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

Revision: r2p3

Technical Reference Manual

Non-Confidential

Issue 10
101566_0203_10_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	Non-Confidential	First DEV release for r0p0
0000-01	28 June 2019	Non-Confidential	Second DEV release for r0p0
0000-02	5 August 2019	Non-Confidential	First ALP release for r0p0
0000-03	30 September 2019	Non-Confidential	Third DEV release for r0p0
0000-04	5 November 2019	Non-Confidential	First BET release for r0p0
0000-05	27 March 2020	Non-Confidential	First LAC release for r0p0
0001-01	17 July 2020	Non-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
0201-09	1 September 2021	Non-Confidential	First REL release for r2p1
0203-10	21 July 2023	Non-Confidential	First REL release for r2p3

Proprietary Notice

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349|version 21.0)

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.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on https://support.developer.arm.com

To provide feedback on the document, fill the following survey: https://developer.arm.com/documentation-feedback-survey.

Inclusive language commitment

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

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

To report offensive language in this document, email terms@arm.com.

Contents

1. Introduction

1.1 Product revision status

The `rxpy` identifier indicates the revision status of the product described in this manual, for example, `r1p2`, 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.

1.2 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.

1.3 Conventions

The following subsections describe conventions used in Arm documents.

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: developer.arm.com/glossary.

Typographic conventions

Arm documentation uses typographical conventions to convey specific meaning.

Convention	Use
italic	Citations.
bold	Terms in descriptive lists, where appropriate.
`monospace`	Text that you can enter at the keyboard, such as commands, file and program names, and source code.
`monospace underline`	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

Convention	Use
<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	Terms that have specific technical meanings as defined in the *Arm® Glossary*. For example, **IMPLEMENTATION DEFINED**, **IMPLEMENTATION SPECIFIC**, **UNKNOWN**, and **UNPREDICTABLE**.

Caution

Recommendations. Not following these recommendations might lead to system failure or damage.

Warning

Requirements for the system. Not following these requirements might result in system failure or damage.

Danger

Requirements for the system. Not following these requirements will result in system failure or damage.

Note

An important piece of information that needs your attention.

Tip

A useful tip that might make it easier, better or faster to perform a task.

Remember

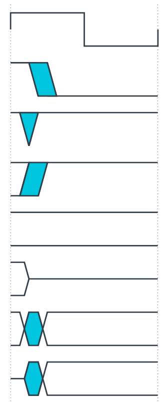
A reminder of something important that relates to the information you are reading.

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 1-1: 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.

1.4 Useful resources

This document contains information that is specific to this product. See the following resources for other useful information.

Access to Arm documents depends on their confidentiality:

- Non-Confidential documents are available at developer.arm.com/documentation. Each document link in the following tables goes to the online version of the document.
- Confidential documents are available to licensees only through the product package.

Arm product resources	Document ID	Confidentiality
Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Configuration and Integration Manual	101567	Confidential
Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Release Note	109241	Confidential
Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Technical Reference Manual	101566	Non-Confidential

Arm architecture and specifications	Document ID	Confidentiality
AMBA® APB Protocol Specification, Version 2.0	IHI 0024C	Non-Confidential
AMBA® AXI and ACE Protocol Specification	IHI 0022H	Non-Confidential
AMBA® Low Power Interface Specification Arm® Q-Channel and P-Channel Interfaces	IHI 0068C	Non-Confidential
Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite	IHI 0033B.b	Non-Confidential
Arm® CoreSight™ Architecture Specification, Version 3.0	IHI 0029E	Non-Confidential
Principles of Arm® Memory Maps White Paper	DEN 0001C	Non-Confidential

Non-Arm resources	Document ID	Organization
JEDEC Standard Manufacturer's Identification Code, JEP106	JEP106	www.jedec.org

Note

Arm tests its PDFs only in Adobe Acrobat and Acrobat Reader. Arm cannot guarantee the quality of its documents when used with any other PDF reader.

Adobe PDF reader products can be downloaded at http://www.adobe.com

2. CoreLink NI-700 Network-on-Chip Interconnect

NI-700 is a highly configurable AMBA-compliant system-level interconnect. With NI-700, you can create a non-coherent interconnect that is optimized to the Power, Performance, and Area (PPA) requirements of your SoC design.

Designed to scale, NI-700 is suitable for large designs as a backplane interconnect. Using multiple routers and various topology options, you can connect multiple upstream and downstream devices that use different AMBA protocols to NI-700.

The features of NI-700 provide flexibility and enable the interconnect to adapt to a wide range of system requirements. For more information, see Key features.

NI-700 supports various AMBA protocols and complies with the relevant specifications. For more information, see Compliance.

Separate NI-700 interfaces receive inputs and send outputs over the different supported protocols. For more information, see Interfaces.

The architecture of NI-700 is designed for high frequency with low latency while also optimizing system bandwidth and PPA. For more information, see Architecture overview.

An NI-700 implementation comprises a network of functional units that process and route traffic. For more information, see Functional units.

Both the overall topology and individual functional units in NI-700 can be configured according to the system requirements. For more information, see Configurable options.

To optimize the performance of a NI-700 implementation, Arm recommends that the steps in the product design flow are completed in order. For more information, see Product design flow.

NI-700 includes documentation that provides detailed information to support each stage of the product design flow. For more information, see Product documentation.

2.1 Key features

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

NI-700 includes the following key features:

- Native support for the following AMBA protocols:
    - AXI5, AXI-G, and AXI-H
    - AHB5
    - APB3 and APB4

- AXI3 on NI-700 AMNIs only.

  For more information, see Compliance.

- Packet transfer over multiple clock, power, and voltage domains. For more information, see Power, clock, and reset management.
- Source-based packet routing. For more information, see Functional units.
- Worm-hole routing with support for multiple Resource Planes (RPs). For more information, see Resource Planes.
- Flit-level credit-based flow control. For more information, see Functional units.
- Quality of Service (QoS) features for prioritization of information transfer. For more information, see Quality of Service.
- Distributed switching mechanism to enable traffic management and protect against network saturation. For more information, see Functional units.
- Variable, configurable topology that is specified through the Socrates IP Tooling platform. For more information, see Configurable options.
- Support for transporting data parity, ECC, or poison information through the interconnect. For more information, see Transporting data parity, ECC, and poison information.
- Support for scan cell insertion as part of the Design for Test strategy. For more information, see Test features.

2.1.1 Test features

NI-700 supports a scan cell insertion methodology for your SoC Design for Test (DFT) strategy. DFT control signals allow you to achieve a very high coverage for your NI-700 test strategy.

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*.

2.2 Compliance

NI-700 complies with various AMBA specifications and standards.

This product is compliant with:

- *AMBA® AXI and ACE Protocol Specification*.

  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*.

This manual must be read in conjuction with the AMBA specifications. Information from these specifications is not reproduced in this document. For more information, see the Additional reading section.

2.2.1 Supported AMBA features

NI-700 supports the AMBA AXI5, ACE5-Lite, AHB5, APB3, and APB4 protocols.

The following specific AXI5 capabilities are supported:

AXI5

- Atomic transactions:

  Transactions that perform more than just a single access and have an associated operation.
- Additional QoS Accept signals:

  Two extra signals that enable a completer to indicate the minimum QoS value of transactions that it accepts.
- Trace signals:

  Signals that can be associated with each channel to support the debugging, tracing, and performance measurement of systems.
- Loopback signals:

  Signals that permit an agent that is issuing transactions to store information that is related to the transaction in an indexed table.
- Wakeup signals:

  Signals that are used to indicate that there is activity that is associated with the interface.
- Non-secure Access Identifiers:

  IDs that control access to particular Non-secure memory locations.

Note

All OPTIONAL AXI5 capabilities, except for wakeup signaling, can be disabled when NI-700 is integrated with an AXI4-based system.

ACE5-Lite

- Cache stashing transactions:

Transactions that enable one component to indicate that a particular cache line must be placed in the cache of another component in the system.

- Deallocating transactions:

  Transactions that are primarily used to deallocate cache lines when they are no longer required.

- Persistent Cache Maintenance Operations (CMOs):

  Operations that are used to ensure that a store operation, potentially held in a Dirty cache line, is moved downstream to persistent memory.

AXI5.G

- Memory System Resource Partitioning and Monitoring (MPAM):

  A technology for partitioning and monitoring memory system resources for physical and virtual machines.

- Unique ID indicator:

  A flag that shows when a request is using an AXI identifier that is unique for in-flight transactions.

- Read data chunking:

  A feature that enables a completer interface to send read data for a transaction in any order using a 128-bit granule.

- CMOs on the write channel:

  Operations that consist of a request on the AW channel and a response on the B channel.

- Read interleaving property:

  A property that indicates whether an interface supports the interleaving of read data beats from different transactions.

Note

ASNIs cannot guarantee that data beats between transactions with different IDs do not interleave. Therefore, the `Read_Interleaving_Disabled` property is always False for ASNIs.

AXI5.H

- Memory Tagging Extensions (MTE):

  A feature that provides a mechanism that can be used to detect memory safety violations.

- Prefetch request:

Requests that enable a requester to signal to the system to prepare a location for reading before making an actual read request.

- Data writes combined with CMOs:

  Operations that allow CMOs to be used in conjunction with a write to memory for improved efficiency.

AXI3

You can configure NI-700 AMNIs with an AXI3 requester interface that connects to completer devices. For supported AXI3 features, see Configurable options.

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

NI-700 supports the following AMBA interfaces:

- AXI5. For supported AXI5 features, see Configurable options.
- AHB5. The AHB5 specification adds a set of OPTIONAL capabilities to AHB-Lite.

  To connect an AHB-Lite requester or completer to NI-700, you must disable the OPTIONAL capabilities on the HMNI or HSNI.
- ACE5-Lite
- APB3 and APB4
- AXI3 on NI-700 AMNIs only

2.2.2 Unsupported AMBA features

NI-700 does not support all AMBA features.

The following AMBA features are not supported:

AXI

- AXI region identifiers (AxREGION signaling).
- Barrier transactions (AxBAR signaling).

AXI3

- NI-700 ASNIs cannot have an AXI3 completer interface. Only NI-700 AMNIs support AXI3 on the requester interface.
- Write data interleaving.
- Locked accesses. NI-700 supports normal and exclusive accesses only.
- Write data dependencies.
  - From AXI4 onwards, the AXI protocol added an extra dependency for write transactions. For NI-700, an AXI3 completer that accepts all write data and provides a write response before accepting the address is not compliant with AXI4 or later. The

AXI specification strongly recommends that any new AXI3 completer 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, the AMNI behavior is **UNPREDICTABLE**. Downstream AXI3 completers must conform to the AXI4 requirement to integrate directly with NI-700.
  - To integrate a downstream AXI3 completer that follows the AXI3 write dependency requirement requires an external wrapper. The external wrapper ensures that a returning write response is not provided until the completer has accepted the appropriate address.

AHB

- Locked transfers. HMASTLOCK indication is ignored at the HSNI.
- Multi-copy atomicity is not supported if early write response is enabled. However, if early write response is disabled, multi-copy atomicity is supported.
- Multiple completer select (HSELx) signaling
- Split and retry

2.2.3 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 requester device through NI-700 to the downstream device. The completer 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.

2.2.3.1 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 upstream devices that have access to the programmer's view, security vulnerabilities can occur. For example:

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

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 completer might block Secure transactions to a different Secure completer.

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 device.

Hierarchical clock gating

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

2.3 Interfaces

NI-700 has both completer and requester interfaces which support various AMBA protocols.

The following definitions apply:

Completer interface

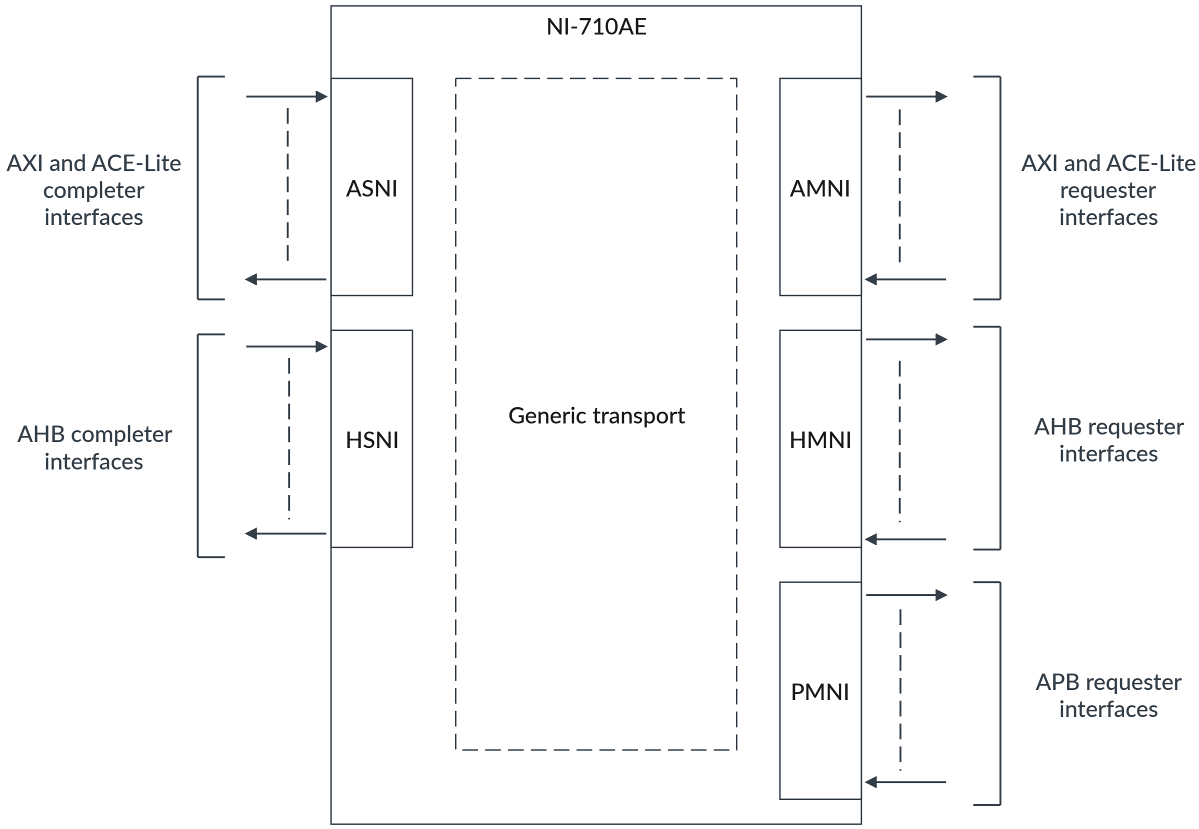
An interface that receives input from a requester device. These interfaces might also be known as downstream interfaces, completer interfaces, or receiver interfaces, depending on the context.

Requester interface

An interface that sends output to a completer device. These interfaces might also be known as upstream interfaces, requester interfaces, or transmitter interfaces, depending on the context.

The following figure shows how AXI requester and completer devices connect to NI-700.

Figure 2-1: NI-700 top-level interfaces



To control the clock and power functions, NI-700 has Low-Power Interfaces (LPIs). These interfaces are not shown in the preceding diagram.

2.4 Architecture overview

The architecture of NI-700 is designed to provide a high frequency, low latency interconnect.

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

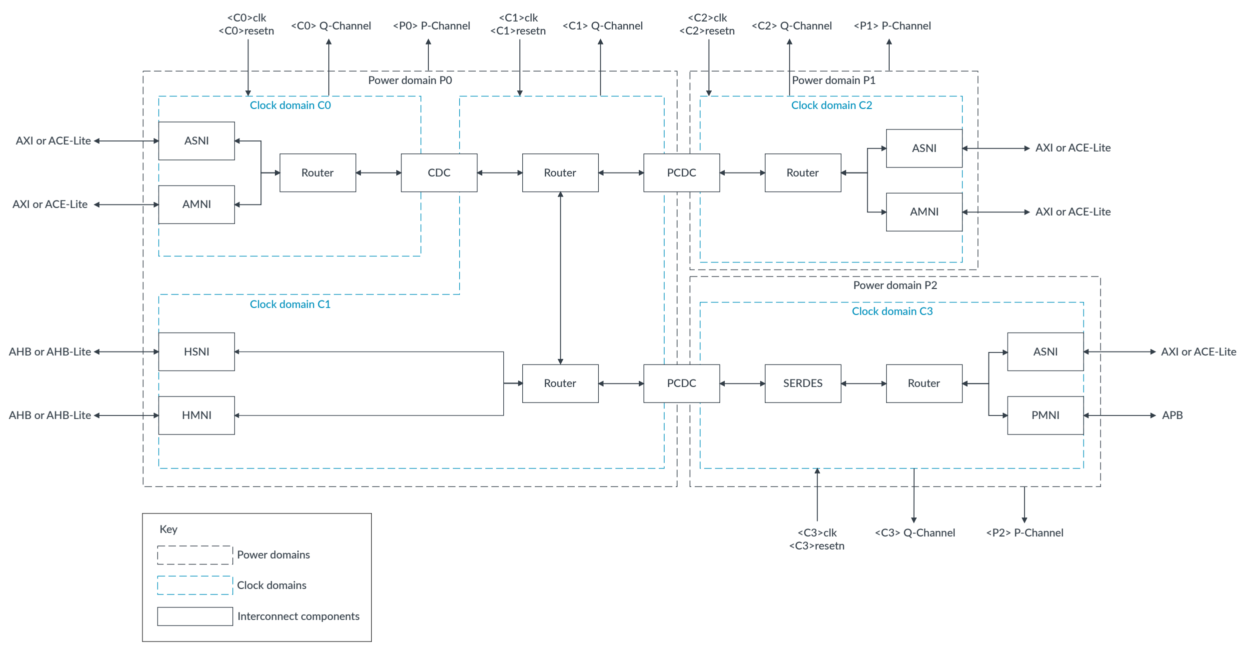
To optimize system bandwidth and PPA, NI-700 provides the following architectural features:

- Multiple requesters and completers with a combination of AXI5, ACE5-Lite, ACE5-LiteACP, AHB5, APB3, and APB4 protocols
- AXI3 protocol on AMNIs only
- Packetizing mechanism that enables configurable link widths from 32-2048 bits
- Independent widths for the defined sideband signals for each channel

- Resource Planes (RPs) to permit traffic isolation
- Non-blocking RPs
- Configurable duplicate links between pairs of router units
- Bandwidth 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 2-2: Top-level architecture example for NI-700



The NI-700 Power and Clock Domain Crossing (PCDC) unit is responsible for bridging between power domains and clock domains. You can configure the PCDC unit to provide only clock domain crossings or both power and clock domain crossings.

2.5 Functional units

NI-700 is constructed from various functional units. Each functional unit has its own transfer function.

You can use the Socrates IP Tooling platform to create network topologies from the functional units.

NI-700 contains the following functional units:

- ASNI
- AMNI
- HSNI
- HMNI
- PMNI
- PCDC
- Routers
- SERDES units
- PMU

The functional units process and route network traffic across the NI-700 network layer by performing the following tasks:

- Converting between AXI transactions and NI-700 Generic Transport (GT) protocol flits
- Converting between AXI, AHB, or APB transactions and NI-700 GT protocol flits
- Routing flits across the network between any completer interface and any requester interface
- Arbitrating flits according to Quality of Service (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

The network interfaces provide the following functions:

- ASNIs convert AXI and ACE-Lite transactions into NI-700 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

All functional units have the following configurable options:

- Number of credits available for each channel
- Flit width for each channel

2.5.1 ASNI

NI-700 ASNI completer units receive and process requests from AXI requester devices.

These units packetize transactions into flits according to the NI-700 Generic Transport (GT) protocol and depacketize GT response flits into AXI responses.

ASNIs perform the following functions:

- Conversion of requests, data, and response transactions between the AXI and GT protocols
- Decoding transaction addresses 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. ASNIs split 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 Quality of Service (QoS) bandwidth regulation
- Timing isolation from the external requester 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 writes

2.5.2 AMNI

NI-700 AMNI units receive and process Generic Transport (GT) packets from the NI-700 network layer.

These units depacketize GT packets, convert them to AXI request transactions, and forward them to connected AXI completer devices. In addition, AMNIs receive AXI responses from completer devices and packetize them into GT response flits.

AMNIs perform the following functions:

- Conversion between network GT requests and AXI transactions
- Route read and write response channel traffic back to request initiators
- Burst splitting of transactions. An AMNI splits bursts if the size of the original transaction is greater than the maximum burst size that the AMNI can issue.
- Data width resizing
- Memory controller bandwidth regulation through VAxQOSACCEPT
- Timing isolation from the external completer 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 writes

Support for AXI3 interface types

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

- When configured as AXI3, an AMNI has a WID pin on the interface that the downstream completer can use to connect to the WID input.
- When configured as AXI3, an AMNI observes the maximum burst length of 16 that AXI3 supports.
- AXI3 locked accesses are not supported and AMNIs do not generate locked accesses.

From AXI4 onwards, the AXI protocol adds an extra dependency for write transactions. An AXI3 completer that accepts all write data, and provides a write response before accepting the address, is not compliant with AXI4. This type of AXI3 completer is not compliant with later versions of the AMBA AXI protocol. The AXI specification strongly recommends that any new AXI3 completer implementation includes this additional dependency.

NI-700 AMNIs conform to the AXI4 write data dependency requirement. Therefore, if an AMNI receives a write response before the write address phase is accepted its behavior is **UNPREDICTABLE**. Downstream AXI3 completers must conform to the AXI4 requirement to integrate directly with NI-700.

To integrate a downstream AXI3 completer that follows the AXI3 write data dependency requirement, an external wrapper is required. The external wrapper ensures a returning write response is not provided until the completer has accepted the appropriate address.

2.5.3 HSNI

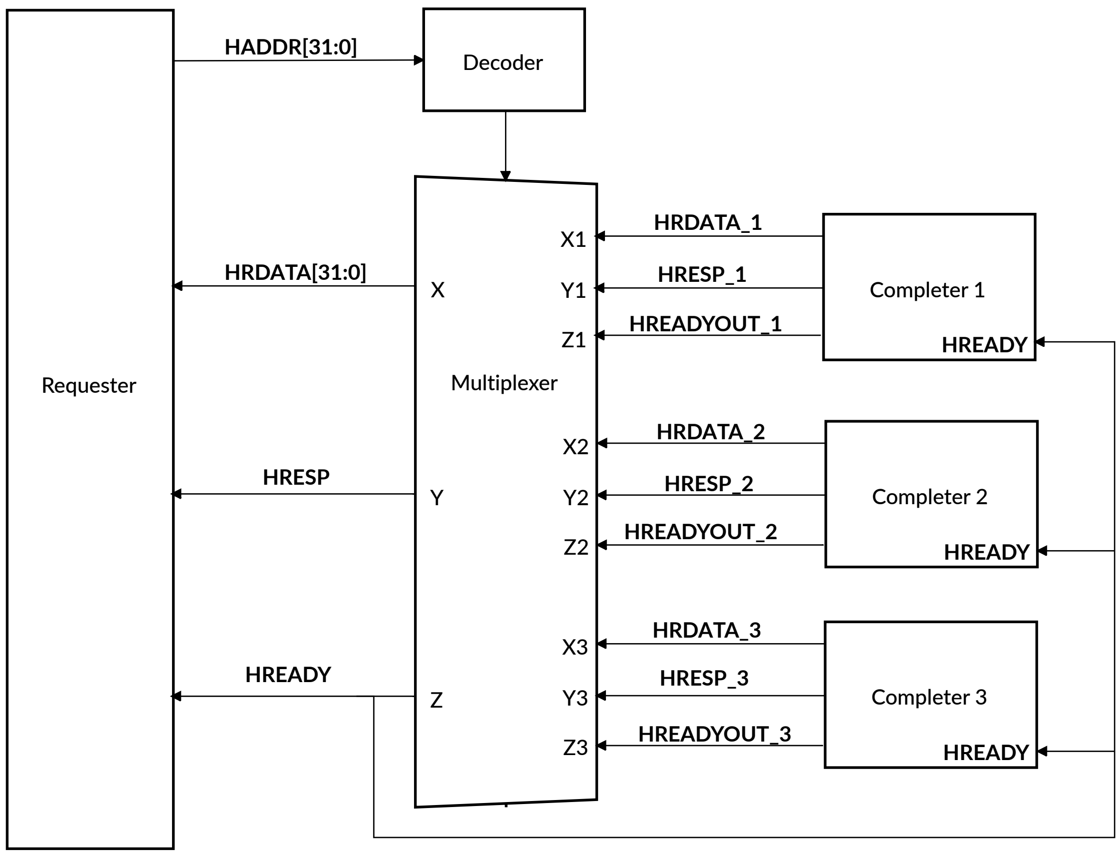
NI-700 HSNI completer units receive and processes requests from AHB and AHB-Lite requester devices.

These units convert AHB and AHB-Lite transactions into Generic Transport (GT) packets and decode GT read and write response packets into AHB responses. The HSNI external interface can be configured as an AHB or AHB-Lite completer interface, or as an AHB or AHB-Lite mirrored requester interface.

HSNI signal names

HREADYOUT is not an input to the HSNI. Rather, this signal is the HSNI port, as defined in the Arm RTL. For clarity, the following figure in the *Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite* shows the HREADY and HREADYOUT signals.

Figure 2-3: AHB protocol multiplexer interconnection scheme



The relationship between the HREADYOUT output and the *Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite* is as follows:

- HREADYOUT is the equivalent of HREADY for mirrored systems. There is no HREADYIN signal to the HSNI for mirrored systems.
- HREADY (HREADYIN) and HREADYOUT are both needed in non-mirrored systems. HREADYOUT from the HSNI matches the preceding multiplexor interconnection scheme from the AHB specification. The HREADYIN input corresponds to the HREADY input port on the completers in the AHB specification multiplexor interconnection scheme.

The following table shows the HSNI system modes, signal names, and signal directions.

Table 2-1: HSNI signals

Mode	Signal name	Direction
Non-mirrored mode systems	HREADY. An HREADYIN input corresponds to the HREADY input port on the completers in the *Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite*.	Input to the HSNI

Mode	Signal name	Direction
Non-mirrored mode systems	HREADYOUT. HREADYOUT from an HSNI corresponds to the multiplexor interconnection scheme in the *Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite*.	Output from the HSNI
Mirrored mode systems	HREADY. HREADYHREADYOUT in the Arm RTL corresponds to HREADY in the *Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite* for mirrored systems. There is no HREADYIN signal to HSNIs for mirrored systems.	Output to the HSNI

HMNI signal names

For the HMNI, HREADYOUT is always an input. HMNI mirrored mode contains two extra outputs, HREADY and HSEL. The following table shows the HMNI system modes, signal names, and signal directions.

Table 2-2: HMNI signals

Mode	Signal name	Direction
Mirrored mode systems	HREADY HSEL HREADYOUT	Output from the HMNI Output from the HMNI Always an input to the HMNI
AHB requester interface (non-mirrored mode systems)	HREADYOUT	Always an input to the HMNI

Mirrored AHB completer interface

The AHB completer interface configures the interface with output signals HSEL, HREADYOUT, and HREADY.

AHB requester interface

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

HSNIs perform the following functions:

- Conversion of requests, data, and response transactions between the AHB and internal GT protocols.
- Address decoding.
- Hazarding. If there are any outstanding writes, a new read transaction is stopped.
- If burst promotion is enabled, conversion of AHB INCR bursts to INCR4 bursts, where possible.
- Burst splitting of incoming transactions. HSNIs split 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 a write response is received from downstream.
- Hard and soft QoS bandwidth regulation. HSNIs have programmable registers for setting these regulators in different modes.

- Timing isolation from the external requester and the network.
- Low-wire mode, where GT request and response channels are shared between reads and writes.
- Receiving responses from requester network interfaces that have data widths that are the same size, a smaller size, or a larger size. When an HSNI receives data from a target with a smaller data width, the read data beats can arrive as fragments. The HSNI collates the fragmented responses to create a data beat of a size that corresponds to its data width.
- Receiving different error responses when combining responses for individual data fragments. In such cases, HSNIs use the following priority order to create the final response that is sent to the AHB requester:
    1. DECERR or SLVERR (highest).
    2. OK.
    3. EXOKAY (lowest).

HSNIs do not support the following features:

- Data widths of 512 bits and 1024 bits.
- Locked transfers. The HMASTLOCK signal not supported.
- Multi-copy atomicity. However, if early write response is disabled, then HSNIs do support multi-copy atomicity.
- Multiple completer select (HSELx) signaling.
- Split and retry

Note

Shareable exclusive transactions are downgraded to Non-shareable exclusive transactions.

2.5.4 HMNI

NI-700 HMNI units receive and process Generic Transport (GT) packets from the network layer.

These units convert GT packets into AHB and AHB-Lite transactions and decode AHB read and write response packets into GT packets. HMNIs can be configured as either AHB requester interfaces or mirrored AHB completer interfaces.

AHB requester interface

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

AHB mirrored completer interface

This option provides all the AHB signals for a completer, which includes HSEL, HREADY input, and HREADY output signals. Using this option enables the direct connection of an AHB completer to an HMNI.

HMNIs perform the following functions:

- Conversion of requests, data, and response transactions between the AHB and GT protocols
- Transaction address decoding into route vectors
- Timing isolation from the external completer and the network
- Low-wire mode, where GT request and response channels are shared between reads and writes
- Non-blocking flow control of concurrent traffic by supporting multiple incoming Resource Planes (RPs)
- Burst handling of incoming WRAP and INCR bursts
- Burst conversion and splitting to handle sparse writes and unaligned accesses

  When splitting any non-modifiable burst, HMNIs assert HMASTLOCK to prevent other requesters accessing the same memory location during the splitting sequence.
- Handling error responses from downstream completers

2.5.5 PMNI

NI-700 PMNI units receive and process Generic Transport (GT) packets from the network layer.

These units convert GT packets into APB transactions and decode APB read and write response packets into GT packets.

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

PMNIs perform 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
- Handling multiplexed read and write traffic on a single channel by using low-wire mode
- Non-blocking flow control of concurrent traffic by supporting multiple incoming Resource Planes (RPs)
- Routing read and write responses back to initiators by using an address decoder
- Supporting up to 16 APB interfaces on a single PMNI. Each interface can be individually specified to be APB3 or APB4. An internal decoder is used to generate the APB PSELx signal for selecting a specific APB requester interface.
- Supporting WriteNoSnoop and ReadNoSnoop opcodes only. All unsupported opcodes are processed as follows:
    - For write requests, the write data is drained instead of forwarding the data onto the APB bus. A write response is issued with an error.

- For read requests, all zero read data beats are forwarded and a read response is issued with an error.

2.5.6 PCDC

NI-700 PCDC units form bridges between different clock domains, power domains, or both clock and power domains. As GT flits are transferred between domains operating at different clock speeds, PCDCs synchronize passing flits to the new clock speed.

If your design contains multiple clock domains, power domains, or clock and power domains, PCDC units are used to control power and clock domain crossing.

To permit entry and exit of flits, PCDC units have one GT input port and one GT output port.

PCDC units have Q-Channel LPIs for each configured power domain, allowing for power domain control. Likewise, there is a Q-Channel LPI for each configured clock domain to enable clock domain control. These 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.

PCDC units perform the following functions:

- Power and clock domain crossing.
- Reordering flits according to RPs. PCDC units do not alter flits as they traverse the block.
- Controlling power domain quiescence.
- Controlling clock domain quiescence.

2.5.7 Routers

NI-700 router units channel GT flits through the network layer of the interconnect.

Routers perform the following functions:

- Transporting GT flits between a configurable number of input ports and output ports according to the flit routing field.
- Routing flits according to RPs. If a router has more than one output port, it updates the flit routing field. Other than this update, routers do not alter flits as they are routed through the unit.

2.5.8 SERDES units

NI-700 SERDES units resize GT flits in the network layer of the interconnect.

SERDES units have the following connections:

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

SERDES units perform the following functions:

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

2.5.9 PMU

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
- A control network interface for programming and read access requests from the 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. You can determine the counter that has overflowed by using the PMU control and configuration registers. You can also use these registers 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 about the PMU, see Performance monitoring.

2.6 Configurable options

You can customize the top-level topology and the individual functional units of NI-700 to meet your specific design requirements.

NI-700 provides the following options for configuration:

- Microarchitecture:
  - Up to a maximum total of 255 upstream and downstream interfaces
  - Up to 128 completer network interfaces, that is, ASNI configuration options and HSNI configuration options
  - Up to 127 requester network interfaces, that is, AMNI configuration options, HMNI configuration options, and PMNI configuration options
- Voltage, power, and clock domains:
  - Up to 32 voltage domains
  - Up to 32 power domains, with each power domain in one voltage domain
  - Up to 32 clock domains, with each clock domain in one power domain
  - Power and clock domain crossing within the network, supporting synchronous, asynchronous, and integer ratio clock domain crossings
  - RTL hierarchy by voltage domain, then power domain, then clock domain
  - RTL hierarchy according to a configurable grouping of components
- Address map:
  - Configurable address map for address-based routing from each upstream interface to the corresponding downstream interfaces
  - Separate address map for each upstream interface
  - Multiple address regions in address maps, with each region aligned and sized based on a 4KB granularity
  - Address map regions can target one downstream interface or can be hashed across two or four downstream interfaces.
- Cache line size:
  - Compatible with specific cache line sizes only. The following table lists the cache line sizes that NI-700 supports. No other cache line sizes are supported.

Table 2-3: Supported cache line sizes

Data width	Cache line size
32 bits	64 bytes
64 bits, 128 bits, 256 bits, 512 bits	64 bytes or 128 bytes
1024 bits	128 bytes

- Topologies:
  - Routers with up to eight inputs and up to eight outputs for flexible topology choices
  - Up to four Resource Planes (RPs) to reduce Head-of-Line (HoL) blocking
  - Configurable link sizes and link crediting, with the option to resize flits within the network by using SERDES components
  - PCDC configuration options for bridging between different power and clock domains
  - Option to merge read and write channels to reduce wire count and area

    - Option to duplicate channels for more bandwidth
- Unit-level configuration:
    - Flexible timing closure options
    - Configurable transaction tracker depths
    - OPTIONAL burst splitting logic. This feature can be included if a design requires transactions to be split, or excluded to save area on designs where burst splitting is not required.
    - Quality of Service (QoS) regulators that can update the QoS value on a transaction according to latency targets
    - OPTIONAL Interconnect Device Management (IDM) feature for configuration and management of system components by the interconnect
    - Configurable FIFO sizes when crossing clock and power domains

2.6.1 ASNI configuration options

ASNI units provide various options that you can configure to meet the specific requirements of your design, including the address and data widths.

You can configure the following options:

- Address width of 32–64 bits
- One of the following data widths:
    - 32 bits
    - 64 bits
    - 128 bits
    - 256 bits
    - 512 bits
    - 1024 bits
- User sideband signal width

  For more information, see User signals.
- Write acceptance capability of 1–512 transactions

  Sometimes an ASNI might accept more transactions than specified in the write acceptance capability. For example, configuring a register slice at the completer interface position increases the acceptance capability.
- Read acceptance capability of 1–512 transactions

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

- Minimum atomic acceptance, which provides a guarantee of the minimum number of read tracker entries that are reserved for atomics

  The minimum atomic acceptance parameter only applies if the `Atomic_Transactions` property is enabled on the ASNI. If the property is enabled, then the total read tracker size is the sum of the read acceptance and the minimum atomic acceptance.

  When atomic transactions are received on the write channel, the atomic variants load, compare, and swap also require a read response. This process uses a tracker entry in the read tracker.
- Optional pipeline register slices for retiming. These pipeline slices provide flexibility in trading latency for higher frequency. For more information, see *Configuring NI-700 unit-level retiming options* in the NI-700 Configuration and Integration Manual.
- Timing isolation:
  - From the external requester
  - From the network
- Read reorder depth of 1–255 entries

  Permitted read reorder depth values are one, two, all multiples of four from 4–252 inclusive, and 255. If atomic transaction support is enabled at the ASNI, then the read reorder depth plus the minimum atomic transaction acceptance depth must be less than 256.
- Write data FIFO depth of 0–32 entries
- ID width of 1–24 bits
- Ordered Write Observation, which is an AXI4 property that specifies whether all agents observe write transactions with the same ID in the issued order.

  Set the property to enabled, disabled, or pin. For more information, see the *AMBA® AXI and ACE Protocol Specification*.
- Enable IDM
- IDM device ID
- Include burst splitting logic
- Include QoS regulators:
  - The read regulator present setting enables a QoS regulator for the AR channel.
  - The write regulator present setting enables a QoS regulator for the AW channel.
  - The combined regulator present setting enables a QoS regulator that regulates traffic according to the combined bandwidth across both the AR and AW channels.

The following table shows the ASNI features that are supported for each type of interface.

Table 2-4: Supported ASNI features by interface type

Interface type	Parameter name	Support
AXI5 and ACE-Lite	`Wakeup_Signals`	Required. Devices attached to NI-700 must support wakeup signaling.

Interface type	Parameter name	Support
	`Check_Type`	Not supported
	`Poison`	Not supported
	`Trace_Signals`	OPTIONAL. Configurable.
	`Unique_ID_Support`	OPTIONAL
	`QoS_Accept`	Not supported
	`Loopback_Signals`	OPTIONAL. If enabled, set to 8 bits only.
	`Untranslated_Transactions`	Not supported
	`NSAccess_Identifiers`	OPTIONAL
	`MPAM_Support`	OPTIONAL
	`Read_Interleaving_Disabled`	Always set to FALSE
	`Read_Data_Chunking`	OPTIONAL
	`Atomic_Transactions`	OPTIONAL
	`MTE_Support`	OPTIONAL
ACE-Lite	`CMO_On_Read`	OPTIONAL
	`CMO_On_Write`	OPTIONAL
	`Persist_CMO`	OPTIONAL
	`Write_Plus_CMO`	OPTIONAL
	`Cache_Stash_Transactions`	OPTIONAL
	`DeAllocation_Transactions`	OPTIONAL
	`Prefetch_Transaction`	OPTIONAL

2.6.2 AMNI configuration options

AMNI units provide various options that you can configure to meet the specific requirements of your design. For example, you can configure the number of Resource Planes (RPs) in the channels.

You can configure the following options:

- Address width of 32–64 bits
- One of the following data widths:
  - 32 bits
  - 64 bits
  - 128 bits
  - 256 bits
  - 512 bits
  - 1024 bits
- User sideband signal width

User signals are applicable to all AMNI interface types including AXI3. For more information, see User signals.

- Number of RPs present in each of the read request, write request, read response, and write response channels
- Write issuing capability of 1–512 transactions
- Read issuing capability of 1–512 transactions
- Minimum atomic issuance, which provides a guarantee of the minimum number of read tracker entries that are reserved for atomics

  The minimum atomic issuance parameter only applies if the `Atomic_Transactions` property is enabled on the AMNI. If the property is enabled, then the total read tracker size is the sum of the read issuance and the minimum atomic issuance.

  When atomic transactions are received on the write channel, the atomic variants load, compare, and swap also require a read response. This process uses a tracker entry in the read tracker.
- Optional pipeline register slices for retiming. These pipeline slices provide flexibility in trading latency for higher frequency. For more information, see *Configuring NI-700 unit-level retiming options* in the NI-700 Configuration and Integration Manual.
- Timing isolation:
  - From the external completer
  - From the network
- Enable IDM
- IDM device ID
- AXI ID width of 1–32 bits

  To form the outgoing AXI ID, the AMNI appends the Source ID (SrcID) of the incoming request to the least significant bits of the AXI ID. For more information on output IDs, see Calculation of output IDs. The SrcID of the incoming request is captured in the node_id field of the ASNI_NODE_TYPE, Node type register for ASNI registers.

Note

The AxREGION signal is not supported.

You can configure AMNIs with AXI5, ACE5-Lite, ACE5-LiteACP, or AXI3 as the requester interface type.

ACE5-LiteACP has several constraints, some of which are transaction constraints and some of which are interface constraints. For more information these constraints, see the *AMBA® AXI and ACE Protocol Specification*.

The following table shows the AMNI features that are supported for each type of interface.

Table 2-5: Supported AMNI features by interface type

Interface type	Parameter name	Support
AXI5 and ACE-Lite	Wakeup_Signals	Required. AMNIs have an output AWAKEUP signal. The downstream completer can choose to use or ignore this signal.
	Check_Type	Not supported
	Poison	Not supported
	Trace_Signals	OPTIONAL
	Unique_ID_Support	OPTIONAL
	Qos_Accept	OPTIONAL
	Loopback_Signals	OPTIONAL. If enabled, this parameter is set to 8 bits only.
	Untranslated_Transactions	Not supported
	NSAccess_Identifiers	OPTIONAL
	MPAM_Support	OPTIONAL
	Read_Interleaving_Disabled	OPTIONAL
	Read_Data_Chunking	OPTIONAL
	Atomic_Transactions	OPTIONAL
	MTE_Support	OPTIONAL
ACE-Lite	CMO_On_Read	OPTIONAL
	CMO_On_Write	OPTIONAL
	Persist_CMO	OPTIONAL
	Write_Plus_CMO	OPTIONAL
	Cache_Stash_Transactions	OPTIONAL
	DeAllocation_Transactions	OPTIONAL
	Prefetch_Transaction	OPTIONAL
ACE5-LiteACP	Atomic_Transactions	Not supported, as specified by the ACE5-LiteACP protocol
	Write_Plus_CMO	Not supported, as specified by the ACE5-LiteACP protocol
	Prefetch_Transaction	Not supported, as specified by the ACE5-LiteACP protocol
	DeAllocation_Transactions	Not supported, as specified by the ACE5-LiteACP protocol
	Cache_Stash_Transactions	OPTIONAL
	CMO_On_Read	Not supported, as specified by the ACE5-LiteACP protocol
	CMO_On_Write	Not supported, as specified by the ACE5-LiteACP protocol
	Persist_CMO	Not supported, as specified by the ACE5-LiteACP protocol
	Trace_Signals	OPTIONAL
	NSAccess_Identifiers	Not supported, as specified by the ACE5-LiteACP protocol
	MPAM_Support	OPTIONAL
	Unique_ID_Support	OPTIONAL
	Read_Data_Chunking	OPTIONAL
	Loopback_Signals	Not supported, as specified by the ACE5-LiteACP protocol
	MTE_Support	Not supported, as specified by the ACE5-LiteACP protocol

Interface type	Parameter name	Support
	`Qos_Accept`	Not supported, as specified by the ACE5-LiteACP protocol
	`Read_Interleaving_Disabled`	OPTIONAL
	`Untranslated_Transactions`	Not supported
	`Check_Type`	Not supported
	`Poison`	Not supported
AXI3	`Atomic_Transactions`	Not supported
	`Trace_Signals`	Not supported
	`NSAccess_Identifiers`	Not supported
	`MPAM_Support`	Not supported
	`Unique_ID_Support`	Not supported
	`Read_Data_Chunking`	Not supported
	`Loopback_Signals`	Not supported
	`MTE_Support`	Not supported
	`QoS_Accept`	Not supported
	`Read_Interleaving_Disabled`	Not supported
	`DeAllocation_Transactions`	Not supported
	`Cache_Stash_Transactions`	Not supported
	`CMO_On_Read`	Not supported
	`CMO_On_Write`	Not supported
	`Persist_CMO`	Not supported
	`Write_Plus_CMO`	Not supported
	`Prefetch_Transaction`	Not supported
	`Untranslated_Transactions`	Not supported
	`Check_Type`	Not supported
	`Poison`	Not supported

2.6.3 HSNI configuration options

HSNI units provide various options that you can configure to meet the specific requirements of your design, including the read and write data widths. However, some HSNI properties are fixed in NI-700.

You can configure the following options:

- Interface type

  HSNIs support the AHB5 standard and mirrored interface types.

- One of the following read and write data widths:
  - 32 bits
  - 64 bits
  - 128 bits

    - 256 bits

  The read and write data widths must be set to the same value.

- User sideband signal width

  For more information, see User signals.

- Write acceptance capability of 1–16 transactions

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

- HMASTER width of 1–8 bits
- Enable Extended Memory Type support
- Enable Secure transfer support
- Enable exclusive transfer support
- Enable early burst termination acceptance
- Enable burst conversion support
- Enable early write response support

  When early write response is enabled, you can configure an HSNI to support 1–16 outstanding writes.

- Write data buffer FIFO depth of 0–16 data beats
- Include programmable QoS regulators
- Optional pipeline register slices for retiming. These pipeline slices provide flexibility in trading latency for higher frequency. For more information, see *Configuring NI-700 unit-level retiming options* in the NI-700 Configuration and Integration Manual.
- Timing isolation:
    - From the external requester
    - From the network
- Enable IDM
- IDM device ID

The following HSNI properties are not configurable:

- Address width

  This value is fixed at 32 bits.

- Endianness

  HSNIs only support word-invariant little-endianness.

Note

HSNIs do not support multiple completer select (HSELx) signaling, split and retry, or locked transfers.

The following table shows the configuration options for HSNIs.

Table 2-6: HSNI configuration options

Parameter name	Support
`Extended_Memory_Types`	OPTIONAL. Can be enabled or disabled.
`Secure_Transfers`	OPTIONAL. Set to pin, programmable, Secure, or Non-secure.
`Endianness`	BE32 only. HSNIs only support word-invariant little-endianness.
`Exclusive_Transfers`	OPTIONAL. Can be enabled or disabled.
`Mirror_Interface`	OPTIONAL. Can be enabled or disabled.
`Multi_Copy_Atomicity`	OPTIONAL. Set to FALSE if early write response is enabled, otherwise set to TRUE.
`Stable_Between_Clock`	Always set to FALSE

2.6.4 HMNI configuration options

HMNI units provide various options that you can configure to meet the specific requirements of your design, including whether to use a standard or mirrored interface. However, some HMNI properties are fixed in NI-700.

You can configure the following options:

- Interface type

  HMNIs support the AHB5 standard and mirrored interface types.

- One of the following read and write data widths:
  - 32 bits
  - 64 bits
  - 128 bits
  - 256 bits

The read and write data widths must be set to the same value.

- User sideband signal width

  For more information, see User signals
- Enable Extended Memory Type support
- Enable Secure transfer support
- Enable Secure access support
- Enable exclusive transfer support
- Optional pipeline register slices for retiming. These pipeline slices provide flexibility in trading latency for higher frequency. For more information, see *Configuring NI-700 unit-level retiming options* in the NI-700 Configuration and Integration Manual.
- Timing isolation:
  - From the external completer
  - From the network
- Enable IDM
- IDM device ID

The following HMNI properties are not configurable:

- Address width

  This value is fixed at 32 bits.
- Endianness

  HMNIs only support word-invariant little-endianness.

Note

HMNIs do not support multiple completer select (HSELx) signaling or split and retry.

The following table shows the configuration options for HMNIs.

Table 2-7: HMNI configuration options

Parameter name	Support
`Extended_Memory_Types`	OPTIONAL. Can be enabled or disabled.
`Secure_Transfers`	OPTIONAL. Set to pin, register, Secure, or Non-secure.

Parameter name	Support
`Endianness`	BE32 only. HSNIs only support word-invariant little-endianness.
`Exclusive_Transfers`	OPTIONAL. Can be enabled or disabled.
`Mirror_Interface`	Can be enabled or disabled
`Stable_Between_Clock`	Always set to FALSE

2.6.5 PMNI configuration options

PMNI units provide various options that you can configure to meet the specific requirements of your design, including the APB protocol to use. However, some PMNI properties are fixed in NI-700.

You can configure the following options:

- APB protocol type

  PMNIs support the APB3 and APB4 protocols.
- Enable Secure access support

  You can control Secure access support through a register or you can determine it from a pin.
- Optional pipeline register slices for retiming. These pipeline slices provide flexibility in trading latency for higher frequency. For more information, see *Configuring NI-700 unit-level retiming options* in the NI-700 Configuration and Integration Manual.
- Timing isolation:
  - From the network
- Enable IDM
- IDM device ID

The following PMNI properties are not configurable:

- Address width

  This value is fixed at 32 bits.
- Read and write data widths

  These values are fixed at 32 bits.

2.6.6 PCDC configuration options

PCDC units provide various options that you can configure to meet the specific requirements of your design, including the flit width for each channel.

You can configure the following options:

- PCDC synchronization mode:
    - Use asynchronous mode when the clocks are not synchronized.
    - Use synchronous (1:1) mode when the clocks are identical.
    - Use synchronous (1:N) mode when the clocks are synchronized and the first clock has a lower frequency than the second clock. The positive edge of the first clock must always coincide with a positive edge of the second clock.
    - Use synchronous (M:1) mode when the clocks are synchronized and the first clock has a higher frequency than the second clock. The positive edge of the second clock must always coincide with a positive edge of the first clock.
- Maximum number of credits for each RP that can be accepted at the input and output ports.
- Flit width for each channel.
- Optional pipeline register slices for retiming. These pipeline slices provide flexibility in trading latency for higher frequency. For more information, see *Configuring NI-700 unit-level retiming options* in the NI-700 Configuration and Integration Manual.

When a PCDC is set to asynchronous, you can also configure:

- Number of synchronizer register stages from two to four
- Buffer depth for data and credit FIFO

2.6.7 Router configuration options

Router units provide various options that you can configure to meet the specific requirements of your design, including the numbers of inputs and outputs.

You can configure the following options:

- Number of router inputs from one to eight
- Number of router outputs from one to eight
- Channel credits for each source, destination, and RP
- Frequency of arbitration decisions that disregard QoS
- Optional pipeline register slices for retiming. These pipeline slices provide flexibility in trading latency for higher frequency. For more information, see *Configuring NI-700 unit-level retiming options* in the NI-700 Configuration and Integration Manual.

2.7 Product design flow

Before using NI-700, several processes must be performed. To obtain the best performance, we recommend that some of the implementation stages are carried out before integrating NI-700 into the wider SoC.

The product design flow comprises the following processes:

Implementation

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

Integration

The integrator connects the implemented design into an SoC, including establishing connections to:

- Memory system
- Processors
- Peripherals

Final SoC implementation

The final, fully-integrated SoC is implemented in silicon.

