Internal power **QREQn** signal, targeting PCDC, goes LOW.
4. If there is no activity in the PCDC:
   a. PCDC performs logical isolation of boundary and indicates to upstream PCDC that it wants to enter P_OFF state.
   b. Upstream PCDC acknowledges P_OFF state request from downstream PCDC, performs logical isolation, and resets PCDC FIFO pointers to reset value.
   c. Downstream receives acknowledgment, resets PCDC FIFO pointers to reset value, and issues **QACCEPT** to PC.
   d. PC issues P-Channel accept to external interface.

5. The external clock controller must request all clock Q-Channels to enter Q_STOPPED.
6. When all P-Channels and Q-Channels are in P_OFF or Q_STOPPED, all power domain pins can be physically isolated.

   ———— **Note** ————

   In NI-700, all isolation values are LOW.

   ————————————

Upstream power domain ON, downstream power domain OFF→ON

The following sequence describes how a downstream power domain transitions from OFF→ON when the upstream power domain is ON:

1. New upstream transaction arrives in CDC.
2. Upstream PCDC (in RAON domain) asserts internal wakeup signal to downstream PC.
3. Downstream PC asserts external higher power state **PACTIVE** asynchronously.
4. External power control must complete the following actions:
   a. Restore power to domain
   b. Apply resets to domain
   c. Remove physical isolation
   d. Remove resets to domain
5. External P-Channel can now request to enter P_ON and clock Q-Channel can request to enter Q_ON.
6. Internal **QREQn** signal, targeting PCDC, goes HIGH.
   a. PCDC removes logical isolation of boundary and issues **QACCEPTn** transition to CC and PC.
   b. PC and CC forward **QACCEPTn** transition to external interface.
   c. Downstream PCDC indicates to upstream PCDC that power is restored and is in the P_ON state.
   d. Upstream PCDC acknowledges and removes logical isolation.

External power domain boundaries

NI-700 uses external power domain boundaries to enable attached devices to switch power state independently of the interconnect. Certain configurations and assumptions apply to this feature.

NI-700 provides power isolation on AXI signals at the boundary of the interconnect and integrated IP. This feature can be used when attached IP is in a switchable power domain, and the interconnect must be in a RAON power domain. For example, power isolation can be applied at the interconnect boundary between an AMNI and its attached AXI slave device, as shown in the following figure:

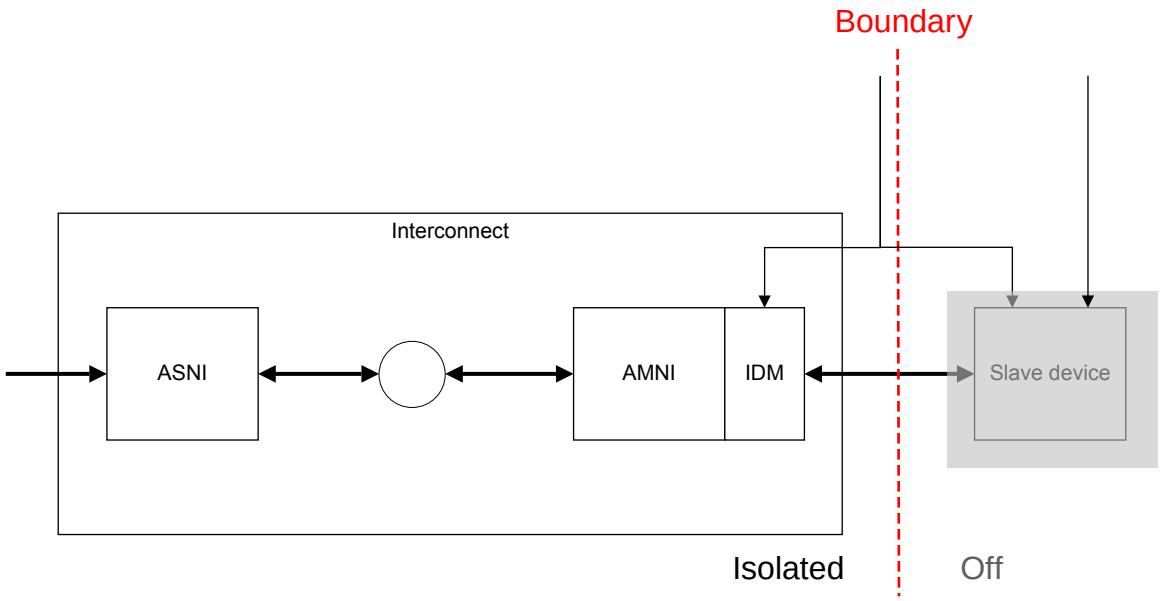

Figure 2-3 NI-700 external power domain boundary

When applying this feature, the interconnect must use IDM isolation to prevent cross boundary accesses. For more information, see *2.5.5 IDM access control* on page 2-70. The clock domain crossing is within the IP block.

Note

Arm assumes that the same clock feeds the interconnect NI and the IP interface.

The interconnect is powered up, IDM is in the isolate state, and the attached device is off. At this point, software can still access enumeration values in IDM registers. For more information, see *2.5.1 IDM and device discovery* on page 2-67.

External power domain powerup sequence

A specific sequence of events must occur in the system to power up a device in an external power domain.

The system uses the following sequence to power up an external power domain:

1. System applies power to the IP domain
2. System removes isolation cell clamp values on the AXI boundary
3. System applies IP reset sequence, either through a full system reset, or an IDM soft reset
4. System releases IDM isolation
5. Configuration or mission access to IP occurs

For more information about IDM soft reset, see *2.5.4 IDM soft reset* on page 2-68.

External power domain power down sequence

A specific sequence of events must occur in the system to power down a device in an external power domain.

The system uses the following sequence to power down an external power domain:

1. IDM is placed into the isolation state.
2. System applies isolation cell clamp values on the IP boundary.
3. System removes the power to the IP power domain.

AHB address phase buffering in HSNI

Extra buffering logic and signals are present in the NI-700 HSNI to enable AHB address phase sampling when the unit is clock gated.

In the AHB protocol, a slave cannot request that the address phase of a transaction is extended. Therefore all HSNIs must be able to sample the address phase, even when clock gated. The HSNI block adds an extra buffer stage to accept the address phase of a transaction when clock gated.

The following figure shows the HSNI buffer mechanism.

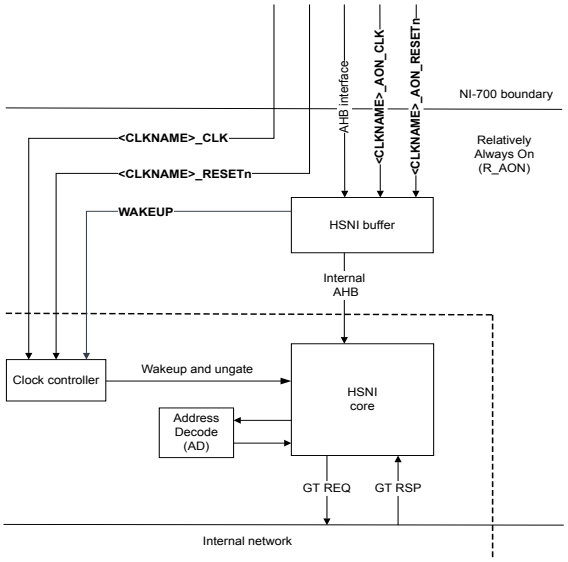

Figure 2-4 HSNI clock gating buffer mechanism

The standard **<CLKNAME>_CLK** and **<CLKNAME>_RESETn** signals behave normally and connect to the clock and power architecture. These signals must follow the same rules that are described in *External Clock Controller* on page 2-43. As such, you can only remove the clock input when the Q-Channel is in the Q_STOPPED state.

NI-700 adds extra **<CLKNAME>_AON_CLK** and **<CLKNAME>_AON_RESETn** signals for the buffer stage. These signals must be on before an initial transaction ingresses into the device. If the network does not follow this constraint, the transaction is lost. The wakeup signal is routed to the clock controller of the respective clock domain. The clock controller can then wake up and ungate the core component so that the core component can start to accept transactions.

The clock for the HSNI buffer and the HSNI core must be driven from the same source clock. There is no synchronization and they are assumed to be in the same clock domain. If not then transactions are likely to be dropped.

The HSNI buffer and the HSNI core must be in the same power domain. When the HSNI buffer and the HSNI core are in the same power domain, there is improved power saving. When the AHB master and HSNI buffer are powered OFF, the HSNI core is also powered OFF which results in a power saving.

2.2.4 Clock and reset control

The NI-700 clock and reset control network consists of CC blocks and several clock control signals.

NI-700 contains an external Q-Channel and reset signal per clock domain. When the Q-Channel is in the Q_STOPPED state, there is logical isolation between clock domains. When the Q-Channel is in this state, all transactions are stalled at the domain boundary.

Clock domains exit reset in Q_STOPPED state when they are logically isolated. Therefore requests cannot be lost. The full Q-Channel sequence to go from Q_STOPPED to Q_RUN must be completed

before requests can enter this clock domain. All clock domains within a single power domain must be reset together.

The following figure shows an example clock and reset control network within the interconnect.

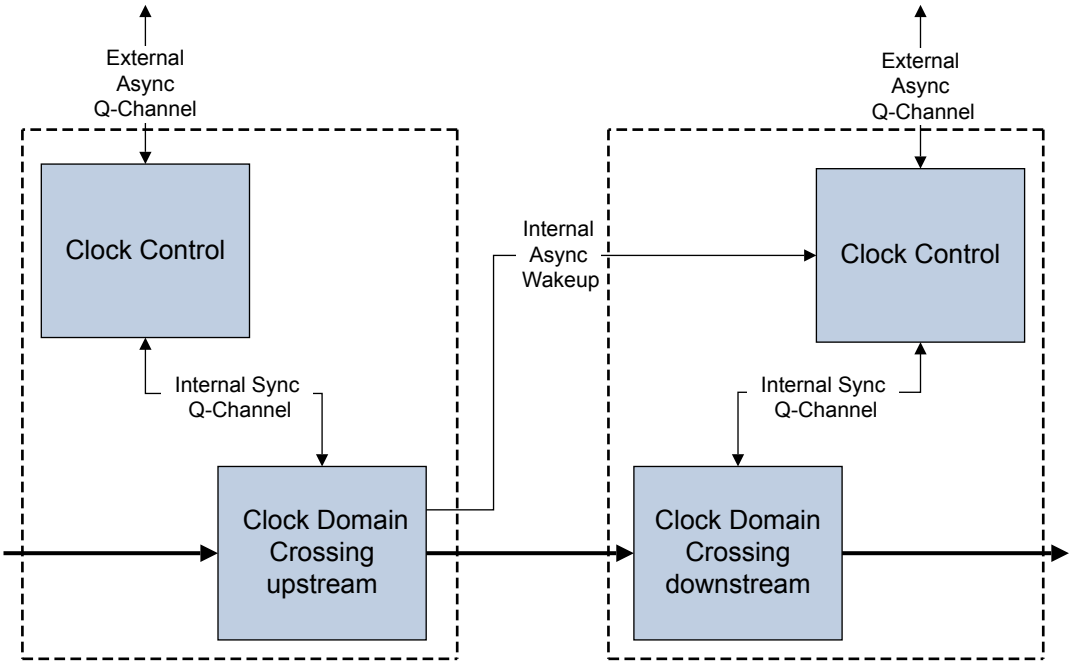

Figure 2-5 Clock and reset control network

Clock control sequences

To permit downstream clock domains to transition between states, the NI-700 clock control network must perform specific sequences of actions.

Upstream clock domain ON, downstream clock domain ON→OFF

The following sequence describes how a downstream clock domain transitions from ON→OFF when the upstream clock domain is ON:

1. Downstream external **QACTIVE** signal is driven LOW, indicating that all activity within the clock domain is complete.
2. External **QREQn** signal goes LOW
3. Internal **QREQn** signal to CDC goes LOW
4. If there is no activity within CDC:
   a. CDC performs logical isolation of boundary and issues **QACCEPTn** to CC
   b. CC forwards **QACCEPTn** to external interface
   c. Clock can be gated externally
5. If there is activity when internal **QREQn** is received:
   a. CDC asserts internal **QACTIVE**
   b. CDC issues internal **QDENY**
   c. The top-level Q-Channel sends an external **QDENY** handshake.
   d. The external clock controller must complete the Q-Channel **QDENY** by reasserting **QREQn**.

Upstream clock domain ON, downstream clock domain OFF→ON

The following sequence describes how a downstream clock domain transitions from OFF→ON when the upstream clock domain is ON:

1. New upstream transaction arrives in CDC
2. Upstream CDC asserts internal wakeup to downstream CC

3. Downstream CC asserts external **QACTIVE** asynchronously
4. Clock signal is restored externally to downstream clock domain
5. External **QREQn** signal goes HIGH
6. Internal **QREQn** signal to CDC goes HIGH
   a. CDC removes logical isolation of clock domain boundary and issues **QACCEPTn** transition to CC.
   b. CC forwards **QACCEPTn** transition to external interface.

Note

NI-700 does not deny requests to a higher clock state, for example OFF→ON.

Reset control sequences

Specific sequences of actions must occur to permit clock domains to exit from the reset state. The sequence differs depending on whether the upstream or downstream clock domain exits resets first.

The following figure shows the logical isolation between clock domains in reset within an example clock and reset control network.

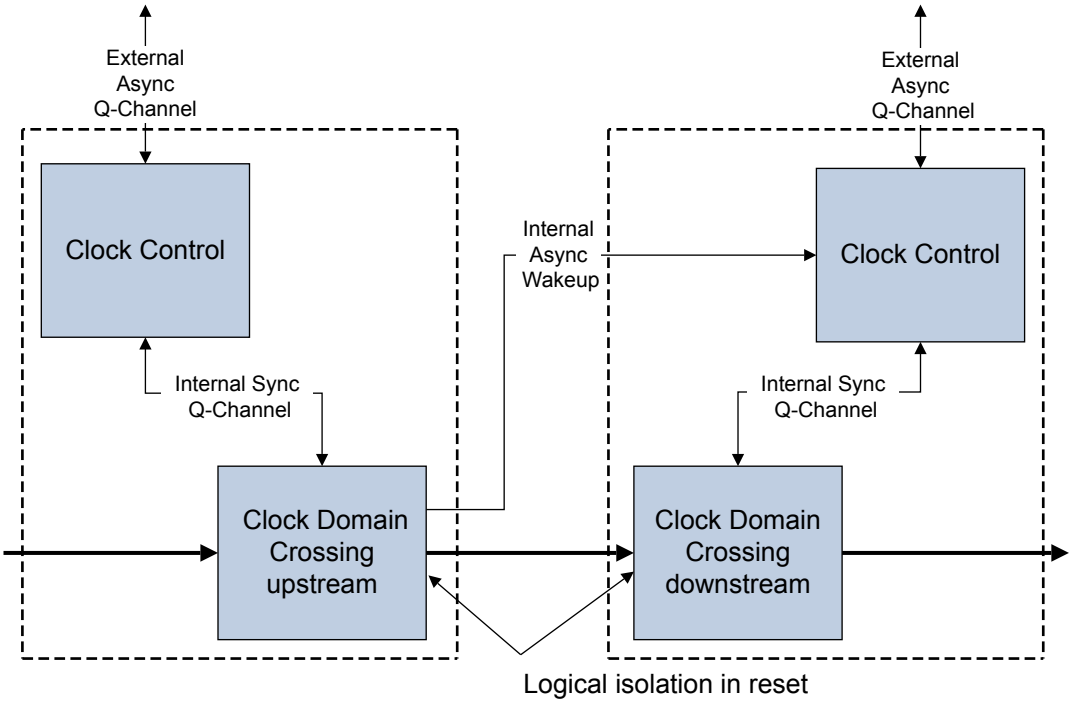

Figure 2-6 Logical isolation between clock domains in reset

Both domains in reset state, upstream exits reset first

The following sequence describes how an upstream clock domain transitions out of reset when both clock domains are in reset:

1. Upstream clock domain completes clock and power handshake to permit operation.
2. New transaction arrives at the upstream CDC.
3. As the downstream clock domain is in reset (Q_STOPPED state), it now follows the same flow as *Upstream clock domain ON, downstream clock domain OFF→ON* on page 2-50, although the downstream clock domain must first exit reset.

Both domains in reset state, downstream exits reset first

The following sequence describes how a downstream clock domain transitions out of reset when both clock domains are in reset:

1. Downstream clock domain completes clock and power handshake to permit operation.
2. Upstream clock domain does not issue transactions until it is out of reset. The downstream clock domain now works as if in normal operation. The downstream clock domain awaits transactions, which can be forwarded after the upstream clock domain exits reset and completes the external clock and power handshake.

2.3 Node ID mapping and discovery

Discovery is a software algorithm that is used to discover the configuration of NI-700.

Each NI-700 node has a corresponding node type value and node ID value. Software uses the discovery mechanism to discover the pointer to the 4KB register programming region for each node. You can identify each node by its node type and node ID. The discovery process also permits software to capture more information about the node configuration. The following information is captured for all the node types:

- Global CFG (NI-700 base)
- Voltage domain
- Power domain
- Clock domain
- Slave interface nodes
- Master interface nodes
- PMU
- Interface ID

NI-700 assigns a unique node ID to all the slave interface nodes. Similarly, all the master interface nodes are assigned a unique node ID. The node ID space of the slave and master interface nodes can overlap but the corresponding node type value distinguishes the nodes.

The node type values that are assigned to each NI-700 node are shown in the following table.

Table 2-4 Node type values

Node	Node type value
NI-700 base	`0x0000`
Voltage domain	`0x0001`
Power domain	`0x0002`
Clock domain	`0x0003`
ASNI	`0x0004`
AMNI	`0x0005`
PMU	`0x0006`
HSNI	`0x0007`
HMNI	`0x0008`
PMNI	`0x0009`

This section contains the following subsections:

2.3.1 Access mechanism

The programming network in the NI-700 follows the distributed nature of the components in the interconnect. Each NI-700 block has programmable registers that software can access. Software can

discover the number and type of these configurable blocks, their attributes, and software accesses these registers for configuration.

The software can discover the system at runtime using a single PERIPHBASE address. All registers are organized into multiple register blocks, referred to as nodes. A node is often associated with either a logical domain or unit within the design, such as any of the following:

- Voltage domain
- Power domain
- Clock domain
- ASNI
- AMNI
- PMU
- HSNI
- HMNI
- PMNI

If a node is not associated with a logical unit, it contains pointers to one or more child nodes within the same logical unit or domain.

If a node contains zero child nodes, it is considered to be a leaf node containing only unit-specific registers. If a node contains one or more child nodes, it contains registers that are local to that node. These nodes can be power domains, alongside registers that contain information indicating the number of child nodes, and a pointer to the start address offset of each child node from PERIPHBASE.

2.3.2 Node configuration register address-mapping overview

All the NI-700 configuration registers are mapped to an address range starting at PERIPHBASE. You can use the Socrates IP tooling platform to define the reset value of PERIPHBASE. All configuration, information, and status registers in a NI-700 interconnect are grouped into 4KB regions, and each is associated with a NI-700 component instance.

The base address of each region can be determined at compile time, or determined at runtime through a software discovery mechanism. Software discovery consists of the following steps:

- Read information in the first 4KB region at PERIPHBASE. This information determines the:
  - Number of voltage domains in NI-700.
  - Offset from PERIPHBASE for each 4KB voltage domain address region.
- Read information in the region that is associated with each voltage domain. This information determines the:
  - Power domains that are associated with that voltage domain.
  - Topology information for those components.
  - Offset from PERIPHBASE for the 4KB base region of each power domain.
- Read information in the region that is associated with each power domain. This information determines the:
  - Clock domains that are associated with that power domain.
  - Topology information for those components.
  - Offset from PERIPHBASE for the 4KB base region of each clock domain.
- Read information in the region that is associated with each clock domain. This information determines the:
  - Components that are associated with that clock domain.
  - Topology information for those components.
  - Offset from PERIPHBASE for the 4KB base region of each component.
- Read information in the 4KB region that is associated with the component. This information determines the:
  - Type of block.
  - Configuration details of the component.

With this sequence, software can build a list of all components in the system, and the addresses of their respective 4KB configuration region.

The following figure shows the access mechanism.

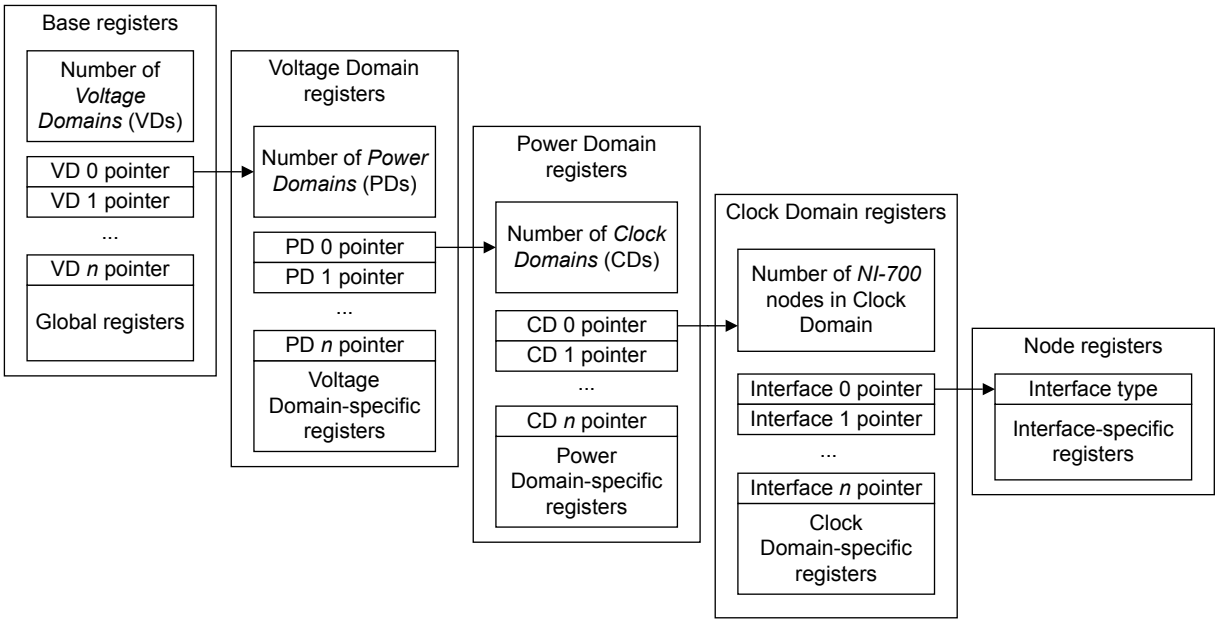

Figure 2-7 Access mechanism

The major node types are as follows:

Base node Describes the number of voltage domains, pointers to voltage domain registers, and global interconnect registers.

Voltage domain Indicates the number of power domains in a voltage domain, and any voltage domain-specific control registers.

Power domain Indicates the number of clock domains in a power domain, and any power domain-specific control registers.

Clock domain Indicates the number of leaf nodes in a clock domain, and any clock domain-specific control registers.

Leaf node Indicates the type of leaf node. In NI-700, the leaf node can be PMU, ASNI, AMNI, HSNI, HMNI, PMNI, and others.

At the end of the discovery process, software builds a discovery tree that provides:

- All pointers to the 4KB register regions corresponding to the voltage domains.
- For each voltage domain identified by a Voltage Domain ID:
  - All pointers to the 4KB register regions corresponding to the power domains.
  - For each power domain identified by the Power Domain ID:
    - All pointers to the 4KB register regions corresponding to the clock domains.
    - For each clock domain identified by the Clock Domain ID:
      - All pointers to the 4KB register regions corresponding to all the leaf nodes. The leaf nodes are the slave interface nodes, master interface nodes, and the PMU node in that clock domain.
      - For each node, the information that is required to discover its Node Type, NodeID, and node information.

2.3.3 Global configuration register region

The first 4KB block above PERIPHBASE contains global information and configuration for NI-700. It also contains the first level of discovery information for components in the system.

The following table highlights the register structure of this lowest 4KB block. For complete register descriptions, see *3.1 About the programmers model* on page 3-116.

Table 2-5 NI-700 ID registers

Offset	Contents
0x0	NI-700 Global node type register.
NI-700 voltage domain configuration mapping	
0x4	Number of Voltage Domain regions present.
0x8	Voltage Domain 0 Base Address, offset from PERIPHBASE.
0xC	Voltage Domain 1 Base Address, offset from PERIPHBASE.
0x10	Voltage Domain 2 Base Address, offset from PERIPHBASE.
…	(Voltage Domain[N:3]), where N is the total number of voltage domains in NI-700.
NI-700 global configuration	
0x00FD0	Peripheral ID4.
0x00FD4	Peripheral ID5.
…	Extra global configuration registers.

Each voltage domain base address register in the table contains the offset from PERIPHBASE, for a 4KB region, and contains the following:

- Information about one voltage domain.
- Discovery information for components that are associated with that voltage domain.

2.3.4 Voltage domain configuration register region

Each voltage domain uses a 4KB configuration register region that contains information about that voltage domain, and offset addresses for all associated power domains.

The following table highlights the register structure of the voltage domain configuration register region.

Table 2-6 Contents of the voltage domain configuration register region

Offset	Contents
Voltage domain ID register	
0x0	Voltage domain ID register.
NI-700 power domain configuration mapping	
0x4	Number of power domain regions present.
0x8	Power domain 0, within voltage domain, base address, offset from PERIPHBASE.
0xC	Power domain 1, within voltage domain, base address, offset from PERIPHBASE.
0x10	Power domain 2, within voltage domain, base address, offset from PERIPHBASE.
…	(Power domain[N:3]), where N is the total number of power domains in this voltage domain.
NI-700 power domain configuration	

Each power domain base address register in the table contains the offset from PERIPHBASE for a 4KB region that contains:

- Information about one power domain.
- Discovery information for clock domains that are associated with that power domain.

2.3.5 Power domain configuration register region

Each power domain contains a 4KB configuration register region that contains information about that power domain, and all associated clock domains.

The following table highlights the register structure of the power domain configuration register region.

Table 2-7 Contents of power domain configuration register region

Offset	Contents
Power domain ID register	
0x0	Power domain ID register.
NI-700 power domain configuration mapping	
0x4	Number of clock domain regions present.
0x8	Clock domain 0, within power domain, base address, offset from PERIPHBASE.
0xC	Clock domain 1, within power domain, base address, offset from PERIPHBASE.
0x10	Clock domain 2, within power domain, base address, offset from PERIPHBASE.
…	(Clock domain[N:3]), where N is the total number of clock domains in this power domain.
NI-700 power domain configuration	
…	Extra configuration registers, as required.

Each clock domain base address register in the table contains the offset from PERIPHBASE, for a 4KB region, that contains:

- Information about one clock domain.
- Discovery information for leaf nodes that are associated with that clock domain.

2.3.6 Clock domain configuration register region

Each clock domain contains a 4KB configuration register region that contains information about that clock domain, and all associated components.

Clock domain configuration register region

The following table highlights the register structure of the clock domain configuration register region.

Table 2-8 Contents of clock domain configuration register region

Offset	Contents
Clock domain ID Register	
0x0	Clock domain ID register.
NI-700 clock domain configuration mapping.	
0x4	Number of components present.
0x8	Component 0, within clock domain, base address, offset from PERIPHBASE.
0xC	Component 1, within clock domain, base address, offset from PERIPHBASE.
0x10	Component 2, within clock domain, base address, offset from PERIPHBASE.
…	(Component[N:3]) where N is the total number of components in this clock domain.
NI-700 clock domain configuration	
…	Extra configuration registers, as required.

Each component base address register in the table contains the offset from PERIPHBASE for a 4KB region that contains information about one component node.

2.3.7 Interface ID

Each external interface contains a unique ID called InterfaceID.

An interface ID routes packets to its external interface destination or CFGNI (*Configuration Node*). The interface ID is therefore the SRCID and TGTID for a packet within an NI-700.

Interface ID values are assigned in two unique pools:
- One for slave network interfaces (xSNI)
- One for target network interfaces (xMNI)

Note

The interface ID diagram below is for illustrative purposes only.

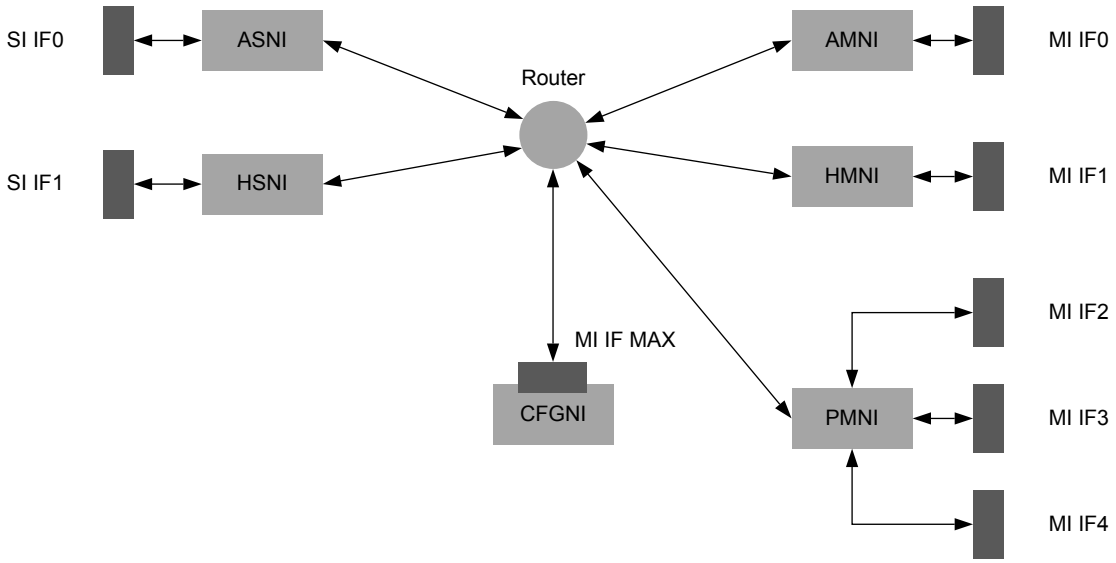

Figure 2-8 Interface ID overview

2.3.8 Configuration register address region calculation

When configuring NI-700, you must specify the size of the address region, as the size depends on your design.

Each *Configuration Node* (CFGNI) occupies 4KB of the address map. The final number of configuration nodes in your design depends on the following:

- Number of voltage domains.
- Number of power domains.
- Number of clock domains.
- Number of endpoints (which is the sum of the number of ASNIs, AMNIs, HSNIs, HMNIs, and PMNIs in your design).
- Number of PMUs.

  Note

  The NI-700 design contains one PMU per clock domain, so the number of PMUs is equivalent to the number of clock domains in your design.

To calculate the size of the configuration register address region, use the following equation.

Config space (in KB) = 4 x (1 + V + P + 2C + E), where:

- V = Number of voltage domains.
- P = Number of power domains.
- C = Number of clock domains.
- E = Number of endpoints.

Note

Regardless of the configuration, the programmers view always has one CFG Node containing the global registers. The global CFG Node is accounted for in the equation.

Configuration address space example for design with multiple voltage, power, and clock domains

NI-700 contains multiple voltage, power, and clock domains that are configurable. The number of domains in your design affects the size of the configuration address space and the layout of the NI-700 programmers view.

The configurable topology of NI-700 alters the programmers view by changing the number of *Configuration Nodes* (CFG Nodes) required. To illustrate how the configurable design of NI-700 affects the programmers view, consider an example configuration, which contains:

- Two voltage domains
- Four power domains
- Eight clock domains
- Eight ASNIs
- Seven AMNIs
- Three HSNIs
- Three HMNIs
- Three PMNIs

The following table shows the programmers view for the example configuration.

Table 2-9 Example programmers view for multiple voltage, power, and clock domain NI-700 configuration

Offset	Contents
0	Global registers
4KB	Voltage domain 0 registers
8KB	Power domain 0 registers
12KB	Clock domain 0 registers
16KB	ASNI 0 registers
20KB	AMNI 0 registers
24KB	Clock domain 1 registers
28KB	ASNI 1 registers
32KB	AMNI 1 registers
36KB	Power domain 1 registers
40KB	Clock domain 2 registers
44KB	ASNI 2 registers
48KB	AMNI 2 registers
52KB	HSNI 0 registers

Table 2-9 Example programmers view for multiple voltage, power, and clock domain NI-700 configuration (continued)

Offset	Contents
56KB	HMNI 0 registers
60KB	PMNI 0 registers
64KB	Clock domain 3 registers
68KB	ASNI 3 registers
72KB	AMNI 3 registers
76KB	Voltage domain 1 registers
80KB	Power domain 2 registers
84KB	Clock domain 4 registers
88KB	ASNI 4 registers
92KB	AMNI 4 registers
96KB	HSNI 1 registers
100KB	HMNI 1 registers
104KB	PMNI 1 registers
108KB	Clock domain 5 registers
112KB	ASNI 5 registers
116KB	AMNI 5 registers
120KB	Power domain 3 registers
124KB	Clock domain 6 registers
128KB	ASNI 6 registers
132KB	AMNI 6 registers
136KB	Clock domain 7 registers
140KB	ASNI 7 registers
144KB	HSNI 2 registers
148KB	HMNI 2 registers
152KB	PMNI 2 registers

Each node type within the NI-700 requires a unique ID to enable device discovery to determine the set of registers at each 4KB region.

2.4 Address decode and mapping

When an AXI master device generates a request, the connected ASNI receives the transaction address through the request channel. The ASNI decodes the address and calculates the target ID for that address region.

The ASNI address decoders are generated when you configure the ASNI through the Socrates IP tooling platform. Separate address decoders exist in the ASNI for the read and write request channel, enabling parallel lookup.

If an address pointing to an unmapped region of memory is presented to the address decoder, an address DECERR response is generated.

This section contains the following subsections:

2.4.1 ASNI address decode

When an AXI master device generates a request, the connected ASNI receives the transaction address through the request channel. The ASNI decodes the address and calculates the target ID for that address region.

The ASNI address decoders are generated when you configure the ASNI through the Socrates IP Tooling platform. Separate address decoders exist in the ASNI for the read and write request channel, enabling parallel lookup. If an address pointing to an unmapped region of memory is presented to the address decoder, an address DECERR response is generated.

2.4.2 HSNI address decode

When an AHB master device generates a request, the connected HSNI receives the transaction address through the request channel. The HSNI decodes the address and calculates the target ID for that address region.

When you use the Socrates IP Tooling platform to configure the HSNI, you generate the HSNI address decoders at the same time. A single address decoder exists in the HSNI as read and write requests come on the same channel. If an address pointing to an unmapped region of memory is presented to the address decoder, it generates an address DECERR response.

2.4.3 PMNI address decode

When an AXI or AHB master generates a request, the connected ASNI or HSNI receives the transaction address through the request channel. The ASNI or HSNI decodes the address and calculates the target ID for that address region.

As described in *2.1.5 APB Master Network Interface* on page 2-37, each PMNI can have up to 16 APB interfaces behind it. If the target ID from the address decode corresponds to a PMNI, the target ID includes information that encodes the exact APB interface behind the PMNI. Correspondingly, the address map in the ASNI and HSNI have address regions defined for each APB interface behind every PMNI instance.

2.4.4 Address striping

NI-700 supports transaction address striping. The ASNI handles address striping as it decodes a transaction address.

An NI-700 configuration must obey the following constraints for address striping:

- You define a Stripe Group by the number of stripe targets that are part of it and the striping granularity.
- NI-700 supports address stripe granules in bytes of 128B, 256B, 512B, 1024B, 2048B, and 4096B.
- NI-700 supports stripe groups which have one, two, or four target interfaces. When there is a stripe group with a single target, all requests to that striped region are sent to the same target. However, the requests are split based on the specified stripe granularity.
- The target interfaces that are part of a stripe group must all be AXI master network interfaces or AHB master network interfaces.
- All AXI and ACE-Lite properties must be the same for all the AXI master network interfaces that are part of the same stripe group.
- All AHB properties must be the same for all the AHB master network interfaces that are part of the same stripe group.
- APB master network interfaces cannot be part of a stripe group.
- It is the responsibility of the SoC integrator and system builder to set up the address maps and stripe groups consistently.

There are several address map restrictions regarding stripe groups:

- Two different stripe groups can have different striping granularity or number of stripe targets or both.
- Two different address regions in an address map can point to two different stripe groups with different stripe granularities or different number of stripe targets or both.
- The default and remap target, or two different remap targets of an address region in an address map can point to two different stripe groups. The two different stripe groups can have two different stripe granularities or different number of stripe targets or both.

Address Hash Function

Two stripe Targets:

- Mask off the lower bits based on the stripe granularity
- XOR all the remaining address bits to generate a single bit: 0 or 1 determines the stripe target

Four stripe Targets:

- Mask off the lower bits based on the stripe granularity
- Generate a 2-bit stripe select to cover four targets:
  - Even stripe select, XOR all the remaining EVEN address bits to generate a single bit
    - Even stripe select drives bit Select[0]
  - Odd stripe select, XOR all the remaining ODD address bits to generate a single bit
    - Odd stripe select drives bit Select[1]

2.4.5 Remap

Registers in the programmers model control the remap functionality.

The address decoder supports up to eight remap states, which are programmed using the address remap vector register. The system must be in a quiesced state before programming the address remap vector register. The **BRESP** response for the configuration writes to the address remap vector register confirms that the register write is complete.

After a write to the address remap vector register occurs, further transactions must only be issued after receiving **BRESP**. This constraint ensures that the interconnect maps transactions correctly.

For more information, see the *Chapter 3 Programmers model* on page 3-115. You can define the remap states using 8 bits of the remap register. A bit in the remap register controls each remap state.

Note

You can use each remap state to control the address decoding for one or more target interfaces. If two remap states that are both asserted affect a target interface, the remap state with the lowest number takes precedence.

You can configure each target interface independently so that a remap state can perform different functions for different masters.

A remap state can:

- Change the target master interface for an address region. The target can change from:
  - One target to a different target
  - A single target to a stripe group
  - One stripe group to a different stripe group
  - A stripe group to a single target
  - Point to ‘No Target’, that is, provide a DECERR
- Remove an address region
- Add an address region

Because of the nature of the distributed register subsystem, the masters receive the updated remap bit states in sequence, and not simultaneously.

The following figures show examples of how different remap states interact with each other. These examples represent the two bottom address ranges of the memory map. The remap bits correspond to these ranges.

While NI-700 can support up to 8 remaps, consider an example configuration that uses three remap bits. The following figure shows the memory map when you set the remap value to `0b000`, representing no remap.

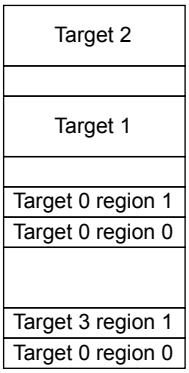

Figure 2-9 No remap, remap set to `0b000`

In the following figure, there is a default memory map that divides target 0 and target 3 into two separate regions. In this example, you can choose to set up a remap value whereby target 3 is aliased over target 0, using the remap code `0b001`. At powerup, target 0 region 0 is aliased over target 3 region 0. After powerup, the target 0 region 0 alias is removed as shown.

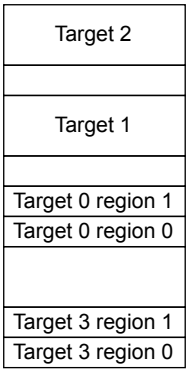

Figure 2-10 Remap set to `0b001`

Alternatively, you might decide to move target 1 to the bottom of the address range by setting remap to `0b010` as the following figure shows.

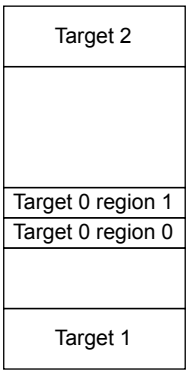

Figure 2-11 Remap set to `0b010`

You can choose to remove entire target regions. The following figure shows that if you set remap to `0b100`, target 3 is removed.

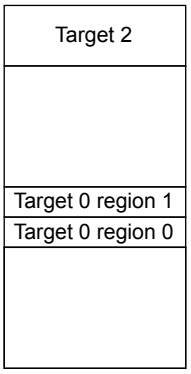

Figure 2-12 Remap set to `0b100`

Remap bit 0 still takes precedence if you set it as the following figure shows.

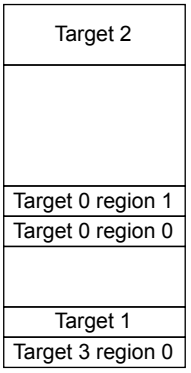

Figure 2-13 Remap set to `0b011`

In addition, you can choose to remove entire memory regions. The following figure shows that if you set remap to `0b101`, target 3 and target 1 are removed.

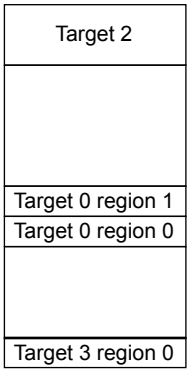

Figure 2-14 Remap set to `0b101`

Note

Program the ADDR_REMAP register in ASNI or HSNI to choose a specific remap, see *ASNI_ADDR_REMAP, Address remap vector register* on page 3-181 and *HSNI_ADDR_REMAP, Address remap vector register* on page 3-225. When you define the address map and remap, ensure you maintain access to the NI-700 programmers model space from at least one ASNI or HSNI. Therefore, the default target of the Config address region from at least one ASNI or HSNI must point to the Config target. If you do not maintain access, you cannot access the NI-700 programmers model to change the address remap option or access any other configuration register. No address region can map to the Config target except the Config address region aligned with Periphbase. The default and remap targets of the Config address region can only be one of 2 values, Config target or No Target.

2.5 Interconnect Device Management

IDM is an OPTIONAL feature that permits the interconnect to configure, manage, and reset individual or groups of system components in isolation, without affecting other components. The IDM functionality integrates with all NI-700 network interface blocks.

If enabled on a network interface, the IDM block is instantiated between the network interface and the device connects to. For example, if enabled on an ASNI, the IDM block is instantiated on the master device to ASNI connection. The following figure shows an example system with IDM blocks integrated with ASNI and AMNI components:

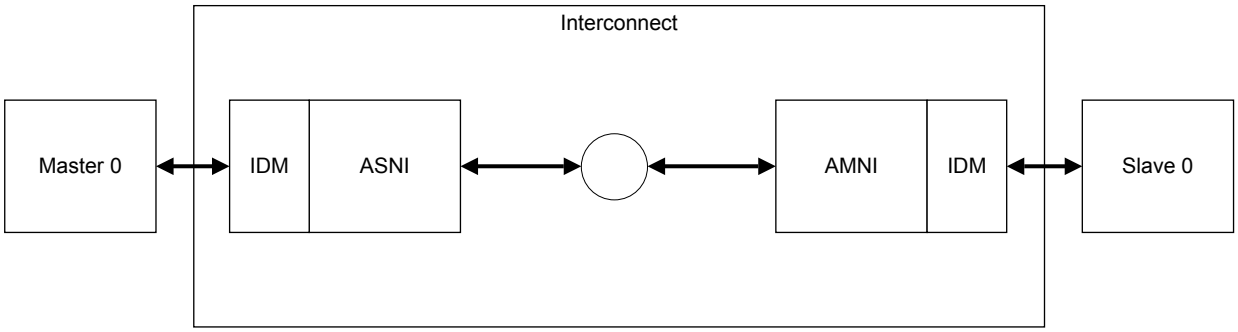

Figure 2-15 IDM integration with NI blocks

Each IDM block on a master device to slave NI connection provides control and status of transactions that the master device issues. Each IDM block on a master NI to slave device connection provides control and status of transactions that are issued to the slave device.

Each IDM block has a set of software-accessible registers. These registers are in the same 4KB NI-700 memory region as all other registers belonging to the same NI.

Both slave and master network interface IDM blocks include the following key features:

- Software configuration, control, and status access through the NI-700 programmers view
- Error logging
- Timeout detection
- Soft reset
- Access control

This NI-700 release has some constraints on specific AXI5 properties regarding IDM. The constraints are:

- Some aspects of AXI5 atomics, for example load, swap, and compare, have both a read and a write response.
  - If you enable IDM, this EAC release does not track a timeout on the read response for the atomic request on the AW channel.
- AXI-G cache maintenance for persistence operations on the write channel, can have a persist response that arrives separately from the completion response.

  Note

  You cannot enable IDM on an AMNI which has the Persist_CMO property set to TRUE.

- AXI-H adds two types of support for *Memory Tagging Extension* (MTE): Basic and Standard.

Note

You cannot enable an AMNI which has MTE support set to Standard.

Note

When you enable IDM and Read Data Chunking features together, protocol violations can exist. Violations occur under real error scenarios where IDM logic must synthesize read and write data beats with SLVERR. The violations also occur if IDM soft reset or isolation entry occurs in the middle of an outstanding request. The IDM logic does not monitor the individual CHUNKNUMs and CHUNKSTRBs that have already arrived for each outstanding request. The monitoring process is very expensive, however the synthesized data beats carry an SLVERR response anyway.

Note

When a slave interface enters soft reset in the middle of a write transaction, ASNI synthesizes any remaining write data beats. This synthesis completes all the write data beats required by the transaction in a protocol-compliant manner. The synthesized data beats have zero write data and zero write strobes, see *slave interface write data transaction leading to a timeout* on page 2-72 in *2.5.6 Soft reset use case examples for master and slave interfaces* on page 2-71. Therefore no memory location is updated or corrupted with this synthesized data beat.

However, AXI protocol violations can occur. For example, a WriteUniqueFull which implies a full cacheline write, requires all write strobes to be set. Similarly, WriteUniqueFull with MTE Tag Update must have all associated WTAGUPDATE bits asserted. For synthesized data beats the WSTRB, WTAGUPDATE, WTRACE are driven to zero. Therefore the synthesized data beats do not update and corrupt the memory.

Note

You must program an adequate timeout value to account for functional scenarios that can lead to delays because of network contention or backpressure from external interfaces. For example, an ASNI has accepted numerous incoming write requests, **AWVALID**, where each request is a very large Burst. It can take a significant amount of time for the ASNI to accept all the incoming write data **WVALID**. As a result the most recent requests observe a large delay between accepting **AWVALID** and receiving **WVALID** corresponding to its first data beat. Furthermore, if there is contention within the interconnect, it can lead to larger delays. Set the timeout value so these functional scenarios are not falsely triggered as misbehaving masters or slaves.

This section contains the following subsections:

2.5.1 IDM and device discovery

The IDM functionality extends and facilitates the NI-700 discovery process by providing a design-time configurable value to identify devices that are attached to the interface.

The discovery mechanism that *2.3 Node ID mapping and discovery* on page 2-53 describes, permits software to discover the voltage, power, and clock domain of any interface in the interconnect. When IDM is enabled on an interface, NI-700 adds a corresponding *IDM DeviceID* on page 3-252 register that contains a design-time configurable 32-bit device_id value. The device_id value is accessible through the programmers' view, and facilitates identification of devices that are attached to the interface and overall system discovery.

For more information about IDM DeviceID configuration, see *Chapter 3 Programmers model* on page 3-115.

2.5.2 Timeout detection through IDM block

The IDM block timeout detection feature uses an interrupt to indicate when transactions from or to the attached device have stalled or failed to progress. This feature is available at both master and slave network interfaces.

Example cases where a slave network interface IDM block indicates stalled or failed transactions from a master device include:

- Failure to receive complete Write Data for a received Write Address phase transaction. The interconnect can indicate failure at any point in the write data beat count.
- Failure to receive a Write Response for an issued Write transaction
- Failure to receive all Read Data beats for an issued Read transaction

Example cases where a master network interface IDM block indicates stalled or failed transactions to a slave device include:

- Failure to accept a received Read Address or Write Address phase transaction
- Failure to accept complete Write Data for a received Write Address phase transaction. The interconnect can indicate failure at any point in the write data beat count.
- Failure to accept a read data beat and write response
- Failure to receive a Write Response for an issued Write transaction
- Failure to receive all Read Data beats for an issued Read transaction

When enabled, this feature produces a level-based interrupt and stores various transaction details for software-based investigation and debug. For more information on the transaction details, see the *3.11.1 Network Interface IDM registers summary* on page 3-251.

Once IDM detects a timeout, the network interface raises a timeout interrupt and automatically enters a mode where:

- The interface gates new transactions from the external device. For example, the interface gates any incoming responses from the downstream slave and prevents them from entering the interconnect at the master interface.
- All outstanding requests are completed in a protocol-compliant manner

This automatic behavior ensures the timeout does not cause the interconnect to start backing up. The NI remains in this mode until the software requests an exit from this mode using a write to the RESET_CONTROL register. For more details see, *2.5.4 IDM soft reset* on page 2-68.

2.5.3 Error logging through IDM block

The IDM block error logging feature uses an interrupt to indicate when an IDM-enabled interface signals an AMBA bus error. This feature is available at both master and slave NIs.

When this feature is enabled, the block produces a level-based interrupt and stores various transaction details for software-based investigation and debug. For more information on the transaction details see, see the *3.11.1 Network Interface IDM registers summary* on page 3-251.

2.5.4 IDM soft reset

The IDM soft reset feature permits software to isolate an endpoint and reset attached erroneous devices, without affecting other endpoints or devices. This feature is available at both master and slave NIs.

Use soft reset together with either or both of the error logging or timeout detection features. For more information, see *2.5.2 Timeout detection through IDM block* on page 2-68 and *2.5.3 Error logging through IDM block* on page 2-68.

There are two different aspects to IDM soft reset functionality:

1. Entry into and exit from a mode when a slave or master network interface gates external incoming transfers. The NI takes responsibility to complete the transactions.
2. Assertion and de-assertion of the external soft reset pin to the external device

Hardware initiated entry based on timeout detection

NI-700 enters this mode if a timeout is detected, see *2.5.2 Timeout detection through IDM block* on page 2-68.

In this mode, the relevant slave or master NI immediately isolates the external interface. If there are outstanding requests at the time NI-700 receives the soft reset request, the NI block completes the transactions in a protocol-compliant manner. To comply with protocol, the NI block generates the required remaining portions of the transaction. The synthesized responses have an SLVERR indication.

On timeout detection at a master network interface, the NI automatically enters this mode but soft reset pin is not toggled:

- No further transactions are sent downstream
- Any incoming responses from downstream are gated and are not permitted to enter
- Any required responses are synthesized with SLVERR and sent upstream to complete transactions in a protocol-compliant manner

On timeout detection at a slave network interface, the NI automatically enters this mode but soft reset pin is not toggled:

- No further incoming transactions are accepted
- Any required responses are synthesized with OK and sent upstream to complete transactions in a protocol-compliant manner

This hardware initiated entry into this mode does not affect the soft reset pin. To toggle the external soft reset pin, software must request this mode by writing the IDM_RESET_CONTROL register. If the endpoint is already in this mode, a write to the IDM_RESET_CONTROL register to enter this mode asserts the external soft reset pin to the external device.

Once in this mode the network interface remains in this mode until software initiates an exit from this mode by writing to the IDM_RESET_CONTROL register.

Writing to the IDM_RESET_CONTROL register to exit this mode does not only exit the mode but also deasserts the external soft reset pin.

For more details on exiting soft reset mode see, *IDM_RESET_CONTROL* on page 3-262.

Software initiated entry

Under software control NI-700 resets the master or slave device attached to the IDM-enabled endpoint. This reset can occur independently of the rest of the interconnect and other external devices. The *IDM_RESET_CONTROL* on page 3-262 associated with the endpoint provides the functionality to request that the attached device is placed into soft reset. This functionality ensures that there are no incomplete transactions at either the master or slave on reset. For more information see, *3.1 About the programmers model* on page 3-116.

When NI-700 receives a soft reset request, the relevant slave or master NI immediately isolates the external interface. If there are outstanding requests at the time NI-700 receives the soft reset request, the NI block completes the transactions in a protocol-compliant manner. To comply with protocol, the NI block generates the required remaining portions of the transaction. The synthesized responses have an SLVERR indication. For information on the transaction flows see, *2.5.6 Soft reset use case examples for master and slave interfaces* on page 2-71.

With software initiated entry, the relevant slave or master NI also toggles the external reset pin to the attached device.

Once in this mode, the network interface remains in this mode until software initiates a write to the IDM_RESET_CONTROL register to exit this mode. Writing to the IDM_RESET_CONTROL register to

exit this mode not only exits the mode but also deasserts the external soft reset pin. For more information on exiting soft reset mode, see, *IDM_RESET_CONTROL* on page 3-262.

IDM_RESET_CONTROL reset initialization input pin

When an endpoint has IDM enabled, it receives an external input pin that is connected to the IDM_RESET_CONTROL register. This pin controls the value of the register out of reset. The SoC-level drives the value of the pin. If the value is set to 0, then there is no change in behavior out of reset. However, if the value is set to 1, when the endpoint exits reset it behaves as if it is already in soft reset mode. This behavior is exactly as if software wrote the bit to enter soft reset mode:

- The asserted external soft reset pin is asserted immediately out of reset.
- The external interface is isolated.
- Any incoming requests are terminated at the endpoint and complete in a protocol-compliant manner with SLVERR responses.

2.5.5 IDM access control

Various scenarios might require software to isolate attached devices from the interconnect in a controlled way. The IDM access control feature enables this isolation and is available at both master and slave NIs.

The IDM access control feature is useful in various situations, for example device power management or disabling of malfunctioning devices.

Using this feature, software can set individual endpoints to prevent transactions from progressing through the interface. By preventing progression of transactions to or from the interconnect, the attached device is isolated from the rest of the interconnect.

The *IDM_ACCESS_CONTROL* on page 3-259 register that is associated with the endpoint provides functionality to request that the attached device is isolated. This functionality ensures that there are no incomplete transactions at either the master or slave on isolation. For more information, see *Chapter 3 Programmers model* on page 3-115.

When NI-700 receives an isolation request, the corresponding slave or master NI waits for the current outstanding transactions to complete normally before entering isolation. This wait is the primary difference between isolation and soft reset.

If the required behavior is to reach a clean point where outstanding transactions are completed and then reset the external device, the following must occur:

- First, software must request isolation entry using the IDM_ACCESS_CONTROL register.
- Once that is successful, software requests a soft reset entry using the IDM_RESET_CONTROL register.

For a slave NI, isolation means the slave does not send incoming requests into the interconnect. For a master NI, isolation means the master does not send incoming requests to the downstream slave. Any new requests received after the isolation request is received, even while there are still pending outstanding transactions to complete, are marked for loopback. That is, they are isolated. The master or slave NI generates internally looped back responses with an SLVERR indication for these new requests. These looped back responses happen when all current outstanding transactions have completed normally.

For example, for the AMNI:

- Requests that are already outstanding complete normally.
- AMNI implements Burst split for cases related to downsizing. If there is an original incoming request that is mid-Burst split, then AMNI issues all Burst split requests corresponding to the original request downstream. AMNI completes those requests normally.
- Any transaction waiting for acceptance downstream, axvalid_o without axready_i, is sent downstream and completes normally.
- Subsequent requests are marked for loopback.
- Requests marked for loopback:

  - Wait for the channel to enter loop-back mode
  - Wait for current outstanding transactions and all transactions sent downstream to complete normally
  - Send SLVERR responses at this point only
- Any new incoming requests are sent with SLVERR and follow the same sequence.

The AXI protocol has independent read and write address channels. Therefore, the read and write channels can enter isolation or loop-back mode at different times after receiving an isolation entry request. However, the *IDM_ACCESS_CONTROL* on page 3-259 reflects isolation entry only after both the read and write channels have entered isolation mode. As a result, either channel can receive loop back responses with SLVERR even before the IDM_ACCESS_CONTROL register reflects a successful isolation entry. Software must either quiesce both the channels before requesting isolation entry, or must be able to handle the preceding behavior.

See *2.5.7 Master and slave interface access control use case examples* on page 2-75 for the transaction flow.

2.5.6 Soft reset use case examples for master and slave interfaces

The examples show the expected use case for the soft reset functionality for a stalled read transaction and a write data transaction at a slave interface. The examples also show the master interface write transaction timeout leading to a soft reset and the master interface read transaction timeout leading to a soft reset.

In this example, the master device has stalled after issuing the second write data-beat. On detection of the timeout, hardware automatically enters the mode to gate the external interface and synthesizes the outstanding write data beats with zeroed write strobes. The write response indication is sent upstream after the ASNI receives it. After the software writes the IDM_RESET_CONTROL register to initiate the soft reset sequence for the endpoint, the external reset pin is asserted. This assertion resets the attached offending master device.

The following figure shows the slave interface write data transaction timeout leading to a soft reset.

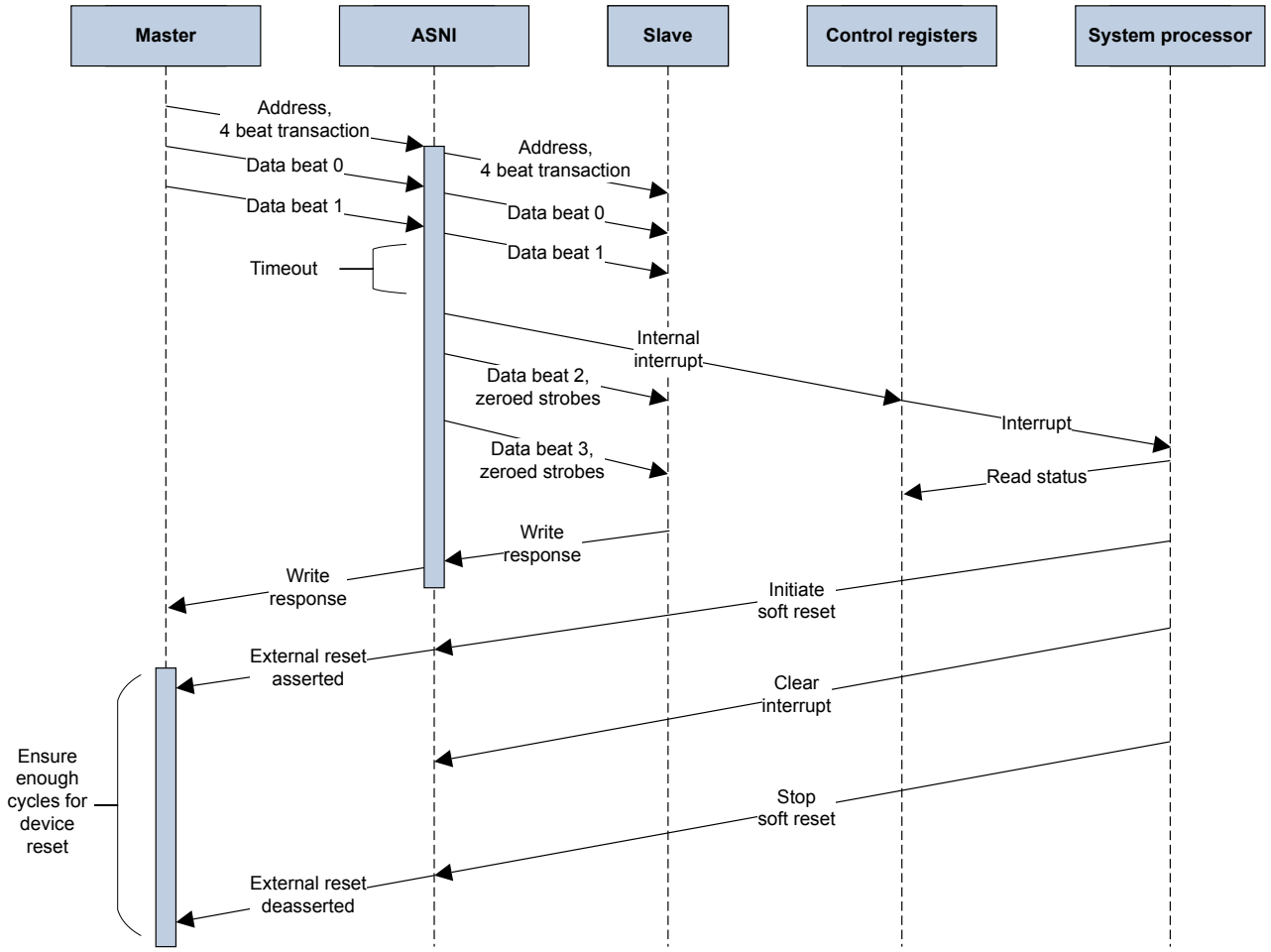

Figure 2-16 Slave interface write data transaction timeout leading to a soft reset

The following diagram demonstrates the expected use case for the soft reset functionality for a read transaction at a slave interface. In this example, a master device stopped accepting read data beats after the second data beat. After the programmed time-out value, an interrupt is asserted for software to handle. On detection of the timeout, hardware automatically enters the mode to gate the external interface. The outstanding read data beats are sunk internally and the request is completed. After the software writes the IDM_RESET_CONTROL register to initiate the soft reset sequence for the endpoint, the external reset pin is also asserted. This assertion resets the attached offending master device.

The following diagram shows the slave interface stalled read transaction, leading to a soft reset.

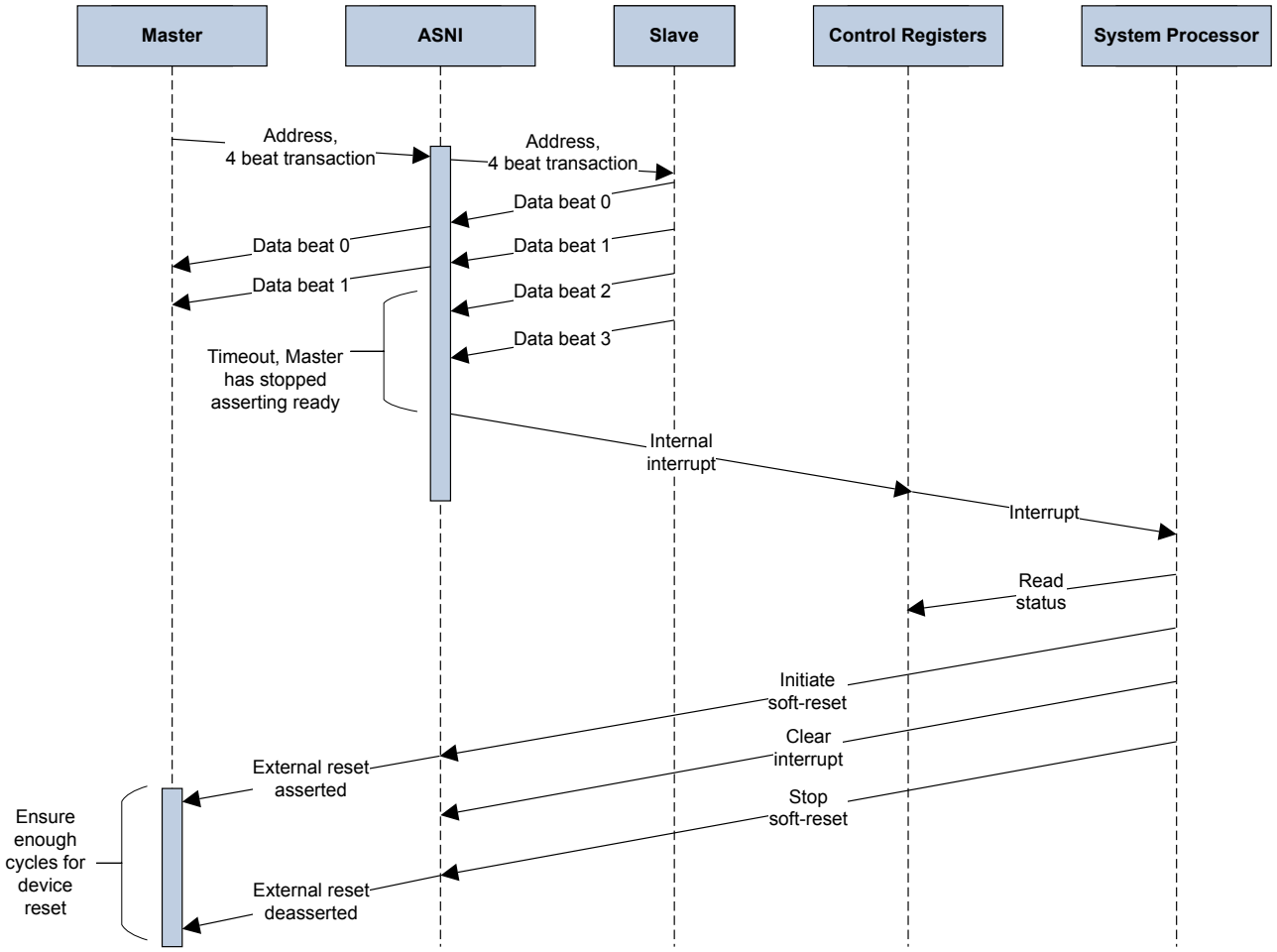

Figure 2-17 Slave interface stalled read transaction leading to a soft reset

The next example demonstrates an expected use case for the soft reset functionality for a write transaction at a master interface. In this example, a slave device has stopped accepting write data beats after the second data beat. After the programmed time-out value, an interrupt is asserted for software to handle. On detection of the timeout, hardware automatically enters the mode to gate the external interface. The outstanding write data beats are sunk internally, a write response with error is generated, and the request is completed. After the software writes the IDM_RESET_CONTROL register to initiate the soft reset sequence for the endpoint, an external reset pin is asserted. This assertion resets the attached offending slave device.

The following diagram shows a master interface write transaction timeout, leading to a soft reset.

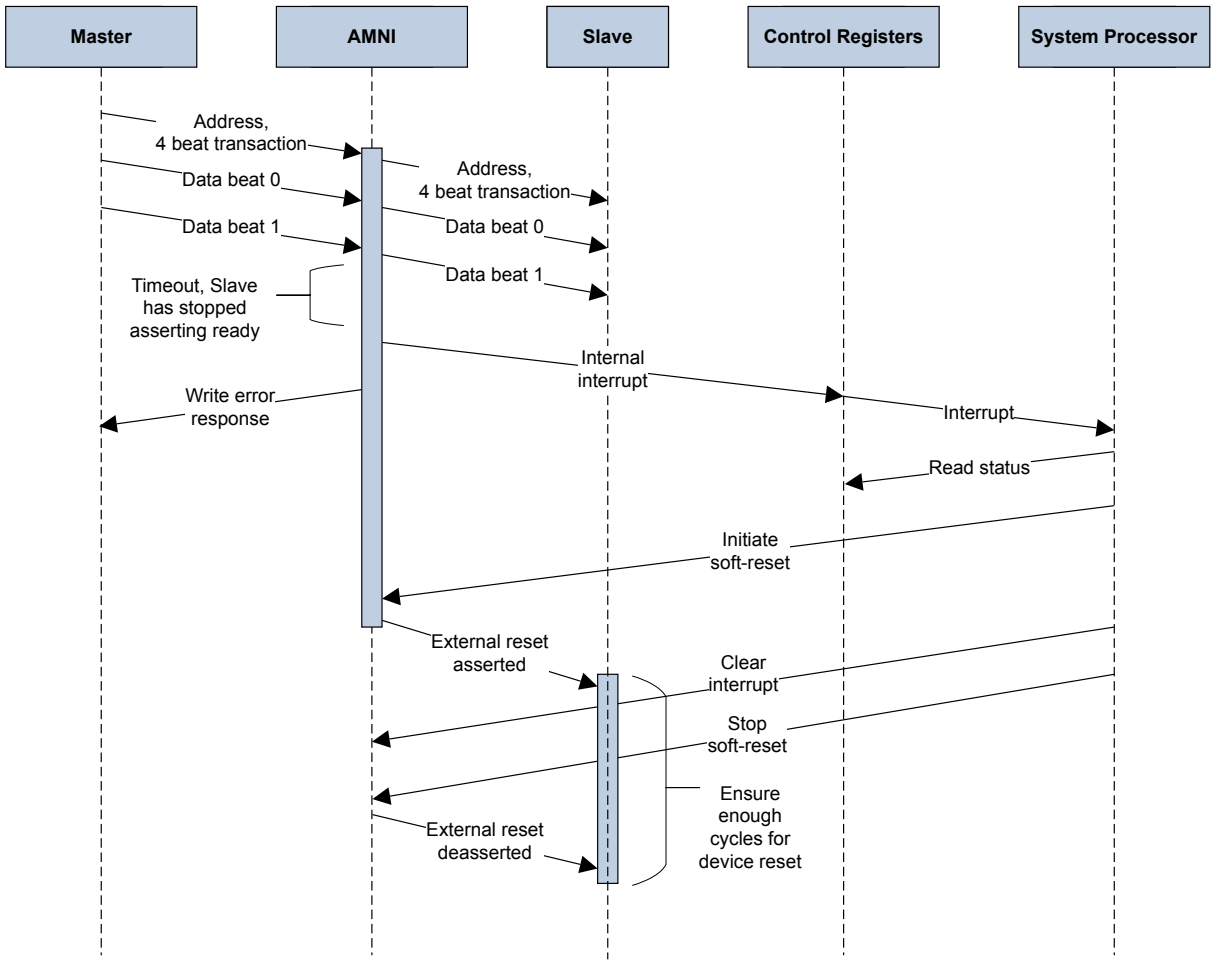

Figure 2-18 Master interface write transaction timeout leading to a soft reset

The following diagram demonstrates an expected use case for the soft reset functionality for a read transaction at a master interface. In this example, a slave device has stopped issuing read data beats after the second data beat. After the programmed time-out value, an interrupt is asserted for software to handle. On detection of the timeout, hardware automatically enters the mode to gate the external interface. The outstanding read data beats are synthesized with zero data and with an error response. After the software writes the IDM_RESET_CONTROL register to initiate the soft reset sequence for the endpoint, the external reset pin is also asserted. This assertion resets the attached offending slave device.