Arm can only provide guidance on the final implementation of Arm products. If Arm provides such guidance for a product, then a separate document is included in the implementation package for that product.

Programming

The system programmer develops the software that is necessary to configure and initialize NI-700, and tests the application software.

Each process in the product design flow:

- Is separate and a different individual or team can complete each process
- Can include implementation and integration choices that affect the behavior and features of NI-700, and therefore the other tasks in the flow

Various NI-700 documents provide more information on these processes. For more information, see Product documentation.

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

The operation of the final device depends on:

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, which can be:

- Area

- Maximum frequency
- Performance of the resulting macrocell

For example, the implementer can set the number of outstanding transactions that each requester and completer 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 the software configuration is specified. The configurations can also limit the options that are 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.

2.8 Product documentation

Each NI-700 document is aimed at a particular audience and is associated with specific tasks in the design flow.

These documents do not reproduce information that is available in the Arm architecture and protocol specifications. For architecture and protocol information that relates to NI-700, see the Additional reading section.

The NI-700 documentation comprises:

Technical Reference Manual

The Technical Reference Manual (TRM) describes the functionality and the effects of functional options on the behavior of NI-700. This document is useful at all stages of the product 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 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:

- Descriptions of NI-700 features
- Design-time configuration options
- Reset-time configuration options
- Available build configuration options and related considerations

- Instructions for configuring the RTL with the build configuration options
- Instructions for running test vectors
- Sign-off processes for the configured design
- Considerations when integrating NI-700 into your system

The Arm product deliverables include reference scripts and information about using these scripts to implement your design. The reference methodology flows that Arm supplies are example reference implementations only. For EDA tool support, contact your EDA tool vendor.

The CIM is a Confidential document that is only available to licensees of NI-700.

3. 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 flexible, the interconnect can occupy various power states and operating modes.

An external P-Channel controls each power domain and defines the power state into which the power domain can enter. 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:

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

If multiple power domains are used, the power domains must be configured at the same level in the domain hierarchy. For more information, see Power.

NI-700 provides different types of clocks that can be arranged hierarchically to allow for different power scenarios. For more information, see Clocks.

Separate blocks are used for power and clock control. For more information, see Power control and Clock and reset control, respectively.

3.1 Power

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

Up to 32 separate power domains and up to 32 separate voltage domains can be configured within an NI-700 design. In designs with multiple power domains, all the power domains must exist at the same level in the domain hierarchy. NI-700 does not support designs with power domains at different hierarchical levels.

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. For more information, see P-Channel low-power interface.

The following asserted P-Channel PACTIVE bits indicate the minimum power state that the power domain requires to guarantee progress. For more information, see Power state requirements and characteristics.

PACTIVE[16]

CONFIG. Enables restricted completer network interface access for the power domain.

PACTIVE[8]

ON. Fully powered state for all logic in the power domain.

PACTIVE[5]

FULL_RET. Static retention state for all logic within the power domain.

PACTIVE[0]

OFF. Fully unpowered state for the power domain.

3.1.1 Power state requirements and characteristics

NI-700 has specific signaling requirements for the different power states that are supported. Only specific power state transitions are permitted, which depend on the starting state.

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

Out of reset, the PSTATE that is presented to NI-700 must be one of the supported values in the following table. If any other value is presented, the behavior is **UNPREDICTABLE**. The highest of the asserted P-Channel PACTIVE bits indicates the minimum power state that the power domain requires to guarantee progress.

Table 3-1: Valid power states for power domains and the requirements of those power states

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

CONFIG power state

The CONFIG state restricts access to completer network interfaces in the power domain. In this state, only completer network interfaces with <SUBORDINATE_INTERFACE>_CONFIG_ACCESS at their reset input pins set HIGH permit ingress of external transactions.

When PACTIVE[16] is HIGH, the CONFIG power state is the lowest power state that is required for the system. For example, this scenario can occur when the only transaction that requires access to fully powered logic is from a CONFIG-defined interface.

If PACTIVE[16] is LOW, then PACTIVE[8] must be checked to determine the required power state. In this case, PACTIVE[8] determines whether the system must transition to ON or whether the system can enter the FULL_RET or OFF states to save power.

State transitions from CONFIG to ON, FULL_RET, or OFF are permitted. The highest PACTIVE bit that is HIGH determines the transition.

ON power state

ON is the fully powered state for all logic in the power domain. The power domain must be in the ON state for all interfaces to progress.

When PACTIVE[8] is HIGH, the ON power state is required, such as when a transaction requires access to fully powered logic.

If PACTIVE[8] is LOW, then 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 permitted. The lowest PACTIVE bit that is HIGH determines the transition. However, we only recommend transitioning to CONFIG if system reconfiguration is required. Otherwise, we recommend a transition to OFF.

FULL_RET power state

FULL_RET is the static retention state for all logic instances within the power domain. In the FULL_RET state, all external flow control signals are held in a state that prevents propagation of any transactions.

State transitions from FULL_RET to ON or CONFIG are permitted. Transitioning from the FULL_RET state to the OFF state is not permitted.

OFF power state

OFF is the fully off state for the power domain. In the OFF state, all external flow control signals are held in a state that prevents propagation of any transactions.

State transitions from OFF to ON or CONFIG are permitted. Transitioning from the OFF state to the FULL_RET state is not permitted.

3.1.2 P-Channel low-power interface

Each power domain in NI-700 is connected to a standard P-Channel LPI 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 highest permitted 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.

The P-Channel uses the following LPI signals to indicate the external power state and to specify the power state into which NI-700 is required to transition.

Table 3-2: P-Channel LPI signals

Name	Direction	Purpose
PACTIVE[16:0]	Output	Vector indicator of the power states that NI-700 is eligible to enter
PSTATE[7:0]	Input	Binary value of the power state into which an external controller requires NI-700 to transition
PREQ	Input	Request signal to initiate a power state transition
PACCEPT	Output	Handshake signal to indicate that the power state transition is complete
PDENY	Output	Handshake signal to indicate that the power state transition cannot be completed

3.2 Clocks

NI-700 provides configurable clock domains and supports hierarchical clock gating.

You can configure up to 32 separate clock domains within your NI-700 design, which can be arranged in a hierarchy. For more information, see Levels of clock gating.

Each of the configured clock domains can be separately gated. For more information, see Hierarchical clock gating.

The clock domains are gated by Q-Channel LPIs. For more information, see Q-Channel low-power interface.

The clock gating process is managed by the External Clock Controller. For more information, see External Clock Controller.

NI-700 requires that connected interfaces support a specific wakeup signal. When this signal is asserted, NI-700 requests activation of the relevant clock domain to ensure that the appropriate system components are ready to receive transactions. For more information, see Clock domain wakeup.

Every clock domain has a single clock pin input, which is labeled <CLKNAME>_CLK.

3.2.1 Levels of clock gating

NI-700 contains different clock types that are arranged in a hierarchy, including clocks supplying clock domains through to local clocks that are created by the RTL.

The following clock types are included in NI-700:

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 the 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 clock sequential elements directly in NI-700. The exact set of local clocks is internal to NI-700 and is not described here.

3.2.2 Hierarchical clock gating

NI-700 supports hierarchical clock gating. During periods of low activity, the system can use hierarchical clock gating to transition to a low-power state.

Transitioning to a low-power state enables the system to save the power that the active clock tree would normally consume. Control over individual clock domains allows for flexible system design and therefore flexible power state design.

Hierarchical clock gating can gate the following regions:

- Completer network interfaces, for example ASNIs
- Requester network interfaces, for example AMNIs
- Routers
- PCDC blocks
- SERDES blocks
- Register blocks

The Q-Channel LPI enables hierarchical clock gating by communicating with the clock domain controller to request that the clock domain becomes quiescent. External clock controllers can use the Q-Channel LPI to request gating of individual clock domains in the interconnect.

On receipt of a request, the interconnect waits until there are no outstanding transactions within the clock domain and then blocks new transactions from entering. When this process is complete, the clock domain sends an acknowledgment to indicate that the clock controller can remove the clock.

3.2.3 Q-Channel low-power interface

Each clock domain in NI-700 is connected to a standard Q-Channel LPI that gates the clock domain. A Q-Channel is present for every clock domain.

The Q-Channel LPI contains LPI signals to control hierarchical clock gating in NI-700. Hierarchical clock gating is always present in the NI-700 configuration. For more information on the function of the Q-Channel LPI signals, see *AMBA® Low Power Interface Specification Arm® Q-Channel and P-Channel Interfaces*.

Table 3-3: Q-Channel LPI signals

Signal	Direction	Description	Source	Destination
QACTIVE	Output, input	Interconnect active	Interconnect	Controller
QREQn	Output, input	System low-power request	Controller	Interconnect
QACKn	Output, input	Low-power request acknowledgment	Interconnect	Controller

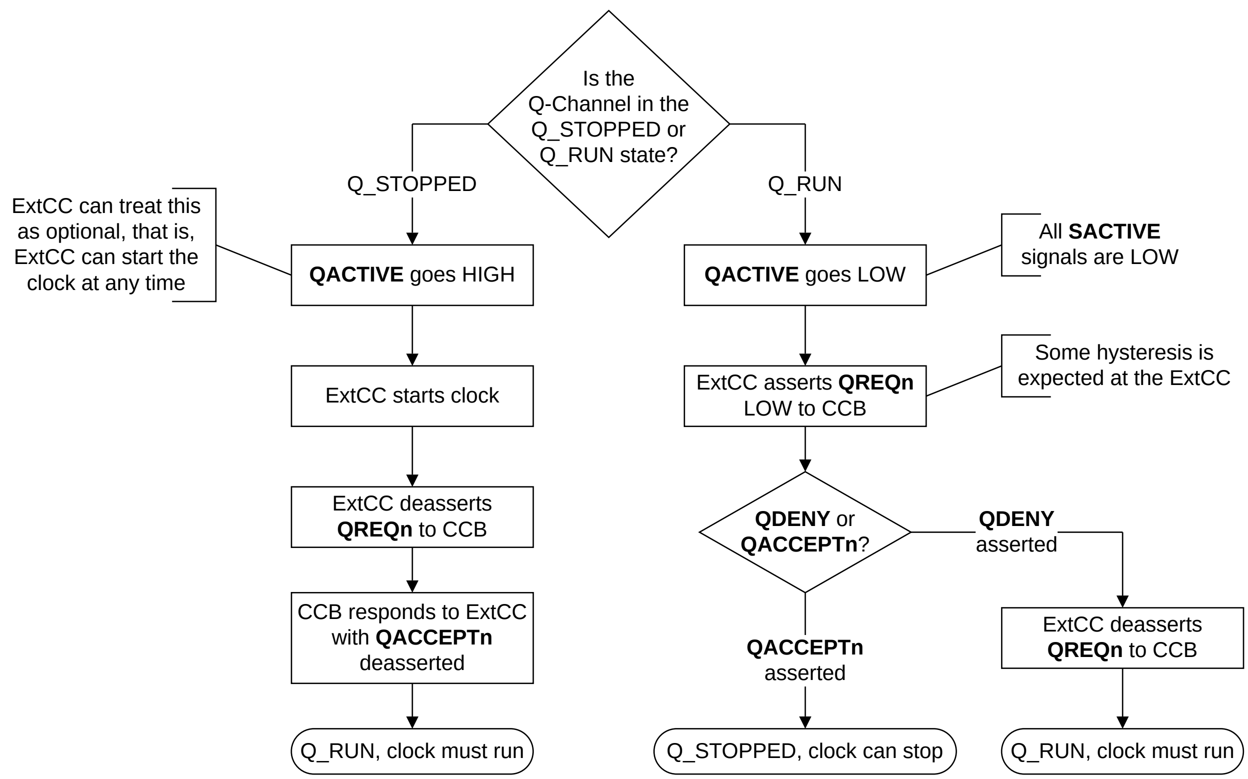
Signal	Direction	Description	Source	Destination
QDENY	Output, input	Negative acknowledgment after receiving a QREQn assertion, indicating NI-700 has refused the request from the controller to prepare to stop the clocks	Interconnect	Controller

3.2.4 External clock controller

The external clock controller controls the clock gating flow.

The following figure shows an example clock gating flow and how the external clock controller controls that flow.

Figure 3-1: Example External Clock Controller (ExtCC) clock gating control flow



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.

The following requirements apply to the external clock controller:

- The external clock controller must supply a clock to NI-700 when the Q-Channel is in any state other than Q_STOPPED.

- The external clock controller 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 external clock controller is responsible for bringing the Q-Channel to the Q_RUN state after reset deassertion.
- The exact behavior of the external clock controller and its usage of QREQn in response to QACTIVE deassertion is not described here. However, the design of the external clock controller is likely to include a control loop with some hysteresis. This feature ensures that hierarchical clock gating is enabled when the system is inactive for long periods. Hierarchical clock gating is not enabled for short periods of inactivity. If the clocks are stopped in response to short periods of inactivity, the performance of NI-700 can be negatively affected.
- It is the responsibility of the SoC designer to fully control the clock management Q-Channel. If a control or configuration bit is required to completely enable or disable hierarchical clock gating, that register or bit must exist outside of NI-700. There is no internal means of disabling hierarchical clock gating in NI-700.

3.2.5 Clock domain wakeup

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

NI-700 requires that upstream requesters support AWAKEUP when connecting those requesters to the interconnect. Similarly, NI-700 drives AWAKEUP from the AXI requester interfaces. If AXI4 requesters support AWAKEUP, they can connect to NI-700.

Each ASNI has an input signal, AWAKEUP, that 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.

3.3 Power control

The NI-700 power control network consists of power control blocks, clock control blocks, and several power control signals.

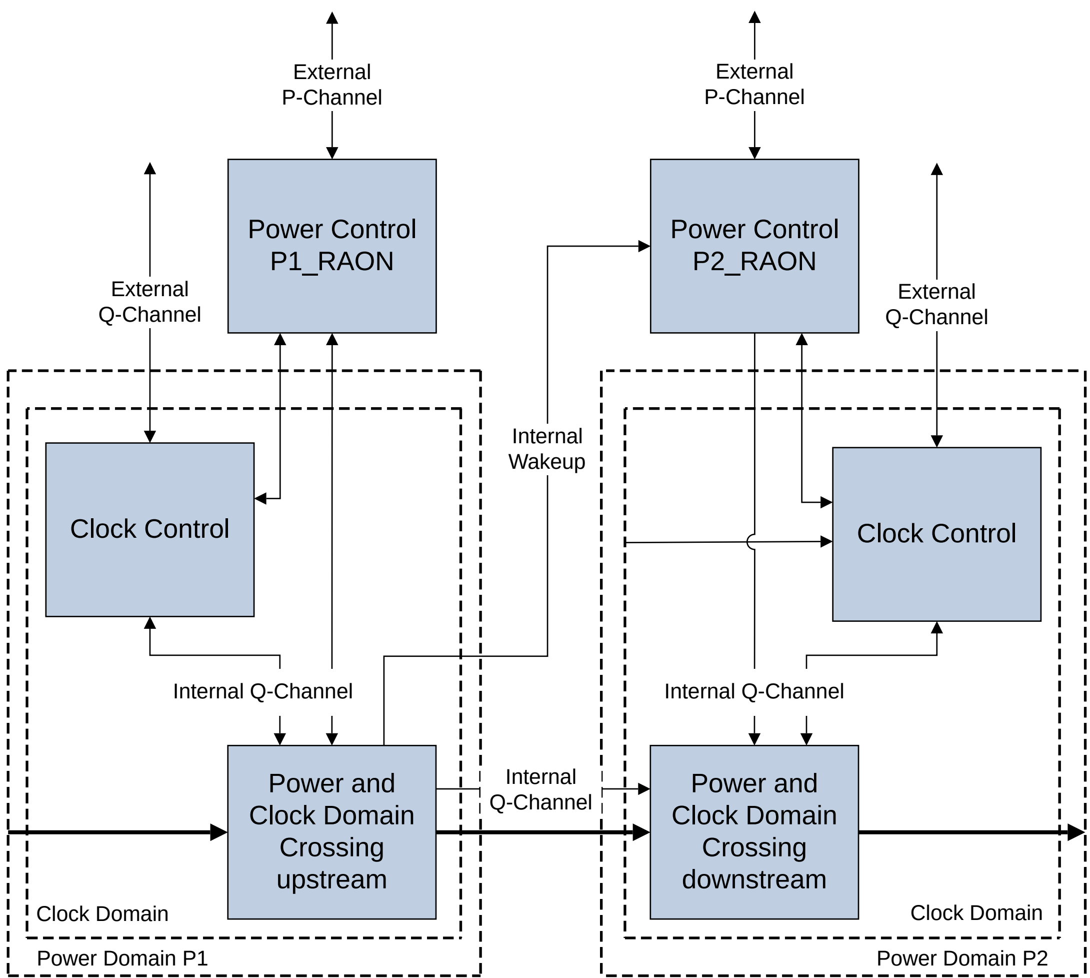
An NI-700 power controller must be in a power domain that is Relatively Always ON (RAON) compared to the power domain that the power controller manages. This requirement enables assertion of the internal wakeup signal and the PACTIVE signal on the external P-Channel as they are in the *_RAON power domain. When asserted, these signals 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 P2 power domain. Similarly, before requesting the power state transition through the *_AON power controller, the SoC power controller must also ensure that the corresponding P1 or P2 power domain is already ON.

The clock and reset to the power controller comes from the clock controller in the corresponding power domain. For example, P1 could contain 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. Therefore, the clock and reset to the P1_RAON power controller comes from the corresponding clock controller in the same clock domain in P1. Before the P1 or P2 power domains power down, the signals crossing between each power domain and the corresponding P1_RAON or P2_RAON power domain are all isolated.

The following figure shows the various elements in the power control network for NI-700.

Figure 3-2: NI-700 power control network



Specific power control steps are required to enable managed power domains to move between the ON and OFF states. For more information, see Power control sequences.

NI-700 includes a feature that enables attached devices to transition between power states while the interconnect remains powered up. For more information, see External power domain boundaries.

HSNIs include logic to ensure that the AHB address phase can be sampled even when the unit is clock gated. For more information, see AHB address phase buffering in HSNIs.

3.3.1 Power control sequences

The NI-700 power control network must perform specific sequences of actions to allow downstream power domains to transition between power states.

The following sections list the steps involved in ON to OFF and OFF to ON power transitions and the order in which they must be performed.

Upstream power domain ON, downstream power domain ON→OFF

The following sequence describes how a downstream power domain transitions from ON to 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. The external P-Channel requests that the power domain enters the P_OFF state.
3. The internal power QREQn signal, which targets the downstream PCDC, goes LOW.
4. If there is no activity in the downstream PCDC:
    a. The downstream PCDC performs logical isolation of the boundary and indicates to the upstream PCDC a requirement to enter the P_OFF state.
    b. The upstream PCDC acknowledges the P_OFF state request from downstream PCDC, performs logical isolation, and resets the PCDC FIFO pointers to the reset value.
    c. The downstream PCDC receives the acknowledgment from the upstream PCDC, resets the PCDC FIFO pointers to the reset value, and issues QACCEPT to the power control block.
    d. The power control block issues a P-Channel accept to the external interface.
5. The external clock controller requests that all clock Q-Channels enter the Q_STOPPED state.
6. When all P-Channels and Q-Channels are in the P_OFF or Q_STOPPED states, all power domain pins are physically isolated, if necessary.

    In NI-700, all isolation values are inactive values of the corresponding signals. Therefore, use 0 for active-HIGH polarity, and 1 for active-LOW polarity.

Upstream power domain ON, downstream power domain OFF→ON

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

1. A new upstream transaction arrives in the CDC.
2. The upstream PCDC, in the RAON domain, asserts an internal wakeup signal to the downstream power controller.

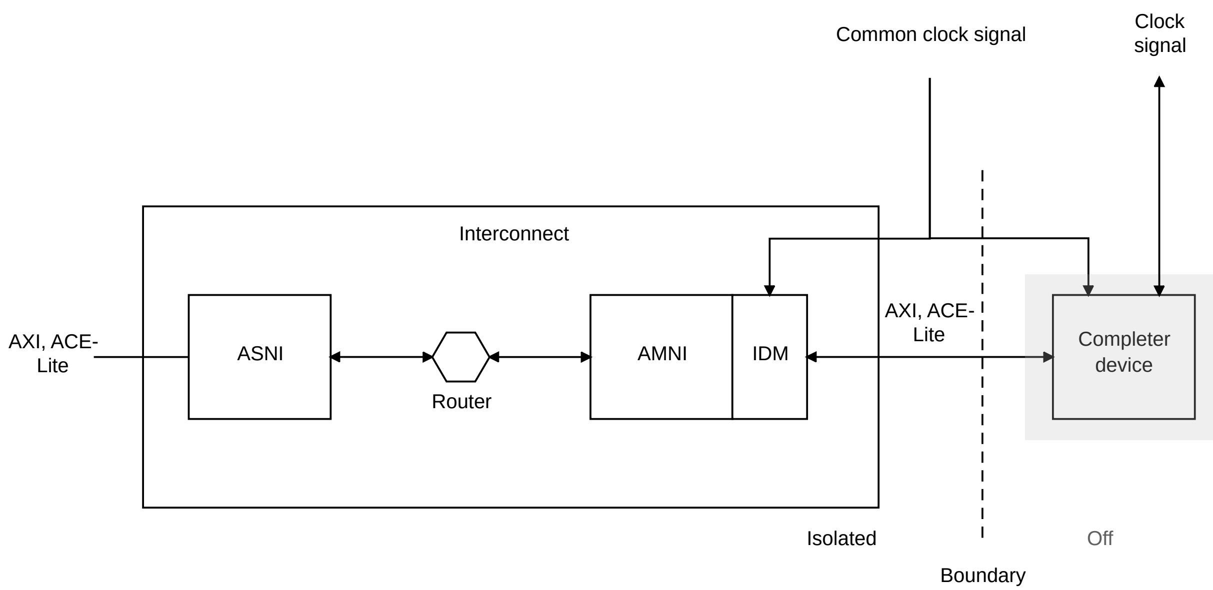
3. The downstream power controller asserts the external higher power state PACTIVE asynchronously.
4. The external power control:
    a. Restores power to the domain.
    b. Applies resets to the domain.
    c. Removes physical isolation.
    d. Removes resets to the domain.
5. The external P-Channel requests to enter P_ON and the clock Q-Channel requests to enter Q_ON.
6. The internal QREQn signal, which targets the downstream PCDC, goes HIGH.
    a. The downstream PCDC removes logical isolation of the boundary and issues a QACCEPTn transition to the clock control and power control blocks.
    b. The clock control and power control blocks forward the QACCEPTn transition to the external interface.
    c. The downstream PCDC indicates to upstream PCDC that power is restored and that the downstream PCDC is in the P_ON state.
    d. The upstream PCDC acknowledges the downstream PCDC and removes logical isolation.

3.3.2 External power domain boundaries

External power domain boundaries are used in NI-700 to enable attached devices to switch power state independently of the interconnect.

NI-700 provides power isolation on AXI signals at the boundary of the interconnect and integrated IP. This feature can be used when the 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 completer device, as the following figure shows:

Figure 3-3: 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 IDM access control.

The clock domain crossing is within the IP block. Arm assumes that the same clock feeds the interconnect network interface and the IP interface.

When 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 IDM and device discovery.

A specific sequence of events must occur in the system to power up or power down a device in an external power domain. The following sections list the steps involved and the order in which they must occur.

External power domain powerup sequence

The following events must occur in the system to power up a device in an external power domain.

1. The system applies power to the IP domain.
2. The system removes isolation cell clamp values on the AXI boundary.
3. The system applies the IP reset sequence, either through a full system reset or by an IDM soft reset.

   For more information about the IDM soft reset feature, see IDM soft reset mode.
4. The system releases IDM isolation.
5. Configuration or mission access to IP occurs.

External power domain power down sequence

The following events must occur in the system to power down a device in an external power domain.

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

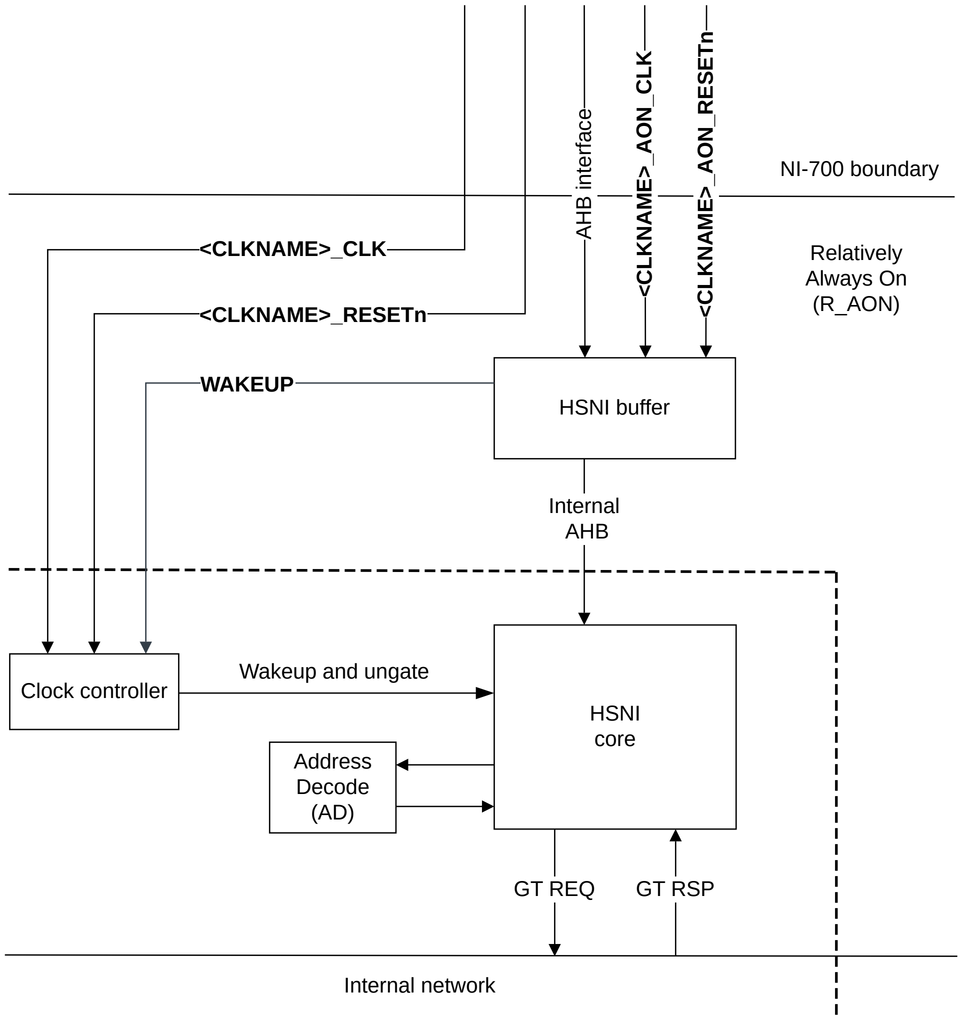
3.3.3 AHB address phase buffering in HSNIs

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

In the AHB protocol, a completer 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 the HSNI is clock gated.

The following figure shows how the HSNI buffer works.

Figure 3-4: 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. So, the clock input can only be removed 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 the buffer and core are assumed to be in the same clock domain. If the buffer and core are not in the same clock domain, then transactions are likely to be dropped.

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

3.4 Clock and reset control

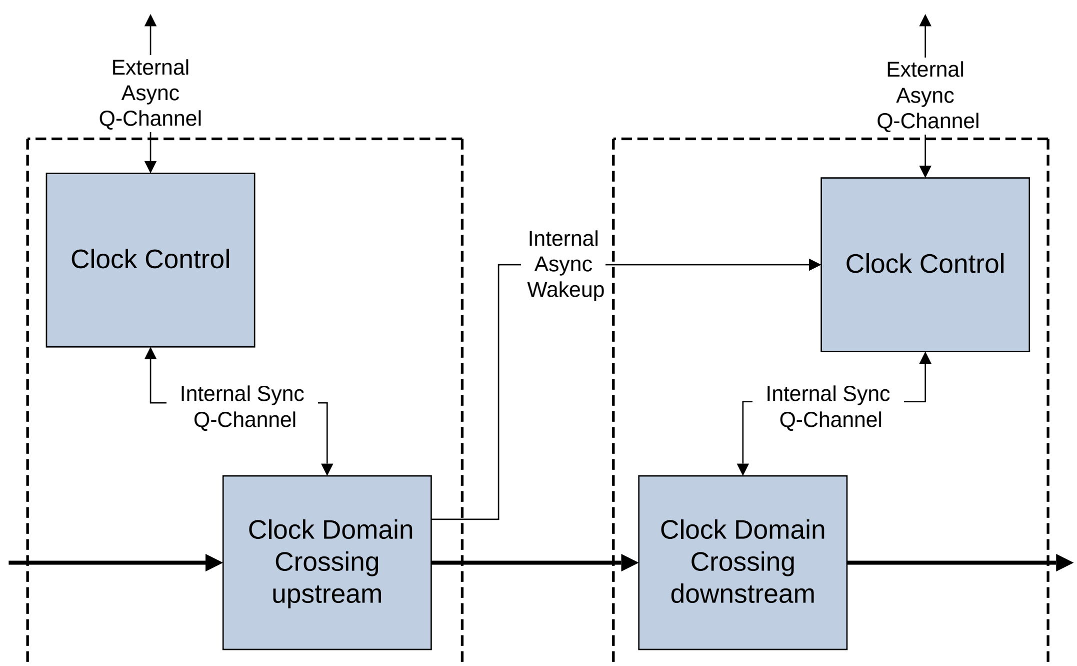
The NI-700 clock and reset control network consists of clock control blocks and several clock control signals.

NI-700 contains one 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. All transactions are stalled at the domain boundary when the Q-Channel is in the Q_STOPPED state.

Clock domains exit reset in the Q_STOPPED state when the domains are logically isolated. Therefore, requests cannot be lost. The full Q-Channel sequence transitioning from Q_STOPPED to Q_RUN must be completed before requests can enter a 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 3-5: NI-700 clock and reset control network



Specific clock control steps are required to enable managed clock domains to move between the ON and OFF states, and to exit from the reset state. For more information, see Clock control sequences and Reset control sequences.

3.4.1 Clock control sequences

The NI-700 clock control network must perform specific sequences of actions to allow downstream clock domains to transition between states.

The following sections list the steps involved in ON to OFF and OFF to ON clock transitions and the order in which they must be performed.

Upstream clock domain ON, downstream clock domain ON→OFF

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

1. The downstream external QACTIVE signal is driven LOW, indicating that all activity within the clock domain is complete.
2. The external QREQn signal goes LOW.
3. The internal QREQn signal to the CDC goes LOW.
4. If there is no activity in the CDC:
    a. The CDC performs logical isolation of the boundary and issues the QACCEPTn signal to the clock control block.
    b. The clock controller forwards the QACCEPTn signal to the external interface.
    c. The clock is gated externally, if necessary.
5. If there is activity when the internal QREQn signal is received:
    a. The CDC asserts the internal QACTIVE signal.
    b. The CDC issues the internal QDENY signal.
    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 to ON when the upstream clock domain is ON.

1. A new upstream transaction arrives in the CDC.
2. The upstream CDC asserts an internal wakeup signal to the downstream clock controller.
3. The downstream clock controller asserts the external QACTIVE signal asynchronously.
4. The clock signal is restored externally to the downstream clock domain.
5. The external QREQn signal goes HIGH.
6. The internal QREQn signal to the CDC goes HIGH.
    a. The CDC removes logical isolation of the clock domain boundary and issues a QACCEPTn transition to the clock controller.
    b. The clock controller forwards the QACCEPTn transition to the external interface.

Note

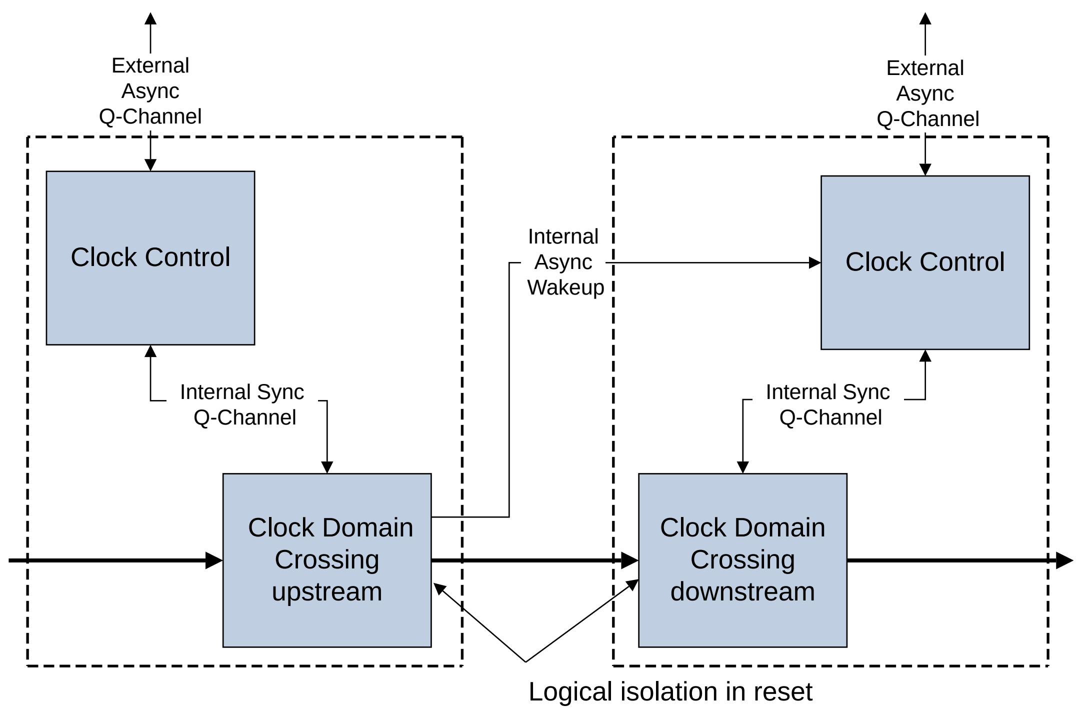
NI-700 does not deny requests to enter a higher clock state, such as a transition from OFF to ON.

3.4.2 Reset control sequences

A specific sequence of actions must occur to permit a clock domain to exit from the reset state. The sequence differs depending on whether the upstream or downstream clock domain exits reset first.

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

Figure 3-6: 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. The upstream clock domain completes a clock and power handshake to permit operation.
2. A new transaction arrives at the upstream CDC.

3. The downstream clock domain is in reset (Q_STOPPED state). So, it now follows the same flow as when the upstream clock domain is ON and the downstream clock domain transitions from OFF to ON. For more information, see Clock control sequences. However, 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. The downstream clock domain completes a clock and power handshake to permit operation.
2. The upstream clock domain does not issue transactions until it is out of reset. The downstream clock domain now functions as if in normal operation. It awaits transactions, which can be forwarded after the upstream clock domain exits reset and completes the external clock and power handshake.

4. Component and interface identifiers

Each NI-700 network interface and external interface has a unique identifier, which is used to ensure that packets are routed correctly.

Different types of identifiers are used for NI-700 components and external interfaces.

Node IDs

When you build a NI-700 configuration, each completer and requester network interface node is assigned a unique node ID. NI-700 uses these node IDs for packet routing. Node IDs are also used by the software discovery process to detect the programming region for each node.

Because each node is also identified by its node type, the node ID spaces of completer and requester network interface nodes can overlap.

Interface IDs

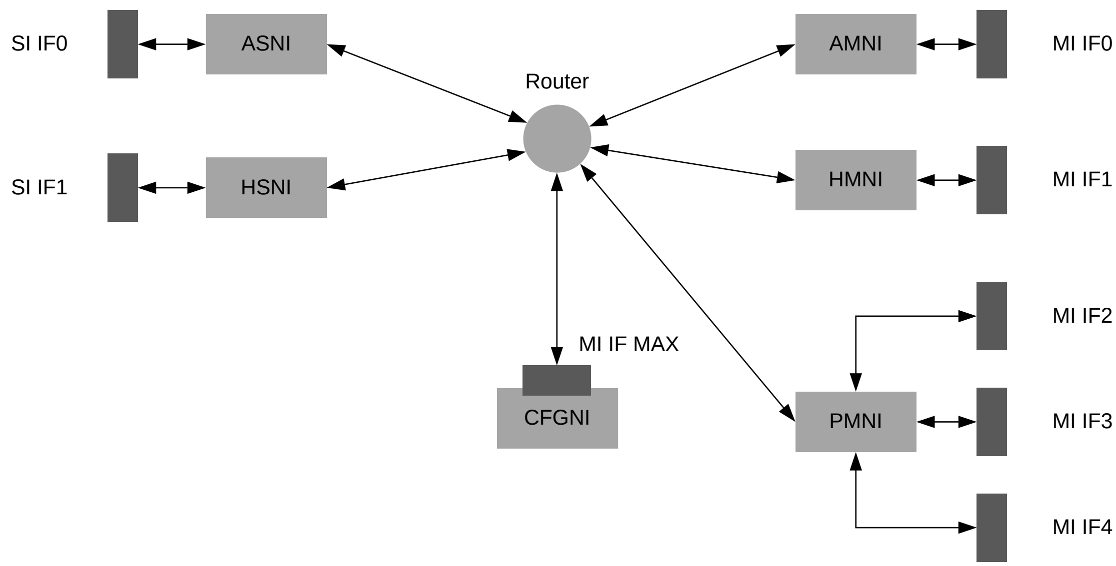
Each external interface contains a unique interface ID. NI-700 uses these interface IDs to route each packet to the correct external interface destination or CFGNI. Transaction requests include a Target ID (TgtID) that identifies the target node and a Source ID (SrcID) that identifies the source node. The interface ID is the SrcID and TgtID for a packet within NI-700.

Interface ID values are assigned in two unique pools:

- Downstream network interfaces (xSNI)
- Target network interfaces (xMNI)

The following diagram shows representative interface IDs in NI-700 and is for illustrative purposes only.

Figure 4-1: Representative requester and completer unique interface IDs



On requester network interfaces, the output ID is derived from two other values. For more information, see Calculation of output IDs.

The width of requester interface output IDs is always reduced according to the maximum ID widths that are used by the completer interfaces. For more information, see ID reduction.

4.1 Calculation of output IDs

The output ID on NI-700 requester interfaces is a function of the input AxID value on the completer interface and the number of completer interfaces.

The SrcID is related to the number of ASNIs. The input ID width is the largest AxID width configured on the ASNIs.

NI-700 does not modify the incoming AxID value. Each NI-700 completer interface is assigned a SrcID to identify the interface that originated a transaction. The SrcID is concatenated with the input AxID.

The SrcID of the incoming request is captured in the node_id field of the ASNI_NODE_TYPE register. For more information on the ASNI_NODE_TYPE register, see ASNI_NODE_TYPE, Node type register for ASNI registers.

4.2 ID reduction

NI-700 always applies ID reduction to the output ID issued from requester interfaces, rather than using the largest AxID and SrcID values in the system.

The output ID width at the AMNIs is <AxID><SrcID>. ID reduction is based on the completer interfaces that have a valid path to a particular requester interface.

The NI-700 tooling determines the maximum AxID width that is used by the completer interfaces to access a requester interface. This width becomes the reduced AxID width that is used to set part of the output ID width.

For the same requester interface, the tooling also determines the largest SrcID width that is used by a completer interface that can access the requester interface. This value sets the SrcID width that is used to calculate the output ID width.

5. Protocol and data width conversion

NI-700 can connect requester and completer devices together that support different AMBA protocols and data widths, and which have different clocking and power requirements. So, NI-700 implements functionality to support a wide variety of external devices.

The NI-700 AXI5 to AHB5 bridge translates AXI exclusive bursts and exclusive transactions into transfer types that are supported by the AHB protocol. For more information, see Exclusive and locked accesses.

According to the *Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite*, requesters that require locked transfers must assert HMASTLOCK, while there is no requirement for completers to implement this signal. So, NI-700 HMNIs and HSNIs respond differently to locked transfers. For more information, see AHB locked transfers.

NI-700 AMNIs include data width upsizing and downsizing functions to support AXI transfers between devices with different data widths. For more information, see Upsizing AXI and ACE-Lite data width function and Downsizing AXI and ACE-Lite data width function.

AXI and AHB interfaces can include an optional set of user-defined signals. These User signals can be used to provide extra information about transactions that is not defined in the AMBA specifications. For more information, see User signals.

The NI-700 SERDES unit resizes, splits, and collates flits to enable flits to move between regions with different link widths. For more information, see Flit resizing and collating.

NI-700 AXI network interfaces include the options to support Memory Tagging Extension (MTE) tags. For more information, see Memory tagging support.

The NI-700 network can be configured as a bridge that crosses between different clock frequencies. For more information, see Network FIFO and clocking function.

NI-700 AXI and AHB network interfaces can be configured to transport data parity, ECC, and poison information. For more information, see Transporting data parity, ECC, and poison information.

5.1 Exclusive and locked accesses

The AXI protocol supports exclusive bursts, but the AHB protocol only supports single (length 1) exclusive transfers. To account for this difference, the AXI5 to AHB5 bridge handles AXI exclusive bursts and single AXI exclusive transactions differently.

AXI exclusive accesses and AHB exclusive transfers are a read transaction followed by a write transaction to the same address range. AXI exclusive bursts are similar except that the read and write transactions comprise sequences of transfers. Exclusive accesses and bursts allow for semaphore-type operations without requiring the bus to remain dedicated to a particular requester throughout the operation.

Unlike AXI exclusive accesses, AHB exclusive transfers must be single beat transfers. So, if the AXI5 to AHB5 bridge receives an AXI exclusive burst, it translates the burst to normal (non-exclusive) AHB transfers. When 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. The bridge returns SLVERR because although OKAY is a valid exclusive response, an OKAY response could cause the AXI requester to repeat the exclusive write indefinitely.

The bridge uses the AXID values to identify the AXI requester that is issuing an exclusive access. For the AHB transfer, the bridge copies the AXID value to HMASTER.

The following table shows the AHB transfer types to which the AXI5 to AHB5 bridge maps different AXI exclusive accesses.

Table 5-1: AXI5 to AHB5 bridge exclusive access mapping

Received AXI access type	Received AXI transaction type	Translated AHB transfer type
AXI exclusive read	Single	Exclusive AHB transfer
AXI exclusive read	Burst	Normal AHB transfers
AXI non-sparse exclusive write	Single	Exclusive AHB transfer
AXI non-sparse exclusive write	Burst	Normal AHB transfers
AXI sparse exclusive write	Single	Normal AHB transfer (SLVERR)
AXI sparse exclusive write	Burst	Normal AHB transfers

5.2 AHB locked transfers

HSNIs and HMNIs behave differently when handling AHB locked transfers.

In the AHB protocol, locked transfers are sequences that are indivisible and must be processed before any other transfers are processed. Typically, locked transfers are used to ensure that a completer does not perform other operations between the read and write phases of an instruction. Requesters that require locked accesses must assert the HMASTLOCK signal.

At an HSNI, HMASTLOCK is ignored for the HSNI. At an 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

Although an AXI burst can cross a 1KB address range, the AHB protocol requires that all transfers in a locked sequence go to the same completer address region. If an HMNI receives a non-modifiable burst with a size of more than 1KB, the burst is sent as a non-modifiable AHB burst. However, HMASTLOCK is not asserted and the response is sent with SLVERR.

5.3 Upsizing AXI and ACE-Lite data width function

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

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

Upsizing only packs write data for write or read transactions that are cacheable. There are several packing rules for different burst types and acceptance capabilities to consider. Aligned and unaligned input bursts are defined as follows:

Aligned input burst

The address is aligned to the output data width boundary after the network aligns the address to the transfer size.

Unaligned input burst

The network does not align the address to the output data width boundary, even after the network aligns the address to the transfer size.

The following transaction rules apply to upsizing:

- If a transaction passes through, the upsizing function does not change the input transaction size and type.
- If the network splits input exclusive transactions into more than one output bus transaction, the exclusive information is removed from the multiple transactions that the network 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

The network upsizes different bursts as follows:

- The network converts INCR bursts into the optimum size based on the output data width. For more information, see Upsizing INCR bursts.
- The network either passes WRAP bursts through unconverted or converts WRAP bursts into INCR bursts. For more information, see Upsizing WRAP bursts.
- The network passes all FIXED bursts through unconverted.

5.3.1 Upsizing 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.

INCR<n> indicates an incrementing burst with n data beats. Bursts are never merged.

The following table shows how the network converts INCR bursts when it upsizes them. In this example, the input data width is 64 bits and the output data width is 128 bits, unless otherwise stated.

Table 5-2: 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

5.3.2 Upsizing WRAP bursts

The network either passes WRAP bursts through unconverted, or converts WRAP bursts to one or two INCR bursts on the output bus.

Input WRAP bursts with a total payload that is less than the output data width are converted to single INCRs.

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

Table 5-3: 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

5.4 Downsizing AXI and ACE-Lite data width function

NI-700 supports transactions from AMNI and ASNI devices with different data widths. The AMNI is responsible for downsizing data that is 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 a Non-cacheable transaction, the downsizing function does not merge data narrower than the destination bus.

5.4.1 Downsizing 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 5-4: 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.

5.4.2 Downsizing 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.

5.4.3 Downsizing 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 5-5: 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.

5.5 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 5-6: 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 5-7: 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 and AHB interfaces.
Legacy mode	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.	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 and AHB interfaces.
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 network interfaces, that is, ASNIs, AMNIs, HSNIs, and HMNIs. 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)
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.	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 network interfaces, that is, ASNIs, AMNIs, HSNIs, and HMNIs. 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)

User signal widths

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

Table 5-8: 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.
AXI	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.
AXI	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.	–
AXI	WUSER	`USER_DATA_WIDTH`	See the note in the preceding User signal mode behavior table for constraints on `USER_DATA_WIDTH`.
AXI	BUSER	`BUSER_RESP_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.
AHB	HRUSER	`USER_DATA_WIDTH`	See the note in the preceding User signal mode behavior table for constraints on `USER_DATA_WIDTH`.
AHB	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 5-9: Parameters and supported range

Parameter	Supported range
`USER_REQ_WIDTH`	0 bits to 256 bits
`USER_DATA_WIDTH`	When `USER_DATA_MODE` = 0 the supported bit range is 0 bits to 64 bits

Parameter	Supported range
	`USER_DATA_MODE` = 1 Supports between one bit and four bits per byte Max `DATA_WIDTH` = 1024 bits Max `USER_DATA_WIDTH` = (1024 / 8) × 4 = 512 bits
`BUSER_RESP_WIDTH`	0 bits to 64 bits
`RUSER_RESP_WIDTH`	0 bits to 64 bits

5.6 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.

At build time you can configure the number of flits to aggregate and store until the packet is released.

5.7 Memory tagging support

You can enable memory tagging on any AXI network interface in NI-700, by using the MTE_Support parameter.

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 5-10: 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.

Where MTE support in the interconnect is based on the least common support across all the ASNIs, then:

- If none of the ASNIs support MTE, then MTE support in the interconnect is false.
- If at least one of the ASNIs is set to MTE basic, and none of the ASNIs are set to MTE standard, then MTE support in the interconnect is basic.
- If at least one of the ASNIs is set to MTE standard, then MTE support in the interconnect is standard.

5.8 Network FIFO and clocking function

If you configure the network as a clock frequency crossing bridge, then non-blocking Resource Plane (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 as:

- 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.

5.8.1 Clock synchronization modes

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.

5.9 Transporting data parity, ECC, and poison information

NI-700 supports transporting data parity, ECC, or poison information through the interconnect.

This support only applies to AXI RDATA and WDATA data and AHB HRDATA and HWDATA. NI-700 only transports the parity, ECC, or poison information therefore there is no support to generate or check parity or ECC within the interconnect. NI-700 uses the user data bits RUSER, WUSER, HRUSER, and HWUSER to transport parity, ECC, or poison information. For more information on data parity, ECC, and poison, see *Data parity, ECC, and poison information* in the *Arm® CoreLink™ NI-700 Network-on-Chip Interconnect Configuration and Integration Manual*.

For this feature, the system builder must configure USER_DATA_MODE = 1 to support upsizing and downsizing the parity or ECC information in the user data bits appropriately.

For more information on the user data mode, see User signals.

6. Secure and Non-secure accesses

NI-700 supports Secure and Non-secure accesses from request and response sources. The mechanisms that it uses to handle these accesses depend on the AMBA protocol that the source supports.

6.1 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.

6.2 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 completer interface. The following table describes each AHB completer interface security configuration option.

Table 6-1: AHB completer 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 completer 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 register.
Always Secure	At build time all requests which originate from this completer interface are marked as Secure.
Always Non-secure	At build time all requests which originate from this completer interface are marked as Non-secure.

You also have four security configuration options for an AHB requester interface. The following table describes the AHB requester interface security configuration options.

Table 6-2: AHB requester interface security configuration options

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

The following table describes the HSNI and HMNI reset values for the programmable security register.

Table 6-3: HSNI and HMNI 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 completer are considered to be Secure

If a Non-secure transaction targets a requester 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 completer network interface is set to always Non-secure or programmed to be Non-secure, then as defined in Register security attribute and security classification it is not permitted access to Secure registers within NI-700. If the Secure access attribute is overridden as defined in Secure access register, no access to Secure registers occurs.

6.3 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. The following table describes each APB configuration option.

Table 6-4: APB security configuration options

Configuration option	Description
Pin	If the protocol is APB4, the PPROT pin communicates the security attribute to the downstream completer. 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 completer. If the register bit is set to 1, the downstream completer is Non-secure. If the register bit is set to 0, then the downstream completer is Secure. Where the protocol is APB4: • If the register is configured to indicate a Non-secure completer, the security attribute is passed on the PPROT pin. • If the register is configured to indicate a Secure completer, 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 completer with the security attribute communicated on the PPROT pin.
Always Secure	Only Secure transactions can access components that are attached to this specific APB requester 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 requester interface. If the protocol is APB4, the security attribute is communicated on the PPROT pin.

The following table contains the reset value and description for the PMNI programmable security register.

Table 6-5: PMNI programmable security register reset value

Interface	Reset value	Description
PMNI	0	Out of reset all assets in the downstream AHB completer 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. For more information on Secure and Non-secure APB interfaces, see PMNI_CTRL.

When configured as programmable, always Secure or always Non-secure, PMNI is responsible for completing the Secure access permission check. If a Non-secure transaction targets a requester 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 completer. 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

6.4 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 TrustZone technology and security.

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 Secure access register.

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 Secure access register.

Secure Only

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

Non-secure

Accessible by Secure and Non-secure requests.

6.5 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
- Upstream and downstream network interface register regions
- PMU register region

Software can program this register to override the Secure access permissions of any specific register region instance. These register region instances include the upstream network interface and downstream network interface regions.

The Secure access register has two 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 Programmers model.

6.6 Secure debug

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

The performance monitoring events corresponding to each upstream and downstream interface are specified in Performance monitoring. The SPNIDEN, SPIDEN, and DBGEN inputs that are described in Performance monitoring and Secure Debug 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.

6.7 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.

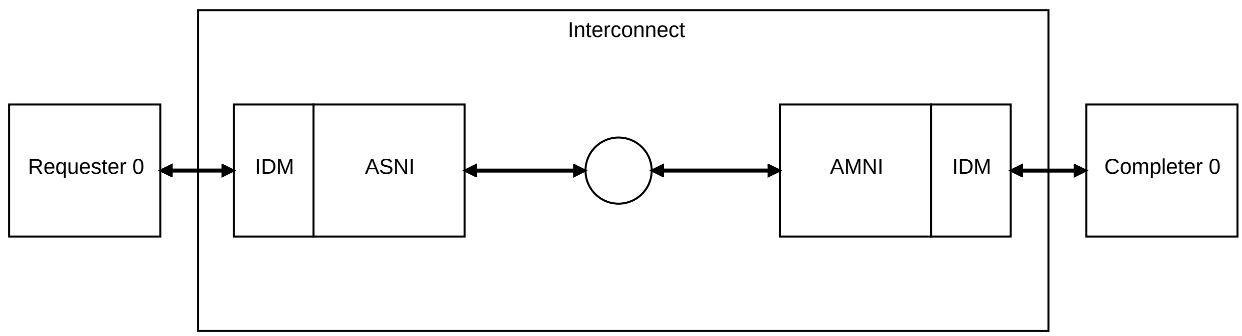
For more information, see Error handling and interrupt security.

7. Interconnect Device Management

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 requester device to ASNI connection. The following figure shows an example system with IDM blocks integrated with ASNI and AMNI components:

Figure 7-1: IDM integration with network interface blocks



Each IDM block on a requester device to requester network interface connection provides control and status of transactions that the requester device issues. Each IDM block on a requester network interface to completer device connection provides control and status of transactions that are issued to the completer 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 network interface.

Both completer and requester 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 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.
    - Standard means that memory tagging is supported on the interface, all MTE signals are present.
    - Basic means that memory tagging is supported on the interface at a basic level. A limited set of tag operations is permitted. BTAGMATCH is not present. BCOMP is not required.

Note

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

IDM and read data chunking features enabled together

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.

Completer interface enters soft reset mode during a write transaction

When a completer 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 *Completer network interface write data transaction timeout leading to a soft reset* in Soft reset use case examples for completer and requester network interfaces. 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.

Adequate timeout value

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 longer delays. Set the timeout value so these functional scenarios are not falsely triggered as misbehaving requesters or completers.

7.1 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 Discovery 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 Device_ID 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 Programmers model.

7.2 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 requester and completer network interfaces.

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

- The external device fails to send a complete write data for a received write address phase transaction. The interconnect can indicate failure at any point in the write data beat count.
- The external device fails to send a write address phase for a write transaction with leading write data.
- The external device fails to accept a write response for an issued write transaction.
- The external device fails to accept all read data beats for an issued read transaction.

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

- The external device fails to accept a read address or write address phase transaction.

- The external device fails to accept a write data beat for a write transaction. The interconnect can indicate failure at any point in the write data beat count.
- The external device fails to send a write response for an issued write transaction.
- The external device fails to send 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 Network Interface IDM registers summary.

Once IDM detects a timeout, if IDM_RESET_CONTROL.auto is asserted the network interface raises a timeout interrupt and enters the soft reset mode:

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

The timeout does not cause the interconnect to start backing up as the network interface completes all outstanding transactions. The network interface remains in this mode until the software requests an exit from this mode using a write to the IDM_RESET_CONTROL.reset register. For more information see, IDM soft reset mode.

7.3 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 requester and completer network interfaces.

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 Network Interface IDM registers summary.

7.4 IDM soft reset mode

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 requester and completer network interfaces.

Use soft reset together with either or both of the error logging or timeout detection features. For more information, see Timeout detection through IDM block and Error logging through IDM block.

IDM soft reset mode consists of two distinct stages:

- Recovery: The network interface gates its external interfaces. Any transactions that were outstanding when soft reset mode is entered are completed in a protocol-compliant fashion by synthesizing remaining transfers. The endpoint synthesizes transfers to complete any new transactions when in recovery mode.

- Soft reset assertion: The external soft reset pin associated with the device connected to the timed-out interface is asserted.

A write of 1 to the IDM_RESET_CONTROL.reset field causes an entry into soft reset mode. If the entry into this mode is not because of an auto entry, the external soft reset pin is activated. A write of 0 to the IDM_RESET_CONTROL.reset field when it is already 1 causes an exit from soft reset mode and the deassertion of the external soft reset pin. Recovery mode occurs when timeout occurs and IDM_RESET_CONTROL.auto = 1, or by writing 1 to the IDM_RESET_CONTROL.reset field.

To enter soft reset assertion, write 1 to the IDM_RESET_CONTROL.reset field.

7.4.1 Hardware initiated entry based on timeout detection

If a timeout is detected, NI-700 enters soft reset mode.

For more information on timeouts, see Timeout detection through IDM block.

In soft reset mode, the relevant network interface immediately asserts the timeout if enabled, and logs errors into the IDM registers. If IDM_RESET_CONTROL.auto = 1, then the network interface enters the recovery stage. If there are also outstanding requests at the time, NI-700 receives the soft reset mode request, the network interface block completes the transactions in a protocol-compliant manner, and the external interface is gated. To comply with protocol, the network interface block generates the required remaining portions of the transaction.

On timeout detection at a requester network interface, the network interface enters soft reset mode if IDM_RESET_CONTROL.auto = 1. In this case, the soft reset pin is not asserted (activated). Therefore:

- 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 completer network interface, the network interface enters soft reset mode if IDM_RESET_CONTROL.auto = 1. In this case, the soft reset pin is not asserted (activated). Therefore:

- 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 soft reset mode does not affect the soft reset pin. To toggle the external soft reset pin, software must request soft reset mode by writing 1 to the IDM_RESET_CONTROL.reset field. If the endpoint is already in soft reset mode, a write of 1 to the IDM_RESET_CONTROL.reset field asserts the external soft reset pin to the device.

When in soft reset mode, the network interface remains in this mode until software initiates an exit from the mode. Exit occurs when there is a write of 0 to the IDM_RESET_CONTROL.reset field.

Writing 0 to the IDM_RESET_CONTROL.reset field not only exits soft reset mode, but also deasserts the external soft reset pin.

For more information on exiting soft reset mode, see IDM_RESET_CONTROL.

For more information on timeouts, see Timeout detection through IDM block.

7.4.2 Software initiated entry

Under software control, NI-700 resets the requester or completer 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 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 requester or completer on reset. For more information on IDM registers, see Network Interface IDM registers summary.

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

With software initiated entry, the relevant network interface also toggles the external reset pin to the attached device.

The network interface remains in soft reset mode until software initiates a write of 0 to the IDM_RESET_CONTROL.reset field to exit soft reset mode. Writing 0 to the IDM_RESET_CONTROL.reset field exits soft reset mode and also deasserts the external soft reset pin. For more information on exiting soft reset mode, see IDM_RESET_CONTROL.

7.4.3 IDM_RESET_CONTROL reset initialization input pin

When an endpoint has IDM enabled, it receives an external input pin, device_sreset_strap_i, that is connected to the IDM_RESET_CONTROL.reset field. The external input pin only controls the IDM_RESET_CONTROL.reset field of the register.

If the pin value is set to 0, then there is no change in behavior out of reset. However, if the pin value is set to 1, when the endpoint exits soft reset it behaves as if it is already in soft reset mode. This behavior is exactly as if software wrote 1 to the IDM_RESET_CONTROL.reset field to enter soft reset mode. Therefore:

- 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

For more information on the IDM_RESET_CONTROL field names and bit assignments, see IDM_RESET_CONTROL.

7.5 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 completer and requester network interfaces.

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 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 requester or completer on isolation. For more information, see Programmers model.

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

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

1. First, software must request isolation entry using the IDM_ACCESS_CONTROL register.
2. When isolation entry is successful, software requests a soft reset entry using the IDM_RESET_CONTROL register.

For a completer network interface, isolation means the network interface does not send incoming requests into the interconnect. For a requester network interface, isolation means the network interface does not send incoming requests to the downstream completer. Any new requests that are 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 completer or requester network interface 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 loopback 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 loopback mode at different times after receiving an isolation entry request. However, the IDM_ACCESS_CONTROL reflects isolation entry only after both the read and write channels have entered isolation mode. As a result, either channel can receive loopback 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.

For more information on the transaction flow, see Access control use case example for requester and completer network interfaces.

7.6 Soft reset use case examples for completer and requester network interfaces

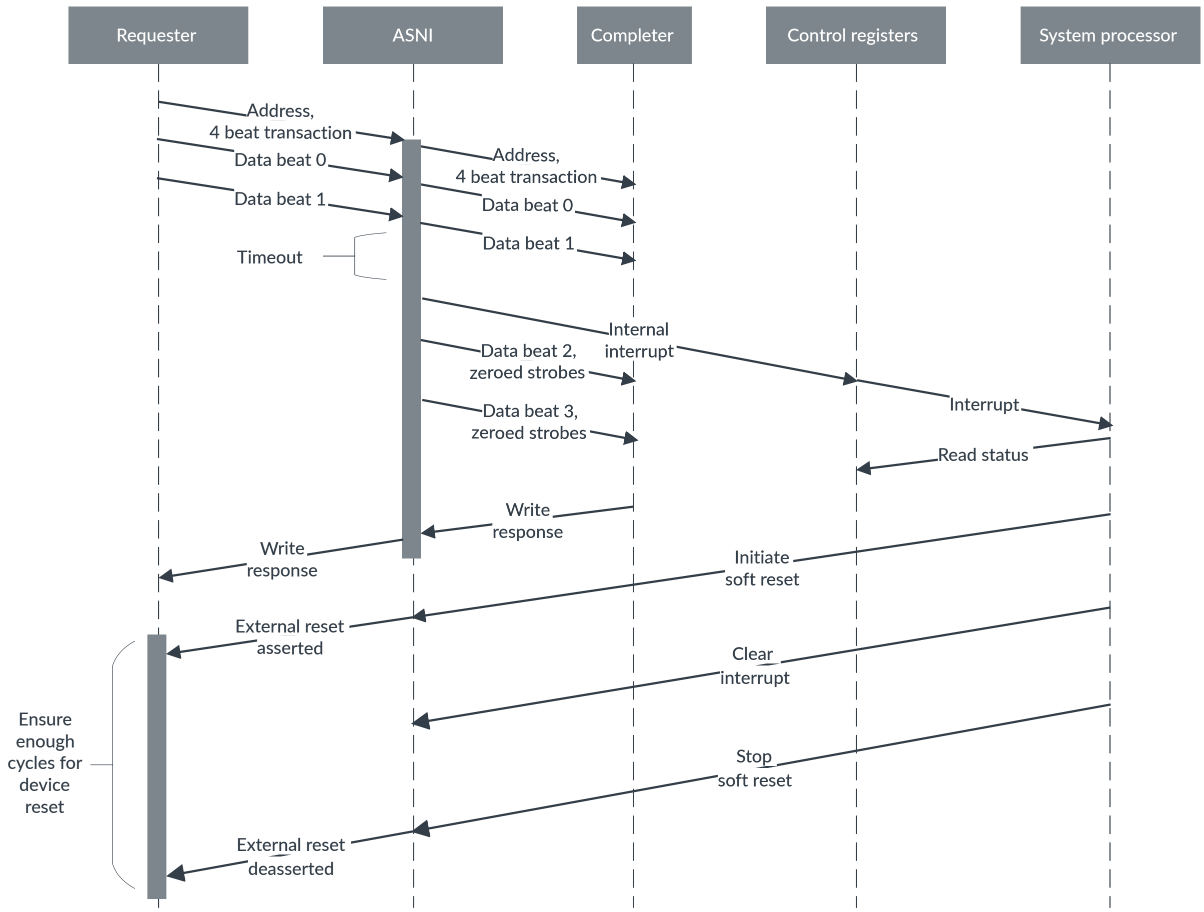
These examples show the expected use case for the soft reset functionality for a stalled read transaction and a write data transaction at a completer network interface. The examples also show a requester network interface write transaction timeout and a requester network interface read transaction timeout, both leading to a soft reset.

Completer network interface write data transaction timeout leading to soft reset

In this example, the requester device has stalled after issuing the second of four write data beats. On detection of the timeout, hardware automatically enters soft reset mode (if the IDM_RESET_CONTROL.auto field is set to 1) to gate the external interface. The hardware also 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 1 to the IDM_RESET_CONTROL.reset field to initiate the soft reset sequence for the endpoint, the external reset pin is asserted. This assertion resets the attached offending completer device.

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

Figure 7-2: ASNI write data transaction timeout leading to a soft reset

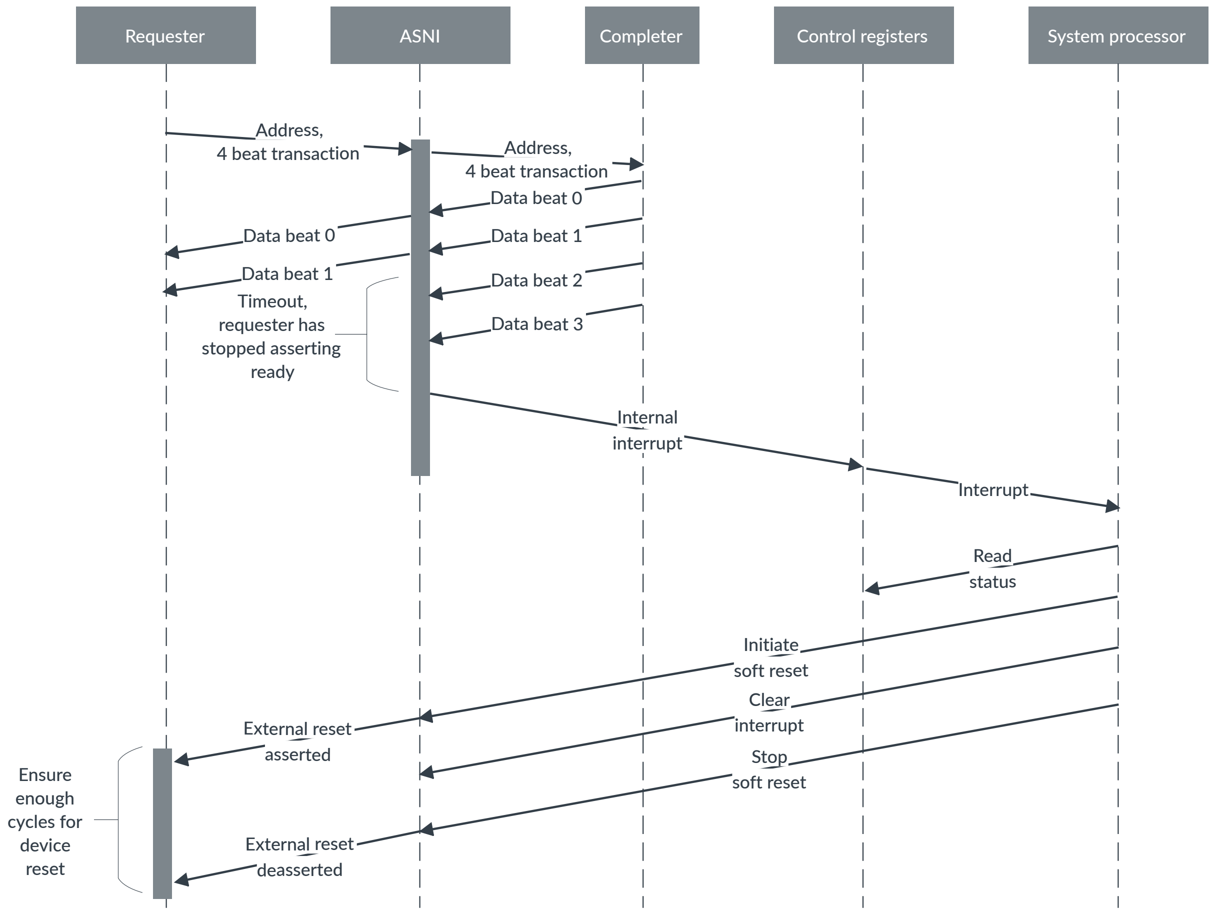


Completer network interface stalled read transaction leading to soft reset

This example demonstrates the expected use case for the soft reset functionality for a read transaction at a completer network interface. In this example, a requester device stops accepting read data beats after the second data beat. After the programmed timeout value, an interrupt is asserted for software to handle. On detection of the timeout, hardware automatically enters soft reset mode (if the IDM_RESET_CONTROL.auto field is set to 1) to gate the external interface. The outstanding read data beats are sunk internally and the request is completed. After the request is completed the software writes 1 to the IDM_RESET_CONTROL.reset field to assert the external SRESETn output.

The following figure shows the ASNI stalled read transaction, leading to a soft reset.

Figure 7-3: ASNI stalled read transaction leading to a soft reset

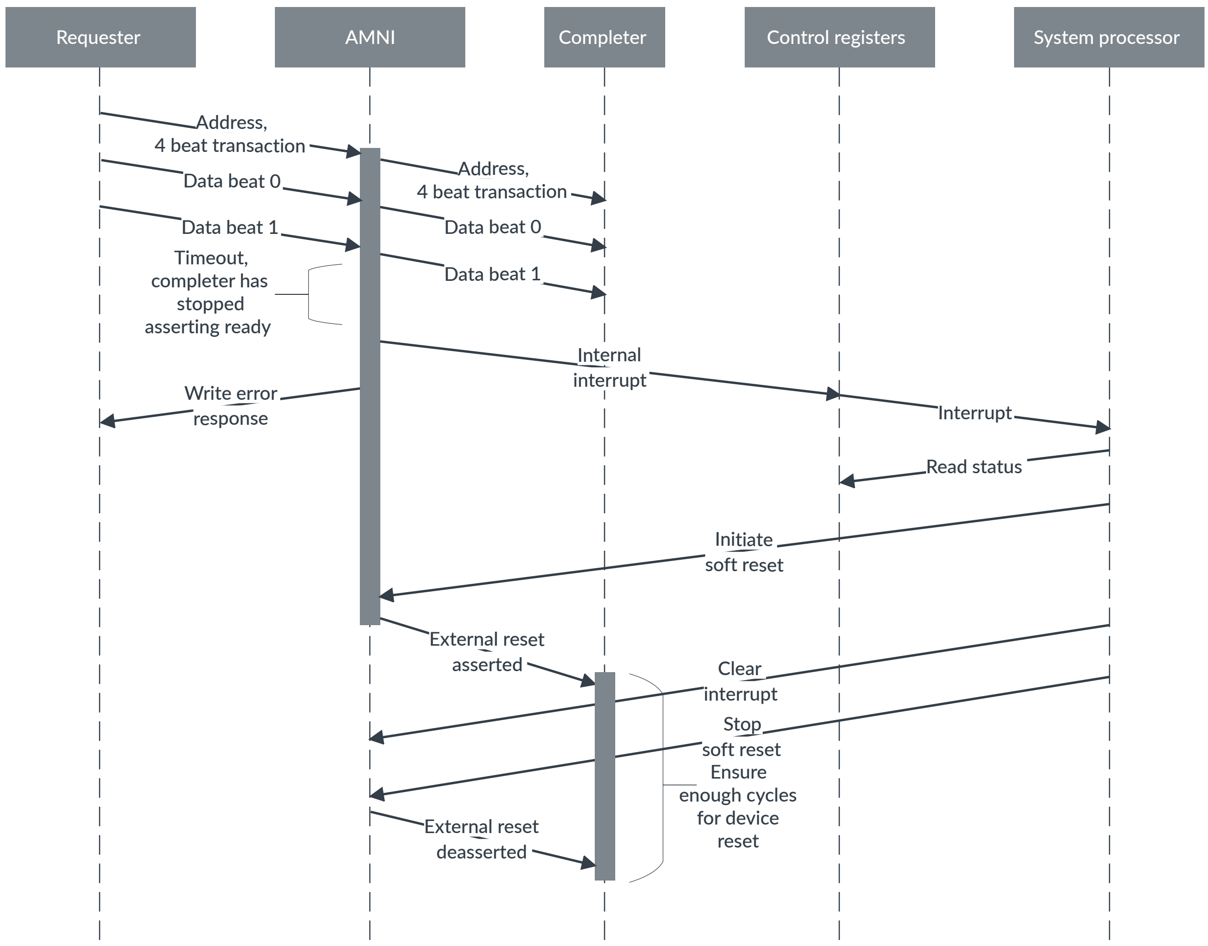


Requester network interface write transaction timeout leading to soft reset

The next example demonstrates an expected use case for the soft reset functionality for a write transaction at an AMNI. In this example, a completer device has stopped accepting write data beats after the second data beat. After the programmed timeout value, an interrupt is asserted for software to handle. On detection of the timeout, hardware automatically enters soft reset mode (if the IDM_RESET_CONTROL.auto field is set to 1) 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 1 to the IDM_RESET_CONTROL.reset field to initiate the soft reset sequence for the endpoint, an external reset pin is asserted. This assertion resets the attached offending completer device.

The following figure shows an AMNI write transaction timeout, leading to a soft reset.

Figure 7-4: AMNI write transaction timeout leading to a soft reset

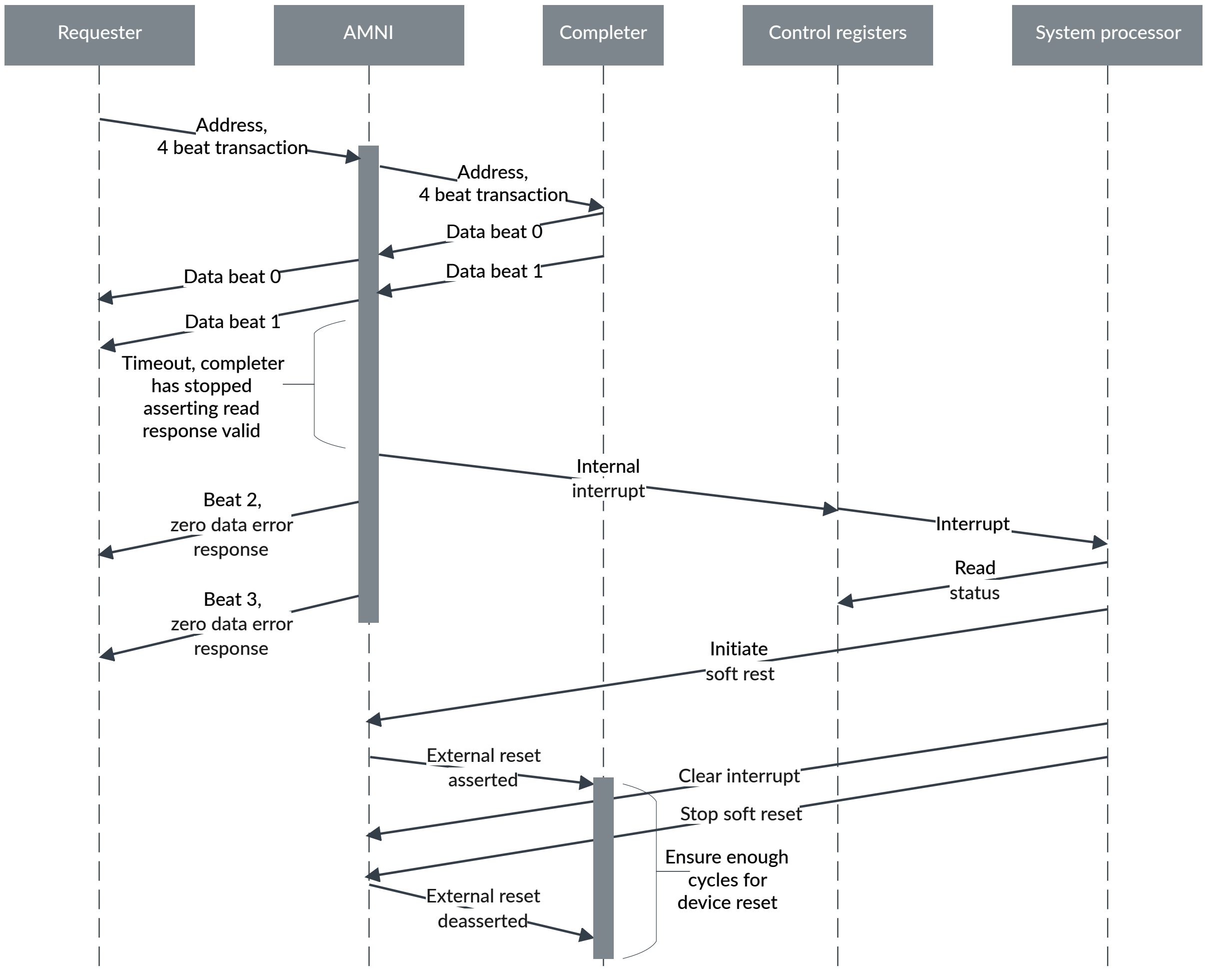


Requester network interface read transaction timeout leading to soft reset

This example demonstrates an expected use case for the soft reset functionality for a read transaction at an AMNI. In this example, a completer device has stopped issuing read data beats after the second data beat. After the programmed timeout value, an interrupt is asserted for software to handle. On detection of the timeout, hardware automatically enters soft reset mode (if the IDM_RESET_CONTROL.auto field is set to 1) to gate the external interface. The outstanding read data beats are synthesized with zero data and with an error response. After the software writes 1 to the IDM_RESET_CONTROL.reset field to initiate the soft reset sequence for the endpoint, the external reset pin is also asserted. This assertion resets the attached offending completer device.

The following figure shows an AMNI read transaction timeout leading to a soft reset.

Figure 7-5: AMNI read transaction timeout leading to soft reset



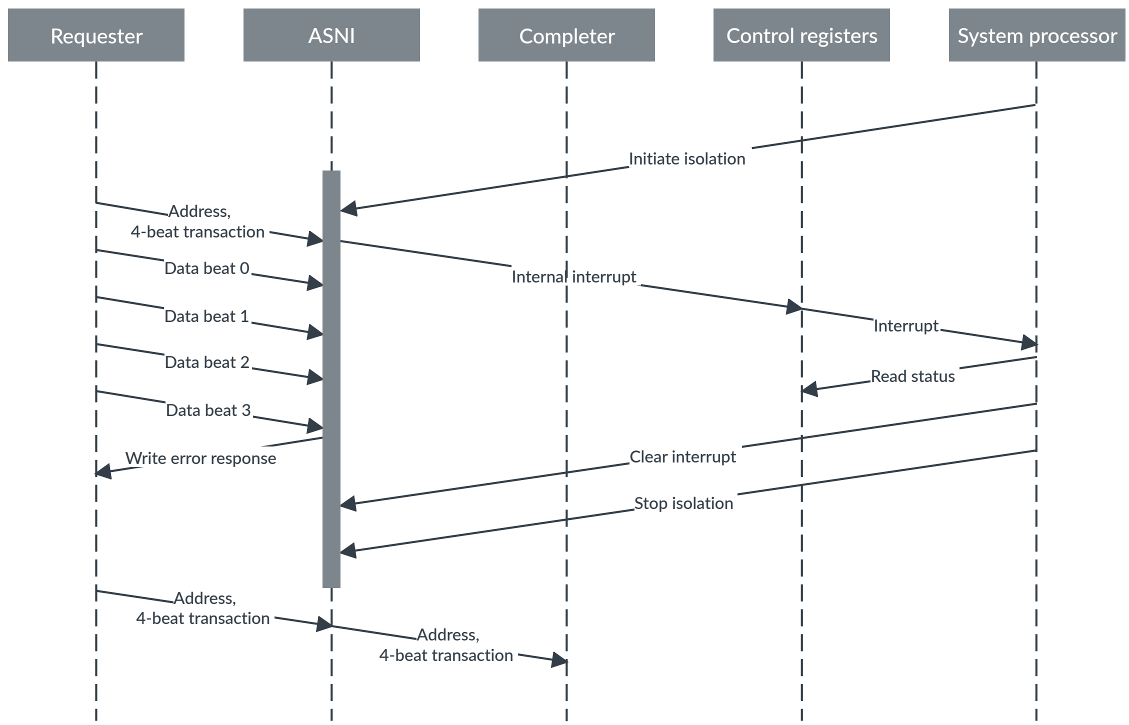
7.7 Access control use case example for requester and completer network interfaces

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

For this example, a requester 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 figure shows a completer interface write transaction arriving during isolation state:

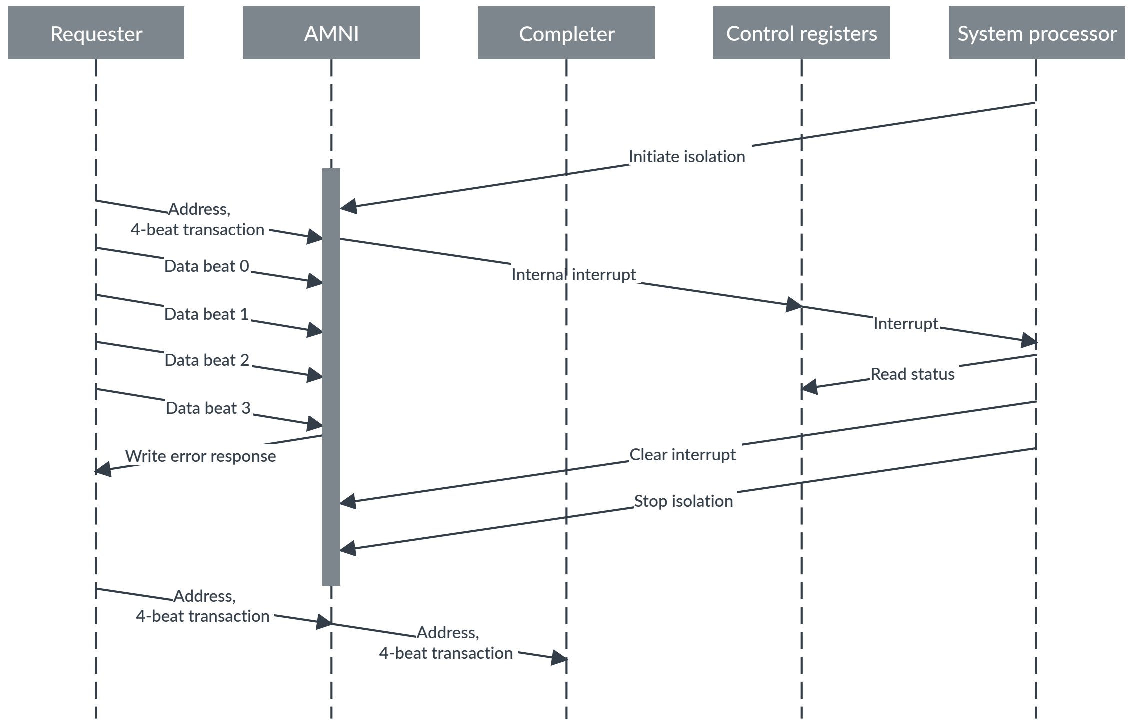
Figure 7-6: ASNI write transaction arriving from an isolated requester



This example demonstrates the expected use case for the access control functionality for a write transaction at a requester interface. In this example, a completer device has been isolated from the interconnect and the AMNI connected to the completer 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 figure shows the requester interface write transaction arriving during isolation state:

Figure 7-7: AMNI write transaction arriving from an isolated completer



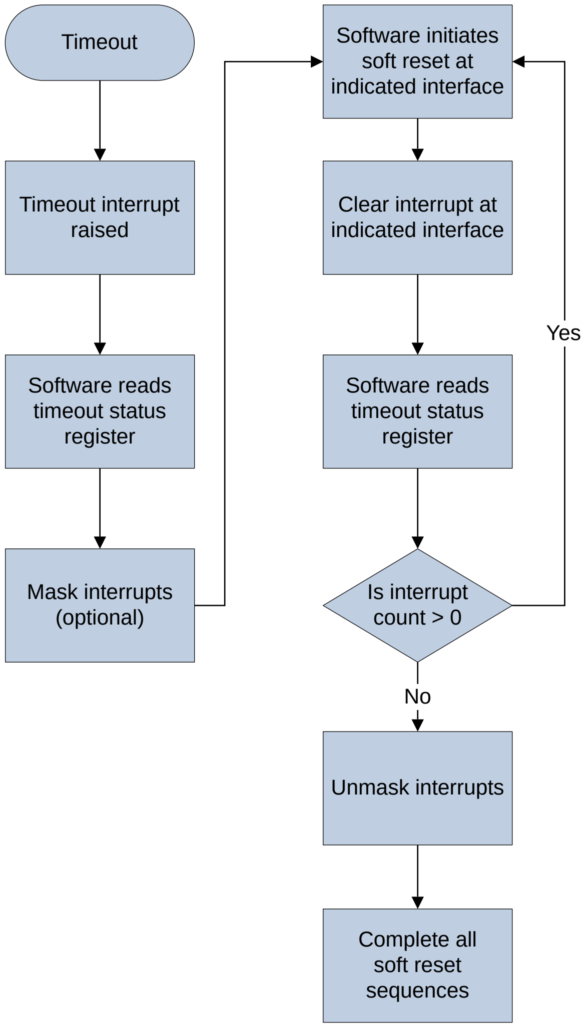
7.8 Example interrupt handling sequence

This example demonstrates the interrupt handling sequence when an interface timeout interrupt is asserted.

The software can 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 figure demonstrates a sample interrupt handling sequence.

Figure 7-8: Interrupt handling sequence



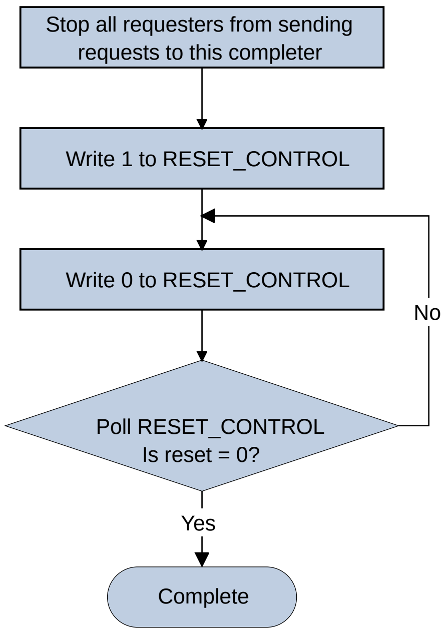
7.9 Soft reset sequence

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

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

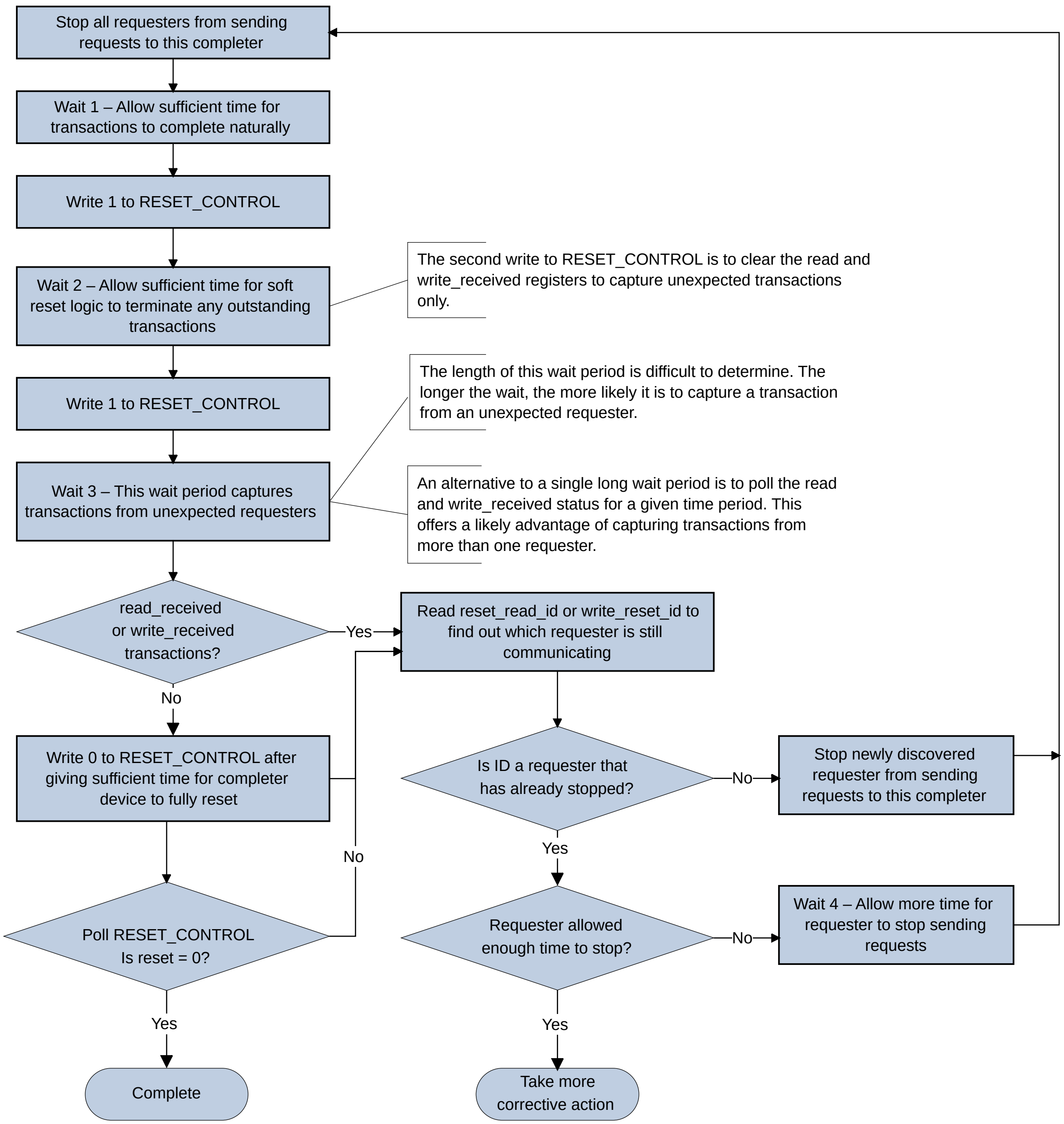
The following figure shows the soft reset sequence to place a completer device into soft reset.

Figure 7-9: Completer device soft reset sequence



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

Figure 7-10: Software using a more cautious interrupt handling sequence



Similar software sequences can apply to IDM access control.

8. Address decode and mapping

When a requester device generates a request, the address is presented to the connected completer network interface. The completer network interfaces have address decoders, which decode the address and maps the request to a target ID for correct transaction routing.

8.1 ASNI address decode

When an AXI requester 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.

8.2 HSNI address decode

When an AHB requester 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.

8.3 PMNI address decode

When an AXI or AHB requester 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 PMNI, 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 has address regions defined for each APB interface behind every PMNI instance.

8.4 Address striping

NI-700 supports transaction address striping. The ASNIs handle 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 the following address stripe granule sizes:
  - 128 bytes
  - 256 bytes
  - 512 bytes
  - 1024 bytes
  - 2048 bytes
  - 4096 bytes
- 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 AMNIs or HMNIs.
- All AXI and ACE-Lite properties must be the same for all the AMNIs that are part of the same stripe group.
- All AHB properties must be the same for all the HMNIs that are part of the same stripe group.
- PMNIs 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 one or both of the following options:
  - Two different stripe groups with different stripe granularities
  - A different number of stripe targets
- 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].

8.5 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 Programmers model. 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 controllers.

A remap state can:

- Change the target requester 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

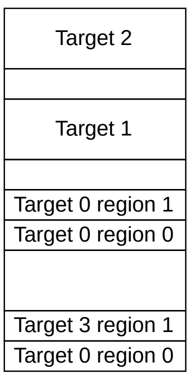
- Add an address region

Because of the nature of the distributed register subsystem, the controllers 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.

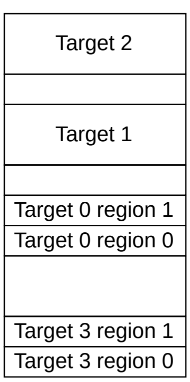
While NI-700 can support up to eight remaps, consider an example configuration that uses three remap bits. The following figure shows the memory map when you set the remap value to `0b000`, representing no remap.

Figure 8-1: No remap, remap set to `0b000`.



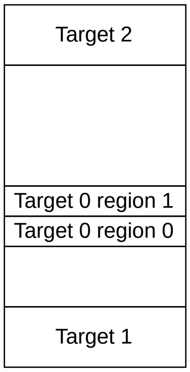
In the following figure, there is a default memory map that divides target 0 and target 3 into two separate regions. In this example, you can choose to set up a remap value whereby target 3 is aliased over target 0, using the remap code 001. At powerup, target 0 region 0 is aliased over target 3 region 0. After powerup, the target 0 region 0 alias is removed as shown.

Figure 8-2: Remap set to `0b001`.



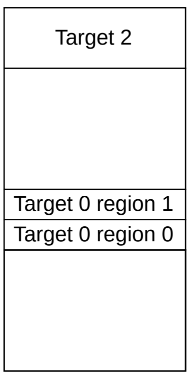
Alternatively, you might decide to move target 1 to the bottom of the address range by setting remap to 010 as the following figure shows.

Figure 8-3: Remap set to `0b010`.



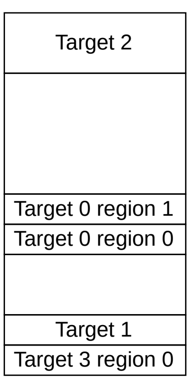
You can choose to remove entire target regions. The following figure shows that if you set remap to 100, target 3 is removed.

Figure 8-4: Remap set to `0b100`.



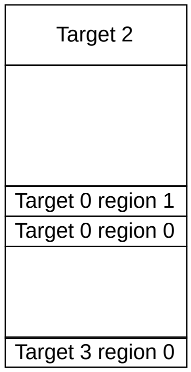
Remap bit 0 still takes precedence if you set it as the following figure shows.

Figure 8-5: Remap set to `0b011`.



In addition, you can choose to remove entire memory regions. The following figure shows that if you set remap to 101, target 3 and target 1 are removed.

Figure 8-6: Remap set to `0b101`.



Caution

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. 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.

Therefore, the default target of the configuration address region from at least one ASNI or HSNI, must point to the configuration target. No address region can map to the configuration target except the configuration address region aligned with PERIPHBASE. The default and remap targets of the configuration address region can only be one of two values, configuration target or no target. To program the ADDR_REMAP register in ASNI or HSNI to choose a specific remap, see ASNI_ADDR_REMAP, Address remap vector register and HSNI_ADDR_REMAP, Address remap vector register.

9. Transaction tracking and ordering

A transaction deadlock can occur when routing multiple transactions concurrently to multiple completers from a point of ingress to the interconnect, such as a completer interface. To prevent such a deadlock, each NI-700 ASNI uses one or both of a configurable reorder buffer and a single completer for each ID Cyclic Dependency Avoidance Scheme (CDAS) mechanism.

ASNIs use the following mechanisms to prevent transaction deadlocks:

- A configurable Transaction reorder buffers
- A single completer for each ID Cyclic Dependency Avoidance Scheme mechanism

9.1 Transaction reorder buffers

To prevent issues when reordering AXI transactions, the ASNI supports transaction reorder buffers. NI-700 contains a write response reorder buffer, which is always present, and an optional configurable read data reorder buffer.

Write response reorder buffer

To improve performance, a write response reorder buffer is always present for write transactions.

Read data reorder buffer

You can configure an optional read data reorder buffer of 1–255 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 a single completer for each ID.

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. In this case, the number of entries reserved is equal to the length of the transaction that reuses the ID.

Read reorder buffer slots are also used to merge partial read responses. This process occurs when read response data beats come from requester network interfaces that have a data width 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. Merging these responses creates a full sized data beat at the ASNI. One read reorder buffer slot is reserved for each transaction outstanding at a requester network interface with a smaller data width.

9.2 Cyclic Dependency Avoidance Scheme

The AXI protocol permits reordering of transactions, therefore it can 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.

The same CDAS mechanism operates independently for read and write transactions.

9.2.1 Single completer for each ID

A single completer for each ID ensures that for an ASNI, specific transactions go to the same destination.

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 a single completer for each ID.

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 completer for each 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.

9.2.2 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 the interface works in a single completer ID mode.

If consecutive writes happen to go to the same target, then the interface 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 the interface waits to receive the BRESP signal before it sends the write.

10. Traffic arbitration schemes

To ensure optimal service is provided to all requesters and completers in a system configuration with a shared interconnect, NI-700 provides configurable traffic arbitration schemes. These schemes arbitrate between different traffic sources and traffic types and help to manage access to shared system components and paths.

NI-700 supports assignment of traffic generators to different Resource Planes (RPs). The network manages the access of each RP to shared system resources so that the RPs do not block each other. For more information, see Resource planes.

NI-700 also uses Quality of Service (QoS) values to arbitrate between traffic of different priority values. For more information, see Quality of Service.

Optionally, NI-700 supports propagation of Memory System Resource Partitioning and Monitoring (MPAM) signals through the interconnect. For more information, see Memory System Resource Partitioning and Monitoring.

10.1 Resource Planes

Resource Planes (RPs) help reduce congestion and blocking between traffic streams that share resources.

Use RPs to help distribute and prioritize traffic flows across connections and end to end paths. Most network components support up to four resource planes, and provide time-multiplexed Virtual Channels (VCs) on a connection link. Each downstream interface, or network initiator, can be assigned to a specific RP. The RP is then applied to all routes originating from the initiator.

Use RPs to help prioritize certain traffic paths through shared resources. For example, two downstream interfaces share routes. One handles high-bandwidth traffic, for example, graphics data and one handles latency sensitive traffic, for example, CPU data. By default both are assigned the same RP, so they compete for bandwidth. However, assigning one of the interfaces to a different RP potentially prevents congestion between different traffic classes in the system. RPs provide the system designer with a solution to support non-blocking flows between different traffic classes and fix potential deadlock situations.

Routers perform arbitration between different traffic streams at the output port:

- The first stage involves arbitrating between traffic from different inputs within each RP or between RPs. The router uses a Quality of Service (QoS) priority based mechanism with a Least Recently Used (LRU) policy when QoS values are equal.
- The second stage involves arbitrating between different requesting RPs. In this stage the router uses LRU only.

You can configure up to four RPs.

10.2 Quality of Service

Throughout NI-700, arbitration nodes decide the order of progression for transactions according to priority. These nodes use the Quality of Service (QoS) value of a transaction as it passes through the interconnect to inform arbitration decisions.

QoS regulation features are also available 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 includes the following features:

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

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

As a side-effect of QoS, 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 every 16 transactions to pass regardless of the QoS priority.

NI-700 supports two types of QoS bandwidth regulation:

Hard bandwidth regulation

You can program NI-700 to block new traffic based on the number of outstanding transactions or based on an acceptable traffic profile that you define. For more information, see Hard bandwidth regulation.

Soft bandwidth regulation

You can program NI-700 to reduce the QoS value of new traffic when the bandwidth limit is exceeded rather than just blocking new transactions. For more information, see Soft bandwidth regulation.

10.2.1 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 downstream interface

10.2.1.1 Outstanding transaction regulation

The NI-700 ASNI tracks outstanding read and write requests that are submitted at its completer 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 you can configure 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 also stall 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 1. 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.

10.2.1.2 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 following table describes each parameter you can configure for the TSPEC regulators.

Note

Transfers in the following parameter descriptions correspond to data beats in read and write transactions.

Table 10-1: Acceptable network traffic profile limit

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

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.

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 for each cycle.

10.2.1.2.1 Calculating TSPEC parameters for traffic

NI-700 supports several Traffic Specification (TSPEC) parameters for tracking and regulating traffic. These parameters are r_value (average number of transfers for each cycle), b_value (burstiness allowance), and p_value (peak number of transfers for each cycle).

Calculating the average number of transfers for each cycle (r_value)

The following example shows how to program the TSPEC regulator to restrict an upstream device from issuing no more than 40MB/s traffic. If the upstream device has a data width of 64 bits and a clock frequency of 100MHz, the max bandwidth of the port is:

100 x 8 = 800MB/s

To restrict the bandwidth to 40MB/s, the limit on this interface must correspond to an average transfer rate of:

40 MBs / (100 x 8) = 0.05

In other words, 5% of the maximum bandwidth of the 64-bit interface at 100MHz. The value of the 0.05 fraction in binary is `0b000011001100110011001101`. However, since the average rate register field is six bits, that corresponds to setting the qosrdavg register to `0b000011` (six most significant bits). Effectively, the r_value parameter is programmed as:

(8 x 100 / 2^6) x qosrdavg register value = 37.5MB/s

So, a rounding error is introduced because of the six bits of granularity of the register.

Calculating the burstiness allowance (b_value)

The burstiness allowance is useful when the traffic from the upstream device follows a uniform pattern, but contains some repeating patterns, burstiness, or both. The b_value parameter allows you to set some level of burstiness variability over the average rate from that device. This parameter specifies an allowance of transfers that can be sent in addition to the average rate.