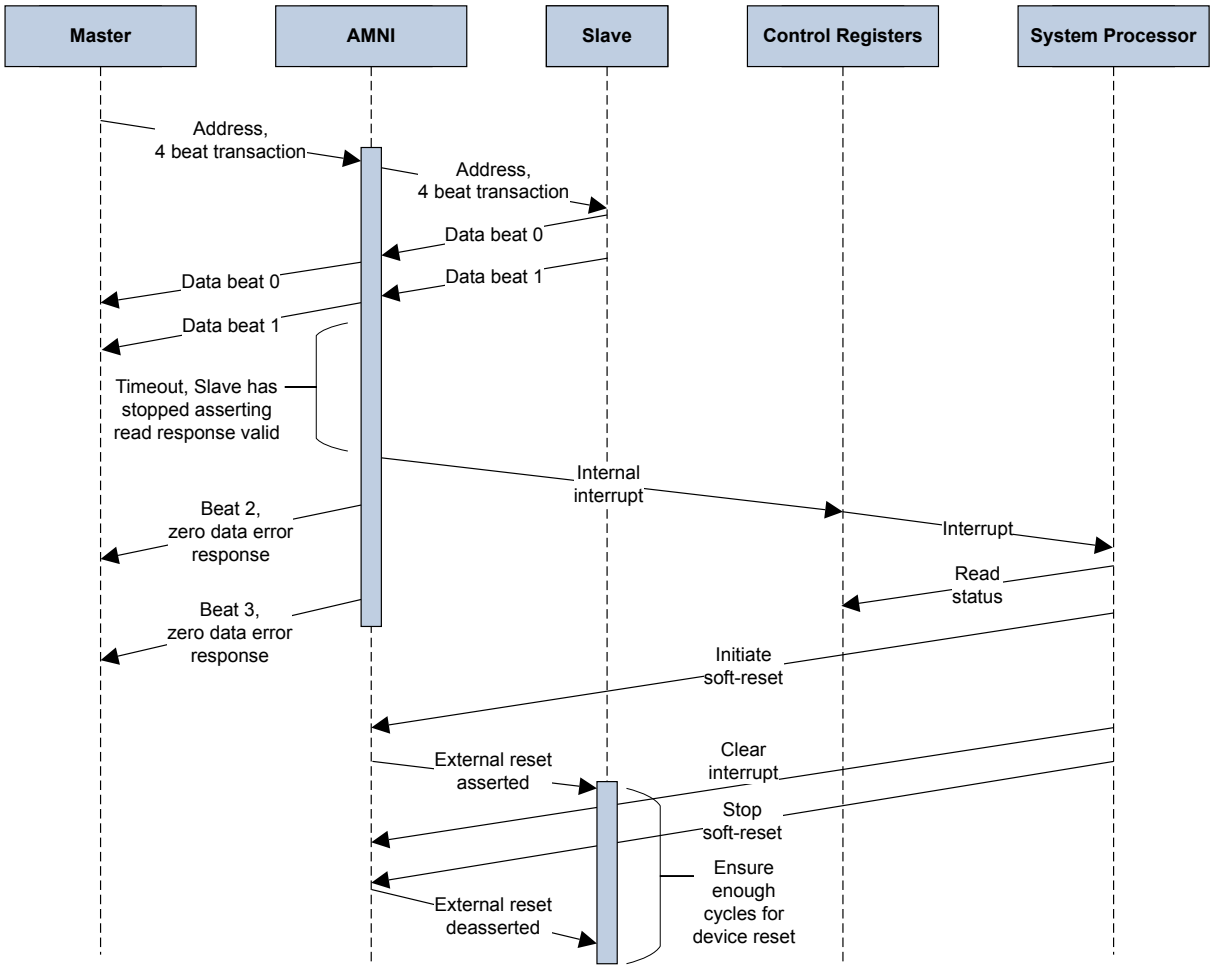

Figure 2-19 Master Interface read transaction timeout leading to soft reset

2.5.7 Master and slave interface access control use case examples

This example demonstrates the expected use case for the access control functionality, for a write transaction at a slave interface.

For this example, a master device has been isolated from the interconnect and issues a new transaction. After the transaction arrival, an error response is generated and an interrupt is asserted for software to handle. The software then removes isolation and allows the transaction to complete later.

The following diagram shows a slave interface write transaction arriving during isolation state:

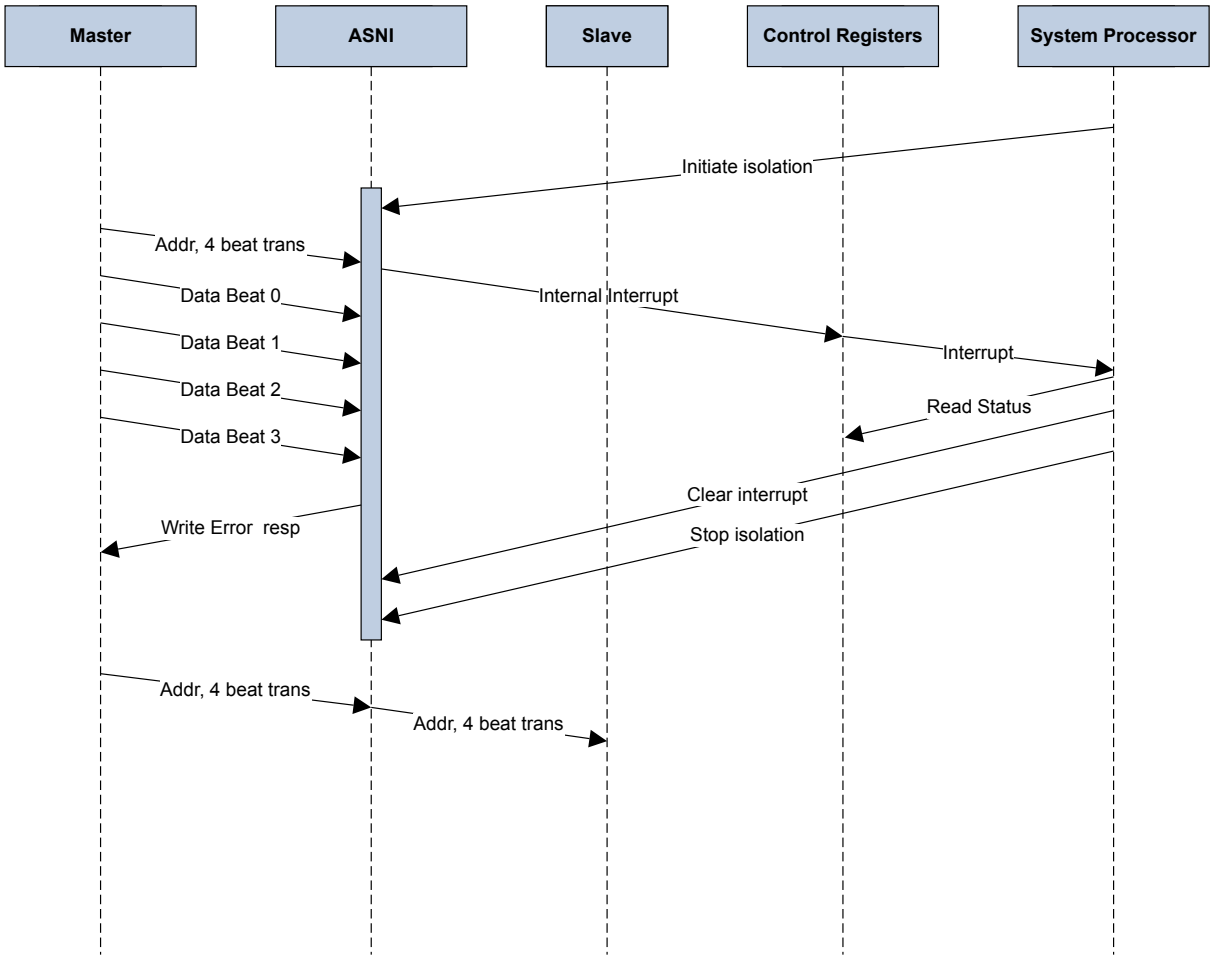

Figure 2-20 Slave interface write transaction arriving during isolation state

This example demonstrates the expected use case for the access control functionality for a write transaction at a master interface. In this example, a slave device has been isolated from the interconnect and the AMNI connected to the slave device receives a new transaction. After the transaction arrival, an error response is generated and an interrupt is asserted for software to handle. The software then removes the isolation and permits the transaction to complete later.

The following diagram shows the master interface write transaction arriving during isolation state:

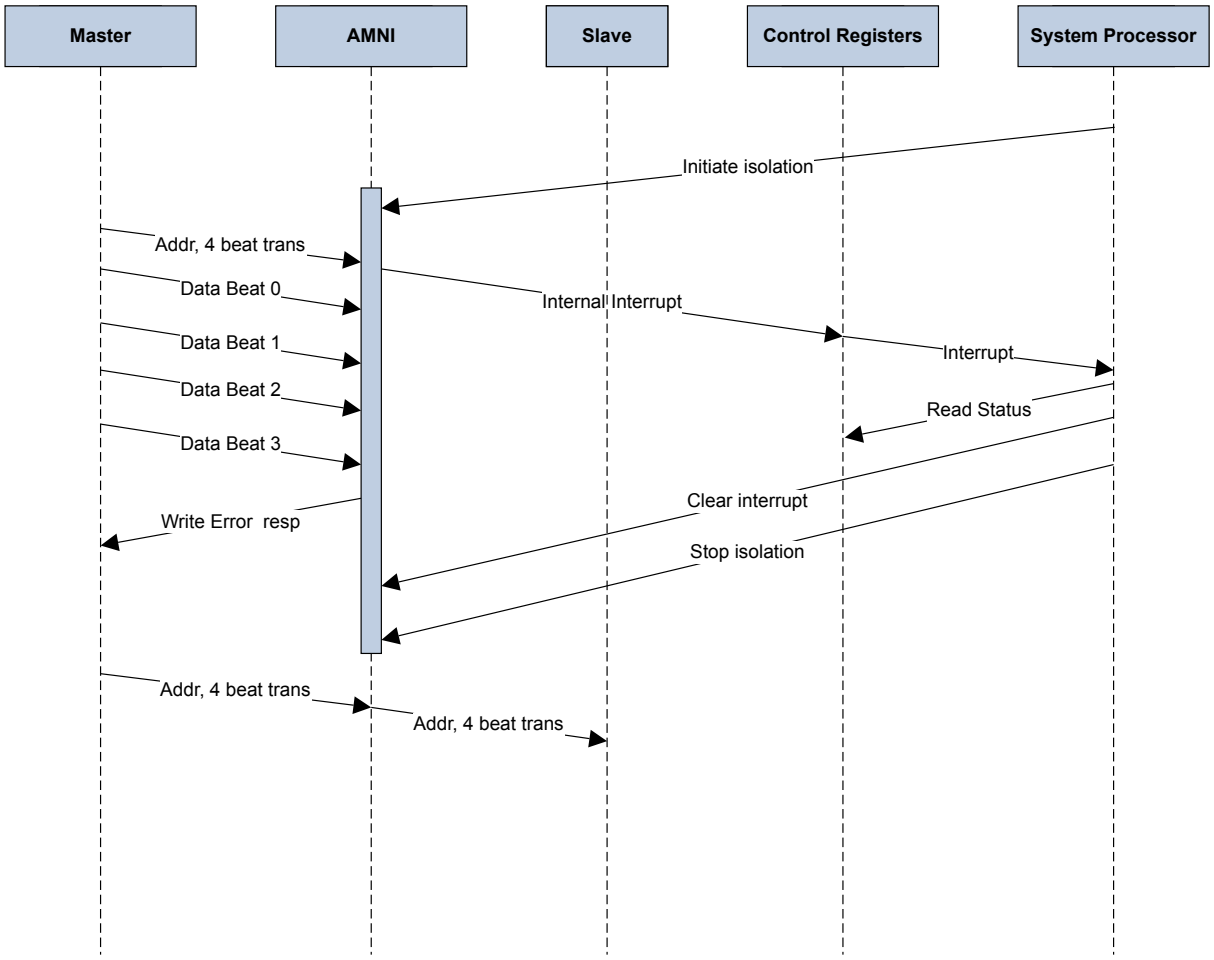

Figure 2-21 Master interface write transaction arriving during isolation state

2.5.8 Sample interrupt handling sequence

In this example, an interface timeout interrupt is asserted.

The software can then read the status register to determine the interface that has asserted the interrupt and if there are multiple assertions. If necessary, all further interrupts can be masked. For this example, the interface indicating a timeout is placed in a soft reset state while its interrupt is cleared. The timeout interrupt status register is checked again to determine if any more interface interrupts require servicing. Once all interrupts have been serviced, the software brings all interfaces out of soft reset.

The following flow chart demonstrates a sample interrupt handling sequence.

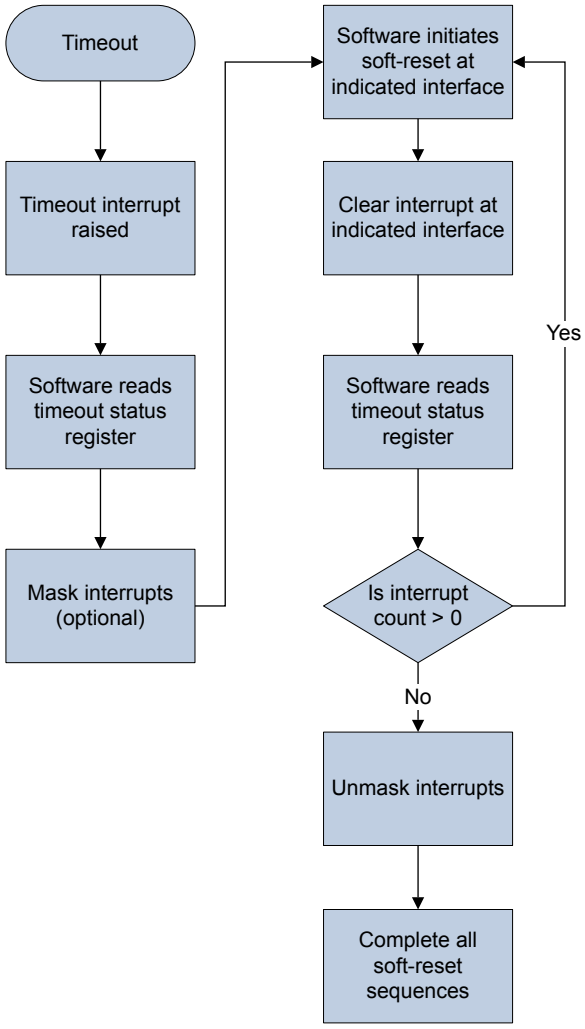

Figure 2-22 Interrupt handling sequence

2.5.9 Soft reset sequence

This example shows a fast sequence for placing a slave device into soft reset.

The software is expected to first stop all the masters that you expect are accessing that slave device. If this stop does not happen, it is possible that the software finds it difficult to leave soft reset at a later stage.

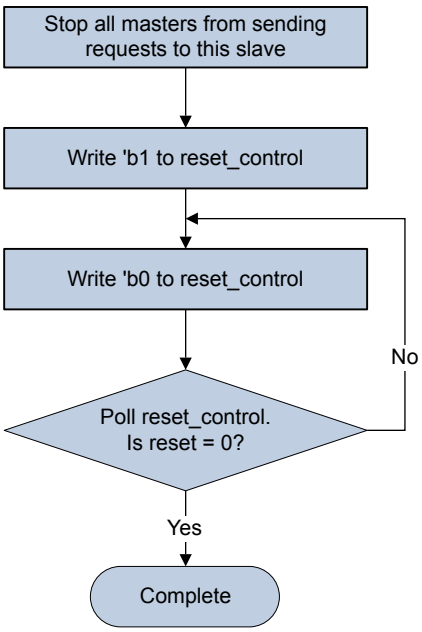

Figure 2-23 Soft reset sequence

Although the preceding sequence is the expected primary use case model, software can use a more cautious sequence. The following diagram shows this more cautious sequence.

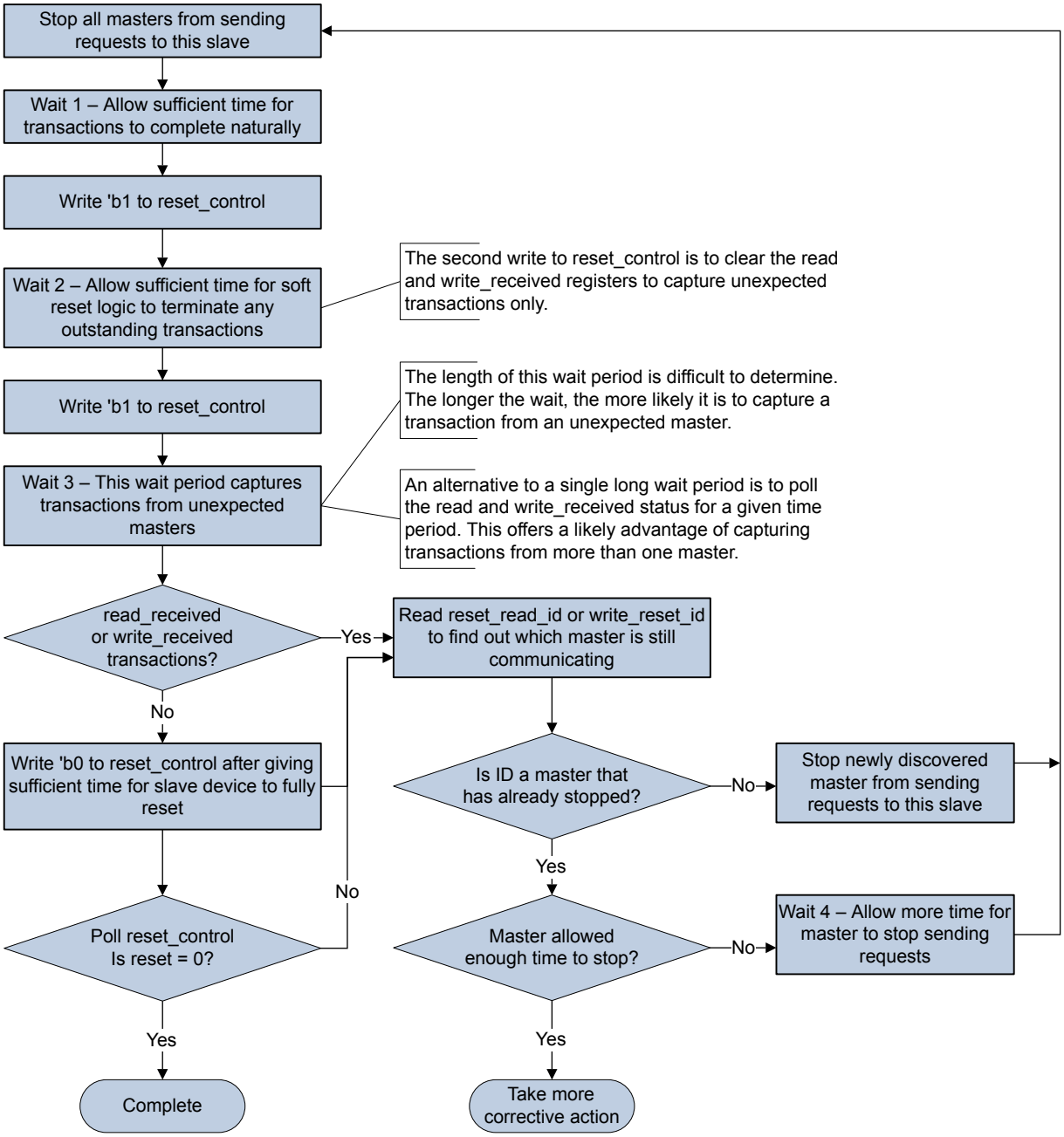

Figure 2-24 Software using a more cautious interrupt handling sequence

Similar software sequences can apply to IDM access control.

2.6 Error handling and interrupts

The NI-700 endpoints have error handling and interrupt functionality. This functionality is related to the IDM functionality and other specific non-IDM conditions.

For more information about the IDM feature, see *2.5 Interconnect Device Management* on page 2-66.

The error logging and interrupt registers are distributed in the NI-700 ASNI, AMNI, HSNI, HMNI, and PMNI endpoints. These registers communicate with central interrupt handling logic in each power domain.

All NI-700 errors are *Uncorrected Errors* (UEs).

This section contains the following subsections:

2.6.1 IDM error logging interrupts and status flags

If you configure an endpoint to include IDM functionality, you enable the logic to trigger error detection and interrupt generation.

The error logging function is integrated with both the IDM soft reset logic and timeout detection logic. For more information, see *2.5.4 IDM soft reset* on page 2-68 and *2.5.2 Timeout detection through IDM block* on page 2-68.

The error logging logic records the transaction that generates an error so that software can examine the transaction. The system uses the following error storage rules on simultaneous error generation:

Table 2-10 IDM error storage rules on simultaneous errors

Condition	Rule
Read and write transaction generate an error simultaneously.	Write transaction has higher priority for error logging.
Timeout is detected in the same cycle as read or write bus error.	Transaction that has timed out has higher priority for error logging.

When the error logging logic receives an error, it raises an interrupt. The error logging logic can raise separate interrupts for the following conditions:

- Bus errors (SLVERR or DECERR)
- Timeout errors
- Endpoint receives incoming requests while it is in soft-reset or isolation state

A software-readable status flag indicates that the logic is processing an error and the type of error being processed. If the logic receives an error while it is processing another error, it uses an overflow flag to record that multiple errors have occurred. This overflow flag is used when simultaneous read, write, and timeout errors occur.

To process an error, software accesses the address and associated characteristics of the transaction that causes the error. When software has processed an error, it can clear the following items separately:

- The interrupt that the error logging logic raised on receiving the error
- The error status flag so that the error logging logic can store another error-causing transaction for processing

When the IDM block detects a timeout for a device, software can use the IDM soft reset or isolation functionality to isolate or reset the external device. However Bus errors or Timeout errors and their

corresponding interrupts can still occur even after entering the active Soft Reset state where the external device has been reset.

These errors can happen under certain circumstances where soft reset was entered in the middle of a pending transaction, and the error status was cleared which permits new timeout errors to be reported. However, since software is aware that any further timeout errors on that interface are not meaningful, it can either choose to disable all error detection (using the *IDM_ERRCTLR* on page 3-254 register) or disable the Timeout error detection (using the *IDM_TIMEOUT_CONTROL* on page 3-266 register) during the period when soft reset is in active state. The ERRSTATUS register (*IDM_ERRSTATUS* on page 3-255) must be cleared for exit from soft reset.

2.6.2 IDM error logging registers

The IDM error logging functionality uses specific registers to store details of the error causing transaction. If you enable IDM on an endpoint, NI-700 includes these registers in the endpoint register block.

NI-700 uses the following registers for error logging:

IDM_ERRSTATUS
Indicates if an error has occurred, the type of error, the overflow flag, and the validity of other error attribute registers for example IDM_ERRMISC0 and IDM_ERRMISC1.

IDM_ERRADR_LSB and IDM_ERRADR_MSB
Stores the address of the transaction causing the error.

IDM_ERRMISC0 and IDM_ERRMISC1
Stores other attributes of the transaction causing the error, including AXI ID, node ID, burst length, and size.

For more information about these registers, see *Chapter 3 Programmers model* on page 3-115.

2.6.3 IDM error processing sequence

When the endpoint logs an IDM error, the system uses a specific sequence of register writes to process the error.

The system uses the following sequence to process IDM errors:

1. Logs the error information in the applicable *IDM_ERRSTATUS* on page 3-255, *IDM_ERRADR_LSB* on page 3-256/*IDM_ERRADR_MSB* on page 3-257, and *ERRMISC0* on page 3-257/*ERRMISC01* on page 3-258 registers.
2. Sets the V and UE fields of the associated *IDM_ERRSTATUS* on page 3-255, register.
3. Sets the UI field of the *IDM_ERRCTLR* on page 3-254 register to mask signaling of the error to the RAS control block.
4. Sets the OF field of the *IDM_ERRSTATUS* on page 3-255 register if there are multiple UEs.

2.6.4 Non-IDM interrupts

The AMNI, HSNI, and HMNI endpoints implement interrupt signals to indicate non-IDM interrupt conditions.

The following table shows the non-IDM interrupt conditions for the AMNI, HSNI, and HMNI.

Table 2-11 Non-IDM interrupt conditions per endpoint

Endpoint	Interrupt condition
AMNI	Non-modifiable transaction is split into multiple individual Burst transactions. Unsupported ACE-LiteACP request.
HSNI	Imprecise errors are detected on actual write response that is received for a request, when early write responses have already been sent for the request.
	Non-modifiable transaction is split into multiple individual Burst transactions.
HMNI	Non-modifiable transaction is Burst split into multiple transactions.

Each endpoint generates interrupts using a set of registers. For more information, see *Chapter 3 Programmers model* on page 3-115.

Table 2-12 ACE-Lite ACP interoperability

Master upstream of ASNI	Slave downstream of AMNI	Interoperability
ACE5-Lite	ACE5-LiteACP	Can connect directly if master upstream of ASNI uses ACE5-LiteACP subset of transactions. If the AMNI interface is configured to ACE5-LiteACP, AMNI expects to receive the subset of transactions defined in the ACE5-LiteACP specification. AMNI checks if transaction properties are satisfying ACE5-LiteACP constraints. If constraints are not met, then AMNI raises an interrupt. For example, if it receives WRAP Burst or if original AxSIZE of transaction was 256-bit, as ACE5-LiteACP permits only INCRs with AxSIZE of 128-bits.
ACE5-LiteACP	ACE5-Lite	ASNI only supports ACE5-Lite. Fully compatible since ACE5-LiteACP is a subset of ACE5-Lite. System integrator can tie off unused inputs to ASNI.

The AXI specification defines ACE5-LiteACP as a subset of ACE5-Lite with specific constraints. NI-700 supports the following combinations.

2.6.5 Two-level interrupt generation

To minimize the number of top-level interrupts in large interconnect designs, NI-700 implements a hierarchical interrupt structure. In this structure, an interface-generated interrupt is passed to an internal status unit per power domain.

The power domain status units are responsible for the following:

- Asserting external interrupts
- Storing the first interface to raise an interrupt of a specific type
- Recording the number of interrupts that are raised internally

Level 1

Each endpoint has interrupt status and mask registers for each type of error that is being reported:

- For IDM-related interrupts, an endpoint has interrupt registers to communicate bus errors, timeout errors, and incoming requests in soft reset and isolation states. For more information, see *2.6.1 IDM error logging interrupts and status flags* on page 2-81.
- For non-IDM interrupts, AMNI, HSNI, and HMNI have a separate set of interrupt registers. For more information, see *2.6.4 Non-IDM interrupts* on page 2-82.

An internal interrupt is asserted whenever any bits in the relevant register are set to `0b1`. The internal interrupt targets the central interrupt handling block in each power domain.

Each endpoint has an interrupt mask register to mask interrupt generation for a specific type of event.

Level 2

The collated control and status registers for each interrupt type contain the number of interfaces that have asserted an interrupt type. These registers can also mask further interrupts. Software can use the information in these registers to determine if there are multiple internal interrupts to clear.

Note

There is only one register to record the node ID of the first interrupt that the system receives. This register updates with further asserted interrupts when the indicated interface has been serviced. To clear an interrupt, software must act on the associated registers that are located within the address region of the interface.

If multiple interfaces raise an interrupt at the same time, the following order is used to determine the first interrupt to report:

1. Master NI, slave device. Highest priority. For multiple endpoints, the endpoint with the lowest internal node ID takes precedence.
2. Slave NI, master device, if there is no conflicting master NI interrupt. Lowest priority. For multiple endpoints, the endpoint with the lowest internal node ID takes precedence.

Note

The programmer's view gives the node ID.

2.6.6 Error interrupt handler flow

A specific sequence of events must occur for software to process both IDM and non-IDM errors.

The following sequence of events describes the process for determining the error source and type of interrupt:

1. An endpoint generates an interrupt. There is a separate wire per interrupt type, which is used to communicate the internal interrupt to the central interrupt handling block of the power domain. Across endpoints, the central interrupt handling block groups individual internal interrupt signals in order of endpoint NodeID.
2. The central interrupt handling block uses a simple arbitration mechanism to record the NodeID of the endpoint for which the external interrupt is raised. For a description of the arbitration mechanism, see *2.6.5 Two-level interrupt generation* on page 2-83. The interrupt handler reads the register and uses the NodeID value to read the corresponding interrupt and error logging registers within the endpoint. See *3.11 Network Interface IDM registers* on page 3-251.
3. The *IDM_ERRSTATUS* on page 3-255 register in the endpoint indicates the type of error. AMNI, HSNI, and HMNI units have Interrupt Status and Interrupt Mask registers. For more information, see the NI registers in *Chapter 3 Programmers model* on page 3-115. The *IDM_ERRADDR_LSB* on page 3-256/*IDM_ERRADDR_MSB* on page 3-257 and *IDM_ERRMISC0* on page 3-257/ *IDM_ERRMISC01* on page 3-258 registers provide further details about the attributes of the request.
4. When software has finished processing an error, it can separately clear any interrupt that was asserted in relation to the error. Software can clear the interrupt by clearing the interrupt status register. Software can also clear the error status flag by clearing the V field of the *IDM_ERRSTATUS* on page 3-255 register. A subsequent error-causing transaction can then be stored and processed processing.

2.6.7 Error handling and interrupt security

NI-700 separates interrupt pins, interrupt registers, and error logging registers into Secure and Non-secure variants.

When a Secure request generates an error, the error properties of the request are logged in the Secure error logging and interrupt registers. The Secure interrupt pin is asserted.

When a Non-secure request generates an error, the error conditions are logged in the Non-secure error logging and interrupt registers. The Non-secure interrupt pin is asserted.

This separation permits Non-secure software to access the Non-secure registers, while preventing access to the Secure registers.

2.7 Master network interface error responses

NI-700 supports error responses from AMNI registers for unsupported transaction types and error responses for CMOs.

AXI Master Network Interface

Requests on the Read Channel

- Incoming ACE-Lite transactions, with **AxDOMAIN** 01, 10, to an AMNI with an AXI interface downstream are terminated at the AMNI with an SLVERR response:
  - If shareable requests must be sent downstream, you must set the interface type to ACE-Lite
  - If connecting to an AXI slave device downstream, then the system integrator can leave the **AxSNOOP** or **AxDOMAIN** unconnected. Leaving **AxSNOOP** or **AxDOMAIN** unconnected effectively downgrades these requests, for example ReadOnce to ReadNoSnoop.
- CMO transactions on the Read channel:
  - If AMNI has an AXI interface downstream, by default incoming CMO requests are terminated at the AMNI with an OK response. However, you can program a configuration control register *AMNI_CONFIG_CTL, Select response* on page 3-209 to change it to an SLVERR response.
  - If AMNI has an ACE-Lite interface downstream but the relevant CMO properties indicate that there is no downstream cache, CMO_ON_READ and CMO_ON_WRITE properties are set to FALSE, then the transaction is terminated at the AMNI with an OK response. However, you can program a configuration control register *AMNI_CONFIG_CTL, Select response* on page 3-209 to change it to an SLVERR response.
  - If AMNI has an ACE-Lite interface downstream and the CMO_ON_WRITE property is set to TRUE to possibly indicate a downstream cache, then the transaction terminates at the AMNI with an SLVERR response.

Requests on the write channel

- Incoming ACE-Lite transactions, with **AxDOMAIN** 01, 10, to an AMNI with an AXI interface downstream are terminated at the AMNI with an SLVERR response:
  - If shareable requests must be sent downstream, you must set the interface type to ACE-Lite
  - If connecting to an AXI slave device downstream, then system integrator can leave the **AxSNOOP** or **AxDOMAIN** unconnected. Leaving them unconnected effectively downgrades these requests for example, WriteUnique to WriteNoSnoop.
- Incoming atomic transactions are terminated at the AMNI with an error response if either:
  - The Atomic_Transactions property is set to FALSE
  - The incoming request has **AxDOMAIN**={01,10}, the Atomic_Transactions property is set to TRUE, but the downstream interface is AXI
- Incoming cache stash transactions that target an AMNI which does not support cache stashing, are converted to the transaction types shown in the following table. However this conversion depends on **AxDOMAIN**.
- Incoming prefetch transactions to an AMNI with an AXI interface or an ACE-Lite interface, always return an OK response

Table 2-13 Cache stash transactions

Cache stash transaction	Domain	ACE-Lite AMNI Cache_Stash_Transactions property set to FALSE	AXI4 AMNI
WriteUniquePtlStash	Non-shareable or System	Convert to WriteNoSnoop	Do not propagate and give an immediate SLVERR response
WriteUniquePtlStash	Inner or Outer Shareable	Convert to WriteUnique (WriteUniquePtl)	Do not propagate and give an immediate SLVERR response
WriteUniqueFullStash	Non-shareable or System	Convert to WriteNoSnoop	Do not propagate and give an immediate SLVERR response
WriteUniqueFullStash	Inner or Outer Shareable	Convert to WriteUnique (WriteUniqueFull)	Do not propagate and give an immediate SLVERR response
StashOnceShared	Any	Do not propagate and give immediate OK response	Do not propagate and give an OK response
StashOnceUnique	Any	Do not propagate and give immediate OK response	Do not propagate and give an OK response

- CMO transactions on the write channel:
  - If AMNI has an AXI interface downstream, incoming CMO requests on the write channel are terminated at the AMNI with an OK response by default . However, you can program a configuration control register *AMNI_CONFIG_CTL, Select response* on page 3-209 to change it to an SLVERR response.
  - If AMNI has an ACE-Lite interface downstream but the relevant CMO properties indicate that there is no downstream cache (CMO_ON_READ and CMO_ON_WRITE properties are set to FALSE), then the transaction is terminated at the AMNI with an OK response by default. However, you can program a configuration control register *AMNI_CONFIG_CTL, Select response* on page 3-209 to change it to an SLVERR response.
  - If AMNI has an ACE-Lite interface downstream and the CMO_ON_READ property is set to TRUE to indicate a possible downstream cache, then the transaction is terminated at the AMNI with an SLVERR response
- Write+CMO transactions on the write channel:
  - If AMNI has an AXI interface downstream, incoming Write+CMO requests on the write channel are terminated at the AMNI with an SLVERR response
  - If AMNI has an ACE-Lite interface downstream but the relevant CMO properties indicate that there is no downstream cache (WRITE_PLUS_CMO, CMO_ON_READ and CMO_ON_WRITE properties are set to FALSE), then the transaction is downgraded. Only the write part of the transaction is issued downstream. The response indication for the downstream write determines the response error indication. The highest priority between the actual response for the write from downstream, and the response value indicated by the value of the configuration control register, determines the response error indication, see *AMNI_CONFIG_CTL, Select response* on page 3-209. For example, if the response from downstream indicates SLVERR, and the value of the configuration control register indicates an OK response, the final response is an SLVERR. Alternatively, if the response from downstream is an OK response, but the configuration control register indicates an SLVERR response, then the final response is an SLVERR.
  - If AMNI has an ACE-Lite interface downstream, WRITE_PLUS_CMO is set to FALSE. However, if either CMO_ON_READ or CMO_ON_WRITE or both are set to TRUE, to possibly indicate a cache downstream, then the transaction is terminated at the AMNI with an SLVERR response.

AHB Master Network Interface

The following request types are terminated at the HMNI and responded to with an SLVERR response based on whether the interface is AHB5 or AHB-Lite. An HMNI with AHB-Lite interface, or an AHB5 interface that does not support extended memory types, responds with an SLVERR to shareable requests with DOMAIN = 2'b01 or 2'b10.

Table 2-14 Request types

Request	AHB5 with Extended Memory Types set to TRUE	AHB-Lite or AHB5 with Extended Memory Types set to FALSE
WriteNoSnoop	Write, Non-shareable	Write, Non-shareable
WriteUnqiue, WriteLineUnique	Write, shareable	SLVERR
WriteUniqueStash, WriteLineUniqueStash	Write, shareable	SLVERR
WriteCMO, WriteLinePlusCMO, WritePlusCMO	SLVERR	SLVERR
WritePrefetch	OK	OK
StashOnceShared, StashOnceUnique	OK	OK
ReadNoSnoop	Read, Non-shareable	Read, Non-shareable
ReadOnce	Read, shareable	SLVERR
DeAllocating transactions (ReadOnceCleanInvalid, ReadOnceMakeInvalid)	Read, shareable	SLVERR
CMO (CleanShared, CleanInvalid, MakeInvalid, CleanSharedPersist)	SLVERR	SLVERR
Atomic transactions (AtomicSwap, AtomicStore, AtomicCompare, AtomicLoad)	SLVERR	SLVERR

APB Master Network Interface

PMNI only supports ReadNoSnoop and WriteNoSnoop request types. All other requests to the PMNI are terminated at the PMNI and responded with an SLVERR response. These requests include WriteUnique, WriteLineUnique, ReadOnce, Cache Maintenance requests, Cache Stashing transactions, DeAllocating transactions, for example ReadOnceCleanInvalid and ReadOnceMakeInvalid, and atomics.

Internal Config Network Interface

All requests that map to the Config Address space are mapped to the internal CFGNI. The CFGNI only supports ReadNoSnoop and WriteNoSnoop request types. All other requests to the CFGNI are terminated at the CFGNI and responded with an SLVERR response. These requests include WriteUnique, WriteLineUnique, ReadOnce, Cache Maintenance requests, Cache Stash transactions, DeAllocating transactions, for example ReadOnceCleanInvalid and ReadOnceMakeInvalid, and atomics.

2.8 Security

NI-700 has a range of security features.

This section contains the following subsections:

2.8.1 TrustZone® technology and security

If you are building a system based on the Secure and Non-secure capabilities provided by TrustZone® technology, Arm TrustZone technology and security apply.

The AXI **AxPROT** signal conveys a Secure or Non-secure attribute for each individual request. This attribute is passed from the Master device through NI-700 to the downstream slave device. The slave device determines the appropriate action from the security access permission of the request.

For accesses to NI-700 internal configuration registers and performance monitoring counters, the security attribute determines the appropriate action. For example, Non-secure accesses to Secure configuration registers are not permitted to read or update the register. If there is a mismatch, reads return zero data and writes are dropped. However, the transaction completes in a protocol-compliant fashion, without indicating any error on the response.

TrustZone® scope

The security checks that TrustZone technology implements cover the scope of a configured network.

For example, security checks that are not within the scope of the network are:

Physical attack
Physical attack on the device.

System implementation information
If you do not consider all the masters that have access to the programmer's view, security vulnerabilities can occur. For example:

- If a Non-secure state master can set QoS requirements that affect its Non-secure transactions, then that Non-secure state master can use this capability. Traffic analysis determines the QoS and priority settings of a Secure master. This feature can be a threat in particular implementations.
- A TrustZone-aware slave requires that you set the connecting network as Non-secure. The network then does not filter the traffic and leaves the slave to determine the correct response. Consider the master that can make this Non-secure configuration and the master, or masters, that can program the TrustZone-aware slave.

Topology issues

It might be possible to suffer timing attacks because of the topology configuration you choose. For example, if two cascaded switches exist with a shared AXI link between them, then continuous Non-secure accesses to a Non-secure slave might block Secure transactions to a different Secure slave.

Resets

It might be possible to carry out a Secure attack by resetting only parts of a data path. The data path might be a section in an individual clock domain within a network, or within a master or slave.

Hierarchical clock gating

It might be possible to carry out a denial-of-service attack by gating clock domains. Only masters in the Secure domain must access the clock controller.

2.8.2 Security access permissions of AXI requests

NI-700 supports signaling security access permissions on incoming AXI requests through **AxPROT[1]**. Depending on the value of **AxPROT[1]**, a request can target specific register types within the interconnect.

NI-700 transports **AxPROT[1]** on each request, which encodes whether the request is Secure or Non-secure. The incoming **AxPROT[1]** value at the ASNI is conveyed on the outgoing interface from the AMNI.

2.8.3 Security access permissions of AHB requests

You can configure whether each instance of HSNI and HMNI in your design supports Secure transfers. Depending on the type of device that is attached, each functional unit also has configurable registers that define how the unit handles request security.

The AHB5 SECURE_TRANSFERS field defines whether the interface supports Secure transfers. For an interface that supports Secure transfers, **HNONSEC** is asserted for a Non-secure transfer and deasserted for a Secure transfer.

You have four security configuration options for an AHB Slave interface:

Table 2-15 AHB Slave interface security configuration options

Configuration option	Description
Pin	HNONSEC pin exists and passes the security attribute.
Programmable	HSNI contains a software programmable register to set the Security attribute for requests from this slave interface. If the register bit is set to 1, then the request is Non-secure and if the bit is set to 0, then the request is Secure. See *HSNI_CTRL* on page 3-223 register.
Always Secure	At build time all requests which originate from this slave interface are marked as Secure.
Always Non-secure	At build time all requests which originate from this slave interface are marked as Non-secure.

You also have four security configuration options for an AHB Master interface:

Table 2-16 AHB Master interface security configuration options

Configuration option	Description
Pin	HNONSEC pin exists and passes the security attribute to the downstream slave.
Programmable	The HMNI contains a software programmable register to set the security attribute of the assets in the downstream slave. If the register bit is set to 1, then the downstream slave is Non-secure. If the register bit is set to 0, then the downstream slave is Secure. See *HMNI_CTRL* on page 3-241 register.
Always Secure	Only Secure transactions access components that are attached to this master interface.
Always Non-secure	Both Secure and Non-secure transactions access components that are attached to this master interface.

The Reset value for the programmable security register is:

Table 2-17 Programmable security register reset values

Interface	Reset value	Description
HSNI	1	Out of reset all requests from HSNI are Non-secure.
HMNI	0	Out of reset all assets in the downstream AHB slave are considered to be Secure.

If a Non-secure transaction targets a master interface which is either programmed as Secure, or is set to Always Secure, the HMNI does not forward the transaction. Instead, HMNI provides the following responses, with no error indication:

Read request Response with zeroed data.

Write request Drops all write data and issues a protocol-compliant write response without error indication.

If a specific instance of an AHB slave network interface is set to Always Non-secure or programmed to be Non-secure, then as defined in *2.8.5 Register security attribute and security classification* on page 2-92 it is not permitted access to Secure registers within NI-700. If the Secure access attribute is overridden as defined in *2.8.6 Secure access register* on page 2-92, no access to Secure registers occurs.

2.8.4 Security access permissions of APB requests

Each PMNI can have up to 16 APB interfaces that are attached to it. Some interfaces can be APB3, and some APB4. You can configure whether each interface supports Secure transfers.

You can independently configure the security behavior of each of the APB interfaces:

Table 2-18 APB security configuration options

Configuration option	Description
Pin	If the protocol is APB4, the PPROT pin communicates the security attribute to the downstream slave. If the protocol is APB3, the pin option is not available as PPROT is not supported on APB3. In this case, only the programmable, Always Secure, and Always Non-secure options are available.
Programmable	PMNI contains a software programmable register to set the security attribute of the assets in the downstream slave. If the register bit is set to 1, the downstream slave is Non-secure. If the register bit is set to 0, then the downstream slave is Secure. Where the protocol is APB4: • If the register is configured to indicate a Non-secure slave, the security attribute is passed on the PPROT pin. • If the register is configured to indicate a Secure slave, the Non-secure requests are not passed downstream. Instead, they are terminated at the PMNI with a protocol-compliant response and SLVERR. Incoming Secure requests are passed downstream to the slave with the security attribute communicated on the PPROT pin.
Always Secure	Only Secure transactions can access components that are attached to this specific APB master interface. If the protocol is APB4, the security attribute is communicated on the PPROT pin. Non-secure requests are not passed downstream. Instead, they are terminated at the PMNI with a protocol-compliant response and SLVERR.
Always Non-secure	Both Secure and Non-secure transactions can access components that are attached to this specific APB master interface. If the protocol is APB4, the security attribute is communicated on the PPROT pin.

The Reset value for the programmable security register is:

Table 2-19 Programmable security register reset value

Interface	Reset value	Description
PMNI	0	Out of reset all assets in the downstream AHB slave are considered to be Secure.

When configured as Programmable, use the PMNI-CTRL software programmable register to indicate the security attribute for each downstream APB interface. Refer to the *PMNI_CTRL* on page 3-289 register for more information on Secure and Non-secure APB interfaces.

When configured to Programmable, Always-Secure or Always Non-secure, PMNI is responsible for completing the Secure access permission check. If a Non-secure transaction targets a master APB interface that is programmed as either Secure or set to be Secure at build time. PMNI does not forward

the transaction to the downstream APB slave. Instead PMNI provides the following responses with no error indication:

Read request Response with zeroed data

Write request Drops all write data and issues a protocol-compliant write response without error indication

2.8.5 Register security attribute and security classification

Each NI-700 register is classified according to its security attribute value. The classification affects the register access permissions.

For requests targeting internal NI-700 registers, the security attribute determines whether the request can access a specific register. For more information, see *2.8.1 TrustZone® technology and security* on page 2-89.

The NI-700 registers are classified according to the following types:

Secure

Accessible only by Secure requests, but this access permission can be overridden. For more information about overriding this access permission, see *2.8.6 Secure access register* on page 2-92.

Secure Debug

Includes PMU registers and Silicon Debug registers. These registers are only accessible by Secure accesses. This access permission can be overridden, but the way that this override is performed is different to standard Secure registers. For more information about overriding this access permission, see *2.8.6 Secure access register* on page 2-92.

Secure Only

Accessible only by Secure requests, but this access permission cannot be overridden.

Non-secure

Accessible by Secure and Non-secure requests.

2.8.6 Secure access register

The NI-700 Secure access register is a Secure Only register that is used to modify the security access permissions of the other Secure registers.

This register is present in all register regions. These register regions include:

- Global configuration region
- Voltage, power, and clock domain register regions
- Slave and Master NI register regions
- PMU register region

Software can program this register to override the Secure access permissions of any specific register region instance, including the Slave NI and Master NI regions.

This register has 2 bits:

[0] Non-secure access override bit. If this bit is set, Non-secure accesses can access all Secure registers within that register region, including the PMU registers and Silicon Debug registers.

[1] Non-secure debug monitor override bit. If this bit is set, Non-secure accesses can access the PMU registers and Silicon Debug registers within that region. If bit 0 is not set, but bit 1 is set, then the security access permissions are only overridden for the PMU and Silicon Debug registers.

For more information, see *Chapter 3 Programmers model* on page 3-115.

2.8.7 Secure debug

NI-700 supports Secure debug through the **SPNIDEN**, **SPIDEN**, and **DBGEN** signals.

The performance monitoring events corresponding to each slave and master interface are specified in *Chapter 4 Performance monitoring* on page 4-292. The **SPNIDEN**, **SPIDEN**, and **DBGEN** inputs that are described in *4.1 PMU and debug* on page 4-293 together determine the conditions for permitting Secure events to be captured in the PMU event counters or Silicon Debug registers. The following equation determines whether Secure debug is permitted:

Secure debug = ((**SPIDEN** & **DBGEN**) | **SPNIDEN**) & (**DBGEN** | **NIDEN**)

If Secure debug is not enabled, then the PMU event counters and Silicon Debug registers can only capture Non-secure events.

2.8.8 Interrupt and error logging register security

NI-700 has separate registers for Secure and Non-secure interrupt status and error logging. NI-700 also has separate Secure and Non-secure interrupt pins per power domain.

For more information, see *2.6.7 Error handling and interrupt security* on page 2-84.

2.9 Memory partitioning and monitoring

NI-700 provides OPTIONAL support for MPAM propagation. It also contains registers which you can use to override the MPAM values that are propagated through the network.

You can configureNI-700 to include MPAM on GT flits. You can also configure individual ASNI and AMNI units to support MPAM. When you enable MPAM on an ASNI or an AMNI, it must include the MPAM signal on all address channels. For more information about the MPAM signals, see *Appendix A Signal descriptions* on page Appx-A-313.

If you enable MPAM support, NI-700 also includes MPAM override registers for each address channel, which are included in the register block of each endpoint. These registers have various use cases:

- Software can use program the MPAM override register to override the MPAM value of a transaction with the value that is stored in the register.
- If you enable MPAM on GT flits, but not on a specific NI instance, then NI-700 ignores the axmpam_override_en field of the MPAM override register. The NI drives the axmpam_override_val onto the GT flit.
- HSNI always drives the axmpam_override_val onto the GT flit when forwarding a transaction that targets an MPAM-enabled downstream slave.

For more information about the MPAM override registers, see *Chapter 3 Programmers model* on page 3-115.

2.10 Memory tagging support

You can enable the MTE_Support parameter, memory tagging, on any AXI master or slave network interface in NI-700.

NI-700 only handles transporting the *Memory Tagging Extension* (MTE) tags appropriately through the interconnect. There is no tag splitter or tag cache within the interconnect. The AMBA AXI specification describes two MTE configurations, that is, basic and standard. All combinations of MTE configurations are supported between the ASNI and AMNI, except when ASNI is configured as standard and AMNI is configured as basic.

The following table describes the AMNI behavior.

Table 2-20 MTE support within NI-700

MTE support in the interconnect	MTE_SUPPORT in the AXI interface at the AMNI	Behavior
False	False	Not supported
False	Basic	Tie off **AxTAGOP** to 0 from the AMNI
False	Standard	Tie off **AxTAGOP** to 0 from the AMNI. **BTAGMATCH** is not present on the AMNI AXI interface.
Basic	False	Ignore tag operation but pass the transactions through. **BTAGMATCH** is not present on the AMNI AXI interface.
Basic	Basic	Propagate **AxTAGOP**. **BTAGMATCH** is not present on the AMNI AXI interface.
Basic	Standard	Propagate **AxTAGOP**. **BTAGMATCH** on the AMNI AXI interface is not used.
Standard	False	Ignore tag operation and pass the transactions through. For setting **BTAGMATCH** in the response upstream from the AMNI, if incoming request is Match then return **BTAGMATCH** as `0b10`, Fail. Otherwise return **BTAGMATCH** as `0b00`.
Standard	Basic	Propagate **AxTAGOP**. For setting **BTAGMATCH** in the response upstream from the AMNI, if incoming request is Match then return **BTAGMATCH** as `0b10`, Fail. Otherwise return **BTAGMATCH** as `0b00`.
Standard	Standard	Propagate **AxTAGOP** and **BTAGMATCH**

2.11 Quality of Service

Throughout NI-700, arbitration nodes decide the order of progression of transactions according to priority. The arbitration nodes use the QoS value of a transaction as it passes through the interconnect to inform arbitration decisions. Furthermore, QoS regulation features are present at traffic injection points in the network. You can use these features to program the behavior of NI-700 during periods of high network traffic.

NI-700 QoS has the following features:

- Configurable QoS options for ASNIs.
- Regulation of read and write requests.
- Programmable QoS facilities for attached AMBA masters.
- **VAxQOSACCEPT[3:0]** signaling. These requests stall transactions with a QoS priority that is less than the current `qosaccept` value.

At any arbitration node, a fixed priority exists for transactions with a different QoS. The highest value has the highest priority. If there are coincident transactions at an arbitration node with the same QoS that require arbitration, then the network uses a *Least Recently Used* (LRU) algorithm.

A side-effect of QoS is that starvation can occur when streams of traffic with high QoS priority block low QoS priority transactions from progressing. To avoid starvation, NI-700 arbitration nodes can be configured to ignore the QoS priority of a set proportion of arbitration decisions. For example, the network can be configured to permit one in 16 transactions to pass regardless of QoS.

NI-700 supports two types of QoS bandwidth regulation:
- Hard bandwidth regulation.
- Soft bandwidth regulation.

This section contains the following subsections:
- *2.11.1 Hard bandwidth regulation* on page 2-96.
- *2.11.2 Soft bandwidth regulation using Bandwidth QoS Value* on page 2-102.
- *2.11.3 QoS override programmable registers* on page 2-104.

2.11.1 Hard bandwidth regulation

You can apply hard bandwidth regulation to the points of network traffic injection in the NI-700, such as the ASNIs.

The NI-700 supports the following types of QoS hard bandwidth regulators:

- *Outstanding Transaction* (OT) regulators
- *Traffic Specification* (TSPEC) regulators

Hard bandwidth QoS regulators block new network traffic according to two constraints:
- The number of transactions that are awaiting a network response.
- An upper bandwidth limit that is applied to the request channels on the slave interface.

Outstanding transaction regulation

The NI-700 ASNI tracks outstanding read and write requests that are submitted at its slave interfaces. You can program the tracking logic to constrain the maximum number of outstanding requests. This feature is known as *Outstanding Transaction Quality of Service* (OT QoS) regulation.

The ASNI tracks all outstanding requests that it receives until it has received the correct number of response GT packets. To reduce congestion within the interconnect, you can constrain request numbers by programming OT QoS regulators.

There are three types of OT QoS regulators that can be configured at the ASNI:

Read regulator
Tracks outstanding read requests.

Write regulator
Tracks outstanding write requests.

Combined regulator
Tracks both read and write requests.

Each OT regulator has an 8-bit programmable register value. If the number of outstanding requests equals the programmed regulator value, then new requests from the corresponding channel are stalled. Requests are also stalled if the number of outstanding requests exceeds the regulator value because of reprogramming. Split Bursts count as multiple entries.

The minimum programmable value for each regulator to maintain OT regulation is one. Programming an OT regulator to zero or more than issuing capability results in no OT regulation.

The OT regulator registers are visible to system software to help performance debug.

Traffic specification regulation

The NI-700 supports hard bandwidth regulation through TSPEC QoS regulators. This regulator is configured in the ASNI and HSNI units.

Multiple TSPEC QoS regulators are present within the ASNI unit. You can configure the ASNI to include TSPEC regulators for the individual AXI read and write request channels, and a combined TSPEC regulator. You can only configure the HSNI to include a combined TSPEC regulator, as AHB only has a single channel.

When programming the TSPEC regulators, you define a limit on the acceptable network traffic profile. The limit is based on the following characteristics:

Table 2-21 Acceptable network traffic profile limit

Parameter	Description
r_value	Average number of transfers per cycle
p_value	Peak number of transfers per cycle
b_value	Burstiness allowance (the amount of data bandwidth more than the average data bandwidth)

——— **Note** ———

Transfers in the preceding parameters correspond to data beats in Read and Write transactions.

———————————

——— **Note** ———

The r_value and the p_value, represent the rate of transfers as a fraction of the maximum bandwidth of the port. The r_value and the p_value are programmed as fractions represented in binary values, see the following examples.

———————————

——— **Note** ———

The b_value represents the total number of transfers that are allowed to be sent above the average rate (r_value). The value is loaded to a counter. Once the counter of permitted transfers is zero, the regulator restricts bandwidth to the exact average rate (r_value). If the port bandwidth drops below the average rate (r_value), then this drop permits all or part of the burstiness allowance to be accepted in addition to the average rate. However this scenario depends on the duration of the low bandwidth or idle window.

———————————

——— **Note** ———

The port bandwidth is limited by the peak rate (p_value) at any time.

———————————

The regulator measures the incoming channel transfer rates and compares them against programmed parameters. Outputs from the TSPEC regulator are used to enforce hard regulation by gating address channel handshake signals. Therefore, incoming requests are blocked until the channel is within specification.

Transactions are stalled if one of the following conditions is met:

1. The total number of transfers exceeds the average number of transfers, plus the burstiness allowance.
2. The data rate exceeds the peak number of transfers per cycle.

How to calculate TSPEC parameters for traffic

How to calculate the r_value (average number of transfers per cycle), b_value (burstiness allowance) and p_value (peak number of transfers per cycle) for the TSPEC parameters.

Calculate the r_value

For example, you would like to program the TSPEC regulator to restrict a master from issuing no more than 40MB/s traffic. If the master has a data width of 64bits and a clock frequency of 100MHz, the max bandwidth of the port is 100 x 8 = 800MB/s. To restrict to 40MB/s, the BW limit on this interface corresponds to the average transfer rate of (40 MBs / (100 x 8)) = 0.05. In other words, 5% of the maximum bandwidth of the 64bit interface at 100MHz.
The value of the 0.05 fraction in binary is 0.0000110011001100 1101. However, since the average rate register field is just 6 bits, that corresponds to setting QOSRDAVG register with value 0'b000011 (6 MSBs). Effectively we are programming an r_value of (8 x 100 / 2^6) x register_value = 37.5 MB/sec introducing a rounding error because of the 6 bits of granularity of the register.

Calculate the b_value

If the traffic from the master is not expected to follow a uniform pattern but contains some repeating patterns or burstiness or both, the b_value parameter permits the setting of some levels of burstiness variability over the average rate from that master. The b_value specifies the number of transfers that are allowed to be sent over the average rate.
For example, if the incoming rate of transfers matches the average rate, an extra b_value number of transfers can be sent until it is exhausted. However, it is not a one-time allowance. As mentioned previously, if the incoming transfer rate falls below the average rate then it permits all or part of the burstiness allowance to be accepted in addition to the average rate depending on the duration of the low bandwidth or idle window.
Following the preceding example, the r_value permits the master to send one transfer every ~21 cycles (8 x 100 / 37.5). If the b_value was set to 0, the regulation enforces exactly that injection rate. But with a nonzero b_value, the regulator permits sending (r_value x total_cycles + b_value) transfers in the monitoring window. The b_value register is 14 bits wide, so it permits 0x3fff more transfers than the average rate over the monitoring window.

Calculate the p_value

The peak rate (p_value) setting defines an upper limit on BW from the master and it is set similar to the average rate as a fraction (r_value). The peak rate settings make sure the BW from the master never exceeds an upper limit when there is considerable burstiness in the traffic and we decided to allow a large chunk of it to pass through using a b_value parameter.
For example, if we set the b_value to the maximum 0x3fff and the p_value to 80MB/s (10% of maximum bandwidth which as a fraction is 0.1 of the channel width and value 0'b000110 in binary fraction). Then the regulator permits transactions to be issued at peak rate of 80MB/s until the b_value worth of transfers is sent above the r_value of 40MB/sec. After the b_value transactions are used, it will enforce a transfer rate of exactly the r_value of 40MB/sec until the burstiness allowance is permitted again.

TSPEC parameter examples

The following examples show how the different TSPEC parameters work together.

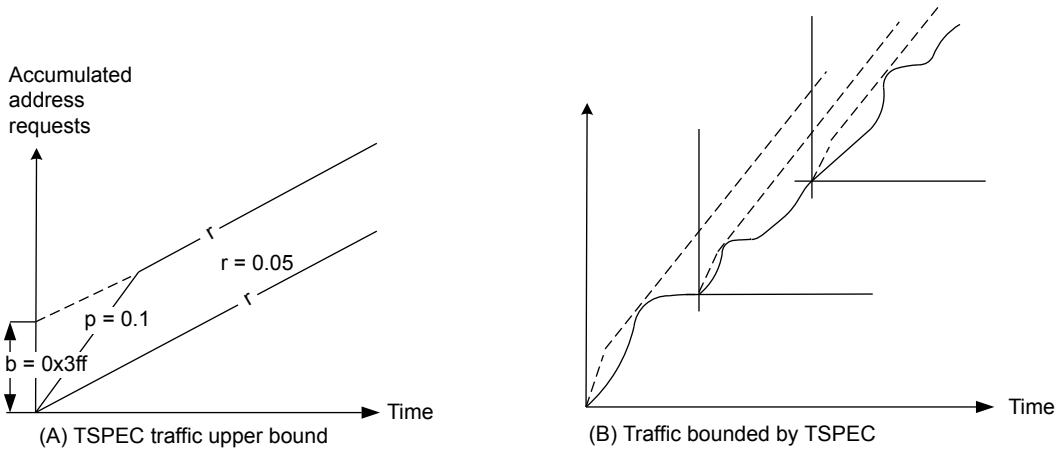

Figure 2-25 TSPEC parameter examples

Example 2-1 Limit set to 50% of maximum interface bandwidth. No allowance for burstiness.

Table 2-22 50% of maximum interface bandwidth

Parameter value	Description	Value
r_value	Average number of transfers per cycle	0.5
b_value	Burstiness allowance	0
p_value	Peak number of transfers per cycle	0

The r_value of 0.5 indicates that only 0.5 beats are permitted every cycle on average.

The rate of incoming transfers is tracked every cycle:
- If there is an incoming transfer, then there is an increment by the number of beats in the transfer.
- Decrement by r_value beats to indicate allowed average rate.

Example 2-2 Dynamically determine when the next transfer is allowed to enter

The following examples show how to use the calculation to dynamically determine when the next transfer is allowed to enter. Since there is no burstiness allowance, incoming transfers are stalled if the counter is ≥ r_value.

In the following scenario, you have an incoming single beat transfer every alternate cycle to achieve a 50% b/w.

Table 2-23 Incoming single beat transfer every alternate cycle

Single beat Transfers	C1	C2	C3	C4	C5	C6	C7	C8
Beats in incoming transfers	1	0	1	0	1	-	-	-
Transfers_nxt	0.5	0	0.5	0	0.5			

In the following scenario, you have an incoming 2-beat transfer every four cycles to achieve a 50% b/w.

Table 2-24 Incoming 2-beat transfer every four cycles

Two beat transfers	C1	C2	C3	C4	C5	C6	C7	C8
Beats in incoming transfers	2	0	0	0	2	0	0	0
Transfers_nxt	1.5	1	0.5	0	1.5	1	0.5	0

Example 2-3 Average 25% of maximum interface bandwidth. Some allowance for burstiness. Limit to peak b/w of 50% maximum interface b/w

Table 2-25 50% of maximum

Parameter value	Description	Value
r_value	Average number of transfers per cycle	0.25
b_value	Burstiness allowance	2
p_value	Peak number of transfers per cycle	0.5

Note

The r_value of 0.25 indicates that only 0.25 beats are allowed every cycle on average. The b_value allows a burstiness allowance of 2 beats but it is only an allowance. Whether the full value of the allowance can be used or not depends on the dynamic window. The p_value of 0.5 indicates that only 0.5 beats are permitted every cycle at peak.

The rate of incoming transfers is tracked every cycle to compare with the average rate:

- If there is an incoming transfer, then increments by the number of beats in the transfer
- Decrement by r_value beats to indicate allowed average rate

Transfers_nxt = Transfers_q + incoming beats – r_value

The rate of incoming transfers is tracked every cycle to also compare with peak rate:

- If there is an incoming transfer, then increments by the number of beats in the transfer
- Decrement by p_value beats to indicate allowed peak rate

Peak_transfers_nxt = Peak_transfers_q + incoming beats – p_value

The following example shows how the calculation is used to dynamically determine how to adjust when the next transfer will be allowed to enter.

- Since there is a burstiness allowance, incoming transfers stall if Transfers_nxt is > {b_value, r_value}
- Incoming transfers also stall if Peak_transfers_nxt ≥ p_value

If either of the preceding conditions is true, incoming transfers are stalled.

The following table shows:

- Cycles C1-C8 and C25-C32 show the peak transfer rate which permits burstiness allowance number of transfers
- Cycle C9-C16 is the average transfer rate
- Cycle C17-C24 is the idle period

Note

In the C1-C8 eight cycle phase, four beats have been transferred which achieves a peak rate of 50%. There are also two extra beats (b_value) permitted over what would have been possible in the same eight cycle window, with an average rate of (25% = 2 beats in 8 cycles).

At the end of the C1-C8 phase, transfers_nxt is two (b_value), therefore in the C9-C16 phase we are able to send only two beats in eight cycles (25% = r_value). So after sending the allowed b_value transfers that exceed the r_value, the bandwidth is limited to average rate (r_value).

Note

Phase C17-C24 is the idle phase and therefore at the end of the phase, transfers_nxt reaches 0. Phase C17-C24 permits phase C25-C32 to repeat and send four beats in eight cycles. The number of beats in each transfer determines the length of each of these phases, that is the r_value, b_value and p_value. So, it is a dynamic window that adjusts each cycle.

Table 2-26 Cycles 1-8, transfer of four beats in eight cycles (peak transfer rate)

Two beat transfers	C1	C2	C3	C4	C5	C6	C7	C8
Transfer	2	0	0	0	2	0	0	0
Transfers_nxt	1.75	1.5	1.25	1	2.75	2.5	2.25	2
Peak_transfers_nxt	1.5	1	0.5	0	1.5	1	0.5	0

Table 2-27 Cycles 9-16, transfer of two beats in eight cycles (average transfer rate)

Two beat transfers	C9	C10	C11	C12	C13	C14	C15	C16
Transfer	2	0	0	0	0	0	0	0
Transfers_nxt	3.75	3.5	3.25	3	2.75	2.5	2.25	2
Peak_transfers_nxt	1.5	1	0.5	0	0	0	0	0

Table 2-28 Cycles 17-24 (idle phase)

Two beat transfers	C17	C18	C19	C20	C21	C22	C23	C24
Transfer	0	0	0	0	0	0	0	0
Transfers_nxt	1.75	1.5	1.25	1	0.75	0.5	0.25	0
Peak_transfers_nxt	0	0	0	0	0	0	0	0

Table 2-29 Cycles 25-32, transfer of four beats in eight cycles (peak transfer rate)

Two beat transfers	C25	C26	C27	C28	C29	C30	C31	C32
Transfer	2	0	0	0	2	0	0	0
Transfers_nxt	1.75	1.5	1.25	1	2.75	2.5	2.25	2
Peak_transfers_nxt	1.5	1	0.5	0	1.5	1	0.5	0

Program the TSPEC parameters

The following list of registers is used to program the TSPEC parameters for read-only, write-only, and read and write combined mode.

r_value, b_value and p_value parameters can be set for read and write channels separately or they can be combined.

Note

HSNI only supports the combined regulator because AHB is a single channel.

When set for read and write channel separately, transfers from only that channel are used to determine if traffic is within specification.

In combined mode, transfers from both read and write channels are combined and are sent to the regulator. So combined rate of read and write channels is checked for being within specification.

The following table shows the registers used to program TSPEC parameters on different channels:

Table 2-30 Registers used to program TSPEC parameters

Register	Channel	Description
ASNI_QOSRDPK, Read TSPEC bandwidth regulator peak rate register on page 3-189	Read	Read hard bandwidth regulator peak rate (p_value)
ASNI_QOSRDAVG, Read TSPEC bandwidth regulator average rate register on page 3-190		Read hard bandwidth regulator average rate (r_value)
ASNI_QOSRDBUR, Read TSPEC bandwidth regulator burstiness allowance register on page 3-189		Read hard bandwidth regulator burstiness allowance (b_value)
ASNI_QOSWRPK, Write TSPEC bandwidth regulator peak rate register on page 3-191	Write	Write hard bandwidth regulator peak rate (p_value)
ASNI_QOSWRAVG, Write TSPEC bandwidth regulator average rate register on page 3-192		Write hard bandwidth regulator average rate (r_value)
ASNI_QOSWRBUR, Write TSPEC bandwidth regulator burstiness allowance register on page 3-191		Write hard bandwidth regulator burstiness allowance (b_value)
ASNI_QOSCOMPK, Combined TSPEC bandwidth regulator peak rate register on page 3-193	Combined Read and Write	Combined TSPEC bandwidth regulator peak rate register (p_value)
ASNI_QOSCOMBUR, Combined TSPEC bandwidth regulator burstiness allowance register on page 3-193		Combined TSPEC bandwidth regulator burstiness allowance register (b_value)
ASNI_QOSCOMAVG, Combined TSPEC bandwidth regulator average rate register on page 3-194		Combined TSPEC bandwidth regulator average rate register (r_value)
HSNI_QOSCOMPK, Combined TSPEC bandwidth regulator peak rate register on page 3-230		Combined hard bandwidth regulator peak rate (p_value)
HSNI_QOSCOMAVG, Combined TSPEC bandwidth regulator average rate register on page 3-231		Combined hard bandwidth regulator average rate (r_value)
HSNI_QOSCOMBUR, Combined TSPEC bandwidth regulator burstiness allowance register on page 3-230		Combined hard bandwidth regulator burstiness allowance (b_value)

2.11.2 Soft bandwidth regulation using Bandwidth QoS Value

The NI-700 ASNI and HSNI support *Bandwidth QoS Value* (BQV) QoS regulators. You can use BQV regulators for soft bandwidth regulation to manage network traffic without restricting transaction requests from entering the network.

BQV regulators do not stall transactions from a particular channel when the programmed bandwidth allocation limit is exceeded. Instead, the regulator overrides the QoS value for transactions on that

channel according to the amount of data the channel has transferred. Therefore, the regulator reduces the QoS value of incoming transactions proportionally to the amount of excess bandwidth that the channel is consuming.

Use the BQV control registers to program the values per regulator. The following table shows the control registers:

Table 2-31 BQV control registers

Parameter	Description
qv_max	Maximum QoS value, default
qv_min	Minimum QoS value
overspend_per_qv	Overspend transfers per QoS value encoded, power of 2
bw_alloc	Average number of transfers per cycle after which QoS reduction starts
bw_burst	Burstiness allowance over the average rate before QoS reduction

By default, the regulator uses the maximum QoS value of the channel.

Bw_alloc and bw_burst work similarly to the r_value and b_value from TSPEC regulation. Use the bw_alloc value and bw_burst value to specify the parameters to determine when regulation begins.

Whenever the total number of transferred data transfers exceeds the limit which makes the channel out of specification (total transfers > bw_alloc x cycles + bw_burst), the extra transfers are divided by the allowed overspend to calculate the reduction from qv_max_i. For example, if extra transfers over the specification were eight and overspend_per_qv value is three, QoS value on the transaction reduces by [8 / (2^3)] = 1. That is, the regulated QoS value is qv_max_i - 1.

The output QoS value qv_o can decrease to qv_min_i, and rise back up to qv_max_i. This fluctuation is because the reduction in QoS value solely depends on current accumulated transfers with regards to the average transfer rate and burstiness value.

As with TSPEC, you can configure the ASNI BQV parameter values for read and write channels separately or together.

Note

HSNI only supports the combined BQV regulator because AHB only has a single channel.

Table 2-32 BQV registers

Register	Description
QOSRDBQV	Read soft BW regulator target bandwidth
QOSWRBQV	Write soft BW regulator target bandwidth
QOSCOMBQV	Combined soft BW regulator target bandwidth

Each of the preceding registers contains fields to set the required parameters to configure BQV regulation as follows:

Table 2-33 Register fields used to configure BQV regulation

Bit assignment	Field name	Description
[31:28]	BQV_OVRSPEND	The excess number of full data bus transfers permitted
[27:14]	BW_BURST	The excess number of full data bus transfers permitted as burstiness allowance
[13:8]	BW_ALLOC	The average number of transfers per cycle

Table 2-33 Register fields used to configure BQV regulation (continued)

Bit assignment	Field name	Description
[7:4]	QVMIN	Minimum value of QoS
[3:0]	QVMAX	Maximum value of QoS

The following diagram shows when QoS reduction begins, regarding excess transfers above the average rate and burstiness allowance:

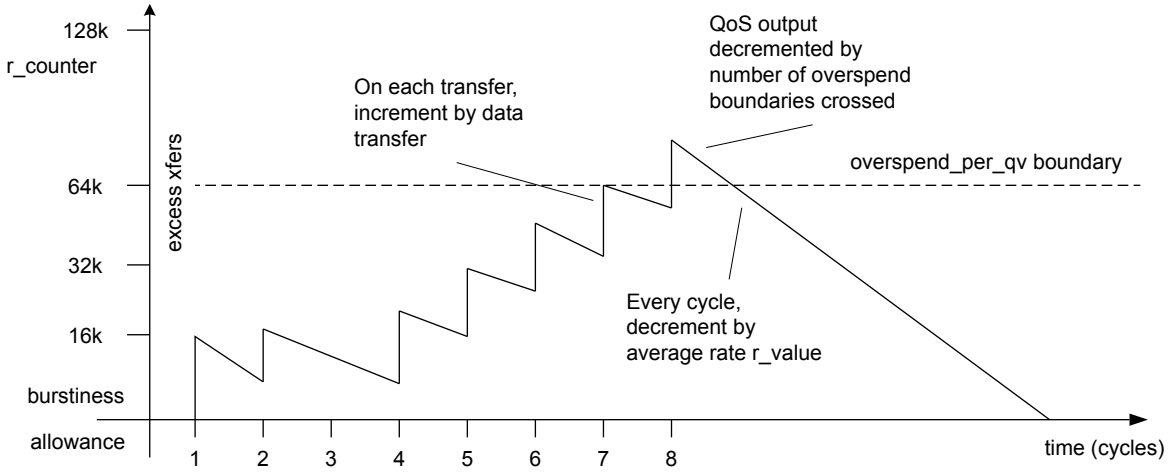

Figure 2-26 Configure BQV regulation

2.11.3 QoS override programmable registers

How to override the incoming AxQOS value independently for the read and write channel using two ASNI channels and how the final QoS value is determined.

NI-700 also supports a programmable way to override the incoming AxQOS value independently for the read and write channel. There are two ASNI registers (*ASNI_ARQOSOVR, Read channel QoS value override register* on page 3-184 and *ASNI_AWQOSOVR, Write channel QoS value override register* on page 3-185) which provide this capability. Program these registers with the AxQoS override value which is applied to transactions when:

- The **QOSOVERRIDE** input signal bit is HIGH or if the **QOS_OVERRIDE_ENABLE** bit of ASNI_QOSCTL register is HIGH, and
- The **BQV** enable bits of ASNI_QOSCTL register are not set.

The table shows how the final QoS value is determined:

Table 2-34 How a QoS final value is determined

QoS override register	QoS override input signal	BQV regulators enabled	Combined regulator QoS < individual RD and WR regulator QoS	Final QoS value
0	0	0	x	AxQOS
x	x	1	0	Individual regulator
x	x	1	1	Combined regulator
0	1	0	x	QoS from register
1	0	0	x	QoS from register
1	1	0	x	QoS from register

2.12 AHB locked transfers

Locked transfer behavior for the *AHB Slave Network Interface* (HSNI) and *AHB Master Network Interface* (HMNI).

At the *AHB Slave Network Interface* (HSNI):

- **HMASTLOCK** is ignored for the *AHB Slave Network Interface* (HSNI)

At the *AHB Master Network Interface* (HMNI):

- Any non-modifiable read or write request is mapped to a locked sequence. **HMASTLOCK** is asserted for AHB transfers belonging to the original non-modifiable read or write request. No arbitration is permitted for the length of the Burst.

  ——— **Note** ———

  If the HMNI does not receive a non-modifiable Burst with burst_size >1KB, this process does not occur. Instead it is converted to a non-modifiable AHB Burst. Therefore **HMASTLOCK** is not asserted and the request receives an SLVERR response.

  ———

2.13 Operation

This section describes how NI-700 operates.

This section contains the following subsections:

2.13.1 Upsizing AXI and ACE-Lite data width function

NI-700 supports transactions from AXI master and slave devices with different data widths. The AMNI is responsible for upsizing data that are sent from a device with a smaller data width than the transaction target.

The AMNI upsizing function can expand the data width by 1:2, 1:4, 1:8, 1:16, or 1:32 ratios.

Upsizing only packs write data for write or read transactions that are cacheable. This section describes the packing rules for different Burst types and acceptance capabilities. The following definitions apply:

- An aligned input Burst means that the address is aligned to the output data width boundary, after the network aligns it to the transfer size.
- An unaligned input Burst means that the network does not align the address to the output data width boundary, even after the network aligns the address to the transfer size.
- If a transaction passes through, the upsizing function does not change the input transaction size and type.

Note

- If the network splits input exclusive transactions into more than one output bus transaction, it removes the exclusive information from the multiple transactions it creates.
- If multiple responses from created transactions are combined into one response, then the order of priority is:
  - DECERR is the highest priority
  - SLVERR is the next highest priority
  - OKAY is the lowest priority

In the examples in this section, the input data width is 64 bits, and the output data width is 128 bits, unless otherwise stated.

INCR Bursts

The network converts all input INCR Bursts that complete within a single output data width into an INCR1 of the minimum SIZE possible. It packs all other INCR Bursts into INCR Bursts of the optimum size possible.

Note

INCR<n> means an incrementing Burst with n data beats.

The following table shows how the network converts INCR Bursts when it upsizes them.

Table 2-35 Conversion of INCR Bursts by the upsizing function

INCR Burst type	Converted to
64-bit INCR1	Passes through unconverted.
64-bit aligned INCR2	INCR1.
64-bit unaligned INCR2	Passes through unconverted.
64-bit aligned INCR4	INCR2.
64-bit unaligned INCR4	Sparse INCR3.

Note

Bursts are never merged.

WRAP Bursts

All WRAP Bursts are either passed through unconverted as WRAP Bursts, or converted to one or two INCR Bursts on the output bus.

The following table shows how the network converts WRAP Bursts when it upsizes them from 64 bits to 128 bits, that is, a ratio of 1:2.

Table 2-36 Conversion of WRAP Bursts by the upsizing function

WRAP Burst type	Converted to
128-bit aligned WRAP2	INCR1.
128-bit aligned WRAP4	WRAP2.
128-bit unaligned WRAP4	Depending on the address: • INCR2 + INCR1. • INCR1 + INCR2.

Note

The network converts input WRAP Bursts with a total payload that is less than the output data width to a single INCR.

Fixed Bursts

All FIXED Bursts pass through unconverted.

2.13.2 Downsizing AXI and ACE-Lite data width function

NI-700 supports transactions from AXI master and slave devices with different data widths. The AMNI is responsible for downsizing data that are sent from a device with a larger data width than the transaction target.

The AMNI downsizing function reduces the data width by 2:1, 4:1, 8:1, 16:1, and 32:1 ratios.

If the transaction is marked as Non-cacheable transaction, the downsizing function does not merge data narrower than the destination bus.

INCR Bursts

NI-700 converts INCR Bursts that fall within the maximum payload size of the output data bus to a single INCR Burst. It converts INCR Bursts that are greater than the maximum payload size of the output data bus to multiple INCR Bursts.

The following table shows how the network converts INCR Bursts when it downsizes them, using a 2:1 downsizing ratio as an example.

Note

The INCR7 output example is only valid if the address is aligned to the destination width, and is not aligned to the source width. For example, if the address is `0x4` for a 64-32 bit downsizer, then an INCR7 output is generated. If the address is `0x1` for a 64-32 bit downsizer, an INCR8 output is generated.

Table 2-37 Conversion of INCR Bursts by the downsizing function

INCR Burst type	Converted to
Aligned INCR4	INCR8
Unaligned INCR4	INCR7
Aligned INCR129	INCR256 + INCR2

INCR Bursts with a size that matches the output data width pass through unconverted.

NI-700 packs INCR Bursts with a SIZE smaller than the output data width to match the output width whenever possible. NI-700 uses the upsizing function to pack the INCR Bursts.

WRAP Bursts

NI-700 always converts WRAP Bursts to WRAP Bursts of twice the length, up to a maximum size of WRAP16. At the maximum size of WRAP16, NI-700 treats the WRAP Burst as two INCR Bursts that can each map onto one or more INCR Bursts.

Note

If a WRAP transaction is aligned to the WRAP boundary, it is converted into an INCR transaction.

FIXED Bursts

NI-700 converts FIXED Bursts to one or more INCR1 or INCRn Bursts, depending on the downsize ratio.

The following table shows how the network converts FIXED Bursts when it downsizes them.

Table 2-38 Conversion of FIXED Bursts by the downsizing function

FIXED Burst type	Converted to
FIXED1	INCR2
FIXED2	INCR2 + INCR2 + ...

NI-700 optimizes unaligned fixed Bursts. If an unaligned input fixed Burst maps onto a single output beat, then the output is a fixed Burst of the optimal size.

2.13.3 Exclusive and locked accesses

How the AXI5 to AHB5 bridge handles AXI Exclusive Bursts and single AXI Exclusive transactions.

The AXI protocol supports Exclusive Bursts but the AHB protocol only supports single (length 1) Exclusive accesses. Therefore, if the AXI5 to AHB5 bridge receives an AXI Exclusive Burst, then it translates the Burst to normal (Non-exclusive) AHB transfers. If the bridge receives a single AXI Exclusive transaction, then it translates the transaction to an Exclusive AHB transfer.

The AXI5 to AHB5 bridge does not support single sparse Exclusive writes, because splitting the write transaction would create an Exclusive AHB Burst. As the preceding Exclusive read might have been answered with HEXOKAY, the bridge always responds with SLVERR for a single sparse Exclusive write.

Note

The bridge returns SLVERR because although OKAY is a valid Exclusive response, an OKAY response could cause the AXI master to repeat the Exclusive write indefinitely.

The bridge uses the **AXID** values to identify which AXI master issues an Exclusive access. For the AHB transfer, the bridge copies the **AXID** value to **HMASTER**.

The following table shows the AXI5 to AHB5 bridge Exclusive transaction mapping.

Table 2-39 AXI5 to AHB5 bridge Exclusive transaction mapping

AXI Exclusive access type	AHB transfer
AXI Exclusive read.	Exclusive AHB transfers, for a single AXI transaction.
	Normal AHB transfers, for a Burst AXI transaction.
AXI non-sparse Exclusive write.	Exclusive AHB transfers, for a single AXI transaction.
	Normal AHB transfers, for a Burst AXI transaction.
AXI sparse Exclusive write.	Normal (SLVERR) AHB transfer, for a single AXI transaction.
	Normal AHB transfers, for a Burst AXI transaction.

2.13.4 Network FIFO and clocking function

If you configure the network as a clock frequency crossing bridge, then non-blocking RP FIFO functions are also configured.

You can configure the FIFO to implement both buffering and clock domain crossing functionality. You can define the FIFO to be:

- SYNC 1:1
- SYNC 1:n
- SYNC m:1
- ASYNC

You can configure the depth value of the FIFO to be 1-8.

Note

You can configure the buffering for multiple flits even if you are using a 1:1 clocking ratio.

All clock boundary crossings are implemented using a FIFO structure with appropriate synchronization for the mode of operation.

Selecting the synchronization mode

Socrates IP tooling platform automatically calculates the mode of synchronization in accordance with the clock relationships that are defined at design entry.

The following options are available:

Asynchronous
Select Asynchronous if the two clocks bear no relationship to one another.

Synchronous (1:1)
Select Synchronous (1:1) if the two clocks are the same.

Synchronous (1:N)
Select Synchronous (1:N) if both of the following are true:
- The first clock has a lower frequency than the second clock.
- The positive edge of the first clock always coincides with a positive edge of the second clock.

Synchronous (M:1)
Select Synchronous (M:1) if both of the following are true:
- The first clock has a higher frequency than the second clock.
- The positive edge of the second clock always coincides with a positive edge of the first clock.

2.13.5 Cyclic Dependency Avoidance Scheme

The AXI protocol permits reordering of transactions, therefore it might be necessary for NI-700 to enforce rules to prevent a transaction deadlock when routing transactions.NI-700 uses a *Cyclic Dependency Avoidance Scheme* (CDAS) to prevent transaction deadlock.

A transaction deadlock can occur when routing multiple transactions concurrently to multiple slaves from a point of ingress to the interconnect, such as a slave interface. To prevent such a deadlock, each ASNI uses one or both of a configurable reorder buffer and a *Single Slave per ID* (SSID) CDAS mechanism. The same CDAS mechanism operates independently for read and write transactions.

NI-700 contains a read reorder buffer and a write response buffer.

Write response buffer

For writes, a response reorder buffer is always present to improve performance.

Read reorder buffer

You can configure an OPTIONAL read data reorder buffer of 1-64 data beats. This option enables a limited number of outstanding requests with the same ID to different destinations.

Responses that are received out-of-order are buffered internally to the ASNI until correct response ordering can be guaranteed. If there is insufficient capacity in the reorder buffer for the total number of read data beats of a transaction, the ASNI uses SSID.

A read reorder buffer entry is allocated on a per transaction basis. This allocation only occurs when required, because of a change in destination for the same traffic ID.

Read reorder buffer slots are also used to merge partial read responses. This process occurs when reorder buffer slots come from AMNIs that have a data width of less than the data width of the ASNI. In this case, NI-700 merges the partial read responses in the same entry of the read reorder buffer to create a full sized data beat at the ASNI.

Non-modifiable accesses, transactions with AxCACHE[1] == 0 to striped memory, do not use the read reorder buffer.

Single Slave per ID

A single slave per ID ensures that for an ASNI, the following transactions go to the same destination:

- All outstanding read transactions with the same ID
- All outstanding write transactions with the same ID, when there are non-modifiable accesses to striped regions and when *Ordered Write Observation* (OWO) is enabled. Otherwise outstanding write transactions with the same ID do not follow SSID.

When the ASNI receives a read transaction that has an ID that:

- Does not match any outstanding transactions, it passes the CDAS.
- Matches the ID of an outstanding transaction, and the destinations also match, it passes the CDAS.
- Matches the ID of an outstanding transaction, and the destinations do not match, it fails the CDAS check, and is stalled.

A stalled transaction remains stalled until one of the rules passes.

AXI non-modifiable transactions which access a striped region, must follow a single slave per ID. That is, if another outstanding transaction has the same ID then it waits for its response to return before it sends out the next one.

Ordered Write Observation

If all other agents in NI-700 observe two write transactions with the same ID and in the same order that the transactions are issued, then an interface can be declared as providing *Ordered Write Observation* (OWO).

NI-700 contains its own logic to check for any outstanding transactions with the same ID for write. If there are any outstanding transactions with the same ID for write and OWO is enabled, then it works in *Single Slave Per ID* (SSID) mode.

If consecutive writes happen to go to the same target, then it sends the requests back to back with full throughput.

If the next write has the same ID, and there is a previous outstanding write to a different destination with the same ID, then it waits to receive the BRESP before it sends the write.

2.13.6 Flit resizing and collating

NI-700 enables traversal of flits between interconnect regions with different link widths. NI-700 provides a configurable SERDES unit to resize flits by collating or dividing them.

You can configure the SERDES unit to resize flits according to various width ratios:

Upsizing (N:M)
Multiple input flits are collated together to form a single large output flit.

Downsizing (M:N)
A single input flit is read into multiple smaller output flits.

After resizing, output flits are aggregated in a FIFO until one of the following conditions is met:

- FIFO tidemark threshold has been reached
- Flit last is received
- The aggregating FIFO is full

When one of the conditions is met, flit aggregation stops and flits exit the block.

A parameter set at design time is used to indicate the number of flits to aggregate and store until they can be released for a packet.

2.13.7 Resource planes

Resource planes provide useful services to NI-700.

The resource plane provides the following services:

- Prevents congestion between traffic flows in the system, by enabling the system designer to separate traffic flows with conflicting requirements on to different resource planes. For example, high-bandwidth bus traffic sources can be prevented from blocking the flow of latency-critical bus traffic.
- Enables you to configure up to four resource planes.

2.13.8 User signals

The NI-700 supports user signal widths for different interface types and supports two different user modes.

The following table describes the supported user signal mode.

Table 2-40 User signal mode description

Mode	Description
User signal mode	Global mode that determines how user data signals, **RUSER** data portion, **WUSER**, **HRUSER**, and **HWUSER**, are handled across all AXI and AHB interfaces. This mode impacts the behavior with upsizing and downsizing.

The following table describes how the two different modes work and which parameters it impacts.

Table 2-41 User signal mode behavior

User signal mode	Upsizing or downsizing	Behavior	Comments
Legacy mode	Downsizing	The interface width of the source is larger than the interface width of the destination. **Note** The user bits which accompanied the original data beat repeat for each of the downsized data beats the original data beat is split into.	This user data mode works if the user bits are per transaction, that is, if they are identical across all beats of the same transaction. If the user bits are different for each data beat, then the scheme is lossy. This difference is clear for the upsizing case where only the bits for the last data beat of the user are retained and the others are lost. In this mode, the user data width is identical across all AXI master and slave interfaces and AHB master and slave interfaces.
	Upsizing	The interface width of the source is smaller than the interface width of the destination. **Note** The user bits which accompanied the last data beat from the source are sent with the upsized data beat. The combination of the smaller data beats creates the upsized data beat.	
Per Byte	Downsizing	The interface width of the source is larger than the interface width of the destination. **Note** The user bits which accompanied the original data beat are appropriately split into corresponding portions. Each portion accompanies each downsized data beat and the original data beat is split into.	This user data mode is suited for use cases where the user bits that accompany the data are expected to scale appropriately with upsizing and downsizing. In this mode, the number of user data bits per byte is identical across all interfaces, that is, AXI master and slave and AHB master and slave interfaces. This identical number enables the user data bits to be scaled up and down along with the `DATA_WIDTH` of each interface without it being lossy. Since the `DATA_WIDTH` of each interface can be different, the `USER_DATA_WIDTH` of different interfaces can be different and is computed as: (`DATA_WIDTH` / 8) * (number of user data bits per byte)
	Upsizing	The interface width of the source is smaller than the interface width of the destination. **Note** The combination of the smaller data beats creates the upsized data beat. Similarly, the user bits which accompanied the individual incoming data beats from the source are combined into a single wider user data bus to accompany the upsized data beat at the destination.	

User signal widths

Specify user signal widths for different interface types in NI-700:

Table 2-42 Supported User signal widths

Interface type	User signal	Signal width parameter	Comments
AXI	**ARUSER**	USER_REQ_WIDTH	This parameter is a single global parameter across all AXI and AHB interfaces. The parameter applies to **ARUSER**, **AWUSER**, and **HAUSER**.
	AWUSER	USER_REQ_WIDTH	This parameter is a single global parameter across all AXI and AHB interfaces. The parameter applies to **ARUSER**, **AWUSER**, and **HAUSER**.
	RUSER	USER_DATA_WIDTH + RUSER_RESP_WIDTH **Note** Issue H of the AXI specification includes a new user parameter for the read response to capture the per-transaction user information. This component of **RUSER** (present in bits RUSER_RESP_WIDTH) is the same for every beat of that transaction. However the USER_DATA_WIDTH component of **RUSER** can be different for every beat. **Note** RUSER_RESP_WIDTH is expected to be 0 when USER_DATA_MODE is 0.	-
	WUSER	USER_DATA_WIDTH	See the note in the preceding User signal mode behavior table for constraints on USER_DATA_WIDTH.
	BUSER	BUSER_RSP_WIDTH	This parameter applies to the AXI write response width.
AHB	**HAUSER**	USER_REQ_WIDTH	This parameter is a single global parameter across all AXI and AHB interfaces and applies to **ARUSER**, **AWUSER**, and **HAUSER**.
	HRUSER	USER_DATA_WIDTH	See the note in the preceding User signal mode behavior table for constraints on USER_DATA_WIDTH.
	HWUSER	USER_DATA_WIDTH	See the note in the preceding User signal mode behavior table for constraints on USER_DATA_WIDTH.

The following table shows the supported user signal parameters and their range:

Table 2-43 Parameters and supported range

Parameter	Supported range
USER_REQ_WIDTH	0-bit to 64-bit
USER_DATA_WIDTH	USER_DATA_MODE = 0 -> 0-bit to 64-bit
	USER_DATA_MODE = 1 Supports between 1-bit and 4-bits per byte Max DATA_WIDTH = 1024-bit Max USER_DATA_WIDTH = (1024 / 8) × 4 = 512-bit
BUSER_RSP_WIDTH	0-bit to 64-bit
RUSER_RESP_WIDTH	0-bit to 64-bit

Chapter 3
Programmers model

This chapter describes the NI-700 programmers model.

It contains the following sections:

3.1 About the programmers model

This section provides general information about the NI-700 register properties.

An NI-700 interconnect consists of various components, such as ASNI, AMNI, HSNI, HMNI, PMNI, and IDM interfaces. The interfaces are accessed through memory-mapped registers for configuration, topology, and status information.

The memory mapped registers are organized in a series of 4KB regions. They are accessed through AXI or ACE-Lite read and write commands.

The following information applies to the NI-700 registers:

- The base address is not fixed, and can be different for any particular system implementation. The offset of each register from the base address is fixed.
- Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in UNPREDICTABLE behavior.
- Unless otherwise stated in the accompanying text:
  - Do-Not-Modify UNDEFINED register bits
  - Ignore UNDEFINED register bits on reads
  - All register bits are reset to 0 by a system or Cold reset
- Access type is described as follows:

RW	Read and write
RO	Read-only
WO	Write-only
RAZ	Read-As-Zero
WI	Write ignored

- Bit positions that are described as reserved are:
  - In an RW register, RAZ/WI
  - In an RO register, RAZ
  - In a WO register, WI
- On **RRESP** and **BRESP** responses, no error is returned.

The NI-700 registers are accessed using the AXI and ACE-Lite slave interfaces that are configured through the Socrates IP tooling platform.

The programmers model contains regions for control, slave NIs, master NIs, and PMUs. Accesses to unmapped or reserved registers are WI or RAZ. Non-secure accesses to Secure registers are WI or RAZ.

NI-700 contains several control registers that enable software to modify NI-700 behavior. Usually, programming the control registers immediately impacts the execution of transactions that flow through the NI-700.

When programming a control register in a specific unit instance, for example a specific instance of the ASNI, we recommend bringing the specific instance of the unit to a quiesced state before programming the register. The **BRESP** response for the configuration write to the register confirms that the register write is complete. After a write to the register occurs, further transactions can be issued after receiving this **BRESP**. Following this recommendation provides a clear boundary after which further transactions to that instance use the updated control register value.

This section contains the following subsection:

- *3.1.1 Requirements of configuration register reads and writes* on page 3-116.

3.1.1 Requirements of configuration register reads and writes

Reads and writes to the NI-700 configuration registers must meet certain requirements.

Reads and writes to the NI-700 configuration registers must meet the following requirements:

- The request must be of Device type
- The request must be ReadNoSnoop or WriteNoSnoop
- The request must be a normal request not an exclusive access
- The **AxDOMAIN** must be System Shareable
- The Burst type must be INCR
- The size must be 4 bytes
- The address must be 32-bit word-aligned
- All write strobes for the 4 bytes must be set

If an incoming request does not obey these constraints, NI-700 returns it with an SLVERR. Reads are handled as RAZ, and writes as WI. However, the transaction completes in a protocol-compliant manner with SLVERR on the **RRESP** or **BRESP** as appropriate.

Secure registers are only accessed through a Secure access (depending on the value of the Secure Access register in the unit). Non-secure registers are accessed through either a Secure or Non-secure access. Security mismatches are not reflected as SLVERR, however other conditions determine the error response indicated. For example, if there is a security mismatch together with an unsupported request opcode, then it is an SLVERR due to the unsupported request opcode. However, if the only cause is the security mismatch then it is an OK response.

3.2 Global registers

This section describes the global registers of NI-700. It contains a summary of the global registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.2.1 Global registers summary

The register summary lists the global NI-700 registers and some key characteristics.

The following table shows the global registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, see your SoC implementation documentation. The offset of each register from the base address is fixed.

Table 3-1 Global registers summary

Offset	Name	Type	Reset	Width	Description
0x000	NODE_TYPE	RO	Configuration dependent	32-bit	*NODE_TYPE, Global node type register* on page 3-119
0x004	CHILD_NODE	RO	N	32-bit	*CHILD_NODE, Child node information register, voltage domains* on page 3-119
0x0008-0x00FF	VOLTAGE_DOMAIN_OFFSET_POINTERS	RO	P	32-bit	*VOLTAGE_DOMAIN_POINTERS, Voltage domain offset pointers register* on page 3-120
0x0F08	SECR_ACC	RW	0x00	2-bit	*SECR_ACC, Secure access register* on page 3-120
0x0FD0	PERIPHERAL_ID4	RO	0x4 Partially device dependent	8-bit	*PERIPHERAL_ID4* on page 3-121
0x0FD4	PERIPHERAL_ID5	RO	0x00	8-bit	*PERIPHERAL_ID5* on page 3-121
0x0FD8	PERIPHERAL_ID6	RO	0x00	8-bit	*PERIPHERAL_ID6* on page 3-122
0x0FDC	PERIPHERAL_ID7	RO	0x00	8-bit	*PERIPHERAL_ID7* on page 3-123
0x0FE0	PERIPHERAL_ID0	RO	0x3B	8-bit	*PERIPHERAL_ID0* on page 3-123
0x0FE4	PERIPHERAL_ID1	RO	0xB4	8-bit	*PERIPHERAL_ID1* on page 3-124
0x0FE8	PERIPHERAL_ID2	RO	0x0B	8-bit	*PERIPHERAL_ID2* on page 3-124
0x0FEC	PERIPHERAL_ID3	RO	0x00	8-bit	*PERIPHERAL_ID3* on page 3-125
0x0FF0	COMPONENT_ID0	RO	0x0D	8-bit	*COMPONENT_ID0* on page 3-126
0x0FF4	COMPONENT_ID1	RO	0xF0	8-bit	*COMPONENT_ID1* on page 3-126
0x0FF8	COMPONENT_ID2	RO	0x05	8-bit	*COMPONENT_ID2* on page 3-127
0x0FFC	COMPONENT_ID3	RO	0xB1	8-bit	*COMPONENT_ID3* on page 3-127

3.2.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

NODE_TYPE, Global node type register

This register identifies the node type as NI-700 global or base registers.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

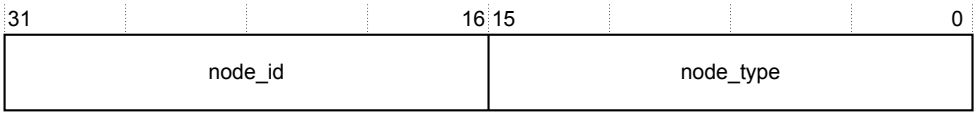

Figure 3-1 NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-2 NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The value of this field is `0x0000` for the global register region.
[15:0]	node_type	The value of this field is `0b00000000`, indicating that the associated node is a global register node.

CHILD_NODE, Child node information register, voltage domains

This register identifies the number of voltage domains that are present in the NI-700 system.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

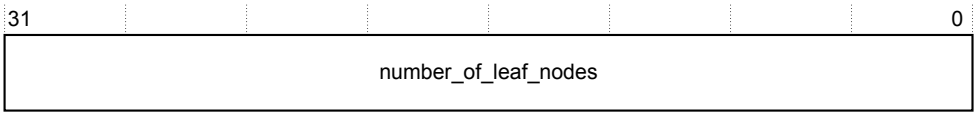

Figure 3-2 CHILD_NODE bit assignments

The following table shows the bit descriptions.

Table 3-3 CHILD_NODE bit descriptions

Bits	Name	Description
[31:0]	number_of_leaf_nodes	The value of this field is the number of voltage domains that are present in the NI-700.

VOLTAGE_DOMAIN_POINTERS, Voltage domain offset pointers register

This register points to the offset from the peripheral base, for the base address of the 4KB voltage domain register region.

Usage constraints

None

Configurations

A copy of this register exists for each voltage domain.
Available in all NI-700 configurations

Attributes

For more information, see *3.3.1 Voltage domain registers summary* on page 3-129.

The following figure shows the bit assignments.

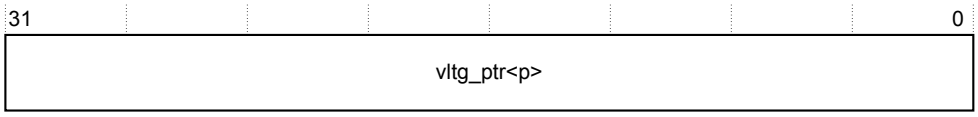

Figure 3-3 VOLTAGE_DOMAIN_POINTERS bit assignments

The following table shows the bit descriptions.

Table 3-4 VOLTAGE_DOMAIN_POINTERS bit descriptions

Bits	Name	Description
[31:0]	vltg_ptr<p>	Offset from the peripheral base, for the base address of the 4KB voltage domain register region.

SECR_ACC, Secure access register

This register controls whether only Non-secure transactions can read and program the NI-700 registers.

Usage constraints

Accessible using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

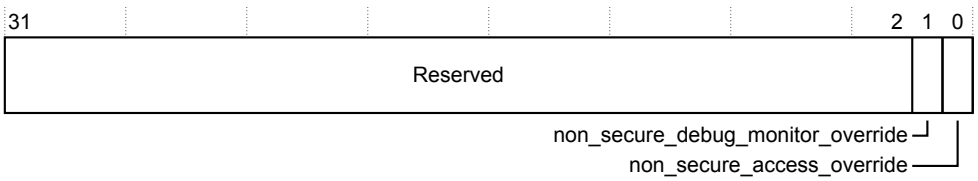

Figure 3-4 SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-5 SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	non_secure_debug_monitor_override	Debug monitor security override: **0** Disable. Non-secure access to the PMU and Interface Monitor registers unless overridden by bit[0]. **1** Enable. Non-secure access to the PMU and Interface Monitor registers.
[0]	non_secure_access_override	Non-secure register access override: **0** Disable. Non-secure access to the Secure registers in this register region. **1** Enable. Non-secure access to the Secure registers in this register region.

PERIPHERAL_ID4

This register indicates the number of 4KB blocks that are occupied, and the value for bits[11:8] of the JEP106 ID code that identifies Arm.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

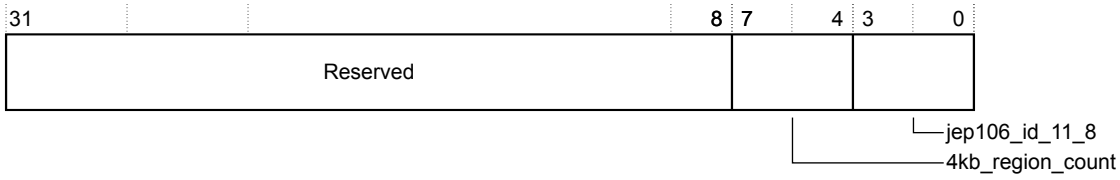

Figure 3-5 PERIPHERAL_ID4 bit assignments

The following table shows the bit descriptions.

Table 3-6 PERIPHERAL_ID4 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:4]	4kb_region_count	The $\log_2$ value of the number of 4KB blocks that are occupied for the NI-700 programmers view.
[3:0]	jep106_id_11_8	Bits[11:8] of the JEP106 ID code that identifies Arm value of `0x4`.

PERIPHERAL_ID5

This register is reserved in the NI-700 design.

Usage constraints

None

Configurations
Reserved in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

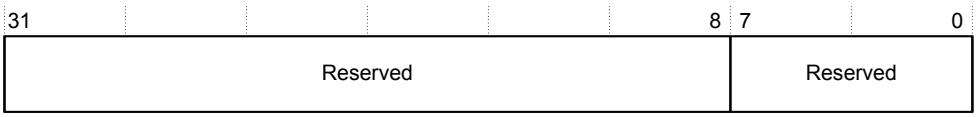

Figure 3-6 PERIPHERAL_ID5 bit assignments

The following table shows the bit descriptions.

Table 3-7 PERIPHERAL_ID5 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	-	Reserved

PERIPHERAL_ID6

This register is reserved in the NI-700 design.

Usage constraints
None

Configurations
Reserved in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

Figure 3-7 PERIPHERAL_ID6 bit assignments

The following table shows the bit descriptions.

Table 3-8 PERIPHERAL_ID6 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	-	Reserved

PERIPHERAL_ID7

This register is reserved in the NI-700 design.

Usage constraints
None

Configurations
Reserved in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

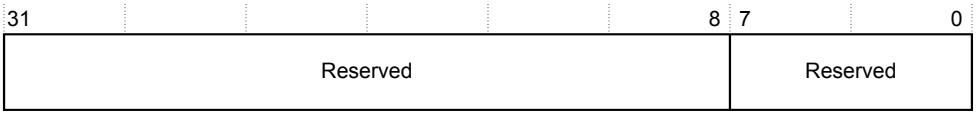

Figure 3-8 PERIPHERAL_ID7 bit assignments

The following table shows the bit descriptions.

Table 3-9 PERIPHERAL_ID7 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	-	Reserved

PERIPHERAL_ID0

This register indicates the value for bits[7:0] of the NI-700 part number.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

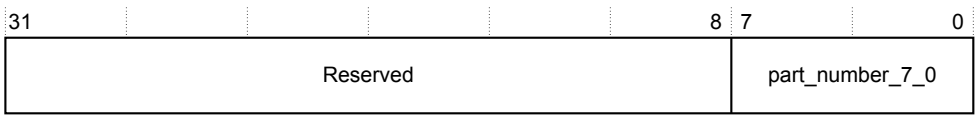

Figure 3-9 Peripheral ID0 bit assignments

The following table shows the bit descriptions.

Table 3-10 Peripheral ID0 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	part_number_7_0	Bits[7:0] of the NI-700 part number with a value of `0x3B`

PERIPHERAL_ID1

This register indicates the value for bits[3:0] of the JEP106 ID code that identifies Arm, and bits[11:8] of the NI-700 part number.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

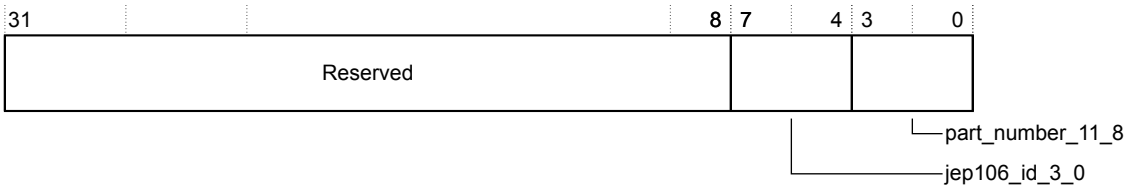

Figure 3-10 PERIPHERAL_ID1 bit assignments

The following table shows the bit descriptions.

Table 3-11 PERIPHERAL_ID1 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:4]	jep106_id_3_0	Bits[3:0] of the JEP106 ID code that identifies Arm with the value of `0xB`.
[3:0]	part_number_11_8	Bits[11:8] of the NI-700 part number with the value of `0x4`.

PERIPHERAL_ID2

This register indicates the NI-700 product version, and the value for bits[6:4] of the JEP106 ID code that identifies Arm.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

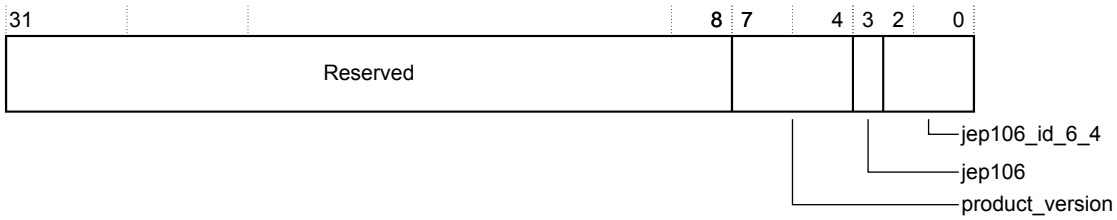

Figure 3-11 Peripheral ID2 bit assignments

The following table shows the bit descriptions.

Table 3-12 PERIPHERAL_ID2 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:4]	product_version	NI-700 revision. • `0x0` indicates r0p0 LAC • `0x1` indicates r1p0 EAC. This value also applies to the r0p1 DEV release. • `0x2` indicates r2p0 EAC.
[3]	jep106	When set, this bit indicates that the JEP106 ID code is used and has a value of 1.
[2:0]	jep106_id_6_4	Bits[6:4] of the JEP106 ID code that identifies Arm and has a value of `0b011`.

PERIPHERAL_ID3

This register indicates the Arm-approved ECO number, and the NI-700 customer modification number.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

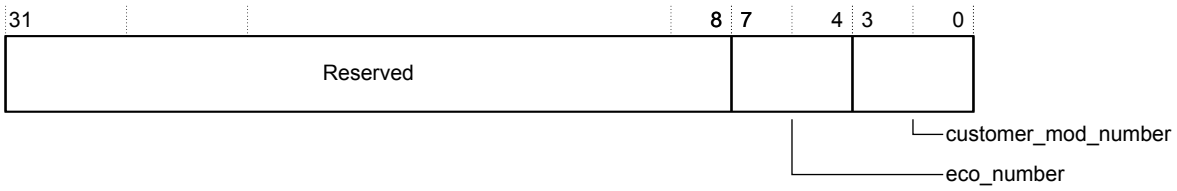

Figure 3-12 PERIPHERAL_ID3 bit assignments

The following table shows the bit descriptions.

Table 3-13 PERIPHERAL_ID3 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:4]	eco_number	Arm approved ECO number. This number has a value of `0x0`.
[3:0]	customer_mod_number	NI-700 customer modification number. This number has a value of `0x0`.

COMPONENT_ID0

This register identifies NI-700 as an Arm component.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

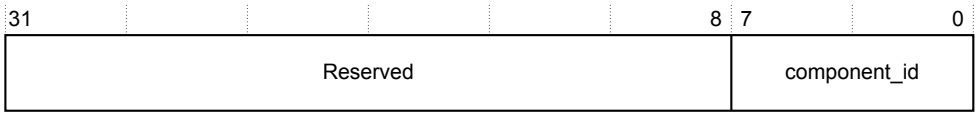

Figure 3-13 COMPONENT_ID0 bit assignments

The following table shows the bit descriptions.

Table 3-14 COMPONENT_ID0 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	component_id	The component_id identifies the NI-700 as an Arm component and has a value of `0x0D`.

COMPONENT_ID1

This register identifies NI-700 as an Arm component.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

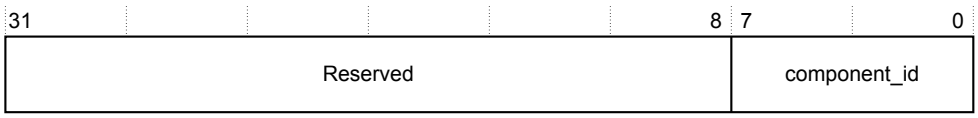

Figure 3-14 COMPONENT_ID1 bit assignments

The following table shows the bit descriptions.

Table 3-15 COMPONENT_ID1 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	component_id	The component_id identifies the NI-700 as an Arm component and has a value of `0xF`, `0x0` (Arm PrimeCell).

COMPONENT_ID2

This register identifies NI-700 as an Arm component.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

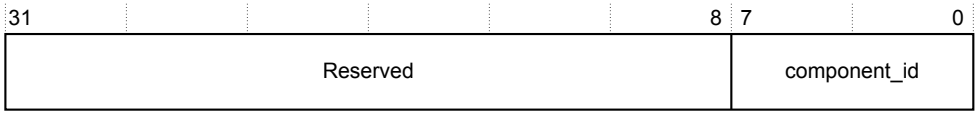

Figure 3-15 COMPONENT_ID2 bit assignments

The following table shows the bit descriptions.

Table 3-16 COMPONENT_ID2 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	component_id	The component_id identifies the NI-700 as an Arm component and has a value of `0x05`.

COMPONENT_ID3

This register identifies NI-700 as an Arm component.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.2.1 Global registers summary* on page 3-118.

The following figure shows the bit assignments.

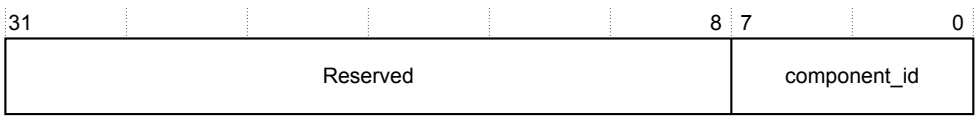

Figure 3-16 COMPONENT_ID3 bit assignments

The following table shows the bit descriptions.

Table 3-17 COMPONENT_ID3 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	component_id	The component_id identifies the NI-700 as an Arm component and has a value of `0xB1`.

3.3 Voltage domain registers

This section describes the NI-700 voltage domain registers. It contains a summary of the voltage domain registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.3.1 Voltage domain registers summary

The register summary lists the NI-700 voltage domain registers and some key characteristics.

The following table shows the voltage domain registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to your SoC implementation documentation. The offset of each register from the base address is fixed.

Table 3-18 Voltage domain registers summary

Offset	Name	Type	Reset	Width	Description
0x000	NODE_TYPE	RO	Configuration dependent	32	*NODE_TYPE, Voltage domain node type register* on page 3-129
0x004	CHILD_NODE_INFORMATION	RO	0P	32	*CHILD_NODE_INFORMATION, Child node information register, power domains* on page 3-130
0x0008-0x08FF	POWER_DOMAIN_POINTERS	RO	0P	32	*POWER_DOMAIN_POINTERS, Power domain pointers register* on page 3-130
0x0F08	SECR_ACC	RW	0x00	2	*SECR_ACC, Secure access register* on page 3-131

3.3.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

NODE_TYPE, Voltage domain node type register

This register identifies the node type as a voltage domain node.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.3.1 Voltage domain registers summary* on page 3-129

The following figure shows the bit assignments.

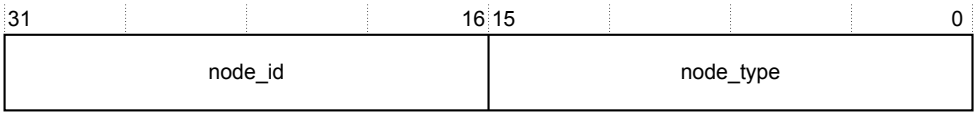

Figure 3-17 NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-19 NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The voltage domain ID that is assigned during network construction.
[15:0]	node_type	The value of this field is `0b00000001`, indicating that the associated node is a voltage domain node.

CHILD_NODE_INFORMATION, Child node information register, power domains

This register indicates the number of power domains that are present in the voltage domain.

Usage constraints
None

Configurations
Available in all NI-700 configurations

Attributes
For more information, see *3.3.1 Voltage domain registers summary* on page 3-129

The following figure shows the bit assignments.

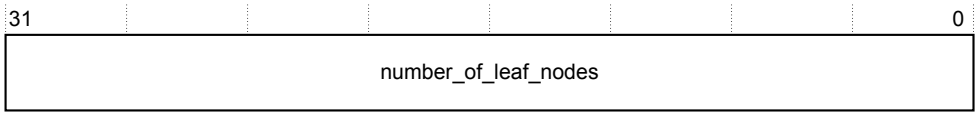

Figure 3-18 CHILD_NODE_INFORMATION bit assignments

The following table shows the bit descriptions.

Table 3-20 CHILD_NODE_INFORMATION bit descriptions

Bits	Name	Description
[31:0]	number_of_leaf_nodes	The number of power domains, leaf nodes, that are present in the voltage domain.

POWER_DOMAIN_POINTERS, Power domain pointers register

This register points to the offset from the peripheral base, for the base address of the 4KB power domain register region.

Usage constraints
None

Configurations
A copy of this register exists for each voltage domain. Available in all NI-700 configurations.

Attributes
For more information, see *3.3.1 Voltage domain registers summary* on page 3-129

The following figure shows the bit assignments.

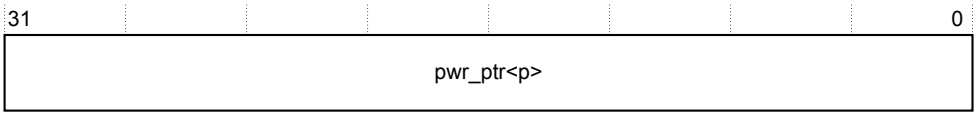

Figure 3-19 POWER_DOMAIN_POINTERS bit assignments

The following table shows the bit descriptions.

Table 3-21 POWER_DOMAIN_POINTERS bit descriptions

Bits	Name	Description
[31:0]	pwr_ptr<p>	The offset from the peripheral base, for the base address of the 4KB power domain register region.

SECR_ACC, Secure access register

This register controls whether only Non-secure transactions can read and program the NI-700 registers.

Usage constraints

Read and write to this register using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.3.1 Voltage domain registers summary* on page 3-129.

The following figure shows the bit assignments.

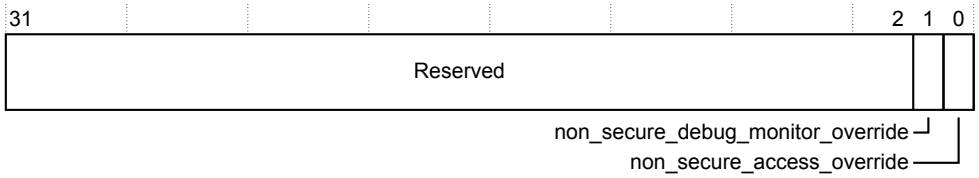

Figure 3-20 SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-22 SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	non_secure_debug_monitor_override	Debug monitor security override: **0** Disable. Non-secure access to the PMU and Interface Monitor Registers unless overridden by bit[0]. **1** Enable. Non-secure access to the PMU and Interface Monitor Registers.
[0]	non_secure_access_override	Non-secure register access override: **0** Disable. Non-secure access to the Secure registers in this register region. **1** Enable. Non-secure access to the Secure registers in this register region.

3.4 Power domain registers

This section describes the NI-700 power domain registers. It contains a summary of the power domain registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.4.1 Power domain registers summary

The register summary lists the NI-700 power domain registers and some key characteristics.

The following table shows the power domain registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to your SoC implementation documentation. The offset of each register from the base address is fixed. Some registers are only present if IDM support is enabled. The Configuration section of a register description shows this information.

Table 3-23 Power domain registers summary

Offset	Name	Type	Reset	Width	Description
0x000	NODE_TYPE	RO	Configuration dependent	32	*NODE_TYPE, Power domain node type register* on page 3-133
0x004	CHILD_NODE_INFORMATION	RO	N	32	*CHILD_NODE_INFORMATION, Child node information register, power domains* on page 3-130
0x008-0x08FF	CLOCK_DOMAIN_POINTERS	RO	P	32	*CLOCK_DOMAIN_POINTERS, Clock domain pointers register* on page 3-134
0x900	ENDPOINT_PD_IRQ_STATUS	RO	0x0	25	*ENDPOINT_PD_IRQ_STATUS, Secure transaction error status register* on page 3-135
0x904	ENDPOINT_PD_IRQ_CONTROL	RW	0x0	1	*ENDPOINT_PD_IRQ_CONTROL* on page 3-136
0x908	IDM_PD_ERROR_STATUS	RO	0x0	25	*IDM_PD_ERROR_STATUS* on page 3-136
0x90C	IDM_PD_ERROR_CONTROL	RW	0x0	1	*IDM_PD_ERROR_CONTROL* on page 3-137
0x910	IDM_PD_TIMEOUT_STATUS	RO	0x0	25	*IDM_PD_TIMEOUT_STATUS* on page 3-137
0x914	IDM_PD_TIMEOUT_CONTROL	RW	0x0	1	*IDM_PD_TIMEOUT_CONTROL* on page 3-138
0x918	IDM_PD_RESET_STATUS	RO	0x0	25	*IDM_PD_RESET_STATUS* on page 3-139
0x91C	IDM_PD_RESET_CONTROL	RW	0x0	1	*IDM_PD_RESET_CONTROL* on page 3-140
0x920	IDM_PD_ACCESS_STATUS	RO	0x0	25	*IDM_PD_ACCESS_STATUS* on page 3-140
0x924	IDM_PD_ACCESS_CONTROL	RW	0x0	1	*IDM_PD_ACCESS_CONTROL* on page 3-141

Table 3-23 Power domain registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x928	ENDPOINT_PD_IRQ_STATUS_NS	RO	0x0	25	*ENDPOINT_PD_IRQ_STATUS_NS* on page 3-142
0x92C	ENDPOINT_PD_IRQ_CONTROL_NS	RW	0x0	1	*ENDPOINT_PD_IRQ_CONTROL_NS* on page 3-142
0x930	IDM_PD_ERROR_STATUS_NS	RO	0x0	25	*IDM_PD_ERROR_STATUS_NS* on page 3-143
0x934	IDM_PD_ERROR_CONTROL_NS	RW	0x0	1	*IDM_PD_ERROR_CONTROL_NS* on page 3-144
0x938	IDM_PD_TIMEOUT_STATUS_NS	RO	0x0	25	*IDM_PD_TIMEOUT_STATUS_NS* on page 3-144
0x93C	IDM_PD_TIMEOUT_CONTROL_NS	RW	0x0	1	*IDM_PD_TIMEOUT_CONTROL_NS* on page 3-145
0x940	IDM_PD_RESET_STATUS_NS	RO	0x0	25	*IDM_PD_RESET_STATUS_NS* on page 3-146
0x944	IDM_PD_RESET_CONTROL_NS	RW	0x0	1	*IDM_PD_RESET_CONTROL_NS* on page 3-146
0x948	IDM_PD_ACCESS_STATUS_NS	RO	0x0	25	*IDM_PD_ACCESS_STATUS_NS* on page 3-147
0x94C	IDM_PD_ACCESS_CONTROL_NS	RW	0x0	1	*IDM_PD_ERROR_CONTROL_NS* on page 3-144
0x0F08	SECR_ACC	RW	0x00	2	*SECR_ACC, Secure access register* on page 3-148

3.4.2 Power domain register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

NODE_TYPE, Power domain node type register

This register identifies the node type as a power domain node.

Usage constraints
None

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

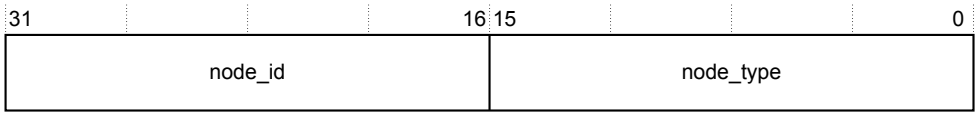

Figure 3-21 NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-24 NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The power domain ID that is assigned during network construction.
[15:0]	node_type	The value of this field is `0b00000010`, indicating that the associated node is a power domain node.

CHILD_NODE_INFORMATION, Clock domains within power domain register

This register indicates the number of clock domains that are present in the power domain.

Usage constraints

None

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

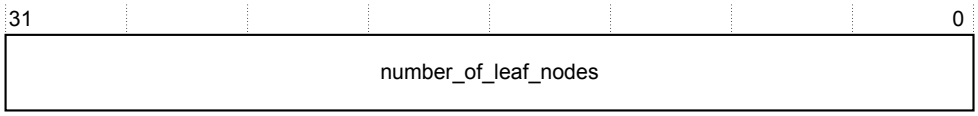

Figure 3-22 CHILD_NODE_INFORMATION bit assignments

The following table shows the bit descriptions.

Table 3-25 Child Node Information Register bit descriptions

Bits	Name	Description
[31:0]	number_of_leaf_nodes	The value of this field is the number of clock domains, leaf nodes, that are present in the power domain.

CLOCK_DOMAIN_POINTERS, Clock domain pointers register

This register points to the offset from the peripheral base, for the base address of the 4KB clock domain register region of the power domain.

Usage constraints

None

Configurations

A copy of this register exists for each clock domain within a given power domain.
Available in all NI-700 configurations.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

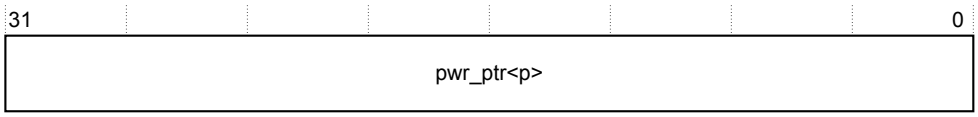

Figure 3-23 CLOCK_DOMAIN_POINTERS bit assignments

The following table shows the bit descriptions.

Table 3-26 CLOCK_DOMAIN_POINTERS bit descriptions

Bits	Name	Description
[31:0]	pwr_ptr<p>	Offset from the peripheral base, for the base address of the 4KB clock domain register region of the power domain.

ENDPOINT_PD_IRQ_STATUS, Secure transaction error status register

This register, which is IDM-related, indicates the error status of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

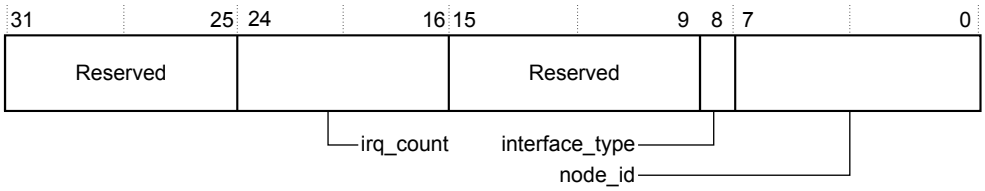

Figure 3-24 ENDPOINT_PD_IRQ_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-27 ENDPOINT_PD_IRQ_STATUS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	irq_count	The number of interfaces currently asserting error interrupt.
[15:9]	-	Reserved, *undefined*, write as zero
[8]	interface_type	The type of interface that the node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The node ID of the first interface raising an error interrupt.

ENDPOINT_PD_IRQ_CONTROL

This register, which is IDM-related, controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

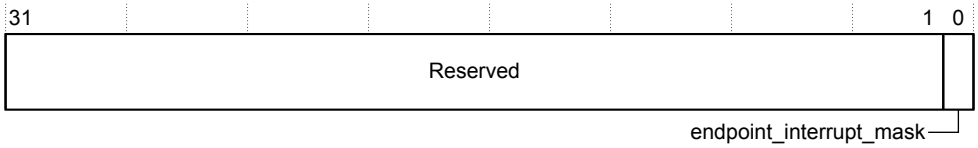

Figure 3-25 ENDPOINT_PD_IRQ_CONTROL bit assignments

The following table shows the bit descriptions.

Table 3-28 ENDPOINT_PD_IRQ_CONTROL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	endpoint_interrupt_mask	When set to 1, enables mask of all error interrupts.

IDM_PD_ERROR_STATUS

This register indicates the error status of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

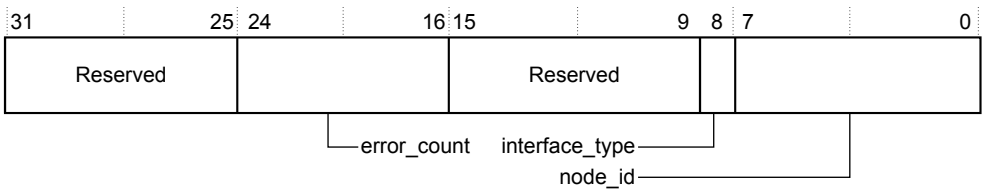

Figure 3-26 IDM_PD_ERROR_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-29 IDM_PD_ERROR_STATUS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The number of interfaces currently asserting error interrupt.
[15:9]	-	Reserved, UNDEFINED, write as zero
[8]	interface_type	The type of interface that the Node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The Node ID of the first interface raising an error interrupt.

IDM_PD_ERROR_CONTROL

This register controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

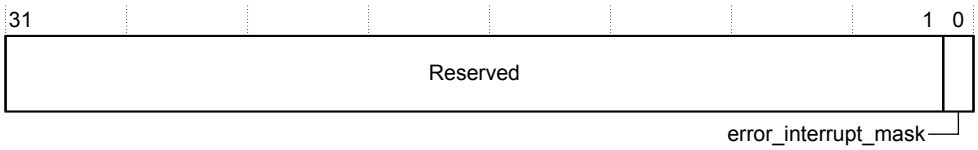

Figure 3-27 IDM_PD_ERROR_CONTROL bit assignments

The bit descriptions are shown in the following table.

Table 3-30 IDM_PD_ERROR_CONTROL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	error_interrupt_mask	When set to 1, enables mask of all error interrupts.

IDM_PD_TIMEOUT_STATUS

This register indicates the timeout status of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

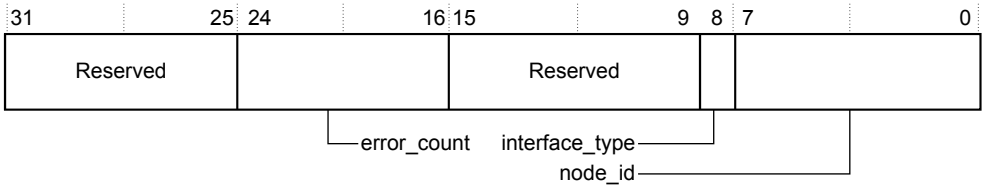

Figure 3-28 IDM_PD_TIMEOUT_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-31 IDM_PD_TIMEOUT_STATUS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The number of interfaces currently asserting timeout interrupt.
[15:9]	-	Reserved, UNDEFINED, write as zero
[8]	interface_type	Indicates the type of interface that is specified by the Node ID: **0** Slave **1** Master
[7:0]	node_id	The Node ID of the first interface raising a timeout interrupt.

IDM_PD_TIMEOUT_CONTROL

This register controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

See *3.4.1 Power domain registers summary* on page 3-132 for more information

The following figure shows the bit assignments.

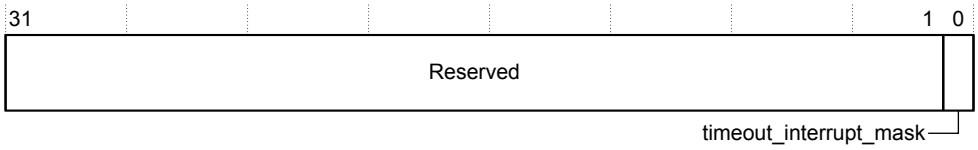

Figure 3-29 IDM_PD_TIMEOUT_CONTROL bit assignments

The following table shows the bit descriptions.

Table 3-32 IDM_PD_TIMEOUT_CONTROL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	timeout_interrupt_mask	When set to 1, enables mask of all error interrupts.

IDM_PD_RESET_STATUS

This register indicates the reset access status of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

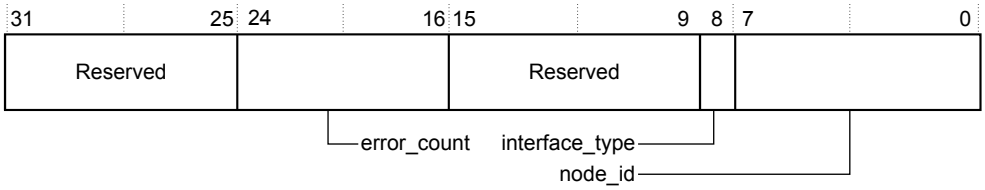

Figure 3-30 IDM_PD_RESET_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-33 IDM_PD_RESET_STATUS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The anount of interfaces currently asserting error interrupt.
[15:9]	-	Reserved, UNDEFINED, write as zero

Table 3-33 IDM_PD_RESET_STATUS bit descriptions (continued)

Bits	Name	Description
[8]	interface_type	The type of interface the Node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The Node ID of the first interface raising an activity while in reset interrupt.

IDM_PD_RESET_CONTROL

This register controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

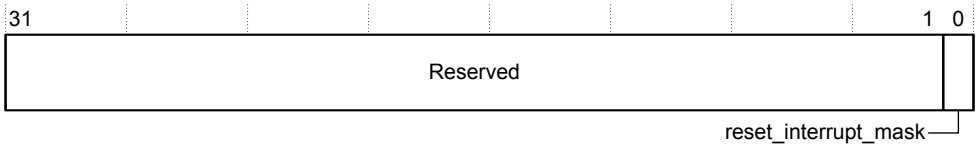

Figure 3-31 IDM_PD_RESET_CONTROL bit assignments

The following table shows the bit descriptions.

Table 3-34 IDM_PD_RESET_CONTROL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	reset_interrupt_mask	When set to 1, enables mask of all error interrupts.

IDM_PD_ACCESS_STATUS

This register indicates the isolation access status of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

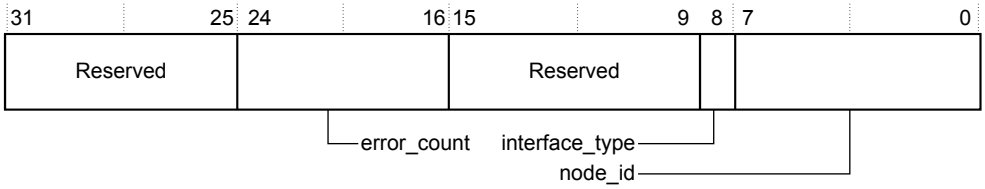

Figure 3-32 IDM_PD_ACCESS_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-35 IDM_PD_ACCESS_STATUS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The amount of interfaces currently asserting error interrupt.
[15:9]	-	Reserved, UNDEFINED, write as zero
[8]	interface_type	The type of interface that the Node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The Node ID of the first interface raising an activity while in reset interrupt.

IDM_PD_ACCESS_CONTROL

This register controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

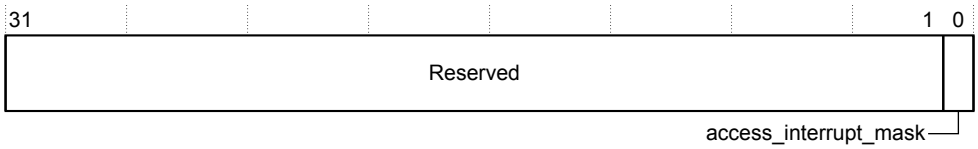

Figure 3-33 IDM_PD_ACCESS_CONTROL bit assignments

The following table shows the bit descriptions.

Table 3-36 IDM_PD_ACCESS_CONTROL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	access_interrupt_mask	When set to 1, enables mask of all error interrupts.

ENDPOINT_PD_IRQ_STATUS_NS

This register, which is IDM related, indicates the error status of Non-secure transactions.

Usage constraints

None

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

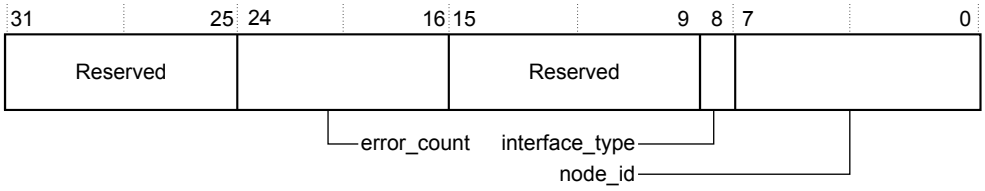

Figure 3-34 ENDPOINT_PD_IRQ_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-37 ENDPOINT_PD_IRQ_STATUS_NS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The number of interfaces currently asserting error interrupt.
[15:9]	-	Reserved, UNDEFINED, write as zero
[8]	interface_type	The type of interface the Node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The Node ID of the first interface raising an error interrupt.

ENDPOINT_PD_IRQ_CONTROL_NS

This register, which is IDM related, controls the interrupts of Non-secure transactions.

Usage constraints

None

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

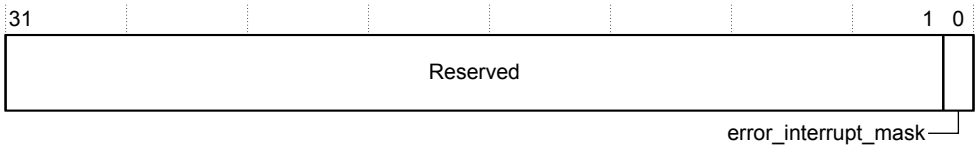

Figure 3-35 ENDPOINT_PD_IRQ_CONTROL_NS bit assignments

The following table shows the bit descriptions.

Table 3-38 ENDPOINT_PD_IRQ_ERROR_CONTROL_NS bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	error_interrupt_mask	When set to 1, enables mask of all error interrupts.

IDM_PD_ERROR_STATUS_NS

This register indicates the error status of Non-secure transactions.

Usage constraints

None

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

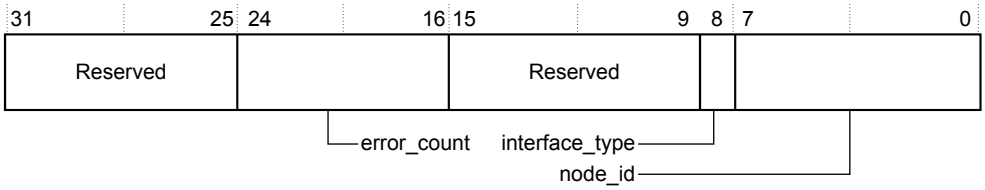

Figure 3-36 IDM_PD_ERROR_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-39 IDM_PD_ERROR_STATUS_NS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The amount of interfaces currently asserting error interrupt.

Table 3-39 IDM_PD_ERROR_STATUS_NS bit descriptions (continued)

Bits	Name	Description
[15:9]	-	Reserved, UNDEFINED, write as zero
[8]	interface_type	The type of interface that the Node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The Node ID of the first interface raising an error interrupt.

IDM_PD_ERROR_CONTROL_NS

This register controls the interrupts of Non-secure transactions.

Usage constraints

None

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132

The following figure shows the bit assignments.

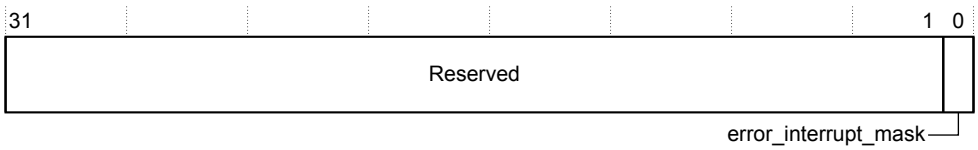

Figure 3-37 IDM_PD_ERROR_CONTROL_NS bit assignments

The following table shows the bit descriptions.

Table 3-40 IDM_PD_ERROR_CONTROL_NS bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	error_interrupt_mask	When set to 1, enables mask of all error interrupts.

IDM_PD_TIMEOUT_STATUS_NS

This register indicates the timeout status of Non-secure transactions.

Usage constraints

None

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132.

The following figure shows the bit assignments.

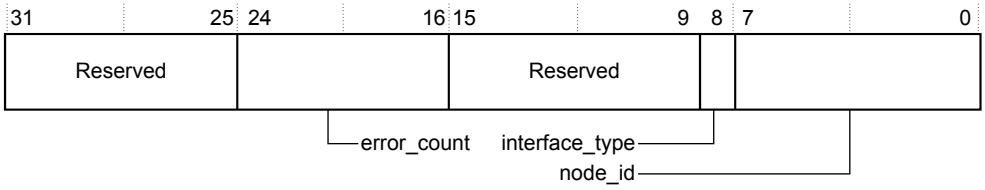

Figure 3-38 IDM_PD_TIMEOUT_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-41 IDM_PD_TIMEOUT_STATUS_NS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The amount of interfaces currently asserting timeout interrupt.
[15:9]	-	Reserved, UNDEFINED, write as zero
[8]	interface_type	The type of interface that the node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The Node ID of the first interface raising a timeout interrupt.

IDM_PD_TIMEOUT_CONTROL_NS

This register controls the interrupts of Non-secure transactions.

Usage constraints

None

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132.

The following figure shows the bit assignments.

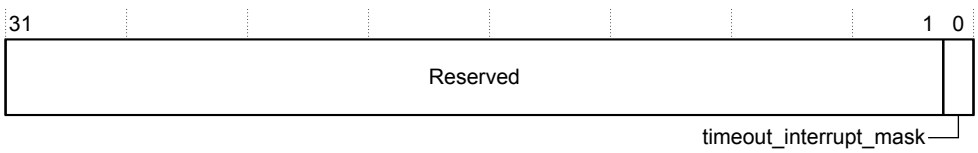

Figure 3-39 IDM_PD_TIMEOUT_CONTROL_NS bit assignments

The following table shows the bit descriptions.

Table 3-42 IDM_PD_TIMEOUT_CONTROL_NS bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	timeout_interrupt_mask	When set to 1, enable mask of all timeout interrupts.

IDM_PD_RESET_STATUS_NS

This register indicates the reset access status of Non-secure transactions.

Usage constraints

None

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132.

The following figure shows the bit assignments.

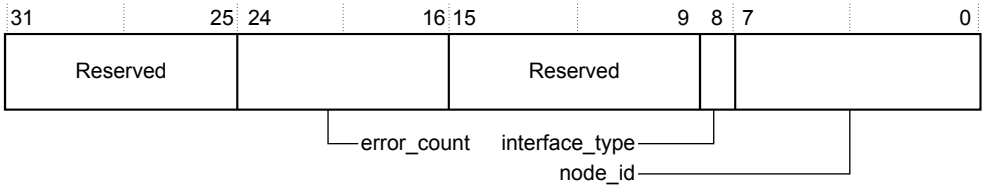

Figure 3-40 IDM_PD_RESET_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-43 IDM_PD_RESET_STATUS_NS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The number of interfaces currently asserting error interrupt.
[15:9]	-	Reserved, UNDEFINED, write as zero
[8]	interface_type	The type of interface that the node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The node ID of the first interface raising an activity while in reset interrupt.

IDM_PD_RESET_CONTROL_NS

This register controls the interrupts of Non-secure transactions.

Usage constraints

None

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132.

The following figure shows the bit assignments.

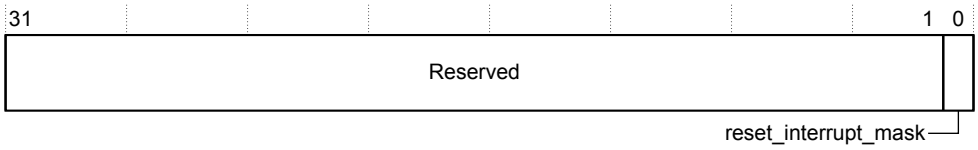

Figure 3-41 IDM_PD_RESET_CONTROL_NS bit assignments

The following table shows the bit descriptions.

Table 3-44 IDM_PD_RESET_CONTROL_NS bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	reset_interrupt_mask	When set to 1, enables mask of all reset interrupts.

IDM_PD_ACCESS_STATUS_NS

This register indicates the isolation access status of Non-secure transactions.

Usage constraints

None

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132.

The following figure shows the bit assignments.

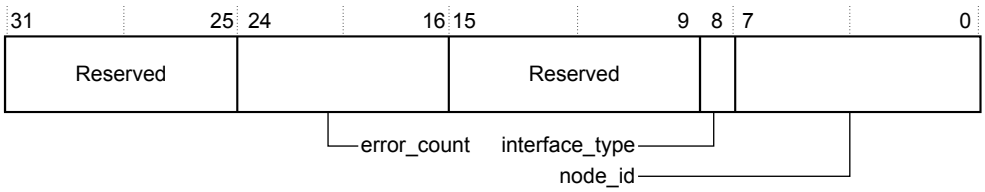

Figure 3-42 IDM_PD_ACCESS_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-45 IDM_PD_ACCESS_STATUS_NS bit descriptions

Bits	Name	Description
[31:25]	-	Reserved, UNDEFINED, write as zero
[24:16]	error_count	The number of interfaces currently asserting error interrupt.
[15:9]	-	Reserved, UNDEFINED, write as zero
[8]	interface_type	The type of interface that the node ID specifies: **0** Slave **1** Master
[7:0]	node_id	The node ID of the first interface raising an activity while in isolation interrupt.

IDM_PD_ACCESS_CONTROL_NS

This register controls the interrupts of Non-secure transactions.

Usage constraints

None

Configurations

This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132.

The following figure shows the bit assignments.

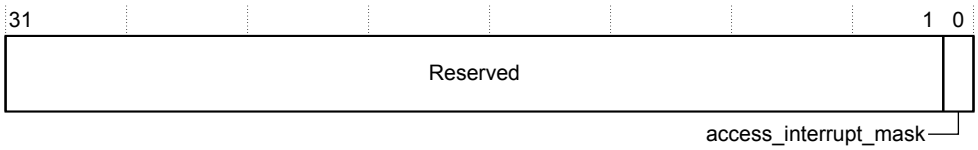

Figure 3-43 IDM_PD_ACCESS_CONTROL_NS bit assignments

The following table shows the bit descriptions.

Table 3-46 IDM_PD_ACCESS_CONTROL_NS bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	access_interrupt_mask	When set to 1, enables mask of all error interrupts.

SECR_ACC, Secure access register

This register controls whether only Non-secure transactions can read and program the NI-700 registers.

Usage constraints

Read and write to this register using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.4.1 Power domain registers summary* on page 3-132.

The following figure shows the bit assignments.

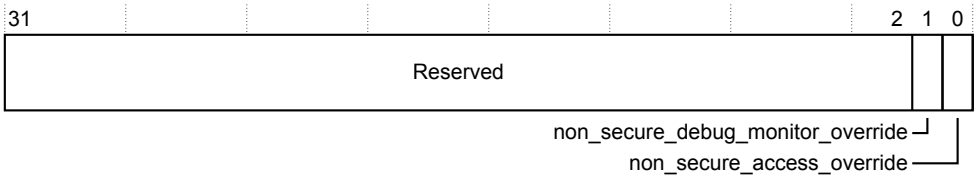

Figure 3-44 SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-47 SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	non_secure_debug_monitor_override	Debug monitor security override: **0** Disable. Non-secure access to the PMU and Interface Monitor Registers unless overridden by bit[0]. **1** Enable. Non-secure access to the PMU and Interface Monitor Registers.
[0]	non_secure_access_override	Non-secure register access override: **0** Disable. Non-secure access to the Secure registers in this register region. **1** Enable. Non-secure access to the Secure registers in this register region.