For example, if the incoming transfer rate matches the average rate, extra transfers can be sent until all these extra transfers are used. However, burstiness is not a one-time allowance. If the incoming transfer rate falls below the average rate, then all or part of the burstiness allowance can be accepted in addition to the average rate. The number of extra transfers that are accepted depends on the duration of the low bandwidth or idle window.

For the 64-bit 100MHz upstream device in the previous example, the r_value setting of 37.5MB/s allows the device to send one transfer approximately every 21 cycles:

8 x 100 / 37.5 = 21.3

When the burstiness allowance is set to 0, the regulator enforces the injection rate given in the preceding equation. But with a nonzero b_value parameter, the regulator allows extra transfers to be sent in the monitoring window, according to:

(r_value x total cycles) + b_value

The b_value register is 14 bits wide, so the burstiness allowance can be up to `0x3fff` more transfers than the average rate over the monitoring window.

Calculating the peak number of transfers for each cycle (p_value)

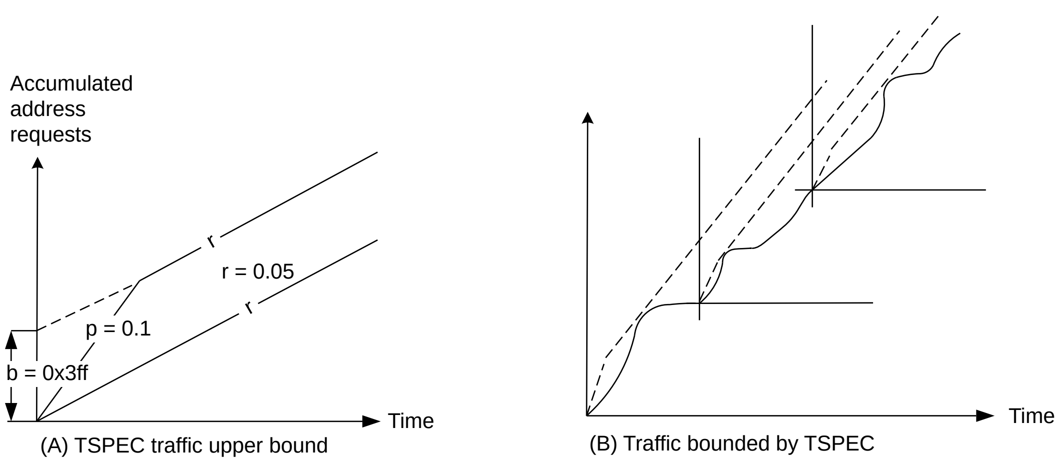
The peak rate setting defines an upper limit on bandwidth from the upstream device and is set as a fraction, similarly to the r_value parameter. Imposing an upper limit by using the p_value setting is useful when there is considerable burstiness in the traffic and a large burstiness allowance is set.

Again using the 64-bit 100MHz upstream device example, assume the b_value parameter is set to the maximum value of `0x3fff` and the p_value setting is 80MB/s. With a maximum bandwidth of 800MB/s, this peak rate setting corresponds to 10% of the maximum bandwidth, which is 0.1 times the channel width. Represented as a binary fraction, this value is `0'b000110`. With this p_value setting, the regulator allows transactions to be issued at a peak rate of 80MB/s until the burstiness allowance of 16383 transfers have been sent. After the burstiness allowance transfers have all been used, the regulator enforces a transfer rate of exactly 40MB/s until the burstiness allowance is available again.

10.2.1.2.2 TSPEC parameter examples

These examples show how the different TSPEC parameters work together.

Figure 10-1: TSPEC parameter examples



Example: Limit set to 50% of maximum interface bandwidth, no allowance for burstiness

Table 10-2: TSPEC parameter values for 50% of maximum interface bandwidth

Parameter value	Description	Value
r_value	Average number of transfers for each cycle	0.5
b_value	Burstiness allowance	0
p_value	Peak number of transfers for each 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: 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% bandwidth.

Table 10-3: 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 two-beat transfer every four cycles to achieve a 50% bandwidth.

Table 10-4: Incoming two-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: An average of 25% of the maximum interface bandwidth with some allowance for burstiness. There is also a peak bandwidth limit set to 50% of the maximum interface bandwidth

Table 10-5: Parameter values and descriptions

Parameter value	Description	Value
r_value	Average number of transfers for each cycle	0.25
b_value	Burstiness allowance	2
p_value	Peak number of transfers for each 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 permits a burstiness allowance of two 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

Therefore, 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

Therefore, 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 is 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 tables show:

- 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

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% = two beats in eight cycles).

Table 10-6: Cycle 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

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).

Table 10-7: Cycle 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

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 10-8: Cycle 17–24 is the 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 10-9: Cycle 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

10.2.1.2.3 TSPEC registers and parameters

NI-700 has programmable registers to configure the Traffic Specification (TSPEC) parameters for hard bandwidth regulation. The registers that you must use depend on the TSPEC mode you require.

NI-700 supports configuring the TSPEC parameters for read-only, write-only, and read and write combined mode. In other words, you can set r_value, b_value and p_value parameters 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 10-10: Registers used to program TSPEC parameters

Register	Channel	Description
ASNI qosrdpk register	Read	Read hard bandwidth regulator peak rate (p_value)
ASNI qosrdavg register	Read	Read hard bandwidth regulator average rate (r_value)

Register	Channel	Description
ASNI qosrdbur register	Read	Read hard bandwidth regulator burstiness allowance (b_value)
ASNI qoswrpk register	Write	Write hard bandwidth regulator peak rate (p_value)
ASNI qoswravg register	Write	Write hard bandwidth regulator average rate (r_value)
ASNI qoswrbur register	Write	Write hard bandwidth regulator burstiness allowance (b_value)
ASNI qoscompk register	Combined read and write	Combined TSPEC bandwidth regulator peak rate register (p_value)
ASNI qoscombur register	Combined read and write	Combined TSPEC bandwidth regulator burstiness allowance register (b_value)
ASNI qoscomavg register	Combined read and write	Combined TSPEC bandwidth regulator average rate register (r_value)
HSNI qoscompk register	Combined read and write	Combined hard bandwidth regulator peak rate (p_value)
HSNI qoscomavg register	Combined read and write	Combined hard bandwidth regulator average rate (r_value)
HSNI qoscombur register	Combined read and write	Combined hard bandwidth regulator burstiness allowance (b_value)

10.2.2 Soft bandwidth regulation

NI-700 ASNIs and HSNIs support Bandwidth QoS Value (BQV) QoS regulators. You can use BQV regulators 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 that the channel has transferred. The QoS value of incoming transactions is reduced in proportion to the amount of excess bandwidth that the channel consumes.

The following table shows the parameters that are used for soft bandwidth regulation. Use the BQV control registers to program the parameters for each regulator.

Table 10-11: BQV control parameters

Parameter	Description
`qv_max`	Maximum QoS value for the channel. Used by default.
`qv_min`	Minimum QoS value for the channel.
`overspend_per_qv`	Number of excess transfers allowed for each QoS value, specified as a power of two.
`bw_alloc`	Threshold value for the average number of transfers in each cycle before QoS value reduction starts.
`bw_burst`	Number of extra transfers allowed above the average transfer threshold before QoS value reduction starts.

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

The `bw_alloc` and `bw_burst` parameters are used to set the point at which bandwidth regulation starts. `bw_alloc` specifies the maximum value that is acceptable for the average number of transfers in a cycle. This value represents the maximum sustained traffic level that is permitted on a channel before the bandwidth is regulated.

The burstiness allowance, `bw_burst`, specifies extra transfers that can be used without triggering bandwidth regulation when the number of transfers in a cycle temporarily exceeds the `bw_alloc` setting. While the number of transfers in each cycle remains above the `bw_alloc` threshold, each extra transfer is counted down from the `bw_burst` value until none remain. Only at this point does bandwidth regulation start. If the average number of transfers in a cycle drops below the `bw_alloc` threshold before the extra `bw_burst` transfers are all used, the counter is reset. The `bw_burst` parameter enables you to ensure that small, occasional spikes in traffic do not trigger bandwidth regulation.

The `bw_alloc` and `bw_burst` parameters function similarly to the `r_value` and `b_value` parameters that are used in TSPEC hard bandwidth regulation. For more examples of how to configure and program these parameters, see Traffic specification regulation.

The channel limit specification is calculated as:

(`bw_alloc` x number of cycles) + `bw_burst`

Whenever the total number of data transfers exceeds this limit, the extra transfers are divided by the allowed overspend, `overspend_per_qv`. The result is subtracted from the maximum QoS value for the channel, `qv_max_i`, to determine the reduced maximum QoS value for bandwidth regulation.

For example, if there are eight extra transfers over the specification and the allowed overspend is three, then the QoS value on the transaction reduces by:

$[8 / (2^3)] = 1$

That is, the regulated QoS value on the transaction 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 occurs because the reduction in QoS value only depends on the current accumulated transfers through the average transfer rate.

As with TSPEC hard bandwidth regulation, you can configure ASNIs with separate BQV parameter values for the read and write channels. ASNIs also include registers that you can program to override incoming AxQOS values for the read and write channels. For more information, see QoS value override programmable registers.

Alternatively, or in addition, you can configure ASNIs with a single combined regulator to manage both the read and write channels according to the same specification. Because AHB only has a single channel, HSNIs must be configured with the combined BQV regulator.

You configure soft bandwidth regulation by programming the BQV regulator registers.

Table 10-12: BQV regulator registers

Register	Description
QOSRDBQV	Read channel BQV regulator target bandwidth register for ASNIs only
QOSWRBQV	Write channel BQV regulator target bandwidth register for ASNIs only
QOSCOMBQV	Combined BQV regulator target bandwidth register for ASNIs and HSNIs

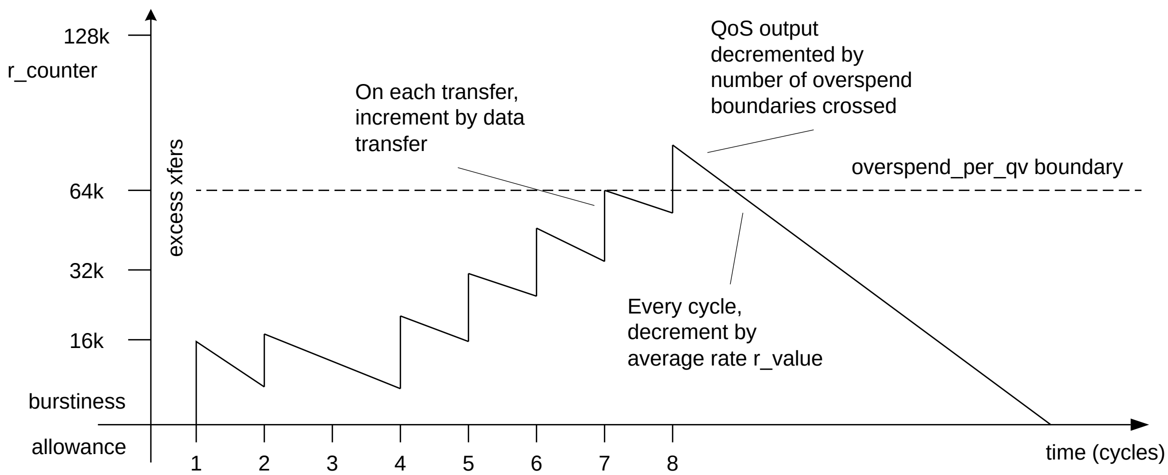
Each of the BQV registers contains the following fields, which are used to configure soft bandwidth regulation.

Table 10-13: BQV regulator register fields

Bit assignment	Field name	Description
[31:28]	BQV_OVRSPEND	Number of excess full data bus transfers permitted for the overspend allowance
[27:14]	BW_BURST	Number of extra full data bus transfers permitted for the burstiness allowance
[13:8]	BW_ALLOC	Threshold value for the average number of transfers in each cycle before BQV starts
[7:4]	QVMIN	Minimum QoS value for the channel
[3:0]	QVMAX	Maximum QoS value for the channel

The following figure shows how excess transfers above the average rate and the burstiness allowance determine when QoS value reduction starts.

Figure 10-2: QoS value reduction triggering



10.2.3 QoS value override programmable registers

NI-700 provides a programmable method to override the incoming AxQOS value independently for the read and write channels by using specific ASNI registers.

The ASNI QoS value override registers are ASNI arqos_value register and ASNI awqos_value register. To override the incoming AxQOS value, program these registers with the final QoS value that is applied to transactions when both of the following are true:

- The QOSOVERRIDE input signal bit is HIGH or the qos_override_enable bit of the ASNI qosctl register is HIGH
- The bqv_enable bits of the ASNI qosctl register are not set

The following table shows how the final QoS value is determined.

Table 10-14: Final QoS value matrix

QoS override register	QoS override input signal	BQV regulators enabled	Combined and individual regulator QoS values	Final QoS value
0	0	0	–	AxQOS (unchanged)
–	–	1	Combined regulator QoS value higher than individual regulator QoS value	Determined by individual regulator
–	–	1	Combined regulator QoS value lower than individual regulator QoS value	Determined by combined regulator
0	1	0	–	QoS value from override register
1	0	0	–	QoS value from override register
1	1	0	–	QoS value from override register

10.3 Memory System Resource Partitioning and Monitoring

NI-700 provides optional support for Memory System Resource Partitioning and Monitoring (MPAM) propagation. When MPAM support is enabled, you can override the MPAM values that are propagated through the network.

You can configure NI-700 to include MPAM on GT flits, and also configure individual ASNI and AMNI units to support MPAM. When you enable MPAM on an ASNI or an AMNI, the interface must include the MPAM signal on all address channels. For more information about the MPAM signals, see Signal descriptions.

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

- Software can 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 endpoint instance, then NI-700 ignores whether override is enabled in the MPAM override register. The endpoint drives the override value from the MPAM override register onto the GT flit.
- HSNIs always drive the override value onto the GT flit when forwarding a transaction that targets an MPAM-enabled downstream device.

For more information about the MPAM override registers, see Programmers model.

11. Performance monitoring

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

11.1 PMU organization

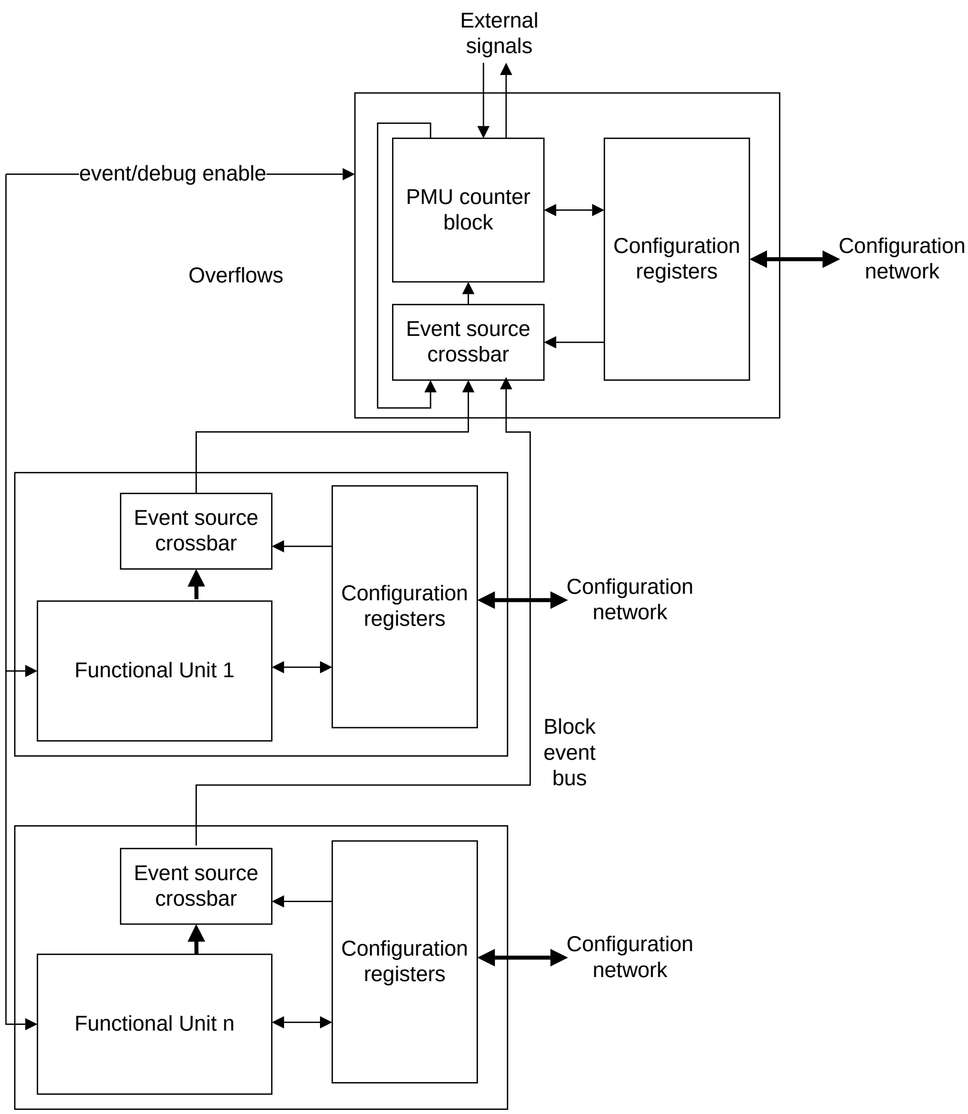
The PMU is distributed across each clock domain. Each clock domain contains software-visible event counters.

Within each clock domain, events are generated from several potential sources and multiplexed onto internal eight-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 the following counters, registers, and crossbar event selection:

- Software-visible event PMU counters
- Snapshot registers and other PMU control registers
- A configuration register for event selection in the event crossbar

Figure 11-1: PMU hierarchical organization block diagram



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 AMNI registers summary and ASNI registers summary.

Event and 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, up to a maximum of eight events, minimizing top-level wiring. By programming two PMU event select registers, you can select up to eight events for publishing on an internal eight-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. The counter block 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 eight 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, and 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. 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.

11.2 PMU system programming

You can program the PMU event counters, snapshot functionality, and interrupts.

For specific register descriptions, see the Programmers model.

11.2.1 Set up the PMU counters

To enable the Performance Monitoring Unit (PMU) to count specific events, you must set up the PMU event counters. You set up the PMU event counters by programming specific PMU-related registers that are located throughout the interconnect.

About this task

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 eight-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 and PMCNTENCLR registers to enable specific PMU event counters.
4. Write to the PMINTENSET and PMINTENCLR registers to enable interrupts for the corresponding specific PMU event counters.
5. Use the PMCNTENSET register to reset cycle and event counters, to write to the PMCR register, and to enable PMU counting.

This action enables all counters, whereas the PMCNTENSET and PMCNTENCLR enables specific counters.

11.2.2 Program PMU snapshot functionality

Program the PMU snapshot functionality to trigger a snapshot of PMU event counters.

About this task

To trigger a snapshot of PMU event counters, use the following methods:

- Set the control bits in the PMSSCR register
- Use a four-phase handshake on the signals, as the following table shows

Table 11-1: PMU snapshot signals

Signal	Direction	Description	Clock relationship
<CLKNAME>_PMUSNAPSHOTREQ	Input	A four-phase request to initiate a snapshot of PMU event counters	Asynchronous
<CLKNAME>_PMUSNAPSHOTACK	Output	Acknowledgment of the PMU snapshot capture	Asynchronous

Note

For PMU operation, NIDEN input must be asserted.

Procedure

1. Program the PMU event counters. See Set up the PMU counters.
2. Write 1 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 both lower and upper, and PMEVCNTSRn registers. Software can poll the PMSSR register to check that the snapshot has completed.

11.2.3 Program PMU interrupts

Program PMU interrupts to identify cycle or event counters that have overflowed.

About this task

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. For more information, see Set up the PMU counters.
   Any PMU counter overflow asserts <CLKNAME>_nPMUINTERRUPT. To determine the event counter or cycle counter that caused the interrupt, when observing assertion of <CLKNAME>_nPMUINTERRUPT, poll the PMOVSSR and PMOVSCLR registers.
3. Write 1 into the corresponding PMOVSCLR register to clear <CLKNAME>_nPMUINTERRUPT.

11.2.4 Performance monitoring and Secure Debug

If Non-secure event triggering is on, the Secure event enable signals, SPIDEN and SPNIDEN, enable the count 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 Performance monitoring.

The following table describes the PMU debug signals, their clock relationship, and signal direction.

Table 11-2: PMU debug signal descriptions, directions, and clock relationships

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 Non-secure events trigger 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) | SPIDEN) & (DBGEN | NIDEN).

11.3 ASNI performance events

The NI-700 ASNI can generate various performance events. Counting these events provides information about the performance of the ASNI as it operates.

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, for example 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 11-3: 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`	Read request: ReadNoSnoop (RNS)	N
`0x03`	Read request: ReadOnce (RO)	N
`0x04`	Cache Maintenance Requests: CleanShared, CleanInvalid, MakeInvalid, CleanSharedPersist **Note:** CleanSharedPersist is only present in ACE5-Lite.	N
`0x05`	Read data beat: any (RVALID & RREADY)	Y
`0x06`	Read data handshake with RLAST set	Y
`0x07`	Write request: any (AWVALID & AWREADY)	N
`0x08`	Write request: device	N
`0x09`	Write request: WriteNoSnoop (WNS)	N
`0x0A`	Write request: WriteLineUnique (WLU)	N
`0x0B`	Write request: WriteUnique (WU)	N
`0x0C`	Write request: Atomic (Store, Load, Swap, Compare)	N
`0x0D`	Write data beat: any (WVALID & WREADY)	Y
`0x0E`	Read request stall: ARVALID HIGH, ARREADY LOW	N
`0x0F`	Read data stall: RVALID HIGH, RREADY LOW	Y
`0x10`	Write request stall: AWVALID HIGH, AWREADY LOW	N
`0x11`	Write data stall: WVALID HIGH, WREADY LOW	Y
`0x12`	Write response stall: BVALID HIGH, BREADY LOW	Y
`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
`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 because WDATA FIFO is full	Y

Event code [5:0]	Event	Secure only
`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
`0x2B`	Read channel read request stall because of a lack of GT credit	Y
`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
`0x31`	Low wire mode arbitration stall on R channel	Y

For events marked as Secure only in the preceding table, 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.

11.4 AMNI performance events

The NI-700 AMNI can generate various performance events. Counting these events provides information about the performance of the AMNI as it operates.

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 11-4: AMNI performance events

Event code [5:0]	Event	Secure only
`0x00`	Read request: any (ARVALID & ARREADY)	N
`0x01`	Read request: device	N
`0x02`	Read request: ReadNoSnoop (RNS)	N
`0x03`	Read request: ReadOnce (RO)	N
`0x04`	Cache Maintenance Requests: CleanShared, CleanInvalid, MakeInvalid, CleanSharedPersist **Note:** CleanSharedPersist is only present in ACE5-Lite.	N

Event code [5:0]	Event	Secure only
`0x05`	Read data beat: any (RVALID & RREADY)	Y
`0x06`	Read data handshake with RLAST set	Y
`0x07`	Write request: any (AWVALID & AWREADY)	N
`0x08`	Write request: device	N
`0x09`	Write request: WriteNoSnoop (WNS)	N
`0x0A`	Write request: WriteLineUnique (WLU)	N
`0x0B`	Write request: WriteUnique (WU)	N
`0x0C`	Write request: Atomic (Store, Load, Swap, Compare)	N
`0x0D`	Write data beat: any (WVALID & WREADY)	Y
`0x0E`	Read request stall: ARVALID HIGH, ARREADY LOW	N
`0x0F`	Read data stall: RVALID HIGH, RREADY LOW	Y
`0x10`	Write request stall: AWVALID HIGH, AWREADY LOW	N
`0x11`	Write data stall: WVALID HIGH, WREADY LOW	Y
`0x12`	Write response stall: BVALID HIGH, BREADY LOW	Y
`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
`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
`0x23`	Read channel read response stall because of a lack of GT credit	Y
`0x24`	Low wire mode arbitration stall on B channel	Y
`0x25`	Low wire mode arbitration stall on R channel	Y

For events marked as Secure only in the preceding table, 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.

11.5 Data bandwidth at ASNI and AMNI

External AXI and ACE-Lite devices connect to the interconnect at ASNIs and AMNIs. You can monitor the data bandwidth through these blocks using specific PMU events.

11.5.1 Read and write bandwidth at ASNI and AMNI

NI-700 provides performance monitoring events to track the number of read and write data beats being transferred. Use these values 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 11-5: 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.

11.5.2 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 11-6: 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

11.5.3 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 11-7: 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

11.6 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, which is reproduced from the *Arm® AMBA® 5 AHB Protocol Specification AHB5, AHB-Lite*.

Table 11-8: 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

11.7 HSNI performance events

The NI-700 HSNI can generate various performance events. Counting these events provides information about the performance of the HSNI as it operates.

The following table shows the performance events that the HSNI can track.

Table 11-9: 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 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-For this event, thenE 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 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.
`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

Event code [4:0]	Event	Secure only
`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 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 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.
`0x2B`	Read request stall because of a lack of GT credit.	Y 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

11.8 HMNI performance events

The NI-700 HMNI can generate various performance events. Counting these events provides information about the performance of the HMNI as it operates.

The following table shows the performance events that the HMNI can track.

Table 11-10: 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 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

Event code [4:0]	Event	Secure only
`0x0D`	Write data beat: any.	Y 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.
`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 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 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

11.9 Data bandwidth at HSNI and HMNI

Data bandwidth performance can be monitored at HSNIs and HMNIs.

11.9.1 Read and write bandwidth at 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 11-11: 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.

11.9.2 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 11-12: Backpressure monitoring events

Event code [5:0]	Description
`0x0E`	Read request stall: HREADY LOW from the completer
`0x0F`	N/A
`0x10`	Write request stall: HREADY LOW
`0x11`	Write data stall: HREADY LOW
`0x12`	N/A

11.9.3 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 11-13: 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.

11.10 PMNI performance events

The NI-700 PMNI can generate various performance events. Counting these events provides information about the performance of the PMNI as it operates.

The following table shows the performance events that the PMNI can track.

Table 11-14: 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

Event code [4:0]	Event	Secure only
`0x25`	N/A	N

12. 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 Interconnect Device Management.

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).

12.1 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 IDM soft reset mode and Timeout detection through IDM block.

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 12-1: IDM error storage rules on simultaneous errors

Condition	Rule
Read and write transactions 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. This scenario 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 register
- Disable the timeout error detection using the IDM_TIMEOUT_CONTROL register during the period when soft reset is in active state

To exit from soft reset, the ERRSTATUS register IDM_ERRSTATUS must be cleared.

12.2 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 Programmers model.

12.3 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. Log the error information in the applicable IDM_ERRSTATUS, IDM error logging registers, IDM_ERRADR_LSB/IDM_ERRADR_MSB, and IDM_ERRMISC0/IDM_ERRMISC1 registers.
2. Set the V and UE fields of the associated IDM_ERRSTATUS register.
3. Set the UI field of the IDM_ERRCTLR register to mask signaling of the error to the RAS control block.
4. If there are multiple UEs, set the OF field of the IDM_ERRSTATUS register.

12.4 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 12-2: Non-IDM interrupt conditions per endpoint

Endpoint	Interrupt condition
AMNI	• Non-modifiable transaction is split into multiple individual burst transactions. • Unsupported ACE5-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.
HSNI	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 Programmers model.

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

Table 12-3: ACE5-LiteACP interoperability

Requester upstream of ASNI	Completer downstream of AMNI	Interoperability
ACE5-Lite	ACE5-LiteACP	Can connect directly if requester upstream of an 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 whether 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 bits, 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.

12.5 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 IDM error logging interrupts and status flags.
- For non-IDM interrupts, AMNI, HSNI, and HMNI have a separate set of interrupt registers. For more information, see Non-IDM interrupts.

An internal interrupt is asserted whenever any bits in the relevant register are set to 1. 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. Requester network interface, completer device. Highest priority. For multiple endpoints, the endpoint with the lowest internal Node ID takes precedence.
2. Completer network interface, requester device, if there is no conflicting requester network interface interrupt. Lowest priority. For multiple endpoints, the endpoint with the lowest internal Node ID takes precedence.

Note

The programmers view provides the Node ID.

12.6 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 Node ID.

2. The central interrupt handling block uses a simple arbitration mechanism to record the Node ID of the endpoint for which the external interrupt is raised.

   For a description of the arbitration mechanism, see Two-level interrupt generation.

   The interrupt handler reads the register and uses the Node ID value to read the corresponding interrupt and error logging registers within the endpoint.

   For more information, see Network Interface IDM registers summary.

3. The IDM_ERRSTATUS 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 network interface registers in Programmers model. For more information on the attributes of the request, see the registers on IDM_ERRADR_LSB, IDM_ERRADR_MSB, IDM_ERRMISC0, and IDM_ERRMISC1.

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 register. A subsequent error-causing transaction can then be stored and processed.

12.7 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.

12.8 Requester network interface error responses

NI-700 supports error responses from AMNI, HMNI, PMNI, and Configuration Network Interface (CFGNI) registers for unsupported transaction types and error responses for CMOs.

AMNI error responses

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 ASNI 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 to change it to an SLVERR response.
    - If AMNI has an ACE-Lite interface downstream but CMO_ON_READ and CMO_ON_WRITE properties are set to FALSE, the CMO properties indicate that there is no downstream cache. In this scenario, the transaction is terminated at the AMNI with an OK response. Alternatively, you can program a configuration control register AMNI_CONFIG_CTL, Select response to change the response to an SLVERR response.
    - If AMNI has an ACE-Lite interface downstream and the CMO_ON_WRITE property is set to TRUE, this configuration indicates a possible downstream cache. In this scenario, 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 ASNI device downstream, then the 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 12-4: AMNI configuration for cache stash transactions and relevant domains

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 to change it to an SLVERR response.
  - If AMNI has an ACE-Lite interface downstream but CMO_ON_READ and CMO_ON_WRITE properties are set to FALSE, the CMO properties indicate that there is no downstream cache. In this scenario, the transaction is terminated at the AMNI with an OK response by default. Alternatively, you can program a configuration control register AMNI_CONFIG_CTL, Select response 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, this configuration indicates a possible downstream cache. In this scenario, 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 WRITE_PLUS_CMO, CMO_ON_READ, and CMO_ON_WRITE properties are FALSE, it indicates that there is no downstream cache. In this scenario, the transaction is downgraded and only the write part of the transaction is issued downstream. The response indication for the downstream write, which comes from one of two sources, determines the response error indication. The two sources are the actual response for the write from downstream, and the response value indicated by the value of the configuration control register. The highest priority between these sources is used for the response indication. For more information on responses, see AMNI_CONFIG_CTL, Select response. For example, if the downstream response indicates SLVERR, and the configuration control register value indicates an OK response, the final response is an SLVERR. Alternatively, if the downstream response is an OK response, but the configuration control register value 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, this configuration indicates a possible cache downstream. In this scenario, the transaction is terminated at the AMNI with an SLVERR response.

HMNI error responses

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 12-5: AHB5 and AHB-Lite extended memory types configured as TRUE or FALSE

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

Request	AHB5 with Extended Memory Types set to TRUE	AHB-Lite or AHB5 with Extended Memory Types set to FALSE
Atomic transactions (AtomicSwap, AtomicStore, AtomicCompare, AtomicLoad)	SLVERR	SLVERR

PMNI error responses

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, and deallocating transactions. For example, ReadOnceCleanInvalid and ReadOnceMakeInvalid and atomics.

Internal Configuration Network Interface

All requests that map to the configuration 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 (ReadOnceCleanInvalid and ReadOnceMakeInvalid), and atomics.

13. Programmers model

This chapter describes the NI-700 programmers model.

13.1 About the programmers model

The 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 completer interfaces that are configured using Socrates.

The programmers model contains regions for control, upstream NIs, downstream 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.

13.2 Requirements of configuration register reads and writes

Reads and writes to the NI-700 configuration registers must meet certain requirements, otherwise the interconnect returns an error response. There are also security considerations to make when reading from and writing to the NI-700 configuration registers.

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 not be 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 four 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.

13.3 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, to identify each node during discovery.

Software uses the discovery mechanism to discover the pointer to the 4KB register programming region for each node. 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 Configuration Node (CFGNI)
- Voltage domain
- Power domain
- Clock domain
- Completer network interface nodes
- Requester network interface nodes
- PMU
- Interface ID

NI-700 assigns a unique Node ID to all the completer network interface nodes. Similarly, all the requester network interface nodes are assigned a unique Node ID. The Node ID space of the completer and requester network 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 13-1: 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

13.3.1 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.

13.3.2 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:

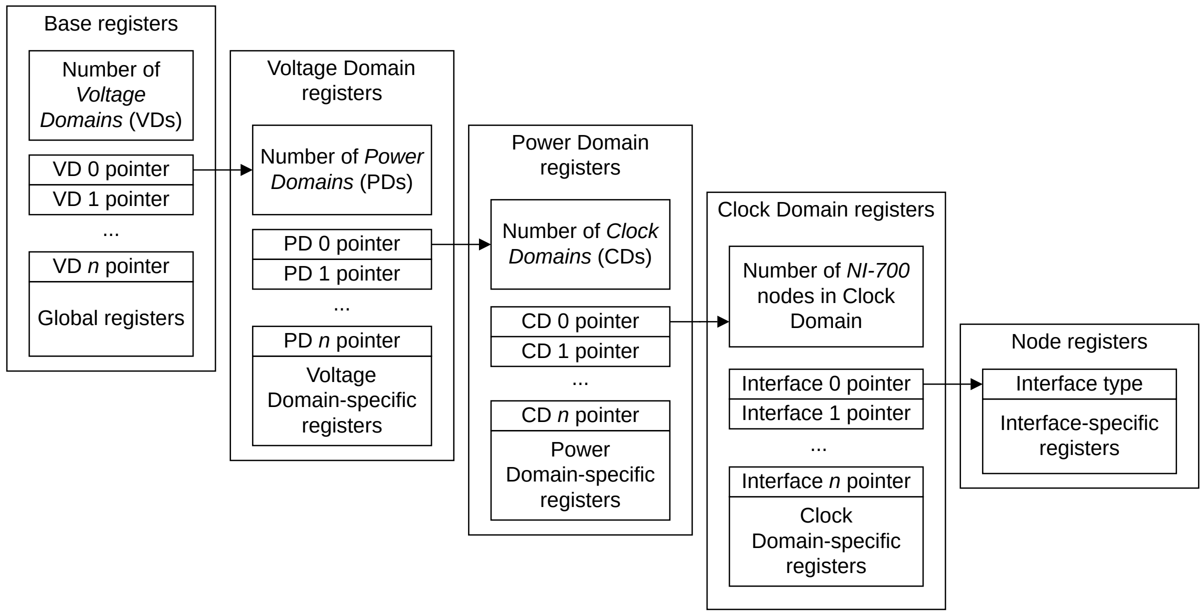
- Read information in the first 4KB region at PERIPHBASE. This information determines the:
    - Number of voltage domains in NI-700.
    - Offset from PERIPHBASE for each 4KB voltage domain address region.

- 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.
  - The topology information for those components.
  - The 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.
  - The topology information for those components.
  - The 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.
  - The topology information for those components.
  - The 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.
  - The 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 13-1: Access mechanism



The following are the major node types:

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 - 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 completer network interface nodes, requester network interface nodes, and the PMU node in that clock domain.
  - For each node, the information that is required to discover its node type, Node ID, and node information.

13.3.3 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 About the programmers model.

Table 13-2: 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

13.3.4 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 13-3: 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

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

13.3.5 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 13-4: 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:

- The information about one clock domain
- The discovery information for leaf nodes that are associated with that clock domain

13.3.6 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.

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

Table 13-5: 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.

13.4 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 CFGNI occupies 4KB of the address map. The final number of configuration nodes in your design depends on the number of:

- Voltage domains.
- Power domains.
- Clock domains.
- Endpoints. The number of endpoints is the sum of the number of ASNIs, AMNIs, HSNIs, HMNIs, and PMNIs in your design.
- 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.

13.5 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 PMUs
- Eight ASNIs
- Seven AMNIs
- Three HSNIs
- Three HMNIs
- Three PMNIs

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

Table 13-6: 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	PMU 0 registers
28KB	Clock domain 1 registers
32KB	ASNI 1 registers
36KB	AMNI 1 registers
40KB	PMU 1 registers
44KB	Power domain 1 registers
48KB	Clock domain 2 registers
52KB	ASNI 2 registers
56KB	AMNI 2 registers
60KB	HSNI 0 registers
64KB	HMNI 0 registers
68KB	PMNI 0 registers
72KB	PMU 2 registers
76KB	Clock domain 3 registers
80KB	ASNI 3 registers
84KB	AMNI 3 registers
88KB	PMU 3 registers
92KB	Voltage domain 1 registers
96KB	Power domain 2 registers
100KB	Clock domain 4 registers
104KB	ASNI 4 registers
108KB	AMNI 4 registers
112KB	HSNI 1 registers
116KB	HMNI 1 registers
120KB	PMNI 1 registers
124KB	PMU 4 registers
128KB	Clock domain 5 registers
132KB	ASNI 5 registers
136KB	AMNI 5 registers
140KB	PMU 5 registers
144KB	Power domain 3 registers
148KB	Clock domain 6 registers

Offset	Contents
152KB	ASNI 6 registers
156KB	AMNI 6 registers
160KB	PMU 6 registers
164KB	Clock domain 7 registers
168KB	ASNI 7 registers
172KB	HSNI 2 registers
176KB	HMNI 2 registers
180KB	PMNI 2 registers
184KB	PMU 7 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.

13.6 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.

13.6.1 Global registers summary

The register summary lists the global $prodname registers and some key characteristics.

The following table shows the global registers in offset order. The base address of $prodname 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 13-7: Global registers summary

Offset	Name	Type	Reset	Width	Description
0x000	NODE_TYPE	RO	Configuration dependent	32	NODE_TYPE, Global node type register
0x004	CHILD_NODE	RO	N Where N = the number of voltage domains	32	CHILD_NODE, Child node information register, voltage domains
0x0008 -0x00FF	VOLTAGE_DOMAIN_OFFSET_POINTERS	RO	P Where P = Pointers to the configuration region for each voltage domain	32	VOLTAGE_DOMAIN_POINTERS, Voltage domain offset pointers register
0x0F08	SECR_ACC	RW	0x00	32	SECR_ACC, Secure access register
0x0FD0	PERIPHERAL_ID4	RO	0x4 Partially device dependent	32	PERIPHERAL_ID4
0x0FD4	PERIPHERAL_ID5	RO	0x00	32	PERIPHERAL_ID5
0x0FD8	PERIPHERAL_ID6	RO	0x00	32	PERIPHERAL_ID6

Offset	Name	Type	Reset	Width	Description
`0x0FDC`	PERIPHERAL_ID7	RO	`0x00`	32	PERIPHERAL_ID7
`0x0FE0`	PERIPHERAL_ID0	RO	`0x3B`	32	PERIPHERAL_ID0
`0x0FE4`	PERIPHERAL_ID1	RO	`0xB4`	32	PERIPHERAL_ID1
`0x0FE8`	PERIPHERAL_ID2	RO	`0x0B`	32	PERIPHERAL_ID2
`0x0FEC`	PERIPHERAL_ID3	RO	`0x00`	32	PERIPHERAL_ID3
`0x0FF0`	COMPONENT_ID0	RO	`0x0D`	32	COMPONENT_ID0
`0x0FF4`	COMPONENT_ID1	RO	`0xF0`	32	COMPONENT_ID1
`0x0FF8`	COMPONENT_ID2	RO	`0x05`	32	COMPONENT_ID2
`0x0FFC`	COMPONENT_ID3	RO	`0xB1`	32	COMPONENT_ID3

13.6.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.6.2.1 NODE_TYPE, Global node type register

This register identifies the node type as NI-700 global or base registers.

Usage constraints

None.

Configurations

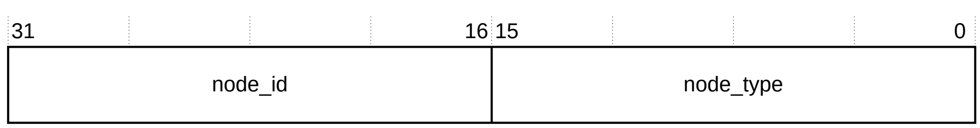
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-2: NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-8: 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.

13.6.2.2 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

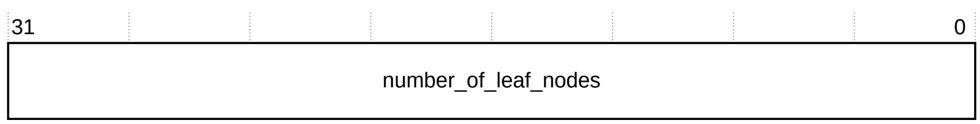
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-3: CHILD_NODE bit assignments



The following table shows the bit descriptions.

Table 13-9: 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.

13.6.2.3 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.

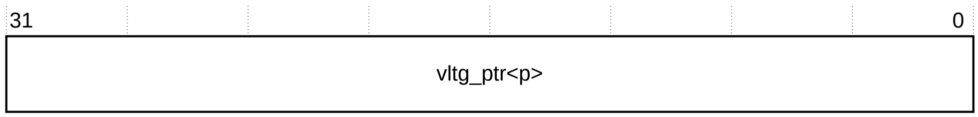
Available in all NI-700 configurations.

Attributes

For more information, see Voltage domain registers summary.

The following figure shows the bit assignments.

Figure 13-4: VOLTAGE_DOMAIN_POINTERS bit assignments



The following table shows the bit descriptions.

Table 13-10: 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.

13.6.2.4 SECR_ACC, Secure access register

This register controls whether only Secure transactions can read and program the NI-700 registers.

Usage constraints

Accessible using Secure transactions only.

Configurations

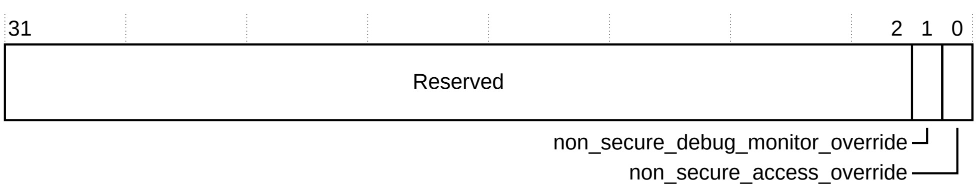
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-5: SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-11: 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.

Bits	Name	Description
[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.

13.6.2.5 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

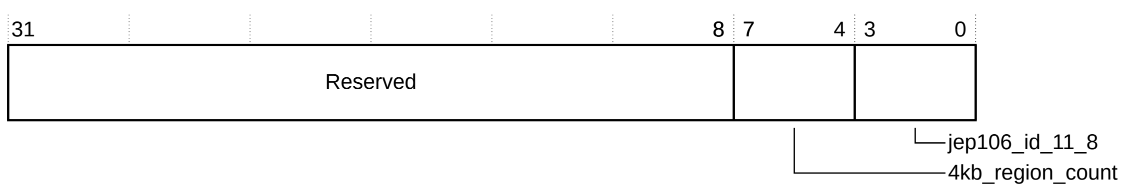
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-6: PERIPHERAL_ID4 bit assignments



The following table shows the bit descriptions.

Table 13-12: 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`.

13.6.2.6 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 Global registers summary.

The following figure shows the bit assignments.

Figure 13-7: PERIPHERAL_ID5 bit assignments

The following table shows the bit descriptions.

Table 13-13: PERIPHERAL_ID5 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	-	Reserved

13.6.2.7 PERIPHERAL_ID6

This register is reserved in the NI-700 design.

Usage constraints

None.

Configurations

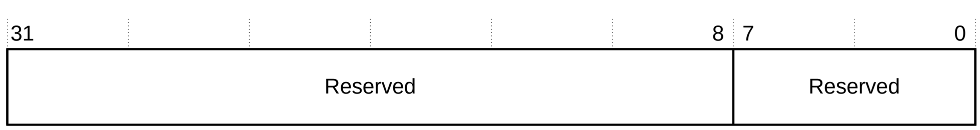
Reserved in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-8: PERIPHERAL_ID6 bit assignments



The following table shows the bit descriptions.

Table 13-14: PERIPHERAL_ID6 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	-	Reserved

13.6.2.8 PERIPHERAL_ID7

This register is reserved in the NI-700 design.

Usage constraints

None.

Configurations

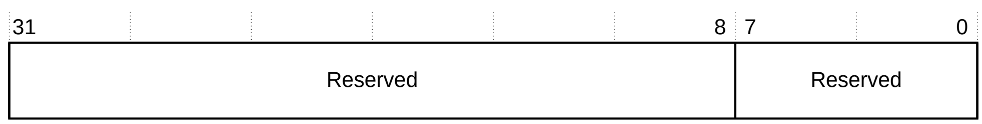
Reserved in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-9: PERIPHERAL_ID7 bit assignments



The following table shows the bit descriptions.

Table 13-15: PERIPHERAL_ID7 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	-	Reserved

13.6.2.9 PERIPHERAL_ID0

This register indicates the value for bits[7:0] of the NI-700 part number.

Usage constraints

None.

Configurations

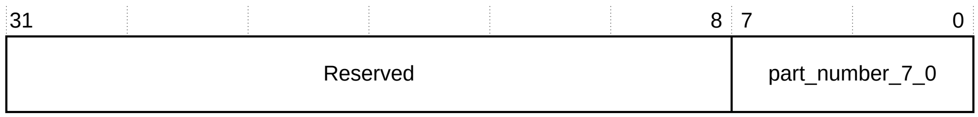
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-10: Peripheral ID0 bit assignments



The following table shows the bit descriptions.

Table 13-16: 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`

13.6.2.10 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

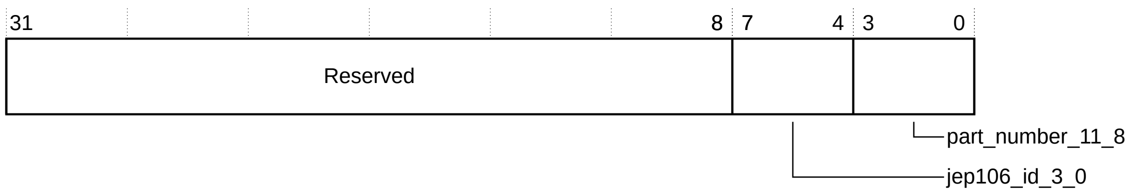
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-11: PERIPHERAL_ID1 bit assignments



The following table shows the bit descriptions.

Table 13-17: 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`.

13.6.2.11 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

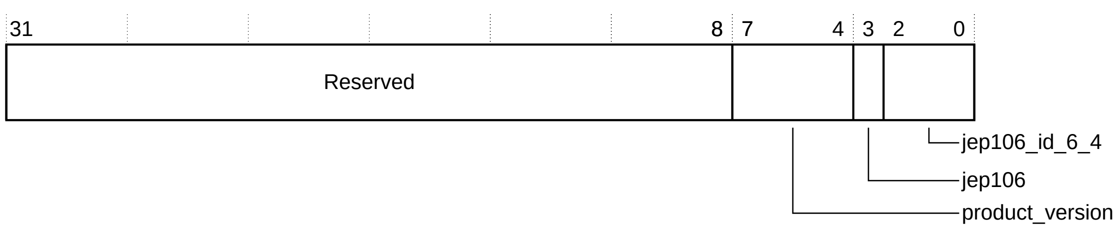
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-12: Peripheral ID2 bit assignments



The following table shows the bit descriptions.

Table 13-18: 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 release. This value also applies to the r2p1 REL release. • `0x3` indicates r2p3 REL release.
[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`.

13.6.2.12 PERIPHERAL_ID3

This register indicates the Arm-approved ECO number, and the NI-700 customer modification number.

Usage constraints

None.

Configurations

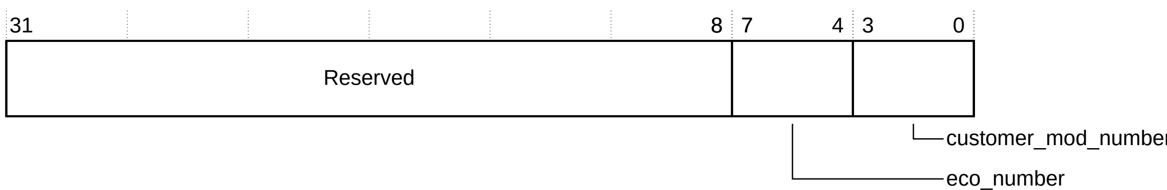
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-13: PERIPHERAL_ID3 bit assignments



The following table shows the bit descriptions.

Table 13-19: PERIPHERAL_ID3 bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:4]	eco_number	Arm approved ECO number. Use the ECOREVNUM input to modify this value. For more information, see Power, clock, reset, IDM, and other control signals
[3:0]	customer_mod_number	The NI-700 customer modification number. Do not modify this number unless you have permission from Arm.

13.6.2.13 COMPONENT_ID0

This register identifies NI-700 as an Arm component.

Usage constraints

None.

Configurations

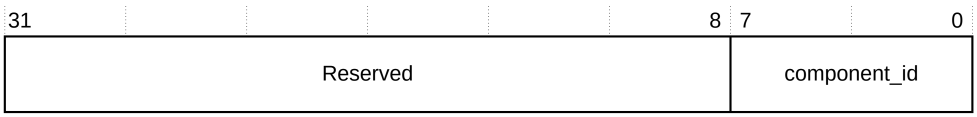
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-14: COMPONENT_ID0 bit assignments



The following table shows the bit descriptions.

Table 13-20: 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`.

13.6.2.14 COMPONENT_ID1

This register identifies NI-700 as an Arm component.

Usage constraints

None.