3.5 Clock domain registers

This section describes the NI-700 clock domain registers. It contains a summary of the clock domain registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.5.1 Clock domain registers summary

The register summary lists the NI-700 clock domain registers and some key characteristics.

The following table shows the clock domain registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. Consult your SoC implementation documentation for more information. The offset of each register from the base address is fixed.

Table 3-48 Clock domain registers summary

Offset	Name	Type	Reset	Width	Description
`0x000`	NODE_TYPE	RO	Configuration dependent	32	*NODE_TYPE, Clock domain node type register* on page 3-150
`0x004`	CHILD_NODE_INFORMATION	RO	N	32	*CHILD_NODE_INFORMATION, Child node information register, network components* on page 3-151
`0x0008-0x08FF`	COMPONENT_NODE	RO	P	32	*COMPONENT_NODE, Component node pointers register* on page 3-151
`0x0F08`	SECR_ACC	RW	`0x00`	2	*SECR_ACC, Secure access register* on page 3-152

3.5.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

NODE_TYPE, Clock domain node type register

This register identifies the node type as a NI-700 clock domain register node.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.5.1 Clock domain registers summary* on page 3-150.

The following figure shows the bit assignments.

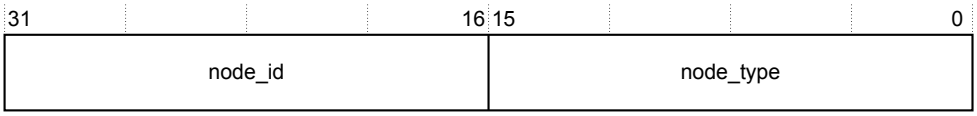

Figure 3-45 NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-49 NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The clock domain ID that is assigned during network construction.
[15:0]	node_type	The value of this field is `0b00000011`, indicating that the associated node contains clock domain registers for a particular NI-700 power domain.

CHILD_NODE_INFORMATION, Child node information register, network components

This register indicates the number of network components, that is, leaf nodes, that are present in the clock domain.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.5.1 Clock domain registers summary* on page 3-150.

The following figure shows the bit assignments.

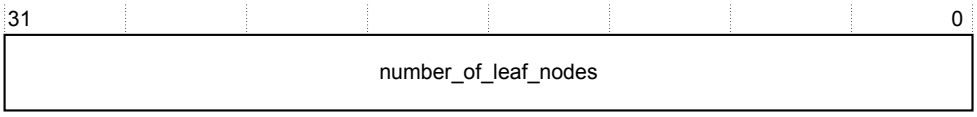

Figure 3-46 CHILD_NODE_INFORMATION bit assignments

The following table shows the bit descriptions.

Table 3-50 CHILD_NODE_INFORMATION bit descriptions

Bits	Name	Description
[31:0]	number_of_leaf_nodes	The value of this field is the number of network components, leaf nodes, that are present in the clock domain.

COMPONENT_NODE, Component node pointers register

This register points to the offset from the peripheral base, for the base address of the 4KB component register region of the clock domain.

Usage constraints

None.

Configurations

A copy of this register exists for each component node within the given clock domain.
Available in all NI-700 configurations.

Attributes

For more information, see *3.5.1 Clock domain registers summary* on page 3-150.

The following figure shows the bit assignments.

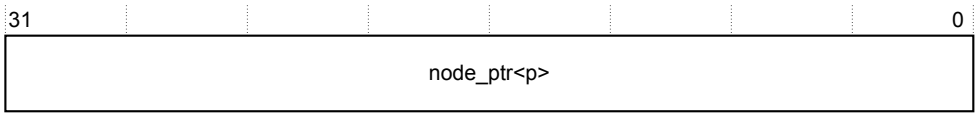

Figure 3-47 COMPONENT_NODE bit assignments

The following table shows the bit descriptions.

Table 3-51 COMPONENT_NODE bit descriptions

Bits	Name	Description
[31:0]	node_ptr<p>	A pointer to the offset from the peripheral base, for the base address of the 4KB component register region of the clock domain.

SECR_ACC, Secure access register

This register controls whether only Non-secure transactions can read and program the NI-700 registers.

Usage constraints

Read and write to this register using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.5.1 Clock domain registers summary* on page 3-150.

The following figure shows the bit assignments.

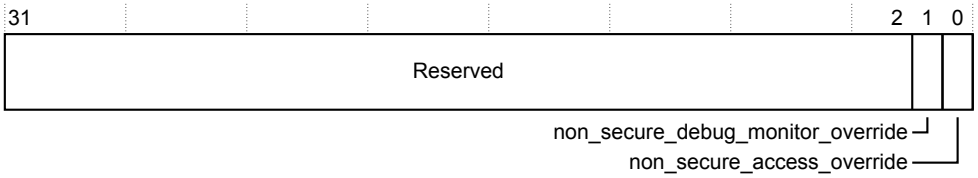

Figure 3-48 SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-52 SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	non_secure_debug_monitor_override	Debug monitor security override: **0** Disable. Non-secure access to the PMU and Interface Monitor Registers unless overridden by bit[0]. **1** Enable. Non-secure access to the PMU and Interface Monitor Registers.
[0]	non_secure_access_override	Non-secure register access override: **0** Disable. Non-secure access to the Secure registers in this register region. **1** Enable. Non-secure access to the Secure registers in this register region.

3.6 Performance Monitoring Unit registers

This section describes the NI-700 PMU registers. It contains a summary of the PMU registers in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.6.1 Performance Monitoring Unit registers summary

The register summary lists the NI-700 PMU registers and some key characteristics.

The PMU registers are shown in the following table in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to SoC implementation documentation. The offset of each register from the base address is fixed.

Table 3-53 PMU registers summary

Address offset	Name	Type	Description
0x000	NODE_TYPE	RO	*NODE_TYPE, Node type register for PMU registers* on page 3-154
0x004	SECR_ACC	RW	*SECR_ACC, Secure access register* on page 3-155
0x008	PMEVCNTRn	RW	*PMEVCNTRn, Performance monitor event counter registers* on page 3-156
0x010			
0x018			
0x020			
0x028			
0x030			
0x038			
0x040			
0x0F8	PMCCNTR_lower	RW	*PMCCNTR_lower, Performance monitors cycle counter register* on page 3-156
0x0FC	PMCCNTR_upper	RW	*PMCCNTR_upper, Performance monitors cycle counter register* on page 3-157
0x400	PMEVTYPERn	RW	*PMEVTYPERn, Performance monitor event type and filter registers* on page 3-157
0x404			
0x408			
0x40C			
0x410			
0x414			
0x418			
0x41C			
0x610	PMSSSR	RO	*PMSSR, PMU snapshot status register* on page 3-158
0x614	PMOVSSR	RO	*PMOVSSR, PMU overflow status snapshot register* on page 3-159
0x618	PMCCNTSR_lower	RO	*PMCCNTSR_lower, Cycle counter snapshot register* on page 3-159
0x61C	PMCCNTSR_upper	RO	*PMCCNTSR_upper, Cycle counter snapshot register* on page 3-160

Table 3-53 PMU registers summary (continued)

Address offset	Name	Type	Description
0x620	PMEVCNTSRn	RO	*PMEVCNTSRn, PMU event counter snapshot registers* on page 3-160
0x624			
0x628			
0x62C			
0x630			
0x634			
0x638			
0x63C			
0x6F0	PMSSCR	WO	*PMSSCR, Performance monitors snapshot capture register* on page 3-161
0xC00	PMCNTENSET	RW	*PMCNTENSET, Performance monitors count enable set register* on page 3-162
0xC20	PMCNTENCLR	RW	*PMCNTENCLR, Performance monitors count enable clear register* on page 3-162
0xC40	PMINTENSET	RW	*PMINTENSET, Performance monitors interrupt enable set register* on page 3-163
0xC60	PMINTENCLR	RW	*PMINTENCLR, Performance monitors interrupt enable clear register* on page 3-164
0xC80	PMOVSCLR	RW	*PMOVSCLR, Performance monitors overflow flag status clear register,* on page 3-165
0xCC0	PMOVSSET	RW	*PMOVSSET, Performance monitors overflow flag status set register* on page 3-166
0xD80	PMCCCGR	RW	*PMCCCGR, Performance monitors cycle counter clock gating* on page 3-167
0xE00	PMCFGR	RO	*PMCFGR, Performance monitors configuration register* on page 3-168
0xE04	PMCR	RW/WO	*PMCR, Performance monitors control register* on page 3-169

3.6.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

NODE_TYPE, Node type register for PMU registers

This register identifies the node type as a NI-700 node for PMU registers.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

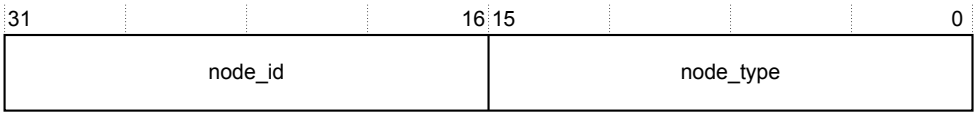

Figure 3-49 NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-54 NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The PMU ID that is assigned during network construction.
[15:0]	node_type	The value of this field is `0x06` and identifies the associated node type as a node for the NI-700 PMU registers.

SECR_ACC, Secure access register

This register controls whether only Non-secure transactions can read and program the NI-700 registers.

Usage constraints

You can read this register using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary on page 3-153*.

The following figure shows the bit assignments.

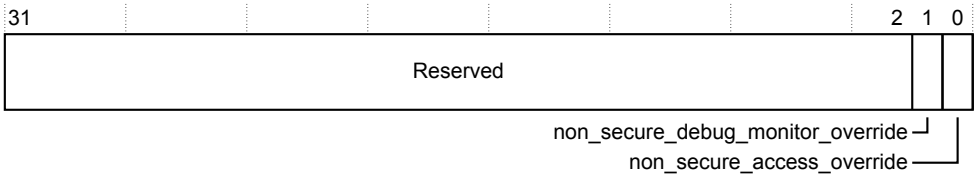

Figure 3-50 SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-55 SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	non_secure_debug_monitor_override	Debug monitor security override: **0** Disable. Non-secure access to the NI-700 PMU and Interface Registers. **1** Enable. Non-secure access to the NI-700 PMU and Interface Registers.
[0]	non_secure_access_override	Non-secure access override: **0** Disable. Non-secure access to the NI-700 registers. **1** Enable. Non-secure access to the NI-700 registers.

PMEVCNTRn, Performance monitor event counter registers

Registers PMEVCNTR0-PMEVCNTR07 record performance events that occur within each clock domain in the NI-700 system.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

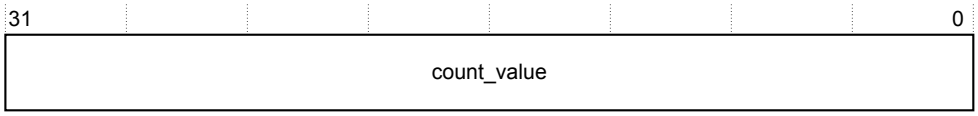

Figure 3-51 PMEVCNTRn bit assignments

The following table shows the bit descriptions.

Table 3-56 PMEVCNTRn bit descriptions

Bits	Name	Description
[31:0]	count_value	The recorded number of program-specified events that have occurred in the clock domain within a programmed period. An event can fire no more than one time in each cycle.

PMCCNTR_lower, Performance monitors cycle counter register

This register contains the value of lower 64-bit cycle counter [31:0].

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

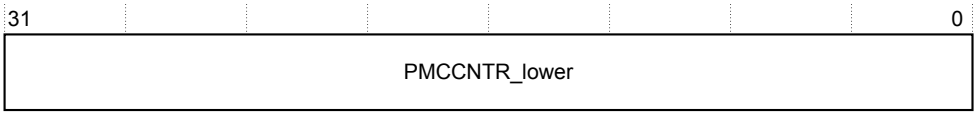

Figure 3-52 PMCCNTR_lower bit assignments

The following table shows the bit descriptions.

Table 3-57 PMCCNTR_lower bit descriptions

Bits	Name	Description
[31:0]	PMCCNTR_lower	The value of lower 64-bit cycle counter [31:0]

PMCCNTR_upper, Performance monitors cycle counter register

This register contains the value of upper 64-bit cycle counter [63:32].

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

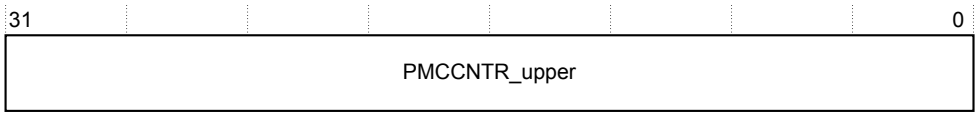

Figure 3-53 PMCCNTR_upper bit assignments

The following table shows the bit descriptions.

Table 3-58 PMCCNTR_upper bit descriptions

Bits	Name	Description
[31:0]	PMCCNTR_lower	The value of upper 64-bit cycle counter [63:32]

PMEVTYPERn, Performance monitor event type and filter registers

Registers PMEVTYPER0-7 control the performance monitor event counter start and stop period, event type, and type and ID of the target node.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

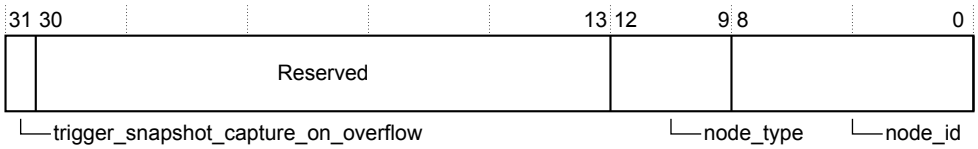

Figure 3-54 PMEVTYPERn bit assignments

The following table shows the bit descriptions.

Table 3-59 PMEVTYPERn bit descriptions

Bits	Name	Description
[31]	trigger_snapshot_capture_on_overflow	Counter enable: **0** Trigger snapshot capture on overflow disabled. **1** Trigger snapshot capture on overflow enabled.
[30:13]	-	Reserved
[12:9]	node_type	The node type.
[8:0]	node_id	The Node ID.

For the performance events, see *Chapter 4 Performance monitoring* on page 4-292.

PMSSR, PMU snapshot status register

This register records the status of a performance event counter when the **<CLKNAME>_PMUSNAPSHOTREQ** input signal enables it.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.
A copy of this register exists for each performance event counter.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

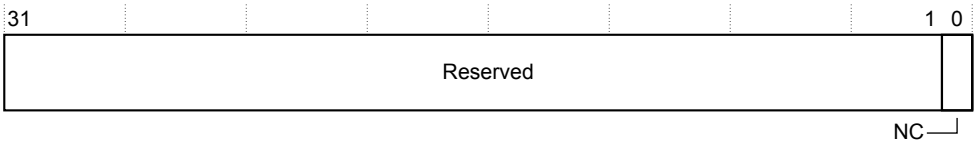

Figure 3-55 PMSSR bit assignments

The following table shows the bit descriptions.

Table 3-60 PMSSR bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	NC	No capture. Indicates whether the PMU counters have been captured. The values are: **0** PMU counters are captured. **1** PMU counters are not captured.

PMOVSSR, PMU overflow status snapshot register

This register records the overflow status of a performance event counter when the **<CLKNAME>_PMUSNAPSHOTREQ** input signal enables it.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.
A copy of this register exists for each performance event counter.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

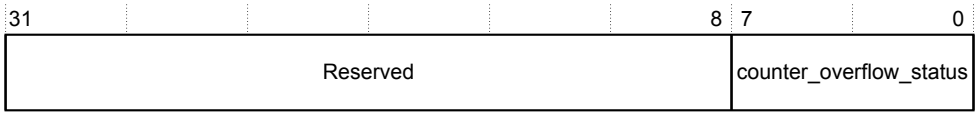

Figure 3-56 PMOVSSR bit assignments

The following table shows the bit descriptions.

Table 3-61 PMOVSSR bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	counter_overflow_status	Counter overflow status.

PMCCNTSR_lower, Cycle counter snapshot register

This register contains the snapshot value of the lower 64-bit cycle counter [31:0].

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary on page 3-153*.

The following figure shows the bit assignments.

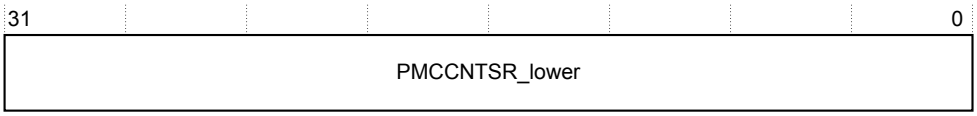

Figure 3-57 PMCCNTSR_lower bit assignments

The following table shows the bit descriptions.

Table 3-62 PMCCNTSR_lower bit descriptions

Bits	Name	Description
[31:0]	PMCCNTSR_lower	The snapshot value of the lower 64-bit cycle counter [31:0]

PMCCNTSR_upper, Cycle counter snapshot register

This register contains the snapshot value of the upper 64-bit cycle counter [63:32].

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register on page 3-155* to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary on page 3-153*.

The following figure shows the bit assignments.

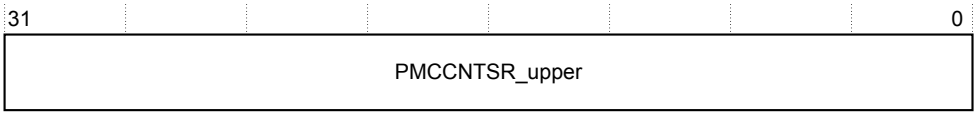

Figure 3-58 PMCCNTSR_upper bit assignments

The following table shows the bit descriptions.

Table 3-63 PMCCNTSR_upper bit descriptions

Bits	Name	Description
[31:0]	PMCCNTSR_upper	The snapshot value of the upper 64-bit cycle counter [63:32]

PMEVCNTSRn, PMU event counter snapshot registers

PMEVCNTSR0-7 are shadow registers that record an Event counter *n* snapshot value of the performance event counters when the **<CLKNAME>_PMUSNAPSHOTREQ** input signal enables them.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

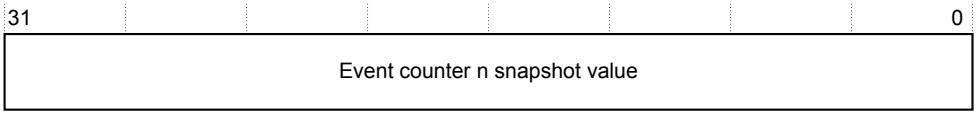

Figure 3-59 PMEVCNTSRn bit assignments

The following table shows the bit descriptions.

Table 3-64 PMEVCNTSRn bit descriptions

Bits	Name	Description
[31:0]	Event counter *n* snapshot value	The event counter *n* snapshot value.

PMSSCR, Performance monitors snapshot capture register

This register captures a snapshot of the performance monitors.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

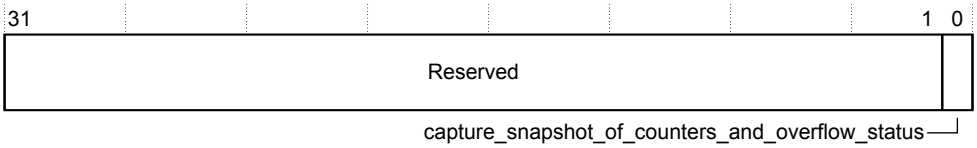

Figure 3-60 PMSSCR bit assignments

The following table shows the bit descriptions.

Table 3-65 PMSSCR bit descriptions

Bits	Name	Description
[31:1]	-	Reserved.
[0]	capture_snapshot_of_counters_and_overflow_status	The capture snapshot of counters and overflow status.

PMCNTENSET, Performance monitors count enable set register

This register sets the performance monitors count enable.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

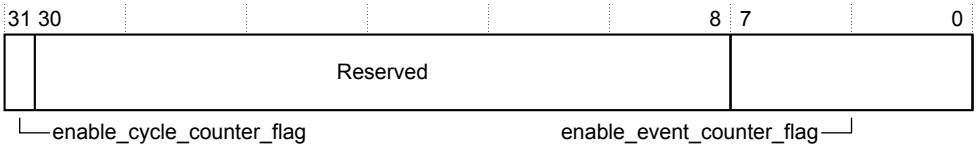

Figure 3-61 PMCNTENSET bit assignments

The following table shows the bit descriptions.

Table 3-66 PMCNTENSET bit descriptions

Bits	Name	Description
[31]	enable_cycle_counter_flag	The PMCCNTR enable bit. Enables the cycle counter register. The values are: **0** When read, means that the cycle counter is disabled. When written, has no effect. **1** When read, means that the cycle counter is enabled. When written, enables the cycle counter.
[30:8]	-	Reserved
[7:0]	enable_event_counter_flag	The event counter enable bit for PMEVCNTR<x>. The values are: **0** When read, means that the PMEVCNTR<x> is disabled. When written, has no effect. **1** When read, means that the PMEVCNTR<x> event counter is enabled. When written, enables PMEVCNTR<x>.

PMCNTENCLR, Performance monitors count enable clear register

This register clears the performance monitors count enable.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

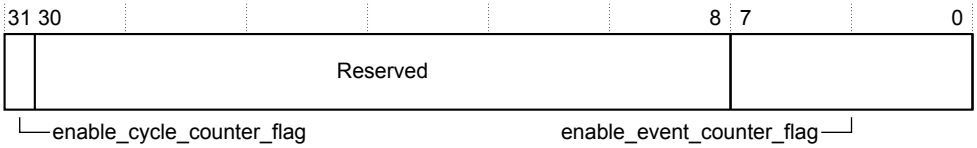

Figure 3-62 PMCNTENCLR bit assignments

The following table shows the bit descriptions.

Table 3-67 PMCNTENCLR bit descriptions

Bits	Name	Description
[31]	enable_cycle_counter_flag	The PMCCNTR disable bit. Disables the cycle counter register. The values are: **0** When read, means that the cycle counter is disabled. When written, has no effect. **1** When read, means that the cycle counter is enabled. When written, disables the cycle counter.
[30:8]	-	Reserved
[7:0]	enable_event_counter_flag	The Event counter disable bit for PMEVCNTR<x>. The values are: **0** When read, means that PMEVCNTR<x> is disabled. When written, has no effect. **1** When read, means that PMEVCNTR<x> is enabled. When written, disables PMEVCNTR<x>.

PMINTENSET, Performance monitors interrupt enable set register

This register sets the performance monitors interrupt enable.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

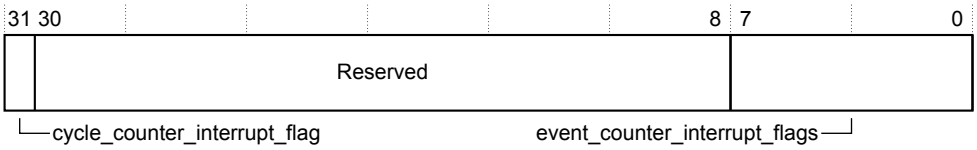

Figure 3-63 PMINTENSET bit assignments

The following table shows the bit descriptions.

Table 3-68 PMINTENSET bit descriptions

Bits	Name	Description
[31]	cycle_counter_interrupt_flag	The PMCCNTR overflow interrupt request enable bit. The values are: **0** When read, means that the cycle counter overflow interrupt request is disabled. When written, has no effect. **1** When read, means that the cycle counter overflow interrupt request is enabled. When written, enables the cycle count overflow interrupt request.
[30:8]	-	Reserved
[7:0]	event_counter_interrupt_flags	Event counter overflow interrupt request enable bit for PMEVCNTR<x>. The values are as follows: **0** When read, means that the PMEVCNTR<x>_EL0 event counter interrupt request is disabled. When written, has no effect. **1** When read, means that the PMEVCNTR<x>_EL0 event counter interrupt request is enabled. When written, enables the PMEVCNTR<x>_EL0 interrupt request.

PMINTENCLR, Performance monitors interrupt enable clear register

This register clears the performance monitors interrupt enable.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

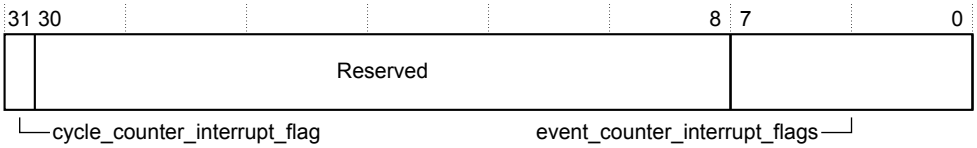

Figure 3-64 PMINTENCLR bit assignments

The following table shows the bit descriptions.

Table 3-69 PMINTENCLR bit descriptions

Bits	Name	Description
[31]	cycle_counter_interrupt_flag	The PMCCNTR overflow interrupt request disable bit. The values are: **0** When read, means that the cycle counter overflow interrupt request is disabled. When written, has no effect. **1** When read, means that the cycle counter overflow interrupt request is enabled. When written, disables the cycle count overflow interrupt request.
[30:8]	-	Reserved
[7:0]	event_counter_interrupt_flags	The event counter overflow interrupt request disable bit for PMEVCNTR<x>. The values are: **0** When read, means that the PMEVCNTR<x> event counter interrupt request is disabled. When written, has no effect. **1** When read, means that the PMEVCNTR<x> event counter interrupt request is enabled. When written, disables the PMEVCNTR<x> interrupt request.

PMOVSCLR, Performance monitors overflow flag status clear register,

This register clears the performance monitors overflow flag status.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

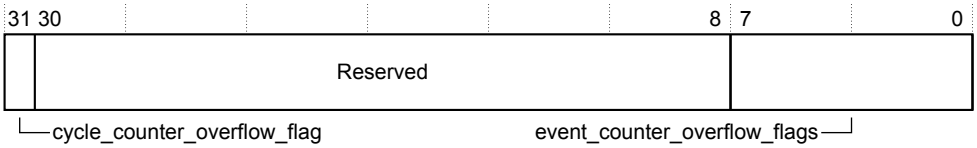

Figure 3-65 PMOVSCLR bit assignments

The following table shows the bit descriptions.

Table 3-70 PMOVSCLR bit descriptions

Bits	Name	Description
[31]	cycle_counter_overflow_flag	The PMCCNTR cycle counter overflow bit. The values are: **0** When read, means that the cycle counter has not overflowed. When written, has no effect. **1** When read, means that the cycle counter has overflowed. When written, clears the overflow bit to 0.
[30:8]	-	Reserved
[7:0]	event_counter_overflow_flags	The event counter overflow clear bit for PMEVCNTR. The values are: **0** When read, means that PMEVCNTR<x> has not overflowed. When written, has no effect. **1** When read, means that PMEVCNTR<x> has overflowed. When written, clears the PMEVCNTR<x> overflow bit to 0.

PMOVSSET, Performance monitors overflow flag status set register

This register sets the performance monitors overflow flag status.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

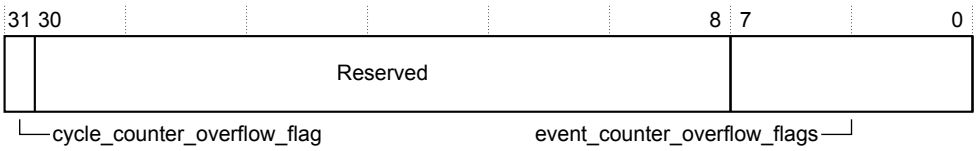

Figure 3-66 PMOVSSET bit assignments

The following table shows the bit descriptions.

Table 3-71 PMOVSSET bit descriptions

Bits	Name	Description
[31]	cycle_counter_overflow_flag	The PMCCNTR cycle counter overflow bit. The values are: **0** When read, means that the cycle counter has not overflowed. When written, has no effect. . **1** When read, means that the cycle counter has overflowed. When written, sets the overflow bit to 1.
[30:8]	-	Reserved
[7:0]	event_counter_overflow_flags	The event counter overflow set bit for PMEVCNTR<x>. The values are: **0** When read, it means that PMEVCNTR<x> has not overflowed. When written, it has no effect. **1** When read, it means that PMEVCNTR<x> has overflowed. When written, it sets the PMEVCNTR<x> overflow bit to 1.

PMCCCGR, Performance monitors cycle counter clock gating

This register sets the performance monitors overflow flag status.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

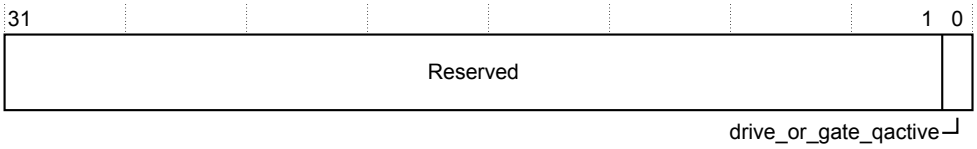

Figure 3-67 PMCCCGR bit assignments

The following table shows the bit descriptions.

Table 3-72 PMCCCGR bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	drive_or_gate_qactive	Defines whether to drive or gate the **QACTIVE**signal. **0** Gate the **QACTIVE** signal for clock domain when no events are present. **1** Drive the **QACTIVE** signal for clock domain when no events are present.

PMCFGR, Performance monitors configuration register

This register controls the performance monitors.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

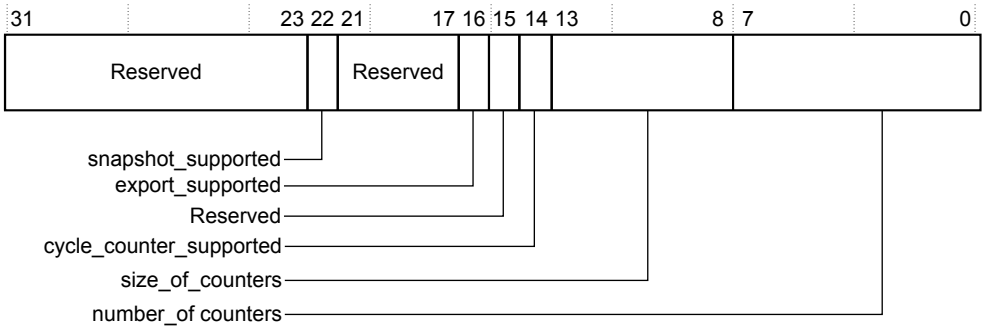

Figure 3-68 PMCFGR bit assignments

The following table shows the bit descriptions.

Table 3-73 PMCFGR bit descriptions

Bits	Name	Description
[31:23]	-	Reserved
[22]	snapshot_supported	Always 1
[21:17]	-	Reserved
[16]	export_supported	Always 1
[15]	-	Reserved
[14]	cycle_counter_supported	Always 1

Table 3-73 PMCFGR bit descriptions (continued)

Bits	Name	Description
[13:8]	size_of_counters	Always `0b111111` (SIZE)
[7:0]	number_of_counters	Always `0b00001000` (8 counters)

PMCR, Performance monitors control register

This register controls the performance monitors.

Usage constraints

Accessible using only Secure accesses, unless you set the *SECR_ACC, Secure access register* on page 3-155 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.6.1 Performance Monitoring Unit registers summary* on page 3-153.

The following figure shows the bit assignments.

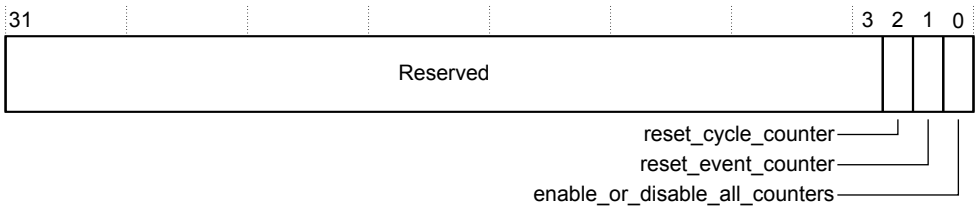

Figure 3-69 PMCR bit assignments

The following table shows the bit descriptions.

Table 3-74 PMCR bit descriptions

Bits	Name	Description
[31:3]	-	Reserved
[2]	reset_cycle_counter	Reset cycle counter, excluding overflow, Read-As-Zero.
[1]	reset_event_counter	Reset event counters, excluding overflows, Read-As-Zero.
[0]	enable_or_disable_all_counters	Enable all counters using the PMCNTENSET register, event and cycle, or disable all counters.

3.7 AXI Slave Network Interface registers

This section describes the NI-700 *AXI Slave Network Interface* (ASNI) registers. It contains a summary of the slave interface registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.7.1 ASNI registers summary

The register summary lists the NI-700 ASNI registers and some key characteristics.

The following table shows the slave interface registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to your SoC implementation documentation. The offset of each register from the base address is fixed.

Table 3-75 ASNI registers summary

Offset	Name	Type	Reset	Width	Description
0x000	ASNI_NODE_TYPE	RO	Configuration dependent	32	*ASNI_NODE_TYPE, Node type register for ASNI registers* on page 3-172
0x004	ASNI_NODE_INFO	RO		13	*ASNI_NODE_INFO, Node information for ASNI register* on page 3-172
0x008	ASNI_SECR_ACC	RW	0x00	2	*ASNI_SECR_ACC, Secure access register* on page 3-174
0x00C	ASNI_PMUSELA	RW	0x0000	32	*ASNI_PMUSELA, Configure ASNI crossbar register* on page 3-175
0x010	ASNI_PMUSELB	RW	0x0000	32	*ASNI_PMUSELB, Configure ASNI crossbar register* on page 3-176
0x014	ASNI_INTERFACEID_0-3	RO	Configuration dependent	32	*ASNI_INTERFACEID, Configure ASNI interface IDs 0-3* on page 3-177
0x018	ASNI_INTERFACEID_4-7	RAZ		32	*ASNI_INTERFACEID, Configure ASNI interface IDs 4-7* on page 3-177
0x01C	ASNI_INTERFACEID_8-11	RAZ		32	*ASNI_INTERFACEID, Configure ASNI interface IDs 8-11* on page 3-178
0x020	ASNI_INTERFACEID_12-15	RAZ		32	*ASNI_INTERFACEID, Configure ASNI interface IDs 12-15* on page 3-179
0x040	ASNI_NODE_FEAT	RAZ	0x0000	32	*ASNI_NODE_FEAT, Node features register* on page 3-179
0x044	ASNI_BURSPLT	RW/RO	-	10	*ASNI_BURSPLT, Burst split control register* on page 3-180
0x048	ASNI_ADDR_REMAP	RW	0x00	8	*ASNI_ADDR_REMAP, Address remap vector register* on page 3-181
0x080	ASNI_SILDBG	RW/RO	0x00	32	*ASNI_SILDBG, ASNI silicon debug monitor register* on page 3-182
0x084	ASNI_QOSCTL	RW	0x00	7	*ASNI_QOSCTL, QoS control register* on page 3-183

Table 3-75 ASNI registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x088	ASNI_WDATTHRS	RW	0x00	5	*ASNI_WDATTHRS, Write data FIFO threshold register* on page 3-184
0x08C	ASNI_ARQOSOVR	RW	0x00	4	*ASNI_ARQOSOVR, Read channel QoS value override register* on page 3-184
0x090	ASNI_AWQOSOVR	RW	0x00	4	*ASNI_AWQOSOVR, Write channel QoS value override register* on page 3-185
0x094	ASNI_ATQOSOT	RW	0x00	10	*ASNI_ATQOSOT, Maximum atomic Outstanding Transactions register* on page 3-186
0x098	ASNI_ARQOSOT	RW	0x00	10	*ASNI_ARQOSOT, Maximum read Outstanding Transactions register* on page 3-187
0x09C	ASNI_AWQOSOT	RW	0x00	10	*ASNI_AWQOSOT, Maximum write Outstanding Transactions register* on page 3-187
0x0A0	ASNI_AXQOSOT	RW	0x00	10	*ASNI_AXQOSOT, Maximum combined Outstanding Transactions register* on page 3-188
0x0A4	ASNI_QOSRDPK	RW	0x00	8	*ASNI_QOSRDPK, Read TSPEC bandwidth regulator peak rate register* on page 3-189
0x0A8	ASNI_QOSRDBUR	RW	0x00	8	*ASNI_QOSRDBUR, Read TSPEC bandwidth regulator burstiness allowance register* on page 3-189
0x0AC	ASNI_QOSRDAVG	RW	0x00	8	*ASNI_QOSRDAVG, Read TSPEC bandwidth regulator average rate register* on page 3-190
0x0B0	ASNI_QOSWRPK	RW	0x00	8	*ASNI_QOSWRPK, Write TSPEC bandwidth regulator peak rate register* on page 3-191
0x0B4	ASNI_QOSWRBUR	RW	0x00	8	*ASNI_QOSWRBUR, Write TSPEC bandwidth regulator burstiness allowance register* on page 3-191
0x0B8	ASNI_QOSWRAVG	RW	0x00	8	*ASNI_QOSWRAVG, Write TSPEC bandwidth regulator average rate register* on page 3-192
0x0BC	ASNI_QOSCOMPK	RW	0x00	8	*ASNI_QOSCOMPK, Combined TSPEC bandwidth regulator peak rate register* on page 3-193
0x0C0	ASNI_QOSCOMBUR	RW	0x0000	14	*ASNI_QOSCOMBUR, Combined TSPEC bandwidth regulator burstiness allowance register* on page 3-193
0x0C4	ASNI_QOSCOMAVG	RW	0x00	8	*ASNI_QOSCOMAVG, Combined TSPEC bandwidth regulator average rate register* on page 3-194
0x0C8	ASNI_QOSRDBQV	RW	0x00	27	*ASNI_QOSRDBQV, Read BQV bandwidth regulator target bandwidth register* on page 3-195
0x0CC	ASNI_QOSWRBQV	RW	0x00	27	*ASNI_QOSWRBQV, Write BQV bandwidth regulator target bandwidth register* on page 3-196
0x0D0	ASNI_QOSCOMBQV	RW	0x00	27	*ASNI_QOSCOMBQV, combined BQV bandwidth regulator target bandwidth register* on page 3-196

Table 3-75 ASNI registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x0E0	ASNI_AR_MPAM_OVERRIDE	RW	0x0	12	*ASNI_AR_MPAM_OVERRIDE, Read channel MPAM override register* on page 3-197
0x0E4	ASNI_AW_MPAM_OVERRIDE	RW	0x0	12	*ASNI_AW_MPAM_OVERRIDE, Write channel MPAM override register* on page 3-198

3.7.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

ASNI_NODE_TYPE, Node type register for ASNI registers

This register identifies the node type as a node for ASNI registers.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

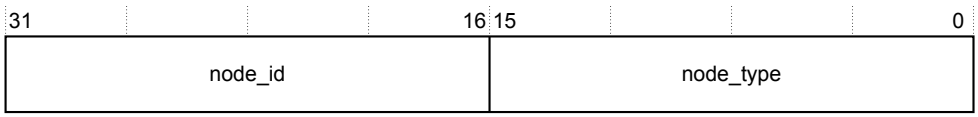

Figure 3-70 ASNI_NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-76 ASNI_NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The ASNI ID that is assigned during network construction.
[15:0]	node_type	The value of this field is 0x04, and it identifies the associated node type as a node for NI-700 ASNI registers.

ASNI_NODE_INFO, Node information for ASNI register

This register provides node information for ASNI, such as data width.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

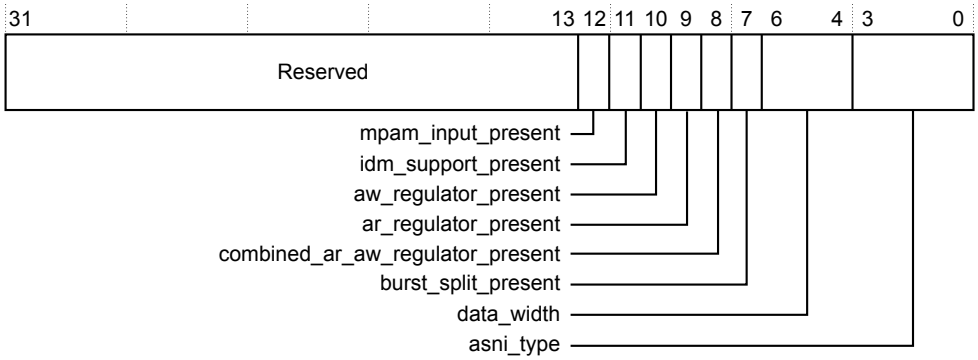

Figure 3-71 ASNI_NODE_INFO bit assignments

The following table shows the bit descriptions.

Table 3-77 ASNI_NODE_INFO bit descriptions

Bits	Name	Description
[31:13]	-	Reserved
[12]	mpam_input_present	MPAM input signals present: **0** MPAM input signal not present. **Note** If MPAM input signals are not present, then the MPAM value is driven from the MPAM override register, regardless of the MPAM override enable bit. **1** MPAM input signal is present.
[11]	idm_support_present	IDM support present: **0** IDM support logic not present. **1** IDM support logic is present.
[10]	aw_regulator_present	AW regulator is present: **0** AW regulator logic not present. **1** AW regulator logic is present.
[9]	ar_regulator_present	AR regulator is present: **0** AR regulator logic not present. **1** AR regulator logic is present.
[8]	combined_ar_aw_regulator_present	Combined AR and AW regulator present: **0** Combined AR and AW regulator logic is not present. **1** Combined AR and AW regulator logic is present.
[7]	burst_split_present	Burst split present: **0** Burst split logic is not present. **1** Burst split logic is present.

Table 3-77 ASNI_NODE_INFO bit descriptions (continued)

Bits	Name	Description
[6:4]	data_width	Data width, AxSIZE encoded: `0b000` Reserved `0b001` Reserved `0b010` 4 bytes `0b011` 8 bytes `0b100` 16 bytes `0b101` 32 bytes `0b110` 64 bytes `0b111` 128 bytes **Note** Reset value: 0x<N> where N is equal to the encoded data width of the interface.
[3:0]	asni_type	ASNI type: `0b0000` Reserved `0b0001` Reserved `0b0010` AXI `0b0011` ACE-Lite `0b0100` AXI-G `0b0101` AXI-H `0110-1111` Reserved **Note** Reset value: **`0b0010`** AXI **`0b0011`** ACE-Lite

ASNI_SECR_ACC, Secure access register

This register controls Secure access.

Usage constraints
Accessible using only Secure accesses.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

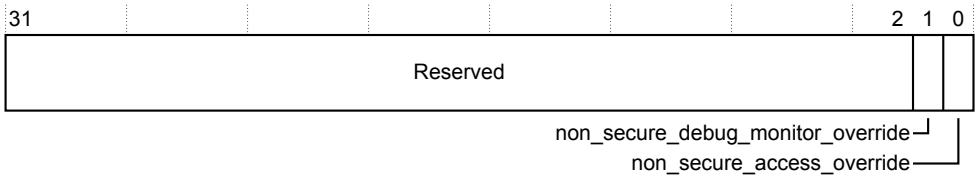

Figure 3-72 ASNI_SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-78 ASNI_SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved.
[1]	non_secure_debug_monitor_override	Non-secure debug monitor override: **0** Disable. Non-secure access to the NI-700 PMU and interface registers. **1** Enable. Non-secure access to the NI-700 PMU and interface registers.
[0]	non_secure_access_override	Non-secure access override: **0** Disable. Non-secure access to the Secure NI-700 registers in this register region. **1** Enable. Non-secure access to the Secure NI-700 registers in this register region.

ASNI_PMUSELA, Configure ASNI crossbar register

This register is used to select the event values in the ASNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

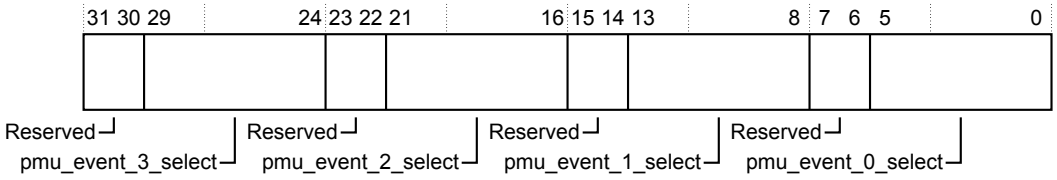

Figure 3-73 ASNI_PMUSELA bit assignments

The following table shows the bit descriptions.

Table 3-79 ASNI_PMUSELA bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_3_select	PMU event 3 select
[23:22]	-	Reserved
[21:16]	pmu_event_2_select	PMU event 2 select
[15:14]	-	Reserved
[13:8]	pmu_event_1_select	PMU event 1 select
[7:6]	-	Reserved
[5:0]	pmu_event_0_select	PMU event 0 select

ASNI_PMUSELB, Configure ASNI crossbar register

This register is used to select the event values in the ASNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

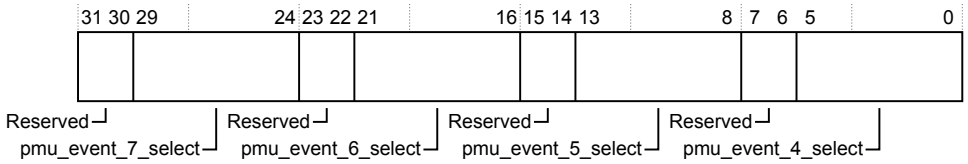

Figure 3-74 ASNI_PMUSELB bit assignments

The following table shows the bit descriptions.

Table 3-80 ASNI_PMUSELB bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_7_select	PMU event 7 select
[23:22]	-	Reserved
[21:16]	pmu_event_6_select	PMU event 6 select
[15:14]	-	Reserved
[13:8]	pmu_event_5_select	PMU event 5 select
[7:6]	-	Reserved
[5:0]	pmu_event_4_select	PMU event 4 select

ASNI_INTERFACEID, Configure ASNI interface IDs 0-3

To configure ASNI interface IDs 0-3, use offset `0x014` in the ASNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

Note

The ASNI node contains a single AXI or ACE-Lite interface connected to it. Therefore, ASNI interface ID 0 is the only meaningful interface ID value which is read from interface_0, bits [7-0] field of the ASNI_INTERFACEID_0-3 register. The remaining fields, bits [31-8], in the ASNI_INTERFACEID_0-3 register are all reserved. Similarly, the other ASNI interface ID registers 4-7, 8-11 and 12-15 are all Reserved.

The following figure shows the bit assignments.

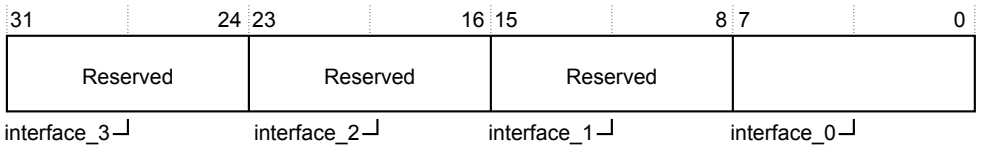

Figure 3-75 ASNI_INTERFACEID bit assignments, ASNI interface IDs 0-3

The following table shows the bit descriptions.

Table 3-81 ASNI_INTERFACEID descriptions, ASNI Interface IDs 0-3

Bits	Name	Description
[31:24]	interface_3	Reserved
[23:16]	interface_2	Reserved
[15:8]	interface_1	Reserved
[7:0]	interface_0	ASNI Interface ID 0

ASNI_INTERFACEID, Configure ASNI interface IDs 4-7

To configure ASNI interface IDs 4-7, use offset `0x018` in the ASNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

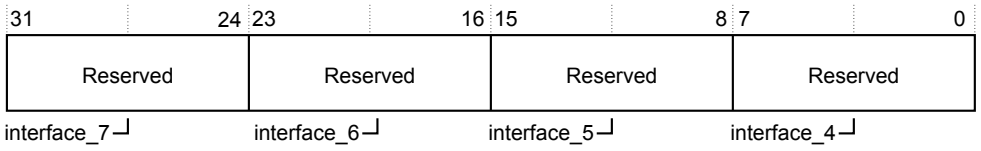

Figure 3-76 ASNI_INTERFACEID bit assignments, ASNI interface IDs 4-7

The following table shows the bit descriptions.

Table 3-82 ASNI_INTERFACEID descriptions, ASNI Interface IDs 4-7

Bits	Name	Description
[31:24]	interface_7	Reserved
[23:16]	interface_6	Reserved
[15:8]	interface_5	Reserved
[7:0]	interface_4	Reserved

ASNI_INTERFACEID, Configure ASNI interface IDs 8-11

To configure ASNI interface IDs 8-11, use offset `0x01C` in the ASNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

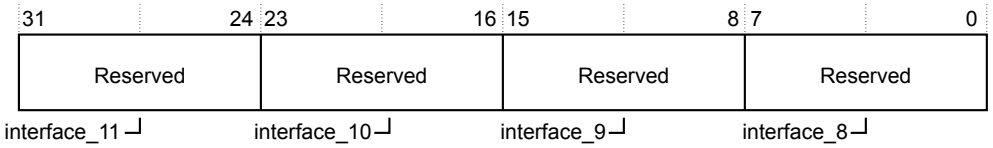

Figure 3-77 ASNI_INTERFACEID bit assignments, ASNI interface IDs 8-11

The following table shows the bit descriptions.

Table 3-83 ASNI_INTERFACEID descriptions, ASNI Interface IDs 8-11

Bits	Name	Description
[31:24]	interface_11	Reserved
[23:16]	interface_10	Reserved
[15:8]	interface_9	Reserved
[7:0]	interface_8	Reserved

ASNI_INTERFACEID, Configure ASNI interface IDs 12-15

To configure ASNI interface IDs 12-15, use offset `0x020` in the ASNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

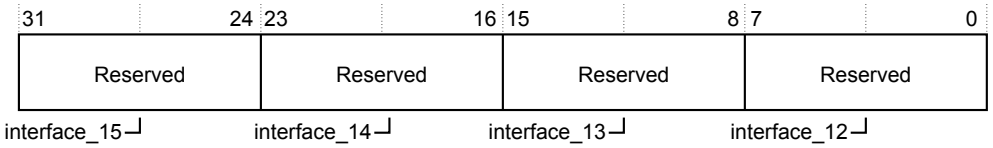

Figure 3-78 ASNI_INTERFACEID bit assignments, ASNI interface IDs 12-15

The following table shows the bit descriptions.

Table 3-84 ASNI_INTERFACEID descriptions, ASNI Interface IDs 12-15

Bits	Name	Description
[31:24]	interface_15	Reserved
[23:16]	interface_14	Reserved
[15:8]	interface_13	Reserved
[7:0]	interface_12	Reserved

ASNI_NODE_FEAT, Node features register

This register configures the node features.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

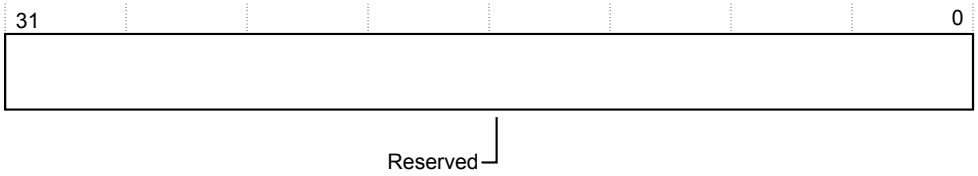

Figure 3-79 ASNI_NODE_FEAT bit assignments

The following table shows the bit descriptions.

Table 3-85 ASNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:0]	-	Reserved

ASNI_BURSPLT, Burst split control register

This register shows the Burst split value to apply and the Burst split value that is applied.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

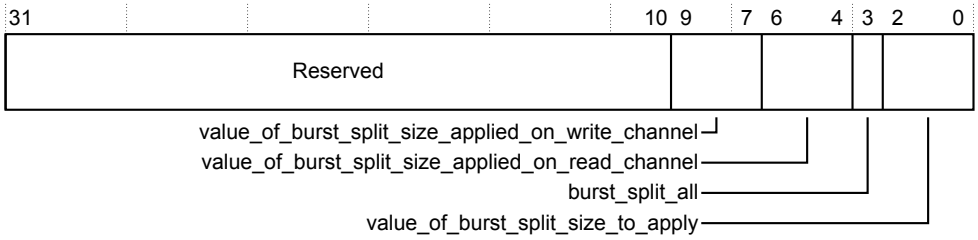

Figure 3-80 BURSPLT bit assignments

The following table shows the bit descriptions.

Table 3-86 BURSPLT bit assignments

Bits	Name	Description
[31:10]	-	Reserved
[9:7]	value_of_burst_split_size_applied_on_write_channel	The value of Burst split size that is applied on the write channel. The value is based on the size of the address stripe. **Note** These register values indicate the applied Burst size. The values are the lower of: • The configured minimum address stripe size, entered through the address map. • This register value, [2:0].

Table 3-86 BURSPLT bit assignments (continued)

Bits	Name	Description
[6:4]	value_of_burst_split_size_applied_on_read_channel	The value of Burst split size that is applied on the read channel. The value is based on the size of the address stripe. **Note** These register values indicate the applied Burst size. The values are the lower of: • The configured minimum address stripe size, entered through the address map. • This register value, [2:0].
[3]	burst_split_all	Burst split all. If set, modifiable Bursts to non-striped are also split.
[2:0]	value_of_burst_split_size_to_apply	The value of Burst split size to apply. The supported encodings are: `0b010` 128 bytes `0b011` 256 bytes `0b100` 512 bytes `0b101` 1024 bytes `0b110` 2048 bytes `0b111` 4096 bytes, no Burst split

Note

1. Modified values are applied only after a current ongoing Burst split sequence is complete. We recommend setting the [3:0] bits while the interface is idle, otherwise it is UNPREDICTABLE when the new Burst split control values take effect.
2. If they cross a split size boundary, transactions to stripe regions are always split.
3. Non-modifiable transactions to non-stripe regions are never split.

ASNI_ADDR_REMAP, Address remap vector register

This register is used to program up to eight remap states that are supported by the address decode in the NI-700.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

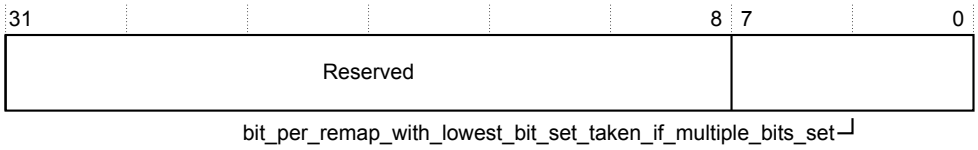

Figure 3-81 ASNI_ADDR_REMAP bit assignments

The following table shows the bit descriptions.

Table 3-87 ASNI_ADDR_REMAP bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	bit_per_remap_with_lowest_bit_set_taken_if_multiple_bits_set	If multiple bits are set, the bit per remap with the lowest bit set is taken.

ASNI_SILDBG, ASNI silicon debug monitor register

This register monitors the status of NI-700 slave interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

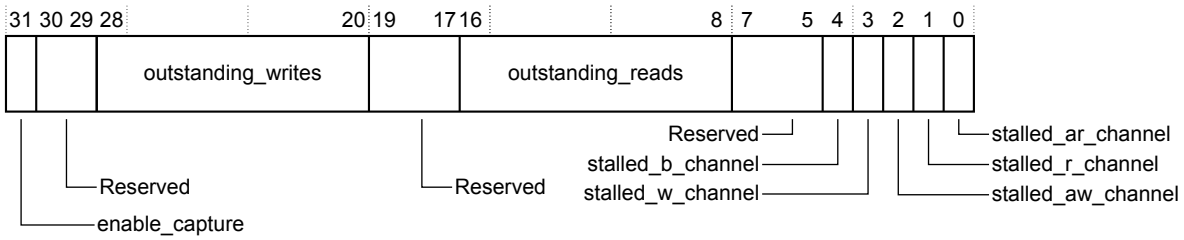

Figure 3-82 ASNI_SILDBG bit assignments

The following table shows the bit descriptions.

Table 3-88 ASNI_SILDBG bit descriptions

Bits	Name	Description
[31]	enable_capture	Enable capture
[30:29]	-	Reserved
[28:20]	outstanding_writes	Indicates that the interface has writes that are outstanding.
[19:17]	-	Reserved

Table 3-88 ASNI_SILDBG bit descriptions (continued)

Bits	Name	Description
[16:8]	outstanding_reads	Indicates that the interface has reads that are outstanding.
[7:5]	-	Reserved
[4]	stalled_b_channel	Indicates that the B channel is stalled.
[3]	stalled_w_channel	Indicates that the W channel is stalled.
[2]	stalled_aw_channel	Indicates that the AW channel is stalled.
[1]	stalled_r_channel	Indicates that the R channel is stalled.
[0]	stalled_ar_channel	Indicates that the AR channel is stalled.

ASNI_QOSCTL, QoS control register

This register controls the QoS settings for NI-700 BQV and TSPEC and enables a QoS value on inbound transactions to be overridden.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

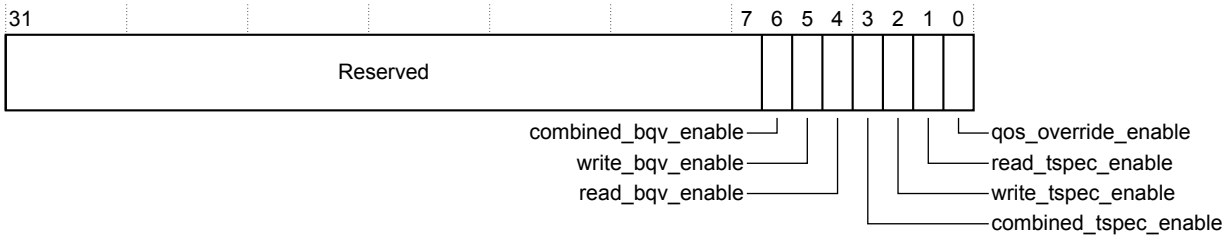

Figure 3-83 ASNI_QOSCTL bit assignments

The following table shows the bit descriptions.

Table 3-89 ASNI_QOSCTL bit descriptions

Bits	Name	Description
[31:7]	-	Reserved
[6]	combined_bqv_enable	Enables combined BQV
[5]	write_bqv_enable	Enables Write BQV
[4]	read_bqv_enable	Enables Read BQV
[3]	combined_tspec_enable	Enables combined TSPEC
[2]	write_tspec_enable	Enables Write TSPEC
[1]	read_tspec_enable	Enables Read TSPEC
[0]	qos_override_enable	When set, this bit enables a QoS value on inbound transactions to be overridden.

ASNI_WDATTHRS, Write data FIFO threshold register

This register specifies the number of write data beats to be queued before the write packet is sent.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

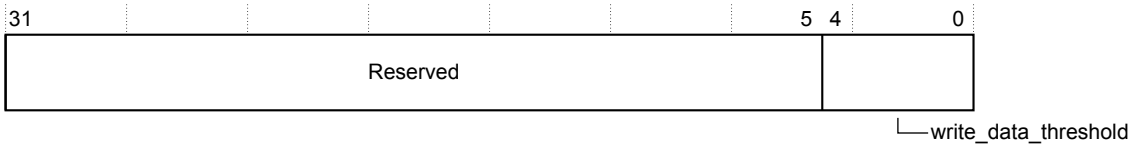

Figure 3-84 ASNI_WDATTHRS bit assignments

The following table shows the bit descriptions.

Table 3-90 ASNI_WDATTHRS bit descriptions

Bits	Name	Description
[31:5]	-	Reserved
[4:0]	write_data_threshold	Write data threshold decimal value Specifies the number of write data beats to be buffered before the write data packet is sent.

ASNI_ARQOSOVR, Read channel QoS value override register

This register stores the override value for the **ARQOS** signal. There is a separate register for each slave interface. This value is applied to transactions when the following configuration scenario applies for the relevant slave interface:

- **QOSOVERRIDE** input signal bit is HIGH or if the **QOS_OVERRIDE_ENABLE** bit of **ASNI_QOSCTL** register is HIGH.
- The **BQV** enable bits of **ASNI_QOSCTL** register have not been set.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
A copy of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

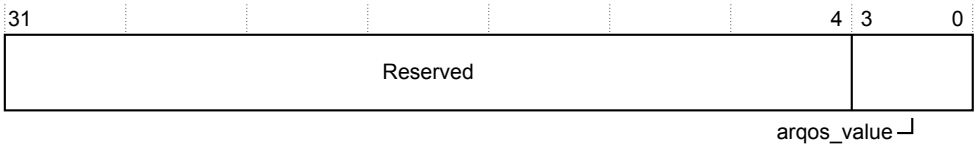

Figure 3-85 ASNI_ARQOSOVR register bit assignments

The following table shows the bit descriptions.

Table 3-91 ASNI_ARQOSOVR bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3:0]	arqos_value	**ARQOS** value override for the slave interface **Note** This value is applied to transactions at this interface if: • The **QOSOVERRIDE** input signal bit is HIGH or if the **QOS_OVERRIDE_ENABLE** bit of **ASNI_QOSCTL** register is HIGH. • The BQV enable bits of **ASNI_QOSCTL** register are not set.

ASNI_AWQOSOVR, Write channel QoS value override register

This register stores the override value for the **AWQOS** signal. There is a separate register for each slave interface. This value is applied to transactions when the following configuration scenario applies for the relevant slave interface:

- The **QOSOVERRIDE** input signal bit is HIGH or if the QOS_OVERRIDE_ENABLE bit of ASNI_QOSCTL register is HIGH.
- The BQV enable bits of ASNI_QOSCTL register are not set for the relevant slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
A copy of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

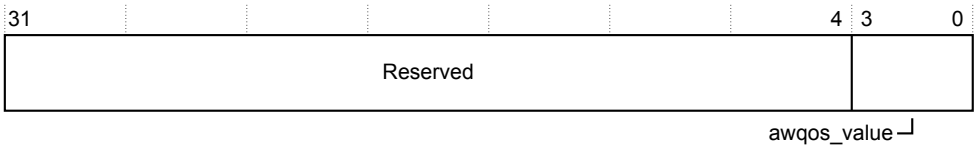

Figure 3-86 ASNI_AWQOSOVR bit assignments

The following table shows the bit descriptions.

Table 3-92 ASNI_AWQOSOVR bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3:0]	awqos_value	The **AWQOS** value override for the slave interface. **Note** This value is applied to transactions at this interface if: • The **QOSOVERRIDE** input signal bit is HIGH or if the QOS_OVERRIDE_ENABLE bit of ASNI_QOSCTL register is HIGH. • The BQV enable bits of ASNI_QOSCTL register are not set.

ASNI_ATQOSOT, Maximum atomic Outstanding Transactions register

This register controls the maximum number of atomic *Outstanding Transactions* (OTs) that are permitted when the OT regulator is enabled for the relevant slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

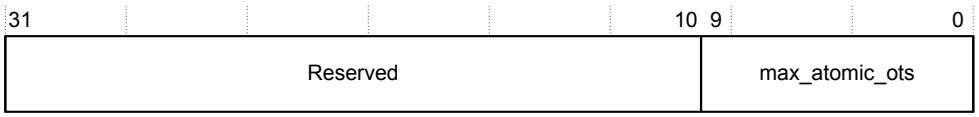

Figure 3-87 ASNI_ATQOSOT bit assignments

The following table shows the bit descriptions.

Table 3-93 ASNI_ATQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9:0]	max_atomic_ots	Specifies the maximum number of outstanding atomic transactions that the slave interface is permitted to issue when OT regulator is enabled on the interface. Atomic transactions are measured as incoming atomic address requests through the interface AW channel. **Note** Atomic transactions require read transaction resources to track generated read responses. Therefore, outstanding atomic transactions are counted in the total outstanding read transactions and must be accounted for when evaluating the traffic profile of your design.

ASNI_ARQOSOT, Maximum read Outstanding Transactions register

This register controls the maximum number of read *Outstanding Transactions* (OTs) that are permitted when the OT regulator is enabled for the relevant slave interface.

Usage constraints

If you set the maximum OT size to a value greater than the value that is configured in the RTL, then the value corresponding to the configured RTL depth is written into this register. The minimum value is 4. Writing values lower than four writes a value of 4 into this register. Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

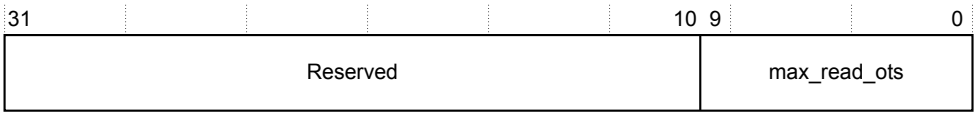

Figure 3-88 ASNI_ARQOSOT bit assignments

The following table shows the bit descriptions.

Table 3-94 ASNI_ARQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9:0]	max_read_ots	The maximum number of outstanding AR address requests when the OT regulator is enabled for the slave interface. **Note** The NI-700 can receive extra transactions at the boundary of the device. Extra transactions can be issued because configurable registering exists between the boundary and the main trackers.

ASNI_AWQOSOT, Maximum write Outstanding Transactions register

This register controls the maximum number of write *Outstanding Transactions* (OTs) that are permitted when the OT regulator is enabled for the relevant slave interface.

Usage constraints

If you set the maximum OT size to a value that is greater than the value that is configured in the RTL, then the value corresponding to the configured RTL depth is written into this register. The minimum value is 4. Writing values lower than four writes a value of 4 into this register. Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

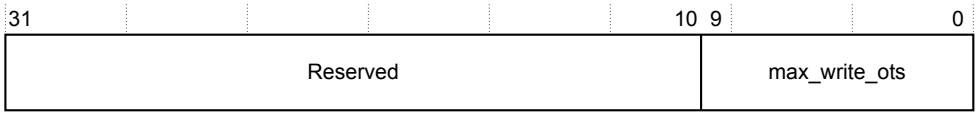

Figure 3-89 ASNI_AWQOSOT bit assignments

The following table shows the bit descriptions.

Table 3-95 ASNI_AWQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9:0]	max_write_ots	The maximum number of write OTs for the slave interface. **Note** Extra transactions can be issued into the NI-700 at the boundary of the device. Extra transactions can be issued because registering exists between the boundary and the main trackers.

ASNI_AXQOSOT, Maximum combined Outstanding Transactions register

This register controls the maximum permitted number of read and write *Outstanding Transactions* (OTs) when the OT regulator is enabled for the relevant slave interface.

Usage constraints

If you configure the maximum OT size to a value greater than the configured RTL value, then the configured RTL value is written into this register. The minimum value is 4. Writing values lower than four writes a value of 4 into this register.
Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

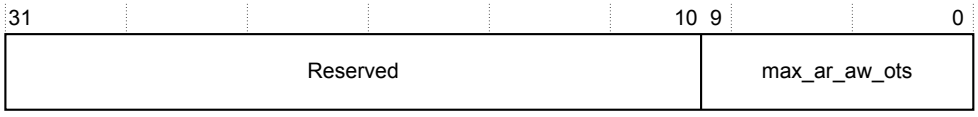

Figure 3-90 ASNI_AXQOSOT bit assignments

The following table shows the bit descriptions.

Table 3-96 ASNI_AXQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9:0]	max_ar_aw_ots	The maximum number of OTs for the slave interface. This value is a combined issuing limit. It represents the maximum number of transactions that the upstream master can issue when the AR and AW channels are considered as one issuing source. **Note** Extra transactions can be issued into the NI-700 at the boundary of the device. Extra transactions can be issued because configurable registering exists between the boundary and the main trackers.

ASNI_QOSRDPK, Read TSPEC bandwidth regulator peak rate register

This register controls the QoS peak rate for the read hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

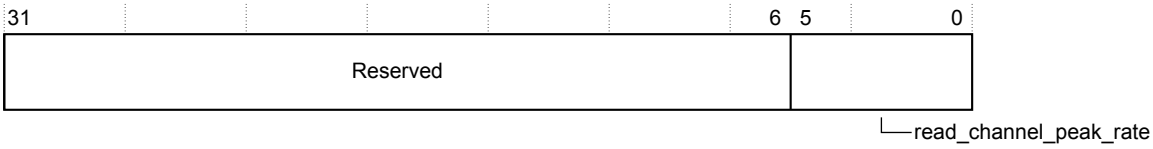

Figure 3-91 ASNI_QOSRDPK bit assignments

The following table shows the bit descriptions.

Table 3-97 ASNI_QOSRDPK bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	read_channel_peak_rate	Read channel peak rate value. The value is a binary fraction of the peak number of read transfers per cycle.

ASNI_QOSRDBUR, Read TSPEC bandwidth regulator burstiness allowance register

This register controls the QoS burstiness for the read hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

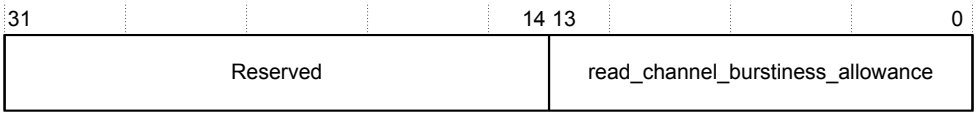

Figure 3-92 ASNI_QOSRDBUR bit assignments

The following table shows the bit descriptions.

Table 3-98 ASNI_QOSRDBUR bit descriptions

Bits	Name	Description
[31:14]	-	Reserved
[13:0]	read_channel_burstiness_allowance	Read channel QoS burstiness allowance value The value is the number of read transfers that is permitted in a transaction.

ASNI_QOSRDAVG, Read TSPEC bandwidth regulator average rate register

This register controls the QoS average rate for the read hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

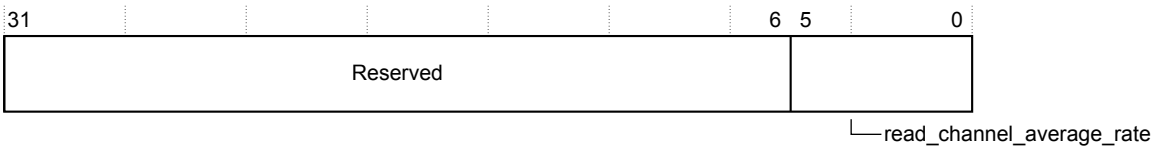

Figure 3-93 ASNI_QOSRDAVG bit assignments

The following table shows the bit descriptions.

Table 3-99 ASNI_QOSRDAVG bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	read_channel_average_rate	Read channel QoS average rate value The value is a binary fraction of the average number of read transfers per cycle.

ASNI_QOSWRPK, Write TSPEC bandwidth regulator peak rate register

This register controls the QoS peak rate for the write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

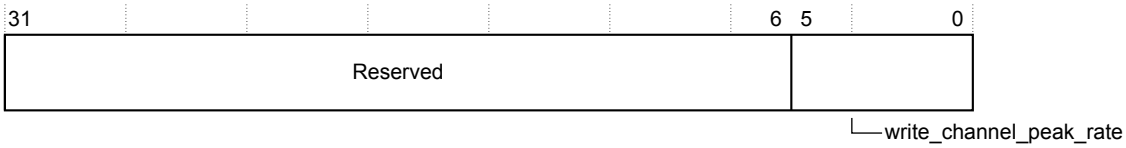

Figure 3-94 ASNI_QOSWRPK bit assignments

The following table shows the bit descriptions.

Table 3-100 ASNI_QOSWRPK bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	write_channel_peak_rate	Write channel peak rate value The value is a binary fraction of the peak number of write transfers per cycle.

ASNI_QOSWRBUR, Write TSPEC bandwidth regulator burstiness allowance register

This register controls the QoS burstiness for the write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

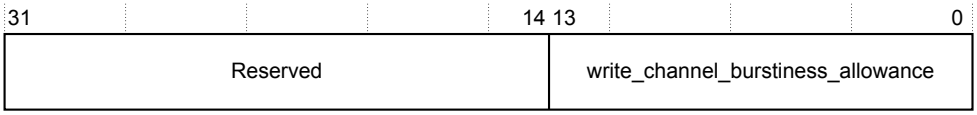

Figure 3-95 ASNI_QOSWRBUR bit assignments

The following table shows the bit descriptions.

Table 3-101 ASNI_QOSWRBUR bit descriptions

Bits	Name	Description
[31:14]	-	Reserved
[13:0]	write_channel_burstiness_allowance	Write channel QoS burstiness allowance value The value is the number of write transfers that are permitted in a transaction.

ASNI_QOSWRAVG, Write TSPEC bandwidth regulator average rate register

This register controls the QoS average rate for the write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

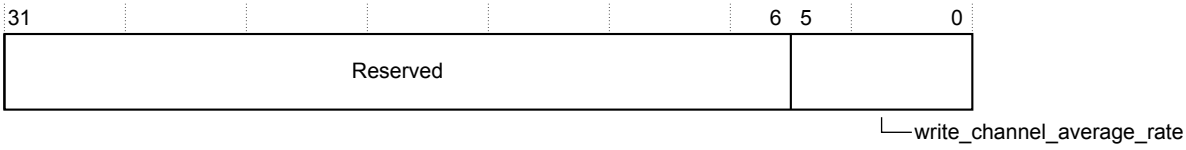

Figure 3-96 ASNI_QOSWRAVG bit assignments

The following table shows the bit descriptions.

Table 3-102 ASNI_QOSWRAVG bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	write_channel_average_rate	Write channel QoS average rate value The value is a binary fraction of the average number of write transfers per cycle.

ASNI_QOSCOMPK, Combined TSPEC bandwidth regulator peak rate register

This register controls the QoS peak rate for both read and write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

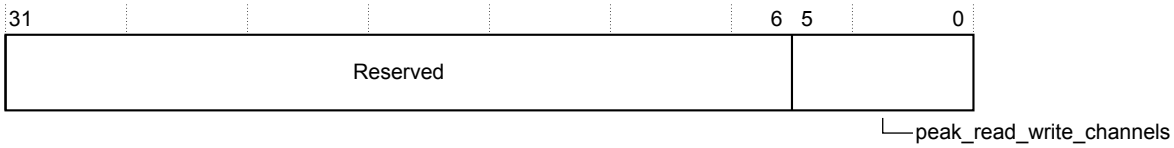

Figure 3-97 ASNI_QOSCOMPK bit assignments

The following table shows the bit descriptions.

Table 3-103 ASNI_QOSCOMPK bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	peak_read_write_channels	The QoS peak rate value of both read and write channels. The value is a binary fraction of the peak number of both read and write transfers per cycle.

ASNI_QOSCOMBUR, Combined TSPEC bandwidth regulator burstiness allowance register

This register controls the QoS burstiness allowance for combined read and write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

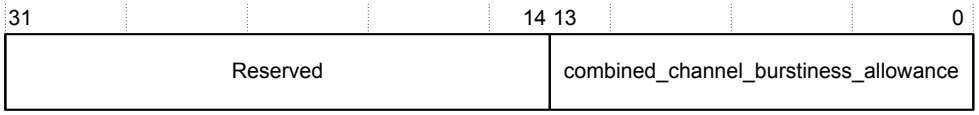

Figure 3-98 ASNI_QOSCOMBUR bit assignments

The following table shows the bit descriptions.

Table 3-104 ASNI_QOSCOMBUR bit descriptions

Bits	Name	Description
[31:14]	-	Reserved
[13:0]	combined_channel_burstiness_allowance	Specifies the combined read and write TSPEC burstiness allowance

ASNI_QOSCOMAVG, Combined TSPEC bandwidth regulator average rate register

This register controls the QoS average rate for both read and write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

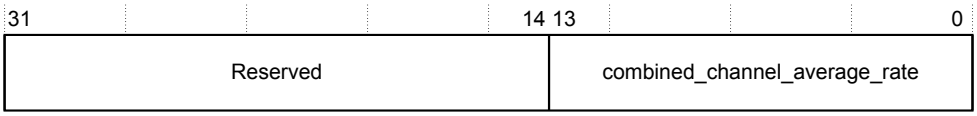

Figure 3-99 ASNI_QOSCOMAVG bit assignments

The following table shows the bit descriptions.

Table 3-105 ASNI_QOSCOMAVG bit descriptions

Bits	Name	Description
[31:14]	-	Reserved
[13:0]	combined_channel_average_rate	The QoS average rate value of both read and write channels. The value is a binary fraction of the average number of both read and write transfers per cycle.

ASNI_QOSRDBQV, Read BQV bandwidth regulator target bandwidth register

This register controls the maximum and minimum QoS values, bandwidth allocation, burstiness, and overspend for read soft bandwidth regulation, BQV, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

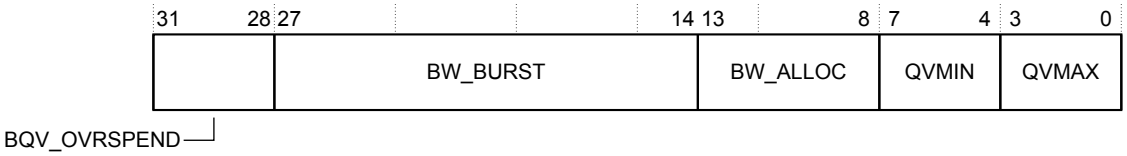

Figure 3-100 ASNI_QOSRDBQV bit assignments

The following table shows the bit descriptions.

Table 3-106 ASNI_QOSRDBQV bit descriptions

Bits	Name	Description
[31:28]	BQV_OVRSPEND	BQV overspend The excess number of full data bus transfers permitted.
[27:14]	BW_BURST	Bandwidth burstiness The excess number of full data bus transfers to permit as burstiness allowance.
[13:8]	BW_ALLOC	Bandwidth allocation The proportion of data bus width for bandwidth allocation.
[7:4]	QVMIN	BQV minimum QoS value The minimum value of **ARQOS**.
[3:0]	QVMAX	BQV maximum QoS value The maximum value of **ARQOS**.

ASNI_QOSWRBQV, Write BQV bandwidth regulator target bandwidth register

This register controls the maximum and minimum QoS values, bandwidth allocation, burstiness, and overspend for write soft bandwidth regulation, BQV, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

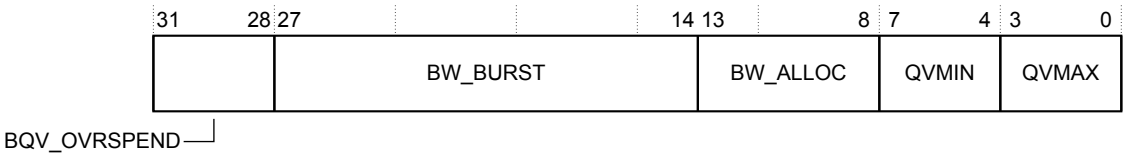

Figure 3-101 ASNI_QOSWRBQV bit assignments

The following table shows the bit descriptions.

Table 3-107 ASNI_QOSWRBQV bit descriptions

Bits	Name	Description
[31:28]	BQV_OVRSPEND	BQV overspend The excess number of full data bus transfers that are permitted.
[27:14]	BW_BURST	Bandwidth burstiness The excess number of full data bus transfers to permit as burstiness allowance.
[13:8]	BW_ALLOC	Bandwidth allocation The proportion of data bus width for bandwidth allocation.
[7:4]	QVMIN	BQV minimum QoS value The minimum value of **AWQOS**.
[3:0]	QVMAX	BQV maximum QoS value The maximum value of **AWQOS**.

ASNI_QOSCOMBQV, combined BQV bandwidth regulator target bandwidth register

This register controls the maximum and minimum QoS values, bandwidth allocation, burstiness, and overspends for both read and write soft bandwidth regulation, BQV, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

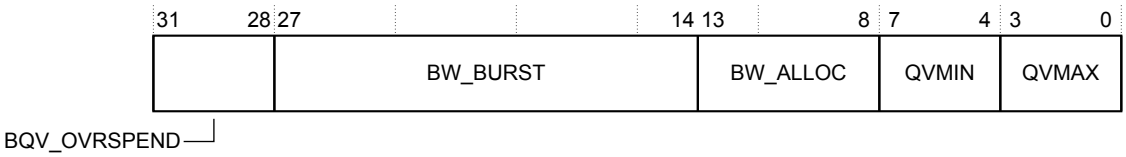

Figure 3-102 ASNI_QOSCOMBQV bit assignments

The following table shows the bit descriptions.

Table 3-108 ASNI_QOSCOMBQV bit descriptions

Bits	Name	Description
[31:28]	bqv_ovrspend	BQV overspend The excess number of full data bus transfers permitted.
[27:14]	bw_burst	Bandwidth burstiness The excess number of full data bus transfers to permit as burstiness allowance.
[13:8]	bw_alloc	Bandwidth allocation The proportion of data bus width for bandwidth allocation.
[7:4]	qvmin	BQV minimum QoS value The minimum value of **AxQOS**.
[3:0]	qvmax	BQV maximum QoS value The maximum value of **AxQOS**.

ASNI_AR_MPAM_OVERRIDE, Read channel MPAM override register

This register controls the ASNI read channel MPAM override register.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

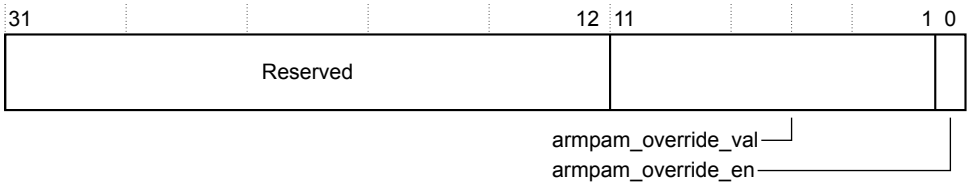

Figure 3-103 ASNI_AR_MPAM_OVERRIDE bit assignments

The following table shows the bit descriptions.

Table 3-109 ASNI_AR_MPAM_OVERRIDE bit descriptions

Bits	Name	Description
[31:12]	-	Reserved
[11:1]	armpam_override_val	ARMPAM override value
[0]	armpam_override_en	When set, ARMPAM value on GT side is driven from the MPAM override value in this register. **Note** If MPAM_SUPPORT = 0 for this specific interface, but GT_MPAM_SUPPORT is enabled, then this register drives the ARMPAM values for this ASNI irrespective of the value of the override bit.

ASNI_AW_MPAM_OVERRIDE, Write channel MPAM override register

This register controls the ASNI write channel MPAM override register.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.7.1 ASNI registers summary* on page 3-170.

The following figure shows the bit assignments.

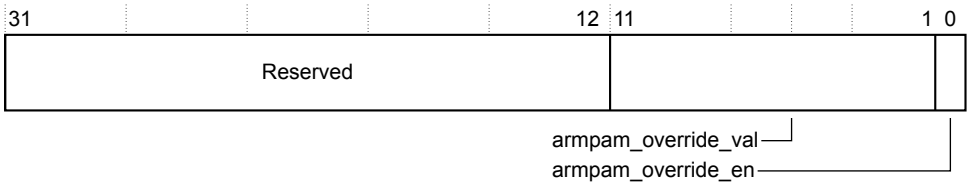

Figure 3-104 ASNI_AW_MPAM_OVERRIDE bit assignments

The following table shows the bit descriptions.

Table 3-110 ASNI_AW_MPAM_OVERRIDE bit descriptions

Bits	Name	Description
[31:12]	-	Reserved
[11:1]	awmpam_override_val	AWMPAM override value
[0:0]	awmpam_override_en	When set, AWMPAM value on GT side is driven from the MPAM override value in this register. **Note** If MPAM_SUPPORT = 0 for this specific interface, but GT_MPAM_SUPPORT is enabled, then this register drives the AWMPAM values for this ASNI irrespective of the value of the override bit.

3.8 AXI Master Network Interface registers

This section describes the NI-700 *AXI Master Network Interface* (AMNI) registers. It contains a summary of the master interface registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.8.1 AMNI registers summary

This register summary lists the NI-700 AMNI registers and some key characteristics.

The following table shows the master interface registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to your SoC implementation documentation. The offset of each register from the base address is fixed.

Table 3-111 AMNI registers summary

Offset	Name	Type	Reset	Width	Description
0x000	AMNI_NODE_TYPE	RO	Configuration dependent	32	*AMNI_NODE_TYPE, Node type register for AMNI registers* on page 3-200
0x004	AMNI_NODE_INFO	RO		9	*AMNI_NODE_INFO, Node information for AMNI register* on page 3-200
0x008	AMNI_SECR_ACC	RW	0x00	2	*AMNI_SECR_ACC, Secure access register* on page 3-202
0x00C	AMNI_PMUSELA	RW	0x0000	32	*AMNI_PMUSELA, Configure AMNI crossbar register* on page 3-203
0x010	AMNI_PMUSELB	RW	0x0000	32	*AMNI_PMUSELB, Configure AMNI crossbar register* on page 3-204
0x014	AMNI_INTERFACEID_0-3	RO	Configuration dependent	32	*AMNI_INTERFACEID, Configure AMNI interface IDs 0-3* on page 3-204
0x018	AMNI_INTERFACEID_4-7	RAZ		32	*AMNI_INTERFACEID, Configure AMNI interface IDs 4-7* on page 3-205
0x01C	AMNI_INTERFACEID_8-11	RAZ		32	*AMNI_INTERFACEID, Configure AMNI interface IDs 8-11* on page 3-206
0x020	AMNI_INTERFACEID_12-15	RAZ		32	*AMNI_INTERFACEID, Configure AMNI interface IDs 12-15* on page 3-206
0x040	AMNI_NODE_FEAT	RAZ	0x0000	32	*AMNI_NODE_FEAT, Node features register* on page 3-207
0x080	AMNI_SILDBG	RW or RO	0x0	32	*AMNI_SILDBG, Silicon debug monitor register* on page 3-207
0x084	AMNI_QOSACC	RW	0x00	2	*AMNI_QOSACC, QoS accept control* on page 3-209
0x088	AMNI_CONFIG_CTL	RW	0b0	1	*AMNI_CONFIG_CTL, Select response* on page 3-209
0x0F0	AMNI_INTERRUPT_STATUS	RW	0b0	3	*AMNI_INTERRUPT_STATUS, Interrupt status register* on page 3-210

Table 3-111 AMNI registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x0F4	AMNI_INTERRUPT_MASK	RW	0b0	3	*AMNI_INTERRUPT_MASK, Interrupt mask register* on page 3-211
0x0F8	AMNI_INTERRUPT_STATUS_NS	RW	0b0	3	*AMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register* on page 3-211
0x0FC	AMNI_INTERRUPT_MASK_NS	RW	0b0	3	*AMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register* on page 3-212

3.8.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

AMNI_NODE_TYPE, Node type register for AMNI registers

This register identifies the node as an NI-700 master interface.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

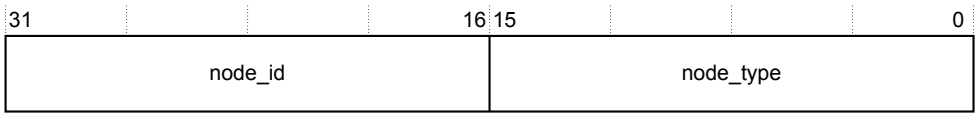

Figure 3-105 AMNI_NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-112 AMNI_NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The AMNI ID assigned during network construction
[15:0]	node_type	The value of this field is 0x0005, and identifies the associated node_type as a master interface for the NI-700 AMNI registers.

AMNI_NODE_INFO, Node information for AMNI register

This register identifies the node as an NI-700 master interface.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

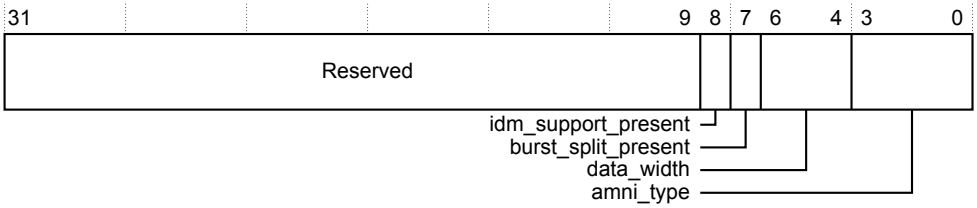

Figure 3-106 AMNI_NODE_INFO bit assignments

The following table shows the bit descriptions.

Table 3-113 AMNI_NODE_INFO bit descriptions

Bits	Name	Description
[31:9]	-	Reserved
[8]	idm_support_present	IDM support present **0** IDM support logic is not present **1** IDM support logic is present
[7]	burst_split_present	Burst split present **0** Burst split logic is not present. **1** Burst split logic is present.

Table 3-113 AMNI_NODE_INFO bit descriptions (continued)

Bits	Name	Description
[6:4]	data_width	Data width, AxSIZE encode `0b000` Reserved `0b001` Reserved `0b010` 4 bytes `0b011` 8 bytes `0b100` 16 bytes `0b101` 32 bytes `0b110` 64 bytes `0b111` 128 bytes
[3:0]	amni_type	AMNI type **`0b0000`** Reserved **`0b0001`** AXI3 **`0b0010`** AXI Issue F **`0b0011`** ACE-Lite **`0b0100`** AXI Issue G **`0b0101`** AXI Issue H **`0110-1111`** Reserved

AMNI_SECR_ACC, Secure access register

This register configures the Non-secure access.

Usage constraints

Read from and write to this register using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

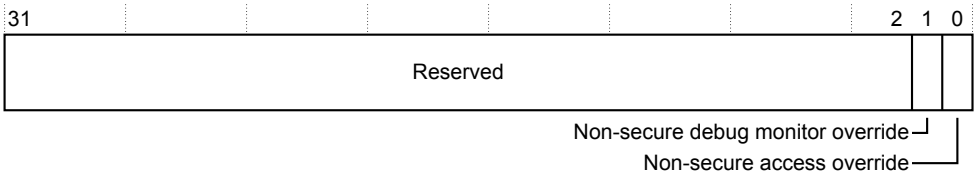

Figure 3-107 SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-114 AMNI_SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	Non-secure debug monitor override	Non-secure debug monitor override
[0]	Non-secure access override	Non-secure access override **0** Disable. Non-secure access to the Secure NI-700 registers in this register region. **1** Enable. Non-secure access to the Secure NI-700 registers in this register region.

AMNI_PMUSELA, Configure AMNI crossbar register

This register is used to select the event values in the AMNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *AMNI_SECR_ACC, Secure access register* on page 3-202 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

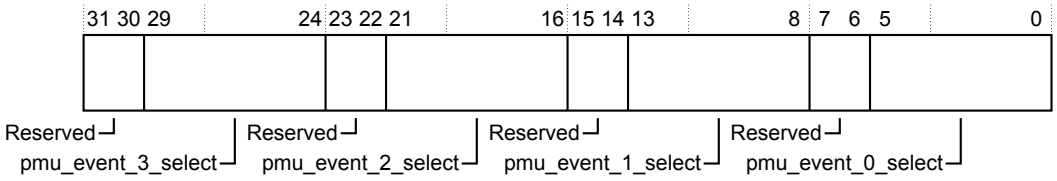

Figure 3-108 AMNI_PMUSELA bit assignments

The following table shows the bit descriptions.

Table 3-115 AMNI_PMUSELA bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_3_select	PMU event 3 select
[23:22]	-	Reserved
[21:16]	pmu_event_2_select	PMU event 2 select
[15:14]	-	Reserved
[13:8]	pmu_event_1_select	PMU event 1 select
[7:6]	-	Reserved
[5:0]	pmu_event_0_select	PMU event 0 select

AMNI_PMUSELB, Configure AMNI crossbar register

This register is used to select the event values in the AMNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *AMNI_SECR_ACC, Secure access register* on page 3-202 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

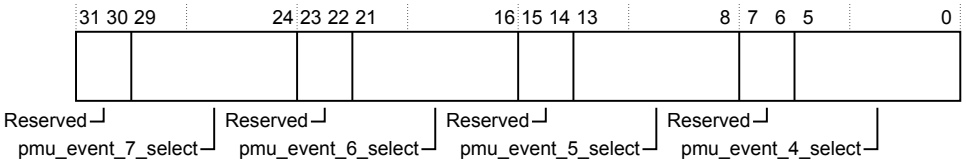

Figure 3-109 AMNI_PMUSELB bit assignments

The following table shows the bit descriptions.

Table 3-116 AMNI_PMUSELB bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_7_select	PMU event 7 select
[23:22]	-	Reserved
[21:16]	pmu_event_6_select	PMU event 6 select
[15:14]	-	Reserved
[13:8]	pmu_event_5_select	PMU event 5 select
[7:6]	-	Reserved
[5:0]	pmu_event_4_select	PMU event 4 select

AMNI_INTERFACEID, Configure AMNI interface IDs 0-3

To configure AMNI interface IDs 0-3, use offset `0x014` in the AMNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

Note

The AMNI node contains a single AXI or ACE-Lite interface connected to it. Therefore, AMNI interface ID 0 is the only meaningful interface ID value which is read from interface_0, bits [7:0], field of the AMNI_INTERFACEID_0-3 register. The remaining fields, bits [31:8], in the

AMNI_INTERFACEID_0-3 register are all Reserved. Similarly, the other AMNI interface ID registers 4-7, 8-11 and 12-15 are all Reserved.

The following figure shows the bit assignments.

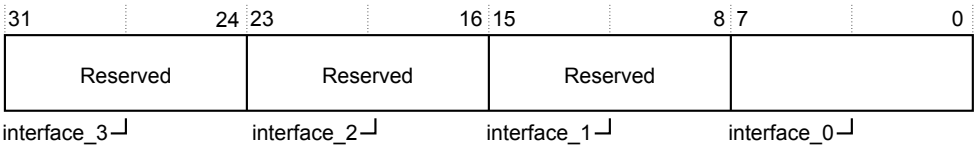

Figure 3-110 AMNI_INTERFACEID bit assignments, AMNI interface IDs 0-3

The following table shows the bit descriptions.

Table 3-117 AMNI_INTERFACEID descriptions, AMNI interface IDs 0-3

Bits	Name	Description
[31:24]	interface_3	Reserved
[23:16]	interface_2	Reserved
[15:8]	interface_1	Reserved
[7:0]	interface_0	AMNI interface ID 0

AMNI_INTERFACEID, Configure AMNI interface IDs 4-7

To configure AMNI interface IDs 4-7, use offset `0x018` in the AMNI_INTERFACEID register.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

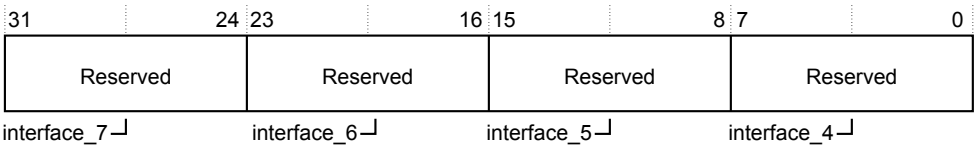

Figure 3-111 AMNI_INTERFACEID bit assignments, AMNI interface IDs 4-7

The following table shows the bit descriptions.

Table 3-118 AMNI_INTERFACEID descriptions, AMNI interface IDs 4-7

Bits	Name	Description
[31:24]	interface_7	Reserved
[23:16]	interface_6	Reserved

Table 3-118 AMNI_INTERFACEID descriptions, AMNI interface IDs 4-7 (continued)

Bits	Name	Description
[15:8]	interface_5	Reserved
[7:0]	interface_4	Reserved

AMNI_INTERFACEID, Configure AMNI interface IDs 8-11

To configure AMNI interface IDs 8-11, use offset `0x01C` in the AMNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

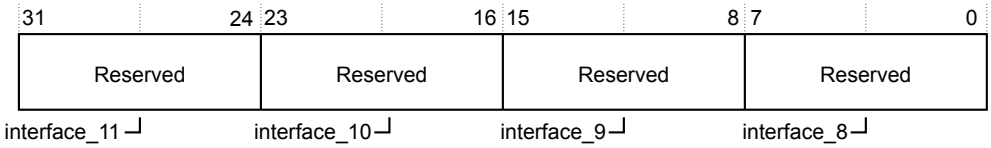

Figure 3-112 AMNI_INTERFACEID bit assignments, AMNI interface IDs 8-11

The following table shows the bit descriptions.

Table 3-119 AMNI_INTERFACEID descriptions, AMNI interface IDs 8-11

Bits	Name	Description
[31:24]	interface_11	Reserved
[23:16]	interface_10	Reserved
[15:8]	interface_9	Reserved
[7:0]	interface_8	Reserved

AMNI_INTERFACEID, Configure AMNI interface IDs 12-15

To configure AMNI interface IDs 12-15, use offset `0x020` in the AMNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

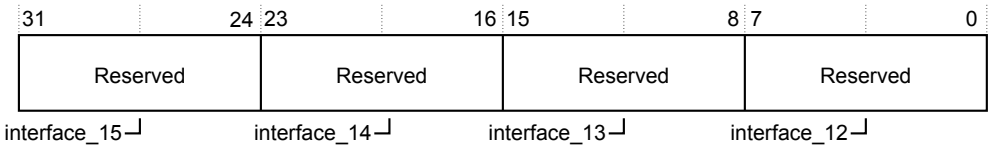

Figure 3-113 AMNI_INTERFACEID bit assignments, AMNI interface IDs 12-15

The following table shows the bit descriptions.

Table 3-120 AMNI_INTERFACEID descriptions, AMNI interface IDs 12-15

Bits	Name	Description
[31:24]	interface_15	Reserved
[23:16]	interface_14	Reserved
[15:8]	interface_13	Reserved
[7:0]	interface_12	Reserved

AMNI_NODE_FEAT, Node features register

This register configures the AMNI node features.

Usage constraints

Accessible using only Secure accesses, unless you set the *AMNI_SECR_ACC, Secure access register* on page 3-202 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

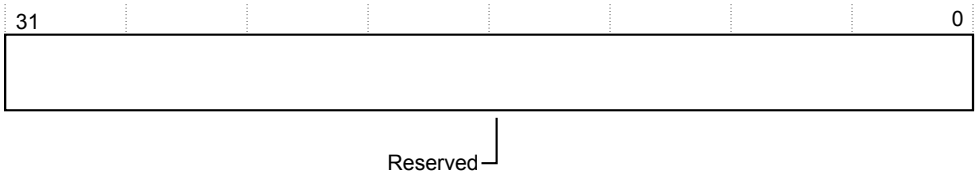

Figure 3-114 AMNI_NODE_FEAT bit assignments

The following table shows the bit descriptions.

Table 3-121 AMNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:0]	-	Reserved

AMNI_SILDBG, Silicon debug monitor register

This register monitors the status of NI-700 master interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the *AMNI_SECR_ACC, Secure access register* on page 3-202 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

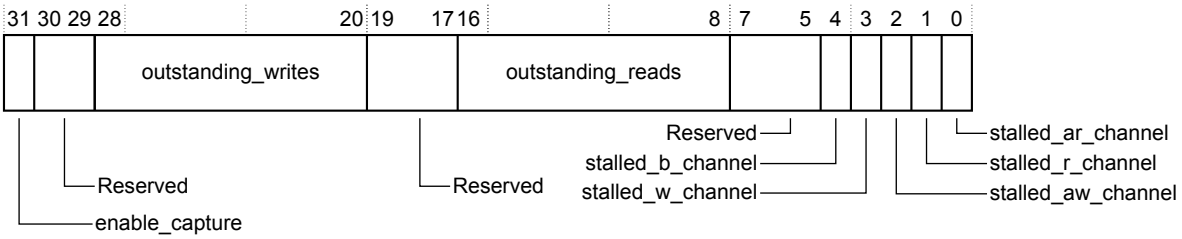

Figure 3-115 AMNI_SILDBG bit assignments

The following table shows the bit descriptions.

Table 3-122 AMNI_SILDBG bit descriptions

Bits	Name	Description
[31]	enable_capture	Enable capture
[30:29]	-	Reserved
[28:20]	outstanding_writes	Number of outstanding write transactions. From request handshake to response.
[19:17]	Reserved	Reserved
[16:8]	outstanding_reads	Number of outstanding read transactions. From request handshake to response.
[7:5]	-	Reserved
[4]	stalled_b_channel	When this bit is set to 1, a transfer is stalled on the B channel, where both: • **BVALID** is HIGH. • **BREADY** is LOW.
[3]	stalled_w_channel	When this bit is set to 1, a transfer is stalled on the W channel, where both: • **WVALID** is HIGH. • **WREADY** is LOW.
[2]	stalled_aw_channel	When this bit is set to 1, a transfer is stalled on the AW channel, where both: • **AWVALID** is HIGH. • **AWREADY** is LOW.
[1]	stalled_r_channel	When this bit is set to 1, a transfer is stalled on the R channel, where both: • **RVALID** is HIGH. • **RREADY** is LOW.
[0]	stalled_ar_channel	When this bit is set to 1, a transfer is stalled on the AR channel, where both: • **ARVALID** is HIGH. • **ARREADY** is LOW.

AMNI_QOSACC, QoS accept control

This register controls QoS acceptance for AMNIs.

Usage constraints

Accessible using only Secure accesses, unless you set the *AMNI_SECR_ACC, Secure access register* on page 3-202 to permit Non-secure accesses. This register can only be modified with prior written permission from Arm.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

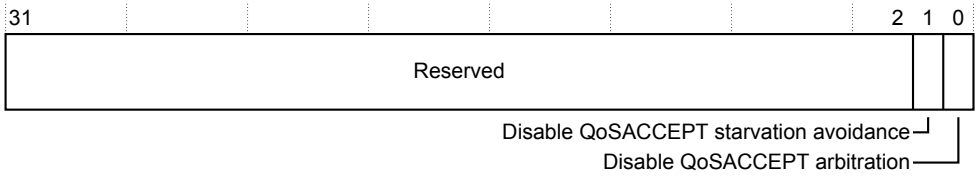

Figure 3-116 AMNI_QOSACC bit assignments

The following table shows the bit descriptions.

Table 3-123 AMNI_QOSACC bit assignments

Bits	Name	Description
[31:2]	-	Reserved
[1]	Disable QoSACCEPT starvation avoidance	Disable QoSACCEPT starvation avoidance
[0]	Disable QoSACCEPT arbitration	Disable QoSACCEPT arbitration

Note

NI-700 does not permit a combined configuration of bit [1] with a value of 1 and bit [0] with a value of 0.

AMNI_CONFIG_CTL, Select response

This register selects between SLVERR or OKAY responses to handle CMOs when downstream slaves do not support it.

Usage constraints

Accessible using Secure transactions only, unless you configure the *AMNI_SECR_ACC, Secure access register* on page 3-202 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199 and *2.7 Master network interface error responses* on page 2-86.

The following figure shows the bit assignments.

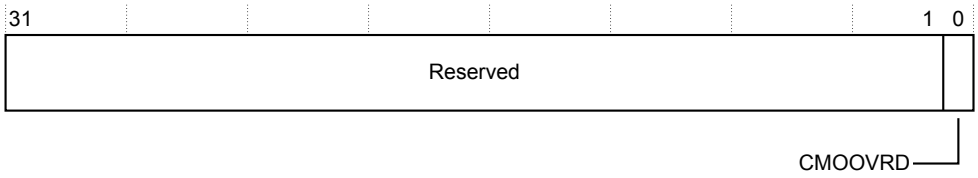

Figure 3-117 AMNI_CONFIG_CTL bit assignments

The following table shows the bit descriptions.

Table 3-124 AMNI_CONFIG_CTL bit assignments

Bits	Name	Description
[31:1]	-	Reserved
[0]	CMOOVRD	Upgrade to SLVERR when set, or use the default response OK. For more information, see *2.7 Master network interface error responses* on page 2-86.

AMNI_INTERRUPT_STATUS, Interrupt status register

This register indicates the interrupt status of Secure transactions.

Usage constraints

Accessible using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

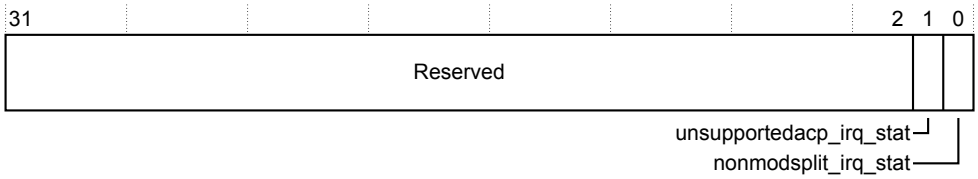

Figure 3-118 AMNI_INTERRUPT_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-125 AMNI_INTERRUPT_STATUS bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	unsupportedacp_irq_stat	Unsupported ACE5-LiteACP request
[0]	nonmodsplit_irq_stat	Non-modifiable Burst split Used for non-modifiable transactions which are split.

Note

A read of 1 for a field indicates that the associated interrupt event has been triggered. A write of 1 to a field in this register clears the associated interrupt event. The interrupt is asserted whenever the appropriate bits in this register are set to 1.

AMNI_INTERRUPT_MASK, Interrupt mask register

This register is the interrupt mask of Secure transactions.

Usage constraints
Accessible using Secure transactions only.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

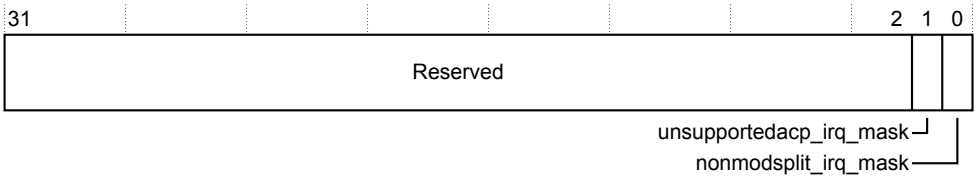

Figure 3-119 AMNI_INTERRUPT_MASK bit assignments

The following table shows the bit descriptions.

Table 3-126 AMNI_INTERRUPT_MASK bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	unsupportedacp_irq_mask	Mask the unsupported ACE5-LiteACP interrupt
[0]	nonmodsplit_irq_mask	Mask the non-modifiable Burst split interrupt

Note

A value of 1 indicates that the interrupt event is masked.

AMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register

This register indicates the interrupt status of Non-secure transactions.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

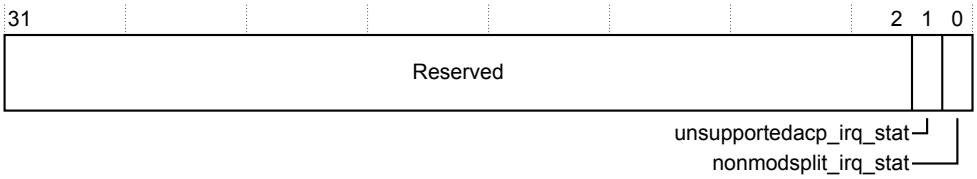

Figure 3-120 AMN_INTERRUPT_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-127 AMNI_INTERRUPT_STATUS_NS bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	unsupportedacp_irq_stat	Unsupported ACE5-LiteACP request
[0]	nonmodsplit_irq_stat	Non-modifiable Burst Split. Used for non-modifiable transactions which are split.

Note

A read of `0x1` for a field indicates that the associated interrupt event has been triggered. A write of `0x1` to a field in this register clears the associated interrupt event. The interrupt is asserted whenever the appropriate bits in this register are set to `0x1`.

AMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

This register is the interrupt mask of Non-secure transactions.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.8.1 AMNI registers summary* on page 3-199.

The following figure shows the bit assignments.

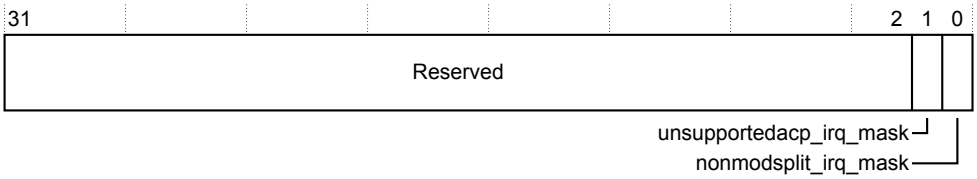

Figure 3-121 AMNI_INTERRUPT_MASK_NS bit assignments

The following table shows the bit descriptions.

Table 3-128 AMNI_INTERRUPT_MASK_NS bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	unsupportedacp_irq_mask	Mask the unsupported ACE5-LiteACP interrupt
[0]	nonmodsplit_irq_mask	Mask the non-modifiable Burst split interrupt

Note

A value of 1 indicates that the interrupt event is masked.

3.9 AHB Slave Network Interface registers

This section describes the NI-700 *AHB Slave Network Interface* (HSNI) registers. It contains a summary of the master interface registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.9.1 HSNI registers summary

This register summary lists the NI-700 HSNI registers and some key characteristics.

The following table shows the slave interface registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to your SoC implementation documentation. The offset of each register from the base address is fixed.

Table 3-129 HSNI registers summary

Offset	Name	Type	Reset	Width	Description
0x000	HSNI_NODE_TYPE	RO	0x0007	32	*HSNI_NODE_TYPE, Node type register for HSNI registers* on page 3-215
0x004	HSNI_NODE_INFO	RO	0x0000	16	*HSNI_NODE_INFO, Node information for HSNI register* on page 3-216
0x008	HSNI_SECR_ACC	RW	0x000	2	*HSNI_SECR_ACC, Secure access register* on page 3-218
0x00C	HSNI_PMUSELA	RW	0x0000	32	*HSNI_PMUSELA, Configure HSNI crossbar register* on page 3-219
0x010	HSNI_PMUSELB	RW	0x0000	32	*HSNI_PMUSELB, Configure HSNI crossbar register* on page 3-219
0x014	HSNI_INTERFACEID_0-3	RO	Configuration dependent	32	*HSNI_INTERFACEID, Configure HSNI interface IDs 0-3* on page 3-220
0x018	HSNI_INTERFACEID_4-7	RAZ		32	*HSNI_INTERFACEID, Configure HSNI interface IDs 4-7* on page 3-221
0x01C	HSNI_INTERFACEID_8-11	RAZ		32	*HSNI_INTERFACEID, Configure HSNI interface IDs 8-11* on page 3-221
0x020	HSNI_INTERFACEID_12-15	RAZ		32	*HSNI_INTERFACEID, Configure HSNI interface IDs 12-15* on page 3-222
0x040	HSNI_NODE_FEAT	RAZ	0x0000	32	*HSNI_NODE_FEAT, Node features register* on page 3-223
0x044	HSNI_CTRL	RW/RO	-	13	*HSNI_CTRL, HSNI control register* on page 3-223
0x048	HSNI_ADDR_REMAP	RW	0x00	8	*HSNI_ADDR_REMAP, Address remap vector register* on page 3-225
0x080	HSNI_SILDBG	RW/RO	0x00	32	*HSNI_SILDBG, HSNI silicon debug monitor register* on page 3-226
0x084	HSNI_QOSCTL	RW	0x00	7	*HSNI_QOSCTL, QoS control register* on page 3-227

Table 3-129 HSNI registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x088	HSNI_WDATTHRS	RW	0x00	4	*HSNI_WDATTHRS, Write data FIFO threshold register* on page 3-228
0x090	HSNI_AWQOSOVR	RW	0x00	4	*HSNI_AWQOSOVR, Write channel QoS value override register* on page 3-228
0x0A0	HSNI_QOSOT	RW	0x00	8	*HSNI_AXQOSOT, Maximum combined Outstanding Transactions register* on page 3-229
0x0BC	HSNI_QOSCOMPK	RW	0x00	6	*HSNI_QOSCOMPK, Combined TSPEC bandwidth regulator peak rate register* on page 3-230
0x0C0	HSNI_QOSCOMBUR	RW	0x0000	14	*HSNI_QOSCOMBUR, Combined TSPEC bandwidth regulator burstiness allowance register* on page 3-230
0x0C4	HSNI_QOSCOMAVG	RW	0x00	6	*HSNI_QOSCOMAVG, Combined TSPEC bandwidth regulator average rate register* on page 3-231
0x0D0	HSNI_QOSCOMBQV	RW	0x00	32	*HSNI_QOSCOMBQV, Combined BQV bandwidth regulator target bandwidth register* on page 3-232
0x0E0	Request MPAM Override	RW	0x0	12	*Request MPAM override register* on page 3-232
0x0F0	HSNI_INTERRUPT_STATUS	RW	0x0	2	*HSNI_INTERRUPT_STATUS, Interrupt status register* on page 3-233
0x0F4	HSNI_INTERRUPT_MASK	RW	0x0	2	*HSNI_INTERRUPT_MASK, Interrupt mask register* on page 3-234
0x0F8	HSNI_INTERRUPT_STATUS_NS	RW	0x0	2	*HSNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register* on page 3-234
0x0FC	HSNI_INTERRUPT_MASK_NS	RW	0x0	2	*HSNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register* on page 3-235

3.9.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

HSNI_NODE_TYPE, Node type register for HSNI registers

This register identifies the node type as a node for HSNI registers.

Usage constraints

Accessible by Secure and Non-secure requests.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

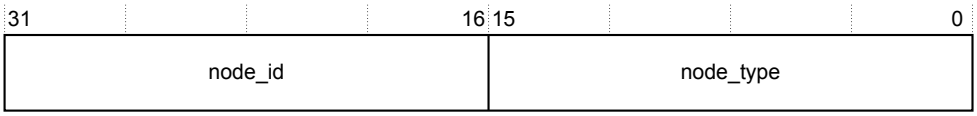

Figure 3-122 HSNI_NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-130 HSNI_NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The HSNI ID that is assigned during network construction.
[15:0]	node_type	The value of this field is `0x07`, and it identifies the associated node type as a node for NI-700 HSNI registers.

HSNI_NODE_INFO, Node information for HSNI register

This register provides node information for HSNI, such as data width.

Usage constraints

Accessible by Secure and Non-secure requests.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

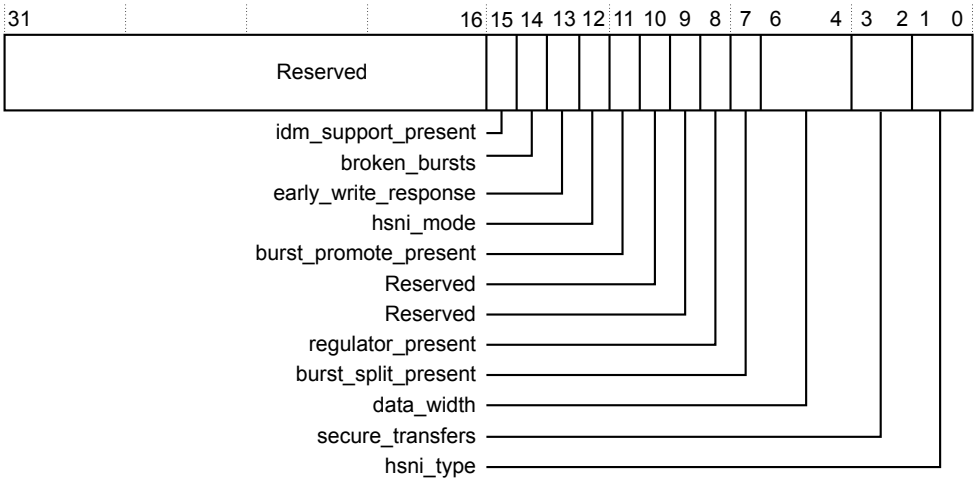

Figure 3-123 HSNI_NODE_INFO bit assignments

The following table shows the bit descriptions.

Table 3-131 HSNI_NODE_INFO bit descriptions

Bits	Name	Description
[31:16]	-	Reserved
[15]	idm_support_present	IDM support present **0** IDM support logic is not present. **1** IDM support logic is present.
[14]	broken_bursts	Broken Bursts **0** There is no logic to handle broken Bursts. **1** There is logic present to handle broken Bursts.
[13]	early_write_response	Early write response **0** HSNI does not generate early write response. **1** HSNI generates early write response.
[12]	hsni_mode	HSNI mode **0** HSNI is not in mirror mode. **1** HSNI is in mirror mode.
[11]	burst_promote_present	Burst promote present **0** Burst promote logic is not present. **1** Burst promote logic is present.
[10]	-	Reserved
[9]	-	Reserved
[8]	Regulator_present	Regulator present **0** Regulator logic is not present. **1** Regulator logic is present.
[7]	burst_split_present	Burst split present **0** Burst split logic is not present. **1** Burst split logic is present.
[6:4]	data_width	Data width, HSIZE encoded `0b000` Reserved `0b001` Reserved `0b010` 4 bytes `0b011` 8 bytes `0b100` 16 bytes `0b101` 32 bytes `0b110` 64 bytes `0b111` 128 bytes

Table 3-131 HSNI_NODE_INFO bit descriptions (continued)

Bits	Name	Description
[3:2]	secure_transfers	`0b00` The software programmable register to set the security attribute for requests from this slave interface. `0b01` The **HSONSEC** pin exists and is used to pass the security attribute. `0b02` All requests which originate from this slave interface are marked Secure. Configure at build time. `0b03` All requests which originate from this slave interface are marked Non-secure. Configure at build time.
[1:0]	hsni_type	HSNI type and property **0** Extended memory type **1** Exclusive transfers

HSNI_SECR_ACC, Secure access register

This register controls Secure access.

Usage constraints
Accessible using Secure transactions only.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

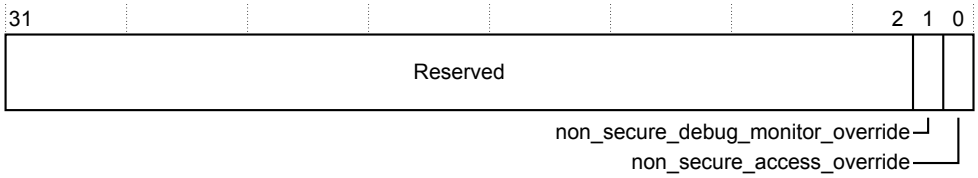

Figure 3-124 HSNI_SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-132 HSNI_SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	non_secure_debug_monitor_override	Non-secure debug monitor override: **0** Disable. Non-secure access to the NI-700 PMU and interface registers. **1** Enable. Non-secure access to the NI-700 PMU and interface registers.
[0]	non_secure_access_override	Non-secure access override: **0** Disable. Non-secure access to the Secure NI-700 registers in this register region. **1** Enable. Non-secure access to the Secure NI-700 registers in this register region.

HSNI_PMUSELA, Configure HSNI crossbar register

This register is used to select the event values in the HSNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

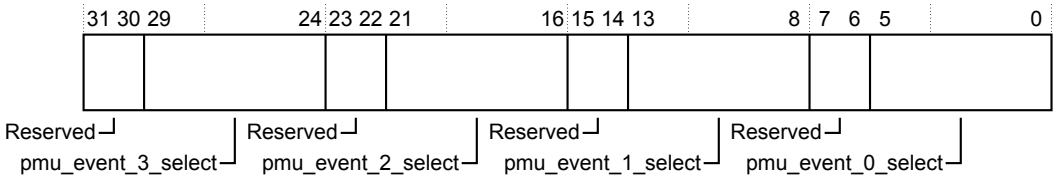

Figure 3-125 HSNI_PMUSELA bit assignments

The following table shows the bit descriptions.

Table 3-133 HSNI_PMUSELA bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_3_select	PMU event 3 select
[23:22]	-	Reserved
[21:16]	pmu_event_2_select	PMU event 2 select
[15:14]	-	Reserved
[13:8]	pmu_event_1_select	PMU event 1 select
[7:6]	-	Reserved
[5:0]	pmu_event_0_select	PMU event 0 select

HSNI_PMUSELB, Configure HSNI crossbar register

This register is used to select the event values in the HSNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

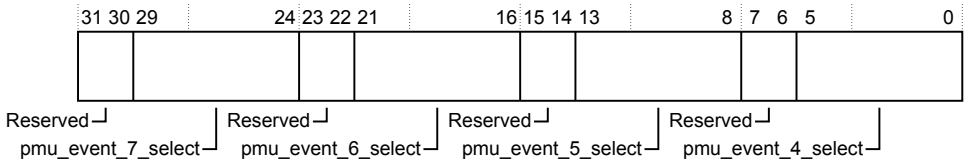

Figure 3-126 HSNI_PMUSELB bit assignments

The following table shows the bit descriptions.

Table 3-134 HSNI_PMUSELB bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_7_select	PMU event 7 select
[23:22]	-	Reserved
[21:16]	pmu_event_6_select	PMU event 6 select
[15:14]	-	Reserved
[13:8]	pmu_event_5_select	PMU event 5 select
[7:6]	-	Reserved
[5:0]	pmu_event_4_select	PMU event 4 select

HSNI_INTERFACEID, Configure HSNI interface IDs 0-3

To configure HSNI interface IDs 0-3, use offset `0x014` in the HSNI_INTERFACEID register.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.9.1 HSNI registers summary* on page 3-214.

Note

The AHB slave network interface (HSNI) node contains a single AHB or ACE-Lite interface connected to it. Therefore, HSNI interface ID 0 is the only meaningful interface ID value which is read from interface_0, bits [7-0], field of the HSNI_INTERFACEID_0-3 register. The remaining fields, bits [31-8], in the HSNI_INTERFACEID_0-3 register are all Reserved. Similarly, the other HSNI interface ID registers 4-7, 8--11 and 12-15 are all Reserved.

The following figure shows the bit assignments.

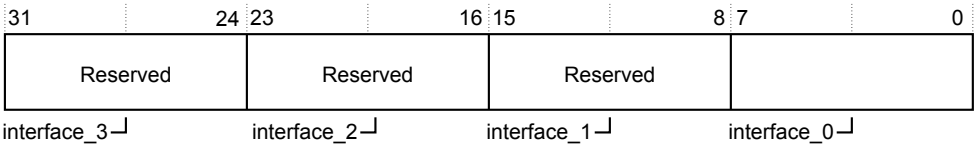

Figure 3-127 HSNI_INTERFACEID bit assignments, HSNI interface IDs 0-3

The following table shows the bit descriptions.

Table 3-135 HSNI_INTERFACEID descriptions, HSNI interface IDs 0-3

Bits	Name	Description
[31:24]	interface_3	Reserved
[23:16]	interface_2	Reserved
[15:8]	interface_1	Reserved
[7:0]	interface_0	HSNI interface ID 0

HSNI_INTERFACEID, Configure HSNI interface IDs 4-7

To configure HSNI interface IDs 4-7, use offset `0x018` in the HSNI_INTERFACEID register.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

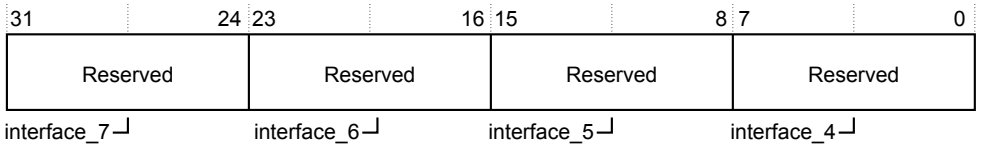

Figure 3-128 HSNI_INTERFACEID bit assignments, HSNI interface IDs 4-7

The following table shows the bit descriptions.

Table 3-136 HSNI_INTERFACEID descriptions, HSNI interface IDs 4-7

Bits	Name	Description
[31:24]	interface_7	Reserved
[23:16]	interface_6	Reserved
[15:8]	interface_5	Reserved
[7:0]	interface_4	Reserved

HSNI_INTERFACEID, Configure HSNI interface IDs 8-11

To configure HSNI interface IDs 8-11, use offset `0x01C` in the HSNI_INTERFACEID register.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

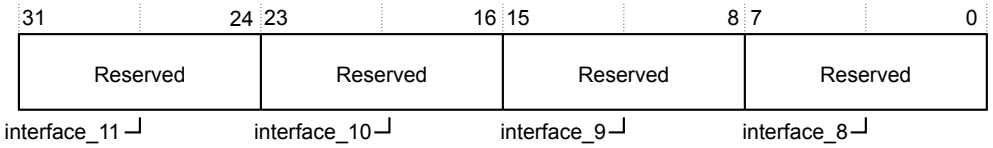

Figure 3-129 HSNI_INTERFACEID bit assignments, HSNI interface IDs 8-11

The following table shows the bit descriptions.

Table 3-137 HSNI_INTERFACEID descriptions, HSNI interface IDs 8-11

Bits	Name	Description
[31:24]	interface_11	Reserved
[23:16]	interface_10	Reserved
[15:8]	interface_9	Reserved
[7:0]	interface_8	Reserved

HSNI_INTERFACEID, Configure HSNI interface IDs 12-15

To configure HSNI interface IDs 12-15, use offset `0x020` in the HSNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

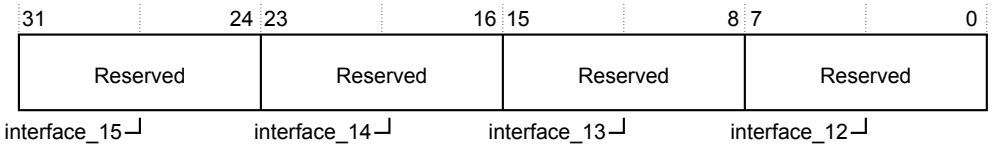

Figure 3-130 HSNI_INTERFACEID bit assignments, HSNI interface IDs 12-15

The following table shows the bit descriptions.

Table 3-138 HSNI_INTERFACEID descriptions, HSNI interface IDs 12-15

Bits	Name	Description
[31:24]	interface_15	Reserved
[23:16]	interface_14	Reserved
[15:8]	interface_13	Reserved
[7:0]	interface_12	Reserved

HSNI_NODE_FEAT, Node features register

This register configures the node features.

Usage constraints

Accessible using only Secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

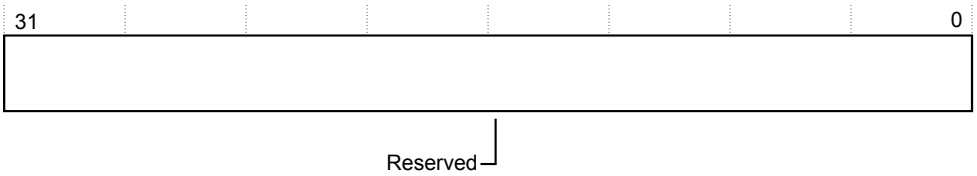

Figure 3-131 HSNI_NODE_FEAT bit assignments

The following table shows the bit descriptions.

Table 3-139 HSNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:0]	-	Reserved

HSNI_CTRL, HSNI control register

This register controls how Bursts are split. If the secure_transfers property is also 0, then it controls mapping of the Non-secure bit. It also provides the applied Burst split value.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

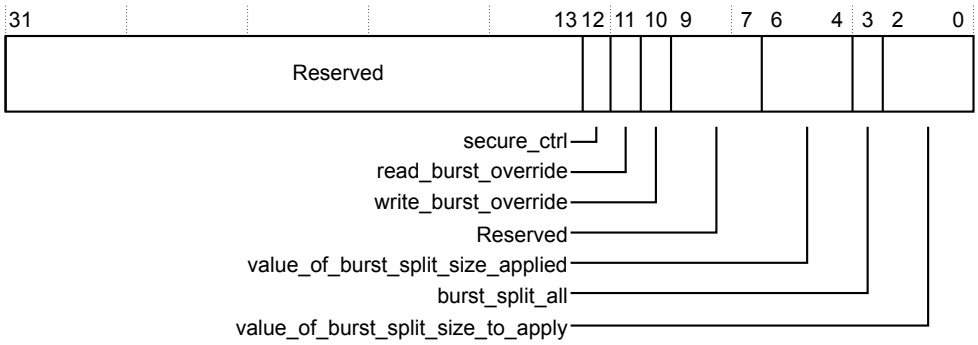

Figure 3-132 HSNI_CTRL bit assignments

The following table shows the bit descriptions.

Table 3-140 HSNI_CTRL bit descriptions

Bits	Name	Description
[31:13]	-	Reserved. Read-As-Zero
[12]	secure_ctrl	If the secure_transfers field for the HSNI_NODE_INFO register = 00 it encodes a software programmable registry. Therefore this field is relevant if the secure_transfers field of HSNI_NODE_INFO = 00. **0** Secure **1** Non-secure **Note** If secure_transfers = 01, it implies that **HNONSEC** pin is supported upstream of HSNI. Therefore this register bit is not relevant. **Note** If secure_transfers = 00, the **HNONSEC** pin is unavailable. Therefore this register bit determines the security attribute of all requests from the upstream slave. **Note** If secure_transfers = 02 or secure_transfers = 03, the **HNONSEC** pin is unavailable. However the security attribute of the HSNI is always Secure or Non-secure and is fixed at build time. This register bit becomes read-only and the reset value is 1 if secure_transfers = 03 and 0 if secure_transfers = 02.
[11]	read_burst_override	If set, all AHB read Bursts are converted into singles if Burst splitter is enabled, that is, parameter BURST_CONVERT [0] = 1.
[10]	write_burst_override	If set, all AHB write Bursts are converted into singles if Burst splitter is enabled, that is, parameter BURST_CONVERT [0] = 1.
[9:7]	-	Reserved. Read-As-Zero.

Table 3-140 HSNI_CTRL bit descriptions (continued)

Bits	Name	Description
[6:4]	value_of_burst_split_size_applied	The value of Burst split size that is applied. **Note** These register values indicate the applied Burst size. The values are the lower of: • The configured minimum address stripe size, entered through the address map. • This register value, [2:0].
[3]	burst_split_all	Burst split all. If set, modifiable Bursts to non-striped are also split.
[2:0]	value_of_burst_split_size_to_apply	The Burst split size value to apply.

Note

1. Register values [6:4] indicate the applied Burst split size. These values are the lower of:
   - Configured min address stripe size, entered through address map
   - Register value [2:0]

   If they cross a split size boundary, transactions to stripe regions are always split.
2. Non-modifiable transactions to non-stripe regions are never split.
3. Modified values are applied only after a current ongoing Burst split sequence is complete. We recommend configuring the [11], [10], [3:0] bits while the interface is idle, otherwise it is UNPREDICTABLE when the new Burst split control values take effect.

HSNI_ADDR_REMAP, Address remap vector register

This register is used to program up to eight remap states supported by the address decode in the NI-700.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

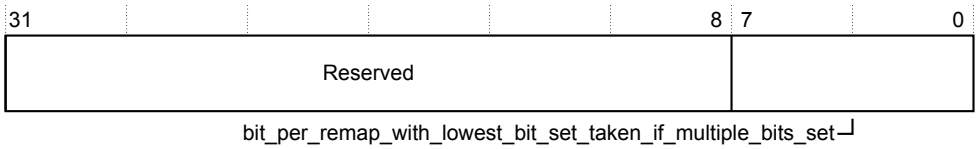

Figure 3-133 HSNI_ADDR_REMAP bit assignments

The following table shows the bit descriptions.

Table 3-141 HSNI_ADDR_REMAP bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	bit_per_remap_with_lowest_bit_set_taken_if_multiple_bits_set	If multiple bits are set, the bit per remap with the lowest bit set is taken.

HSNI_SILDBG, HSNI silicon debug monitor register

This register monitors the status of NI-700 slave interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

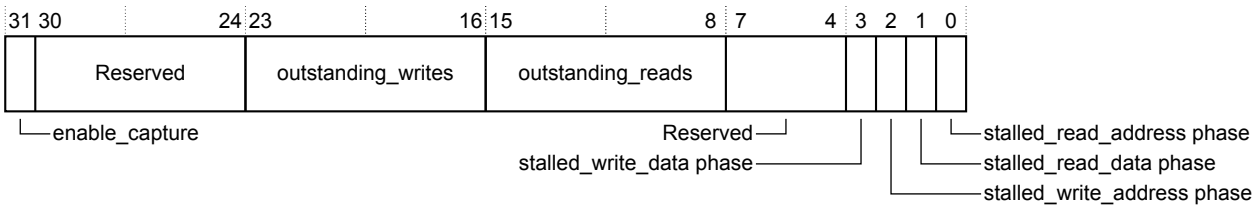

Figure 3-134 HSNI_SILDBG bit assignments

The following table shows the bit descriptions.

Table 3-142 HSNI_SILDBG bit descriptions

Bits	Name	Description
[31]	enable_capture	Enable capture
[30:24]	-	Reserved
[23:16]	outstanding_writes	Indicates that the interface has outstanding writes
[15:8]	outstanding_reads	Indicates that the interface has outstanding read requests. Maximum value is 1.
[7:4]	-	Reserved
[3]	stalled_write_data_phase	Prior write address phase, **HREADY** LOW
[2]	stalled_write_address_phase	Not implemented in the HSNI, tied to 0
[1]	stalled_read_data_phase	Prior read address phase, **HREADY** LOW
[0]	stalled_read_address_phase	Not implemented in the HSNI, tied to 0

Note

Arm recommends you enable capture when the interface is in a quiescent state. If capture is enabled in the middle of the address or data phase of an ongoing request, it is possible the stalls are not captured correctly.

HSNI_QOSCTL, QoS control register

This register controls the QoS settings for NI-700 BQV and TSPEC, and enables a QoS value on inbound transactions to be overridden.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

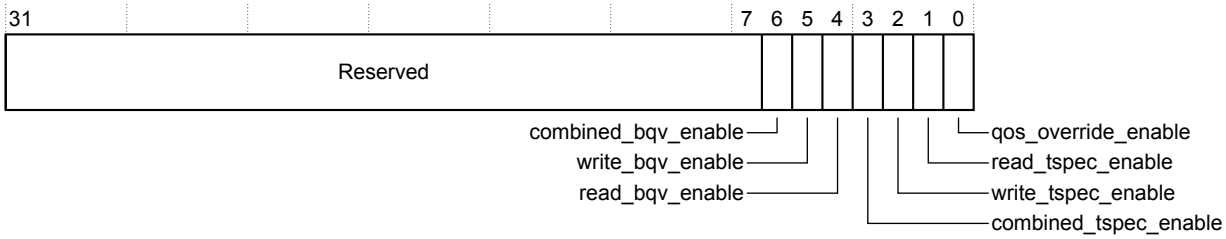

Figure 3-135 HSNI_QOSCTL bit assignments

The following table shows the bit descriptions.

Table 3-143 HSNI_QOSCTL bit descriptions

Bits	Name	Description
[31:7]	-	Reserved
[6]	combined_bqv_enable	Enables BQV For BQV, both of the following conditions (in *soft BW Regulator Target Bandwidth register) are true: 1. BW_ALLOC > 0 2. QVMAX > QVMIN
[5]	write_bqv_enable	Enables write BQV
[4]	read_bqv_enable	Enables read BQV
[3]	combined_tspec_enable	Enables TSPEC For TSPEC, the following conditions are true: 1. *Hard Bandwidth Regulator Average Rate > 0 and: 2. Either: a. Peak regulation is disabled that is, *Hard Bandwidth Regulator Peak Rate = 0 OR: b. Both of the following conditions are true if peak regulation is enabled: a. Hard Bandwidth Regulator Burstiness Allowance > 0 b. Hard Bandwidth Regulator Peak Rate > *Hard Bandwidth Regulator Average Rate
[2]	write_tspec_enable	Reserved

Table 3-143 HSNI_QOSCTL bit descriptions (continued)

Bits	Name	Description
[1]	read_tspec_enable	Reserved
[0]	qos_override_enable	Reserved

HSNI_WDATTHRS, Write data FIFO threshold register

This register specifies the number of write data beats to be queued before the write packet is sent.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

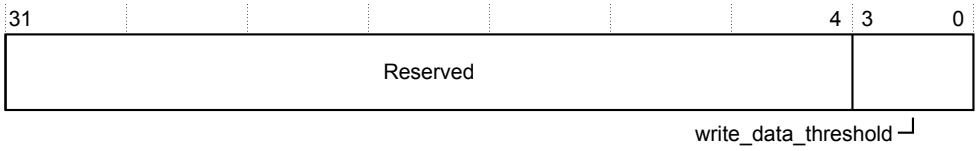

Figure 3-136 HSNI_WDATTHRS bit assignments

The following table shows the bit descriptions.

Table 3-144 HSNI_WDATTHRS bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3:0]	write_data_threshold	Write data threshold decimal value Specifies the number of write data beats to be buffered before the write data packet is sent.

HSNI_AWQOSOVR, Write channel QoS value override register

This register stores the value that is applied to GT transactions if the BQV regulator is not present or enabled.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
A copy of this register exists for each slave interface.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

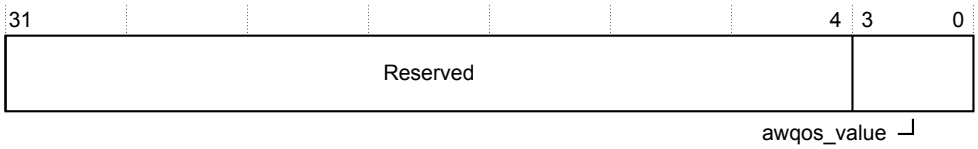

Figure 3-137 HSNI_AWQOSOVR bit assignments

The following table shows the bit descriptions.

Table 3-145 HSNI_AWQOSOVR bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3:0]	awqos_value	**AWQOS** value override for the slave interface.

HSNI_AXQOSOT, Maximum combined Outstanding Transactions register

This register controls the maximum number of read and write *Outstanding Transactions* (OTs) that are permitted when the OT regulator is enabled for the relevant slave interface.

Usage constraints

If you set the maximum OT size to a value greater than the value configured in the RTL, then the value of the configured RTL depth is written to this register. The minimum value is 4. Writing values lower than four, writes a value of 4 into this register.
Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

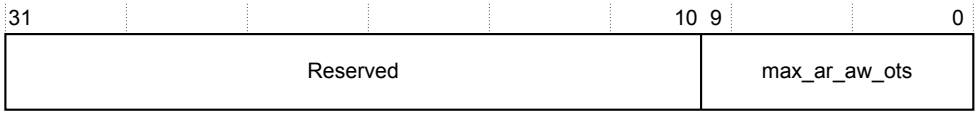

Figure 3-138 HSNI_AXQOSOT bit assignments

The bit descriptions are shown in the following table.

Table 3-146 HSNI_AXQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9:0]	max_ar_aw_ots	The maximum number of OTs for the slave interface. This value is a combined issuing limit. It represents the maximum number of transactions that the upstream master can issue when the AR and AW channels are considered as one issuing source. ——— **Note** ——— Extra transactions can be issued into the NI-700 at the boundary of the device. Extra transactions can be issued because configurable registering exists between the boundary and the main trackers. ———

HSNI_QOSCOMPK, Combined TSPEC bandwidth regulator peak rate register

This register controls the QoS peak rate for both read and write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

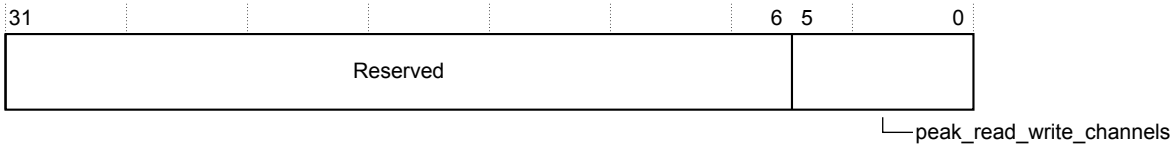

Figure 3-139 HSNI_QOSCOMPK bit assignments

The following table shows the bit descriptions.

Table 3-147 HSNI_QOSCOMPK bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	peak_read_write_channels	The peak rate value of both read and write channels. The value is a binary fraction of the peak number of both read and write transfers per cycle.

HSNI_QOSCOMBUR, Combined TSPEC bandwidth regulator burstiness allowance register

This register controls the QoS burstiness allowance for combined read and write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

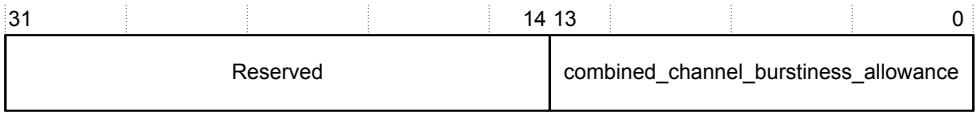

Figure 3-140 HSNI_QOSCOMBUR bit assignments

The following table shows the bit descriptions.

Table 3-148 HSNI_QOSCOMBUR bit descriptions

Bits	Name	Description
[31:14]	-	Reserved
[13:0]	combined_channel_burstiness_allowance	Specifies the combined read and write TSPEC burstiness allowance.

HSNI_QOSCOMAVG, Combined TSPEC bandwidth regulator average rate register

This register controls the QoS average rate for both read and write hard bandwidth regulation, TSPEC, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.
An instance of this register exists for each slave interface.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

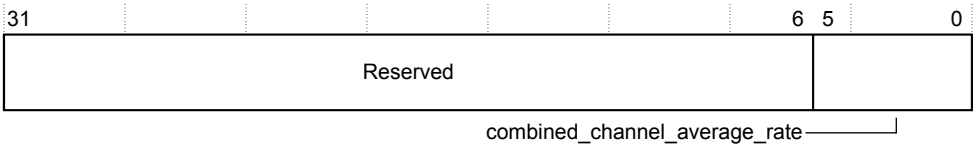

Figure 3-141 HSNI_QOSCOMAVG bit assignments

The following table shows the bit descriptions.

Table 3-149 HSNI_QOSCOMAVG bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	combined_channel_average_rate	The QoS average rate value of both read and write channels. The value is a binary fraction of the average number of both read and write transfers per cycle.

HSNI_QOSCOMBQV, Combined BQV bandwidth regulator target bandwidth register

The register controls the maximum and minimum QoS values, bandwidth allocation, burstiness, and overspend for both read and write soft bandwidth regulation, BQV, of a slave interface.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

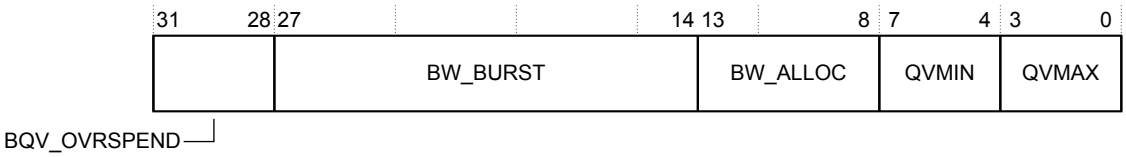

Figure 3-142 HSNI_QOSCOMBQV bit assignments

The following table shows the bit descriptions.

Table 3-150 HSNI_QOSCOMBQV bit descriptions

Bits	Name	Description
[31:28]	bqv_ovrspend	BQV overspend The excess number of full data bus transfers permitted.
[27:14]	bw_burst	Bandwidth burstiness The excess number of full data bus transfers to permit as burstiness allowance.
[13:8]	bw_alloc	Bandwidth allocation The proportion of data bus width for bandwidth allocation.
[7:4]	qvmin	BQV minimum QoS value The minimum value of **ARQOS**.
[3:0]	qvmax	BQV maximum QoS value The maximum value of **ARQOS**.

Request MPAM override register

If GT_MPAM_SUPPORT is enabled, the register drives the MPAM values for a specific HSNI.

Usage constraints

Accessible using only Secure accesses, unless you set the *HSNI_SECR_ACC, Secure access register* on page 3-218 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

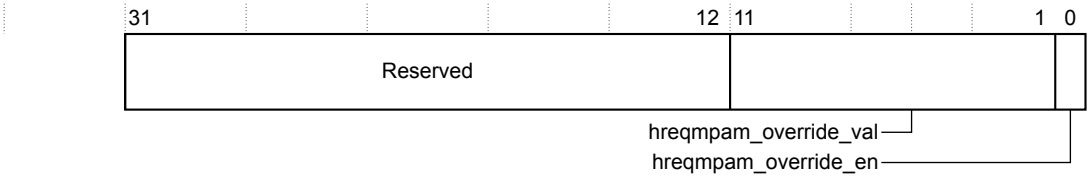

Figure 3-143 Request MPAM override bit assignments

The following table shows the bit descriptions.