Configurations

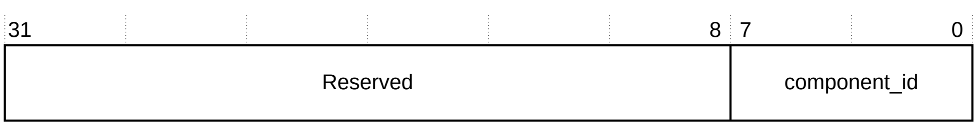
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-15: COMPONENT_ID1 bit assignments



The following table shows the bit descriptions.

Table 13-21: 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`, 0 (Arm PrimeCell).

13.6.2.15 COMPONENT_ID2

This register identifies NI-700 as an Arm component.

Usage constraints

None.

Configurations

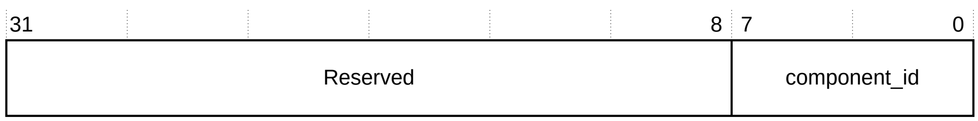
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-16: COMPONENT_ID2 bit assignments



The following table shows the bit descriptions.

Table 13-22: 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`.

13.6.2.16 COMPONENT_ID3

This register identifies NI-700 as an Arm component.

Usage constraints

None.

Configurations

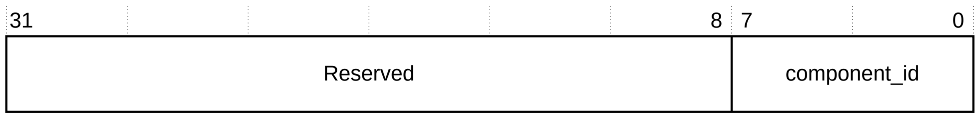
Available in all NI-700 configurations.

Attributes

For more information, see Global registers summary.

The following figure shows the bit assignments.

Figure 13-17: COMPONENT_ID3 bit assignments



The following table shows the bit descriptions.

Table 13-23: 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`.

13.7 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.

13.7.1 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 13-24: Voltage domain registers summary

Offset	Name	Type	Reset	Width	Description
`0x000`	NODE_TYPE	RO	Configuration dependent	32	NODE_TYPE, Voltage domain node type register
`0x004`	CHILD_NODE_INFORMATION	RO	0P	32	CHILD_NODE_INFORMATION, Child node information register, power domains
`0x0008 -0x08FF`	POWER_DOMAIN_POINTERS	RO	0P	32	POWER_DOMAIN_POINTERS, Power domain pointers register
`0x0F08`	SECR_ACC	RW	`0x00`	32	SECR_ACC, Secure access register

13.7.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.7.2.1 NODE_TYPE, Voltage domain node type register

This register identifies the node type as a voltage domain node.

Usage constraints

None.

Configurations

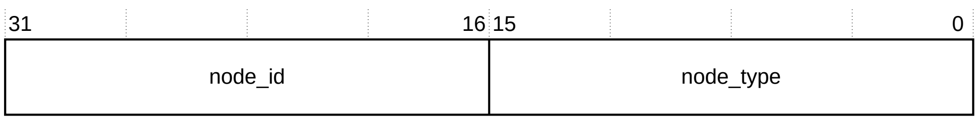
Available in all NI-700 configurations.

Attributes

For more information, see Voltage domain registers summary.

The following figure shows the bit assignments.

Figure 13-18: NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-25: 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.

13.7.2.2 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

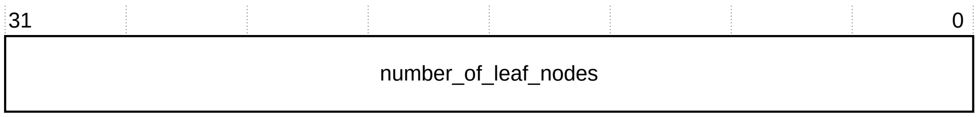
Available in all NI-700 configurations.

Attributes

For more information, see Voltage domain registers summary.

The following figure shows the bit assignments.

Figure 13-19: CHILD_NODE_INFORMATION bit assignments



The following table shows the bit descriptions.

Table 13-26: 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.

13.7.2.3 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

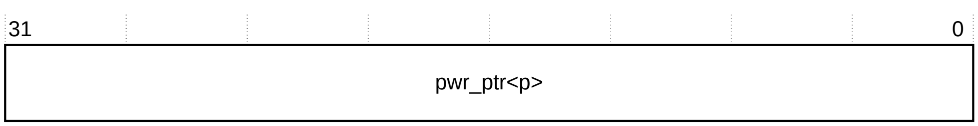
A copy of this register exists for each voltage domain. Available in all NI-700 configurations.

Attributes

For more information, see Voltage domain registers summary.

The following figure shows the bit assignments.

Figure 13-20: POWER_DOMAIN_POINTERS bit assignments



The following table shows the bit descriptions.

Table 13-27: 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.

13.7.2.4 SECR_ACC, Secure access register

This register controls whether only Secure transactions can read and program the NI-700 registers.

Usage constraints

Read and write to this register using Secure transactions only.

Configurations

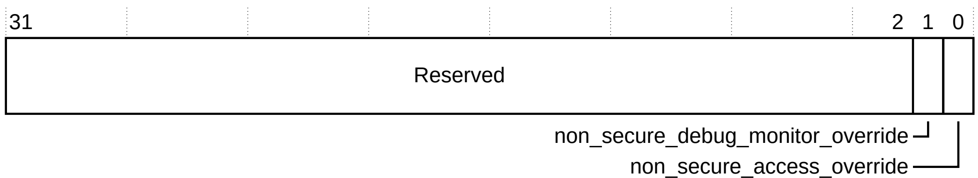
Available in all NI-700 configurations.

Attributes

For more information, see Voltage domain registers summary.

The following figure shows the bit assignments.

Figure 13-21: SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-28: 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.

13.8 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.

13.8.1 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 13-29: Power domain registers summary

Offset	Name	Type	Reset	Width	Description
0x000	NODE_TYPE	RO	Configuration dependent	32	NODE_TYPE, Power domain node type register
0x004	CHILD_NODE_INFORMATION	RO	N	32	CHILD_NODE_INFORMATION, Clock domains within power domain register
0x008 -0x08FF	CLOCK_DOMAIN_POINTERS	RO	P	32	CLOCK_DOMAIN_POINTERS, Clock domain pointers register
0x900	ENDPOINT_PD_IRQ_STATUS	RO	0x0	32	ENDPOINT_PD_IRQ_STATUS, Secure transaction error status register
0x904	ENDPOINT_PD_IRQ_CONTROL	RW	0x0	32	ENDPOINT_PD_IRQ_CONTROL
0x908	IDM_PD_ERROR_STATUS	RO	0x0	32	IDM_PD_ERROR_STATUS
0x90C	IDM_PD_ERROR_CONTROL	RW	0x0	32	IDM_PD_ERROR_CONTROL
0x910	IDM_PD_TIMEOUT_STATUS	RO	0x0	32	IDM_PD_TIMEOUT_STATUS
0x914	IDM_PD_TIMEOUT_CONTROL	RW	0x0	32	IDM_PD_TIMEOUT_CONTROL
0x918	IDM_PD_RESET_STATUS	RO	0x0	32	IDM_PD_RESET_STATUS
0x91C	IDM_PD_RESET_CONTROL	RW	0x0	32	IDM_PD_RESET_CONTROL
0x920	IDM_PD_ACCESS_STATUS	RO	0x0	32	IDM_PD_ACCESS_STATUS
0x924	IDM_PD_ACCESS_CONTROL	RW	0x0	32	IDM_PD_ACCESS_CONTROL
0x928	ENDPOINT_PD_IRQ_STATUS_NS	RO	0x0	32	ENDPOINT_PD_IRQ_STATUS_NS
0x92C	ENDPOINT_PD_IRQ_CONTROL_NS	RW	0x0	32	ENDPOINT_PD_IRQ_CONTROL_NS
0x930	IDM_PD_ERROR_STATUS_NS	RO	0x0	32	IDM_PD_ERROR_STATUS_NS
0x934	IDM_PD_ERROR_CONTROL_NS	RW	0x0	32	IDM_PD_ERROR_CONTROL_NS
0x938	IDM_PD_TIMEOUT_STATUS_NS	RO	0x0	32	IDM_PD_TIMEOUT_STATUS_NS
0x93C	IDM_PD_TIMEOUT_CONTROL_NS	RW	0x0	32	IDM_PD_TIMEOUT_CONTROL_NS
0x940	IDM_PD_RESET_STATUS_NS	RO	0x0	32	IDM_PD_RESET_STATUS_NS
0x944	IDM_PD_RESET_CONTROL_NS	RW	0x0	32	IDM_PD_RESET_CONTROL_NS
0x948	IDM_PD_ACCESS_STATUS_NS	RO	0x0	32	IDM_PD_ACCESS_STATUS_NS
0x94C	IDM_PD_ACCESS_CONTROL_NS	RW	0x0	32	IDM_PD_ACCESS_CONTROL_NS
0x0F08	SECR_ACC	RW	0x00	32	SECR_ACC, Secure access register

13.8.2 Power domain register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.8.2.1 NODE_TYPE, Power domain node type register

This register identifies the node type as a power domain node.

Usage constraints

None.

Configurations

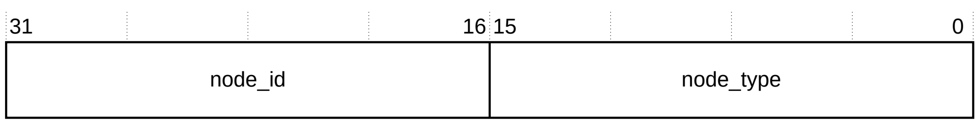
Available in all NI-700 configurations.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-22: NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-30: 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.

13.8.2.2 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

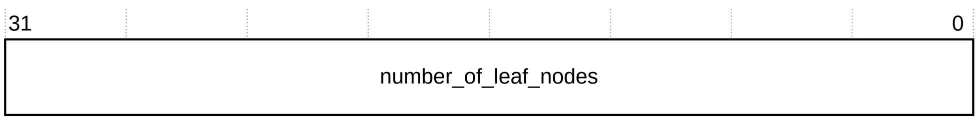
Available in all NI-700 configurations.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-23: CHILD_NODE_INFORMATION bit assignments



The following table shows the bit descriptions.

Table 13-31: CHILD_NODE_INFORMATION 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.

13.8.2.3 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.

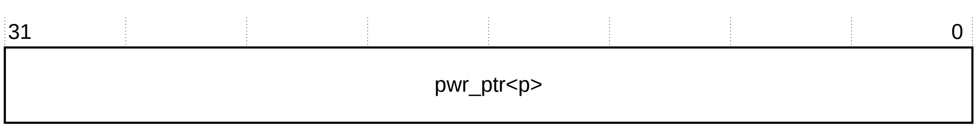
Available in all NI-700 configurations.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-24: CLOCK_DOMAIN_POINTERS bit assignments



The following table shows the bit descriptions.

Table 13-32: 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.

13.8.2.4 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

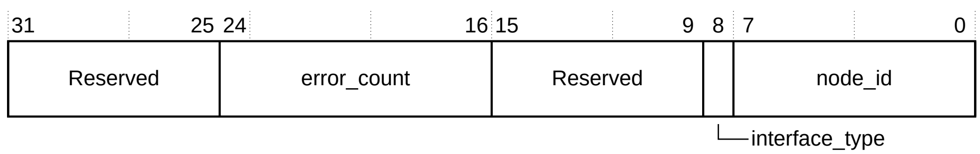
Available in all NI-700 configurations.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-25: ENDPOINT_PD_IRQ_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-33: ENDPOINT_PD_IRQ_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** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising an error interrupt.

13.8.2.5 ENDPOINT_PD_IRQ_CONTROL

This register, which is IDM-related, controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

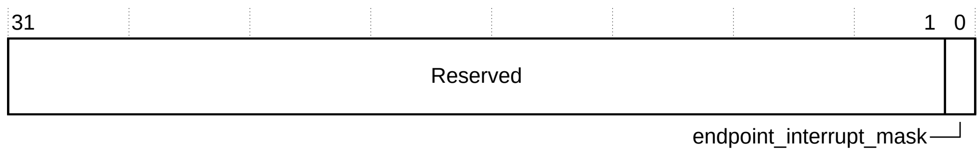
Available in all NI-700 configurations.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-26: ENDPOINT_PD_IRQ_CONTROL bit assignments



The following table shows the bit descriptions.

Table 13-34: 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.

13.8.2.6 IDM_PD_ERROR_STATUS

This register indicates the error status of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

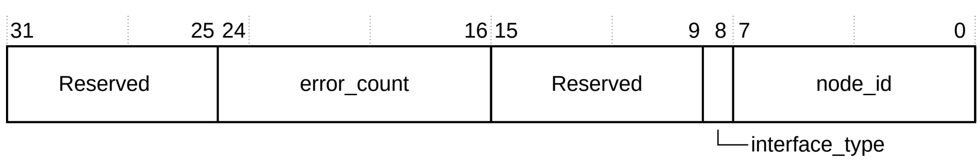
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-27: IDM_PD_ERROR_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-35: 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** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising an error interrupt.

13.8.2.7 IDM_PD_ERROR_CONTROL

This register controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

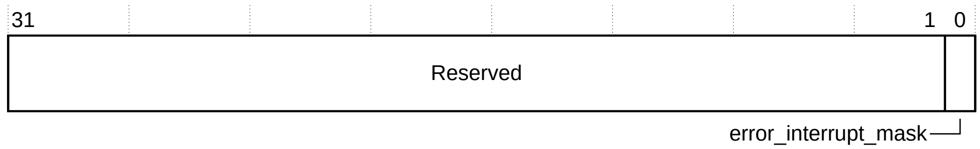
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-28: IDM_PD_ERROR_CONTROL bit assignments



The bit descriptions are shown in the following table.

Table 13-36: 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.

13.8.2.8 IDM_PD_TIMEOUT_STATUS

This register indicates the timeout status of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

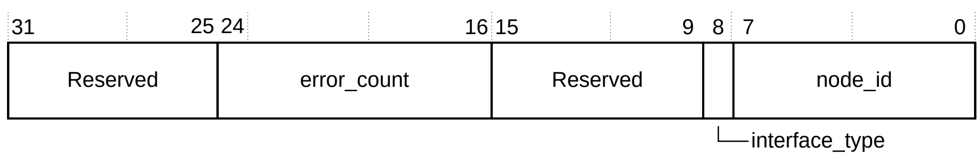
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-29: IDM_PD_TIMEOUT_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-37: 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** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising a timeout interrupt.

13.8.2.9 IDM_PD_TIMEOUT_CONTROL

This register controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

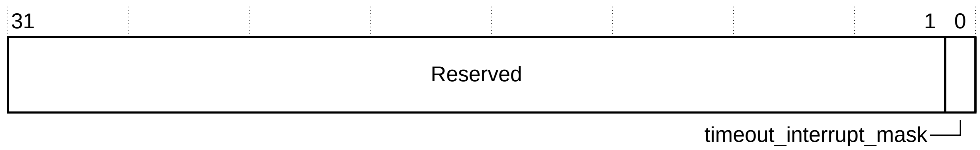
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

See Power domain registers summary for more information.

The following figure shows the bit assignments.

Figure 13-30: IDM_PD_TIMEOUT_CONTROL bit assignments



The following table shows the bit descriptions.

Table 13-38: 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.

13.8.2.10 IDM_PD_RESET_STATUS

This register indicates the reset access status of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

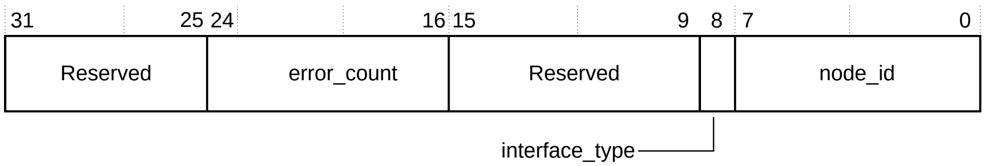
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-31: IDM_PD_RESET_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-39: IDM_PD_RESET_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 the Node ID specifies: **0** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising an activity while in reset interrupt.

13.8.2.11 IDM_PD_RESET_CONTROL

This register controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

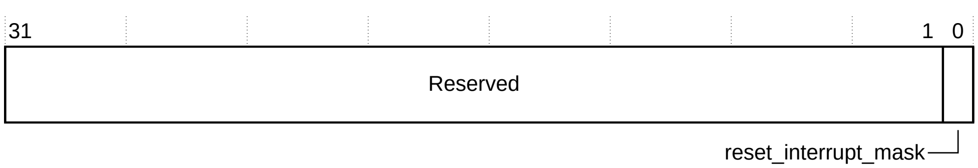
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-32: IDM_PD_RESET_CONTROL bit assignments



The following table shows the bit descriptions.

Table 13-40: 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.

13.8.2.12 IDM_PD_ACCESS_STATUS

This register indicates the isolation access status of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

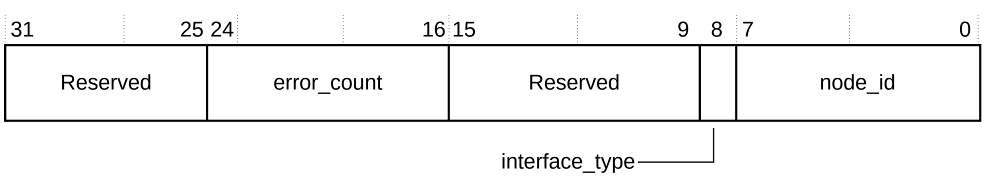
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-33: IDM_PD_ACCESS_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-41: IDM_PD_ACCESS_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** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising an activity while in access interrupt.

13.8.2.13 IDM_PD_ACCESS_CONTROL

This register controls the interrupts of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

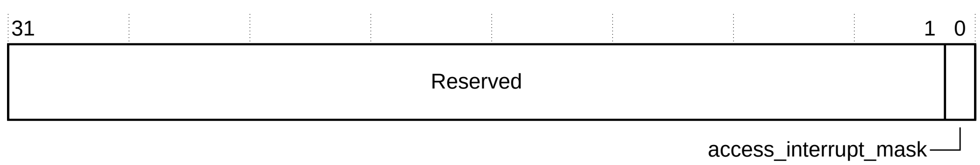
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-34: IDM_PD_ACCESS_CONTROL bit assignments



The following table shows the bit descriptions.

Table 13-42: 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.

13.8.2.14 ENDPOINT_PD_IRQ_STATUS_NS

This register, which is IDM related, indicates the error status of Non-secure transactions.

Usage constraints

None.

Configurations

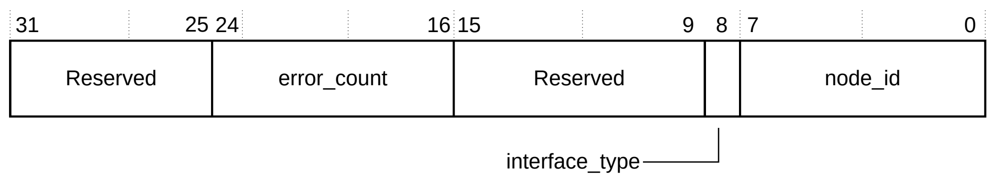
Available in all NI-700 configurations.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-35: ENDPOINT_PD_IRQ_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-43: 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** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising an error interrupt.

13.8.2.15 ENDPOINT_PD_IRQ_CONTROL_NS

This register, which is IDM related, controls the interrupts of Non-secure transactions.

Usage constraints

None.

Configurations

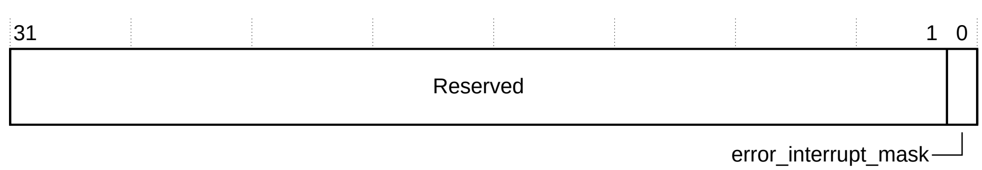
Available in all NI-700 configurations.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-36: ENDPOINT_PD_IRQ_CONTROL_NS bit assignments



The following table shows the bit descriptions.

Table 13-44: 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.

13.8.2.16 IDM_PD_ERROR_STATUS_NS

This register indicates the error status of Non-secure transactions.

Usage constraints

None.

Configurations

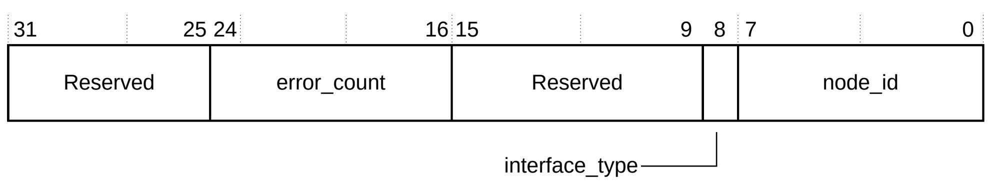
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-37: IDM_PD_ERROR_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-45: IDM_PD_ERROR_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** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising an error interrupt.

13.8.2.17 IDM_PD_ERROR_CONTROL_NS

This register controls the interrupts of Non-secure transactions.

Usage constraints

None.

Configurations

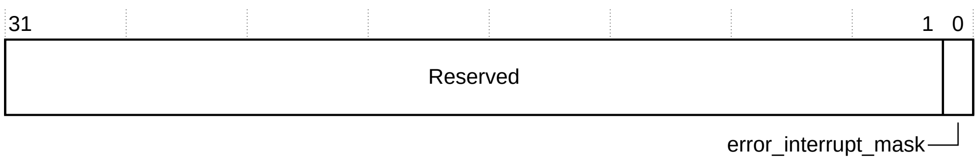
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-38: IDM_PD_ERROR_CONTROL_NS bit assignments



The following table shows the bit descriptions.

Table 13-46: 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.

13.8.2.18 IDM_PD_TIMEOUT_STATUS_NS

This register indicates the timeout status of Non-secure transactions.

Usage constraints

None.

Configurations

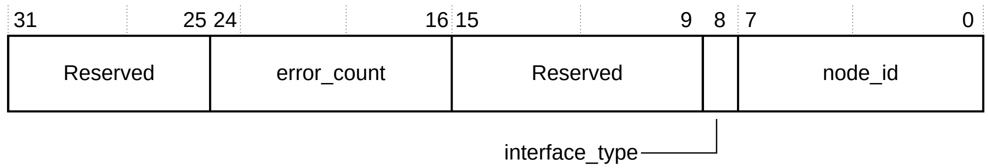
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-39: IDM_PD_TIMEOUT_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-47: IDM_PD_TIMEOUT_STATUS_NS 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	The type of interface that the Node ID specifies: **0** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising a timeout interrupt.

13.8.2.19 IDM_PD_TIMEOUT_CONTROL_NS

This register controls the interrupts of Non-secure transactions.

Usage constraints

None.

Configurations

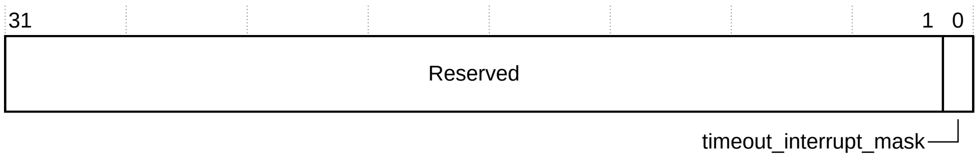
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-40: IDM_PD_TIMEOUT_CONTROL_NS bit assignments



The following table shows the bit descriptions.

Table 13-48: 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.

13.8.2.20 IDM_PD_RESET_STATUS_NS

This register indicates the reset access status of Non-secure transactions.

Usage constraints

None.

Configurations

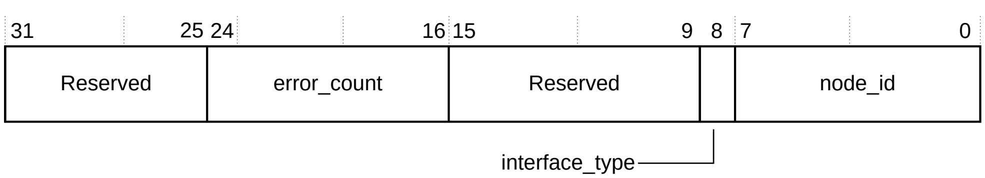
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-41: IDM_PD_RESET_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-49: 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** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising an activity while in reset interrupt.

13.8.2.21 IDM_PD_RESET_CONTROL_NS

This register controls the interrupts of Non-secure transactions.

Usage constraints

None.

Configurations

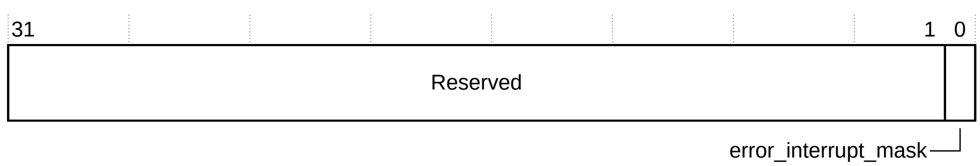
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-42: IDM_PD_RESET_CONTROL_NS bit assignments



The following table shows the bit descriptions.

Table 13-50: 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.

13.8.2.22 IDM_PD_ACCESS_STATUS_NS

This register indicates the isolation access status of Non-secure transactions.

Usage constraints

None.

Configurations

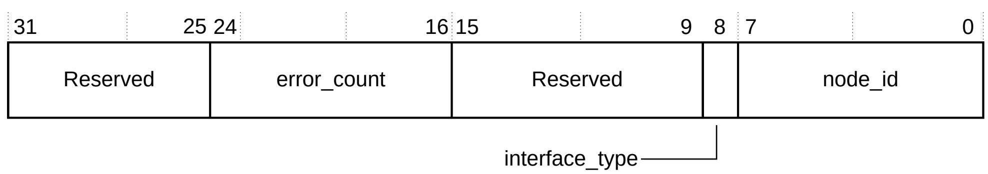
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-43: IDM_PD_ACCESS_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-51: 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** Completer **1** Requester
[7:0]	node_id	The Node ID of the first interface raising an activity while in isolation interrupt.

13.8.2.23 IDM_PD_ACCESS_CONTROL_NS

This register controls the interrupts of Non-secure transactions.

Usage constraints

None.

Configurations

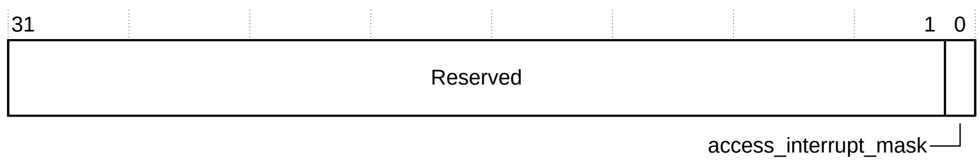
This register is implemented if there are endpoints enabled for IDM in that power domain. If there are no endpoints enabled for IDM in that power domain, then this register is not present in NI-700.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-44: IDM_PD_ACCESS_CONTROL_NS bit assignments



The following table shows the bit descriptions.

Table 13-52: 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.

13.8.2.24 SECR_ACC, Secure access register

This register controls whether only Secure transactions can read and program the NI-700 registers.

Usage constraints

Read and write to this register using Secure transactions only.

Configurations

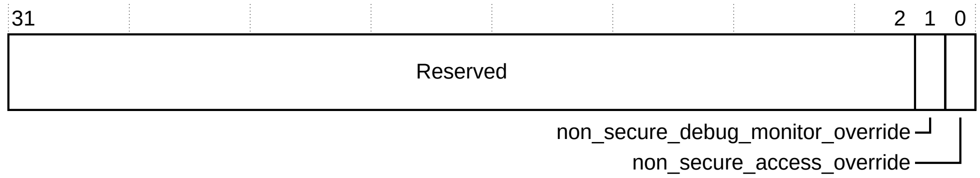
Available in all NI-700 configurations.

Attributes

For more information, see Power domain registers summary.

The following figure shows the bit assignments.

Figure 13-45: SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-53: 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.

13.9 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.

13.9.1 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 13-54: Clock domain registers summary

Offset	Name	Type	Reset	Width	Description
`0x000`	NODE_TYPE	RO	Configuration dependent	32	NODE_TYPE, Clock domain node type register
`0x004`	CHILD_NODE_INFORMATION	RO	N	32	CHILD_NODE_INFORMATION, Child node information register, network components
`0x0008 -0x08FF`	COMPONENT_NODE	RO	P	32	COMPONENT_NODE, Component node pointers register

Offset	Name	Type	Reset	Width	Description
0x0F08	SECR_ACC	RW	0x00	32	SECR_ACC, Secure access register

13.9.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.9.2.1 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

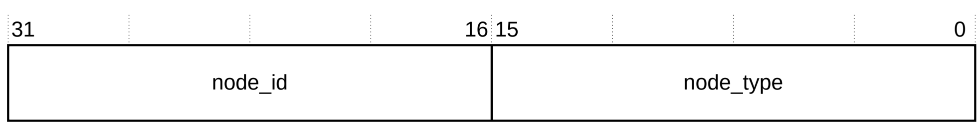
Available in all NI-700 configurations.

Attributes

For more information, see Clock domain registers summary.

The following figure shows the bit assignments.

Figure 13-46: NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-55: 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.

13.9.2.2 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

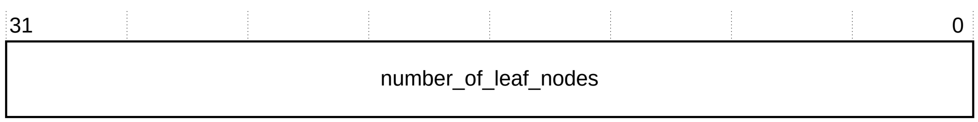
Available in all NI-700 configurations.

Attributes

For more information, see Clock domain registers summary.

The following figure shows the bit assignments.

Figure 13-47: CHILD_NODE_INFORMATION bit assignments



The following table shows the bit descriptions.

Table 13-56: 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.

13.9.2.3 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.

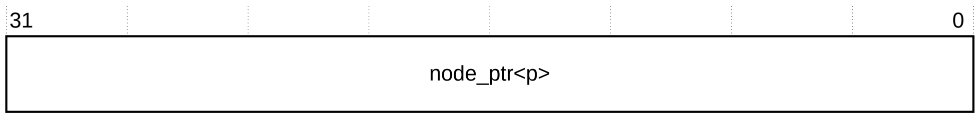
Available in all NI-700 configurations.

Attributes

For more information, see Clock domain registers summary.

The following figure shows the bit assignments.

Figure 13-48: COMPONENT_NODE bit assignments



The following table shows the bit descriptions.

Table 13-57: 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.

13.9.2.4 SECR_ACC, Secure access register

This register controls whether only Secure transactions can read and program the NI-700 registers.

Usage constraints

Read and write to this register using Secure transactions only.

Configurations

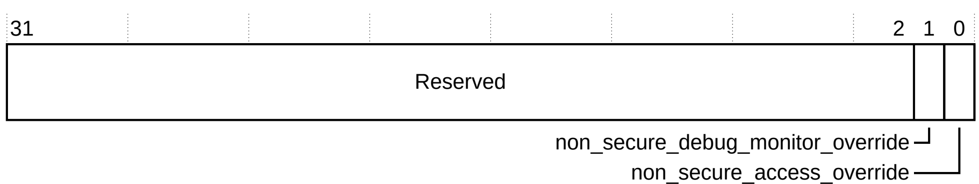
Available in all NI-700 configurations.

Attributes

For more information, see Clock domain registers summary.

The following figure shows the bit assignments.

Figure 13-49: SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-58: 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.

Bits	Name	Description
[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.

13.10 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.

13.10.1 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 13-59: PMU registers summary

Offset	Name	Type	Reset	Width	Description
0x000	NODE_TYPE	RO	[31:16] - Node ID for this PMU [15:0] - 0x0006	32	NODE_TYPE, Node type register for PMU registers
0x004	SECR_ACC	RW	0x00	32	SECR_ACC, Secure access register
0x008 0x010 0x018 0x020 0x028 0x030 0x038 0x040	PMEVCNTRn	RW	0x0	32	PMEVCNTRn, Performance monitor event counter registers
0x0F8	PMCCNTR_lower	RW	0x0	32	PMCCNTR_lower, Performance monitors cycle counter register
0x0FC	PMCCNTR_upper	RW	0x0	32	PMCCNTR_upper, Performance monitors cycle counter register
0x400 0x404 0x408 0x40C	PMEVTYPERn	RW	0x0	32	PMEVTYPERn, Performance monitor event type and filter registers

Offset	Name	Type	Reset	Width	Description
0x410					
0x414					
0x418					
0x41C					
0x610	PMSSR	RO	0x0	32	PMSSR, PMU snapshot status register
0x614	PMOVSSR	RO	0x00	32	PMOVSSR, PMU overflow status snapshot register
0x618	PMCCNTSR_lower	RO	0x0	32	PMCCNTSR_lower, Cycle counter snapshot register
0x61C	PMCCNTSR_upper	RO	0x0	32	PMCCNTSR_upper, Cycle counter snapshot register
0x620	PMEVCNTSRn	RO	0x0	32	PMEVCNTSRn, PMU event counter snapshot registers
0x624					
0x628					
0x62C					
0x630					
0x634					
0x638					
0x63C					
0x6F0	PMSSCR	WO	0x0	32	PMSSCR, Performance monitors snapshot capture register
0xC00	PMCNTENSET	RW	0x0	32	PMCNTENSET, Performance monitors count enable set register
0xC20	PMCNTENCLR	RW	0x0	32	PMCNTENCLR, Performance monitors count enable clear register
0xC40	PMINTENSET	RW	0x0	32	PMINTENSET, Performance monitors interrupt enable set register
0xC60	PMINTENCLR	RW	0x0	32	PMINTENCLR, Performance monitors interrupt enable clear register
0xC80	PMOVSCLR	RW	0x0	32	PMOVSCLR, Performance monitors overflow flag status clear register,
0xCC0	PMOVSSET	RW	0x0	32	PMOVSSET, Performance monitors overflow flag status set register
0xD80	PMCCCGR	RW	0x0	32	PMCCCGR, Performance monitors cycle counter clock gating
0xE00	PMCFGR	RO	0x00417F08	32	PMCFGR, Performance monitors configuration register
0xE04	PMCR	RW/ WO	0x0	32	PMCR, Performance monitors control register

13.10.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.10.2.1 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

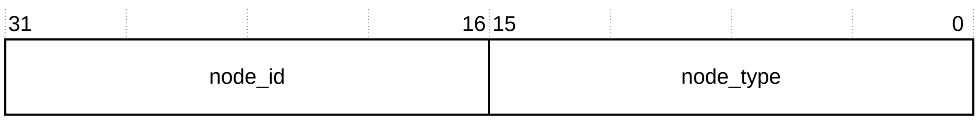
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-50: NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-60: 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.

13.10.2.2 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

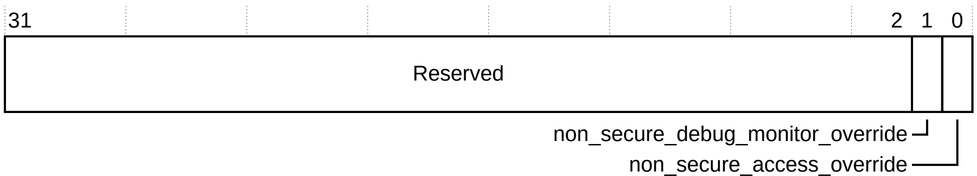
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-51: SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-61: 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.

13.10.2.3 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 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

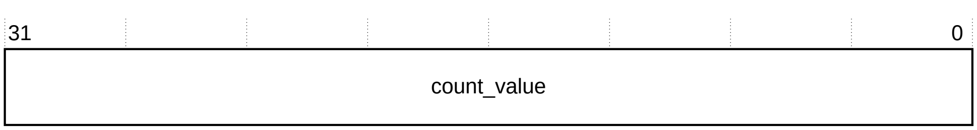
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-52: PMEVCNTRn bit assignments



The following table shows the bit descriptions.

Table 13-62: 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.

13.10.2.4 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 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

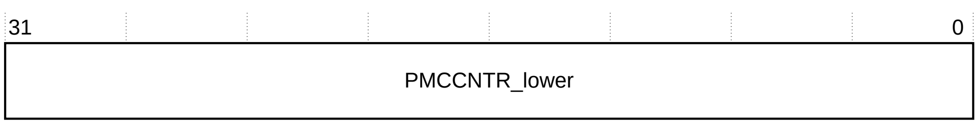
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-53: PMCCNTR_lower bit assignments



The following table shows the bit descriptions.

Table 13-63: PMCCNTR_lower bit descriptions

Bits	Name	Description
[31:0]	PMCCNTR_lower	The value of lower 64-bit cycle counter [31:0]

13.10.2.5 PMCCNTR_upper, Performance monitors cycle counter register

This register contains the 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 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

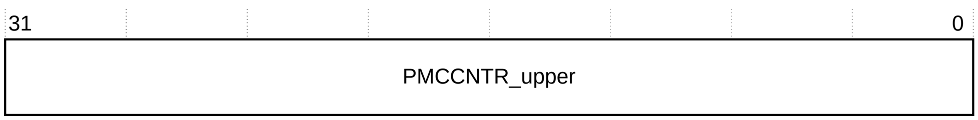
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-54: PMCCNTR_upper bit assignments



The following table shows the bit descriptions.

Table 13-64: PMCCNTR_upper bit descriptions

Bits	Name	Description
[31:0]	PMCCNTR_upper	The value of upper 64-bit cycle counter [63:32]

13.10.2.6 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 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

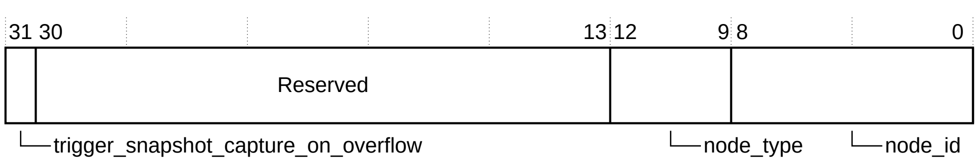
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-55: PMEVTYPERn bit assignments



The following table shows the bit descriptions.

Table 13-65: 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 Performance monitoring.

13.10.2.7 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 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.

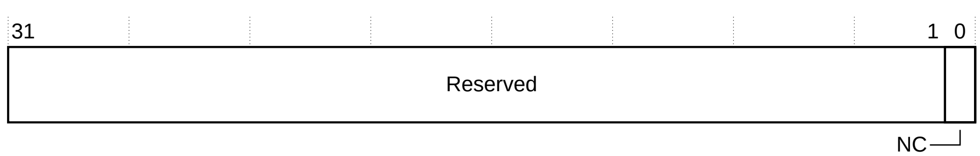
A copy of this register exists for each performance event counter.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-56: PMSSR bit assignments



The following table shows the bit descriptions.

Table 13-66: PMSSR bit descriptions

Bits	Name	Description
[31:1]	-	Reserved

Bits	Name	Description
[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.

13.10.2.8 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 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.

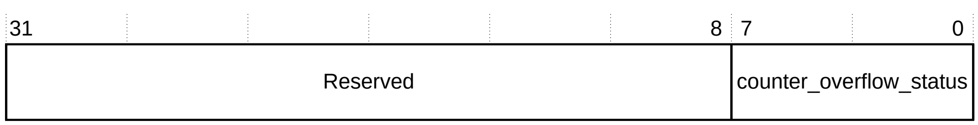
A copy of this register exists for each performance event counter.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-57: PMOVSSR bit assignments



The following table shows the bit descriptions.

Table 13-67: PMOVSSR bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	counter_overflow_status	Counter overflow status.

13.10.2.9 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 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

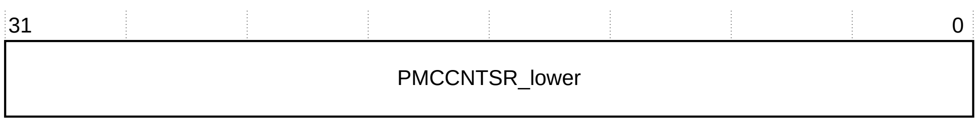
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-58: PMCCNTSR_lower bit assignments



The following table shows the bit descriptions.

Table 13-68: PMCCNTSR_lower bit descriptions

Bits	Name	Description
[31:0]	PMCCNTSR_lower	The snapshot value of the lower 64-bit cycle counter [31:0]

13.10.2.10 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 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

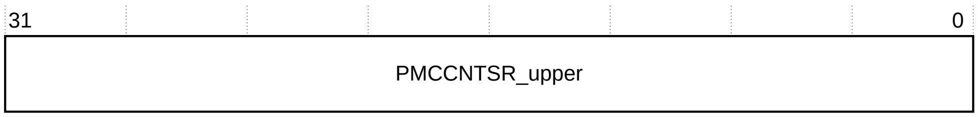
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-59: PMCCNTSR_upper bit assignments



The following table shows the bit descriptions.

Table 13-69: PMCCNTSR_upper bit descriptions

Bits	Name	Description
[31:0]	PMCCNTSR_upper	The snapshot value of the upper 64-bit cycle counter [63:32]

13.10.2.11 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 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

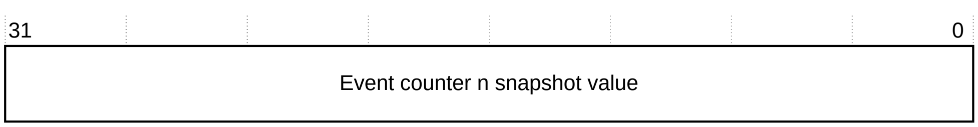
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-60: PMEVCNTSRn bit assignments



The following table shows the bit descriptions.

Table 13-70: PMEVCNTSRn bit descriptions

Bits	Name	Description
[31:0]	Event counter *n* snapshot value	The event counter *n* snapshot value.

13.10.2.12 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 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

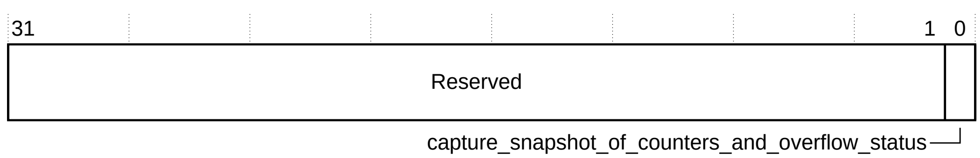
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-61: PMSSCR bit assignments



The following table shows the bit descriptions.

Table 13-71: 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.

13.10.2.13 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 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

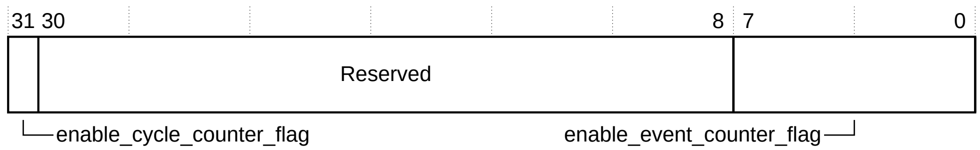
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-62: PMCNTENSET bit assignments



The following table shows the bit descriptions.

Table 13-72: 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>.

13.10.2.14 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 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

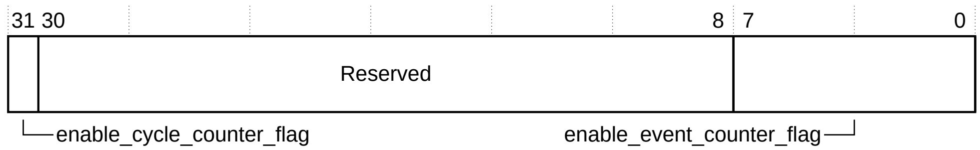
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-63: PMCNTENCLR bit assignments



The following table shows the bit descriptions.

Table 13-73: 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>.

13.10.2.15 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 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

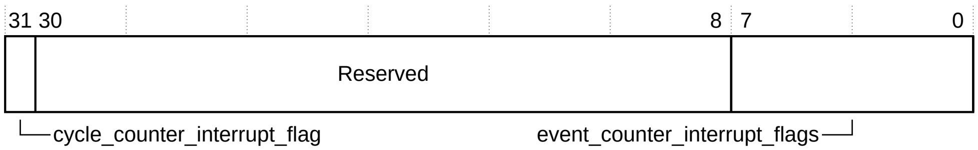
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-64: PMINTENSET bit assignments



The following table shows the bit descriptions.

Table 13-74: 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.

13.10.2.16 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 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

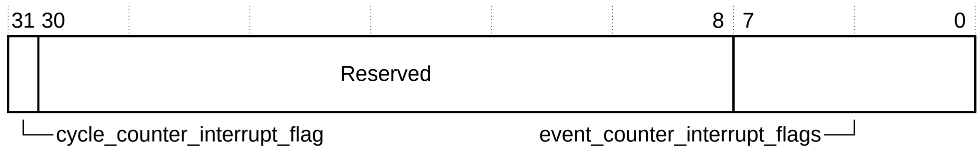
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-65: PMINTENCLR bit assignments



The following table shows the bit descriptions.

Table 13-75: 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.

13.10.2.17 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 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

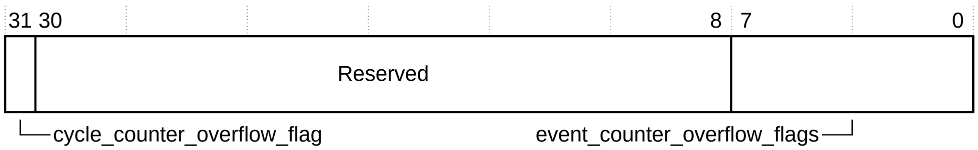
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-66: PMOVSCLR bit assignments



The following table shows the bit descriptions.

Table 13-76: 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.

13.10.2.18 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 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

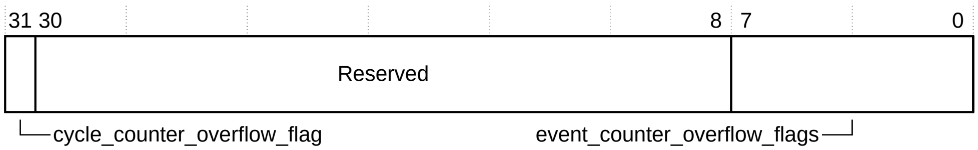
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-67: PMOVSSET bit assignments



The following table shows the bit descriptions.

Table 13-77: 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.

13.10.2.19 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 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

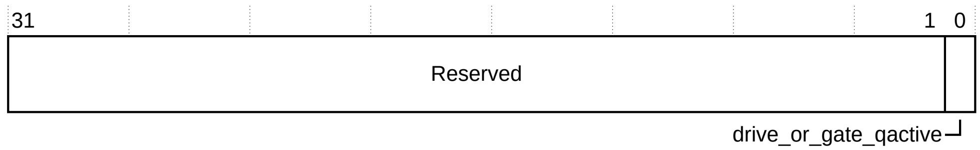
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-68: PMCCCGR bit assignments



The following table shows the bit descriptions.

Table 13-78: PMCCCGR bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	drive_or_gate_qactive	Defines whether to drive or gate the QACTIVEsignal. **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.

13.10.2.20 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 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

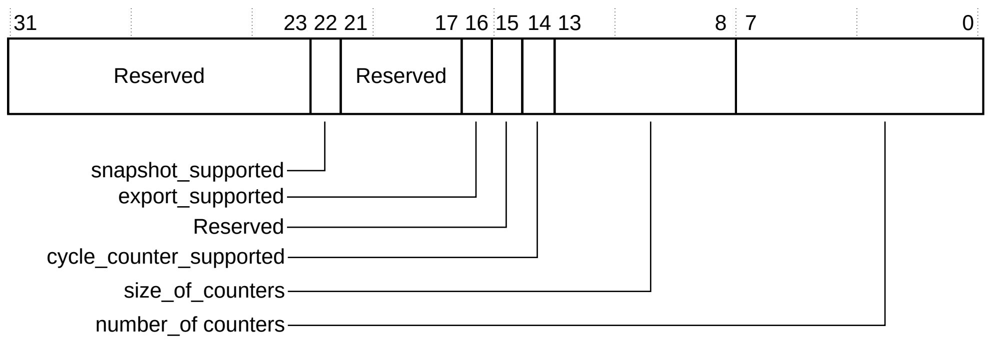
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-69: PMCFGR bit assignments



The following table shows the bit descriptions.

Table 13-79: 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
[13:8]	size_of_counters	Always `0b111111` (SIZE)
[7:0]	number_of_counters	Always `0b00001000` (8 counters)

13.10.2.21 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 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

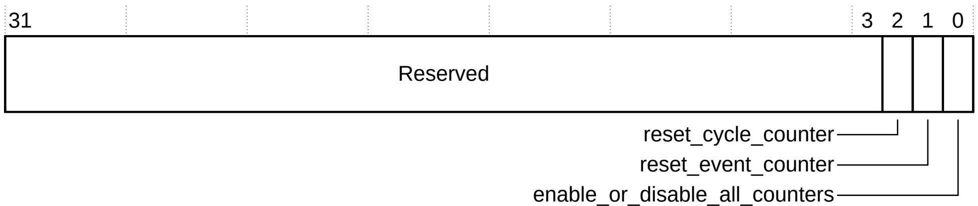
Available in all NI-700 configurations.

Attributes

For more information, see Performance Monitoring Unit registers summary.

The following figure shows the bit assignments.

Figure 13-70: PMCR bit assignments



The following table shows the bit descriptions.

Table 13-80: 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.

13.11 ASNI registers

This section describes the NI-700 ASNI registers. It contains a summary of the completer interface registers, in order of address offset, and a description of the bitfields for each register.

13.11.1 ASNI registers summary

The register summary lists the NI-700 ASNI registers and some key characteristics.