Table 3-151 Request MPAM override bit descriptions

Bits	Name	Description
[31:12]	-	Reserved
[11:1]	hreqpam_override_val	ARMPAM override value
[0]	hreqmpam_override_en	For AHB interfaces, the MPAM override value is always used if GT_MPAM_SUPPORT is enabled irrespective of this bit value.

HSNI_INTERRUPT_STATUS, Interrupt status register

This register indicates the interrupt status of Secure transactions.

Usage constraints

Accessible using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

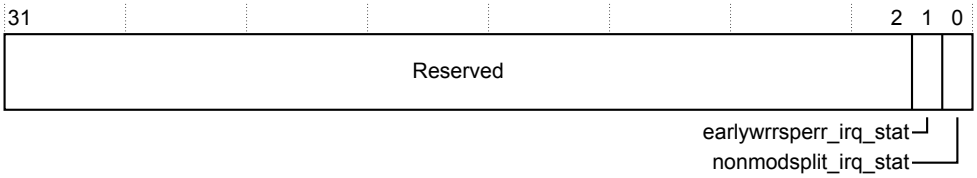

Figure 3-144 HSNI_INTERRUPT_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-152 HSNI_INTERRUPT_STATUS bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	earlywrrsperr_irq_stat	HSNI implements an interrupt mechanism to signal imprecise errors that are detected on actual write responses received for requests for which early write responses were already provided.
[0]	nonmodsplit_irq_stat	If there is a burst split, an interrupt is generated if a non-modifiable transaction is split.

Note

A read of 1 for a field indicates that the associated interrupt event has been triggered. A write of 1 to a field in this register clears the associated interrupt event. The interrupt is asserted whenever the appropriate bits in this register are set to 1.

HSNI_INTERRUPT_MASK, Interrupt mask register

This register is the interrupt mask of Secure transactions.

Usage constraints
Accessible using Secure transactions only.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

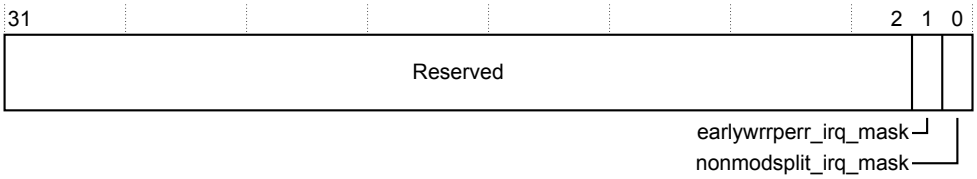

Figure 3-145 HSNI_INTERRUPT_MASK bit assignments

The following table shows the bit descriptions.

Table 3-153 HSNI_INTERRUPT_MASK bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	earlywrrsperr_irq_mask	Mask the early write response with imprecise error interrupt.
[0]	nonmodsplit_irq_mask	Mask the non-modifiable burst split interrupt.

Note

A value of 1 indicates that the interrupt event is masked.

HSNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register

This register indicates the interrupt status of Non-secure transactions.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

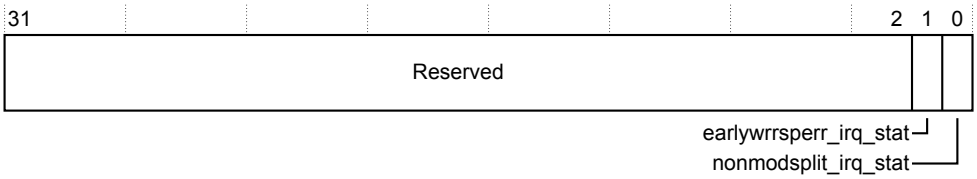

Figure 3-146 HSNI_INTERRUPT_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-154 Interrupt Status (Non-secure) bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	earlywrrsperr_irq_stat	HSNI implements an interrupt mechanism to signal imprecise errors that are detected on actual write responses received for requests for which early write responses were already provided.
[0]	nonmodsplit_irq_stat	If there is a burst split, an interrupt is generated if a non-modifiable transaction is split.

Note

A read of 1 for a field indicates that the associated interrupt event has been triggered. A write of 1 to a field in this register clears the associated interrupt event. The interrupt is asserted whenever the appropriate bits in this register are set to 1.

HSNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

This register is the interrupt mask of Non-secure transactions.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.9.1 HSNI registers summary* on page 3-214.

The following figure shows the bit assignments.

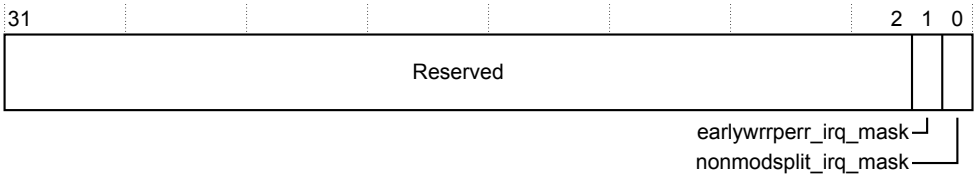

Figure 3-147 HSNI_INTERRUPT_MASK_NS bit assignments

The following table shows the bit descriptions.

Table 3-155 HSNI_INTERRUPT_MASK_NS bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	earlywrrsperr_irq_mask	Mask the early write response with imprecise error interrupt.
[0]	nonmodsplit_irq_mask	Mask the non-modifiable burst split interrupt.

Note

A value of 1 indicates that the interrupt event is masked.

3.10 AHB Master Network Interface registers

This section describes the NI-700 *AHB Master Network Interface* (HMNI) registers. It contains a summary of the master interface registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.10.1 HMNI registers summary

This register summary lists the NI-700 HMNI registers and some key characteristics.

The following table shows the slave interface registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to your SoC implementation documentation. The offset of each register from the base address is fixed.

Table 3-156 HMNI registers summary

Offset	Name	Type	Reset	Width	Description
0x000	HMNI_NODE_TYPE	RO	0x0008	32	*HMNI_NODE_TYPE, Node type register for HMNI registers* on page 3-238
0x004	HMNI_NODE_INFO	RO	0x0000	10	*HMNI_NODE_INFO, Node information for HMNI register* on page 3-238
0x040	HMNI_NODE_FEAT	RAZ	0x0000	32	*HMNI_NODE_FEAT, Node features register* on page 3-240
0x008	HMNI_SECR_ACC	RW	0x000	2	*HMNI_SECR_ACC, Secure access register* on page 3-240
0x044	HMNI_CTRL	RW	0x0	1	*HMNI_CTRL, HMNI control register* on page 3-241
0x00C	HMNI_PMUSELA	RW	0x0000	32	*HMNI_PMUSELA, Configure HMNI crossbar register* on page 3-242
0x010	HMNI_PMUSELB	RW	0x0000	32	*HMNI_PMUSELB, Configure HMNI crossbar register* on page 3-243
0x014	HMNI_INTERFACEID_0-3	RO	Configuration dependent	32	*HMNI_INTERFACEID, Configure HMNI interface IDs 0-3* on page 3-243
0x018	HMNI_INTERFACEID_4-7	RAZ		32	*HMNI_INTERFACEID, Configure HMNI interface IDs 4-7* on page 3-244
0x01C	HMNI_INTERFACEID_8-11	RAZ		32	*HMNI_INTERFACEID, Configure HMNI interface IDs 8-11* on page 3-245
0x020	HMNI_INTERFACEID_12-15	RAZ		32	*HMNI_INTERFACEID, Configure HMNI interface IDs 12-15* on page 3-246
0x080	HMNI_SILDBG	RW/RO	0x00	32	*HMNI_SILDBG, HMNI Silicon debug monitor register* on page 3-246
0x0F0	HMNI_INTERRUPT_STATUS	RW	0x00	2	*HMNI_INTERRUPT_STATUS, Interrupt status register* on page 3-247
0x0F4	HMNI_INTERRUPT_MASK	RW	0x00	2	*HMNI_INTERRUPT_MASK, Interrupt mask register* on page 3-248

Table 3-156 HMNI registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x0F8	HMNI_INTERRUPT_STATUS_NS	RW	0x00	2	*HMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register* on page 3-249
0x0FC	HMNI_INTERRUPT_MASK_NS	RW	0x00	2	*HMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register* on page 3-249

3.10.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

HMNI_NODE_TYPE, Node type register for HMNI registers

This register identifies the node type as a node for HMNI registers.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

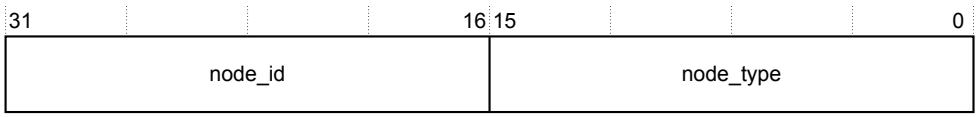

Figure 3-148 HMNI_NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-157 HMNI_NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The HMNI ID that is assigned during network construction.
[15:0]	node_type	The value of this field is 0x0008, and it identifies the associated node type as a node for NI-700 HMNI registers.

HMNI_NODE_INFO, Node information for HMNI register

This register provides node information for HMNI, such as data width.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

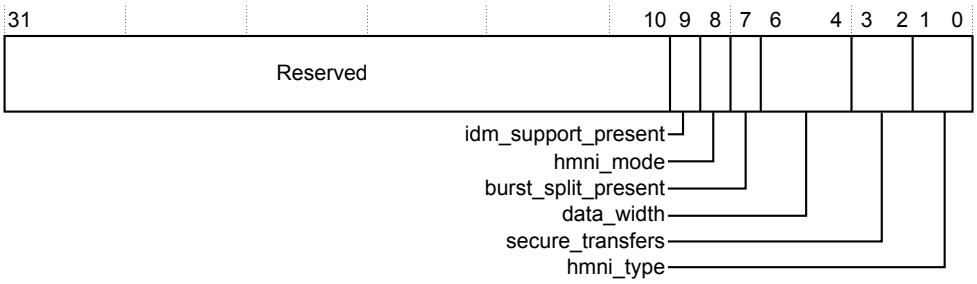

Figure 3-149 HMNI_NODE_INFO bit assignments

The following table shows the bit descriptions.

Table 3-158 HMNI_NODE_INFO bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9]	idm_support_present	IDM support present **0** IDM support logic is not present. **1** IDM support logic is present.
[8]	hmni_mode	HMNI mode **0** HMNI is not in mirror mode. **1** HMNI is in mirror mode.
[7]	burst_split_present	Burst split present **0** Burst split logic is not present. **1** Burst split logic is present.
[6:4]	data_width	Data width, HSIZE encoded `0b000` Reserved `0b001` Reserved `0b010` 4 bytes `0b011` 8 bytes `0b100` 16 bytes `0b101` 32 bytes `0b110` 64 bytes `0b111` 128 bytes

Table 3-158 HMNI_NODE_INFO bit descriptions (continued)

Bits	Name	Description
[3:2]	secure_transfers	`0b00` If secure_transfers = 00 the software programs this register to set the security attribute of the downstream slave of this master interface. **Note** If secure_transfers = 02, then transfers are always set to Secure. The downstream AHB slave interface assets of the master are Secure. Therefore only Secure requests can travel downstream. **Note** If secure_transfers = 03 then transfers are always Non-secure. The downstream AHB slave interface assets of the master are Non-secure. Both Secure and Non-secure requests can travel downstream.
[1:0]	hmni_type	HMNI type and property **0** Extended memory type **1** Exclusive transfers

HMNI_NODE_FEAT, Node features register

This register configures the node features.

Usage constraints
Accessible using only Secure accesses.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

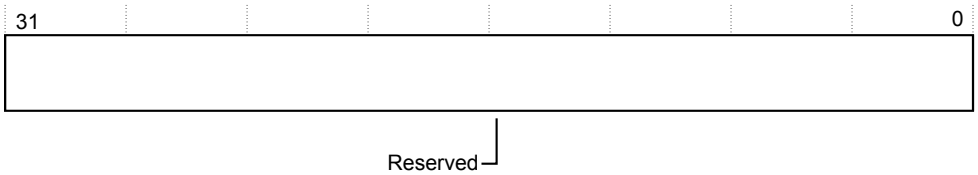

Figure 3-150 HMNI_NODE_FEAT bit assignments

The following table shows the bit descriptions.

Table 3-159 HMNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:0]	-	Reserved

HMNI_SECR_ACC, Secure access register

This register controls Secure access.

Usage constraints
Accessible using Secure transactions only.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

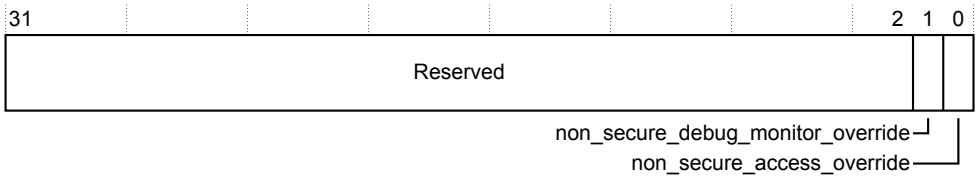

Figure 3-151 HMNI_SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-160 HMNI_SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	non_secure_debug_monitor_override	Non-secure debug monitor override: **0** Disable Non-secure access to the NI-700 PMU and interface registers. **1** Enable Non-secure access to the NI-700 PMU and interface registers.
[0]	non_secure_access_override	Non-secure access override: **0** Disable Non-secure access to the Secure NI-700 registers in this register region. **1** Enable Non-secure access to the Secure NI-700 registers in this register region.

HMNI_CTRL, HMNI control register

This register controls how transactions access components that are attached to the master.

Usage constraints
Accessible using only Secure accesses, unless you set the *HMNI_SECR_ACC, Secure access register* on page 3-240 to permit Non-secure accesses.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

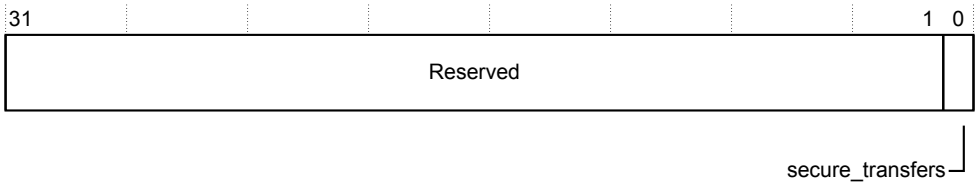

Figure 3-152 HMNI_CTRL bit assignments

The following table shows the bit descriptions.

Table 3-161 HMNI_CTRL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	secure_ctrl	If the secure_transfers field of the HMNI NODE_INFO registry is 00, it encodes a software programmable registry. Therefore the secure_ctrl field marks downstream slaves as Secure or Non-secure based on its configuration setting. **0** Secure. Only Secure transactions can travel downstream. **1** Non-secure. Both Secure and Non-secure transactions can travel downstream. If the incoming request is Non-secure, and the downstream slave is configured as Secure, then the transaction is not sent downstream. A Non-secure read transaction returns zero data. The data corresponding to a Non-secure write transaction is dropped but a protocol-compliant write response is returned. The read or write response does not contain an error indication. If secure_transfers = 02 or secure_transfers = 03, then the **HNONSEC** pin is unavailable. However the interface security attribute is fixed at build time to either Always Secure or Always Non-secure. Therefore this register bit becomes read-only. However if secure_transfers = 03 the reset value is 1. If secure_transfers = 02 the reset value is 0.

HMNI_PMUSELA, Configure HMNI crossbar register

This register is used to select the event values in the HMNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *HMNI_SECR_ACC, Secure access register* on page 3-240 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

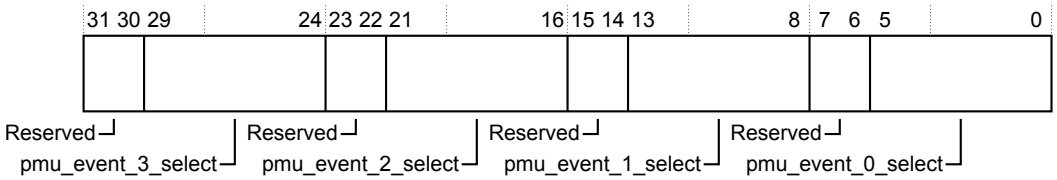

Figure 3-153 HMNI_PMUSELA bit assignments

The following table shows the bit descriptions.

Table 3-162 HMNI_PMUSELA bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_3_select	PMU event 3 select
[23:22]	-	Reserved
[21:16]	pmu_event_2_select	PMU event 2 select

Table 3-162 HMNI_PMUSELA bit descriptions (continued)

Bits	Name	Description
[15:14]	-	Reserved
[13:8]	pmu_event_1_select	PMU event 1 select
[7:6]	-	Reserved
[5:0]	pmu_event_0_select	PMU event 0 select

HMNI_PMUSELB, Configure HMNI crossbar register

This register is used to select the event values in the HMNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *HMNI_SECR_ACC, Secure access register* on page 3-240 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

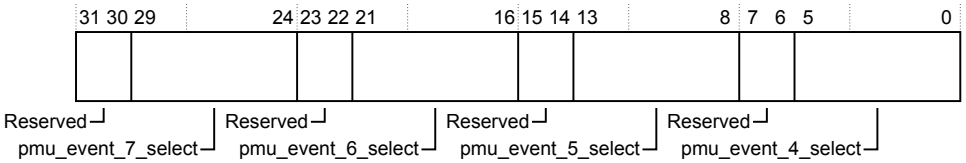

Figure 3-154 HMNI_PMUSELB bit assignments

The following table shows the bit descriptions.

Table 3-163 HMNI_PMUSELB bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_7_select	PMU event 7 select
[23:22]	-	Reserved
[21:16]	pmu_event_6_select	PMU event 6 select
[15:14]	-	Reserved
[13:8]	pmu_event_5_select	PMU event 5 select
[7:6]	-	Reserved
[5:0]	pmu_event_4_select	PMU event 4 select

HMNI_INTERFACEID, Configure HMNI interface IDs 0-3

To configure HMNI interface IDs 0-3, use offset `0x014` in the HMNI_INTERFACEID register.

Usage constraints

None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.10.1 HMNI registers summary* on page 3-237.

Note

The AHB master network interface (HMNI) node contains a single AHB or ACE-Lite interface connected to it. Therefore, HMNI interface ID 0 is the only meaningful interface ID value which is read from interface_0, bits [7:0], field of the HMNI_INTERFACEID_0-3 register. The remaining fields, bits [31:8], in the HMNI_INTERFACEID_0-3 register are all Reserved. Similarly, the other HMNI interface ID registers 4-7, 8-11 and 12-15 are all Reserved.

The following figure shows the bit assignments.

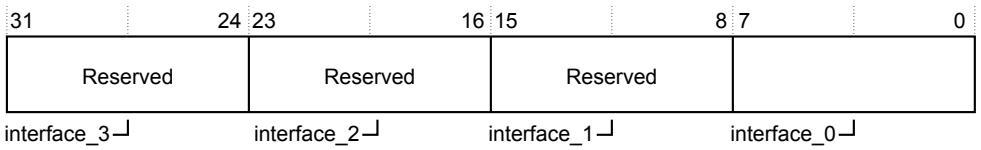

Figure 3-155 HMNI_INTERFACEID bit assignments, HMNI interface IDs 0-3

The following table shows the bit descriptions.

Table 3-164 HMNI_INTERFACEID descriptions, HMNI interface IDs 0-3

Bits	Name	Description
[31:24]	interface_3	Reserved
[23:16]	interface_2	Reserved
[15:8]	interface_1	Reserved
[7:0]	interface_0	HMNI interface ID 0

HMNI_INTERFACEID, Configure HMNI interface IDs 4-7

To configure HMNI interface IDs 4-7, use offset `0x018` in the HMNI_INTERFACEID register.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

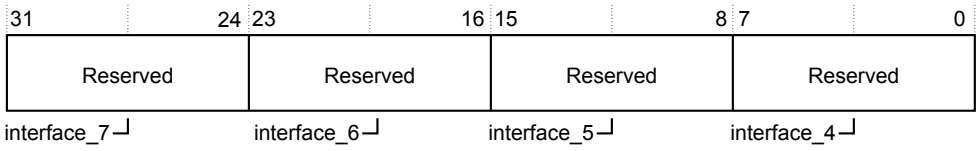

Figure 3-156 HMNI_INTERFACEID bit assignments, HMNI interface IDs 4-7

The following table shows the bit descriptions.

Table 3-165 HMNI_INTERFACEID descriptions, HMNI interface IDs 4-7

Bits	Name	Description
[31:24]	interface_7	Reserved
[23:16]	interface_6	Reserved
[15:8]	interface_5	Reserved
[7:0]	interface_4	Reserved

HMNI_INTERFACEID, Configure HMNI interface IDs 8-11

To configure HMNI interface IDs 8-11, use offset `0x01C` in the HMNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

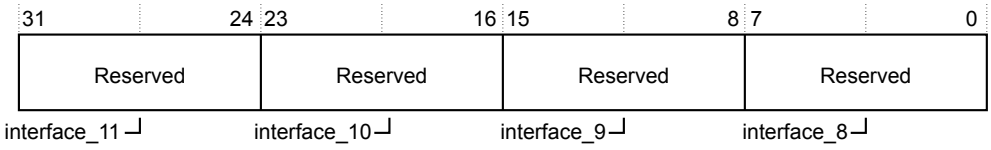

Figure 3-157 HMNI_INTERFACEID bit assignments, HMNI interface IDs 8-11

The following table shows the bit descriptions.

Table 3-166 HMNI_INTERFACEID descriptions, HMNI interface IDs 8-11

Bits	Name	Description
[31:24]	interface_11	Reserved
[23:16]	interface_10	Reserved
[15:8]	interface_9	Reserved
[7:0]	interface_8	Reserved

HMNI_INTERFACEID, Configure HMNI interface IDs 12-15

To configure HMNI interface IDs 12-15, use offset `0x020` in the HMNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

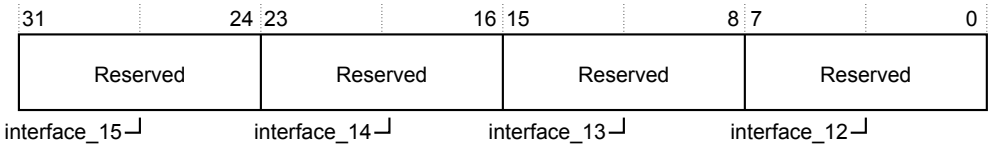

Figure 3-158 HMNI_INTERFACEID bit assignments, HMNI interface IDs 12-15

The following table shows the bit descriptions.

Table 3-167 HMNI_INTERFACEID descriptions, HMNI interface IDs 12-15

Bits	Name	Description
[31:24]	interface_15	Reserved
[23:16]	interface_14	Reserved
[15:8]	interface_13	Reserved
[7:0]	interface_12	Reserved

HMNI_SILDBG, HMNI Silicon debug monitor register

This register monitors the status of NI-700 master interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the *HMNI_SECR_ACC, Secure access register* on page 3-240 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access this register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

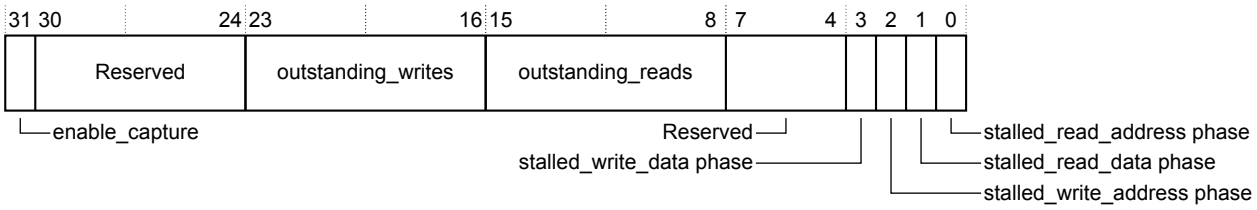

Figure 3-159 HMNI_SILDBG bit assignments

The following table shows the bit descriptions.

Table 3-168 HMNI_SILDBG bit descriptions

Bits	Name	Description
[31]	enable_capture	Enable capture
[30:24]	-	Reserved
[23:16]	outstanding_writes	Indicates that the interface has outstanding writes. Maximum value is 1.
[15:8]	outstanding_reads	Indicates that the interface has outstanding read requests. Maximum value is 1.
[7:4]	-	Reserved
[3]	stalled_write_data_phase	Prior write address phase, **HREADY** LOW
[2]	stalled_write_address_phase	**HTRANS[1]** HIGH, **HWRITE** HIGH, **HREADY** LOW
[1]	stalled_read_data_phase	Prior read address phase, **HREADY** LOW
[0]	stalled_read_address_phase	**HTRANS[1]** HIGH, **HWRITE** LOW, **HREADY** LOW

Note

Arm recommends you enable capture when the interface is in a quiescent state. If capture is enabled in the middle of the address or data phase of an ongoing request, it is possible the stalls are not captured correctly.

HMNI_INTERRUPT_STATUS, Interrupt status register

This register indicates the interrupt status of Secure transactions.

Usage constraints

Accessible using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

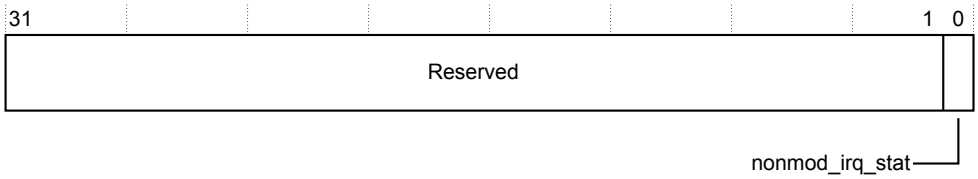

Figure 3-160 HMNI_INTERRUPT_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-169 HMNI_INTERRUPT_STATUS bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	nonmod_irq_stat	If there is a burst split, an interrupt is generated if a non--modifiable transaction is split.

Note

A read of 1 for a field indicates that the associated interrupt event has been triggered. A write of 1to a field in this register clears the associated interrupt event. The interrupt is asserted whenever the appropriate bits in this register are set to 1.

HMNI_INTERRUPT_MASK, Interrupt mask register

This register is the interrupt mask of Secure transactions.

Usage constraints

Accessible using Secure transactions only.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

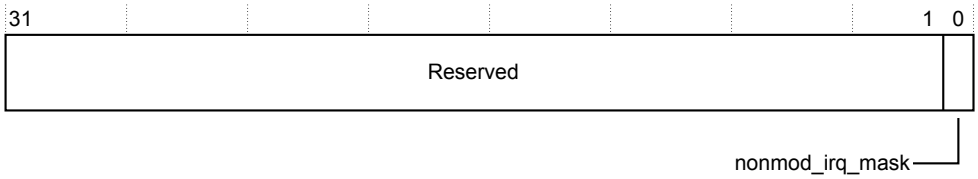

Figure 3-161 HMNI_INTERRUPT_MASK bit assignments

The following table shows the bit descriptions.

Table 3-170 HMNI_INTERRUPT_MASK bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	nonmod_irq_mask	Mask the non-modifiable split interrupt

Note

A value of 1 indicates that the interrupt event is masked.

HMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register

This register indicates the interrupt status of Non-secure transactions.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

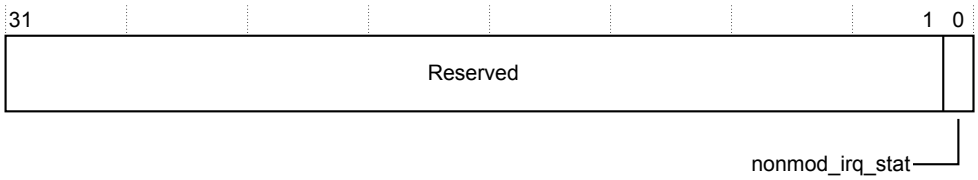

Figure 3-162 HMNI_INTERRUPT_STATUS_NS bit assignments

The following table shows the bit assignments.

Table 3-171 HMNI_INTERRUPT_STATUS_NS bit assignments

Bits	Name	Description
[31:1]	-	Reserved
[0]	nonmod_irq_stat	If there is a burst split, an interrupt is generated if a non-modifiable transaction is split.

Note

A read of 1 for a field indicates that the associated interrupt event has been triggered. A write of 1 to a field in this register clears the associated interrupt event. The interrupt is asserted whenever the appropriate bits in this register are set to 1 .

HMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

This register is the interrupt mask of Non-secure transactions.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.10.1 HMNI registers summary* on page 3-237.

The following figure shows the bit assignments.

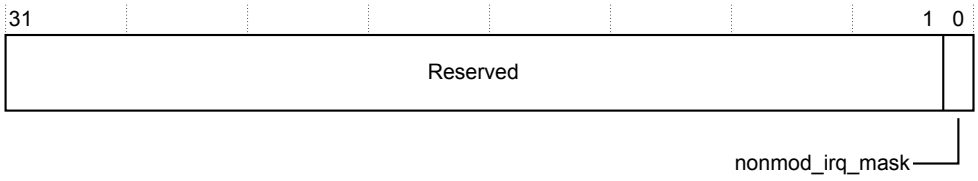

Figure 3-163 HMNI_INTERRUPT_MASK_NS bit assignments

The following table shows the bit descriptions.

Table 3-172 HMNI_INTERRUPT_MASK_NS bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	nonmod_irq_mask	Mask the non-modifiable split interrupt.

Note

A value of 1 indicates that the interrupt event is masked.

3.11 Network Interface IDM registers

This section describes the NI-700 IDM registers. It contains a summary of the NI registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.11.1 Network Interface IDM registers summary

Enabling IDM functionality on a slave or master NI adds IDM registers to the 4KB configuration region for the NI. The IDM registers control IDM behavior for the associated NI.

IDM functionality can be configured on all NI-700 NI types. If IDM is enabled on an NI, the IDM registers start as offset `0x100` from the base address of the NI. The register name is prefixed with the type of NI to which it belongs, for example ASNI_IDM_DEVICE_ID.

The NI IDM registers are shown in the following table in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to your SoC implementation documentation.

Table 3-173 NI IDM registers summary

Offset	Name	Type	Reset	Width	Description
`0x100`	IDM_DEVICE_ID	RO	Device-specific. Use the NI-700 tooling to configure static value.	32	*IDM_DEVICE_ID, Device ID register* on page 3-252
`0x104`	IDM_CONFIG	RO	Device-specific	9	*IDM_CONFIG, IDM configuration register* on page 3-253
`0x108`	IDM_ERRCTLR	RW	`0x0`	3	*IDM_ERRCTLR* on page 3-254
`0x110`	IDM_ERRSTATUS	RW or RO	`0x0`	32	*IDM_ERRSTATUS* on page 3-255
`0x114`	IDM_ERRADDR_LSB	RO	`0x0`	32	*IDM_ERRADDR_LSB* on page 3-256
`0x118`	IDM_ERRADDR_MSB	RO	`0x0`	32	*IDM_ERRADDR_MSB* on page 3-257
`0x128`	IDM_ERRMISC0	RO	`0x0`	32	*IDM_ERRMISC0* on page 3-257
`0x12C`	IDM_ERRMISC1	RO	`0x0`	25	*IDM_ERRMISC1* on page 3-258
`0x130`	IDM_ACCESS_CONTROL	RW	`0x0`	1	*IDM_ACCESS_CONTROL* on page 3-259
`0x134`	IDM_ACCESS_STATUS	RW or RO	`0x0`	14	*IDM_ACCESS_STATUS* on page 3-260
`0x138`	IDM_ACCESS_READID	RO	`0x0`	8	*IDM_ACCESS_READID* on page 3-261
`0x13C`	IDM_ACCESS_WRITEID	RO	`0x0`	8	*IDM_ACCESS_WRITEID* on page 3-261
`0x140`	IDM_RESET_CONTROL	RW	`0x0`	1	*IDM_RESET_CONTROL* on page 3-262
`0x144`	IDM_RESET_STATUS	RO	`0x0`	6	*IDM_RESET_STATUS* on page 3-263
`0x148`	IDM_RESET_READID	RO	`0x0`	8	*IDM_RESET_READID* on page 3-264
`0x14C`	IDM_RESET_WRITEID	RO	`0x0`	8	*IDM_RESET_WRITEID* on page 3-265
`0x150`	IDM_TIMEOUT_CONTROL	RW	`0x0`	1	*IDM_TIMEOUT_CONTROL* on page 3-266

Table 3-173 NI IDM registers summary (continued)

Offset	Name	Type	Reset	Width	Description
0x154	IDM_TIMEOUT_VALUE	RW	0x0	32	*IDM_TIMEOUT_VALUE* on page 3-266
0x158	IDM_INTERRUPT_STATUS	RW	0x0	4	*IDM_INTERRUPT_STATUS* on page 3-267
0x15C	IDM_INTERRUPT_MASK	RW	0x0	4	*IDM_INTERRUPT_MASK* on page 3-268
0x160	IDM_ERRSTATUS_NS	RW or RO	0x0	32	*IDM_ERRSTATUS_NS* on page 3-269
0x164	IDM_ERRADDR_LSB_NS	RO	0x0	32	*IDM_ERRADDR_LSB_NS* on page 3-271
0x168	IDM_ERRADDR_MSB_NS	RO	0x0	32	*IDM_ERRADDR_MSB_NS* on page 3-271
0x178	IDM_ERRMISC0_NS	RO	0x0	32	*IDM_ERRMISC0_NS* on page 3-272
0x17C	IDM_ERRMISC1_NS	RO	0x0	25	*IDM_ERRMISC1_NS* on page 3-272
0x184	IDM_ACCESS_STATUS_NS	RW or RO	0x0	12	*IDM_ACCESS_STATUS_NS* on page 3-273
0x188	IDM_ACCESS_READID_NS	RO	0x0	8	*IDM_ACCESS_READID* on page 3-274
0x18C	IDM_ACCESS_WRITEID_NS	RO	0x0	8	*IDM_ACCESS_WRITEID_NS* on page 3-275
0x194	IDM_RESET_STATUS_NS	RO	0x0	4	*IDM_RESET_STATUS_NS* on page 3-275
0x198	IDM_RESET_READID_NS	RO	0x0	8	*IDM_RESET_READID_NS* on page 3-276
0x19C	IDM_RESET_WRITEID_NS	RO	0x0	8	*IDM_RESET_WRITEID_NS* on page 3-277
0x1A8	IDM_INTERRUPT_STATUS_NS	RW	0x0	4	*IDM_INTERRUPT_STATUS_NS* on page 3-277
0x1AC	IDM_INTERRUPT_MASK_NS	RW	0x0	4	*IDM_INTERRUPT_MASK_NS* on page 3-278

3.11.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

IDM_DEVICE_ID, Device ID register

This register indicates the statically configured device ID value and is implemented if IDM is enabled.

Usage constraints
None.

Configurations
If IDM is enabled, this register is implemented in NI-700.

Attributes
For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

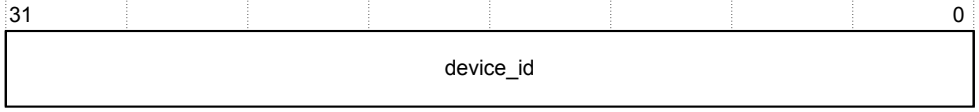

Figure 3-164 ASNI_IDM_DEVICE_ID bit assignments

The following table shows the bit descriptions.

Table 3-174 IDM_DEVICE_ID bit descriptions

Bits	Name	Description
[31:0]	device_id	Returns statically configured ID value

IDM_CONFIG, IDM configuration register

This register enables transaction logging, error detection, timeout detection, access control, and reset control.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

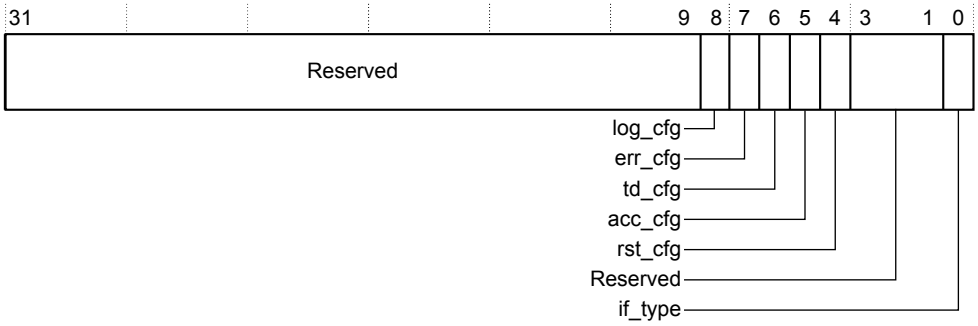

Figure 3-165 IDM_CONFIG bit assignments

The following table shows the bit descriptions.

Table 3-175 IDM_CONFIG bit descriptions

Bits	Name	Description
[31:9]	-	Reserved
[8]	log_cfg	Transaction logging present
[7]	err_cfg	Error detection present
[6]	td_cfg	Timeout detection present
[5]	acc_cfg	Access control present
[4]	rst_cfg	Reset control present
[3:1]	-	Reserved
[0]	if_type	Interface type: **0** Slave **1** Master

Note

For this r2p0 release and previous releases of NI-700, if IDM support is enabled, then all of these features are enabled.

IDM_ERRCTLR

This register controls how errors are handled.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

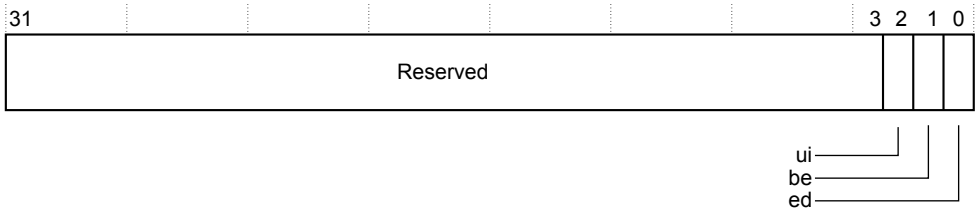

Figure 3-166 IDM_ERRCTLR bit assignments

The following table shows the bit descriptions.

Table 3-176 IDM_ERRCTLR bit descriptions

Bits	Name	Description
[31:3]	-	Reserved
[2]	ui	Enable error interrupt for uncorrected error as indicated by IDM_ERRSTATUS.UE fields
[1]	be	Enable bus error detection: **0** Disabled **1** Enabled when an error is detected and idm_errctlr [ed] is enabled: • Logged if the transaction log is empty. If not, the logged transaction overflow bit is set. • An error interrupt event is generated (unless masked)
[0]	ed	Error detection global enable **0** Disabled **1** Enabled when an error is detected • Enabled. When an error, time out error or bus error, is detected and its respective detection enable register bit, Timeout_control[TD_EN], or idm_errctlr[be] is also set. • Logged if the transaction log is empty. If not, the logged transaction overflow bit is set. An error interrupt event is generated, unless masked.

IDM_ERRSTATUS

This register indicates the error status of Secure transactions. If timeout is configured, but error logging is not configured then OF is never set and SERR only reads as no error or timeout error.

Usage constraints

Accessible using only Secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

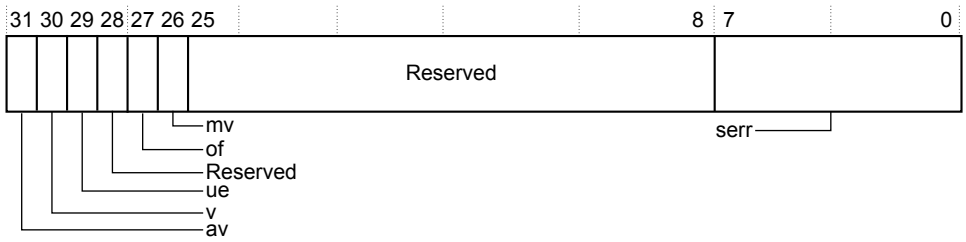

Figure 3-167 idm_errstatus bit assignments

The following table shows the bit descriptions.

Table 3-177 IDM_ERRSTATUS bit descriptions

Bits	Name	Description
[31]	av	Address valid. The values are: **0** ERRADDR is not valid. **1** ERRADDR contains an address that is associated with the highest priority error which this record records. This bit ignores writes if IDM_ERRSTATUS.UE is set to 1 and is not cleared to zero in the same write. This bit is read, or write 1 to clear.
[30]	v	Status register is valid. The values are: **0** IDM_ERRSTATUS not valid **1** IDM_ERRSTATUS valid. At least one error has been recorded. This bit ignores writes if any of the following fields is set to 1 and is not being cleared to zero in the same write: • IDM_ERRSTATUS.UE • IDM_ERRSTATUS.AV • IDM_ERRSTATUS.OF • IDM_ERRSTATUS.MV This bit is read, or write 1 to clear.
[29]	ue	Uncorrected error. The values are: **0** No errors have been detected, or all detected errors have been either corrected or deferred. **1** At least one detected error was not corrected and not deferred. This bit ignores writes if IDM_ERRSTATUS.OF is set to 1 and is not being cleared to zero in the same write. This bit is not valid and reads UNKNOWN if IDM_ERRSTATUS.V is set to 0. This bit is read, or write 1 to clear.

Table 3-177 IDM_ERRSTATUS bit descriptions (continued)

Bits	Name	Description
[28]	-	Reserved
[27]	of	Returns whether a second error has been received while handling a first error. The values are: **1** Second error received **0** No other error received This bit is read, or write 1 to clear.
[26]	mv	Miscellaneous registers valid. The values are: **0** IDM_ERRMISC0 and IDM_ERRMISC1 not valid **1** The IMPLEMENTATION DEFINED contents of the IDM_ IDM_ERRMISC0 and IDM_ERRMISC1 registers contains additional information for an error that this record records. This bit ignores writes if IDM_ERRSTATUS.UE is set to 1, and is not being cleared to 0 in the same write. This bit is a read, or write 1 to clear.
[25:8]	-	Reserved
[7:0]	serr	Primary error code. Indicates the type of error. The values are: **00** No error **13** Illegal address - decode error **18** Error response from slave **20** Internal timeout

IDM_ERRADDR_LSB

This register is the error log of Secure transactions.

Usage constraints
Accessible using only Secure accesses.

Configurations
If IDM is enabled, this register is implemented in NI-700.

Attributes
For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

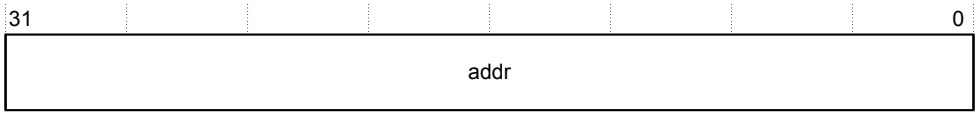

Figure 3-168 IDM_ERRADDR_LSB bit assignments

The following table shows the bit descriptions.

Table 3-178 IDM_ERRADDR_LSB bit descriptions

Bits	Name	Description
[31:0]	addr	Returns bits 31:0 of an address causing an error

IDM_ERRADDR_MSB

This register is the error log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

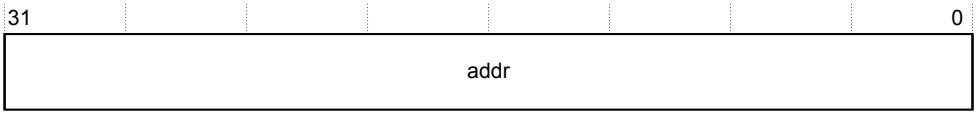

Figure 3-169 IDM_ERRADDR_MSB bit assignments

The following table shows the bit descriptions.

Table 3-179 IDM_ERRADDR_MSB bit descriptions

Bits	Name	Description
[31:0]	addr	Returns bits[63:32] of an address causing an error

IDM_ERRMISC0

This register is the error log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

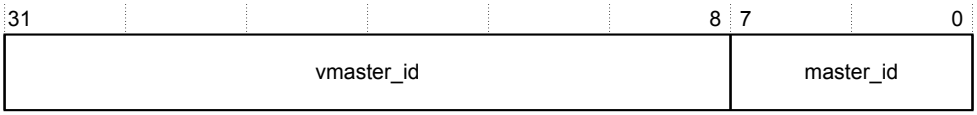

Figure 3-170 IDM_ERRMISC0 bit assignments

The following table shows the bit descriptions.

Table 3-180 IDM_ERRMISC0 bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **AxID** into ASNI of the transaction causing an error. The assumption here is there is no manipulation of incoming AXI **AxID** in ASNI.
[7:0]	master_id	The ASNI node ID of the transaction causing an error.

IDM_ERRMISC1

This register is the error log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

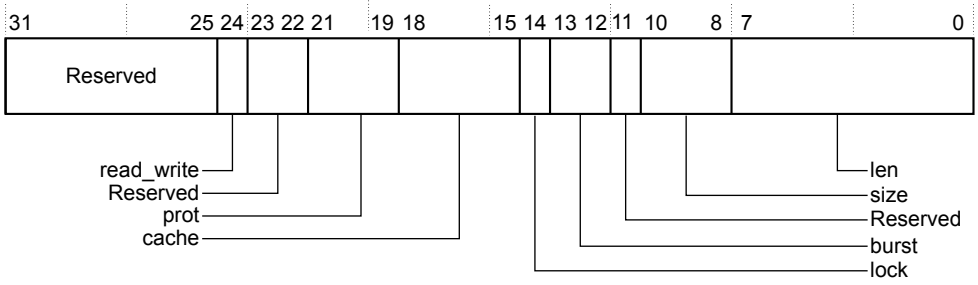

Figure 3-171 IDM_ERRMISC1 bit assignments

The following table shows the bit descriptions.

Table 3-181 IDM_ERRMISC1 bit descriptions

Bits	Name	Description
[31:25]	-	Reserved
[24]	read_write	The AXI read or write information of a transaction causing an error **1** Write **0** Read
[23:22]	-	Reserved
[21:19]	prot	The AXI prot information of a transaction causing an error.
[18:15]	cache	The AXI cache information of a transaction causing an error.
[14]	lock	The AXI lock information of a transaction causing an error.
[13:12]	burst	The AXI burst information of a transaction causing an error.
[11]	-	Reserved

Table 3-181 IDM_ERRMISC1 bit descriptions (continued)

Bits	Name	Description
[10:8]	size	The AXI size information of a transaction causing an error.
[7:0]	len	The AXI len information of a transaction causing an error.

IDM_ACCESS_CONTROL

This register controls the state, gated or ungated, of a device.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

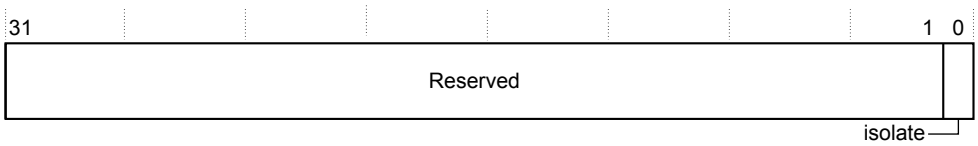

Figure 3-172 IDM_ACCESS_CONTROL bit assignments

The following table shows the bit descriptions.

Table 3-182 IDM_ACCESS_CONTROL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	isolate	Perform gating off a device Reading 1 indicates that the slave device is gated or isolated. Reading 0 indicates that the slave device is ungated or de-isolated. Write 1 to enter gated state Write 0 to exit gated state There is some delay to updating this field with the intended write value. Exit from gated state is only successful if there are no outstanding transactions and all error status register bits are cleared. Entry into gated state is only successful if there are no outstanding transactions. While in pending isolation entry state or in active isolation state, a write of 1 to this bit causes reentry to isolation state. The write causes the write_received and read_received fields of IDM_ACCESS_STATUS and the IDM_access_readid and IDM_access_writeid registers to be cleared. A write of 0 is ignored. While in pending isolation exit state, a write of 0 to this bit causes a re-exit to the exit state. The write causes the write_received and read_received fields of IDM_ACCESS_STATUS, and the IDM_access_readid and IDM_access_writeid registers to be cleared. A write of 1 is ignored.

IDM_ACCESS_STATUS

This register indicates the Secure access status of a device.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

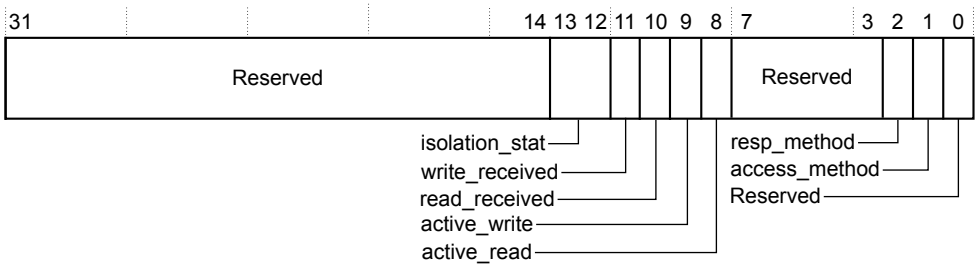

Figure 3-173 IDM_ACCESS_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-183 IDM_ACCESS_STATUS bit descriptions

Bits	Name	Description
[31:14]	0	Reserved, *UNDEFINED*, write as zero
[13:12]	0	Isolation status: **00** Isolation exit or entry is successful or not in gated or isolation state **01** Isolation exit is unsuccessful or pending because of uncleared error status bits, idm_errstatus **10** Isolation entry is unsuccessful or pending because of outstanding transactions **11** Reserved
[11]	0	A 1 indicates that an active write transaction has occurred since the IDM entered the isolation state. This bit is cleared to zero on: • Reentry to isolation state. Write 1 into bit 0 of IDM_ACCESS_CONTROL register when already in pending isolation entry state, or isolation active state. • Re-exit from isolation state. Write 0 into bit 0 of IDM_ACCESS_CONTROL register when already in pending isolation exit state.
[10]	0	A 1 indicates that an active read transaction has occurred since the IDM entered the isolation state. This bit is cleared to zero on: • Reentry to isolation state. Write 1 into bit 0 of IDM_ACCESS_CONTROL register when already in pending isolation entry state, or isolation active state. • Re-exit from isolation state. Write 0 into bit 0 of IDM_ACCESS_CONTROL register when already in pending isolation exit state.
[9]	0	Active write transactions A 1 indicates there is at least one write transaction currently in progress.

Table 3-183 IDM_ACCESS_STATUS bit descriptions (continued)

Bits	Name	Description
[8]	0	Active read transactions A 1 indicates there is at least one read transaction currently in progress.
[7:3]	0	Reserved, *UNDEFINED*, write as zero
[2]	0	Indicates device generates errors in gated access
[1]	1	Wait for all outstanding to complete, then block input
[0]	0	Reserved, *UNDEFINED*, write as zero

IDM_ACCESS_READID

This register is the access log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

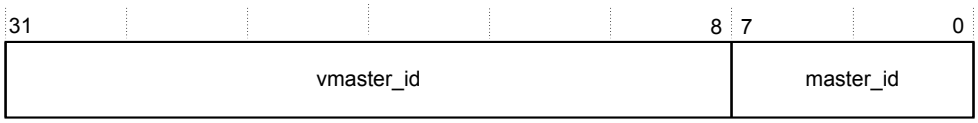

Figure 3-174 IDM_ACCESS_READID bit assignments

The following table shows the bit descriptions.

Table 3-184 IDM_ACCESS_READID bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **ARID** into endpoint of the first transaction to arrive after isolation when the access_status active_read field is HIGH. The assumption here is there is no manipulation of incoming AXI **ARID** signal in ASNI.
[7:0]	master_id	The ASNI node ID of the first transaction to arrive after isolation when the access_status active_read field is HIGH.

IDM_ACCESS_WRITEID

This register is the access log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

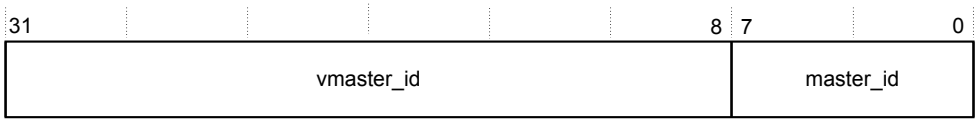

Figure 3-175 IDM_ACCESS_WRITEID bit assignments

The following table shows the bit descriptions.

Table 3-185 IDM_ACCESS_WRITEID bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **AWID** into the endpoint of the first transaction to arrive after isolation when the access_status active_read field is HIGH. The assumption here is there is no manipulation of incoming AXI **AWID** signal in ASNI.
[7:0]	master_id	The ASNI node ID of the first transaction to arrive after isolation when the access_status active_write field is HIGH.

IDM_RESET_CONTROL

This register controls the reset of a device that is attached to NI-700.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

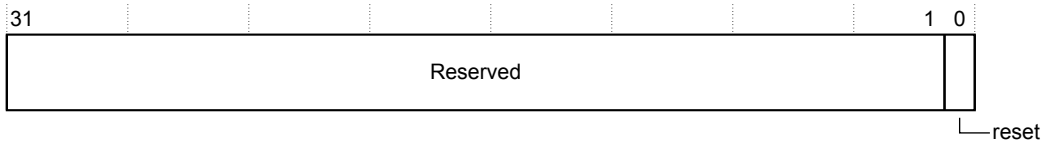

Figure 3-176 IDM_RESET_CONTROL bit assignments

The following table shows the bit descriptions:

Table 3-186 IDM_RESET_Control

Bits	Name	Description
[31:1]	-	Reserved, UNDEFINED, write as zero
[0]	reset	Performs soft reset of attached device Writes have the following effect: **1** Request attached device to enter reset. Write is ignored if it occurs before soft reset exit has occurred. **0** Request attached device to exit reset. Write is ignored if it occurs before soft reset entry has occurred. Software polls this register to determine if soft reset entry or exit has occurred, using the following values: **1** Indicates that the device is in reset **0** Indicates that the device is not in reset This register value updates to reflect a request for reset entry or reset exit, but the update can only occur after required internal conditions are met. Until these conditions are met, a read to this register returns the old value. For example, outstanding transactions currently being handled must complete before this register value updates. To ensure reset propagation within the device, it is the responsibility of the software to permit enough cycles after soft reset assertion is reflected in the reset control register before exiting soft reset by triggering a write of 0. If this responsibility is not met, the behavior is UNDEFINED or UNPREDICTABLE. When this register value is 1, the external soft reset pin that connects to the attached AXI master or slave device is asserted, using the correct polarity of the reset pin. When this register value is 0, the external soft reset pin that connects to the attached AXI master or slave device is deasserted, using the correct polarity of the reset pin. When in pending soft reset entry state or in active soft reset state, a write of 1 to this bit causes reentry to soft reset state. This write causes the write_received and read_received fields of the IDM_RESET_STATUS, the IDM_RESET_READID and IDM_RESET_WRITEID registers to be cleared. A write of 0 is ignored. While in pending soft reset exit state, a write of 0 to this bit causes re-exit to exit state. This write causes the write_received and read_received fields of IDM_RESET_STATUS, the IDM_RESET_READID and IDM_RESET_WRITEID registers to be cleared. A write of 1is ignored.

IDM_RESET_STATUS

This register indicates the reset state of a device.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

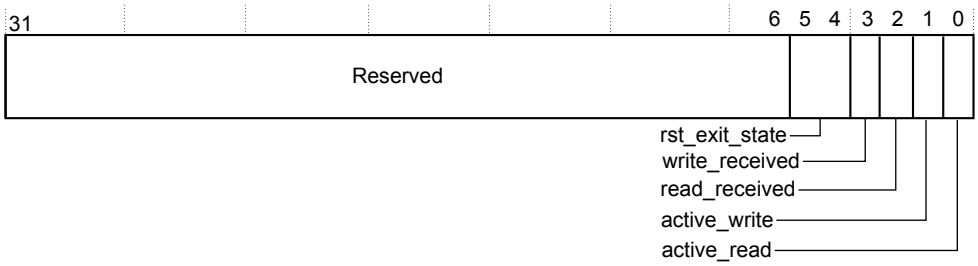

Figure 3-177 IDM_RESET_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-187 IDM_RESET_STATUS bit descriptions

Bits	Name	Description
[31:6]	-	Reserved, *UNDEFINED*, write as zero
[5:4]	rst_exit_state	Reset exit state **00** Reset exit or entry is successful or not in reset state **01** Reset exit is unsuccessful or pending because of uncleared error status bits, idm_errstatus **10** Reset exit is unsuccessful or pending because of outstanding transactions **11** Reset exit is unsuccessful or pending because of both uncleared error status bits and outstanding transactions
[3]	write_received	A 1 indicates that an active write transaction has occurred since the IDM entered the soft_reset state. This bit is cleared to zero on: • Reentry to soft reset state Write 1 into bit 0 of IDM_RESET_CONTROL register when already in pending soft reset entry state, or soft reset active state. • Re-exit from soft reset state Write 0 into bit 0 of IDM_RESET_CONTROL register when already in pending soft reset exit state.
[2]	read_received	A 1 indicates that there has been an active read transaction since a write of 1 to reset_control. This bit is cleared to zero on: • Reentry to soft reset state Write 1 into bit 0 of IDM_RESET_CONTROL register when already in pending soft reset entry state, or soft reset active state. • Re-exit from soft reset state Write 0 into bit 0 of IDM_RESET_CONTROL register when already in pending soft reset exit state.
[1]	active_write	Active write transactions A 1 indicates there is at least one write transaction currently in progress.
[0]	active_read	Active read transactions A 1 indicates there is at least one read transaction currently in progress.

IDM_RESET_READID

This register is the reset access log of Secure transactions.

Usage constraints
Accessible using only Secure accesses.

Configurations
If IDM is enabled, this register is implemented in NI-700.

Attributes
For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

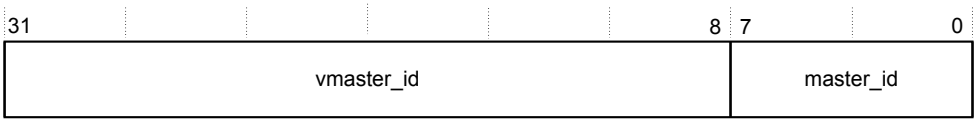

Figure 3-178 IDM_RESET_READID bit assignments

The following table shows the bit descriptions.

Table 3-188 IDM_RESET_READID bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **ARID** into the endpoint of the first transaction to arrive after isolation when the reset_status active_read field is HIGH. The assumption is there is no manipulation of incoming AXI **ARID** in ASNI.
[7:0]	master_id	The ASNI node ID of the first transaction to arrive after isolation when the reset_status active_read field is HIGH.

IDM_RESET_WRITEID

This register is the reset access log of Secure transactions.

Usage constraints
Accessible using only Secure accesses.

Configurations
If IDM is enabled, this register is implemented in NI-700.

Attributes
For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

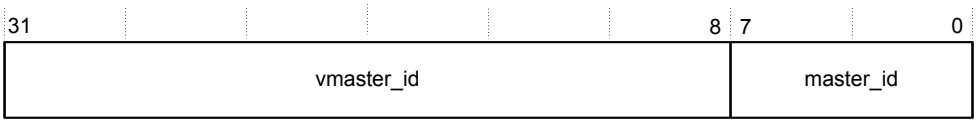

Figure 3-179 IDM_RESET_WRITEID bit assignments

The following table shows the bit descriptions.

Table 3-189 IDM_RESET_WRITEID bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **AWID** into the endpoint of the first transaction to arrive after isolation when the reset_status active_write field is HIGH. The assumption is there is no manipulation of incoming AXI **AWID** in ASNI
[7:0]	master_id	The ASNI node ID of the first transaction to arrive after isolation when the reset_status active_write field is HIGH.

IDM_TIMEOUT_CONTROL

This register is present when timeout detection is configured.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

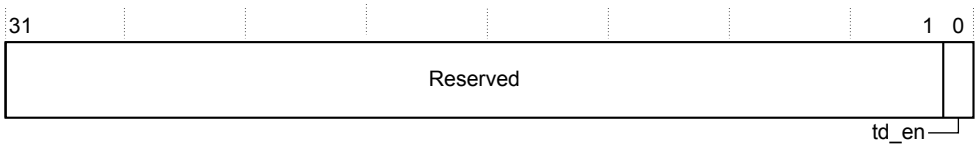

Figure 3-180 IDM_TIMEOUT_CONTROL bit assignments

The following table shows the bit descriptions.

Table 3-190 IDM_TIMEOUT_CONTROL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	td_en	Timeout detection enable **0** Disabled **1** Enabled when a timeout is detected. Logged if the transaction log is empty. If not, the logged transaction overflow bit is set. A timeout interrupt event is generated, unless it is masked.

IDM_TIMEOUT_VALUE

This register controls the duration that is used to determine if a transaction has timed out. If the timeout_en configuration parameter is 0, then the timeout_* fields that are read as zero and writes are ignored.

Usage constraints

Accessible using only Secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

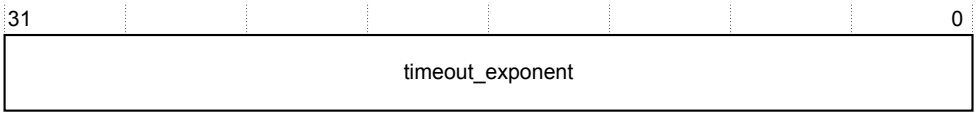

Figure 3-181 IDM_TIMEOUT_VALUE bit assignments

The following table shows the bit descriptions.

Table 3-191 IDM_TIMEOUT_VALUE bit descriptions

Bits	Name	Description
[31:0]	timeout_exponent	Controls the duration that is used to determine if a transaction has timed out. The actual duration is $2^{timeout\_exponent}$ cycles. The minimum value is 4. Values of 0, 1, 2, or 3 are treated as 4.

IDM_INTERRUPT_STATUS

This register indicates the interrupt status of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

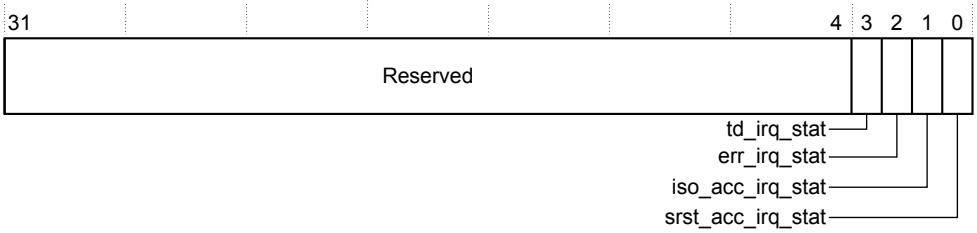

Figure 3-182 IDM_INTERRUPT_STATUS bit assignments

The following table shows the bit descriptions.

Table 3-192 IDM_INTERRUPT_STATUS bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3]	td_irq_stat	Timeout detection event Interface has detected a timeout.

Table 3-192 IDM_INTERRUPT_STATUS bit descriptions (continued)

Bits	Name	Description
[2]	err_irq_stat	Error detection event Interface has detected a protocol error.
[1]	iso_acc_irq_stat	Isolation access event Interface access while the IDM is closed.
[0]	srst_acc_irq_stat	Reset access event Interface access while the IDM is closed.

Note

A read of 1 for a field indicates that the associated interrupt event has been triggered. A write of 1 to a field in this register clears the associated interrupt event. The interrupt is asserted whenever the appropriate bits in this register are set to 1.

IDM_INTERRUPT_MASK

This register is the interrupt mask of Secure transactions.

Usage constraints

Accessible using Secure transactions only.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

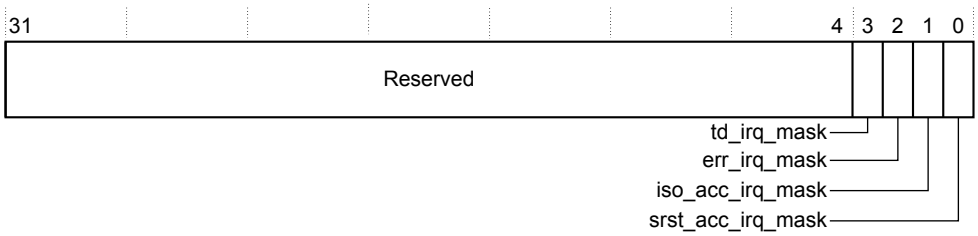

Figure 3-183 IDM_INTERRUPT_MASK bit assignments

The following table shows the bit descriptions.

Table 3-193 IDM_INTERRUPT_MASK bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3]	td_irq_mask	Timeout detection event mask
[2]	err_irq_mask	Error detection event mask

Table 3-193 IDM_INTERRUPT_MASK bit descriptions (continued)

Bits	Name	Description
[1]	iso_acc_irq_mask	Access event mask
[0]	srst_acc_irq_mask	Access event mask

Note

A value of 1 indicates that the interrupt event is masked.

IDM_ERRSTATUS_NS

This register indicates the error status of Non-secure transactions. If timeout is configured, but error logging is not configured then OF is never set. Therefore SERR only reads as no error or timeout error.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

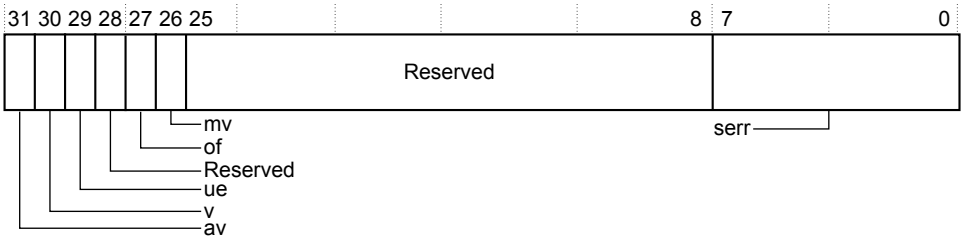

Figure 3-184 IDM_ERRSTATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-194 IDM_ERRSTATUS_NS bit descriptions

Bits	Name	Description
[31]	av	Address valid The values are: **0** ERRADDR is not valid. **1** ERRADDR contains an address that is associated with the highest priority error that this record captures. This bit ignores writes if IDM_ERRSTATUS.UE is set to 1 and is not cleared to 0 in the same write. This bit is read, or write 1 to clear.
[30]	v	Status register valid The values are: **0** IDM_ERRSTATUS is not valid. **1** IDM_ERRSTATUS is valid. At least one error has been recorded. This bit ignores writes if IDM_ERRSTATUS.UE is set to 1 and is not being cleared to 0 in the same write. This bit is read, or write 1 to clear.
[29]	ue	Uncorrected error The values are: **0** No errors have been detected, or all detected errors have been either corrected or deferred. **1** At least one detected error was not corrected and not deferred. This bit ignores writes if IDM_ERRSTATUS.OF is set to 1 and is not being cleared to 0 in the same write. This bit is not valid and reads UNKNOWN if IDM_ERRSTATUS.V is set to 0. This bit is read, or write 1 to clear.
[28]	-	Reserved
[27]	of	Returns whether a second error has been received while handling a first error. The values are: **1** Second error received **0** No other error received This bit is read, or write 1 to clear.
[26]	mv	Miscellaneous registers valid The values are: **0** IDM_ERRMISC0 and IDM_ERRMISC1 are not valid. **1** The IMPLEMENTATION DEFINED contents of the IDM_ IDM_ERRMISC0 and IDM_ERRMISC1 registers contains additional information for an error that this record captures. This bit ignores writes if IDM_ERRSTATUS.UE is set to 1, and is not being cleared to 0 in the same write. This bit is read, or write 1 to clear.

Table 3-194 IDM_ERRSTATUS_NS bit descriptions (continued)

Bits	Name	Description
[25:8]	-	Reserved
[7:0]	serr	Primary error code, indicates the type of error The values are: **00** No error **13** Illegal address - decode error **18** Error response from slave **20** Internal timeout

IDM_ERRADDR_LSB_NS

This register is the error log of Non-secure transactions.

Usage constraints
None.

Configurations
If IDM is enabled, this register is implemented in NI-700.

Attributes
For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

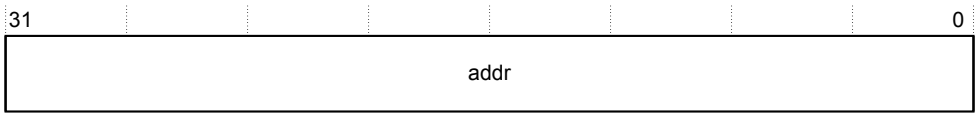

Figure 3-185 IDM_ERRADDR_LSB_NS bit assignments

The following table shows the bit descriptions.

Table 3-195 IDM_ERRADDR_LSB_NS descriptions

Bits	Name	Description
[31:0]	addr	Returns bits [31:0] of an address causing an error

IDM_ERRADDR_MSB_NS

This register is the error log of Non-secure transactions.

Usage constraints
None.

Configurations
If IDM is enabled, this register is implemented in NI-700.

Attributes
For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

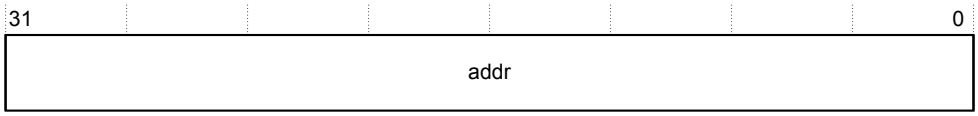

Figure 3-186 IDM_ERRADDR_MSB_NS bit assignments

The following table shows the bit descriptions.

Table 3-196 IDM_ERRADDR_MSB_NS bit descriptions

Bits	Name	Description
[31:0]	addr	Returns bits [63:32] of an address causing an error

IDM_ERRMISC0_NS

This register is the error log of Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

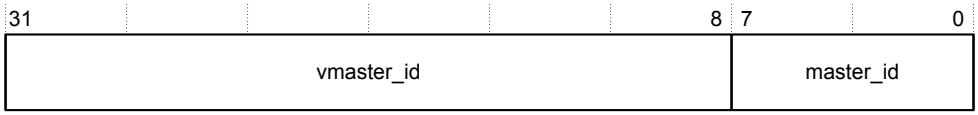

Figure 3-187 IDM_ERRMISC0_NS bit assignments

The following table shows the bit descriptions.

Table 3-197 IDM_ERRMISC0_NS descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **AxID** into ASNI of the transaction causing an error. The assumption is no manipulation of incoming AXI **AxID** in ASNI.
[7:0]	master_id	The ASNI node ID of the transaction causing an error.

IDM_ERRMISC1_NS

This register is the error log of Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

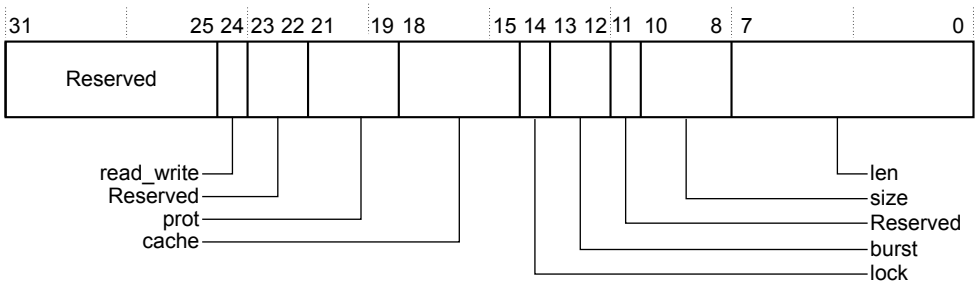

Figure 3-188 IDM_ERRMISC1_NS bit assignments

The following table shows the bit descriptions.