The following table shows the completer 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 13-81: 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
`0x004`	ASNI_NODE_INFO	RO		32	ASNI_NODE_INFO, Node information for ASNI register
`0x008`	ASNI_SECR_ACC	RW	`0x00`	32	ASNI_SECR_ACC, Secure access register
`0x00C`	ASNI_PMUSELA	RW	`0x0000`	32	ASNI_PMUSELA, Configure ASNI crossbar register
`0x010`	ASNI_PMUSELB	RW	`0x0000`	32	ASNI_PMUSELB, Configure ASNI crossbar register
`0x014`	ASNI_INTERFACEID_0-3	RO	Configuration dependent	32	ASNI_INTERFACEID, Configure ASNI interface IDs 0-3
`0x018`	ASNI_INTERFACEID_4-7	RAZ		32	ASNI_INTERFACEID, Configure ASNI interface IDs 4-7

Offset	Name	Type	Reset	Width	Description
0x01C	ASNI_INTERFACEID_8-11	RAZ		32	ASNI_INTERFACEID, Configure ASNI interface IDs 8-11
0x020	ASNI_INTERFACEID_12-15	RAZ		32	ASNI_INTERFACEID, Configure ASNI interface IDs 12-15
0x040	ASNI_NODE_FEAT	RAZ	0x0000	32	ASNI_NODE_FEAT, Node features register
0x044	ASNI_BURSPLT	RW/ RO	0x0007	32	ASNI_BURSPLT, Burst split control register
0x048	ASNI_ADDR_REMAP	RW	0x00	32	ASNI_ADDR_REMAP, Address remap vector register
0x080	ASNI_SILDBG	RW/ RO	0x00	32	ASNI_SILDBG, ASNI silicon debug monitor register
0x084	ASNI_QOSCTL	RW	0x00	32	ASNI_QOSCTL, QoS control register
0x088	ASNI_WDATTHRS	RW	0x00	32	ASNI_WDATTHRS, Write data FIFO threshold register
0x08C	ASNI_ARQOSOVR	RW	0x00	32	ASNI_ARQOSOVR, Read channel QoS value override register
0x090	ASNI_AWQOSOVR	RW	0x00	32	ASNI_AWQOSOVR, Write channel QoS value override register
0x094	ASNI_ATQOSOT	RW	0x00	32	ASNI_ATQOSOT, Maximum atomic Outstanding Transactions register
0x098	ASNI_ARQOSOT	RW	0x00	32	ASNI_ARQOSOT, Maximum read Outstanding Transactions register
0x09C	ASNI_AWQOSOT	RW	0x00	32	ASNI_AWQOSOT, Maximum write Outstanding Transactions register
0x0A0	ASNI_AXQOSOT	RW	0x00	32	ASNI_AXQOSOT, Maximum combined Outstanding Transactions register
0x0A4	ASNI_QOSRDPK	RW	0x00	32	ASNI_QOSRDPK, Read TSPEC bandwidth regulator peak rate register
0x0A8	ASNI_QOSRDBUR	RW	0x00	32	ASNI_QOSRDBUR, Read TSPEC bandwidth regulator burstiness allowance register
0x0AC	ASNI_QOSRDAVG	RW	0x00	32	ASNI_QOSRDAVG, Read TSPEC bandwidth regulator average rate register
0x0B0	ASNI_QOSWRPK	RW	0x00	32	ASNI_QOSWRPK, Write TSPEC bandwidth regulator peak rate register
0x0B4	ASNI_QOSWRBUR	RW	0x00	32	ASNI_QOSWRBUR, Write TSPEC bandwidth regulator burstiness allowance register
0x0B8	ASNI_QOSWRAVG	RW	0x00	32	ASNI_QOSWRAVG, Write TSPEC bandwidth regulator average rate register
0x0BC	ASNI_QOSCOMPK	RW	0x00	32	ASNI_QOSCOMPK, Combined TSPEC bandwidth regulator peak rate register
0x0C0	ASNI_QOSCOMBUR	RW	0x0000	32	ASNI_QOSCOMBUR, Combined TSPEC bandwidth regulator burstiness allowance register
0x0C4	ASNI_QOSCOMAVG	RW	0x00	32	ASNI_QOSCOMAVG, Combined TSPEC bandwidth regulator average rate register
0x0C8	ASNI_QOSRDBQV	RW	0x00	32	ASNI_QOSRDBQV, Read BQV bandwidth regulator target bandwidth register
0x0CC	ASNI_QOSWRBQV	RW	0x00	32	ASNI_QOSWRBQV, Write BQV bandwidth regulator target bandwidth register
0x0D0	ASNI_QOSCOMBQV	RW	0x00	32	ASNI_QOSCOMBQV, Combined BQV bandwidth regulator target bandwidth register

Offset	Name	Type	Reset	Width	Description
`0x0E0`	ASNI_AR_MPAM_OVERRIDE	RW	`0x0`	32	ASNI_AR_MPAM_OVERRIDE, Read channel MPAM override register
`0x0E4`	ASNI_AW_MPAM_OVERRIDE	RW	`0x0`	32	ASNI_AW_MPAM_OVERRIDE, Write channel MPAM override register

13.11.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.11.2.1 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

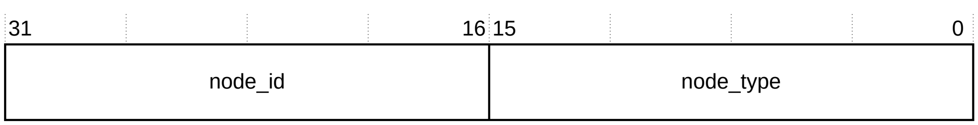
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-71: ASNI_NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-82: 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	Identifies the associated node type as a node for NI-700 ASNI registers. The reset value of this field is `0x4`.

13.11.2.2 ASNI_NODE_INFO, Node information for ASNI register

This register provides node information for ASNI, such as data width.

Usage constraints

None.

Configurations

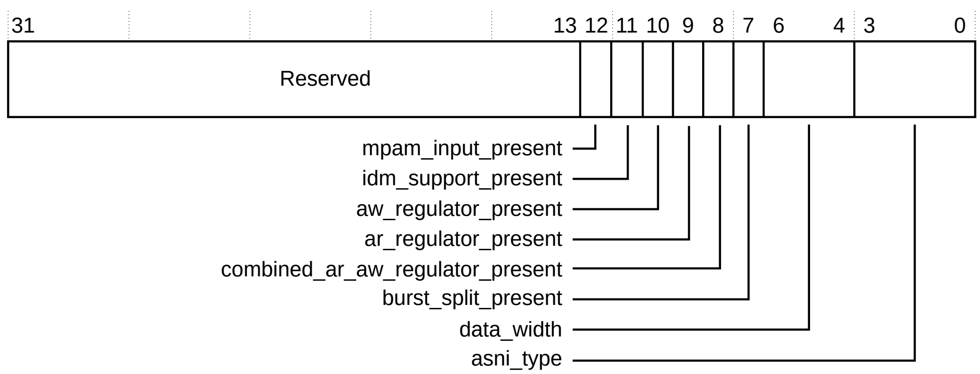
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-72: ASNI_NODE_INFO bit assignments



The following table shows the bit descriptions.

Table 13-83: 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.

Bits	Name	Description
[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.
[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 **`0b0110-0b1111`** Reserved **Note:** Reset value: **`0b0010`** AXI **`0b0011`** ACE-Lite

13.11.2.3 ASNI_SECR_ACC, Secure access register

This register controls Secure access.

Usage constraints

Accessible using only Secure accesses.

Configurations

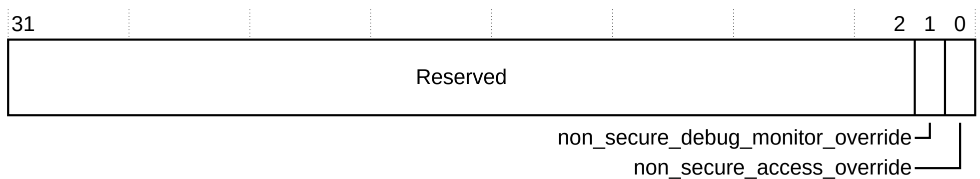
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-73: ASNI_SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-84: 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.

13.11.2.4 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 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

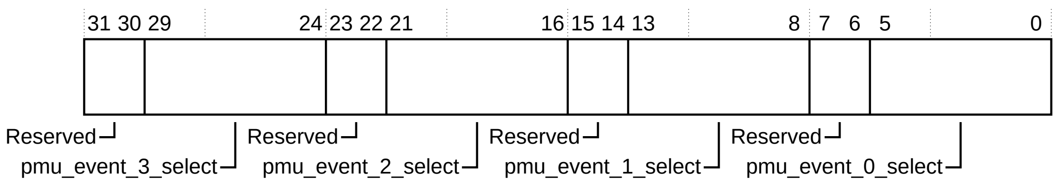
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-74: ASNI_PMUSELA bit assignments



The following table shows the bit descriptions.

Table 13-85: 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

13.11.2.5 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 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

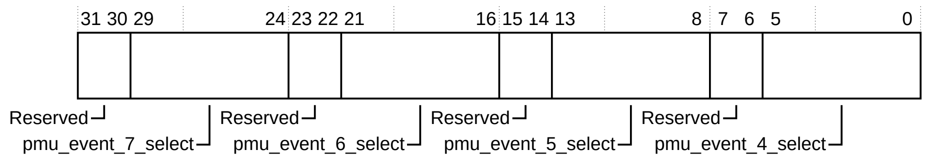
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-75: ASNI_PMUSELB bit assignments



The following table shows the bit descriptions.

Table 13-86: 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

13.11.2.6 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

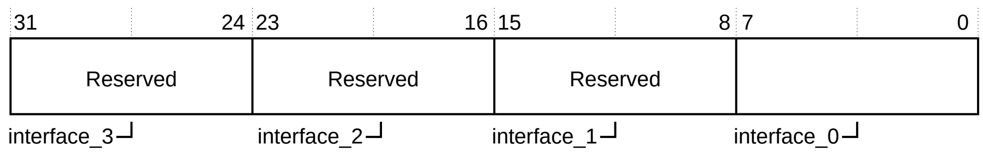
For more information, see ASNI registers summary.

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 13-76: ASNI_INTERFACEID bit assignments, ASNI interface IDs 0-3



The following table shows the bit descriptions.

Table 13-87: 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

13.11.2.7 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

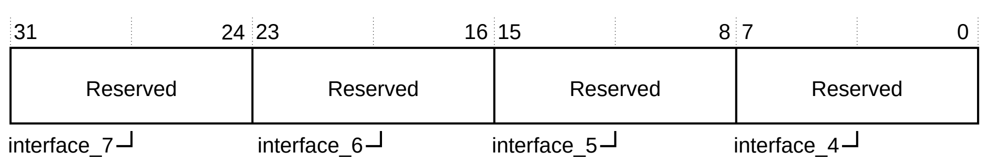
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-77: ASNI_INTERFACEID bit assignments, ASNI interface IDs 4-7



The following table shows the bit descriptions.

Table 13-88: 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

13.11.2.8 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

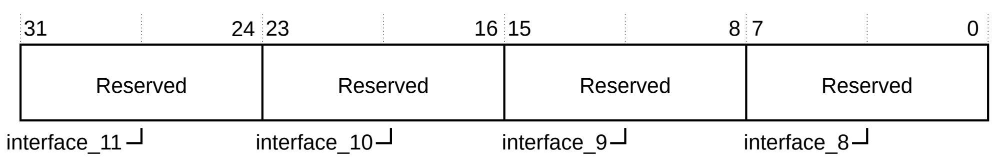
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-78: ASNI_INTERFACEID bit assignments, ASNI interface IDs 8-11



The following table shows the bit descriptions.

Table 13-89: 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

13.11.2.9 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

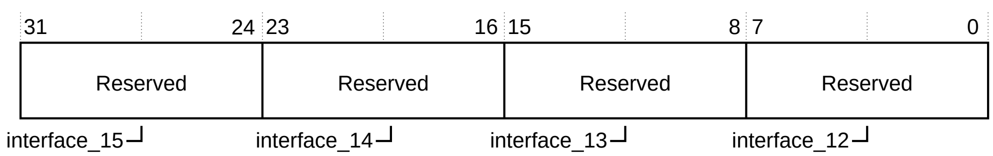
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-79: ASNI_INTERFACEID bit assignments, ASNI interface IDs 12-15



The following table shows the bit descriptions.

Table 13-90: 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

13.11.2.10 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 to permit Non-secure accesses.

Configurations

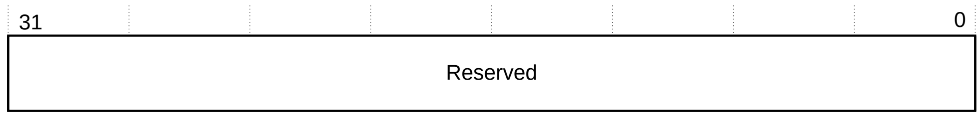
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-80: ASNI_NODE_FEAT bit assignments



The following table shows the bit descriptions.

Table 13-91: ASNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:0]	-	Reserved

13.11.2.11 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 to permit Non-secure accesses.

Configurations

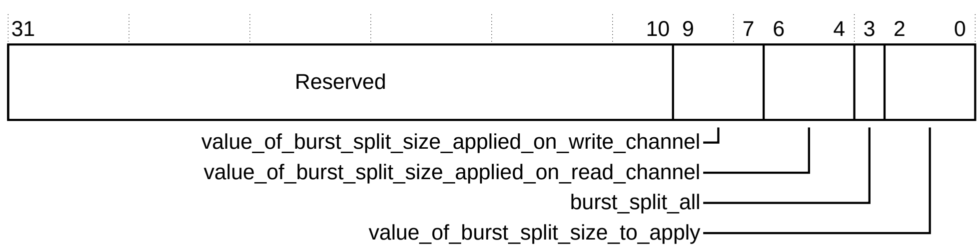
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-81: BURSPLT bit assignments



The following table shows the bit descriptions.

Table 13-92: 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]. This field is read only.
[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]. This field is read only.
[3]	burst_split_all	Burst split all. If set, modifiable Bursts to non-striped are also split. This field is read/write.
[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 This field is read/write.

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.

13.11.2.12 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 to permit Non-secure accesses.

Configurations

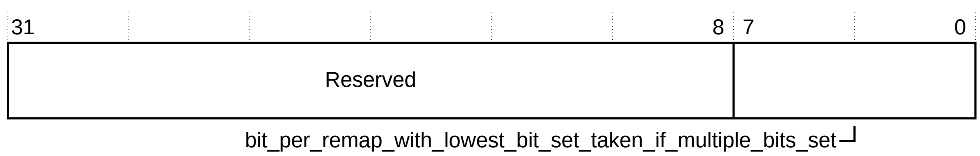
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-82: ASNI_ADDR_REMAP bit assignments



The following table shows the bit descriptions.

Table 13-93: 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.

13.11.2.13 ASNI_SILDBG, ASNI silicon debug monitor register

This register monitors the status of NI-700 completer interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register 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

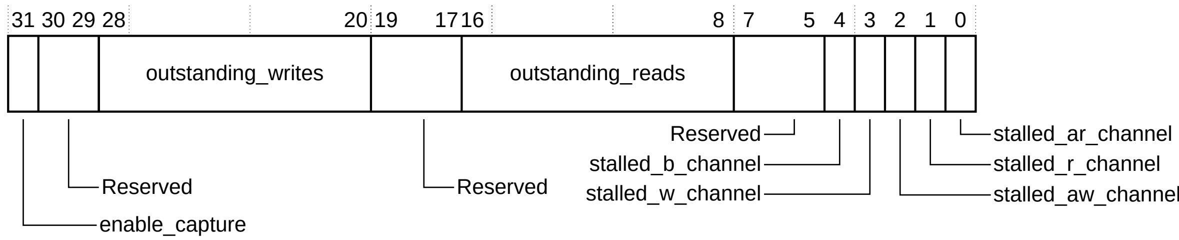
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-83: ASNI_SILDBG bit assignments



The following table shows the bit descriptions.

Table 13-94: 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
[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.

13.11.2.14 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 to permit Non-secure accesses.

Configurations

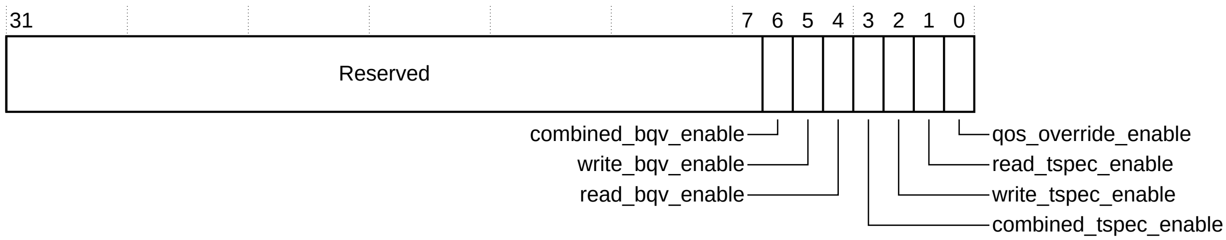
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-84: ASNI_QOSCTL bit assignments



The following table shows the bit descriptions.

Table 13-95: 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.

13.11.2.15 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 to permit Non-secure accesses.

Configurations

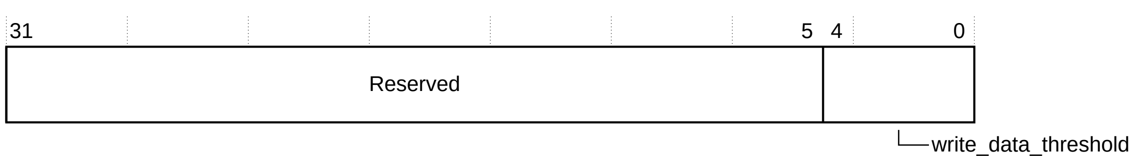
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-85: ASNI_WDATTHRS bit assignments



The following table shows the bit descriptions.

Table 13-96: 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.

13.11.2.16 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 completer interface. This value is applied to transactions when the following configuration scenario applies for the relevant completer 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 to permit Non-secure accesses.

Configurations

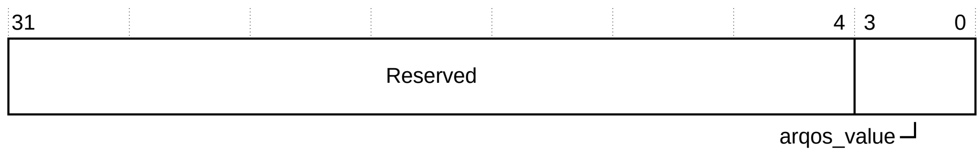
An instance of this register exists for each completer interface when a read channel QoS override value has been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-86: ASNI_ARQOSOVR register bit assignments



The following table shows the bit descriptions.

Table 13-97: ASNI_ARQOSOVR bit descriptions

Bits	Name	Description
[31:4]	-	Reserved

Bits	Name	Description
[3:0]	arqos_value	ARQOS value override for the completer 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.

13.11.2.17 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 completer interface. This value is applied to transactions when the following configuration scenario applies for the relevant completer 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

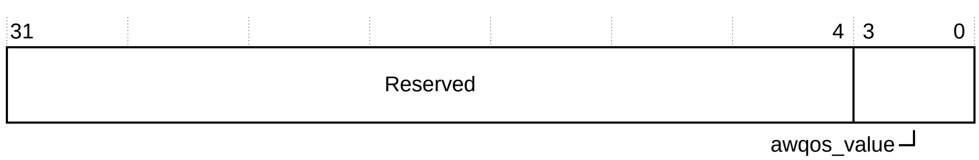
An instance of this register exists for each completer interface when a write channel QoS override value has been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-87: ASNI_AWQOSOVR bit assignments



The following table shows the bit descriptions.

Table 13-98: ASNI_AWQOSOVR bit descriptions

Bits	Name	Description
[31:4]	-	Reserved

Bits	Name	Description
[3:0]	awqos_value	The AWQOS value override for the completer 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.

13.11.2.18 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

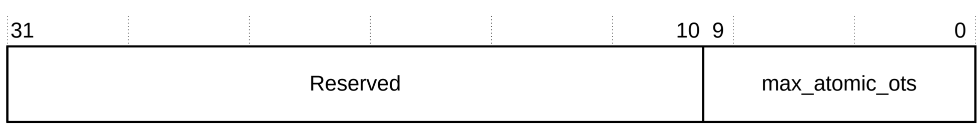
An instance of this register exists for each completer interface when a maximum number of atomic outstanding transactions has been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-88: ASNI_ATQOSOT bit assignments



The following table shows the bit descriptions.

Table 13-99: ASNI_ATQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved

Bits	Name	Description
[9:0]	max_atomic_ots	Specifies the maximum number of outstanding atomic transactions that the completer 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.

13.11.2.19 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 completer 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 to permit Non-secure accesses.

Configurations

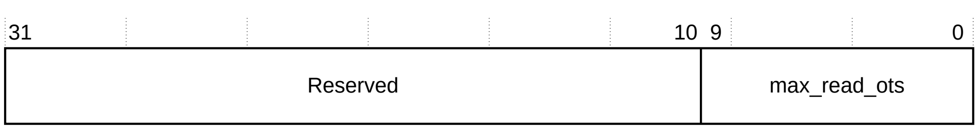
An instance of this register exists for each completer interface when a maximum read number of outstanding transactions has been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-89: ASNI_ARQOSOT bit assignments



The following table shows the bit descriptions.

Table 13-100: ASNI_ARQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved

Bits	Name	Description
[9:0]	max_read_ots	The maximum number of outstanding AR address requests when the OT regulator is enabled for the completer 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.

13.11.2.20 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 completer 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 to permit Non-secure accesses.

Configurations

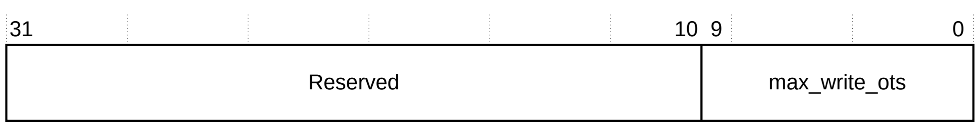
An instance of this register exists for each completer interface when a maximum number of outstanding write transactions has been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-90: ASNI_AWQOSOT bit assignments



The following table shows the bit descriptions.

Table 13-101: ASNI_AWQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9:0]	max_write_ots	The maximum number of write OTs for the completer 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.

13.11.2.21 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 completer 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 to permit Non-secure accesses.

Configurations

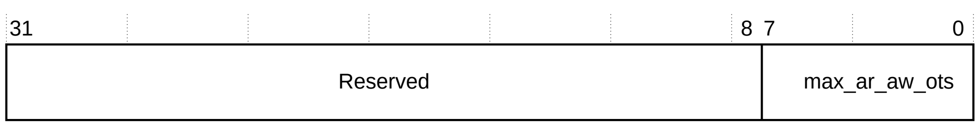
An instance of this register exists for each completer interface when a maximum combined number of outstanding transactions has been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-91: ASNI_AXQOSOT bit assignments



The following table shows the bit descriptions.

Table 13-102: ASNI_AXQOSOT bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9:0]	max_ar_aw_ots	The maximum number of OTs for the completer interface. This value is a combined issuing limit. It represents the maximum number of transactions that the upstream requester 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.

13.11.2.22 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

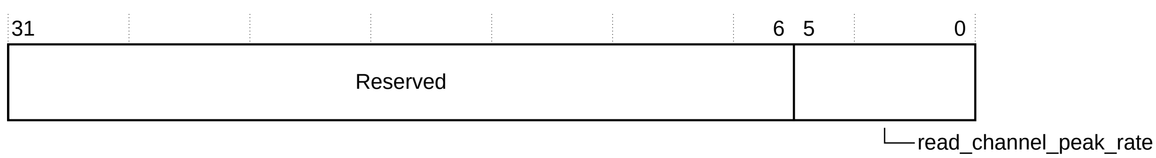
An instance of this register exists for each completer interface when read QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-92: ASNI_QOSRDPK bit assignments



The following table shows the bit descriptions.

Table 13-103: 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.

13.11.2.23 ASNI_QOSRDBUR, Read TSPEC bandwidth regulator burstiness allowance register

This register controls the QoS burstiness for the read hard bandwidth regulation, TSPEC, of a completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

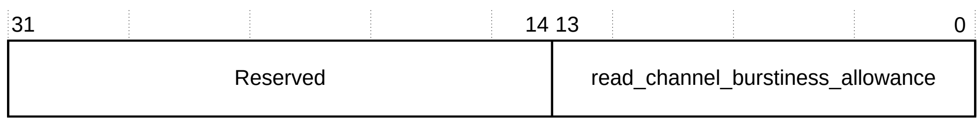
An instance of this register exists for each completer interface when read QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-93: ASNI_QOSRDBUR bit assignments



The following table shows the bit descriptions.

Table 13-104: 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.

13.11.2.24 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

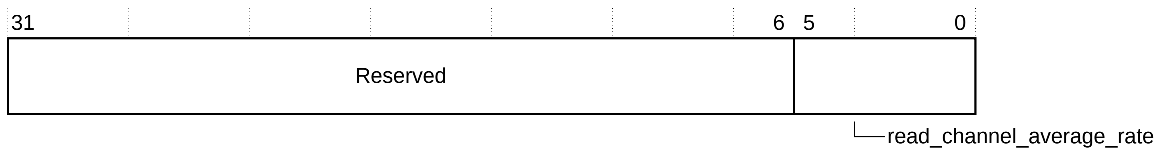
An instance of this register exists for each completer interface when read QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-94: ASNI_QOSRDAVG bit assignments



The following table shows the bit descriptions.

Table 13-105: 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.

13.11.2.25 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

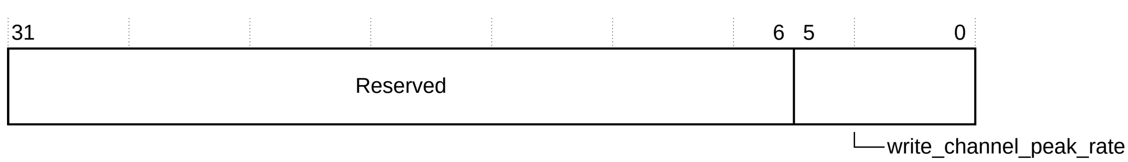
An instance of this register exists for each completer interface when write QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-95: ASNI_QOSWRPK bit assignments



The following table shows the bit descriptions.

Table 13-106: 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.

13.11.2.26 ASNI_QOSWRBUR, Write TSPEC bandwidth regulator burstiness allowance register

This register controls the QoS burstiness for the write hard bandwidth regulation, TSPEC, of a completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

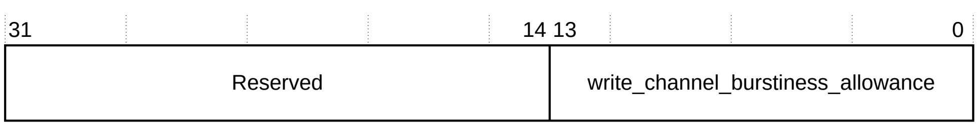
An instance of this register exists for each completer interface when write QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-96: ASNI_QOSWRBUR bit assignments



The following table shows the bit descriptions.

Table 13-107: 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.

13.11.2.27 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

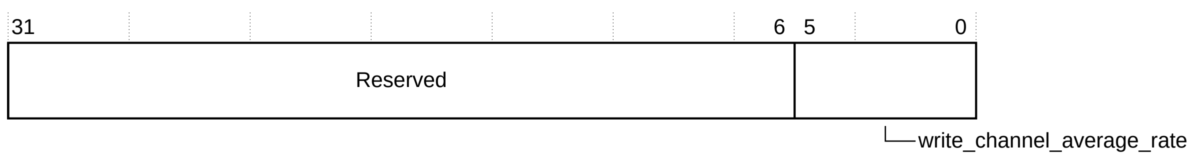
An instance of this register exists for each completer interface when write QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-97: ASNI_QOSWRAVG bit assignments



The following table shows the bit descriptions.

Table 13-108: 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.

13.11.2.28 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

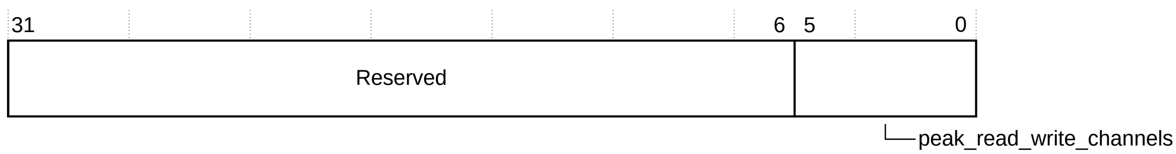
An instance of this register exists for each completer interface when combined QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-98: ASNI_QOSCOMPK bit assignments



The following table shows the bit descriptions.

Table 13-109: 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.

13.11.2.29 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

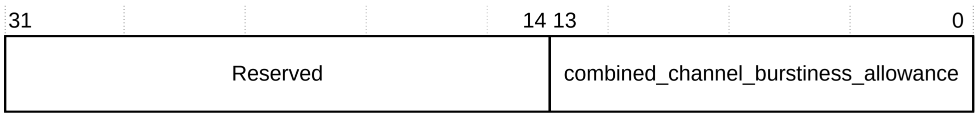
An instance of this register exists for each completer interface when combined QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-99: ASNI_QOSCOMBUR bit assignments



The following table shows the bit descriptions.

Table 13-110: 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

13.11.2.30 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

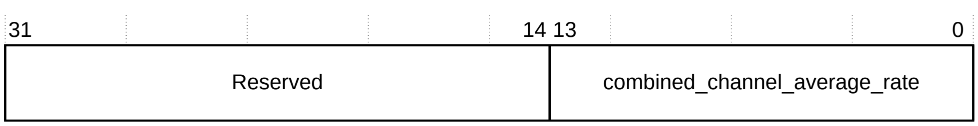
An instance of this register exists for each completer interface when combined QoS regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-100: ASNI_QOSCOMAVG bit assignments



The following table shows the bit descriptions.

Table 13-111: ASNI_QOSCOMAVG bit descriptions

Bits	Name	Description
[31:14]	-	Reserved

Bits	Name	Description
[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.

13.11.2.31 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

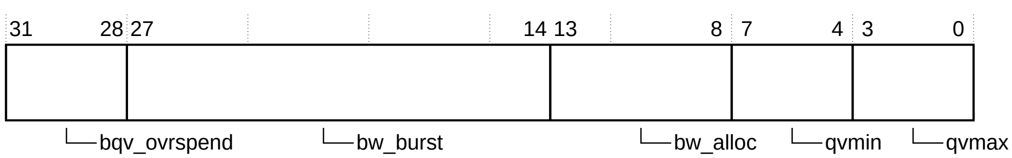
An instance of this register exists for each completer interface when read QoS bandwidth regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-101: ASNI_QOSRDBQV bit assignments



The following table shows the bit descriptions.

Table 13-112: 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.

Bits	Name	Description
[3:0]	QVMAX	BQV maximum QoS value The maximum value of ARQOS.

13.11.2.32 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

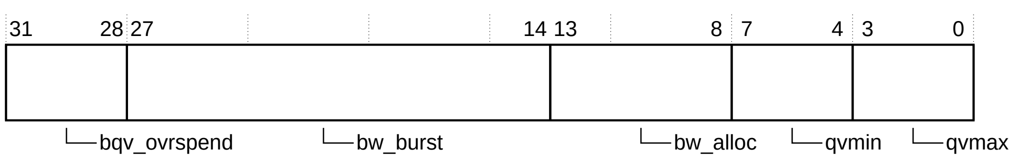
An instance of this register exists for each completer interface when write QoS bandwidth regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-102: ASNI_QOSWRBQV bit assignments



The following table shows the bit descriptions.

Table 13-113: 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.

13.11.2.33 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

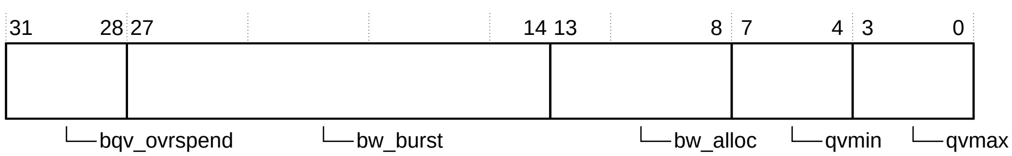
An instance of this register exists for each completer interface when combined QoS bandwidth regulators have been configured in the Socrates IP Tooling.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-103: ASNI_QOSCOMBQV bit assignments



The following table shows the bit descriptions.

Table 13-114: 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.

13.11.2.34 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 to permit Non-secure accesses.

Configurations

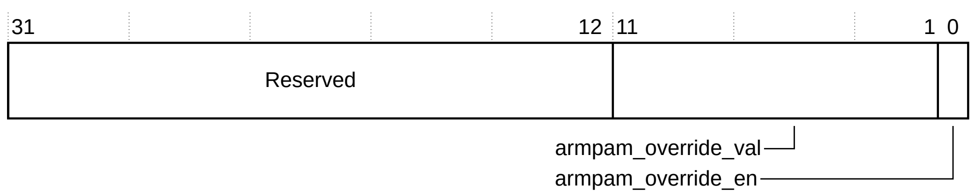
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-104: ASNI_AR_MPAM_OVERRIDE bit assignments



The following table shows the bit descriptions.

Table 13-115: 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.

13.11.2.35 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 to permit Non-secure accesses.

Configurations

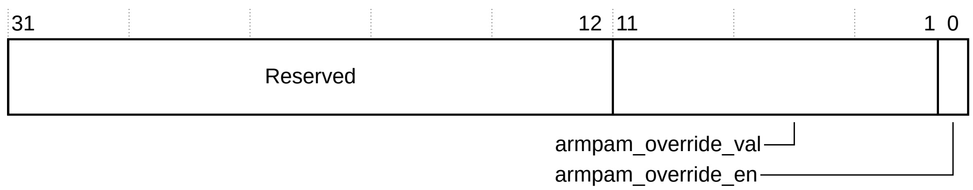
Available in all NI-700 configurations.

Attributes

For more information, see ASNI registers summary.

The following figure shows the bit assignments.

Figure 13-105: ASNI_AW_MPAM_OVERRIDE bit assignments



The following table shows the bit descriptions.

Table 13-116: ASNI_AW_MPAM_OVERRIDE bit descriptions

Bits	Name	Description
[31:12]	-	Reserved
[11:1]	awmpam_override_val	AWMPAM override value
[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.

13.12 AMNI registers

This section describes the NI-700 AMNI registers. It contains a summary of the requester interface registers, in order of address offset, and a description of the bitfields for each register.

13.12.1 AMNI registers summary

This register summary lists the NI-700 AMNI registers and some key characteristics.

The following table shows the requester 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 13-117: 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

Offset	Name	Type	Reset	Width	Description
0x004	AMNI_NODE_INFO	RO		32	AMNI_NODE_INFO, Node information for AMNI register
0x008	AMNI_SECR_ACC	RW	0x00	32	AMNI_SECR_ACC, Secure access register
0x00C	AMNI_PMUSELA	RW	0x0000	32	AMNI_PMUSELA, Configure AMNI crossbar register
0x010	AMNI_PMUSELB	RW	0x0000	32	AMNI_PMUSELB, Configure AMNI crossbar register
0x014	AMNI_INTERFACEID_0-3	RO	Configuration dependent	32	AMNI_INTERFACEID, Configure AMNI interface IDs 0-3
0x018	AMNI_INTERFACEID_4-7	RAZ		32	AMNI_INTERFACEID, Configure AMNI interface IDs 4-7
0x01C	AMNI_INTERFACEID_8-11	RAZ		32	AMNI_INTERFACEID, Configure AMNI interface IDs 8-11
0x020	AMNI_INTERFACEID_12-15	RAZ		32	AMNI_INTERFACEID, Configure AMNI interface IDs 12-15
0x040	AMNI_NODE_FEAT	RAZ	0x0000	32	AMNI_NODE_FEAT, Node features register
0x080	AMNI_SILDBG	RW or RO	0x00	32	AMNI_SILDBG, Silicon debug monitor register
0x084	AMNI_QOSACC	RW	0x00	32	AMNI_QOSACC, QoS accept control
0x088	AMNI_CONFIG_CTL	RW	0b0	32	AMNI_CONFIG_CTL, Select response
0x0F0	AMNI_INTERRUPT_STATUS	RW	0b0	32	AMNI_INTERRUPT_STATUS, Interrupt status register
0x0F4	AMNI_INTERRUPT_MASK	RW	0b0	32	AMNI_INTERRUPT_MASK, Interrupt mask register
0x0F8	AMNI_INTERRUPT_STATUS_NS	RW	0b0	32	AMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register
0x0FC	AMNI_INTERRUPT_MASK_NS	RW	0b0	32	AMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

13.12.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.12.2.1 AMNI_NODE_TYPE, Node type register for AMNI registers

This register identifies the node as an NI-700 requester interface.

Usage constraints

None.

Configurations

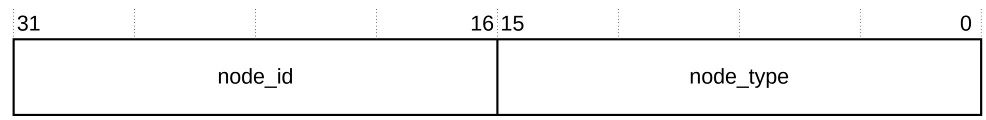
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-106: AMNI_NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-118: 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 requester interface for the NI-700 AMNI registers.

13.12.2.2 AMNI_NODE_INFO, Node information for AMNI register

This register identifies the node as an NI-700 requester interface.

Usage constraints

None.

Configurations

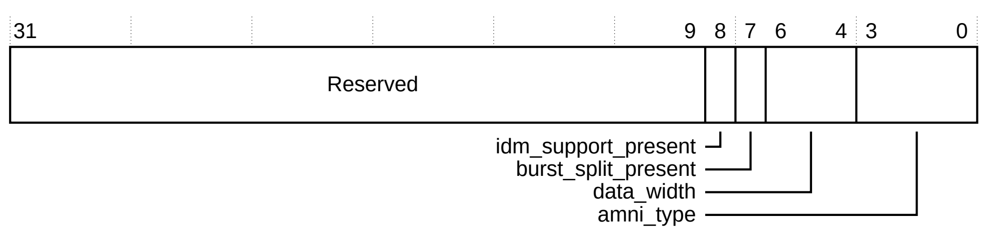
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-107: AMNI_NODE_INFO bit assignments



The following table shows the bit descriptions.

Table 13-119: AMNI_NODE_INFO bit descriptions

Bits	Name	Description
[31:9]	-	Reserved

Bits	Name	Description
[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.
[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

13.12.2.3 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. To enable Non-secure access configure bit[0].

Configurations

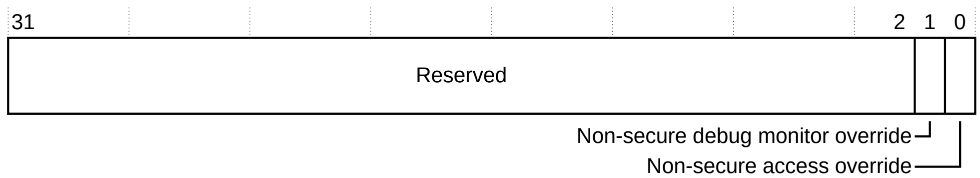
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-108: SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-120: 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.

13.12.2.4 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 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

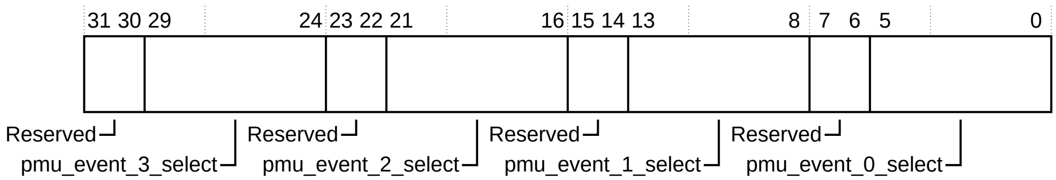
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-109: AMNI_PMUSELA bit assignments



The following table shows the bit descriptions.

Table 13-121: 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

13.12.2.5 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 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

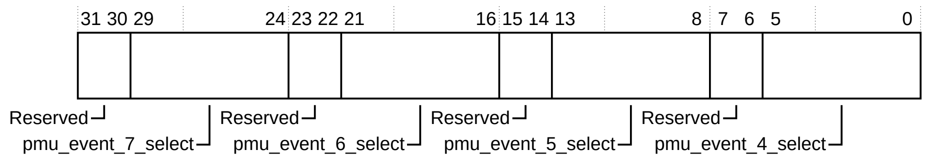
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-110: AMNI_PMUSELB bit assignments



The following table shows the bit descriptions.

Table 13-122: 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

13.12.2.6 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

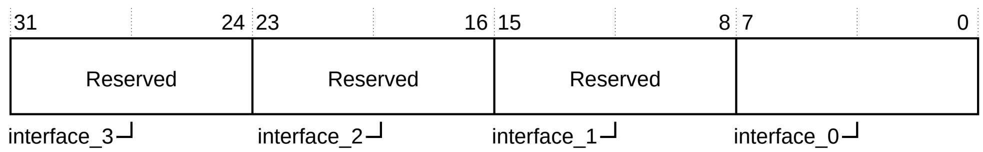
For more information, see AMNI registers summary.

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 13-111: AMNI_INTERFACEID bit assignments, AMNI interface IDs 0-3



The following table shows the bit descriptions.

Table 13-123: 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

13.12.2.7 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

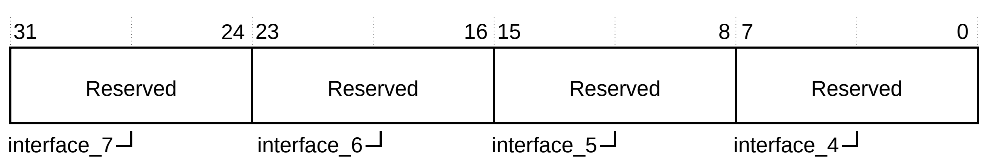
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-112: AMNI_INTERFACEID bit assignments, AMNI interface IDs 4-7



The following table shows the bit descriptions.

Table 13-124: AMNI_INTERFACEID descriptions, AMNI 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

13.12.2.8 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

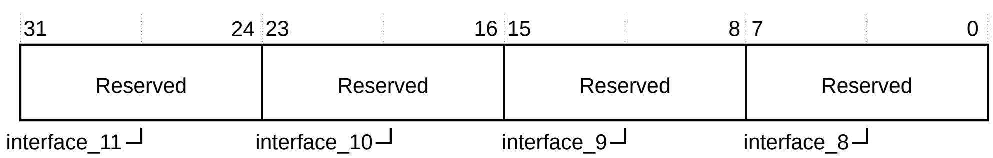
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-113: AMNI_INTERFACEID bit assignments, AMNI interface IDs 8-11



The following table shows the bit descriptions.

Table 13-125: 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

13.12.2.9 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

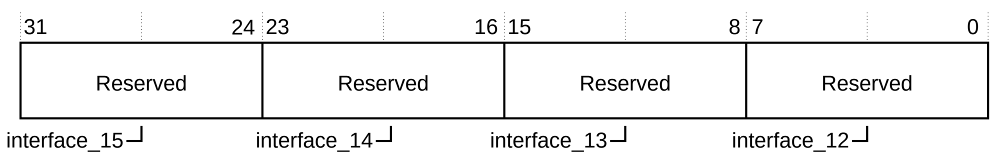
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-114: AMNI_INTERFACEID bit assignments, AMNI interface IDs 12-15



The following table shows the bit descriptions.

Table 13-126: 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

13.12.2.10 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 to permit Non-secure accesses.

Configurations

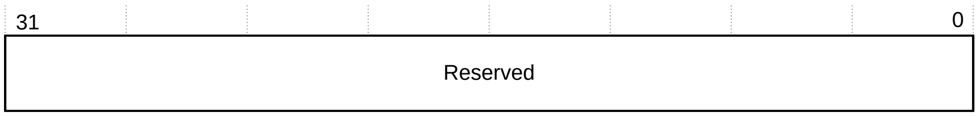
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-115: AMNI_NODE_FEAT bit assignments



The following table shows the bit descriptions.

Table 13-127: AMNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:0]	-	Reserved

13.12.2.11 AMNI_SILDBG, Silicon debug monitor register

This register monitors the status of NI-700 requester interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the AMNI_SECR_ACC, Secure access register 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

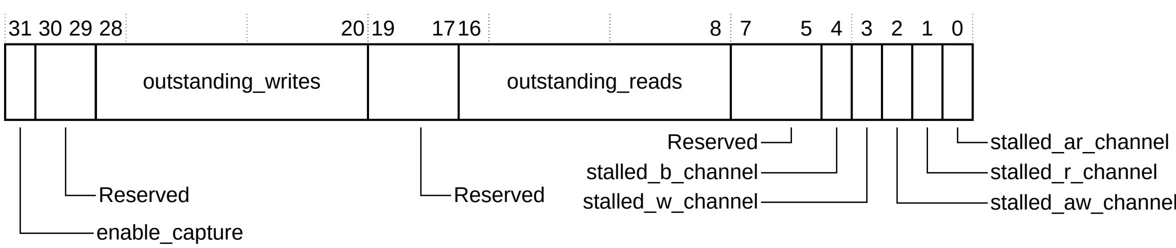
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-116: AMNI_SILDBG bit assignments



The following table shows the bit descriptions.

Table 13-128: 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.

13.12.2.12 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 to permit Non-secure accesses. This register can only be modified with prior written permission from Arm.

Configurations

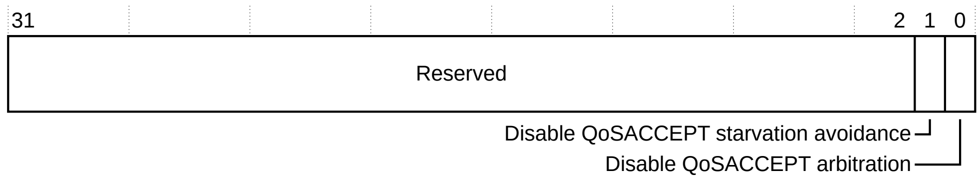
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-117: AMNI_QOSACC bit assignments



The following table shows the bit descriptions.

Table 13-129: 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.

13.12.2.13 AMNI_CONFIG_CTL, Select response

This register selects between SLVERR or OKAY responses to handle CMOs when downstream completers do not support it.

Usage constraints

Accessible using Secure transactions only, unless you configure the AMNI_SECR_ACC, Secure access register to permit Non-secure accesses. Configure bit[0] of the Secure access register to permit Non-secure accesses to access the register.

Configurations

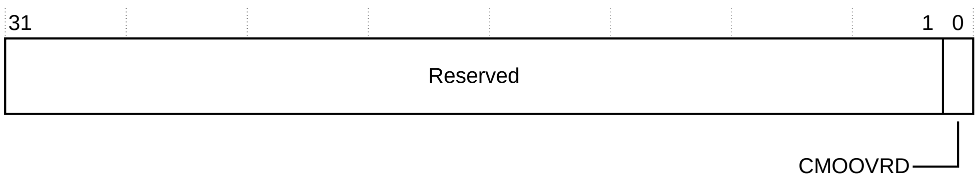
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary and Requester network interface error responses.

The following figure shows the bit assignments.

Figure 13-118: AMNI_CONFIG_CTL bit assignments



The following table shows the bit descriptions.

Table 13-130: 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 Requester network interface error responses.

13.12.2.14 AMNI_INTERRUPT_STATUS, Interrupt status register

This register indicates the interrupt status of Secure transactions.

Usage constraints

Accessible using Secure transactions only. To permit Non-secure accesses configure bit[0] of the AMNI_SECR_ACC register. For more information on Non-secure accesses, see AMNI_SECR_ACC, Secure access register.

Configurations

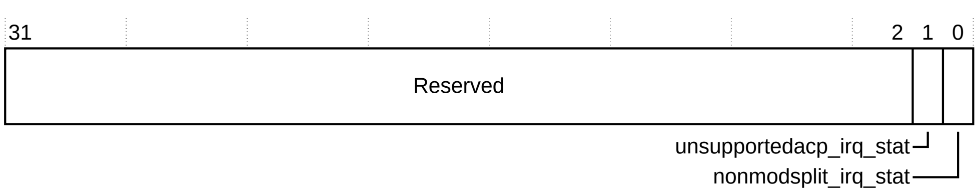
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-119: AMNI_INTERRUPT_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-131: 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.

13.12.2.15 AMNI_INTERRUPT_MASK, Interrupt mask register

This register is the interrupt mask of Secure transactions.

Usage constraints

Accessible using Secure transactions only. To permit Non-secure accesses configure bit[0] of the AMNI_SECR_ACC register. For more information on Non-secure accesses, see AMNI_SECR_ACC, Secure access register.

Configurations

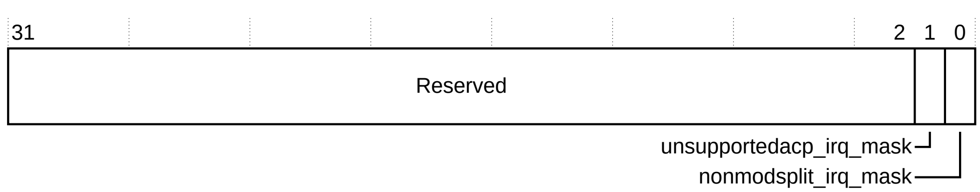
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-120: AMNI_INTERRUPT_MASK bit assignments



The following table shows the bit descriptions.

Table 13-132: AMNI_INTERRUPT_MASK bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	unsupportedacp_irq_mask	Mask the unsupported ACE5-LiteACP interrupt

Bits	Name	Description
[0]	nonmodsplit_irq_mask	Mask the non-modifiable Burst split interrupt

Note

A value of 1 indicates that the interrupt event is masked.

13.12.2.16 AMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register

This register indicates the interrupt status of Non-secure transactions.