Table 3-198 IDM_ERRMISC1_NS descriptions

Bits	Name	Description
[31:25]	-	Reserved
[24]	read_write	Returns the AXI read or write information of a transaction causing an error: **1** Write **0** Read
[23:22]	-	Reserved
[21:19]	prot	Returns the AXI prot information of a transaction causing an error.
[18:15]	cache	Returns the AXI cache information of a transaction causing an error.
[14]	lock	Returns the AXI lock information of a transaction causing an error.
[13:12]	burst	Returns the AXI burst information of a transaction causing an error.
[11]	-	Reserved
[10:8]	size	Returns the AXI size information of a transaction causing an error.
[7:0]	len	Returns the AXI len information of a transaction causing an error.

IDM_ACCESS_STATUS_NS

This register indicates the access status for Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

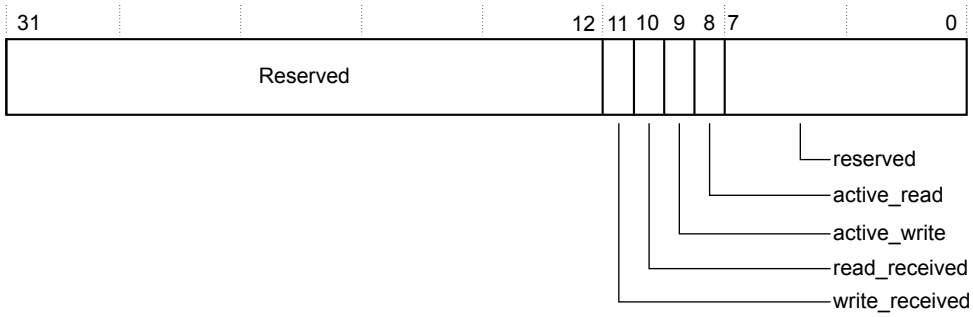

Figure 3-189 IDM_ACCESS_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-199 IDM_ACCESS_STATUS_NS descriptions

Bits	Name	Description
[31:12]	-	Reserved, UNDEFINED, write as zero
[11]	write_received	A 1 indicates that an active write transaction has occurred since the IDM entered the isolation state. This bit is cleared to zero on: • Reentry to isolation state. Write 1 into bit 0 of the IDM_ACCESS_CONTROL register when already in pending isolation entry state, or isolation active state. • Re-exit from isolation state. Write 1 into bit 0 of the IDM_ACCESS_CONTROL register when already in pending isolation exit state.
[10]	read_received	A 1 indicates that an active read transaction has occurred since the IDM entered the isolation state. This bit is cleared to zero on: • Reentry to isolation state. Write 1 into bit 0 of IDM_ACCESS_CONTROL register when already in pending isolation entry state, or isolation active state. • Re-exit from isolation state. Write 1 into bit 0 of IDM_ACCESS_CONTROL register when already in pending isolation exit state.
[9]	active_write	Active write transactions A 1 indicates there is at least one write transaction currently in progress.
[8]	active_read	Active read transactions A 1 indicates there is at least one read transaction currently in progress.
[7:0]	Reserved	Reserved, UNDEFINED, write as zero

IDM_ACCESS_READID

This register is the access log of Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

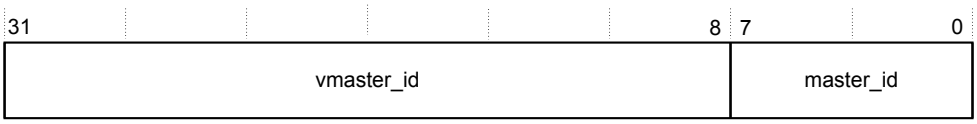

Figure 3-190 IDM_ACCESS_READID_NS bit assignments

The following table shows the bit descriptions.

Table 3-200 IDM_ACCESS_READID_NS bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **ARID** into the endpoint of the first transaction to arrive after isolation when the access_status active_read field is HIGH. The assumption is no manipulation of incoming AXI **ARID** in ASNI.
[7:0]	master_id	The ASNI node ID of the first transaction to arrive after isolation when the access_status active_read field is HIGH.

IDM_ACCESS_WRITEID_NS

This register is the access log of Non-secure transactions.

Usage constraints
None.

Configurations
If IDM is enabled, this register is implemented in NI-700.

Attributes
For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

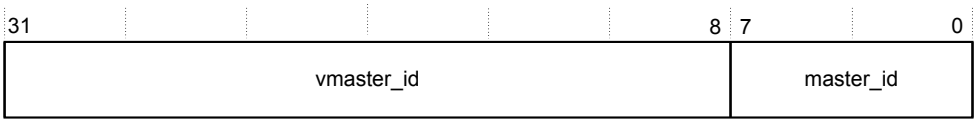

Figure 3-191 IDM_ACCESS_WRITEID_NS bit assignments

The following table shows the bit descriptions.

Table 3-201 IDM_ACCESS_WRITEID_NS bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **AWID** into endpoint of the first transaction to arrive after isolation when the access_status active_write field is HIGH. The assumption is there is no manipulation of incoming AXI **AWID** in ASNI.
[7:0]	master_id	The ASNI node ID of the first transaction to arrive after isolation when the access_status active_write field is HIGH.

IDM_RESET_STATUS_NS

This register indicates the reset status for Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

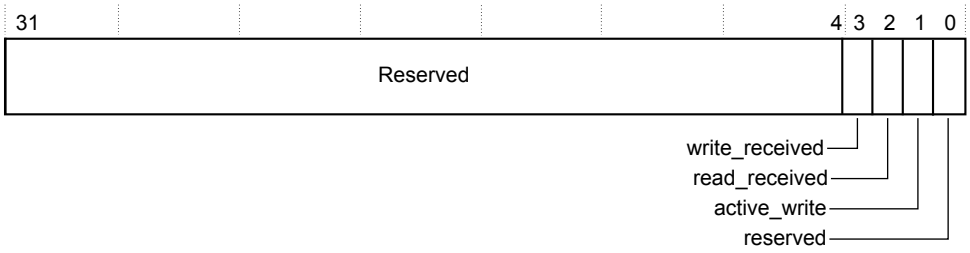

Figure 3-192 IDM_RESET_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-202 IDM_RESET_STATUS_NS descriptions

Bits	Name	Description
[31:4]	-	Reserved, UNDEFINED, write as zero
[3]	write_received	A 1 indicates that an active write transaction has occurred since the IDM entered the soft_reset state. This bit is cleared to zero on: • Reentry to soft reset state. Write 1 into bit 0 of IDM_RESET_CONTROL register when already in pending soft reset entry state, or soft reset active state. • Re-exit from soft reset state. Wite 0 into bit 0 of the IDM_RESET_CONTROL register when already in pending soft reset exit state.
[2]	read_received	A 1 indicates that there has been an active read transaction since a write of 1 to reset_control. This bit is cleared to 0 on: • Reentry to soft reset state. Write 1 into bit 0 of IDM_RESET_CONTROL register when already in pending soft reset entry state, or soft reset active state. • Re-exit from soft reset state. Write 0 into bit 0 of the IDM_RESET_CONTROL register when already in pending soft reset exit state.
[1]	active_write	Active write transactions A 1 indicates that there is at least one write transaction currently in progress.
[0]	Reserved	Active read transactions. A 1 indicates that there is at least one read transaction currently in progress.

IDM_RESET_READID_NS

This register is the reset access log of Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following table shows the bit descriptions.

Table 3-203 IDM_RESET_READID_NS bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **ARID** into the endpoint of the first transaction to arrive after isolation when the reset_status active_read field is HIGH. The assumption is no manipulation of incoming AXI **ARID** in ASNI.
[7:0]	master_id	This id is the ASNI node ID of the first transaction to arrive after isolation when the reset_status active_read field is HIGH.

IDM_RESET_WRITEID_NS

This register is the reset access log of Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit descriptions.

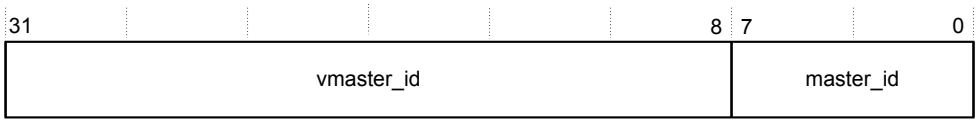

Figure 3-193 IDM_RESET_WRITEID_NS bit assignments

The following table shows the bit descriptions.

Table 3-204 IDM_RESET_WRITEID_NS bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI **AWID** into the endpoint of the first transaction to arrive after isolation when the reset_status active_write field is HIGH. The assumption is no manipulation of incoming AXI **AWID** in ASNI)
[7:0]	master_id	The ASNI node ID of the first transaction to arrive after isolation when the reset_status active_write field is HIGH.

IDM_INTERRUPT_STATUS_NS

This register indicates the interrupt status of Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

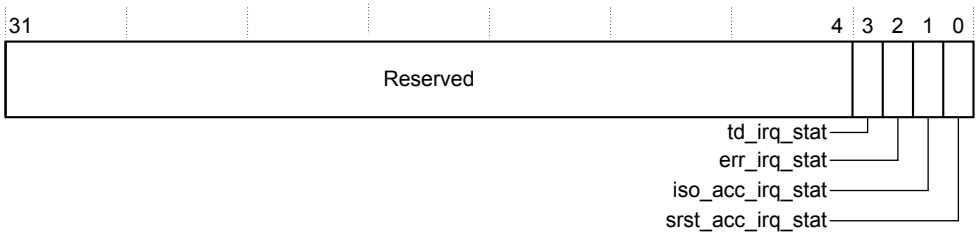

Figure 3-194 IDM_INTERRUPT_STATUS_NS bit assignments

The following table shows the bit descriptions.

Table 3-205 IDM_INTERRUPT_STATUS_NS bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3]	td_irq_stat	Timeout detection event Interface has detected a timeout.
[2]	err_irq_stat	Error detection event Interface has detected a protocol error.
[1]	iso_acc_irq_stat	Isolation access event Interface access while the IDM is closed.
[0]	srst_acc_irq_stat	Reset access event Interface access while the IDM is closed.

Note

A read of 1 for a field indicates that the associated interrupt event has been triggered. A write of 1 to a field in this register clears the associated interrupt event. The interrupt is asserted whenever the appropriate bits in this register are set to 1.

IDM_INTERRUPT_MASK_NS

This register is the interrupt mask of Non-secure transactions.

Usage constraints

None.

Configurations

If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see *3.11.1 Network Interface IDM registers summary* on page 3-251.

The following figure shows the bit assignments.

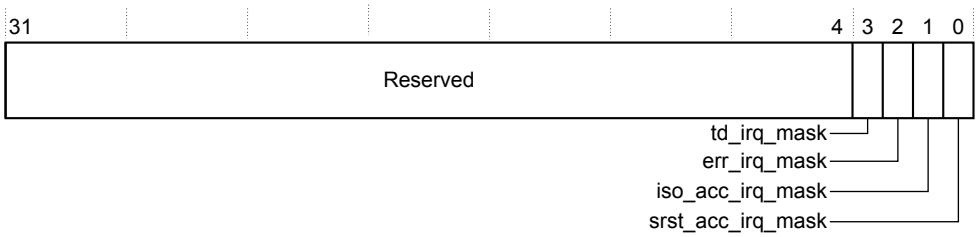

Figure 3-195 IDM_INTERRUPT_MASK_NS bit assignments

The following table shows the bit descriptions.

Table 3-206 IDM_INTERRUPT_MASK_NS bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3]	td_irq_mask	Timeout detection event mask
[2]	err_irq_mask	Error detection event mask
[1]	iso_acc_irq_mask	Access event mask
[0]	srst_acc_irq_mask	Access event mask

Note

A value of 1 indicates that the interrupt event is masked.

3.12 APB Master Network Interface registers

This section describes the NI-700 *APB Master Network Interface* (PMNI) registers. It contains a summary of the master interface registers, in order of address offset, and a description of the bitfields for each register.

This section contains the following subsections:

3.12.1 PMNI registers summary

This register summary lists the NI-700 PMNI registers and some key characteristics.

The following table shows the master interface registers in offset order. The base address of NI-700 is not fixed, and can be different for any particular system implementation. For more information, refer to your SoC implementation documentation. The offset of each register from the base address is fixed.

Table 3-207 PMNI registers summary

Offset	Name	Type	Reset	Width	Description
0x000	PMNI_NODE_TYPE	RO	0x0009	32	*PMNI_NODE_TYPE, Node type register for PMNI registers* on page 3-281
0x004	PMNI_NODE_INFO	RO	0x0000	9	*PMNI_NODE_INFO, Node information for PMNI register* on page 3-281
0x008	PMNI_SECR_ACC	RW	0x00	2	*PMNI_SECR_ACC, Secure access register* on page 3-282
0x00C	PMNI_PMUSELA	RW	0x0000	32	*PMNI_PMUSELA, Configure PMNI crossbar register* on page 3-283
0x010	PMNI_PMUSELB	RW	0x0000	32	*PMNI_PMUSELB, Configure PMNI crossbar register* on page 3-284
0x014	PMNI_INTERFACEID_0:3	RO	Configuration dependent	32	*PMNI_INTERFACEID, Configure APB interface IDs 0-3* on page 3-284
0x018	PMNI_INTERFACEID_4:7	RO		32	*PMNI_INTERFACEID, Configure APB interface IDs 4-7* on page 3-285
0x01C	PMNI_INTERFACEID_8:11	RO		32	*PMNI_INTERFACEID, Configure APB interface IDs 8-11* on page 3-286
0x020	PMNI_INTERFACEID_12:15	RO		32	*PMNI_INTERFACEID, Configure APB interface IDs 12-15* on page 3-286
0x030	PMNI_SECURE_INFO	RO		32	*PMNI_SECURE_INFO, security attribute of downstream APB interfaces register* on page 3-287
0x040	PMNI_NODE_FEAT	RAZ	0x0000	32	*PMNI_NODE_FEAT, Node features register* on page 3-288
0x044	PMNI_CTRL	RW	Configuration dependent	Configuration dependent	*PMNI_CTRL, PMNI control register* on page 3-289
0x080	PMNI_SILDBG	RW/RO	0x00	32	*PMNI_SILDBG, PMNI silicon debug monitor register* on page 3-290

3.12.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

PMNI_NODE_TYPE, Node type register for PMNI registers

This register identifies the node type as a node for PMNI registers.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

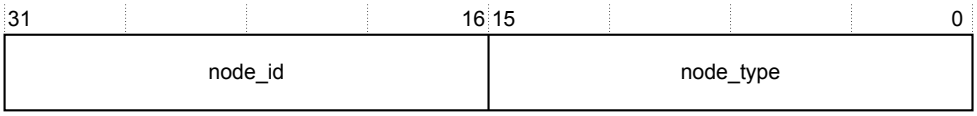

Figure 3-196 PMNI_NODE_TYPE bit assignments

The following table shows the bit descriptions.

Table 3-208 PMNI_NODE_TYPE bit descriptions

Bits	Name	Description
[31:16]	node_id	The PMNI ID that is assigned during network construction.
[15:0]	node_type	The value of this field is `0x0009`, and it identifies the associated node type as a node for NI-700 PMNI registers.

PMNI_NODE_INFO, Node information for PMNI register

This register provides node information for PMNI, such as data width.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

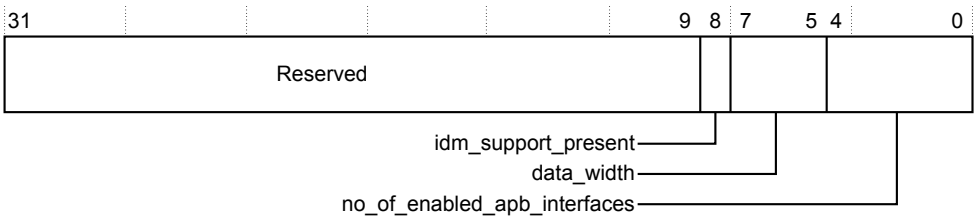

Figure 3-197 PMNI_NODE_INFO bit assignments

The following table shows the bit descriptions.

Table 3-209 PMNI_NODE_INFO bit descriptions

Bits	Name	Description
[31:9]	-	Reserved
[8]	idm_support_present	IDM support present **0** IDM support logic is not present. **1** IDM support logic is present.
[7:5]	data_width	Data width, HSIZE encoded: `0b000` Reserved `0b001` Reserved `0b010` 4 bytes `0b011` Reserved `0b100` Reserved `0b101` Reserved `0b110` Reserved `0b111` Reserved
[4:0]	no_of_enabled_apb_interfaces	The number of enabled APB interfaces at a specific PMNI. Permitted values are between 1 and 16.

PMNI_SECR_ACC, Secure access register

This register controls Secure access.

Usage constraints
Accessible using Secure transactions only.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

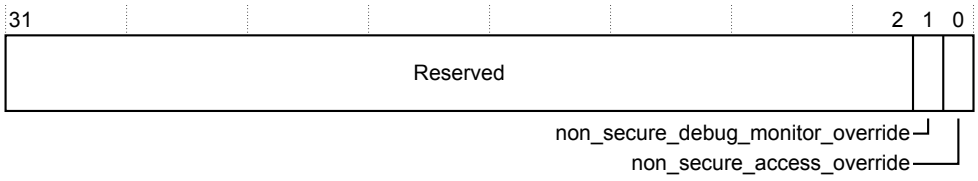

Figure 3-198 PMNI_SECR_ACC bit assignments

The following table shows the bit descriptions.

Table 3-210 PMNI_SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	non_secure_debug_monitor_override	Non-secure debug monitor override: **0** Disable. Non-secure access to the NI-700 PMU and interface registers. **1** Enable. Non-secure access to the NI-700 PMU and interface registers.
[0]	non_secure_access_override	Non-secure access override: **0** Disable. Non-secure access to the Secure NI-700 registers in this register region. **1** Enable. Non-secure access to the Secure NI-700 registers in this register region.

PMNI_PMUSELA, Configure PMNI crossbar register

This register is used to select the event values in the PMNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *PMNI_SECR_ACC, Secure access register* on page 3-282 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

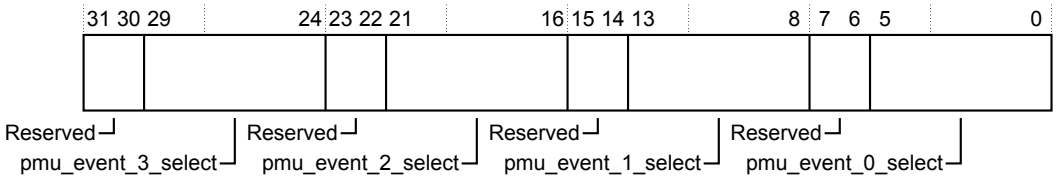

Figure 3-199 PMNI_PMUSELA bit assignments

The following table shows the bit descriptions.

Table 3-211 PMNI_PMUSELA bit descriptions

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_3_select	PMU event 3 select
[23:22]	-	Reserved
[21:16]	pmu_event_2_select	PMU event 2 select
[15:14]	-	Reserved
[13:8]	pmu_event_1_select	PMU event 1 select
[7:6]	-	Reserved
[5:0]	pmu_event_0_select	PMU event 0 select

PMNI_PMUSELB, Configure PMNI crossbar register

This register is used to select the event values in the PMNI event crossbar.

Usage constraints

Accessible using only Secure accesses, unless you set the *ASNI_SECR_ACC, Secure access register* on page 3-174 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access the register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

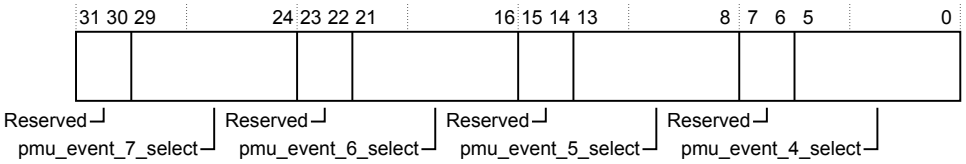

Figure 3-200 PMNI_PMUSELB bit assignments

The following table shows the bit assignments.

Table 3-212 PMNI_PMUSELB bit assignments

Bits	Name	Description
[31:30]	-	Reserved
[29:24]	pmu_event_7_select	PMU event 7 select
[23:22]	-	Reserved
[21:16]	pmu_event_6_select	PMU event 6 select
[15:14]	-	Reserved
[13:8]	pmu_event_5_select	PMU event 5 select
[7:6]	-	Reserved
[5:0]	pmu_event_4_select	PMU event 4 select

PMNI_INTERFACEID, Configure APB interface IDs 0-3

To configure APB interface IDs 0-3, use offset `0x014` in the PMNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

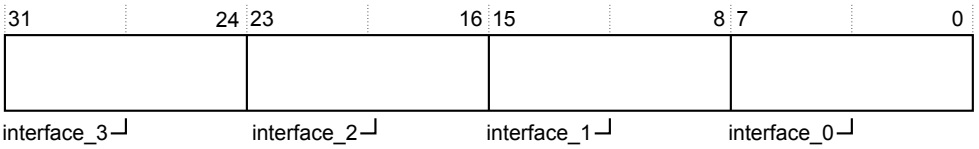

Figure 3-201 PMNI_INTERFACEID bit assignments, APB interface IDs 0-3

The following table shows the bit descriptions.

Table 3-213 PMNI_INTERFACEID descriptions, APB interface IDs 0-3

Bits	Name	Description
[31:24]	interface_3	APB interface ID 3
[23:16]	interface_2	APB interface ID 2
[15:8]	interface_1	APB interface ID 1
[7:0]	interface_0	APB interface ID 0

PMNI_INTERFACEID, Configure APB interface IDs 4-7

To configure APB interface IDs 4-7, use offset `0x018` in the PMNI_INTERFACEID register.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

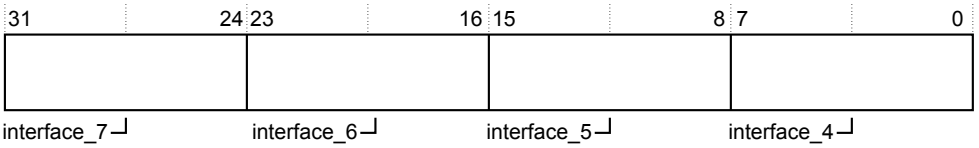

Figure 3-202 PMNI_INTERFACEID bit assignments, APB interface IDs 4-7

The following table shows the bit descriptions.

Table 3-214 PMNI_INTERFACEID descriptions, APB interface IDs 4-7

Bits	Name	Description
[31:24]	interface_7	APB interface ID 7
[23:16]	interface_6	APB interface ID 6
[15:8]	interface_5	APB interface ID 5
[7:0]	interface_4	APB interface ID 4

PMNI_INTERFACEID, Configure APB interface IDs 8-11

To configure APB interface IDs 8-11, use offset `0x01C` in the PMNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

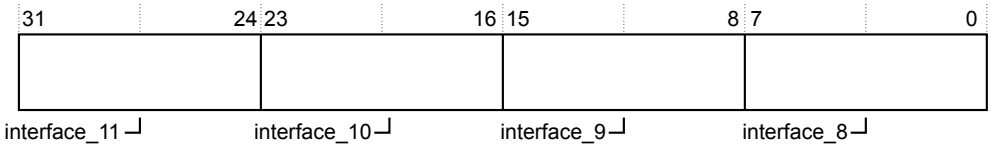

Figure 3-203 PMNI_INTERFACEID bit assignments, APB interface IDs 8-11

The following table shows the bit descriptions.

Table 3-215 PMNI_INTERFACEID descriptions, APB Interface IDs 8-11

Bits	Name	Description
[31:24]	interface_11	APB interface ID 11
[23:16]	interface_10	APB interface ID 10
[15:8]	interface_9	APB interface ID 9
[7:0]	interface_8	APB interface ID 8

PMNI_INTERFACEID, Configure APB interface IDs 12-15

To configure APB interface IDs 12-15, use offset `0x020` in the PMNI_INTERFACEID register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

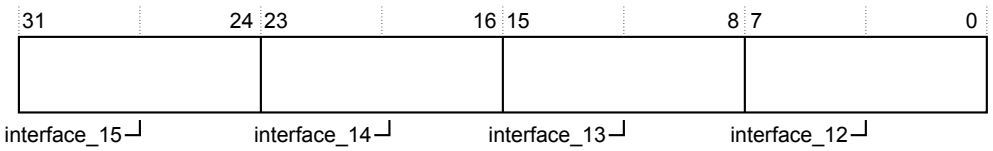

Figure 3-204 PMNI_INTERFACEID bit assignments, APB interface IDs 12-15

The following table shows the bit descriptions.

Table 3-216 PMNI_INTERFACEID descriptions, APB Interface IDs 12-15

Bits	Name	Description
[31:24]	interface_15	APB interface ID 15
[23:16]	interface_14	APB interface ID 14
[15:8]	interface_13	APB interface ID 13
[7:0]	interface_12	APB interface ID 12

PMNI_SECURE_INFO, security attribute of downstream APB interfaces register

To view the security attribute for each of the APB interfaces downstream of the PMNI, use the PMNI_SECURE_INFO register.

Usage constraints

None.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

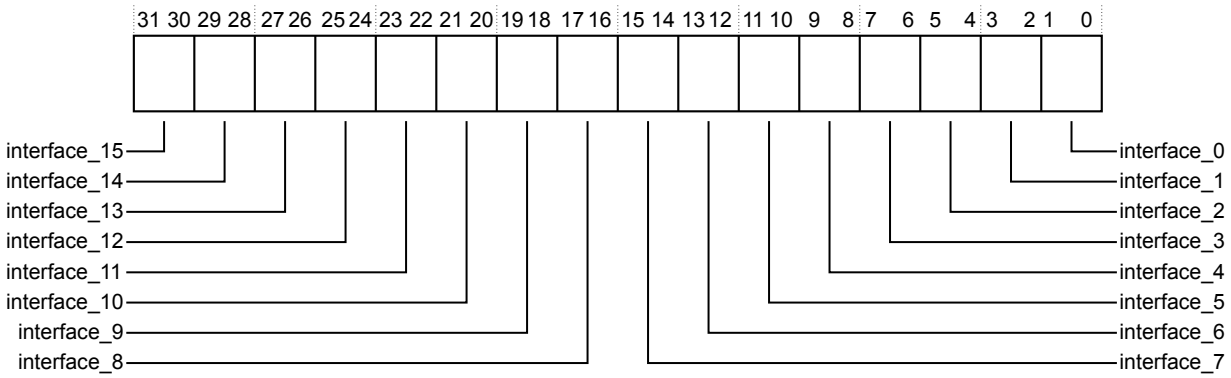

Figure 3-205 PMNI_SECURE_INFO

The following table shows the bit descriptions.

Table 3-217 PMNI_SECURE_INFO

Bits	Name	Description
[31:30]	interface_15	Security attribute for interface 15
[29:28]	interface_14	Security attribute for interface 14
[27:26]	interface_13	Security attribute for interface 13
[25:24]	interface_12	Security attribute for interface 12.
[23:22]	interface_11	Security attribute for interface 11
[21:20]	interface_10	Security attribute for interface 10
[19:18]	interface_9	Security attribute for interface 9
[17:16]	interface_8	Security attribute for interface 8

Table 3-217 PMNI_SECURE_INFO (continued)

Bits	Name	Description
[15:14]	interface_7	Security attribute for interface 7
[13:12]	interface_6	Security attribute for interface 6
[11:10]	interface_5	Security attribute for interface 5
[9:8]	interface_4	Security attribute for interface 4
[7:6]	interface_3	Security attribute for interface 3
[5:4]	interface_2	Security attribute for interface 2
[3:2]	interface_1	Security attribute for interface 1
[1:0]	interface_0	Security attribute for interface 0 `0b00` Software-programmable register to set the security attribute for the downstream slave. `0b01` Pin exists and is used to pass the security attribute. Downstream filters out based on **PPROT[1]**. `0b02` Always Secure. Only Secure transactions access the slave attached to this APB master interface. `0b03` Always Non-secure. Both Secure and Non-secure transactions access the slave attached to this APB master interface.

PMNI_NODE_FEAT, Node features register

This register configures the node features. You can configure up to 16 APB interfaces for a PMNI. Use 2 bits to identify the APB protocol for a specific interface.

Usage constraints
None.

Configurations
Available in all NI-700 configurations.

Attributes
For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

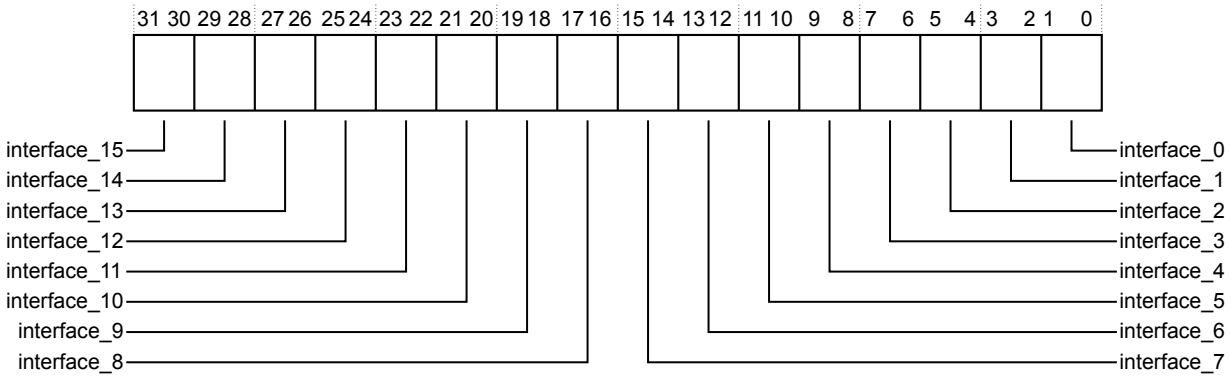

Figure 3-206 PMNI_NODE_FEAT bit assignments

The following table shows the bit descriptions.

Table 3-218 PMNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:30]	interface_15	Interface 15 APB protocol type
[29:28]	interface_14	Interface 14 APB protocol type
[27:26]	interface_13	Interface 13 APB protocol type
[25:24]	interface_12	Interface 12 APB protocol type
[23:22]	interface_11	Interface 11 APB protocol type
[21:20]	interface_10	Interface 10 APB protocol type
[19:18]	interface_9	Interface 9 APB protocol type
[17:16]	interface_8	Interface 8 APB protocol type
[15:14]	interface_7	Interface 7 APB protocol type
[13:12]	interface_6	Interface 6 APB protocol type
[11:10]	interface_5	Interface 5 APB protocol type
[9:8]	interface_4	Interface 4 APB protocol type
[7:6]	interface_3	Interface 3 APB protocol type
[5:4]	interface_2	Interface 2 APB protocol type
[3:2]	interface_1	Interface 1 APB protocol type
[1]	interface_0	Interface 0 APB protocol type The encoding is common across all the interfaces: **0b00** Reserved **0b01** APB3 **0b10** APB4 **0b11** Reserved

PMNI_CTRL, PMNI control register

This register indicates the security status, Secure or Non-secure, of APB interfaces that are attached to a PMNI.

Usage constraints

Accessible using only Secure accesses, unless you set the *PMNI_SECR_ACC, Secure access register* on page 3-282 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

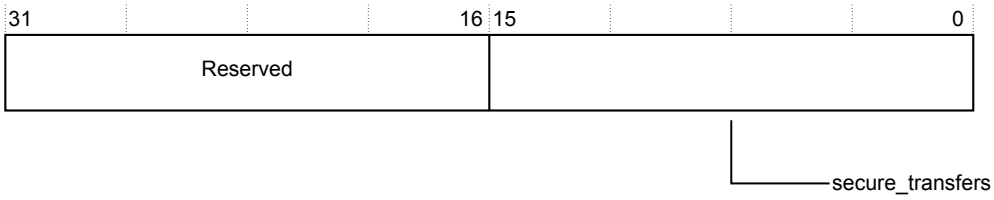

Figure 3-207 PMNI_CTL bit assignments

The following table shows the bit descriptions.

Table 3-219 PMNI_CTRL bit descriptions

Bits	Name	Description
[31:16]	-	Reserved
[15:0]	secure_transfers	The width depends on the number of APB ports, up to 16 ports. A single bit is assigned for each port to indicate the security status, either Secure or Non-secure, of the downstream slave. **0** Secure. Only Secure transactions can travel downstream. **1** Non-secure. Both Secure and Non-secure transactions can travel downstream. This register bit is relevant based on the secure_transfers field in the *PMNI_SECURE_INFO, security attribute of downstream APB interfaces register* on page 3-287. `0b00` If secure_transfers is 00, the **PPROT** pin is unavailable. This register bit determines the security attribute of the downstream slave. The security access permission check occurs within the PMNI. `0b01` If secure_transfers = 01, the **PPROT** pin is supported downstream of the PMNI. The incoming security attribute is passed on to the pin, therefore this register bit is irrelevant. If the incoming request is Non-secure, and the downstream slave is configured as Secure, then the transaction is not sent downstream. A Non-secure read transaction returns zero data. The data corresponding to a Non-secure write transaction is dropped but a protocol- compliant write response is returned. The read or write response does not contain an error indication. **`0b02` or `0b03`** If secure_transfers = 02 or secure_transfers = 03, then the **PPROT** pin is unavailable. However the APB interface security attribute is fixed at build time to either Always Secure or Always Non-secure. This register bit becomes read-only. However if secure_transfers = 03, the reset value is 1 and if secure_transfers = 02, the reset value is 0.

PMNI_SILDBG, PMNI silicon debug monitor register

This register monitors the status of NI-700 master interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the *PMNI_SECR_ACC, Secure access register* on page 3-282 to permit Non-secure accesses. Setting either bit [0] or bit [1] of the Secure access register permits Non-secure accesses to access this register.

Configurations

Available in all NI-700 configurations.

Attributes

For more information, see *3.12.1 PMNI registers summary* on page 3-280.

The following figure shows the bit assignments.

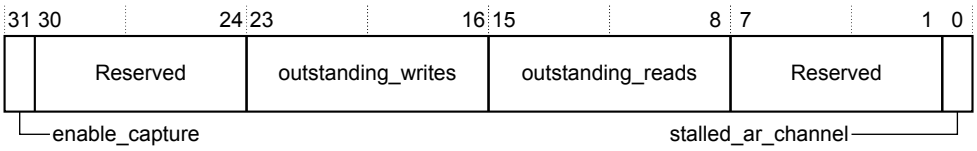

Figure 3-208 PMNI_SILDBG bit assignments

The following table shows the bit descriptions.

Table 3-220 PMNI_SILDBG bit descriptions

Bits	Name	Description
[31]	enable_capture	Enable capture
[30:24]	-	Reserved
[23:16]	outstanding_writes	Indicates that the interface has writes that are outstanding.
[15:8]	outstanding_reads	Outstanding read
[7:1]	-	Reserved
[0]	stalled_ar_channel	Indicates stalled read request

Chapter 4
Performance monitoring

This chapter describes the Performance Monitoring Unit (PMU), which enables system integrators to monitor events to optimize the design of the system.

It contains the following sections:

4.1 PMU and debug

This section describes the PMU and debug functionality of NI-700.

This section contains the following subsections:

4.1.1 PMU organization

The PMU is distributed across each clock domain with software visible event counters in each clock domain.

Within each clock domain, events are generated from several potential sources and multiplexed onto internal 8-bit event busses. These internal buses are in turn routed to the central set of software visible PMU counters for that clock domain. Each performance event counter has a corresponding set of shadow snapshot registers to permit all counters to be sampled simultaneously and then read out in series.

The following figure shows the two-level hierarchical organization of the PMU. The Configuration registers comprise:

1. Software visible event PMU counters
2. Snapshot registers and other PMU control registers
3. A configuration register for event selection in the event crossbar

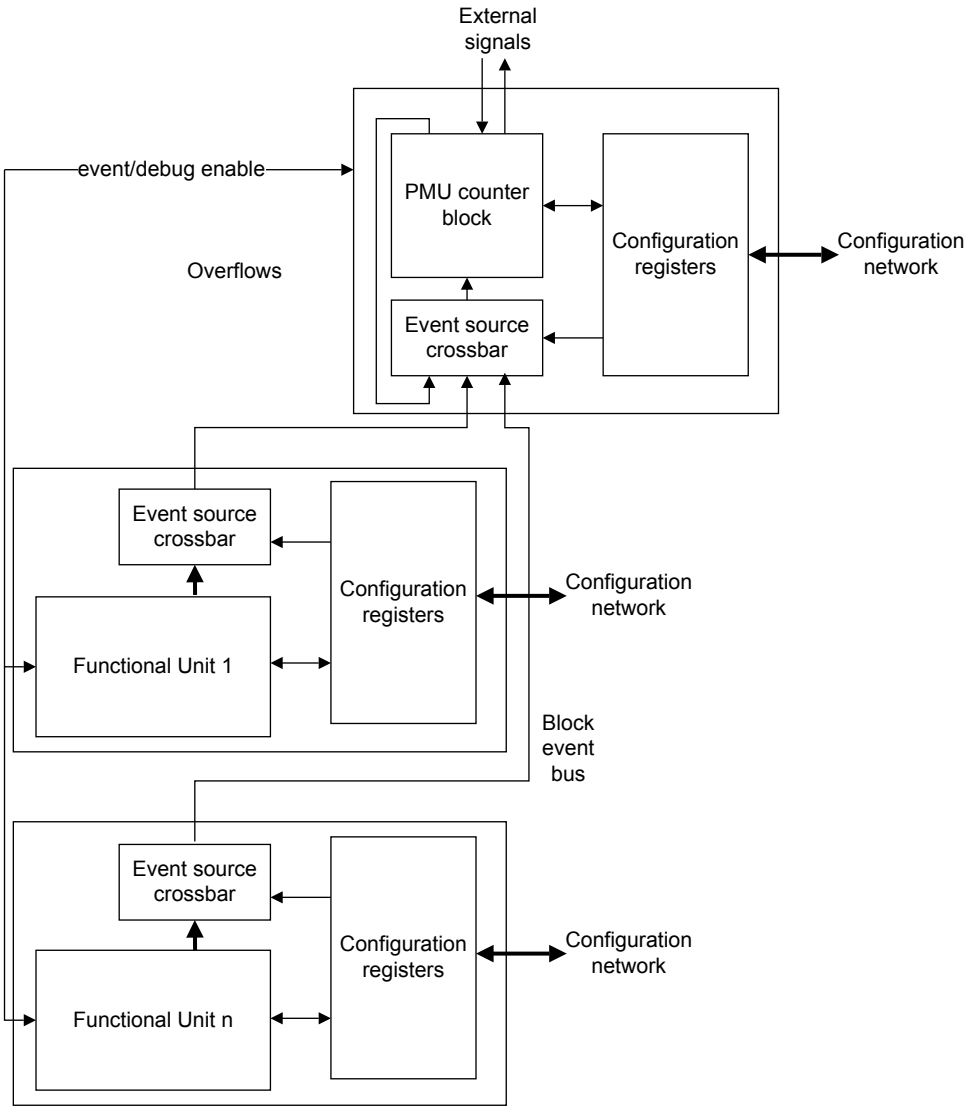

Figure 4-1 PMU hierarchical organization

The first level of the hierarchy is at the level of the functional unit. Each functional unit, such as an individual ASNI or AMNI, can define up to 64 events. The PMU events are configured by programming the PMUSELA and PMUSELB registers in the node. See *3.8 AXI Master Network Interface registers* on page 3-199 and *3.7 AXI Slave Network Interface registers* on page 3-170.

Event/Debug enable signals start the generation of events for a unit. An event source crossbar is configured through the programming interface to reduce the number of possible events to a maximum of eight events minimizing top-level wiring. By programming two PMU event select registers, up to eight events can be selected for publishing on an internal 8-bit event bus from each unit in that clock domain.

The individual event buses are routed to a centralized PMU counter block per clock domain that has the second level of the PMU event selection logic. It consists of:

- A bank of eight 32-bit counters with overflow and snapshot functionality. These counters are responsible for counting the programmed events and giving memory-mapped read access to both the counters and counter snapshots.
- A programmable event source crossbar to permit selection of a particular event for a counter to monitor.

  - Each of the 8 bits of the internal event bus from each unit is routed to one of the eight counters with the matching index. For example, bit 0 to counter 0, bit 1 to counter 1).
  - The second level PMU event source crossbar supports PMU event type and filter registers. See *PMEVTYPERn, Performance monitor event type and filter registers* on page 3-157. These registers provide a programming interface that permits software to specify which unit event bus input each counter selects, according to type and source index.
- The event source crossbar can configure the PMU counters to trigger from the overflow of another counter within the PMU block. This feature permits extension of the counter range. For example, the crossbar can extend a single event counter up to a maximum 256-bit range with a single overflow.

4.1.2 PMU system programming

This section contains procedures for programming the PMU event counters, snapshot functionality, and interrupts.

For specific register descriptions, see the *Chapter 3 Programmers model* on page 3-115.

Setting up the PMU counters

This section describes the procedure for setting up the PMU event counters.

Note

For PMU operation, **NIDEN** input must be asserted.

Procedure

1. Program the *_PMUSELA/*_PMUSELB registers in the individual endpoints (for example, ASNI and AMNI) to select the events that are published on the internal 8-bit event bus.
2. Program the eight PMEVTYPERn registers in the PMU block in every clock domain to program the PMU event source crossbar.
3. Write to the PMCNTENSET/PMCNTENCLR registers to enable specific PMU event counters.
4. Write to the PMINTENSET/PMINTENCLR registers to enable interrupts for the corresponding specific PMU event counters.
5. Use the PMCNTENSET register to reset cycle/event counters, to write to the PMCR register, and to enable PMU counting.

   This action enables all counters, whereas the PMCNTENSET/PMCNTENCLR enables specific counters.

Programming PMU snapshot functionality

This section describes the procedure for programming PMU snapshot functionality.

A snapshot of PMU event counters can be triggered in the following ways:
- By setting the control bits in the PMSSCR register
- By using a four-phase handshake on the signals that *Table 4-1 PMU snapshot signals* on page 4-295 shows

Table 4-1 PMU snapshot signals

Signal	Direction	Description	Clock relationship
<CLKNAME>_PMUSNAPSHOTREQ	Input	Four-phase request to initiate snapshot of PMU event counters	Asynchronous
<CLKNAME>_PMUSNAPSHOTACK	Output	Acknowledgement of PMU snapshot capture	Asynchronous

Note

For PMU operation, **NIDEN** input must be asserted.

Procedure

1. Program the PMU event counters. See *Setting up the PMU counters* on page 4-295
2. Write `0b1` to the PMSSCR register to capture a snapshot of the contents of the PMU event counters, cycle counter, and overflow status

After the PMU snapshot process has completed, the PMU block updates the PMSSR, PMOVSSR, PMCCNTSR (lower, upper), and PMEVCNTSRn registers. Software can poll the PMSSR register to check that the snapshot has been done.

Programming PMU interrupts

This section describes the procedure for programming PMU interrupts.

If an event or cycle counter overflows, an interrupt is triggered. This interrupt is connected to the top-level interrupt, **<CLKNAME>_nPMUINTERRUPT**. You can determine the counter that has overflowed from the PMU control and configuration registers. These registers can also clear any counter overflow flags so that the interrupt can be cleared.

Procedure

1. For PMU operation, **NIDEN** input has to be asserted.
2. Program the PMU counters. See *Setting up the PMU counters* on page 4-295.

   Any PMU counter overflow asserts **<CLKNAME>_nPMUINTERRUPT**. When observing assertion of **<CLKNAME>_nPMUINTERRUPT**, the PMOVSSR and PMOVSCLR registers should be polled to determine the event counter or cycle counter that caused the interrupt.
3. Write `0b1` into the corresponding PMOVSCLR register to clear **<CLKNAME>_nPMUINTERRUPT**.

Performance monitoring and Secure debug

If Non-secure event triggering is enabled, the Secure event enables named **SPIDEN** and **SPNIDEN** enable the counting and export of both Non-secure and Secure events.

Some events are counted irrespective of the **SPNIDEN** input and these events are shown as Secure exempt in the PMU event list. For the event lists, see *Chapter 4 Performance monitoring* on page 4-292.

Table 4-2 PMU and debug signals

Signal	Direction	Description	Clock relationship
<CLKNAME>_NIDEN	Input	Non-invasive debug enable. If HIGH, the signal enables counting and export of PMU events.	Synchronous
<CLKNAME>_SPNIDEN	Input	Secure privileged non-invasive debug enable. When HIGH, this signal enables the counting of both Non-secure and Secure events, provided **NIDEN** is also HIGH.	Synchronous
<CLKNAME>_DBGEN	Input	Invasive debug enable. If HIGH, enables the counting and export of PMU events.	Synchronous
<CLKNAME>_SPIDEN	Input	Secure privileged invasive debug enable. When HIGH, this signal enables the counting of both Non-secure and Secure events, provided **DBGEN** is also HIGH.	Synchronous

The counting and export of events that are triggered by Non-secure events are enabled by the **DBGEN** and **NIDEN** inputs: Debug enable = **DBGEN** | **NIDEN**.

The full expression for counting Secure and Non-secure events is:

Secure debug = ((**SPIDEN** & **DBGEN**) | **SPNIDEN**) & (**DBGEN** | **NIDEN**).

4.2 AXI Master Network Interface performance events

The NI-700 AMNI can generate various performance events.

The following table shows the performance events that the AMNI can track.

——— **Note** ———

SPNIDEN determines whether Secure events are counted or not. However, some events such as Read Data do not have the Secure or Non-secure attribute. Therefore, these events are marked as Secure only. You can only count the events if you enable them.

———————————

Table 4-3 AMNI performance events

Event code [5:0]	Event	Secure only
`0x00`	Read request: any (**ARVALID** & **ARREADY**)	N
`0x01`	Read request: device	N
`0x02`	Transaction Group “Non-snooping”, Read Request: ReadNoSnoop (RNS)	N
`0x03`	Transaction Group “Coherent”, I/O coherent Read Request: ReadOnce (RO)	N
`0x04`	Transaction Group “Cache Maintenance” Requests: CleanShared, CleanInvalid, MakeInvalid, CleanSharedPersist ——— **Note** ——— CleanSharedPersist is only present in ACE5-Lite. ———————————	N
`0x05`	Read data beat: any (**RVALID** & **RREADY**)	Y[a]
`0x06`	Read data handshake with **RLAST** set	Y[a]
`0x07`	Write request: any (**AWVALID** & **AWREADY**)	N
`0x08`	Write request: device	N
`0x09`	Transaction Group “Non-snooping”, Write Request: WriteNoSnoop (WNS)	N
`0x0A`	Transaction Group “Coherent”, I/O coherent Write Request: WriteLineUnique (WLU)	N
`0x0B`	Transaction Group “Coherent”, I/O coherent Write Request: WriteUnique (WU)	N
`0x0C`	Write request: Atomic (Store, Load, Swap, Compare)	N
`0x0D`	Write data beat: any (**WVALID** & **WREADY**)	Y[a]
`0x0E`	Read request stall: **ARVALID** HIGH, **ARREADY** LOW	N
`0x0F`	Read data stall: **RVALID** HIGH, **RREADY** LOW	Y[a]
`0x10`	Write request stall: **AWVALID** HIGH, **AWREADY** LOW	N
`0x11`	Write data stall: **WVALID** HIGH, **WREADY** LOW	Y[a]
`0x12`	Write response stall: **BVALID** HIGH, **BREADY** LOW	Y[a]
`0x13`	Write request: Cache Stash transactions	N
`0x14`	Write Channel: CMOs, Combined Write+CMOs (non-persistence type) ((**AWSNOOP** == `0b0110`) || (**AWSNOOP** == `0b1010`) || (**AWSNOOP** == `0b1011`)) && (**AWCMO** == non persist encodings)	N

Table 4-3 AMNI performance events (continued)

Event code [5:0]	Event	Secure only
0x15	Write Channel: CMOs, Combined Write+CMOs (persist and deep persist type) ((**AWSNOOP** == 0b0110) || (**AWSNOOP** == 0b1010) || (**AWSNOOP** == 0b1011)) && (**AWCMO** == persist and deep persist encodings)	N
0x16	Read requests with nonzero memory tagging operation	N
0x17	Write requests with nonzero memory tagging operation	N
0x20	Request stall because of read tracker occupancy	N
0x21	Request stall because of write tracker occupancy	N
0x22	Write channel B response stall because of a lack of GT credit	Y[a]
0x23	Read channel read response stall because of a lack of GT credit	Y[a]
0x24	Low wire mode arbitration stall on B channel	Y[a]
0x25	Low wire mode arbitration stall on R channel	Y[a]

[a] For this event, the request security attribute (Secure or Non-secure) is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.

4.3 AXI Slave Network Interface performance events

The NI-700 ASNI can generate various performance events.

The following table shows the performance events that the ASNI can track.

——— **Note** ———

SPNIDEN determines whether Secure events are counted or not. However, some events such as Read Data do not have the Secure or Non-secure attribute. Therefore, these events are marked as Secure Exempt. They do not expose any Secure information but only the number of such events.

———————————

Table 4-4 ASNI performance events

Event code [5:0]	Event	Secure only
0x00	Read request: any (**ARVALID** & **ARREADY**)	N
0x01	Read request: device **ARCACHE[3:1]** == 0b000	N
0x02	Transaction Group “Non-snooping”, Read Request: ReadNoSnoop (RNS)	N
0x03	Transaction Group “Coherent”, I/O coherent Read Request: ReadOnce (RO)	N
0x04	Transaction Group “Cache Maintenance” Requests: CleanShared, CleanInvalid, MakeInvalid, CleanSharedPersist ——— **Note** ——— CleanSharedPersist is only present in ACE5-Lite. ———————————	N
0x05	Read data beat: any (**RVALID** & **RREADY**)	Y[b]
0x06	Read data handshake with **RLAST** set	Y[b]
0x07	Write request: any (**AWVALID** & **AWREADY**)	N
0x08	Write request: device	N
0x09	Transaction Group “Non-snooping”, Write Request: WriteNoSnoop (WNS)	N
0x0A	Transaction Group “Coherent”, I/O coherent Write Request: WriteLineUnique (WLU)	N
0x0B	Transaction Group “Coherent”, I/O coherent Write Request: WriteUnique (WU)	N
0x0C	Write request: Atomic (Store, Load, Swap, Compare)	N
0x0D	Write data beat: any (**WVALID** & **WREADY**)	Y[b]
0x0E	Read request stall: **ARVALID** HIGH, **ARREADY** LOW	N
0x0F	Read data stall: **RVALID** HIGH, **RREADY** LOW	Y[b]
0x10	Write request stall: **AWVALID** HIGH, **AWREADY** LOW	N
0x11	Write data stall: **WVALID** HIGH, **WREADY** LOW	Y[b]
0x12	Write response stall: **BVALID** HIGH, **BREADY** LOW	Y[b]
0x13	Write request: Cache Stash transactions	N
0x14	Write Channel: CMOs, Combined Write+CMOs (non-persistence type) ((**AWSNOOP** == 0b0110) || (**AWSNOOP** == 0b1010) || (**AWSNOOP** == 0b1011)) && (**AWCMO** == non persist encodings)	N

Table 4-4 ASNI performance events (continued)

Event code [5:0]	Event	Secure only
0x15	Write Channel: CMOs, Combined Write+CMOs (persist and deep persist types) ((**AWSNOOP** == 0b0110) || (**AWSNOOP** == 0b1010) || (**AWSNOOP** == 0b1011)) && (**AWCMO** == persist and deep persist encodings)	N
0x16	Read requests with nonzero memory tagging operation	N
0x17	Write requests with nonzero memory tagging operation	N
0x20	Request stall cycle because of the OT transaction limit	N
0x21	Request stall cycle because of the hard bandwidth (TSPEC) regulation limit	N
0x22	Request stall because of the arbitration caused by collision of read and write request onto shared resources for atomics	N
0x23	Request stall because of read tracker occupancy	N
0x24	Request stall because of write tracker occupancy	N
0x25	AW stall becasue **WDATA** FIFO is full	Y[b]
0x26	AR stall because reorder buffer is full	N
0x27	AW CDAS stall	N
0x28	AR CDAS stall	N
0x29	Atomic RD stall because read resource is unavailable	N
0x2A	Write channel write request stall because of a lack of GT credit	Y[b]
0x2B	Read channel read request stall because of a lack of GT credit	Y[b]
0x2C	AW stall because of AW or combined OT regulation	N
0x2D	AR stall because of AR or combined OT regulation	N
0x2E	AW stall because of AW or combined TSPEC regulation.	N
0x2F	AR stall because of AR or combined TSPEC regulation	N
0x30	Low wire mode arbitration stall on W channel	Y[b]
0x31	Low wire mode arbitration stall on R channel	Y[b]

[b] For this event, the request security attribute (Secure or Non-secure) is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.

4.4 Data bandwidth at ASNI and AMNI

External AXI and ACE-Lite devices connect to the interconnect at ASNIs and AMNIs.

This section contains the following subsections:

4.4.1 Read and Write bandwidth (ASNI and AMNI)

NI-700 provides performance monitoring events to track the number of read and write data beats being transferred. These values can be used to calculate the total read and write bandwidth in the interconnect.

The following table shows the events that measure the number of read and write data beats.

Table 4-5 Read and Write data beat tracking events

Event code [5:0]	Description
`0x05`	Read data beat: Any (**RVALID** & **RREADY**)
`0x0D`	Write data beat: Any (**WVALID** & **WREADY**)

Calculate the read and write bandwidth according to the following calculations:

- Read bandwidth = ((Number Read Data beats × AXIDataBeatSize) / Cycles) × Frequency
- Write bandwidth = ((Number Write Data beats × AXIDataBeatSize) / Cycles) × Frequency

Note

AXIDataBeatSize is the number of bytes for each AXI beat. Usually, this number is the same size as **AxSIZE**.

4.4.2 Delays at ASNI and AMNI because of backpressure

To analyze the delays in ASNI and AMNI, NI-700 enables you to monitor the source of backpressure.

The following table shows the events that monitor such backpressure:

Table 4-6 Backpressure monitoring events

Event code [5:0]	Description
`0x0E`	Read request stall: **ARVALID** HIGH, **ARREADY** LOW
`0x0F`	Read data stall: **RVALID** HIGH, **RREADY** LOW
`0x10`	Write request stall: **AWVALID** HIGH, **AWREADY** LOW
`0x11`	Write data stall: **WVALID** HIGH, **WREADY** LOW
`0x12`	Write response stall: **BVALID** HIGH, **BREADY** LOW
`0x2A` (ASNI) / `0x22` (AMNI)	Write request stall because of a lack of GT credit
`0x2B` (ASNI) / `0x23` (AMNI)	Read request stall because of a lack of GT credit

4.4.3 Delays at ASNI because of structural backpressure

To analyze the delays in ASNI specifically, NI-700 enables you to monitor the source of backpressure because of structure full or other AXI ordering conditions.

The following table shows events that monitor such backpressure.

Table 4-7 Structural backpressure monitoring events

Event code [5:0]	Description
0x23	AR stall because of read tracker occupancy
0x24	AW stall because of write tracker occupancy
0x25	W stall because **WDATA** FIFO is full
0x26	AR stall because of reorder buffer full
0x27	AW CDAS stall
0x28	AR CDAS stall
0x29	Atomic RD stall because of read resource unavailable

4.5 AHB performance event mapping

NI-700 AHB performance events are mapped to AHB memory types.

The AHB PMU events are based on the memory types that are shown in the following table reproduced from the *Arm® AMBA® 5 AHB Protocol Specification, AHB5, AHB-Lite*.

Table 4-8 AHB memory types

HPROT[6] Shareable	HPROT[5] Allocate	HPROT[4] Lookup	HPROT[3] Modifiable	HPROT[2] Bufferable	Memory type
0	0	0	0	0	Device-nE
0	0	0	0	1	Device-E
0	0	0	1	0	Normal Non-cacheable, Non-shareable
0	0 or 1	1	1	0	Write-Through, Non-shareable
0	0 or 1	1	1	1	Write back, Non-shareable
1	0	0	1	0	Normal Non-cacheable, Shareable
0	0 or 1	1	1	0	Write-Through, Shareable
0	0 or 1	1	1	1	Write back, Shareable

4.6 AHB Slave Network Interface performance events

The NI-700 HSNI can generate various performance events.

The following table shows the performance events that the HSNI can track.

Table 4-9 HSNI performance events

Event code [4:0]	Event	Secure only
0x00	Read request: any	N
0x01	Read request: Device Device-nE and Device-E	N
0x02	Read request: 1. Normal Non-cacheable, Non-shareable. 2. Write-Through, Non-shareable. 3. Write-Back, Non-shareable.	N
0x03	Read request: Normal, Non-cacheable, Shareable.	N
0x04	Read request: 1. Write-Through, Shareable 2. Write-Back, Shareable	N/A
0x05	Read data beat: any	Y **Note** For this event, the request security attribute Secure or Non-secure is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.
0x06	N/A	Y
0x07	Write request: any	N
0x08	Write request: device (Device-nE and Device-E)	N
0x09	Write request: Normal, Non-cacheable, Non-shareable.	N
0x0A	Write request: Write-Through or Write-Back, Shareable, Non-shareable	N
0x0B	Write request: Normal, Non-cacheable, Shareable	N
0x0C	Write request: 1. Write-Through, Shareable 2. Write-Back, Shareable	N
0x0D	Write data beat: any	Y **Note** For this event, the request security attribute Secure or Non-secure is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.

Table 4-9 HSNI performance events (continued)

Event code [4:0]	Event	Secure only
0x0E	Read address phase stall. Not implemented in the HSNI, tied to 0.	N/A
0x0F	Read data phase stall. Prior read address phase, **HREADY** LOW.	Y
0x10	Write address phase stall. Not implemented in the HSNI, tied to 0.	N/A
0x11	Write data phase stall. Prior write address phase, **HREADY** LOW.	Y
0x12	Reserved	N/A
0x13	N/A	N
0x20	Request stall cycle because of OT transaction limit	N
0x21	Request stall cycle because of Hard BW (TSPEC) regulation limit	N
0x22	Read request stall because of early write responses: Early write response needs read hazarding until all the write responses have returned on GT. This condition leads to stalling of read request.	N
0x23	N/A	N
0x24	Request stall because of nonzero outstanding write counter	N
0x25	W stall because WDATA FIFO is full. HSNI uses the WDATA FIFO to store and forward data for improving GT efficiency.	Y **Note** For this event, the request security attribute Secure or Non-secure is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.
0x26	N/A	N
0x27	N/A	N
0x28	N/A	N
0x29	N/A	N
0x2A	Write request stall because of a lack of GT credit	Y **Note** For this event, the request security attribute Secure or Non-secure is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.

Table 4-9 HSNI performance events (continued)

Event code [4:0]	Event	Secure only
0x2B	Read request stall because of a lack of GT credit	Y **Note** For this event, the request security attribute Secure or Non-secure is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.
0x2C	N/A	N
0x2D	N/A	N
0x2E	N/A	N
0x2F	N/A	N
0x30	N/A	N
0x31	N/A	N

4.7 AHB Master Network Interface performance events

The NI-700 HMNI can generate various performance events.

The following table shows the performance events that the HMNI can track.

Table 4-10 HMNI performance events

Event code [4:0]	Event	Secure only
0x00	Read request: any	N
0x01	Read request: Device, Device-nE, and Device-E	N
0x02	Read request: 1. Normal Non-cacheable, Non-shareable 2. Write-Through, Non-shareable 3. Write-back, Non-shareable	N
0x03	Read request: Normal, Non-cacheable, Shareable	N
0x04	Read request: 1. Write-Through, Shareable. 2. Write-back, Shareable	
0x05	Read data beat: any	Y **Note** For this event, the request security attribute (Secure or Non-secure) is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.
0x06	N/A	N
0x07	Write request: any	N
0x08	Write request: device (Device-nE and Device-E)	N
0x09	Write request: Normal, Non-cacheable, Non-shareable	N
0x0A	Write request: Write-Through or Write-Back, Non-shareable	N
0x0B	Write request: Normal, Non-cacheable, Shareable	N
0x0C	Write request: 1. Write-Through, Shareable 2. Write-back, Shareable	N
0x0D	Write data beat: any	Y **Note** For this event, the request security attribute (Secure or Non-secure) is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.

Table 4-10 HMNI performance events (continued)

Event code [4:0]	Event	Secure only
0x0E	Read address phase stall. **HTRANS[1]** HIGH, **HWRITE** LOW, **HREADY** LOW.	N
0x0F	Read data phase stall. Prior read address phase, **HREADY** LOW.	Y
0x10	Write address phase stall. **HTRANS[1]** HIGH, **HWRITE** HIGH, **HREADY** LOW.	N
0x11	Write data phase stall. Prior write address phase, **HREADY** LOW.	Y
0x12	Reserved	N/A
0x13	N/A	N
0x20	N/A	N
0x21	N/A	N
0x22	Write response stall because of a lack of GT credit.	Y **Note** For this event, the request security attribute (Secure or Non-secure) is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.
0x23	Read response stall because of a lack of GT credit	Y **Note** For this event, the request security attribute (Secure or Non-secure) is not available at the point the event is captured. Therefore, to ensure Secure information is not exposed, the event is captured only when Secure Debug is enabled.
0x24	N/A	N
0x25	N/A	N

4.8 Data bandwidth at HSNI and HMNI

Data bandwidth performance can be monitored at AHB slave and network interfaces.

This section contains the following subsections:

4.8.1 Read and Write bandwidth (HSNI and HNMI)

NI-700 provides performance monitoring events to track the number of read and write data beats being transferred. These values can be used to calculate the total read and write bandwidth in the interconnect.

The following table shows the events that measure the number of read and write data beats.

Table 4-11 Read and Write data beat tracking events

Event code [5:0]	Description
0x05	Read data beat: any
0x0D	Write data beat: any

Calculate the read and write bandwidth according to the following calculations:

- Read bandwidth = ((Number Read Data beats × AHBDataBeatSize) / Cycles) × Frequency
- Write bandwidth = ((Number Write Data beats × AHBDataBeatSize) / Cycles) × Frequency

Note

AHBDataBeatSize is the number of bytes for each AHB beat. **HSIZE** determines this number which must be less than or equal to the size of the AHB bus.

4.8.2 Delays at HSNI and HMNI because of backpressure

To analyze the delays in HSNI and HMNI, NI-700 enables you to monitor the source of backpressure.

The following table shows the events that monitor such backpressure.

Table 4-12 Backpressure monitoring events

Event code [5:0]	Description
0x0E	Read request stall: **HREADY** LOW from the Slave
0x0F	N/A
0x10	Write request stall: **HREADY** LOW
0x11	Write data stall: **HREADY** LOW
0x12	N/A

4.8.3 Delays at HSNI because of structural backpressure

To analyze the delays in HSNI specifically, NI-700 enables you to monitor the source of backpressure because of structure full or other AXI ordering conditions.

The following table shows events that monitor such backpressure.

Table 4-13 Structural backpressure monitoring events

Event code [5:0]	Description
0x24	Request stall because of nonzero outstanding write counter
0x25	W stall because WDATA FIFO is full. HSNI uses the WDATA FIFO to store and forward data for improving GT efficiency.

4.9 APB Master Network Interface performance events

The NI-700 PMNI can generate various performance events.

The following table shows the performance events that the PMNI can track.

Table 4-14 PMNI performance events

Event code [4:0]	Event	Secure Only
0x00	Read request: any (**PENABLE** & **PREADY**) and ~**PWRITE**	N
0x01	Read request: device	N
0x02	Read request: Non-shareable (Domain == Non-shareable or system shareable)	N
0x03	N/A	N
0x04	N/A	N
0x05	Read data beat: any **PRDATA**	Y
0x06	N/A	Y
0x07	Write request: any (**PENABLE** & **PREADY**) and **PWRITE**	N
0x08	Write request: device	N
0x09	Write request: Non-shareable (Domain == Non-shareable or system shareable)	N
0x0A	N/A	N
0x0B	N/A	N
0x0C	N/A	N
0x0D	Write data beat: any (**PWDATA** & **PREADY**) and write	N
0x0E	Read request stall: **PREADY** LOW for Read, when **PENABLE** is HIGH	N
0x0F	Read data stall: **PREADY** LOW for Read, when **PENABLE** is HIGH	N
0x10	Write request stall: **PREADY** LOW for Write, when **PENABLE**is HIGH	N
0x11	Write data stall: **PREADY** LOW for Write, when **PENABLE** is HIGH	N
0x12	N/A	N
0x13	N/A	N
0x20	N/A	N
0x21	N/A	N
0x22	Write response stall because of a lack of GT credit	N
0x23	Read response stall because of a lack of GT credit	N
0x24	N/A	N
0x25	N/A	N

Appendix A
Signal descriptions

This appendix describes the external signals of the NI-700.

Note

Unless specified otherwise, signals are active-HIGH.

It contains the following sections:

A.1 AXI signals

The following sections describe the AXI interface signals.

This section contains the following subsections:

A.1.1 AXI slave signals on the write address channel

The following table shows the slave interface signals on the write address channel.

Table A-1 Write address channel slave interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_AWID[n:0]	Input	Write address ID, width is configurable
<prefix>_AWADDR[n:0]	Input	Write address, width is configurable from 32-bit to 64-bit.
<prefix>_AWLEN[7:0]	Input	Write Burst length
<prefix>_AWSIZE[2:0]	Input	Write Burst size
<prefix>_AWBURST[1:0]	Input	Write Burst type
<prefix>_AWLOCK	Input	Write lock type
<prefix>_AWCACHE[3:0]	Input	Write cache type
<prefix>_AWPROT[2:0]	Input	Write protection type
<prefix>_AWQOS[3:0]	Input	Write *Quality of Service* (QoS) value
<prefix>_AWUSER[n:0]	Input	User-specified extension to AW payload
<prefix>_AWVALID	Input	Write address valid
<prefix>_AWNSAID[3:0]	Input	NSAID signal associated with write address channel
<prefix>_AWREADY	Output	Write address ready

The following table shows the write address channel ACE-Lite specific signals.

Table A-2 Write address channel ACE-Lite specific signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_AWSNOOP[3:0]	Input	Transaction type for shareable write transactions
<prefix>_AWDOMAIN[1:0]	Input	Indicates the Shareability domain of a write transaction

The following table shows the write address channel AXI5 extension and ACE-Lite signals.

Table A-3 Write address channel AXI5 extension and ACE-Lite signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_AWTRACE	Input	Trace signals that are associated with the AW write address channel. AXI5 and ACE-Lite only.
<prefix>_AWSTASHNID	Input	Indicates the node identifier of the physical interface that is the target interface for the Cache Stash operation
<prefix>_AWSTASHNIDEN	Input	When asserted, this signal indicates the **AWSTASHNID** signal is valid and must be used
<prefix>_AWSTASHLPID	Input	Indicates the logical processor subunit associated with the physical interface that is the target for the Cache Stash operation
<prefix>_AWSTASHLPIDEN	Input	When asserted, this signal indicates the **AWSTASHLPID** signal is enabled and must be used.
<prefix>_AWATOP	Input	The signal is **AWATOP**, AW Atomic Operation
<prefix>_AWLOOP	Input	**LOOP** signal associated with the AW Write Address channel
<prefix>_AWMPAM	Input	Write address channel MPAM information
<prefix>_AWIDUNQ	Input	Write address channel unique ID indicator, active-HIGH
<prefix>_AWCMO	Input	Indicates the type of Cache Maintenance Operation
<prefix>_AWTAGOP	Input	Write request tag is in operation. Encoded as: **00** Invalid **01** Transfer **10** Update **11** Match

A.1.2 AXI slave signals on the write data channel

The following table shows the slave interface signals on the write data channel.

Table A-4 Write data channel slave interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_WDATA[DATA_WIDTH-1:0]	Input	Write data
<prefix>_WSTRB[(DATA_WIDTH/8)-1:0]	Input	Write byte lane strobes
<prefix>_WLAST	Input	Write data last transfer indication
<prefix>_WUSER[n:0]	Input	User-specified extension to W payload
<prefix>_WVALID	Input	Write data valid
<prefix>_WREADY	Output	Write data ready

The following table shows the write data channel AXI5 extension and ACE-Lite signals.

Table A-5 Write data channel AXI5 extension and ACE-Lite signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_WTRACE	Input	Trace signals that are associated with the W write data channel. AXI5 and ACE-Lite only.
<prefix>_WTAG	Input	The tag associated with write data. There is a 4-bit tag per 128-bits of data, with a minimum of 4-bits. **WTAG**[((4 × n)-1):4 × (n-1)] corresponds to **WDATA**[((128 × n)-1):128 × (n-1)] ——— **Note** ——— **WTAG** has the same validity rules as **WDATA**. ———
<prefix>_WTAGUPDATE	Input	Indicates which tags must be written to memory when an Update operation occurs. • If a bit is asserted, then the corresponding tags must be written to memory. • If a bit is deasserted, then the corresponding tags are invalid. There is 1-bit per 4-bits of tag. **WTAGUPDATE[n]** corresponds to **WTAG**[(4n)+3:(4n)]. **WTAGUPDATE** bits outside of the transaction container must be deasserted. For operations other than Update, **WTAGUPDATE** must be deasserted. It can be asserted or deasserted for Update operations.

A.1.3 AXI slave signals on the write response channel

The following table shows the slave interface signals on the write response channel.

Table A-6 Write response channel slave interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_BID[n:0]	Output	Write response ID, width is configurable
<prefix>_BRESP[1:0]	Output	Write response
<prefix>_BUSER[n:0]	Output	User-specified extension to B payload
<prefix>_BVALID	Output	Write response valid
<prefix>_BREADY	Input	Write response ready

The following table shows the write response channel AXI5 extension and ACE-Lite signals.

Table A-7 Write response channel AXI5 extension and ACE-Lite signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_BTRACE	Output	Trace signals that are associated with the B write response channel. AXI5 and ACE-Lite only.
<prefix>_BLOOP	Output	LOOP signal associated with the B Write Response channel.
<prefix>_BIDUNQ	Output	Write response channel unique ID indicator, active-HIGH.

Table A-7 Write response channel AXI5 extension and ACE-Lite signals (continued)

Signal	Direction	Description
<prefix>_BCOMP	Output	Indicates that the write is observable.
<prefix>_BPERSIST	Output	Indicates that the write data is updated in persistent memory. Can only be asserted for transactions where AWCMO is CleanSharedPersist or CleanSharedDeepPersist.
<prefix>_BTAGMATCH	Output	Indicates the result of a tag comparison on a write transaction: **00** Not a match transaction **01** No match result **10** Fail **11** Pass

A.1.4 AXI slave signals on the read address channel

The following table shows the slave interface signals on the read address channel.

Table A-8 Read address channel slave interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_ARID[n:0]	Input	Read data ID. Width is configurable.
<prefix>_ARADDR	Input	Address of the first transfer in a read transaction
<prefix>_ARLEN	Input	Length. The exact number of data transfers in a read transaction.
<prefix>_ARSIZE	Input	Size. The number of bytes in each data transfer in a read transaction.
<prefix>_ARBURST	Input	Burst type. Indicates how address changes between each transfer in a read transaction.
<prefix>_ARLOCK	Input	Information about the atomic characteristics of a read transaction
<prefix>_ARCACHE	Input	Indicates how a read transaction is required to progress through a system.
<prefix>_ARPROT	Input	Protection attributes of a read transaction: privilege, security level, and access type
<prefix>_ARQOS	Input	Quality of Service identifier for a read transaction
<prefix>_ARUSER	Input	User-defined extension for the read address channel
<prefix>_ARVALID	Input	Indicates that the read address channel signals are valid.
<prefix>_ARNSAID	Input	NSAID associated with the read address channel
<prefix>_ARREADY	Output	Indicates that a transfer on the read address channel can be accepted.