Usage constraints

None.

Configurations

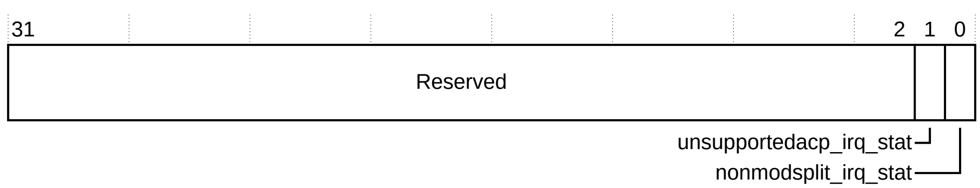
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-121: AMN_INTERRUPT_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-133: 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 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.

13.12.2.17 AMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

This register is the interrupt mask of Non-secure transactions.

Usage constraints

None.

Configurations

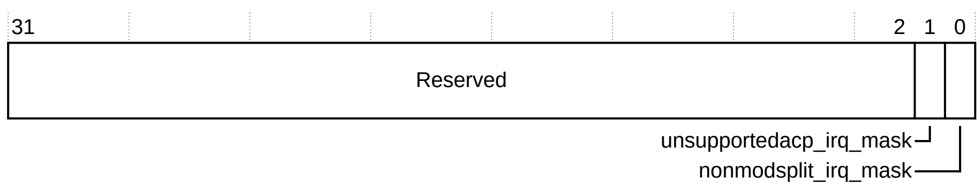
Available in all NI-700 configurations.

Attributes

For more information, see AMNI registers summary.

The following figure shows the bit assignments.

Figure 13-122: AMNI_INTERRUPT_MASK_NS bit assignments



The following table shows the bit descriptions.

Table 13-134: 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.

13.13 HSNI registers

This section describes the NI-700 HSNI registers. It contains a summary of the comppleter interface registers, in order of address offset, and a description of the bitfields for each register.

13.13.1 HSNI registers summary

This register summary lists the NI-700 HSNI registers and some key characteristics.

The following table shows the completer 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 13-135: 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
0x004	HSNI_NODE_INFO	RO	0x0000	32	HSNI_NODE_INFO, Node information for HSNI register
0x008	HSNI_SECR_ACC	RW	0x000	32	HSNI_SECR_ACC, Secure access register
0x00C	HSNI_PMUSELA	RW	0x0000	32	HSNI_PMUSELA, Configure HSNI crossbar register
0x010	HSNI_PMUSELB	RW	0x0000	32	HSNI_PMUSELB, Configure HSNI crossbar register
0x014	HSNI_INTERFACEID_0-3	RO	Configuration dependent	32	HSNI_INTERFACEID, Configure HSNI interface IDs 0-3
0x018	HSNI_INTERFACEID_4-7	RAZ		32	HSNI_INTERFACEID, Configure HSNI interface IDs 4-7
0x01C	HSNI_INTERFACEID_8-11	RAZ		32	HSNI_INTERFACEID, Configure HSNI interface IDs 8-11
0x020	HSNI_INTERFACEID_12-15	RAZ		32	HSNI_INTERFACEID, Configure HSNI interface IDs 12-15
0x040	HSNI_NODE_FEAT	RAZ	0x0000	32	HSNI_NODE_FEAT, Node features register
0x044	HSNI_CTRL	RW/RO	-	32	HSNI_CTRL, HSNI control register
0x048	HSNI_ADDR_REMAP	RW	0x00	32	HSNI_ADDR_REMAP, Address remap vector register
0x080	HSNI_SILDBG	RW/RO	0x00	32	HSNI_SILDBG, HSNI silicon debug monitor register
0x084	HSNI_QOSCTL	RW	0x00	32	HSNI_QOSCTL, QoS control register
0x088	HSNI_WDATTHRS	RW	0x00	32	HSNI_WDATTHRS, Write data FIFO threshold register
0x090	HSNI_AWQOSOVR	RW	0x00	32	HSNI_AWQOSOVR, Write channel QoS value override register
0x0A0	HSNI_QOSOT	RW	0x00	32	HSNI_AXQOSOT, Maximum combined Outstanding Transactions register
0x0BC	HSNI_QOSCOMPK	RW	0x00	32	HSNI_QOSCOMPK, Combined TSPEC bandwidth regulator peak rate register
0x0C0	HSNI_QOSCOMBUR	RW	0x0000	32	HSNI_QOSCOMBUR, Combined TSPEC bandwidth regulator burstiness allowance register
0x0C4	HSNI_QOSCOMAVG	RW	0x00	32	HSNI_QOSCOMAVG, Combined TSPEC bandwidth regulator average rate register
0x0D0	HSNI_QOSCOMBQV	RW	0x00	32	HSNI_QOSCOMBQV, Combined BQV bandwidth regulator target bandwidth register
0x0E0	Request MPAM Override	RW	0x0	32	Request MPAM override register
0x0F0	HSNI_INTERRUPT_STATUS	RW	0x0	32	HSNI_INTERRUPT_STATUS, Interrupt status register
0x0F4	HSNI_INTERRUPT_MASK	RW	0x0	32	HSNI_INTERRUPT_MASK, Interrupt mask register

Offset	Name	Type	Reset	Width	Description
`0x0F8`	HSNI_INTERRUPT_STATUS_NS	RW	`0x0`	32	HSNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register
`0x0FC`	HSNI_INTERRUPT_MASK_NS	RW	`0x0`	32	HSNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

13.13.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.13.2.1 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

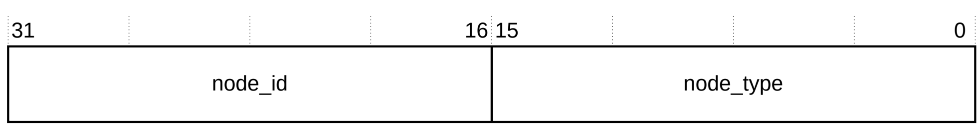
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-123: HSNI_NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-136: 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.

13.13.2.2 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

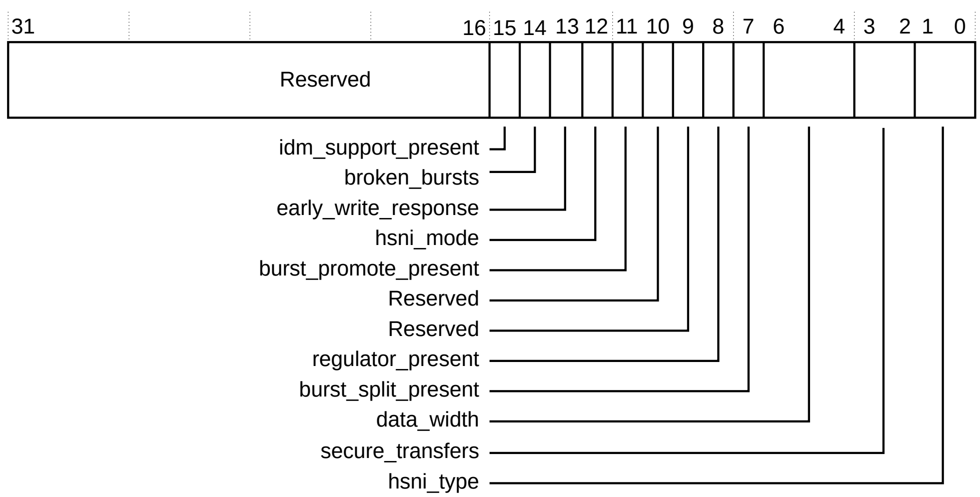
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-124: HSNI_NODE_INFO bit assignments



The following table shows the bit descriptions.

Table 13-137: 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.

Bits	Name	Description
[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
[3:2]	secure_transfers	**`0b00`** The software programmable register to set the security attribute for requests from this completer interface. **`0b01`** The HSONSEC pin exists and is used to pass the security attribute. **`0b02`** All requests which originate from this completer interface are marked Secure. Configure at build time. **`0b03`** All requests which originate from this completer interface are marked Non-secure. Configure at build time.
[1:0]	hsni_type	HSNI type and property **0** Extended memory type **1** Exclusive transfers

13.13.2.3 HSNI_SECR_ACC, Secure access register

This register controls Secure access.

Usage constraints

Read from and write to this register using Secure transactions only. To enable Non-secure access configure bit[0].

Configurations

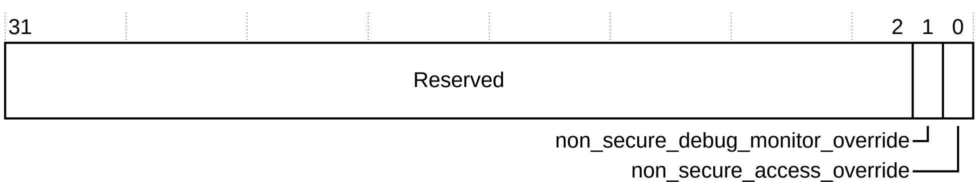
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-125: HSNI_SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-138: 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.

13.13.2.4 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 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

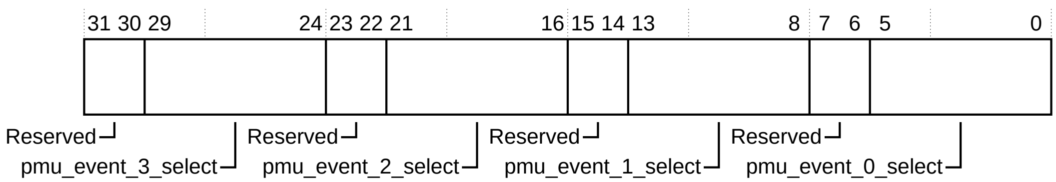
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-126: HSNI_PMUSELA bit assignments



The following table shows the bit descriptions.

Table 13-139: 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

13.13.2.5 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 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

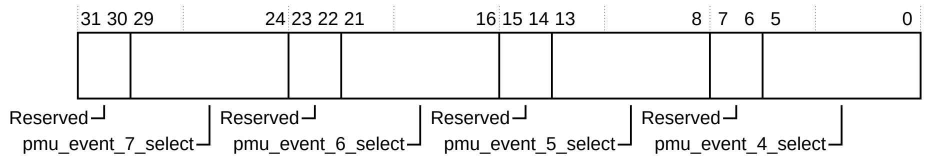
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-127: HSNI_PMUSELB bit assignments



The following table shows the bit descriptions.

Table 13-140: 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

13.13.2.6 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

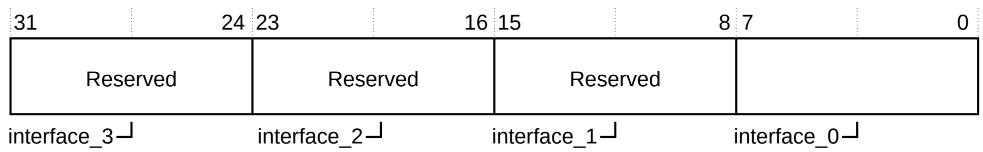
For more information, see HSNI registers summary.

Note

The 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 13-128: HSNI_INTERFACEID bit assignments, HSNI interface IDs 0-3



The following table shows the bit descriptions.

Table 13-141: 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

13.13.2.7 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

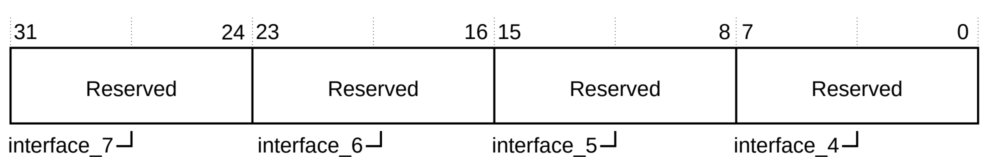
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-129: HSNI_INTERFACEID bit assignments, HSNI interface IDs 4-7



The following table shows the bit descriptions.

Table 13-142: 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

13.13.2.8 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

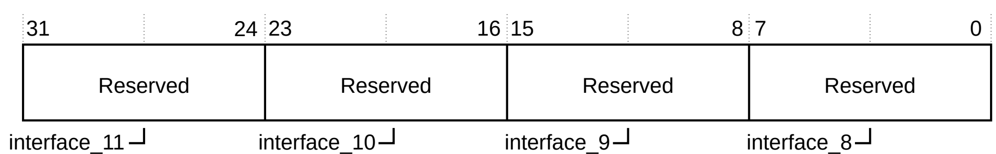
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-130: HSNI_INTERFACEID bit assignments, HSNI interface IDs 8-11



The following table shows the bit descriptions.

Table 13-143: 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

13.13.2.9 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

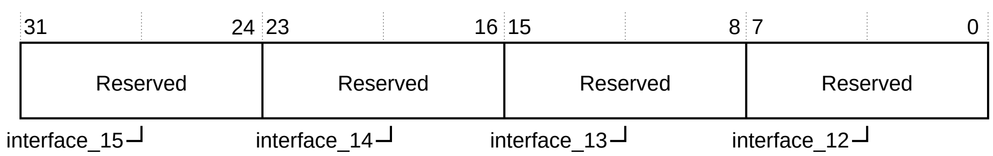
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-131: HSNI_INTERFACEID bit assignments, HSNI interface IDs 12-15



The following table shows the bit descriptions.

Table 13-144: 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

13.13.2.10 HSNI_NODE_FEAT, Node features register

This register configures the node features.

Usage constraints

Accessible using only Secure accesses.

Configurations

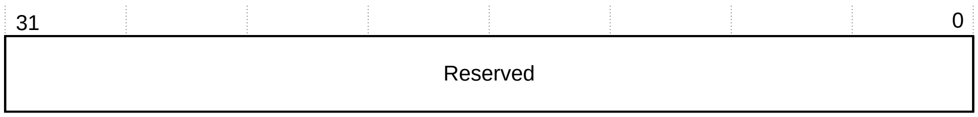
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-132: HSNI_NODE_FEAT bit assignments



The following table shows the bit descriptions.

Table 13-145: HSNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:0]	-	Reserved

13.13.2.11 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 to permit Non-secure accesses.

Configurations

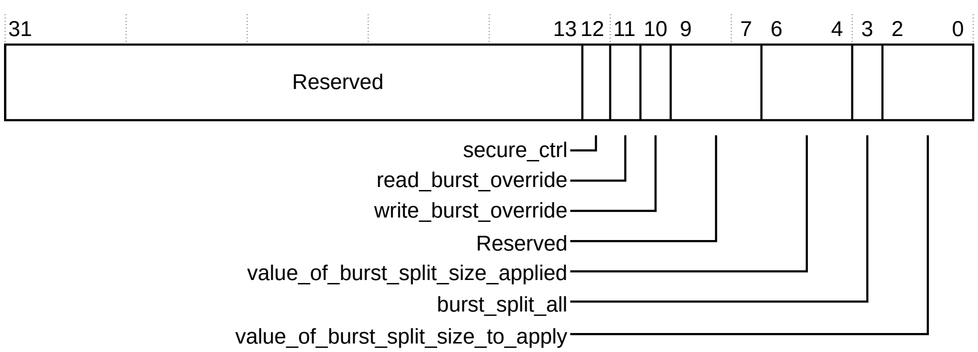
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-133: HSNI_CTRL bit assignments



The following table shows the bit descriptions.

Table 13-146: 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 completer. **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.
[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.

13.13.2.12 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 to permit Non-secure accesses.

Configurations

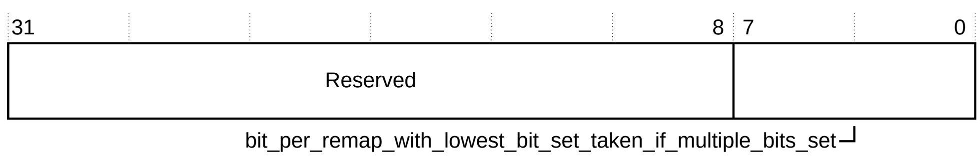
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-134: HSNI_ADDR_REMAP bit assignments



The following table shows the bit descriptions.

Table 13-147: 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.

13.13.2.13 HSNI_SILDBG, HSNI silicon debug monitor register

This register monitors the status of NI-700 completer interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the HSNI_SECR_ACC, Secure access register 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

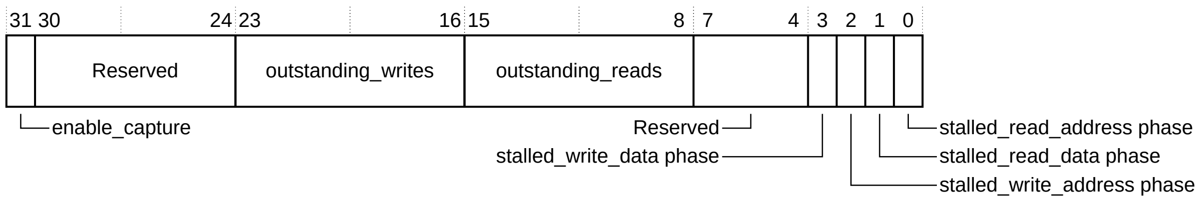
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-135: HSNI_SILDBG bit assignments



The following table shows the bit descriptions.

Table 13-148: 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.

13.13.2.14 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 to permit Non-secure accesses.

Configurations

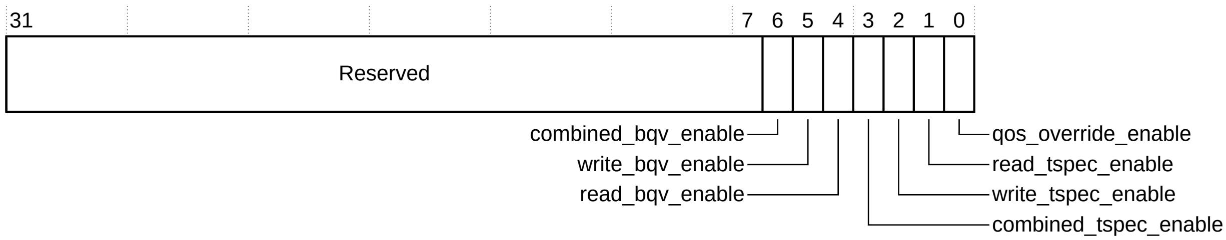
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-136: HSNI_QOSCTL bit assignments



The following table shows the bit descriptions.

Table 13-149: 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: 1. Hard Bandwidth Regulator Burstiness Allowance > 0 2. Hard Bandwidth Regulator Peak Rate > *Hard Bandwidth Regulator Average Rate
[2]	write_tspec_enable	Reserved
[1]	read_tspec_enable	Reserved
[0]	qos_override_enable	Reserved

13.13.2.15 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 to permit Non-secure accesses.

Configurations

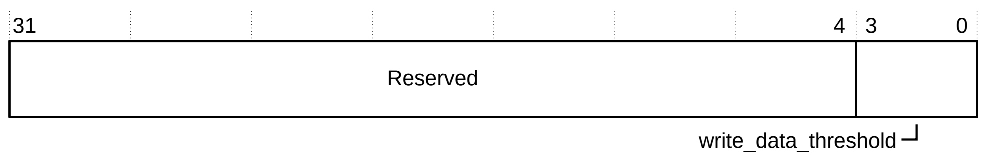
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-137: HSNI_WDATTHRS bit assignments



The following table shows the bit descriptions.

Table 13-150: 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.

13.13.2.16 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 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

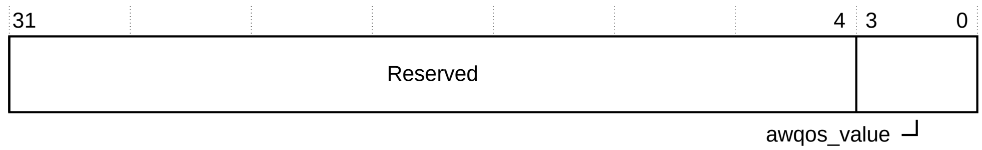
A copy of this register exists for each completer interface.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-138: HSNI_AWQOSOVR bit assignments



The following table shows the bit descriptions.

Table 13-151: HSNI_AWQOSOVR bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3:0]	awqos_value	AWQOS value override for the completer interface.

13.13.2.17 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 completer 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 to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

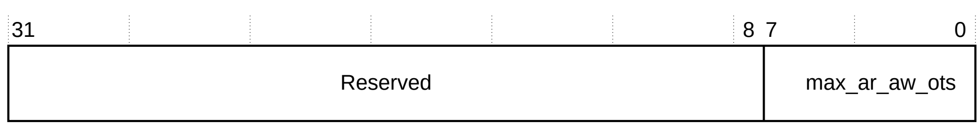
An instance of this register exists for each completer interface.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-139: HSNI_AXQOSOT bit assignments



The bit descriptions are shown in the following table.

Table 13-152: HSNI_AXQOSOT bit descriptions

Bits	Name	Description
[31:8]	-	Reserved
[7:0]	max_ar_aw_ots	The maximum number of OTs for the completer interface. This value is a combined issuing limit. It represents the maximum number of transactions that the upstream requester 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.

13.13.2.18 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the HSNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

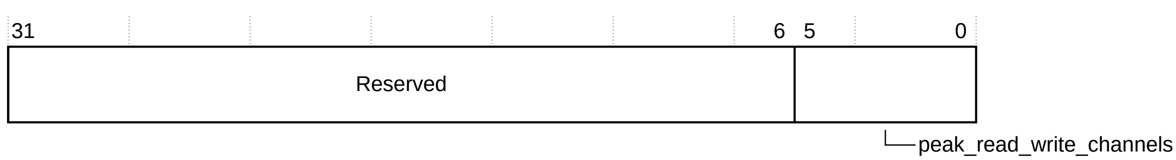
An instance of this register exists for each completer interface.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-140: HSNI_QOSCOMPK bit assignments



The following table shows the bit descriptions.

Table 13-153: 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.

13.13.2.19 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the HSNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

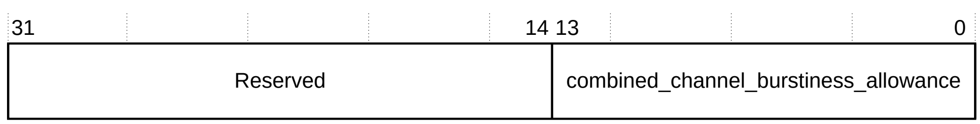
An instance of this register exists for each completer interface.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-141: HSNI_QOSCOMBUR bit assignments



The following table shows the bit descriptions.

Table 13-154: 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.

13.13.2.20 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the HSNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

Available in all NI-700 configurations.

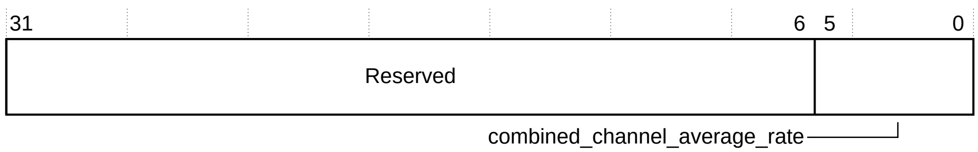
An instance of this register exists for each completer interface.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-142: HSNI_QOSCOMAVG bit assignments



The following table shows the bit descriptions.

Table 13-155: 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.

13.13.2.21 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 completer interface.

Usage constraints

Accessible using only Secure accesses, unless you set the HSNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

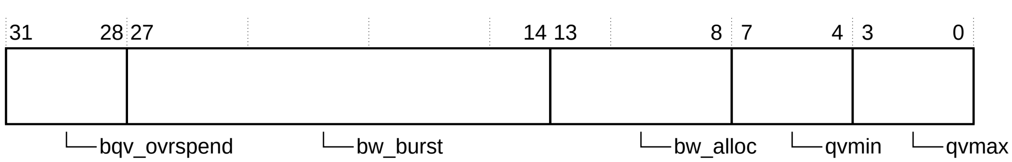
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-143: HSNI_QOSCOMBQV bit assignments



The following table shows the bit descriptions.

Table 13-156: 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.

13.13.2.22 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 to permit Non-secure accesses.

Configurations

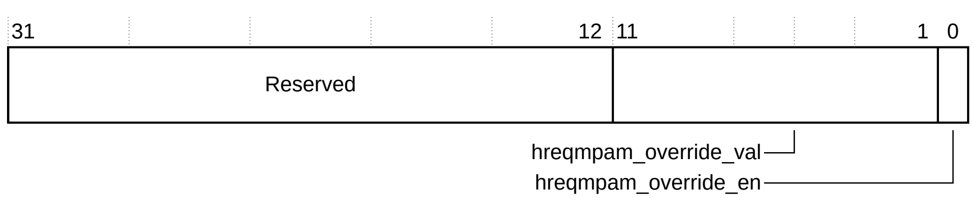
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-144: Request MPAM override bit assignments



The following table shows the bit descriptions.

Table 13-157: Request MPAM override bit descriptions

Bits	Name	Description
[31:12]	-	Reserved
[11:1]	hreqpam_override_val	ARMPAM override value

Bits	Name	Description
[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.

13.13.2.23 HSNI_INTERRUPT_STATUS, Interrupt status register

This register indicates the interrupt status of Secure transactions.

Usage constraints

Accessible using Secure transactions only.Accessible using Secure transactions only. To permit Non-secure accesses configure bit[0] of the HSNI_SECR_ACC register. For more information on Non-secure accesses, see HSNI_SECR_ACC, Secure access register.

Configurations

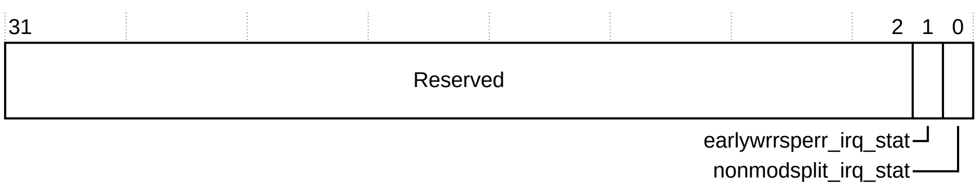
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-145: HSNI_INTERRUPT_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-158: 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.

13.13.2.24 HSNI_INTERRUPT_MASK, Interrupt mask register

This register is the interrupt mask of Secure transactions.

Usage constraints

Accessible using Secure transactions only. To permit Non-secure accesses configure bit[0] of the HSNI_SECR_ACC register. For more information on Non-secure accesses, see HSNI_SECR_ACC, Secure access register.

Configurations

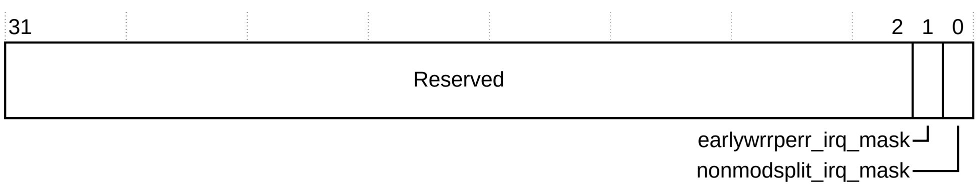
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-146: HSNI_INTERRUPT_MASK bit assignments



The following table shows the bit descriptions.

Table 13-159: 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.

13.13.2.25 HSNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register

This register indicates the interrupt status of Non-secure transactions.

Usage constraints

None.

Configurations

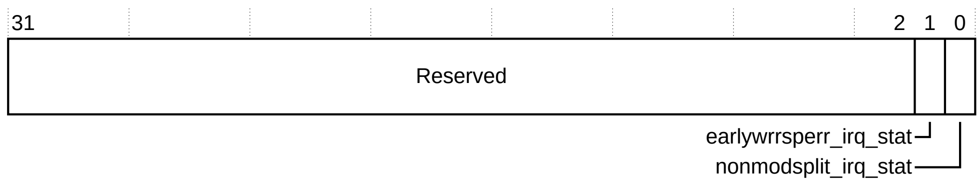
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-147: HSNI_INTERRUPT_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-160: 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.

13.13.2.26 HSNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

This register is the interrupt mask of Non-secure transactions.

Usage constraints

None.

Configurations

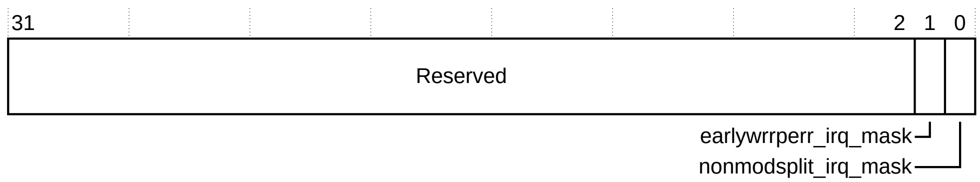
Available in all NI-700 configurations.

Attributes

For more information, see HSNI registers summary.

The following figure shows the bit assignments.

Figure 13-148: HSNI_INTERRUPT_MASK_NS bit assignments



The following table shows the bit descriptions.

Table 13-161: 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.

13.14 HMNI registers

This section describes the NI-700 *HMNI* registers. It contains a summary of the requester interface registers, in order of address offset, and a description of the bitfields for each register.

13.14.1 HMNI registers summary

This register summary lists the NI-700 HMNI registers and some key characteristics.

The following table shows the requester 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 13-162: 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
0x004	HMNI_NODE_INFO	RO	0x0000	32	HMNI_NODE_INFO, Node information for HMNI register
0x040	HMNI_NODE_FEAT	RAZ	0x0000	32	HMNI_NODE_FEAT, Node features register

Offset	Name	Type	Reset	Width	Description
0x008	HMNI_SECR_ACC	RW	0x00	32	HMNI_SECR_ACC, Secure access register
0x044	HMNI_CTRL	RW	0x0	32	HMNI_CTRL, HMNI control register
0x00C	HMNI_PMUSELA	RW	0x0000	32	HMNI_PMUSELA, Configure HMNI crossbar register
0x010	HMNI_PMUSELB	RW	0x0000	32	HMNI_PMUSELB, Configure HMNI crossbar register
0x014	HMNI_INTERFACEID_0-3	RO	Configuration dependent	32	HMNI_INTERFACEID, Configure HMNI interface IDs 0-3
0x018	HMNI_INTERFACEID_4-7	RAZ		32	HMNI_INTERFACEID, Configure HMNI interface IDs 4-7
0x01C	HMNI_INTERFACEID_8-11	RAZ		32	HMNI_INTERFACEID, Configure HMNI interface IDs 8-11
0x020	HMNI_INTERFACEID_12-15	RAZ		32	HMNI_INTERFACEID, Configure HMNI interface IDs 12-15
0x080	HMNI_SILDBG	RW/ RO	0x00	32	HMNI_SILDBG, HMNI Silicon debug monitor register
0x0F0	HMNI_INTERRUPT_STATUS	RW	0x0	32	HMNI_INTERRUPT_STATUS, Interrupt status register
0x0F4	HMNI_INTERRUPT_MASK	RW	0x0	32	HMNI_INTERRUPT_MASK, Interrupt mask register
0x0F8	HMNI_INTERRUPT_STATUS_NS	RW	0x0	32	HMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register
0x0FC	HMNI_INTERRUPT_MASK_NS	RW	0x0	32	HMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

13.14.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.14.2.1 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

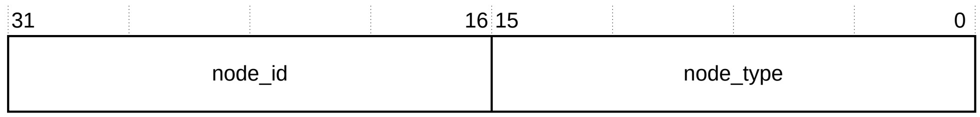
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-149: HMNI_NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-163: 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.

13.14.2.2 HMNI_NODE_INFO, Node information for HMNI register

This register provides node information for HMNI, such as data width.

Usage constraints

None.

Configurations

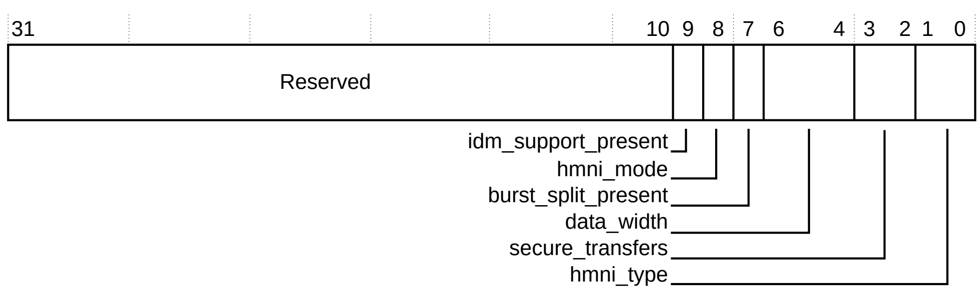
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-150: HMNI_NODE_INFO bit assignments



The following table shows the bit descriptions.

Table 13-164: 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
[3:2]	secure_transfers	**0b00** If secure_transfers = 00 the software programs this register to set the security attribute of the downstream completer of this requester interface. **Note:** If secure_transfers = 02, then transfers are always set to Secure. The downstream HMNI completer interface assets of the requester are Secure. Therefore only Secure requests can travel downstream. **Note:** If secure_transfers = 03 then transfers are always Non-secure. The downstream HMNI completer interface assets of the requester 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

13.14.2.3 HMNI_NODE_FEAT, Node features register

This register configures the node features.

Usage constraints

Accessible using only Secure accesses.

Configurations

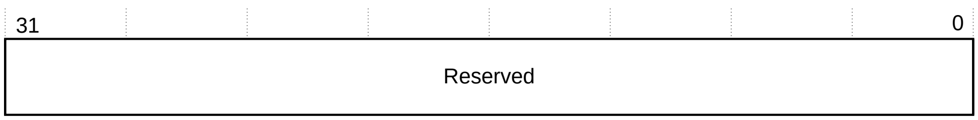
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-151: HMNI_NODE_FEAT bit assignments



The following table shows the bit descriptions.

Table 13-165: HMNI_NODE_FEAT bit descriptions

Bits	Name	Description
[31:0]	-	Reserved

13.14.2.4 HMNI_SECR_ACC, Secure access register

This register controls Secure access.

Usage constraints

Read from and write to this register using Secure transactions only. To enable Non-secure access configure bit[0].

Configurations

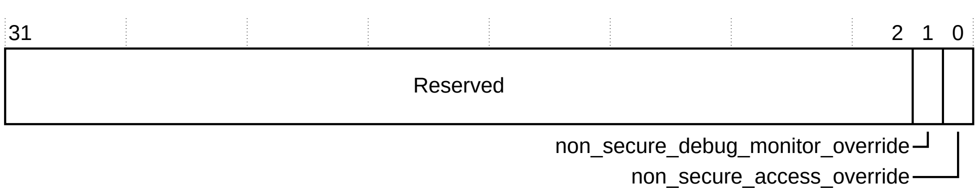
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-152: HMNI_SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-166: 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.

13.14.2.5 HMNI_CTRL, HMNI control register

This register controls how transactions access components that are attached to the requester.

Usage constraints

Accessible using only Secure accesses, unless you set the HMNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

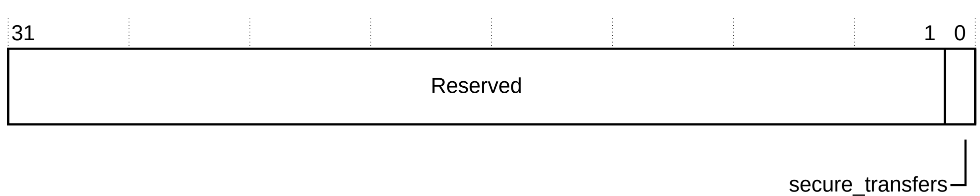
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-153: HMNI_CTRL bit assignments



The following table shows the bit descriptions.

Table 13-167: HMNI_CTRL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved

Bits	Name	Description
[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 completers 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 completer 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.

13.14.2.6 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 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

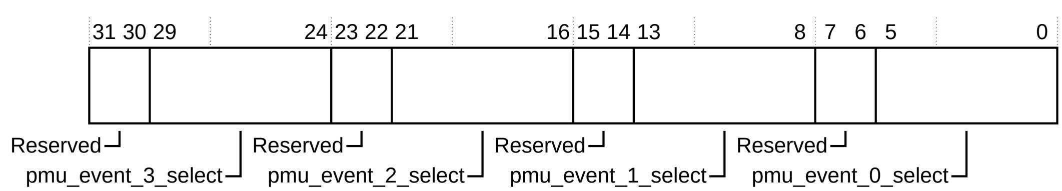
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-154: HMNI_PMUSELA bit assignments



The following table shows the bit descriptions.

Table 13-168: 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
[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

13.14.2.7 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 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

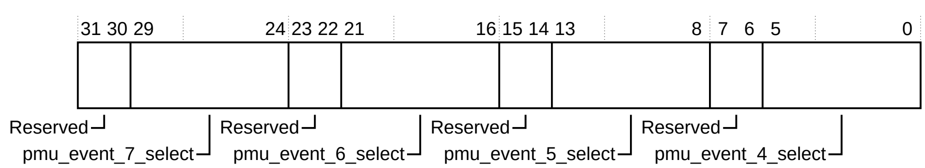
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-155: HMNI_PMUSELB bit assignments



The following table shows the bit descriptions.

Table 13-169: 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

Bits	Name	Description
[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

13.14.2.8 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

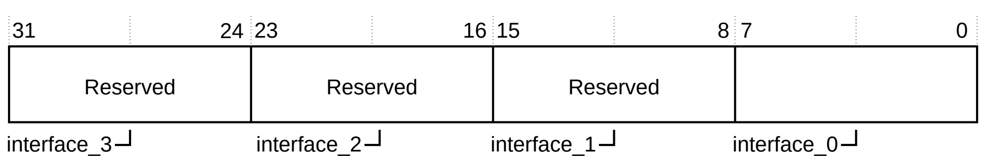
For more information, see HMNI registers summary.

Note

The 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 13-156: HMNI_INTERFACEID bit assignments, HMNI interface IDs 0-3



The following table shows the bit descriptions.

Table 13-170: 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

Bits	Name	Description
[7:0]	interface_0	HMNI interface ID 0

13.14.2.9 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

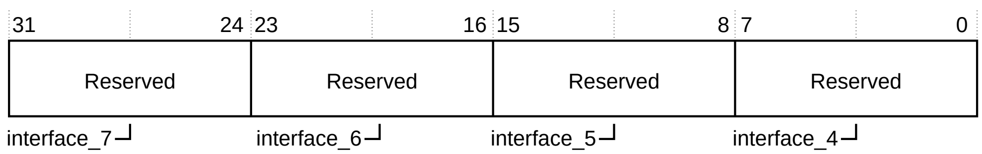
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-157: HMNI_INTERFACEID bit assignments, HMNI interface IDs 4-7



The following table shows the bit descriptions.

Table 13-171: 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

13.14.2.10 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

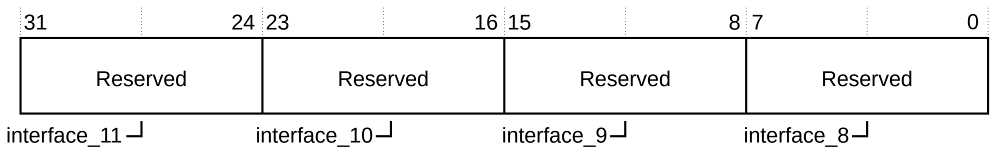
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-158: HMNI_INTERFACEID bit assignments, HMNI interface IDs 8-11



The following table shows the bit descriptions.

Table 13-172: 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

13.14.2.11 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

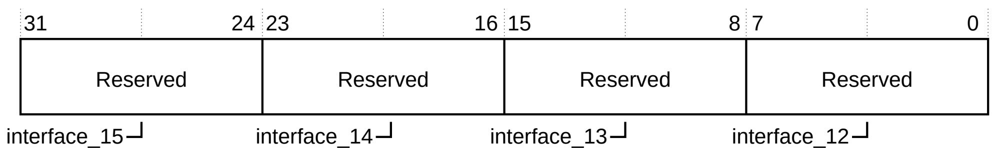
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-159: HMNI_INTERFACEID bit assignments, HMNI interface IDs 12-15



The following table shows the bit descriptions.

Table 13-173: 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

13.14.2.12 HMNI_SILDBG, HMNI Silicon debug monitor register

This register monitors the status of NI-700 requester interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the HMNI_SECR_ACC, Secure access register 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

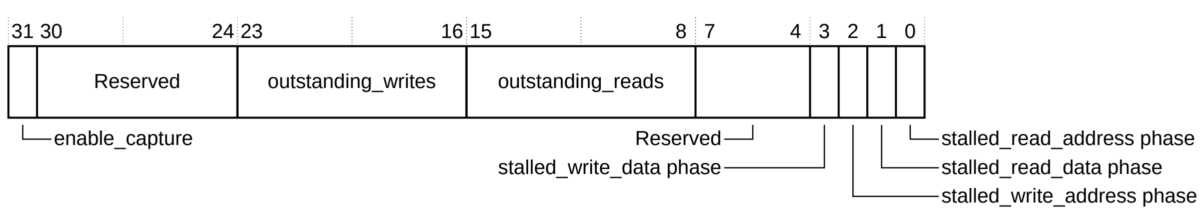
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-160: HMNI_SILDBG bit assignments



The following table shows the bit descriptions.

Table 13-174: 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

Bits	Name	Description
[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.

13.14.2.13 HMNI_INTERRUPT_STATUS, Interrupt status register

This register indicates the interrupt status of Secure transactions.

Usage constraints

Accessible using Secure transactions only. To permit Non-secure accesses configure bit[0] of the HMNI_SECR_ACC register. For more information on Non-secure accesses, see HMNI_SECR_ACC, Secure access register.

Configurations

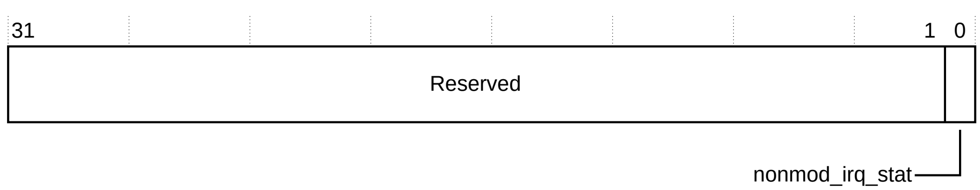
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-161: HMNI_INTERRUPT_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-175: 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.

13.14.2.14 HMNI_INTERRUPT_MASK, Interrupt mask register

This register is the interrupt mask of Secure transactions.

Usage constraints

Accessible using Secure transactions only. To permit Non-secure accesses configure bit[0] of the HMNI_SECR_ACC register. For more information on Non-secure accesses, see HMNI_SECR_ACC, Secure access register.

Configurations

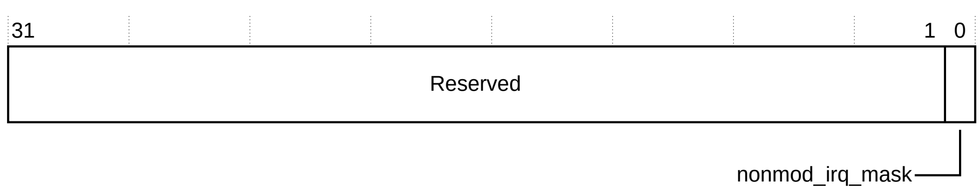
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-162: HMNI_INTERRUPT_MASK bit assignments



The following table shows the bit descriptions.

Table 13-176: 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.

13.14.2.15 HMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register

This register indicates the interrupt status of Non-secure transactions.

Usage constraints

None.

Configurations

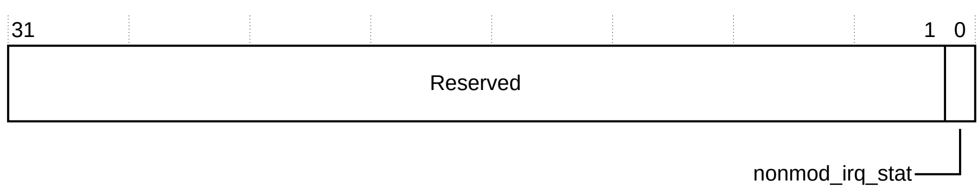
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-163: HMNI_INTERRUPT_STATUS_NS bit assignments



The following table shows the bit assignments.

Table 13-177: 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 .

13.14.2.16 HMNI_INTERRUPT_MASK_NS, Interrupt mask (Non-secure) register

This register is the interrupt mask of Non-secure transactions.

Usage constraints

None.

Configurations

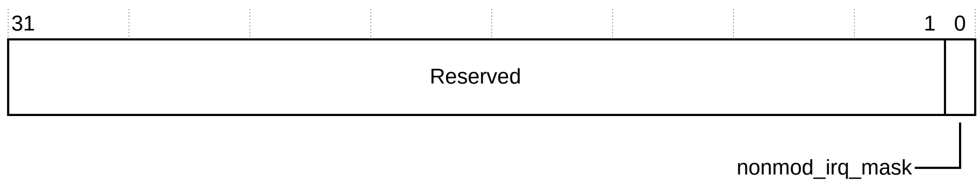
Available in all NI-700 configurations.

Attributes

For more information, see HMNI registers summary.

The following figure shows the bit assignments.

Figure 13-164: HMNI_INTERRUPT_MASK_NS bit assignments



The following table shows the bit descriptions.

Table 13-178: 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.

13.15 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.

13.15.1 Network Interface IDM registers summary

Enabling IDM functionality on a completer or requester network interface adds IDM registers to the 4KB configuration region for the network interface. The IDM registers control IDM behavior for the associated network interface.

IDM functionality can be configured on all NI-700 network interface types. If IDM is enabled on a network interface, the IDM registers start as offset `0x100` from the base address of the network interface. The register name is prefixed with the type of network interface to which it belongs, for example ASNI_IDM_DEVICE_ID.

The network interface 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 13-179: Network interface 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
0x104	IDM_CONFIG	RO	Device-specific	32	IDM_CONFIG, IDM configuration register
0x108	IDM_ERRCTLR	RW	0x0	32	IDM_ERRCTLR
0x110	IDM_ERRSTATUS	RW or RO	0x0	32	IDM_ERRSTATUS
0x114	IDM_ERRADDR_LSB	RO	0x0	32	IDM_ERRADDR_LSB
0x118	IDM_ERRADDR_MSB	RO	0x0	32	IDM_ERRADDR_MSB
0x128	IDM_ERRMISC0	RO	0x0	32	IDM_ERRMISC0
0x12C	IDM_ERRMISC1	RO	0x0	32	IDM_ERRMISC1
0x130	IDM_ACCESS_CONTROL	RW	0x0	32	IDM_ACCESS_CONTROL
0x134	IDM_ACCESS_STATUS	RW or RO	0x2	32	IDM_ACCESS_STATUS
0x138	IDM_ACCESS_READID	RO	0x0	32	IDM_ACCESS_READID
0x13C	IDM_ACCESS_WRITEID	RO	0x0	32	IDM_ACCESS_WRITEID
0x140	IDM_RESET_CONTROL	RW	0x10	32	IDM_RESET_CONTROL
0x144	IDM_RESET_STATUS	RO	0x0	32	IDM_RESET_STATUS
0x148	IDM_RESET_READID	RO	0x0	32	IDM_RESET_READID
0x14C	IDM_RESET_WRITEID	RO	0x0	32	IDM_RESET_WRITEID
0x150	IDM_TIMEOUT_CONTROL	RW	0x0	32	IDM_TIMEOUT_CONTROL
0x154	IDM_TIMEOUT_VALUE	RW	0x4	32	IDM_TIMEOUT_VALUE
0x158	IDM_INTERRUPT_STATUS	RW	0x0	32	IDM_INTERRUPT_STATUS
0x15C	IDM_INTERRUPT_MASK	RW	0x0	32	IDM_INTERRUPT_MASK
0x160	IDM_ERRSTATUS_NS	RW or RO	0x0	32	IDM_ERRSTATUS_NS
0x164	IDM_ERRADDR_LSB_NS	RO	0x0	32	IDM_ERRADDR_LSB_NS
0x168	IDM_ERRADDR_MSB_NS	RO	0x0	32	IDM_ERRADDR_MSB_NS
0x178	IDM_ERRMISC0_NS	RO	0x0	32	IDM_ERRMISC0_NS
0x17C	IDM_ERRMISC1_NS	RO	0x0	32	IDM_ERRMISC1_NS
0x184	IDM_ACCESS_STATUS_NS	RW or RO	0x0	32	IDM_ACCESS_STATUS_NS
0x188	IDM_ACCESS_READID_NS	RO	0x0	32	IDM_ACCESS_READID_NS
0x18C	IDM_ACCESS_WRITEID_NS	RO	0x0	32	IDM_ACCESS_WRITEID_NS
0x194	IDM_RESET_STATUS_NS	RO	0x0	32	IDM_RESET_STATUS_NS
0x198	IDM_RESET_READID_NS	RO	0x0	32	IDM_RESET_READID_NS
0x19C	IDM_RESET_WRITEID_NS	RO	0x0	32	IDM_RESET_WRITEID_NS

Offset	Name	Type	Reset	Width	Description
`0x1A8`	IDM_INTERRUPT_STATUS_NS	RW	`0x0`	32	IDM_INTERRUPT_STATUS_NS
`0x1AC`	IDM_INTERRUPT_MASK_NS	RW	`0x0`	32	IDM_INTERRUPT_MASK_NS

13.15.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.15.2.1 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

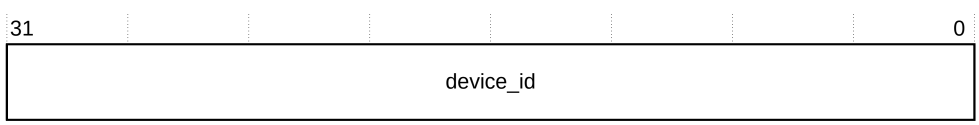
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-165: ASNI_IDM_DEVICE_ID bit assignments



The following table shows the bit descriptions.

Table 13-180: IDM_DEVICE_ID bit descriptions

Bits	Name	Description
[31:0]	device_id	Returns statically configured ID value

13.15.2.2 IDM_CONFIG, IDM configuration register

This register enables transaction logging, error detection, timeout detection, access control, and reset control.