The following table shows the read address channel ACE-Lite specific signals.

Table A-9 Read address channel ACE-Lite specific signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_ARSNOOP[3:0]	Input	Transaction type for shareable read transactions.
<prefix>_ARDOMAIN[1:0]	Input	Shareability domain of a read transaction.

The following table shows the read address channel AXI5 extension and ACE-Lite signals.

Table A-10 Read address channel AXI5 extension and ACE-Lite signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_ARTRACE	Input	Trace signal that is associated with the AR read address channel. AXI5 and ACE-Lite only.
<prefix>_ARLOOP	Input	LOOP signal associated with the AR Read Address channel.
<prefix>_ARMPAM	Input	Read address channel MPAM information.
<prefix>_ARIDUNQ	Input	Read address channel unique ID indicator, active-HIGH.
<prefix>_ARCHUNKEN	Input	If this signal is asserted, read data for this transaction can be returned out of order, in 128-bit chunks.
<prefix>_ARTAGOP	Input	Read request tag operation. Encoded as: **0b00** Invalid **0b01** Transfer **0b10** Reserved **0b11** Fetch

A.1.5 AXI slave signals on the read data channel

The following table shows the slave interface signals on the read data channel.

Table A-11 Read data channel slave interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_RID[n:0]	Output	Read data ID, width is configurable
<prefix>_RDATA[DATA_WIDTH-1:0]	Output	Read data
<prefix>_RRESP[3:0]	Output	Read data response
<prefix>_RLAST	Output	Read data last transfer indication
<prefix>_RUSER[n:0]	Output	User-specified extension to R payload
<prefix>_RVALID	Output	Read data valid
<prefix>_RREADY	Input	Read data ready

The following table shows the read data channel AXI5 and ACE-Lite extension signals.

Table A-12 Read data channel AXI5 and ACE-Lite extension signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_RTRACE	Output	Trace signal that is associated with the R read data channel. AXI5 and ACE-Lite only.
<prefix>_RLOOP	Output	**LOOP** signal associated with the R Read Data channel.
<prefix>_RIDUNQ	Output	Read data channel unique ID indicator, active-HIGH.
<prefix>_RCHUNKV	Output	If this signal is asserted, **RCHUNKNUM** and **RCHUNKSTRB** are valid for this transfer.
<prefix>_RCHUNKNUM	Output	Indicates the chunk number being transferred. Chunks are numbered incrementally from zero, according to the data width and base address of the transaction.
<prefix>_RCHUNKSTRB	Output	Indicates which part of read data is valid for this transfer, each bit corresponds to 128- bits of data. For example: • **RCHUNKSTRB[0]** corresponds to **RDATA[127:0]** • **RCHUNKSTRB[1]** corresponds to **RDATA[255:128]**
<prefix>_RTAG	Output	The tag associated with read data. There is a 4-bit tag per 128-bits of data, with a minimum of 4-bits. **RTAG**[((4 × n)-1) : 4 × (n-1)] corresponds to **RDATA**[((128 × n)-1) : 128 × (n-1)] ——— **Note** ——— **RTAG** has the same validity rules as **RDATA**. ———

A.1.6 Other AXI signals

The following table shows other AXI signals.

Table A-13 Other AXI signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_QOSOVERRIDE	Input	Sample at reset QoS override. See Chapter 2, Operation, of the *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Technical Reference Manual*.
<prefix>_ORDERED_WRITE_OBSERVATION	Input	Enables OWO on this slave interface if asserted. Refer to the Ordered Write Observation feature in the AXI protocol specification. The Cyclic Dependency Avoidance Scheme has a sub-section on Ordered Write Observation. See Chapter 2, Operation, of the *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Technical Reference Manual*.

A.1.7 AXI master signals on the write address channel

The following table shows the master interface signals on the write address channel.

Table A-14 Write address channel master interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_AWID[n:0]	Output	Write address ID
<prefix>_AWADDR[n:0]	Output	Write address. The width is configurable from 32-64.
<prefix>_AWLEN[7:0]	Output	Write Burst length
<prefix>_AWSIZE[2:0]	Output	Write Burst size
<prefix>_AWBURST[1:0]	Output	Write Burst type
<prefix>_AWLOCK	Output	Write lock type
<prefix>_AWCACHE[3:0]	Output	Write cache type
<prefix>_AWPROT[2:0]	Output	Write protection type
<prefix>_AWQOS[3:0]	Output	Write *Quality of Service* (QoS) value
<prefix>_AWUSER[n:0]	Output	User-specified extension to AW payload
<prefix>_AWVALID	Output	Write address valid
<prefix>_AWNSAID[3:0]	Output	NSAID signal that is associated to the write address channel
<prefix>_AWREADY	Input	Write address ready

The following table shows the write address channel ACE-Lite specific signals.

Table A-15 Write address channel ACE-Lite specific signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_AWSNOOP[3:0]	Output	The transaction type for shareable write transactions.
<prefix>_AWDOMAIN[1:0]	Output	Indicates the Shareability domain of a write transaction

The following table shows the write address channel AXI5 extension and ACE-Lite signals.

Table A-16 Write address channel AXI5 extension and ACE-Lite signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_AWTRACE	Output	Trace signals that are associated with the AW write address channel. AXI5 and ACE-Lite only.
<prefix>_AWSTASHNID	Output	Indicates the node identifier of the physical interface that is the target interface for the Cache Stash operation.
<prefix>_AWSTASHNIDEN	Output	When asserted, this signal indicates that the **AWSTASHNID** signal is valid and must be used.
<prefix>_AWSTASHLPID	Output	Indicates the logical processor subunit associated with the physical interface that is the target for the Cache Stash operation.
<prefix>_AWSTASHLPIDEN	Output	When asserted, this signal indicates that the **AWSTASHLPID** signal is enabled and must be used.

Table A-16 Write address channel AXI5 extension and ACE-Lite signals (continued)

Signal	Direction	Description
<prefix>_AWATOP	Output	This signal is AWATOP, AW Atomic Operation
<prefix>_AWLOOP	Output	LOOP signal that is associated with the AW Write Address channel
<prefix>_AWMPAM	Output	Write address channel MPAM information
<prefix>_AWIDUNQ	Output	Write address channel unique ID indicator, active-HIGH
<prefix>_AWCMO	Output	Indicates the type of Cache Maintenance Operation
<prefix>_AWTAGOP	Output	The Write request tag is in operation. Encoded as: **00** Invalid **01** Transfer **10** Update **11** Match

A.1.8 AXI master signals on the write data channel

The following table shows the master interface signals on the write data channel.

Table A-17 Write data channel master interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_WDATA[n:0]	Output	Write data
<prefix>_WID[n:0]	Output	The output write data ID. **Note** NI-700 does not perform write data interleaving across transactions. The signal exists only for integration purposes.
<prefix>_WSTRB[(DATA_WIDTH/8)-1:0]	Output	Write byte lane strobes
<prefix>_WLAST	Output	Write data last transfer indication
<prefix>_WUSER[n:0]	Output	User-specified extension to W payload
<prefix>_WVALID	Output	Write data valid
<prefix>_WREADY	Input	Write data ready

The following table shows the write data channel AXI5 extension and ACE-Lite signals.

Table A-18 Write data channel AXI5 extension and ACE-Lite extension signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_WTRACE	Output	Trace signals associated with the W write data channel. AXI5 and ACE-Lite only.

Table A-18 Write data channel AXI5 extension and ACE-Lite extension signals (continued)

Signal	Direction	Description
<prefix>_WTAG	Output	The tag associated with write data. There is a 4-bit tag per 128-bits of data, with a minimum of 4-bits. **WTAG**[((4 × n)-1):4 × (n-1)] corresponds to **WDATA**[((128 × n)-1):128 × (n-1)] ——— **Note** ——— **WTAG** has the same validity rules as **WDATA**. ———
<prefix>_WTAGUPDATE	Output	Indicates which tags must be written to memory when an Update operation occurs. • If a bit is asserted, then the corresponding tags must be written to memory. • If a bit is deasserted, then the corresponding tags are invalid. There is 1-bit per 4-bits of tag. **WTAGUPDATE[n]** corresponds to **WTAG**[(4n)+3:(4n)] **WTAGUPDATE** bits outside of the transaction container must be deasserted For operations other than Update, **WTAGUPDATE** must be deasserted. It can be asserted or deasserted for Update operations.

A.1.9 AXI master signals on the write response channel

The following table shows the master interface signals on the write response channel.

Table A-19 Write response channel master interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_BID[n:0]	Input	Write response ID. Width is configurable.
<prefix>_BRESP[1:0]	Input	Write response.
<prefix>_BUSER[n:0]	Input	User-specified extension to B payload.
<prefix>_BVALID	Input	Write response valid.
<prefix>_BREADY	Output	Write response ready.

The following table shows the write response channel AXI5 extension and ACE-Lite signals.

Table A-20 Write response channel AXI5 and ACE-Lite extension signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_BTRACE	Input	Trace signal that is associated with the B writes response channel. AXI5 and ACE-Lite only.
<prefix>_BLOOP	Input	LOOP signal that is associated with the B Write Response channel.
<prefix>_BIDUNQ	Input	Write response channel unique ID indicator, active-HIGH.
<prefix>_BCOMP	Input	Indicates that the write is observable.

Table A-20 Write response channel AXI5 and ACE-Lite extension signals (continued)

Signal	Direction	Description
<prefix>_BPERSIST	Input	Indicates that the write data is updated in persistent memory. Can only be asserted for transactions where **AWCMO** is CleanSharedPersist or CleanSharedDeepPersist.
<prefix>_BTAGMATCH	Input	Indicates the result of a tag comparison on a write transaction: **00** Not a match transaction **01** No match result **10** Fail **11** Pass

A.1.10 AXI master signals on the read address channel

The following table shows the master interface signals on the read address channel.

Table A-21 Read address channel master interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_ARID[n:0]	Output	Read data ID. Width is configurable.
<prefix>_ARADDR[n:0]	Output	Address of the first transfer in a read transaction.
<prefix>_ARLEN[7:0]	Output	Length. The exact number of data transfers in a read transaction.
<prefix>_ARSIZE[2:0]	Output	Size. The number of bytes in each data transfer in a read transaction.
<prefix>_ARBURST[1:0]	Output	Burst type. Indicates how address changes between each transfer in a read transaction.
<prefix>_ARLOCK	Output	Information about the atomic characteristics of a read transaction.
<prefix>_ARCACHE[3:0]	Output	Indicates how a read transaction is required to progress through a system.
<prefix>_ARPROT[2:0]	Output	Protection attributes of a read transaction: • Privilege • Security level • Access type
<prefix>_ARQOS[3:0]	Output	Quality of Service identifier for a read transaction.
<prefix>_ARUSER[n:0]	Output	User-defined extension for the read address channel.
<prefix>_ARVALID	Output	Indicates that the read address channel signals are valid.
<prefix>_ARNSAID[3:0]	Input	NSAID associated with the read address channel.
<prefix>_ARREADY	Input	Indicates that a transfer on the read address channel can be accepted.

The following table shows the read address channel ACE-Lite specific signals.

Table A-22 Read address channel ACE-Lite specific signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_ARSNOOP[3:0]	Output	Transaction type for shareable read transactions.
<prefix>_ARDOMAIN[1:0]	Output	Shareability domain of a read transaction.

The following table shows the read address channel AXI5 and ACE-Lite signals extension.

Table A-23 Read address channel AXI5 extension and ACE-Lite signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_ARTRACE	Output	Trace signal that is associated with the AR read address channel. AXI5 and ACE-Lite only.
<prefix>_ARLOOP	Output	The LOOP signal that is associated with the AR Read Address channel.
<prefix>_ARMPAM	Output	Read address channel MPAM information.
<prefix>_ARIDUNQ	Output	Read address channel unique ID indicator, active-HIGH.
<prefix>_ARCHUNKEN	Output	If this signal is asserted, read data for this transaction can be returned out of order, in 128-bit chunks.
<prefix>_ARTAGOP	Output	Read request tag operation. Encoded as: `0b00` Invalid `0b01` Transfer `0b10` Reserved `0b11` Fetch

A.1.11 AXI master signals on the read data channel

The following table shows the master interface signals on the read data channel.

Table A-24 Read data channel master interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_RID[n:0]	Input	Read data ID. Width is configurable.
<prefix>_RDATA[DATA_WIDTH-1:0]	Input	Read data
<prefix>_RRESP[3:0]	Input	Read data response
<prefix>_RLAST	Input	Read data last transfer indication
<prefix>_RUSER[n:0]	Input	User-specified extension to R payload
<prefix>_RVALID	Input	Read data valid
<prefix>_RREADY	Output	Read data ready

The following table shows the read data channel AXI5 extension and ACE-Lite signals.

Table A-25 Read data channel AXI5 extension and ACE-Lite signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_RTRACE	Input	Trace signal that is associated with the R read data channel. AXI5 and ACE-Lite only.
<prefix>_RLOOP	Input	**LOOP** signal associated with the R Read Data channel
<prefix>_RIDUNQ	Input	Read data channel unique ID indicator, active-HIGH
<prefix>_RCHUNKVOP	Input	If this signal is asserted, **RCHUNKNUM** and **RCHUNKSTRB** are valid for this transfer.
<prefix>_RCHUNKNUM	Input	Indicates the chunk number being transferred. Chunks are numbered incrementally from zero, according to the data width and base address of the transaction.
<prefix>_RCHUNKSTRB	Input	Indicates which part of read data is valid for this transfer, each bit corresponds to 128-bits of data. For example: • **RCHUNKSTRB[0]** corresponds to **RDATA**[127:0] • **RCHUNKSTRB[1]** corresponds to **RDATA[255:128]**
<prefix>_RTAG	Input	The tag associated with read data. There is a 4-bit tag per 128-bits of data, with a minimum of 4-bits. **RTAG**[((4 × n)-1) : 4 × (n-1)] corresponds to **RDATA**[((128 × n)-1) : 128 × (n-1)] ——— **Note** ——— **RTAG** has the same validity rules as **RDATA**. ———

A.1.12 AXI3 master network interface signals

These signal tables show the changes in the AXI signals when the AXI master network interface type is configured as AXI3.

The following table shows the AXI3 master interface signals on the read address channel.

Table A-26 Read address channel AXI3 master interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_ARLEN[3:0]	Input	Length. The exact number of data transfers in a read transaction.
<prefix>_ARLOCK[1:0]	Input	Information about the atomic characteristics of a read transaction

The following table shows the AXI3 master interface signals on the write address channel.

Table A-27 Write address channel AXI3 master interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_AWLEN[3:0]	Output	Write Burst length
<prefix>_AWLOCK[1:0]	Output	Write lock type

The following table shows the AXI3 master interface signals on the write data channel.

Table A-28 Write data channel AXI3 master interface signals

Signal	Direction	Description
Where <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_WID[n:0]	Output	WID pin

A.2 AHB signals

Each configurable *AHB Slave Network Interface* (HSNI) contains slave interface signals. The following sections describe the AHB interface signals.

This section contains the following subsections:

A.2.1 AHB slave request signals

The following table shows the AHB slave interface request signals.

Table A-29 Request signals

Signal	Direction	Description
Where <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HADDR	Input	AHB address bus
<prefix>_HBURST	Input	The Burst type
<prefix>_HMASTLOCK	Input	When HIGH, indicates that the current transfer is part of a locked sequence.
<prefix>_HPROT[3:0]	Input	The protection control signals.
<prefix>_HSIZE	Input	Indicates the size of the transfer
<prefix>_HTRANS	Input	Indicates the transfer type of the current transfer
<prefix>_HWDATA	Input	The write data
<prefix>_HWRITE	Input	Indicates the transfer direction being write or read
<prefix>_HAUSER	Input	Address channel User signals
<prefix>_HWUSER	Input	Write data channel User signals

A.2.2 AHB slave response signals

The following table shows the AHB slave interface response signals.

Table A-30 Response signals

Signal	Direction	Description
Where <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HRDATA	Output	The read data from the multiplexor.
<prefix>_HREADY	Output	Ready output from HSNI core
<prefix>_HRESP	Output	The transfer response from the multiplexor.
<prefix>_HRUSER	Output	The read data channel User signal from the multiplexor.

A.2.3 AHB slave request signals in addition to AHB-Lite

The following table shows the AHB5 slave request signals in addition to AHB-Lite.

Table A-31 AHB5 slave request signals in addition to AHB-Lite

Signal	Direction	Description
Where <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HPROT	Input	The 3-bit extension of the **HPROT** signal that adds extended memory types.
<prefix>_HNONSEC	Input	Indicates whether the transfer is Secure or Non-secure
<prefix>_HEXCL	Input	Exclusive transfer
<prefix>_HMASTER	Input	The master identifier which is only used for Exclusive transfer.

A.2.4 AHB slave response signals in addition to AHB-Lite

The following table shows the AHB5 slave response signals in addition to AHB-Lite.

Table A-32 AHB5 slave response signals in addition to AHB-Lite

Signal	Direction	Description
Where <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HEXOKAY	Output	Exclusive Okay response

A.2.5 Other AHB slave signals

The following table shows other AHB slave signals, for use when HSNI is not configured as a master mirror interface.

Table A-33 Other AHB slave signals when HSNI is not configured as a master mirror interface

Signal	Direction	Description
Where <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HREADY	Input	The **HREADY** from the multiplexor going to all masters and slaves.
<prefix>_HSEL	Input	The slave select signals from the decoder.

A.2.6 AHB master request signals

The following table shows the AHB master interface request signals.

Table A-34 AHB master request signals

Signal	Direction	Description
Where <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HADDR	Output	AHB address bus
<prefix>_HBURST	Output	The Burst type
<prefix>_HMASTLOCK	Output	When HIGH, indicates that the current transfer is part of a locked sequence.
<prefix>_HPROT[3:0]	Output	Protection control signals
<prefix>_HSIZE	Output	Indicates the size of the transfer
<prefix>_HTRANS	Output	Indicates the transfer type of the current transfer

Table A-34 AHB master request signals (continued)

Signal	Direction	Description
<prefix>_HWDATA	Output	The write data
<prefix>_HWRITE	Output	Indicates the transfer direction being write or read
<prefix>_HAUSER	Output	Address channel User signals
<prefix>_HWUSER	Output	Write data channel User signals

A.2.7 AHB master response signals

The following table shows the AHB master interface response signals.

Table A-35 AHB master response signals

Signal	Direction	Description
Where <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HRDATA	Input	The read data from the multiplexor
<prefix>_HREADY/HREADYOUT	Input	As AHB master interface, it is the **HREADY** signal from the multiplexor. In AHB mirror mode, it is the **HREADYOUT** signal from the slave.
<prefix>_HRESP	Input	The transfer response from the multiplexor
<prefix>_HRUSER	Input	The read data channel User signal from the multiplexor

A.2.8 AHB master request signals in addition to AHB-Lite

The following table shows the AHB master request signals in addition to AHB-Lite.

Table A-36 AHB5 master request signals in addition to AHB-Lite

Signal	Direction	Description
Where <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HPROT[6:4]	Output	The 3-bit extension of the **HPROT** signal that adds extended memory types.
<prefix>_HNONSEC	Output	Indicates whether the transfer is Secure or Non-secure
<prefix>_HEXCL	Output	Exclusive transfer
<prefix>_HMASTER	Output	Master identifier which is only used for Exclusive transfer

A.2.9 AHB master response signals in addition to AHB-Lite

The following table shows the AHB master response signals in addition to AHB-Lite.

Table A-37 AHB5 master response signals in addition to AHB-Lite

Signal	Direction	Description
Where <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HEXOKAY	Input	Exclusive Okay response

A.2.10 Other AHB master signals

The following table shows other AHB master signals, for use when HSNI is not configured as a master mirror interface.

Table A-38 Other AHB signals when HMNI is configured as a mirrored slave interface

Signal	Direction	Description
Where <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_HREADY	Output	The **HREADY** from the multiplexor, which goes to all masters and slaves.
<prefix>_HSEL	Output	The slave select signals from the decoder

A.3 APB signals

The following sections describe the APB master interface signals.

This section contains the following subsections:

A.3.1 APB master request signals

You can configure the PMNI *(APB Master Network Interface)* to have an APB2, APB3, or APB4 slave interface. The following table shows the request signals.

Table A-39 APB2, APB3, and APB4 request signals

Signal	Direction	Description
Where <prefix> represents APB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_PADDR_0..15	Output	APB address bus.
<prefix>_PSEL_0..15	Output	APB Slave Device Select. PMNI supports up to 16 APB Slaves.
<prefix>_PENABLE_0..15	Output	ENABLE. This signal indicates the second and subsequent cycles of an APB transfer.
<prefix>_PWRITE_0..15	Output	This signal indicates an APB read or write access: **1'b0** APB read access **1'b1** APB write access
<prefix>_PWDATA_0..15	Output	Write data.

A.3.2 APB4 master request signals

APB4 specific master request signals are shown in the following table.

Table A-40 Master request signals (APB4)

Signal	Direction	Description
Where <prefix> represents APB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_PPROT_0..15	Output	**Note** NI-700 only supports Secure or Non-secure access indication corresponding to PPROT[1]. NI-700 does not transport normal or privileged access and data or instruction access.
<prefix>_PSTRB_0..15	Output	APB write data strobes. This signal indicates which byte lanes to update during a write transfer. One write strobe for each 8-bit of the write data bus. Therefore **PSTRB[n]** corresponds to **PWDATA[(8n+7):(8n)]**. Write strobes must not be active during a read transfer.

A.3.3 APB master response signals

You can configure the PMNI to have an APB2, APB3, or APB4 slave interface. The following table shows the master response signals.

Table A-41 APB2, APB3, and APB4 master response signals

Signal	Direction	Description
Where <prefix> represents APB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_PRDATA_0..15	Input	APB read data.

A.3.4 APB3 and APB4 master response signals

You can configure the PMNI to have an APB2, APB3, or APB4 slave interface. The following table shows the APB3 and APB4 specific master response signals.

Table A-42 APB3 and APB4 master response signals

Signal	Direction	Description
Where <prefix> represents APB_MASTER_<ENDPOINT_INTERFACE_NAME>		
<prefix>_PREADY_0..15	Input	Ready. The APB Slave uses this signal to extend an APB transfer (wait states).
<prefix>_PSLVERR_0..15	Input	This signal indicates a transfer failure. APB peripherals are not required to support the **PSLVERR** pin. Where a peripheral does not include this pin then the appropriate input to PMNI is tied LOW.

A.4 Power, clock, reset, and IDM control signals

NI-700 has power, clock, and reset control signals.

The following table shows the power, clock, and reset control signals.

Table A-43 Power, clock, and reset control signals

Signal	Direction	Description
<AXI>_MASTER_<ENDPOINT_INTERFACE_NAME>_AWAKEUP	Output	Indicates that the master interface has active transactions. It can be used as an indicator to turn on the clock to downstream components.
<AXI>_SLAVE_<ENDPOINT_INTERFACE_NAME>_AWAKEUP	Input	Indicates that the AXI or ACE-Lite slave interface has pending active transactions. This signal requests a clock for the NI-700.
<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_SRESETN	Output	External IDM soft reset.
<PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>_SRESETN	Output	
<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_IDM_SRESET_STRAP	Input	Sample-at-reset input pin at every master and slave interface where IDM is enabled. The value of this pin determines the value of the IDM_RESET_CONTROL register out of reset. The value of the pin also determines the external IDM soft reset pin at that interface.
<PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>_IDM_SRESET_STRAP	Input	
<ENDPOINT_INTERFACE_NAME>_CONFIG_ACCESS	Input	Sample-at-reset input pin per slave interface. This signal indicates the slave interfaces that are permitted to accept new transactions in the CONFIG power state.
<CLKNAME>_CLK	Input	Clock input for that clock domain.
<CLKNAME>_RESETN	Input	Reset signal that is associated with the clock domain. Active-LOW.

Table A-43 Power, clock, and reset control signals (continued)

Signal	Direction	Description
<CLKNAME>_AON_CLK	Input	Feeds the HSNI buffer stage and must be on before the initial transaction ingresses into the device so the transaction is not lost.
<CLKNAME>_AON_RESETN	Input	Reset signal that feeds the HSNI buffer stage.
<CLKNAME>_QREQN	Input	Request to disable the **<CLKNAME>_CLK** input. Active-LOW.
<CLKNAME>_QACCEPTN	Output	Clock disable acceptance response. Active-LOW.
<CLKNAME>_QDENY	Output	Clock disable denial response.
<CLKNAME>_QACTIVE	Output	Indicates that the NI-700 requires the **<CLKNAME>_CLK** input to run.
<PDOMAIN>_PREQ	Input	Request to change power state for power domain **<PDOMAIN>**.
<PDOMAIN>_PSTATE[7:0]	Input	Required power state.
<PDOMAIN>_PACCEPT	Output	Power state transition acceptance.
<PDOMAIN>_PDENY	Output	Power state transition denial.
<PDOMAIN>_PACTIVE[16:0]	Output	Indicates the available power states for the non-coherent interconnect.

Table A-43 Power, clock, and reset control signals (continued)

Signal	Direction	Description
<PDOMAIN>_INTERRUPT	Output	Secure interrupt per power domain. This interrupt is used to indicate specific conditions (IDM or non-IDM) within a slave or master interface. This interrupt is rising edge triggered. See *Chapter 3 Programmers model* on page 3-115 for the conditions.
<PDOMAIN>_NS_INTERRUPT	Output	Non-secure interrupt per power domain. This interrupt is used to indicate specific conditions (IDM or non-IDM) within a slave or master interface. This interrupt is rising edge triggered. See *Chapter 3 Programmers model* on page 3-115 for the conditions.

A.5 Design for Test signals

NI-700 contains *Design for Test* (DFT) signals.

The following table shows the DFT signals.

Table A-44 DFT signals

Signal	Direction	Description
DFTRSTDISABLE[1:0]	Input	Internal resets are disabled
DFTCGEN	Input	To enable architectural clock gates for **CLKNAME** clocks, assert HIGH during scan shift.
DFT<CLKNAME>CLKDISABLE	Input	Disable clock **Note** Each clock domain in a NI-700 configuration is assigned a separate bit of **DFT<CLKNAME>CLKDISABLE**.

A.6 PMU and debug signals

NI-700 contains PMU and debug signals.

The following table shows the PMU and debug signals.

Table A-45 PMU and debug signals

Signal	Direction	Description
<CLKNAME>_NIDEN	Input	Non-invasive debug enable. If HIGH, the signal enables counting and export of PMU events.
<CLKNAME>_SPNIDEN	Input	Secure privileged non-invasive debug enable. When HIGH, this signal enables the counting of both Non-secure and Secure events, provided **NIDEN** is also HIGH.
<CLKNAME>_DBGEN	Input	Invasive debug enable. If HIGH, enables the counting and export of PMU events.
<CLKNAME>_SPIDEN	Input	Secure privileged invasive debug enable. When HIGH, this signal enables the counting of both Non-secure and Secure events, provided that **DBGEN** is also HIGH.
<CLKNAME>_PMUSNAPSHOTREQ	Input	Four-phase request to initiate snapshot of PMU counters.
<CLKNAME>_PMUSNAPSHOTACK	Output	Acknowledgment of PMU snapshot capture
<CLKNAME>_nPMUINTERRUPT	Output	Active-LOW interrupt to indicate a counter (event or cycle) has overflowed. This interrupt is rising edge triggered.

Appendix B
Revisions

This appendix describes the technical changes between released issues of this book.

It contains the following section:

- *B.1 Revisions* on page Appx-B-339.

B.1 Revisions

This appendix describes the technical changes between released issues of this book.

Table B-1 Issue 0000-00

Change	Location
First dev release	-

Table B-2 Differences between issue 0000-00 and issue 0000-01

Change	Location
Minor editorial and technical updates throughout the document.	All sections.
Added description of AXI5 **AWAKEUP** signal implementation.	*1.1 About the CoreLink NI-700 Network-on-Chip Interconnect* on page 1-12
Added Configurable options section.	*1.6 Configurable options* on page 1-19
Updated top-level architecture diagram and associated note (describing top-level PCDC configuration).	*1.5 Architecture overview* on page 1-17
Added information about the configurable options that apply to all functional units.	*2.1 About the functional units* on page 2-33
Added description of burst splitting scenarios and low and high-wire modes for ASNI and AMNI units.	*2.1.1 AXI Slave Network Interface* on page 2-33, *2.1.2 AXI Master Network Interface* on page 2-34
Updated ASNI, AMNI, PCDC, and Router configuration options.	*1.6.1 ASNI configuration options* on page 1-20 *1.6.2 AMNI configuration options* on page 1-22 *Configuration options* on page 2-38 *Configuration options* on page 2-38
Added information about combining Q-Channel LPIs at the top level.	*2.1.6 Power and Clock Domain Crossing* on page 2-37
Merged functional description of the NI-700 resets with functional description of power and clock management.	*2.2 Power, clock, and reset management* on page 2-40
Added section describing the NI-700 clock gating hierarchy.	*Levels of clock gating* on page 2-42
Added External clock controller section.	*External Clock Controller* on page 2-43
Added Power control section.	*2.2.3 Power control* on page 2-45
Added Clock and reset control section.	*2.2.4 Clock and reset control* on page 2-49
Added NodeID mapping and discovery section and moved descriptions of configuration register regions and access to this section.	*2.3 Node ID mapping and discovery* on page 2-53

Table B-2 Differences between issue 0000-00 and issue 0000-01 (continued)

Change	Location
Updated description of node configuration register address map to add a description of the discovery tree that is built by software at the end of discovery.	*2.3.2 Node configuration register address-mapping overview* on page 2-54
Added Security section.	*2.8 Security* on page 2-89
Updated description of remap configuration and constraints.	*2.4.5 Remap* on page 2-62
Updated description of security attribute mismatch handling.	*2.8.1 TrustZone® technology and security* on page 2-89
Added Interconnect Device Management section.	*2.5 Interconnect Device Management* on page 2-66
Added footnote to Peripheral ID4 register reset value to indicate that it is partially device dependent.	*3.2.1 Global registers summary* on page 3-118
Added voltage domain, power domain, and clock domain Secure Access Registers summaries and descriptions.	*Table 3-18 Voltage domain registers summary* on page 3-129, *Table 3-23 Power domain registers summary* on page 3-132, *Table 3-48 Clock domain registers summary* on page 3-150
Updated description of global, ASNI, and AMNI Secure Access Registers.	*SECR_ACC, Secure access register* on page 3-120, *ASNI_SECR_ACC, Secure access register* on page 3-174, *AMNI_SECR_ACC, Secure access register* on page 3-202
Updated Slave interface registers summary table.	*Table 3-75 ASNI registers summary* on page 3-170
Added ASNI_IDM_DEVICE_ID and ASNI_IDM_RESET_CONTROL register summary and description.	*Table 3-75 ASNI registers summary* on page 3-170
Updated ASNI Address Remap Vector Register description.	*ASNI_ADDR_REMAP, Address remap vector register* on page 3-181
Updated Usage constraints of various registers.	*ASNI_SILDBG, ASNI silicon debug monitor register* on page 3-182, *AMNI_SILDBG, Silicon debug monitor register* on page 3-207
Updated Master interface registers summary table.	*Table 3-111 AMNI registers summary* on page 3-199
Added AMNI_IDM_DEVICE_ID and AMNI_IDM_RESET_CONTROL registers summary and description.	*3.8.1 AMNI registers summary* on page 3-199
Updated Performance Monitoring Unit registers summary table	*Table 3-53 PMU registers summary* on page 3-153
Updated PMEVTYPERn, PMSSCR, and PMCFGR Register descriptions	*PMEVTYPERn, Performance monitor event type and filter registers* on page 3-157, *PMSSCR, Performance monitors snapshot capture register* on page 3-161, *PMCFGR, Performance monitors configuration register* on page 3-168
Added Performance optimization and monitoring chapter.	*Chapter 4 Performance monitoring* on page 4-292
Updated master interface signal tables.	*Appendix A Signal descriptions* on page Appx-A-313
Updated slave interface signal tables.	*Appendix A Signal descriptions* on page Appx-A-313

Table B-2 Differences between issue 0000-00 and issue 0000-01 (continued)

Change	Location
Updated power and clock control signal tables.	*A.4 Power, clock, reset, and IDM control signals* on page Appx-A-333
Added PMU and debug signal descriptions.	*A.6 PMU and debug signals* on page Appx-A-337

Table B-3 Differences between issue 0000-01 and issue 0000-02

Change	Location
Updated ASNI configuration options.	*1.6.1 ASNI configuration options* on page 1-20
Updated AMNI configuration options.	*1.6.2 AMNI configuration options* on page 1-22
Added description of minimum latency for HSNI requests.	*1.5 Architecture overview* on page 1-17
Added NIs to more than one clock domain.	*2.3.8 Configuration register address region calculation* on page 2-58
Updated IDM description.	*2.5 Interconnect Device Management* on page 2-66
Updated QoS features.	*2.11 Quality of Service* on page 2-96
Updated programmers model.	*Chapter 3 Programmers model* on page 3-115
Updated signal descriptions.	*Appendix A Signal descriptions* on page Appx-A-313

Table B-4 Differences between issue 0000-02 and issue 0000-03

Change	Location
Added information on error handling and interrupts.	*2.6 Error handling and interrupts* on page 2-81
Added information on network interface IDM registers.	*3.11 Network Interface IDM registers* on page 3-251
Added configuration information to functional units	*2.1 About the functional units* on page 2-33
Extended security information	*2.8 Security* on page 2-89

Table B-5 Differences between issue 0000-03 and issue 0000-04

Change	Location
Updated protocol information about the NI-700 Interconnect.	*1.1 About the CoreLink NI-700 Network-on-Chip Interconnect* on page 1-12
Updated protocol information about the NI-700 Interconnect.	*1.5 Architecture overview* on page 1-17
Added information on master/slave interface configuration options.	*1.6 Configurable options* on page 1-19
Updated the HSNI configuration options.	*1.6.3 HSNI configuration options* on page 1-25
Updated the HMNI configuration options.	*1.6.4 HMNI configuration options* on page 1-27
Added configuration data on Pipeline slices to the AXI Slave Network Interface functional description.	*2.1.1 AXI Slave Network Interface* on page 2-33
Added configuration data on Pipeline slices to the AXI Master Network Interface functional description.	*2.1.2 AXI Master Network Interface* on page 2-34

Table B-5 Differences between issue 0000-03 and issue 0000-04 (continued)

Change	Location
Added configuration data on pipeline slices to the AHB Slave Network Interface functional description and scenarios for multi-copy atomicity.	*2.1.3 AHB Slave Network Interface* on page 2-35
Added configuration data on pipeline slices to the AHB Master Network Interface functional description.	*2.1.4 AHB Master Network Interface* on page 2-36
Added configuration data on pipeline slices to the APB Master Network Interface functional description.	*2.1.5 APB Master Network Interface* on page 2-37
Updated the functional description of AHB address phase buffering in HSNI.	*AHB address phase buffering in HSNI* on page 2-49
Added a section on external interfaces and their InterfaceID with an explanatory diagram.	*2.3.7 Interface ID* on page 2-58
Updated the APB security configuration options.	*2.8.4 Security access permissions of APB requests* on page 2-91
Added a note to Interconnect Device Management.	*2.5 Interconnect Device Management* on page 2-66
Added a section on the user signal widths.	*2.13.8 User signals* on page 2-112
Added a section on the ASNI address decoders.	*2.4.1 ASNI address decode* on page 2-61
Added a section on the HSNI address decoders.	*2.4.2 HSNI address decode* on page 2-61
Added a section on the PMNI address decoders.	*2.4.3 PMNI address decode* on page 2-61
Added information on the Address Hash Function in Address striping.	*2.4.4 Address striping* on page 2-61
Updated the functional description of AHB address phase buffering in HSNI.	*AHB address phase buffering in HSNI* on page 2-49
Updated configuration information for all Secure and Non-secure IDM Power Domain register descriptions.	For Secure IDM_PD registers: *IDM_PD_ERROR_STATUS* on page 3-136, to *IDM_PD_ACCESS_CONTROL* on page 3-141. For Non-secure IDM_PD registers: *IDM_PD_ERROR_STATUS_NS* on page 3-143 to *IDM_PD_ACCESS_CONTROL_NS* on page 3-148.
Added a section on PMNI_INTERFACEID to configure APB interfaces 0-3.	*PMNI_INTERFACEID, Configure APB interface IDs 0-3* on page 3-284
Added a section on PMNI_INTERFACEID to configure APB interfaces 4-7.	*PMNI_INTERFACEID, Configure APB interface IDs 4-7* on page 3-285
Added a section on PMNI_INTERFACEID to configure APB interfaces 8-11.	*PMNI_INTERFACEID, Configure APB interface IDs 8-11* on page 3-286
Added a section on PMNI_INTERFACEID to configure APB interfaces 12-15.	*PMNI_INTERFACEID, Configure APB interface IDs 12-15* on page 3-286
Added additional information on Secure Exempt for HSNI performance events 0x25, 0x2A and 0x2B.	*4.6 AHB Slave Network Interface performance events* on page 4-305
Updated the APB Master Network Interface performance events (0x03, 0x0A, 0x0B, 0x20 and 0x20).	*4.9 APB Master Network Interface performance events* on page 4-312
Added additional information on Secure Exempt for HMNI performance events 0x22 and 0x23.	*4.7 AHB Master Network Interface performance events* on page 4-308

Table B-5 Differences between issue 0000-03 and issue 0000-04 (continued)

Change	Location
Updated the APB security configuration option.	*2.8.4 Security access permissions of APB requests* on page 2-91
Updated the HSNI_NODE_TYPE register to include secure_transfers field description, values, location in bit assignment diagram and hsni_type note removed.	*HSNI_NODE_TYPE, Node type register for HSNI registers* on page 3-215
Updated the HSNI_CTRL register to include secure_ctrl field description, values and location in bit assignment diagram.	*HSNI_CTRL, HSNI control register* on page 3-223
Updated the HMNI_NODE_INFO register to include secure_transfers field description, values, location in bit assignment diagram and updated hmni_type field description.	*HMNI_NODE_INFO, Node information for HMNI register* on page 3-238
Updated the HMNI_CTRL register to include secure_ctrl field description, values and location in bit assignment diagram.	*HMNI_CTRL, HMNI control register* on page 3-241
Updated the HMNI_INTERRUPT_STATUS register to remove bit 1 from the bit assignment diagram and update bit 0 name and description.	*HMNI_INTERRUPT_STATUS, Interrupt status register* on page 3-247
Updated the HMNI_INTERRUPT_MASK register to remove bit 1 from the bit assignment diagram and update bit 0 name and description.	*HMNI_INTERRUPT_MASK, Interrupt mask register* on page 3-248
	HMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register on page 3-249
Updated the HMNI_INTERRUPT_MASK_NS register to remove bit 1 from the bit assignment diagram and update bit 0 name and description.Updated the HMNI_INTERRUPT_STATUS_NS register to remove bit 1 from the bit assignment diagram and update bit 0 name and description.	*HMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register* on page 3-249
Updated the PMNI_NODE_INFO register to include the no_of_enabled_apb_interfaces field.	*PMNI_NODE_INFO, Node information for PMNI register* on page 3-281
Added a section on PMNI_SECURE_INFO register to view security attributes of the APB interfaces downstream of the PMNI.	*PMNI_SECR_ACC, Secure access register* on page 3-282
Updated the PMNI_CTRL register to include secure_transfers field, values, description and location within the bit assignment diagram.	*PMNI_CTRL, PMNI control register* on page 3-289
Updated the APB master request signals to delete the Protection Types table.	*A.3.1 APB master request signals* on page Appx-A-331
Updated the Power and clock control AWAKEUP signals to identify the protocol as AXI.	*A.4 Power, clock, reset, and IDM control signals* on page Appx-A-333
Updated the **DFTRSTDISABLE** and **DFTCGEN** signal names in Design For Test (DFT) signals.	*A.5 Design for Test signals* on page Appx-A-336

Table B-6 Differences between issue 0000-04 and issue 0000-05

Change	Location
Changed protocols to packets for AXI-H section.	*1.1 About the CoreLink NI-700 Network-on-Chip Interconnect* on page 1-12
Updated the unsupported AMBA features.	*1.2 Key features* on page 1-14
Revised the amount of configurable devices.	*1.6 Configurable options* on page 1-19
Loopback_Signals option updated for ASNI.	*1.6.1 ASNI configuration options* on page 1-20
Loopback_Signals option updated for AMNI.	*1.6.2 AMNI configuration options* on page 1-22
Content on clock disable pin added.	*1.7 Test features* on page 1-29
Updated the HMNI_INTERRUPT_STATUS_NS register toNew note added on shareable exclusive transactions.	*2.1.3 AHB Slave Network Interface* on page 2-35
Updated supported protocols, removed APB2.	*2.1.5 APB Master Network Interface* on page 2-37
Moved content on Address decode and mapping to Functional description section.	*2.4 Address decode and mapping* on page 2-61
Updated Address striping section.	*2.4.4 Address striping* on page 2-61
Repositioned Interconnect Device Management content within Functional description section.	*2.5 Interconnect Device Management* on page 2-66
New example case for master NI IDM block which fails to accept read data beat and write response.	*2.5.2 Timeout detection through IDM block* on page 2-68
Added new content on IDM error logging interrupts and status flags.	*2.6.1 IDM error logging interrupts and status flags* on page 2-81
Added new section on Error Handling and interrupt security.	*2.6.7 Error handling and interrupt security* on page 2-84
New section added on Master Network Interface error responses.	*2.7 Master network interface error responses* on page 2-86
Updated the AHB security access permissions.	*2.8.3 Security access permissions of AHB requests* on page 2-90
Added text and cross-referencing to Register security attribute and security classification and Secure register access.	*2.8.3 Security access permissions of AHB requests* on page 2-90
Added new content and table to Quality of Service. Moved it to the Functional description section.	*2.11 Quality of Service* on page 2-96
New section added on AHB locked transfers.	*2.12 AHB locked transfers* on page 2-105
Added section on Exclusive and locked accesses.	*2.13.3 Exclusive and locked accesses* on page 2-108
Updated the User signals content.	*2.13.8 User signals* on page 2-112
Updated About the programmers model section.	*3.1 About the programmers model* on page 3-116
Updated the Reset value for the NODE_TYPE Global register.	*3.2.1 Global registers summary* on page 3-118
Updated the Reset value for the NODE_TYPE Voltage domain register.	*3.3 Voltage domain registers* on page 3-129
Updated the Reset value for the NODE_TYPE Power domain register.	*3.4 Power domain registers* on page 3-132
Updated the Reset value for the NODE_TYPE Clock domain register.	*3.5 Clock domain registers* on page 3-150

Table B-6 Differences between issue 0000-04 and issue 0000-05 (continued)

Change	Location
SECR_ACC reset value changed to 00 for Global registers, Voltage domain registers, Power domain registers and Clock domain registers.	*SECR_ACC, Secure access register* on page 3-120, *SECR_ACC, Secure access register* on page 3-131, *SECR_ACC, Secure access register* on page 3-148, *SECR_ACC, Secure access register* on page 3-152
Performance monitor configuration register PMCFGR changed to RO in registers summary.	*3.6.1 Performance Monitoring Unit registers summary* on page 3-153
Performance monitor control register PMCR updated to reflect Write Only and RW bits.	*PMCR, Performance monitors control register* on page 3-169
Updated ASNI registers summary to remove repeated occurrence of ASNI_NODE_INFO and include ASNI Interface Ids 0:15.	*3.7.1 ASNI registers summary* on page 3-170
Added the mpam_input_present bit to the bit assignment diagram.	*ASNI_NODE_INFO, Node information for ASNI register* on page 3-172
Added section on ASNI Interface IDs 0-3.	*ASNI_INTERFACEID, Configure ASNI interface IDs 0-3* on page 3-177
Added section on ASNI Interface IDs 4-7.	*ASNI_INTERFACEID, Configure ASNI interface IDs 4-7* on page 3-177
Added section on ASNI Interface IDs 8-11.	*ASNI_INTERFACEID, Configure ASNI interface IDs 8-11* on page 3-178
Added section on ASNI Interface IDs 12-15.	*ASNI_INTERFACEID, Configure ASNI interface IDs 12-15* on page 3-179
Added 'Type' column to ASNI_BURSPLT bit assignment table.	*ASNI_BURSPLT, Burst split control register* on page 3-180
Added 'Type' column to ASNI_SILDBG bit assignment table.	*ASNI_SILDBG, ASNI silicon debug monitor register* on page 3-182
Updated the ASNI_ARQOSOVR, Read channel description and the arqos_value bit description.	*ASNI_ARQOSOVR, Read channel QoS value override register* on page 3-184
Updated the ASNI_AWQOSOVR, Write channel description and the awqos_value bit description.	*ASNI_AWQOSOVR, Write channel QoS value override register* on page 3-185
Updated AMNI registers summary to include AMNI Interface Ids 0:15.	*3.8.1 AMNI registers summary* on page 3-199
Added consent required to modify AMNI_QOSACC, QoS Accept Control for AMNI.	*AMNI_QOSACC, QoS accept control* on page 3-209
Updated AMNI registers summary to reflect AMNI_SILDBG register as RW/RO.	*3.8.2 Register descriptions* on page 3-200
Added section on AMNI Interface IDs 0-3.	*AMNI_INTERFACEID, Configure AMNI interface IDs 0-3* on page 3-204
Added section on AMNI Interface IDs 4-7.	*AMNI_INTERFACEID, Configure AMNI interface IDs 4-7* on page 3-205
Added section on AMNI Interface IDs 8-11.	*AMNI_INTERFACEID, Configure AMNI interface IDs 8-11* on page 3-206
Added section on AMNI Interface IDs 12-15.	*AMNI_INTERFACEID, Configure AMNI interface IDs 12-15* on page 3-206

Table B-6 Differences between issue 0000-04 and issue 0000-05 (continued)

Change	Location
Updated HSNI registers summary to include HSNI Interface Ids 0:15.	*3.9.1 HSNI registers summary* on page 3-214
Updated HSNI registers summary to reflect HSNNI_SILDBG registers as RW/RO.	*3.9.1 HSNI registers summary* on page 3-214
Added section on HSNI Interface IDs 0-3.	*HSNI_INTERFACEID, Configure HSNI interface IDs 0-3* on page 3-220
Added section on HSNI Interface IDs 4-7.	*HSNI_INTERFACEID, Configure HSNI interface IDs 4-7* on page 3-221
Added section on HSNI Interface IDs 8-11.	*HSNI_INTERFACEID, Configure HSNI interface IDs 8-11* on page 3-221
Added section on HSNI Interface IDs 12-15.	*HSNI_INTERFACEID, Configure HSNI interface IDs 12-15* on page 3-222
Updated HMNI registers summary to reflect HSNI_CTRL and HMNNI_SILDBG registers as RW/RO.	*3.10.1 HMNI registers summary* on page 3-237
Updated HMNI registers summary to include HMNI Interface Ids 0:15.	*3.10.2 Register descriptions* on page 3-238
Added section on HMNI Interface IDs 0-3.	*HMNI_INTERFACEID, Configure HMNI interface IDs 0-3* on page 3-243
Added section on HMNI Interface IDs 4-7.	*HMNI_INTERFACEID, Configure HMNI interface IDs 4-7* on page 3-244
Added section on HMNI Interface IDs 8-11.	*HMNI_INTERFACEID, Configure HMNI interface IDs 8-11* on page 3-245
Added section on HMNI Interface IDs 12-15.	*HMNI_INTERFACEID, Configure HMNI interface IDs 12-15* on page 3-246
Updated Network Interface IDM registers summary: IDM_ERRSTATUS, IDM_ERRSTATUS_NS, IDM_RESET_WRITEID and IDM_RESET_WRITEID_NS type of access changed to RO.	*3.11.1 Network Interface IDM registers summary* on page 3-251
Updated the bit descriptions for IDM_ERRCTL.	*IDM_ERRCTLR* on page 3-254
Updated IDM_ERRSTATUS and IDM_ERRSTATUS_NS to reflect Reserved fields as RO and SERR field as RO	*IDM_ERRSTATUS* on page 3-255, *IDM_ERRSTATUS_NS* on page 3-269
Notes added to **isolate** bit description.	*IDM_ACCESS_CONTROL* on page 3-259
Bit descriptions updated for IDM_ACCESS_STATUS	*IDM_ACCESS_STATUS* on page 3-260
Added new notes to the IDM_RESET_CONTROL descriptions.	*IDM_RESET_CONTROL* on page 3-262
Updated IDM_RESET_STATUS bit descriptions.	*IDM_RESET_STATUS* on page 3-263
Updated PMNI register summary to show PMNI_SILDBG register as RW/RO.	*3.12.1 PMNI registers summary* on page 3-280
Updated AMNI note on Secure Events and table heading, Secure exempt, replaced with Secure Only and relevant bookmarks added.	*4.2 AXI Master Network Interface performance events* on page 4-298
ASNI table heading, Secure exempt, replaced with Secure Only and relevant bookmarks added.	*4.3 AXI Slave Network Interface performance events* on page 4-300

Table B-6 Differences between issue 0000-04 and issue 0000-05 (continued)

Change	Location
HSNI table heading, Secure exempt, replaced with Secure Only and relevant bookmarks added.	*4.6 AHB Slave Network Interface performance events* on page 4-305
HMNI table heading, Secure exempt, replaced with Secure Only and relevant bookmarks added.	*4.7 AHB Master Network Interface performance events* on page 4-308
Table heading, Secure exempt, replaced with Secure Only.	*4.9 APB Master Network Interface performance events* on page 4-312
Updated AHB slave request signals.	*A.2.1 AHB slave request signals* on page Appx-A-327
Updated Power and clock control signals.	*A.4 Power, clock, reset, and IDM control signals* on page Appx-A-333
Updated **DFT<CLKNAME>CLKDISABLE** description.	*A.5 Design for Test signals* on page Appx-A-336

Table B-7 Differences between issue 0000-05 and issue 0001-01

Change	Location
Virtual Channel (VC) replaced with *Resource Plane* (RP)	Throughout
Updated privileged and unprivileged accesses and data instructions and accesses in AXI and AHB unsupported protocols	*1.2 Key features* on page 1-14
Updated ASNI configuration options: Updated the maximum number of slave NIs for ASNIs and HSNIs and the maximum number of master NIs for AMNIs, HMNIs, and PMNIs. Updated the write acceptance capability from 1-64 transactions to 1-256 transactions. Updated the read acceptance capability from 1-64 transactions to 1-256 transactions. Updated the read reorder depth from 1-32 to 1-64.	*1.6.1 ASNI configuration options* on page 1-20
Updated the AMNI configuration options read and write issuing capability from 1-64 transactions to 1-256	*1.6.2 AMNI configuration options* on page 1-22
Defined input signals for AHB mirrored master interface and AHB slave interface	*2.1.3 AHB Slave Network Interface* on page 2-35
Removed information on the AMNI output signal **AWAKEUP** in clock domain wakeup content	*Clock domain wakeup* on page 2-44
Added a note to explain what happens when IDM and Read Data Chunking features are enabled together	*2.5 Interconnect Device Management* on page 2-66
Added new use case examples for master and slave interface soft resets	*2.5.6 Soft reset use case examples for master and slave interfaces* on page 2-71
Added new content on the write response buffer	*Write response buffer* on page 2-110
Updated title to (Read) reorder buffer, added new content on read reorder buffer allocation and merging partial read responses	*Read reorder buffer* on page 2-110
Added new content on AXI non-modifiable transactions	*Single Slave per ID* on page 2-110
Added new section on Ordered Write Observation (OWO)	*Ordered Write Observation* on page 2-111

Table B-7 Differences between issue 0000-05 and issue 0001-01 (continued)

Change	Location
Added a new note on per transaction User bits to the user signals content	*2.13.8 User signals* on page 2-112
Updated global register PERIPHERAL_ID2 product_version bit to identify EAC r1p0 and DEV r0p1 product versions	*PERIPHERAL_ID2* on page 3-124
Updated ASNI registers summary to reflect changes in width to 10 for the registers ASNI_ATQOSOT, ASNI_ARQOSOT, ASNI_AWQOSOT, and ASNI_AXQOSOT	*3.7.1 ASNI registers summary* on page 3-170
Updated the ASNI_SILDBG register bit assignments	*ASNI_SILDBG, ASNI silicon debug monitor register* on page 3-182
Updated the max_atomic_ots bit assignment to [9:0] in the ASNI_ATQOSOT register	*ASNI_ATQOSOT, Maximum atomic Outstanding Transactions register* on page 3-186
Updated the max_read_ots bit assignment to [9:0] in the ASNI_ARQOSOT register	*ASNI_ARQOSOT, Maximum read Outstanding Transactions register* on page 3-187
Updated the max_write_ots bit assignment to [9:0] in the ASNI_AWQOSOT register	*ASNI_AWQOSOT, Maximum write Outstanding Transactions register* on page 3-187
Updated the max_ar_aw_ots bit assignment to [9:0] in the ASNI_AXQOSOT register	*ASNI_AXQOSOT, Maximum combined Outstanding Transactions register* on page 3-188
Updated the AMNI_SILDBG register bit assignments	*AMNI_SILDBG, Silicon debug monitor register* on page 3-207
Added a new note to state NI-700 does not permit a value combination of 1 for bit [1] and 0 for bit [0]	*AMNI_QOSACC, QoS accept control* on page 3-209
Updated descriptions for the AHB Slave Network Interface performance events read request stall on 0x0E and write request stall on 0x10	*4.6 AHB Slave Network Interface performance events* on page 4-305
Updated descriptions for several AHB Master Network Interface performance events: Read request Stall: (**HTRANS** &!**HREADY**) on `0x0E`, Read request Stall: **HREADY_IN**= 0 when **HREADY** = 1 on `0x0F` and Write request Stall: (**HTRANS** &!**HREADY**) on `0x10`	*4.7 AHB Master Network Interface performance events* on page 4-308
<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_ AWID[n:0] width is not configurable	*A.1.7 AXI master signals on the write address channel* on page Appx-A-319
Added new signal name and description for **<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_ WID[n:0]**	*A.1.8 AXI master signals on the write data channel* on page Appx-A-321
Added signal widths to read address channel master interface signals	*A.1.10 AXI master signals on the read address channel* on page Appx-A-323

Table B-7 Differences between issue 0000-05 and issue 0001-01 (continued)

Change	Location
Updated the **<PDOMAIN>_INTERRUPT** and **<PDOMAIN>_NS_INTERRUPT** interrupt signal descriptions to state these interrupts are rising edge triggered	*A.4 Power, clock, reset, and IDM control signals* on page Appx-A-333
Updated the signal directions and descriptions for **<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_SRESETN** and **<PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>_SRESETN**	*A.4 Power, clock, reset, and IDM control signals* on page Appx-A-333
Updated the **<CLKNAME>_nPMUINTERRUPT** signal description to state this interrupt is rising edge triggered	*A.6 PMU and debug signals* on page Appx-A-337

Table B-8 Differences between issue 0001-01 and issue 0100-01

Change	Location
Replaced references to Booker-NCI with new product name NI-700	Throughout
Renamed top level interface diagram title to NI-700	*1.4 Interfaces* on page 1-16
Removed content on limitations of the r0p1 DEV release	
• Updated the supported number of master NIs (AMNIs, HMNIs, and PMNIs) to 127 and the supported number of slave NIs (ASNIs and HSNIs) changed to 128 • Added new content on cache line size	*1.6 Configurable options* on page 1-19
Updated ASNI configuration options: • User sideband signal width of 0-64 bits removed and crossreference added to User signals • Loopback_Signals None changed to Loopback_Signals OPTIONAL within topic table • Read_Interleaving_Disabled Not supported changed to Read_Interleaving_Disabled Must always be set to FALSE • Prefetch_Transaction OPTIONAL moved to ACE-Lite section of table, new content added on minimum atomic acceptance • The permitted values for read reorder depth to, 1, 2, and multiples of 4 including 64	*1.6.1 ASNI configuration options* on page 1-20
Updated AMNI configuration options: • User sideband signal width of 0-64 bits removed and crossreference added to User signals • AXI ID width changed from 1-24 bits to 1-32 bits • Loopback_Signals_None changed to Loopback_Signals OPTIONAL within topic table • Prefetch_Transaction OPTIONAL moved to ACE-Lite section of table New content added on minimum atomic issue	*1.6.2 AMNI configuration options* on page 1-22
Updated HSNI configuration option User sideband signal width of 0-64 bits removed and crossreference added to User signals	*1.6.3 HSNI configuration options* on page 1-25
Updated HMNI configuration option User sideband signal width of 0-64 bits removed and crossreference added to User signals	*1.6.4 HMNI configuration options* on page 1-27
Updated the HSNI clock gating buffer diagram to remove a text reference to NCI and replace it with NI-700	*AHB address phase buffering in HSNI* on page 2-49
Updated Access mechanism diagram to remove text reference to Non-coherent interconnect (NCI), and replace with NI-700	*2.3.2 Node configuration register address-mapping overview* on page 2-54

Table B-8 Differences between issue 0001-01 and issue 0100-01 (continued)

Change	Location
Updated content: • Deleted bullet point: All stripe groups in a memory map that an AXI or AHB slave network interface subscribes to, must have the same number of stripe targets • Added new text regarding the responsibility of the SOC integrator and system builder to setup the address maps and stripe groups consistently • Added new content on address map restrictions and changed several notes to bullets • Added new content on a stripe group with a single target interface	*2.4.4 Address striping* on page 2-61
Updated content: • Replaced the word slave (interface) with target within the text, updated all remap diagrams with target instead of slave • Added a new note to the end of the topic on maintaining access to the programmers model and Config target when remapping occurs	*2.4.5 Remap* on page 2-62
Removed content on IDM unsupported AMBA 5 features and reworded existing content	*2.5 Interconnect Device Management* on page 2-66
Added content when IDM detects a timeout, software must trigger a soft reset before resuming normal operation	*2.5.2 Timeout detection through IDM block* on page 2-68
Added content on how NI-700 handles outstanding requests and soft reset requests	*2.5.4 IDM soft reset* on page 2-68
Added content on how NI-700 handles an isolation request and the difference between isolation and soft reset	*2.5.5 IDM access control* on page 2-70
Added use cases for the soft-reset functionality for master and slave interfaces	*2.5.6 Soft reset use case examples for master and slave interfaces* on page 2-71
Added a use case for the access control functionality for a write transaction at a slave interface	*2.5.7 Master and slave interface access control use case examples* on page 2-75
Added new content to demonstrate an interrupt handling sequence	*2.5.8 Sample interrupt handling sequence* on page 2-77
Added an example to show a fast sequence for placing a slave slave device into soft-reset	*2.5.9 Soft reset sequence* on page 2-78
Updated existing content: • Added new content on CMO transactions on the write channel, Write + CMO transactions on the Write Channel • Updated the Request types table	*2.7 Master network interface error responses* on page 2-86
Added new content on memory tagging support and relevant behavior in the NI-700	*2.10 Memory tagging support* on page 2-95
Added new content on how to calculate TSPEC parameters for traffic	*How to calculate TSPEC parameters for traffic* on page 2-98
Added examples on how to calculate TSPEC parameters for traffic	*TSPEC parameter examples* on page 2-98
Updated the content on programming the TSPEC parameters to include the ASNI registers ASNI_QOSCOMPK, ASNI_QOSCOMBUR, and ASNI_QOSCOMAVG for combined Read and Write mode	*Program the TSPEC parameters* on page 2-102

Table B-8 Differences between issue 0001-01 and issue 0100-01 (continued)

Change	Location
Added new content on: • BQV control register settings • BQV register settings • Added a new image which shows excess transfers over the average rate and burstiness allowance	*2.11.2 Soft bandwidth regulation using Bandwidth QoS Value* on page 2-102
Updated references to the AXI specification from issue G to H and removed Issue F reference from `USER_DATA_ MODE` = 0 in User signals	*2.13.8 User signals* on page 2-112
Added new content on requirements for configuration register reads and writes	*3.1.1 Requirements of configuration register reads and writes* on page 3-116
Updated the note in ASNI_BURSPLT register to change bits from [2:0] to [3:0] and edited the note text to 'it is unpredictable when the new burst split control values take effect'	*ASNI_BURSPLT, Burst split control register* on page 3-180
Updated secure_ctrl register bits from [1:0] to [0]	*HMNI_CTRL, HMNI control register* on page 3-241
Updated IDM register summary to include two new registers IDM_ACCESS_STATUS_NS and IDM_RESET_STATUS_NS	*3.11.1 Network Interface IDM registers summary* on page 3-251
Added new register IDM_ACCESS_STATUS_NS	*IDM_ACCESS_STATUS_NS* on page 3-273
Added new register IDM_RESET_STATUS_NS	*IDM_RESET_STATUS_NS* on page 3-275
Removed **<PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>** and **<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>** from before AXI signal names. These were replaced with a prefix.	All AXI signals
Updated content: • Changed AWSNOOP value for slave write address channel ACE-Lite specific signals from [2:0] to [3:0] • Removed signal name **<PROTOCOL>_ SLAVE_<ENDPOINT_INTERFACE _NAME>_ AWSNOOP[3]** from top row of table Write address channel AX15 extension and ACE-Lite signals • Removed table on AWSNOOP Encodings • Added new signal name **<PROTOCOL>_ SLAVE_<ENDPOINT_INTERFACE _NAME>_ AWTAGOP** to the Write address channel AXI5 extension and ACE-Lite signals table	*A.1.1 AXI slave signals on the write address channel* on page Appx-A-314
Added two new signal names to the Write data channel AXI5 extension and ACE-Lite signals table: • **<PROTOCOL>_ SLAVE_<ENDPOINT_INTERFACE _NAME>_WTAG** • **<PROTOCOL>_ SLAVE_<ENDPOINT_INTERFACE _NAME>_WTAGUPDATE** Updated all signal names with a reference to <prefix> beneath the table titles	*A.1.2 AXI slave signals on the write data channel* on page Appx-A-315
Added a new signal name **<PROTOCOL>_ SLAVE_<ENDPOINT_INTERFACE _NAME>_BTAGMATCH** and description to the table Write address channel AXI5 extension and ACE-Lite signals	*A.1.3 AXI slave signals on the write response channel* on page Appx-A-316

Table B-8 Differences between issue 0001-01 and issue 0100-01 (continued)

Change	Location
Updated content: • Added a new signal name **<PROTOCOL>_ SLAVE_<ENDPOINT_INTERFACE_NAME>_ARTAGOP** to the table Write address channel AXI5 extension and ACE-Lite signals • Updated all signal names with a reference to <prefix> beneath the table titles • Changed all signal directions to inputs, except **<PROTOCOL>_ SLAVE_<ENDPOINT_INTERFACE _NAME>_ ARREADY** • Value descriptions for ARTAGOP in Read address channel AXI5 extension and ACE-Lite signals, updated to Transfer and Fetch	*A.1.4 AXI slave signals on the read address channel* on page Appx-A-317
Updated content: • Added a new signal name **<PROTOCOL>_ SLAVE_<ENDPOINT_INTERFACE_NAME>_RTAG** • Updated all signal names with a reference to <prefix> beneath the table titles	*A.1.5 AXI slave signals on the read data channel* on page Appx-A-318
Updated content: • Removed <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME> from the beginning of each AXI signal name • Added cross reference to *2.11.3 QoS override programmable registers* on page 2-104 for the **QOSOVERRIDE** signal • Added cross reference to *Ordered Write Observation* on page 2-111 for the **ORDERED_WRITE_OBSERVATION** signal	*A.1.6 Other AXI signals* on page Appx-A-319
Updated content: • Changed `AWSNOOP` value for master write address channel ACE-Lite specific signals from [2:0] to [3:0] • Removed signal name **<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE _NAME>_AWSNOOP[3]** from top row of table Write address channel AX15 extension and ACE-Lite signals • Added new signal **<PROTOCOL>_ MASTER_<ENDPOINT_INTERFACE _NAME>_ AWTAGOP**	*A.1.7 AXI master signals on the write address channel* on page Appx-A-319
Added two new signal names to the Write data channel AXI5 extension and ACE-Lite signals table: • **<PROTOCOL>_ MASTER_<ENDPOINT_INTERFACE _NAME>_WTAG** • **<PROTOCOL>_ MASTER_<ENDPOINT_INTERFACE _NAME>_WTAGUPDATE** Updated all signal names with a reference to <prefix> beneath the table titles	*A.1.8 AXI master signals on the write data channel* on page Appx-A-321
Added a new signal name **<PROTOCOL>_ MASTER_<ENDPOINT_INTERFACE _NAME>_BTAGMATCH** and description to the table write response channel AXI5 extension and ACE-Lite signals	*A.1.9 AXI master signals on the write response channel* on page Appx-A-322
Updated signal direction for **<PROTOCOL>_ MASTER_<ENDPOINT_INTERFACE_NAME>_ARREADY** to an input in read address channel master interface signals table	*A.1.10 AXI master signals on the read address channel* on page Appx-A-323
Added new signal **<PROTOCOL>_ MASTER_<ENDPOINT_INTERFACE _NAME>_ARTAGOP** to master read address channel AX15 extension and ACE-Lite signals table	*A.1.10 AXI master signals on the read address channel* on page Appx-A-323

Table B-8 Differences between issue 0001-01 and issue 0100-01 (continued)

Change	Location
Added new signal **<PROTOCOL>_ MASTER_<ENDPOINT_INTERFACE _NAME>_RTAG** to read data channel AX15 extension and ACE-Lite signals table. All signals in this table changed to inputs.	*A.1.11 AXI master signals on the read data channel* on page Appx-A-324
Removed **<PROTOCOL>_<MASTER_>ENDPOINT_INTERFACE_NAME>_PWAKEUP>** signal from power and clock control signals	*A.4 Power, clock, reset, and IDM control signals* on page Appx-A-333

Table B-9 Differences between issue 0100-01 and issue 0200-08

Change	Location
Updated document issue number to current numbering process	
Added content that NI-700 also supports AMBA AXI3 on the master interface connection to downstream slaves.	*1.1 About the CoreLink NI-700 Network-on-Chip Interconnect* on page 1-12
Updated content: • NI-700 supports AXI3 AMBA protocol, but only on NI-700 master interfaces • Removed unsupported features in the AMBA AXI protocol: 1. Privileged and unprivileged accesses (AxPROT[0]) are not transported 2. Data instruction and accesses (AxPROT[2]) are not transported. • Removed unsupported features in the AHB AXI protocol: 1. Data instruction and accesses (HPROT[0]) are not transported 2. Privileged and unprivileged accesses (HPROT[1]) are not transported • Added AXI3 unsupported features to Write data dependencies in the unsupported AMBA features table	*1.2 Key features* on page 1-14
Removed AXI3 from the list of unsupported specifications	*1.3 Compliance* on page 1-15
Removed the sentence, 'The design of NI-700 permits frequencies up to 1GHz on 16nm FinFET compact (16FFC) and 7FF process nodes.' Added a new bullet point that the AXI3 protocol only supports master interfaces	*1.5 Architecture overview* on page 1-17
Added new content: • An AMNI can have AXI3 as the master interface type • User signals are applicable to all AMNI interface types including AXI3 • Added reference to the AMBA AXI specification on transaction and interface constraints for ACE5-Lite ACP • Updated the support type Supported to Optional in the table Features that the AMNI supports for a specific interface type	*1.6.2 AMNI configuration options* on page 1-22
Added a new section on support for AXI3 interface types and updated content on write data dependency constraints	*2.1.2 AXI Master Network Interface* on page 2-34
Removed content on Resets. This content is now in the confidential document, *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Configuration and Integration Manual*.	
Removed content on System-level reset. This content is now in the confidential document, *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Configuration and Integration Manual*.	

Table B-9 Differences between issue 0100-01 and issue 0200-08 (continued)

Change	Location
Updated references to diagram and power control network diagram	*2.2.3 Power control* on page 2-45
Updated HSNI clock gating buffer block diagram to show **<CLKNAME>_CLK** and **<CLKNAME>_RESETn** signal direction	*AHB address phase buffering in HSNI* on page 2-49
Added support for address stripe granules, in bytes, 128B, 256B, 512B, 1024B, 2048B, and 4096B	*2.4.4 Address striping* on page 2-61
Updated content to describe what happens once IDM detects a timeout and the mode the interface enters	*2.5.2 Timeout detection through IDM block* on page 2-68
Removed content on when the IDM block detects a bus error on its interface and the software uses the IDM soft reset functionality	*2.5.3 Error logging through IDM block* on page 2-68
Added new sections: • Hardware initiated entry based on timeout detection • Software initiated entry • IDM_RESET_CONTROL reset initialization input pin	*2.5.4 IDM soft reset* on page 2-68
Added new content on master NI isolation and how a master handles requests and transactions	*2.5.5 IDM access control* on page 2-70
Removed the note on hardware must receive a soft reset request to resume normal operation and updated all diagrams and relevant content	*2.5.6 Soft reset use case examples for master and slave interfaces* on page 2-71
Removed duplicate text	*2.5.8 Sample interrupt handling sequence* on page 2-77
Added Write response dependency violation to the AMNI non-IDM interrupt conditions per endpoint	*2.6.4 Non-IDM interrupts* on page 2-82
Added new content on Memory tagging support (MTE)	*2.10 Memory tagging support* on page 2-95
Added the AMNI_CONFIG_CTL register to the AMNI registers summary.	*3.8.1 AMNI registers summary* on page 3-199
Updated description for bits [3:0] amni_type, to reflect relevant technical specifications	*AMNI_NODE_INFO, Node information for AMNI register* on page 3-200
Added AMNI_CONFIG_CTL register and updated its configuration constraints	*AMNI_CONFIG_CTL, Select response* on page 3-209
Updated bit descriptions and bit numbers to reflect changes to the address and data phases of AHB. Bit 4 is now reserved as there is not a separate response channel.	*HSNI_SILDBG, HSNI silicon debug monitor register* on page 3-226

Table B-9 Differences between issue 0100-01 and issue 0200-08 (continued)

Change	Location
Updated bit descriptions and bit numbers to reflect changes to the address and data phases of AHB. Bit 4 is now reserved as there is not a separate response channel.	*HMNI_SILDBG, HMNI Silicon debug monitor register* on page 3-246
Updated HSNI performance events `0x0E` to `0x12` to reflect address and data phases of AHB in the HSNI_SILDBG and HMNI_SILDBG registers	*4.6 AHB Slave Network Interface performance events* on page 4-305
Updated HMNI performance events `0x0E` to `0x12` to reflect address and data phases of AHB in the HSNI_SILDBG and HMNI_SILDBG registers	*4.7 AHB Master Network Interface performance events* on page 4-308
Corrected signal name from **NSAIDW[3:0]** to **AWNSAID[3:0]**	*A.1.1 AXI slave signals on the write address channel* on page Appx-A-314
Added **ARNSAID** signal name, input type and description	*A.1.4 AXI slave signals on the read address channel* on page Appx-A-317
Updated signal name from **NSAIDW[3:0]** to **AWNSAID[3:0]**.	*A.1.7 AXI master signals on the write address channel* on page Appx-A-319
Added signal **ARNSAID[3:0]**	*A.1.10 AXI master signals on the read address channel* on page Appx-A-323
Added new AXI3 master signals	*A.1.12 AXI3 master network interface signals* on page Appx-A-325
Updated the title of the topic. Added two new signals: • **<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_IDM_SRESET_STRAP** • **<PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>_IDM_SRESET_STRAP**	*A.4 Power, clock, reset, and IDM control signals* on page Appx-A-333