Usage constraints

None.

Configurations

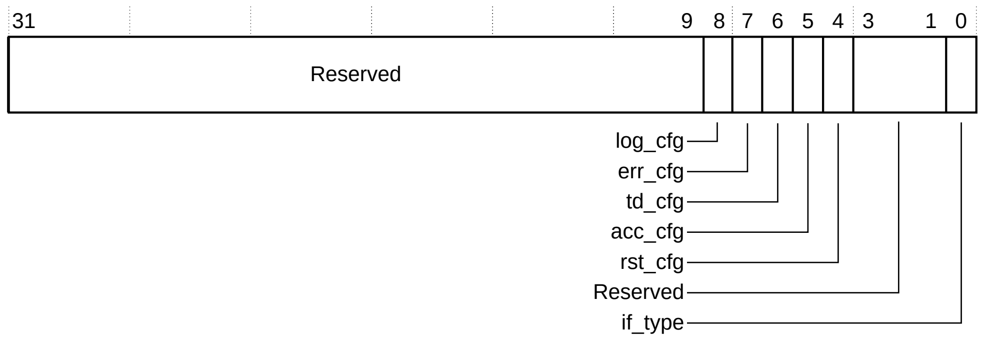
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-166: IDM_CONFIG bit assignments



The following table shows the bit descriptions.

Table 13-181: 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** Completer **1** Requester

Note

For this r2p1 release and previous releases of NI-700, if IDM support is enabled, then all of these features are enabled.

13.15.2.3 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 to permit Non-secure accesses.

Configurations

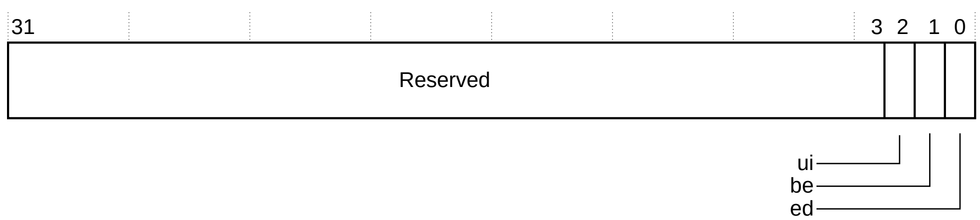
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-167: IDM_ERRCTLR bit assignments



The following table shows the bit descriptions.

Table 13-182: 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.

13.15.2.4 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

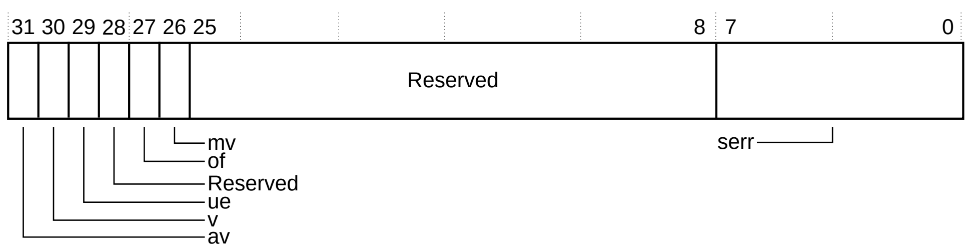
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-168: IDM_ERRSTATUS bit assignments



The following table shows the bit descriptions.

Table 13-183: 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.

Bits	Name	Description
[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.
[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 completer **20** Internal timeout

13.15.2.5 IDM_ERRADDR_LSB

This register is the error log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

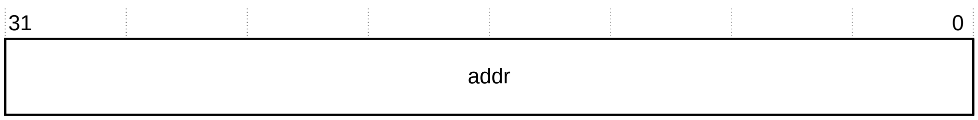
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-169: IDM_ERRADDR_LSB bit assignments



The following table shows the bit descriptions.

Table 13-184: IDM_ERRADDR_LSB bit descriptions

Bits	Name	Description
[31:0]	addr	Returns bits 31:0 of an address causing an error

13.15.2.6 IDM_ERRADDR_MSB

This register is the error log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

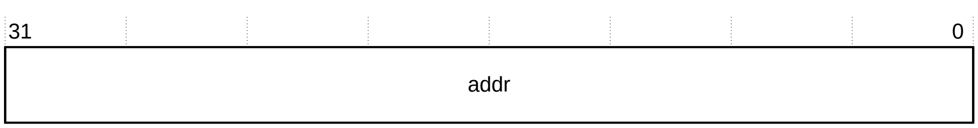
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-170: IDM_ERRADDR_MSB bit assignments



The following table shows the bit descriptions.

Table 13-185: IDM_ERRADDR_MSB bit descriptions

Bits	Name	Description
[31:0]	addr	Returns bits[63:32] of an address causing an error

13.15.2.7 IDM_ERRMISC0

This register is the error log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

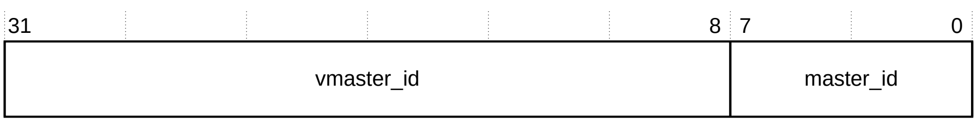
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-171: IDM_ERRMISC0 bit assignments



The following table shows the bit descriptions.

Table 13-186: 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.

13.15.2.8 IDM_ERRMISC1

This register is the error log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

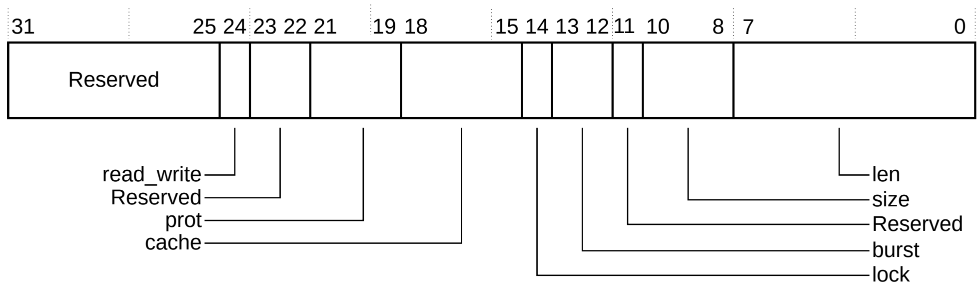
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-172: IDM_ERRMISC1 bit assignments



The following table shows the bit descriptions.

Table 13-187: 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
[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.

13.15.2.9 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 to permit Non-secure accesses.

Configurations

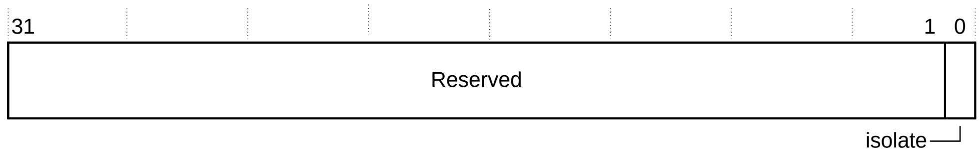
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-173: IDM_ACCESS_CONTROL bit assignments



The following table shows the bit descriptions.

Table 13-188: IDM_ACCESS_CONTROL bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	isolate	Perform gating off a device Reading 1 indicates that the completer device is gated or isolated. Reading 0 indicates that the completer 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.

13.15.2.10 IDM_ACCESS_STATUS

This register indicates the access status for Secure transactions.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

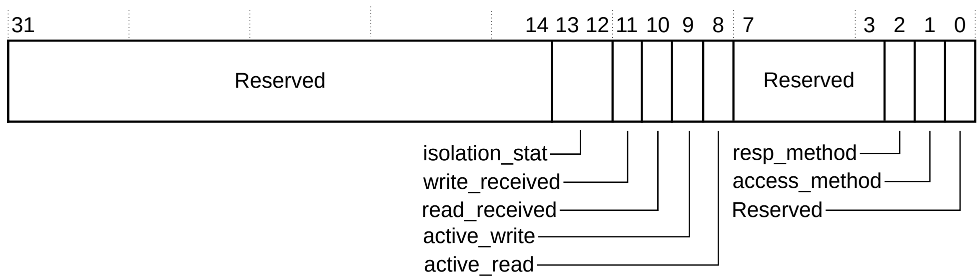
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-174: IDM_ACCESS_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-189: IDM_ACCESS_STATUS bit descriptions

Bits	Name	Description
[31:14]	Reserved	Reserved, **UNDEFINED**, write as zero
[13:12]	isolation_stat	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]	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 to 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 0 to 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 the IDM_ACCESS_CONTROL register when already in pending isolation entry state, or isolation active state. • Re-exit from isolation state. Write 0 to bit[0] of the 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.
[7:3]	Reserved	Reserved, **UNDEFINED**, write as zero

Bits	Name	Description
[2]	resp_method	Indicates device generates errors in gated access
[1]	access_method	Wait for all outstanding to complete, then block input
[0]	Reserved	Reserved, **UNDEFINED**, write as zero

13.15.2.11 IDM_ACCESS_READID

This register is the access log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

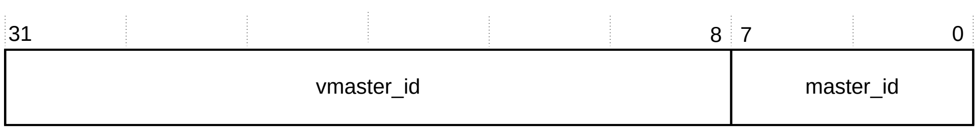
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-175: IDM_ACCESS_READID bit assignments



The following table shows the bit descriptions.

Table 13-190: IDM_ACCESS_READID bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming signal into the endpoint of the first transaction to arrive after isolation when the active_read field of the IDM_ACCESS_STATUS register is HIGH. This field depends on the incoming endpoint. Therefore vmaster_id contains the ARID of the transaction on ASNI and contains the HMASTER on HSNI. For AMNI, PMNI, and HMNI the vmaster_id matches the ID of the originating ARID or HMASTER transaction. There is no manipulation of the incoming AXI ARID signal in ASNI.
[7:0]	master_id	The originating Node ID of the ASNI or HSNI of the first transaction to arrive after isolation when the active_read field of the IDM_ACCESS_STATUS register is HIGH.

13.15.2.12 IDM_ACCESS_WRITEID

This register is the access log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

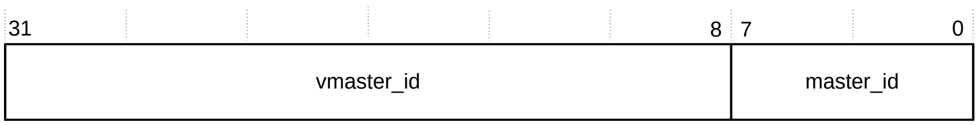
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-176: IDM_ACCESS_WRITEID bit assignments



The following table shows the bit descriptions.

Table 13-191: IDM_ACCESS_WRITEID bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming AXI AWID signal into the endpoint of the first transaction to arrive after isolation when the active_write field of the IDM_ACCESS_STATUS register is HIGH. This field depends on the incoming endpoint. Therefore vmaster_id contains the AWID of the transaction on ASNI and contains the HMASTER on HSNI. For AMNI, PMNI, and HMNI the vmaster_id matches the ID of the originating AWID or HMASTER transaction. There is no manipulation of the incoming AXI AWID signal in ASNI.
[7:0]	master_id	The originating Node ID of the ASNI or HSNI of the first transaction to arrive after isolation when the active_write field of the IDM_ACCESS_STATUS register is HIGH.

13.15.2.13 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 to permit Non-secure accesses.

Configurations

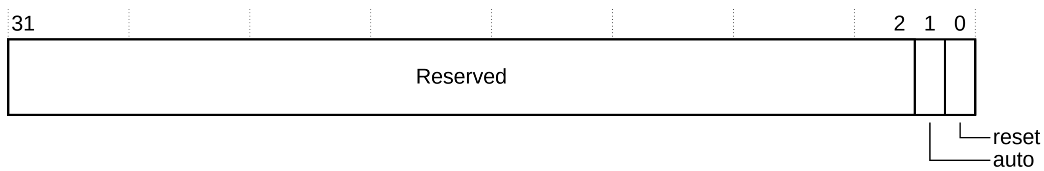
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-177: IDM_RESET_CONTROL bit assignments



The following table shows the bit descriptions:

Table 13-192: IDM_RESET_Control bit descriptions

Bits	Name	Description
[31:2]	-	Reserved, **UNDEFINED**, write as zero
[1]	auto	Configures the device for auto or internal reset mode. For more information on IDM soft reset modes, see [IDM soft reset mode]. 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 requester or completer network interfaces.. There are several constraints on this field: • You can only change this field during initialization or when the interface is fully quiesced. • Arm does not support changing this field while the interface is active. If you change this field during runtime, behavior is **UNPREDICTABLE**. Reads have the following effect: **1** A read of 1 indicates that the device is in auto or internal reset mode. **0** A read of 0 indicates that the device is not in auto or internal reset mode. Writes have the following effect: **1** A write of 1 configures the device for auto or internal reset mode. **0** A write of 0 disables auto or internal reset mode. For more information on IDM soft reset modes, see IDM soft reset mode. Bit[1] of the IDM_RESET_CONTROL register is 1 out of reset. This bit enables internal recovery mode out of reset. When not in auto reset mode and a timeout is detected, a write of 1 to the IDM_RESET_CONTROL.reset field initiates internal recovery mode. Changing this bit while the interface is not in idle mode, results in **UNPREDICTABLE** behavior.

Bits	Name	Description
[0]	reset	Performs soft reset of attached device If the auto bit is set to 1 the network interface gates the external interface, however the soft reset pin is not activated. If the auto bit is 0, the interfaces are not gated until there is a write to bit[0]. In this case, the soft reset pin is activated. Writes have the following effect: **1** Request the attached device to enter reset. If the write occurs before soft reset exit has occurred, the write is ignored. **0** Request the attached device to exit reset. If the write occurs before soft reset entry has occurred, the write is ignored. 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 IDM_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 requester or completer 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 requester or completer 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, 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. A write of 0 also clears the write_received and read_received fields of the IDM_RESET_STATUS, IDM_RESET_READID, and IDM_RESET_WRITEID registers. A write of 1 is ignored.

13.15.2.14 IDM_RESET_STATUS

This register indicates the reset status of Secure transactions.

Usage constraints

Accessible using only Secure accesses, unless you set the ASNI_SECR_ACC, Secure access register to permit Non-secure accesses.

Configurations

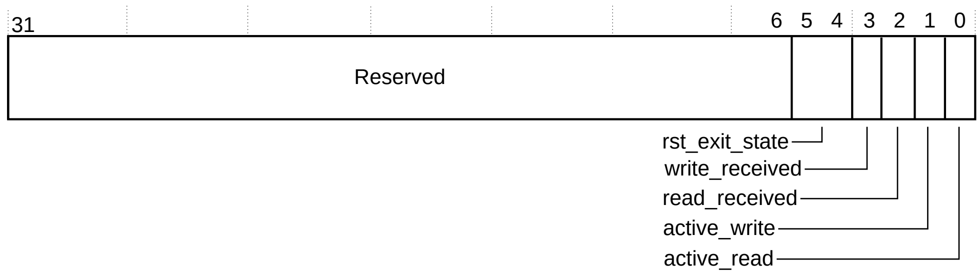
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-178: IDM_RESET_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-193: 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 Secure 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 to bit[0] of the 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 to 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 the IDM_RESET_CONTROL register. This bit is cleared to zero on: • Reentry to soft reset state Write 1 to bit[0] of the 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 to 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 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.

13.15.2.15 IDM_RESET_READID

This register is the reset access log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

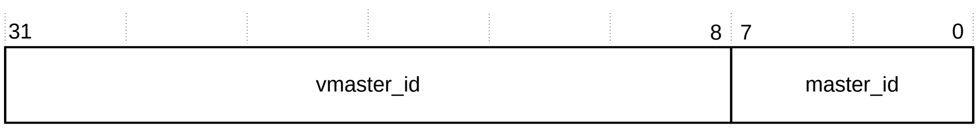
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-179: IDM_RESET_READID bit assignments



The following table shows the bit descriptions.

Table 13-194: IDM_RESET_READID bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming signal into the endpoint of the first transaction to arrive after isolation when the active_read field of the IDM_RESET_STATUS register is HIGH. This field depends on the incoming endpoint. Therefore vmaster_id contains the ARID of the transaction on ASNI and contains the HMASTER on HSNI. For AMNI, PMNI, and HMNI the vmaster_id matches the ID of the originating ARID or HMASTER transaction. There is no manipulation of the incoming AXI ARID signal in ASNI.
[7:0]	master_id	The originating Node ID of the ASNI or HSNI of the first transaction to arrive after isolation when the active_read field of the IDM_RESET_STATUS register is HIGH.

13.15.2.16 IDM_RESET_WRITEID

This register is the reset access log of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

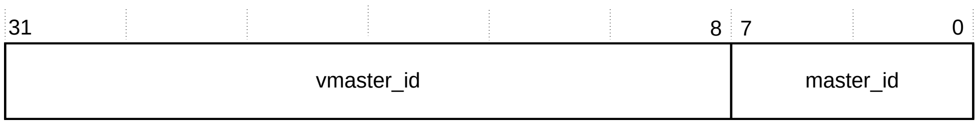
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-180: IDM_RESET_WRITEID bit assignments



The following table shows the bit descriptions.

Table 13-195: IDM_RESET_WRITEID bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming signal into the endpoint of the first transaction to arrive after isolation when the active_write field of the IDM_RESET_STATUS register is HIGH. This field depends on the incoming endpoint. Therefore vmaster_id contains the AWID of the transaction on ASNI and contains the HMASTER on HSNI. For AMNI, PMNI, and HMNI the vmaster_id matches the ID of the originating AWID or HMASTER transaction. There is no manipulation of the incoming AXI AWID signal in ASNI.
[7:0]	master_id	The originating Node ID of the ASNI or HSNI of the first transaction to arrive after isolation when the active_write field of the IDM_RESET_STATUS register is HIGH.

13.15.2.17 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 to permit Non-secure accesses.

Configurations

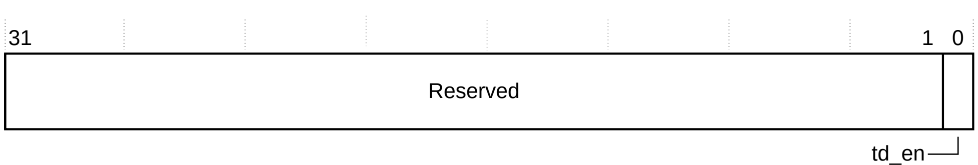
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-181: IDM_TIMEOUT_CONTROL bit assignments



The following table shows the bit descriptions.

Table 13-196: 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.

13.15.2.18 IDM_TIMEOUT_VALUE

This register controls the duration that is used to determine if a transaction has timed out.

Usage constraints

Accessible using only Secure accesses.

Configurations

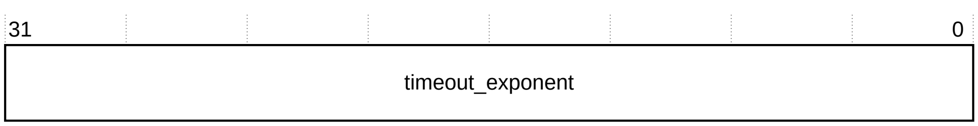
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-182: IDM_TIMEOUT_VALUE bit assignments



The following table shows the bit descriptions.

Table 13-197: 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. The maximum value is 30. Values greater than 30 are treated as 30.

13.15.2.19 IDM_INTERRUPT_STATUS

This register indicates the interrupt status of Secure transactions.

Usage constraints

Accessible using only Secure accesses.

Configurations

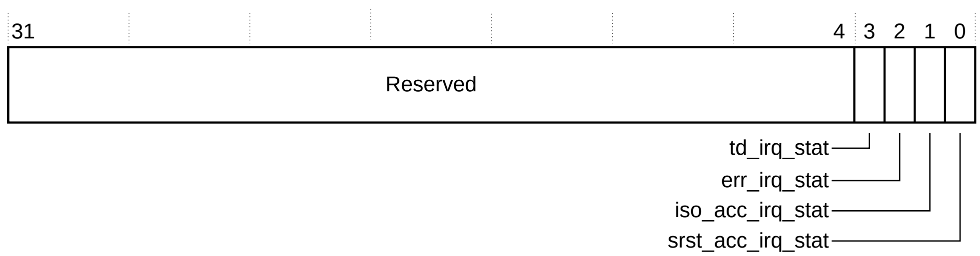
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-183: IDM_INTERRUPT_STATUS bit assignments



The following table shows the bit descriptions.

Table 13-198: IDM_INTERRUPT_STATUS 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.

13.15.2.20 IDM_INTERRUPT_MASK

This register is the interrupt mask of Secure transactions.

Usage constraints

Accessible using Secure transactions only.

Configurations

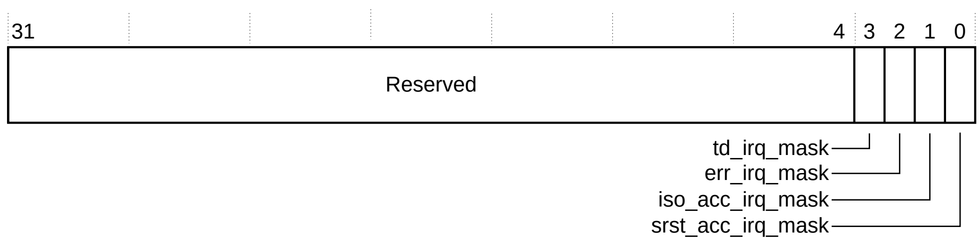
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-184: IDM_INTERRUPT_MASK bit assignments



The following table shows the bit descriptions.

Table 13-199: 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
[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.

13.15.2.21 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

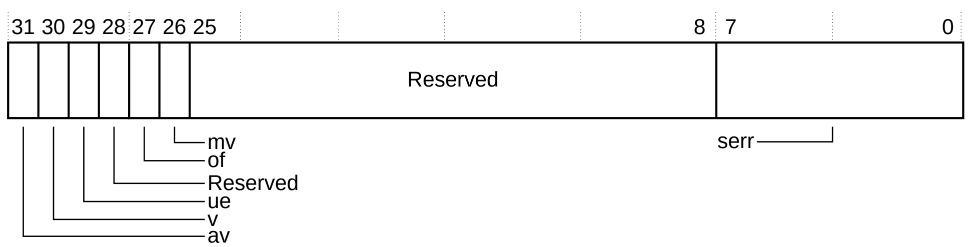
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-185: IDM_ERRSTATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-200: 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 the ue field of the IDM_ERRSTATUS_NS register 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_NS is not valid. **1** IDM_ERRSTATUS_NS is valid. At least one error has been recorded. This bit ignores writes if the ue field of the IDM_ERRSTATUS_NS register is set to 1 and is not being cleared to 0 in the same write. This bit is read, or write 1 to clear.

Bits	Name	Description
[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 the oe field of the IDM_ERRSTATUS_NS register is set to 1 and is not being cleared to 0 in the same write. This bit is not valid and reads UNKNOWN if the v field of the IDM_ERRSTATUS_NS register 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_NS and IDM_ERRMISC1_NS are not valid. **1** The IMPLEMENTATION DEFINED contents of the IDM_ IDM_ERRMISC0_NS and IDM_ERRMISC1_NS registers contains additional information for an error that this record captures. This bit ignores writes if the ue field of the IDM_ERRSTATUS_NS register is set to 1, and is not being cleared to 0 in the same write. This bit is 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 completer **20** Internal timeout

13.15.2.22 IDM_ERRADDR_LSB_NS

This register is the error log of Non-secure transactions.

Usage constraints

None.

Configurations

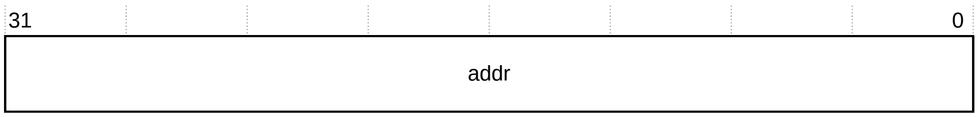
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-186: IDM_ERRADDR_LSB_NS bit assignments



The following table shows the bit descriptions.

Table 13-201: IDM_ERRADDR_LSB_NS descriptions

Bits	Name	Description
[31:0]	addr	Returns bits [31:0] of an address causing an error

13.15.2.23 IDM_ERRADDR_MSB_NS

This register is the error log of Non-secure transactions.

Usage constraints

None.

Configurations

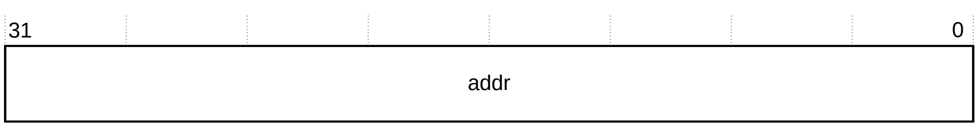
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-187: IDM_ERRADDR_MSB_NS bit assignments



The following table shows the bit descriptions.

Table 13-202: IDM_ERRADDR_MSB_NS bit descriptions

Bits	Name	Description
[31:0]	addr	Returns bits [63:32] of an address causing an error

13.15.2.24 IDM_ERRMISC0_NS

This register is the error log of Non-secure transactions.

Usage constraints

None.

Configurations

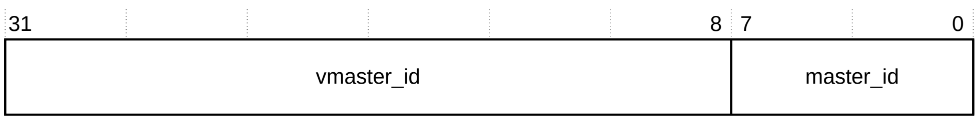
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-188: IDM_ERRMISC0_NS bit assignments



The following table shows the bit descriptions.

Table 13-203: 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.

13.15.2.25 IDM_ERRMISC1_NS

This register is the error log of Non-secure transactions.

Usage constraints

None.

Configurations

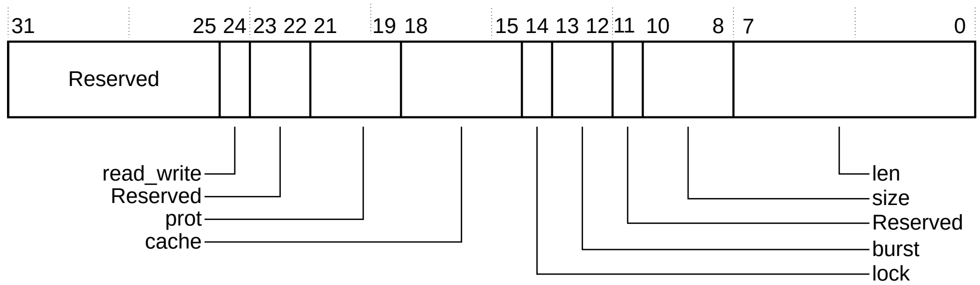
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-189: IDM_ERRMISC1_NS bit assignments



The following table shows the bit descriptions.

Table 13-204: 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.

13.15.2.26 IDM_ACCESS_STATUS_NS

This register indicates the access status for Non-secure transactions.

Usage constraints

None.

Configurations

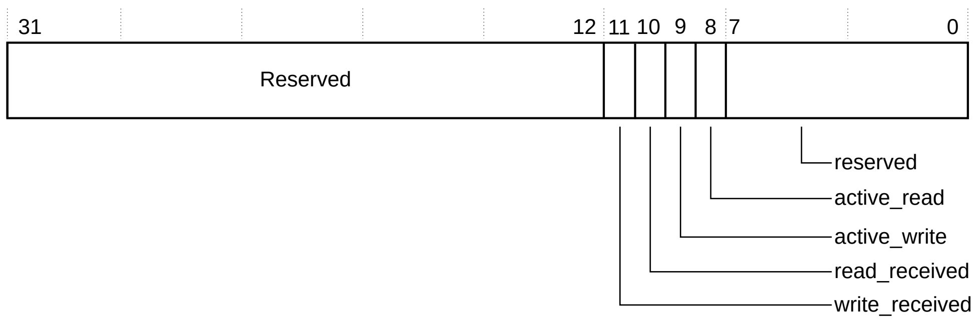
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-190: IDM_ACCESS_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-205: 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

13.15.2.27 IDM_ACCESS_READID_NS

This register is the access log of Non-secure transactions.

Usage constraints

None.

Configurations

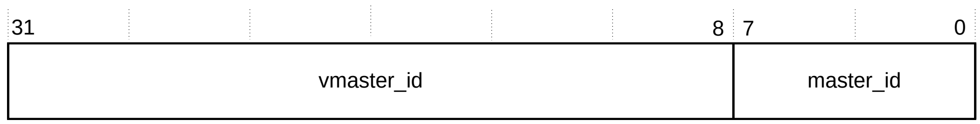
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-191: IDM_ACCESS_READID_NS bit assignments



The following table shows the bit descriptions.

Table 13-206: IDM_ACCESS_READID_NS bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming signal into the endpoint of the first transaction to arrive after isolation when the active_read field of the IDM_ACCESS_STATUS_NS register is HIGH. This field depends on the incoming endpoint. Therefore vmaster_id contains the ARID of the transaction on ASNI and contains the HMASTER on HSNI. For AMNI, PMNI, and HMNI the vmaster_id matches the ID of the originating ARID or HMASTER transaction. There is no manipulation of the incoming AXI ARID signal in ASNI.
[7:0]	master_id	The originating Node ID of the ASNI or HSNI of the first transaction to arrive after isolation when the active_read field of the IDM_ACCESS_STATUS_NS register is HIGH.

13.15.2.28 IDM_ACCESS_WRITEID_NS

This register is the access log of Non-secure transactions.

Usage constraints

None.

Configurations

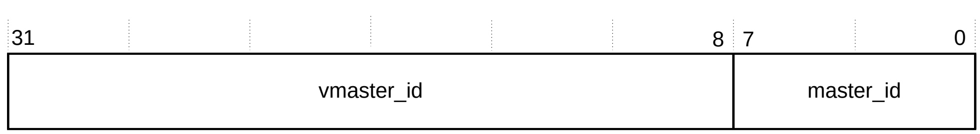
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-192: IDM_ACCESS_WRITEID_NS bit assignments



The following table shows the bit descriptions.

Table 13-207: IDM_ACCESS_WRITEID_NS bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming signal into the endpoint of the first transaction to arrive after isolation when the IDM_ACCESS_STATUS_NS register field active_write is HIGH. This field depends on the incoming endpoint. Therefore vmaster_id contains the AWID of the transaction on ASNI and contains the HMASTER on HSNI. For AMNI, PMNI, and HMNI the vmaster_id matches the ID of the originating AWID or HMASTER transaction. There is no manipulation of the incoming AXI AWID signal in ASNI.
[7:0]	master_id	The originating Node ID of the ASNI or HSNI of the first transaction to arrive after isolation when the active_write field of the IDM_ACCESS_STATUS_NS register is HIGH.

13.15.2.29 IDM_RESET_STATUS_NS

This register indicates the reset status of Non-secure transactions.

Usage constraints

None.

Configurations

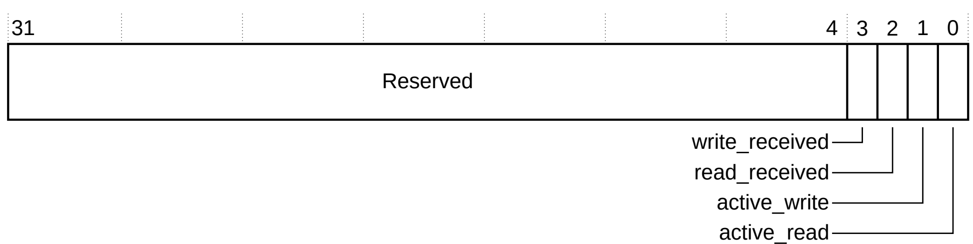
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-193: IDM_RESET_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-208: IDM_RESET_STATUS_NS descriptions

Bits	Name	Description
[31:4]	-	Reserved, **UNDEFINED**, write as zero

Bits	Name	Description
[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 to bit[0] of the 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 to 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 the IDM_RESET_CONTROL register. This bit is cleared to 0 on: • Reentry to soft reset state. Write 1 to bit[0] of the 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 to 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]	active_read	Active read transactions. A 1 indicates that there is at least one read transaction currently in progress.

13.15.2.30 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 Network Interface IDM registers summary.

The following table shows the bit descriptions.

Table 13-209: IDM_RESET_READID_NS bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming signal into the endpoint of the first transaction to arrive after isolation when the active_read field of the IDM_RESET_STATUS_NS register is HIGH. This field depends on the incoming endpoint. Therefore vmaster_id contains the ARID of the transaction on ASNI and contains the HMASTER on HSNI. For AMNI, PMNI, and HMNI the vmaster_id matches the ID of the originating ARID or HMASTER transaction. There is no manipulation of the incoming AXI ARID signal in ASNI.
[7:0]	master_id	The originating Node ID of the ASNI or HSNI of the first transaction to arrive after isolation when the active_read field of the IDM_RESET_STATUS_NS register is HIGH.

13.15.2.31 IDM_RESET_WRITEID_NS

This register is the reset access log of Non-secure transactions.

Usage constraints

None.

Configurations

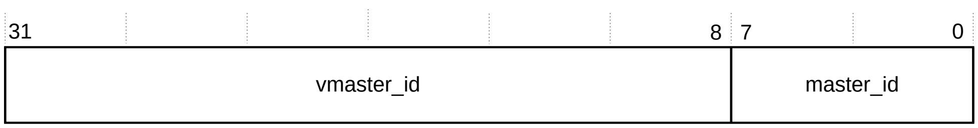
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit descriptions.

Figure 13-194: IDM_RESET_WRITEID_NS bit assignments



The following table shows the bit descriptions.

Table 13-210: IDM_RESET_WRITEID_NS bit descriptions

Bits	Name	Description
[31:8]	vmaster_id	The incoming signal into the endpoint of the first transaction to arrive after isolation when the active_write field of the IDM_RESET_STATUS_NS register is HIGH. This field depends on the incoming endpoint. Therefore vmaster_id contains the AWID of the transaction on ASNI and contains the HMASTER on HSNI. For AMNI, PMNI, and HMNI the vmaster_id matches the ID of the originating AWID or HMASTER transaction. There is no manipulation of the incoming AXI AWID signal in ASNI.
[7:0]	master_id	The originating Node ID of the ASNI or HSNI of the first transaction to arrive after isolation when active_write field of the IDM_RESET_STATUS_NS register is HIGH.

13.15.2.32 IDM_INTERRUPT_STATUS_NS

This register indicates the interrupt status of Non-secure transactions.

Usage constraints

None.

Configurations

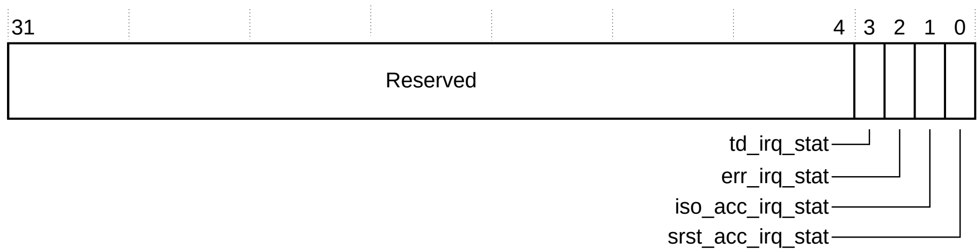
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-195: IDM_INTERRUPT_STATUS_NS bit assignments



The following table shows the bit descriptions.

Table 13-211: 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.

13.15.2.33 IDM_INTERRUPT_MASK_NS

This register is the interrupt mask of Non-secure transactions.

Usage constraints

None.

Configurations

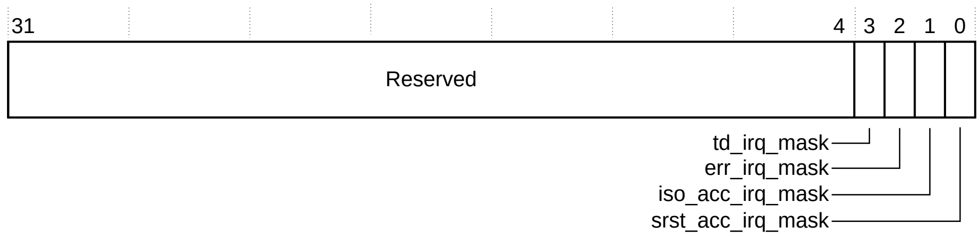
If IDM is enabled, this register is implemented in NI-700.

Attributes

For more information, see Network Interface IDM registers summary.

The following figure shows the bit assignments.

Figure 13-196: IDM_INTERRUPT_MASK_NS bit assignments



The following table shows the bit descriptions.

Table 13-212: 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.

13.16 PMNI registers

This section describes the NI-700 PMNI requester registers. It contains a summary of the requester interface registers, in order of address offset, and a description of the bitfields for each register.

13.16.1 PMNI registers summary

This register summary lists the NI-700 APB registers and some key characteristics.

The following table shows the requester 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 13-213: 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
0x04	PMNI_NODE_INFO	RO	0x0000	32	PMNI_NODE_INFO, Node information for PMNI register
0x08	PMNI_SECR_ACC	RW	0x00	32	PMNI_SECR_ACC, Secure access register
0x00C	PMNI_PMUSELA	RW	0x0000	32	PMNI_PMUSELA, Configure PMNI crossbar register
0x010	PMNI_PMUSELB	RW	0x0000	32	PMNI_PMUSELB, Configure PMNI crossbar register
0x014	PMNI_INTERFACEID_0:3	RO	Configuration dependent	32	PMNI_INTERFACEID, Configure APB interface IDs 0-3
0x018	PMNI_INTERFACEID_4:7	RO		32	PMNI_INTERFACEID, Configure APB interface IDs 4-7
0x01C	PMNI_INTERFACEID_8:11	RO		32	PMNI_INTERFACEID, Configure APB interface IDs 8-11
0x020	PMNI_INTERFACEID_12:15	RO		32	PMNI_INTERFACEID, Configure APB interface IDs 12-15
0x030	PMNI_SECURE_INFO	RO		32	PMNI_SECURE_INFO, Security attribute of downstream APB interfaces register
0x040	PMNI_NODE_FEAT	RO	Configuration dependent	32	PMNI_NODE_FEAT, Node features register
0x044	PMNI_CTRL	RW		Configuration dependent	PMNI_CTRL, PMNI control register
0x080	PMNI_SILDBG	RW/RO	0x00	32	PMNI_SILDBG, PMNI silicon debug monitor register

13.16.2 Register descriptions

Each register description provides information about the register, such as usage constraints, configurations, attributes, and bit assignments.

13.16.2.1 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

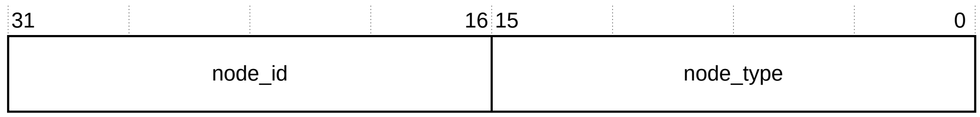
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-197: PMNI_NODE_TYPE bit assignments



The following table shows the bit descriptions.

Table 13-214: 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.

13.16.2.2 PMNI_NODE_INFO, Node information for PMNI register

This register provides node information for PMNI, such as data width.

Usage constraints

None.

Configurations

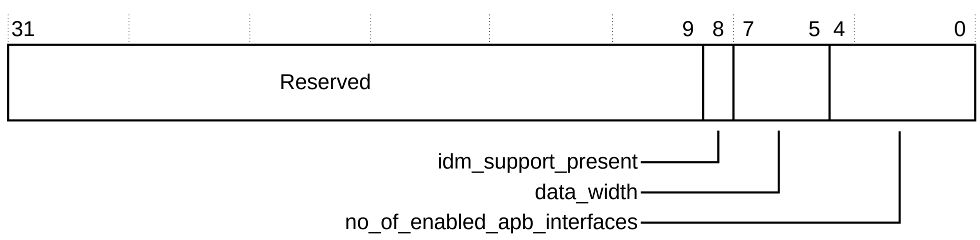
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-198: PMNI_NODE_INFO bit assignments



The following table shows the bit descriptions.

Table 13-215: PMNI_NODE_INFO bit descriptions

Bits	Name	Description
[31:9]	-	Reserved

Bits	Name	Description
[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.

13.16.2.3 PMNI_SECR_ACC, Secure access register

This register controls Secure access.

Usage constraints

Accessible using Secure transactions only.

Configurations

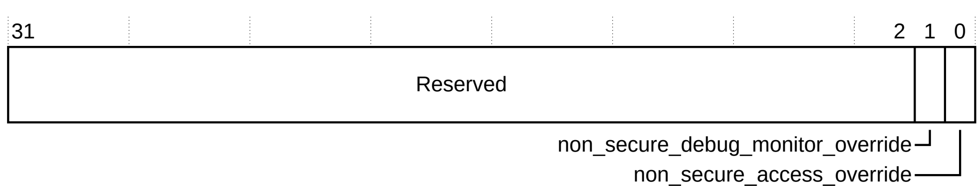
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-199: PMNI_SECR_ACC bit assignments



The following table shows the bit descriptions.

Table 13-216: PMNI_SECR_ACC bit descriptions

Bits	Name	Description
[31:2]	-	Reserved

Bits	Name	Description
[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.

13.16.2.4 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 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

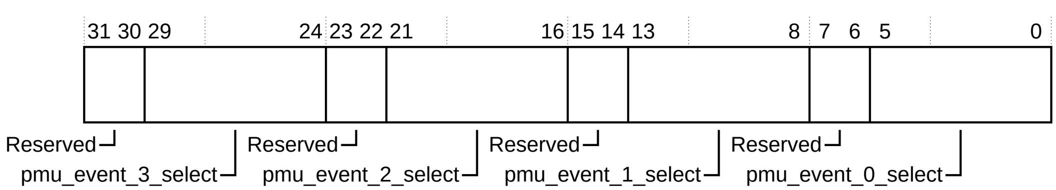
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-200: PMNI_PMUSELA bit assignments



The following table shows the bit descriptions.

Table 13-217: 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

Bits	Name	Description
[7:6]	-	Reserved
[5:0]	pmu_event_0_select	PMU event 0 select

13.16.2.5 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 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

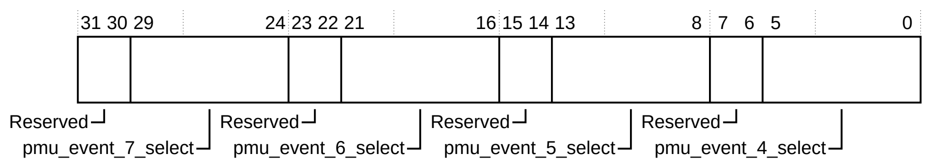
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-201: PMNI_PMUSELB bit assignments



The following table shows the bit assignments.

Table 13-218: 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

13.16.2.6 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

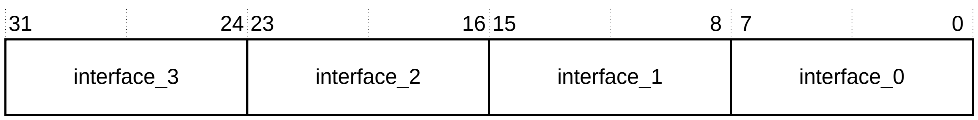
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-202: PMNI_INTERFACEID bit assignments, APB interface IDs 0-3



The following table shows the bit descriptions.

Table 13-219: 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

13.16.2.7 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

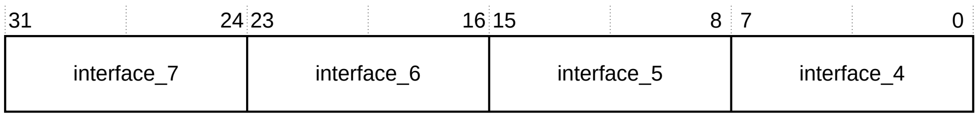
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-203: PMNI_INTERFACEID bit assignments, APB interface IDs 4-7



The following table shows the bit descriptions.

Table 13-220: 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

13.16.2.8 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

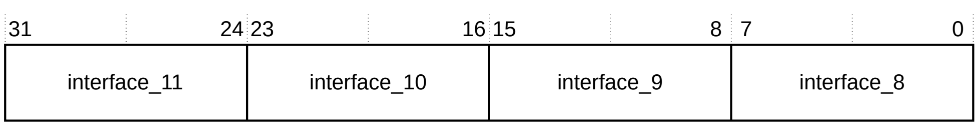
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-204: PMNI_INTERFACEID bit assignments, APB interface IDs 8-11



The following table shows the bit descriptions.

Table 13-221: 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

Bits	Name	Description
[15:8]	interface_9	APB interface ID 9
[7:0]	interface_8	APB interface ID 8

13.16.2.9 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

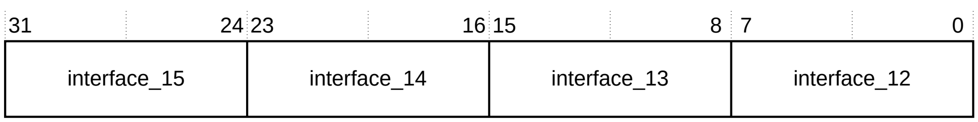
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-205: PMNI_INTERFACEID bit assignments, APB interface IDs 12-15



The following table shows the bit descriptions.

Table 13-222: 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

13.16.2.10 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

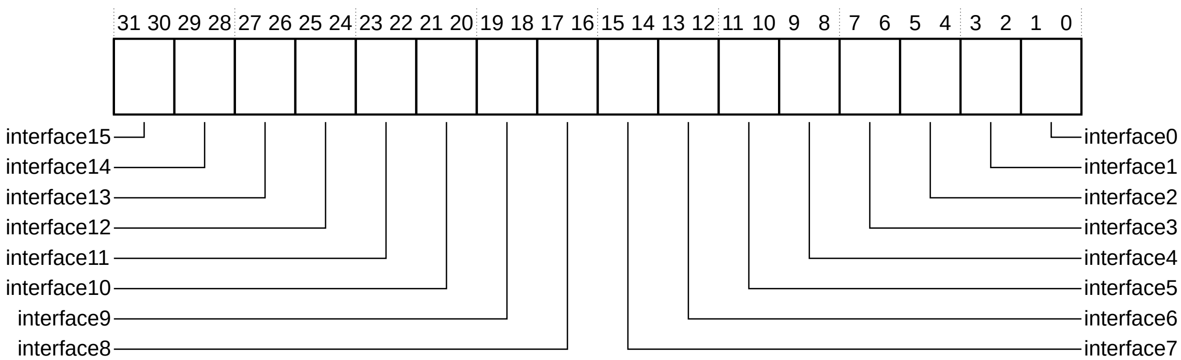
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-206: PMNI_SECURE_INFO



The following table shows the bit descriptions.

Table 13-223: 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
[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

Bits	Name	Description
[1:0]	interface_0	Security attribute for interface 0 `0b00` Software-programmable register to set the security attribute for the downstream completer. `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 completer attached to this APB requester interface. `0b03` Always Non-secure. Both Secure and Non-secure transactions access the completer attached to this APB requester interface.

13.16.2.11 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

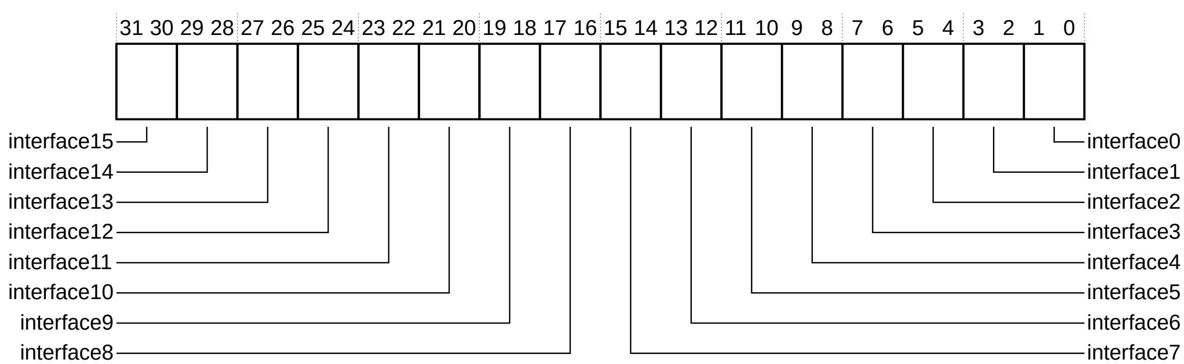
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-207: PMNI_NODE_FEAT bit assignments



The following table shows the bit descriptions.

Table 13-224: 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

Bits	Name	Description
[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

13.16.2.12 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 to permit Non-secure accesses.

Configurations

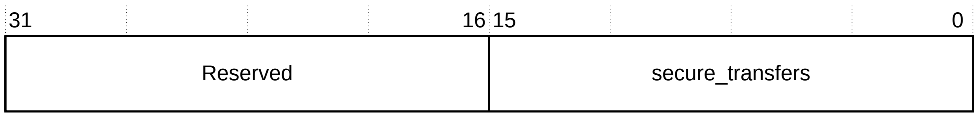
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-208: PMNI_CTL bit assignments



The following table shows the bit descriptions.

Table 13-225: 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 completer. **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. `0b00` If secure_transfers is 00, the PPROT pin is unavailable. This register bit determines the security attribute of the downstream completer. 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 completer 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.

13.16.2.13 PMNI_SILDBG, PMNI silicon debug monitor register

This register monitors the status of NI-700 requester interface channels.

Usage constraints

Accessible using only Secure accesses, unless you set the PMNI_SECR_ACC, Secure access register 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

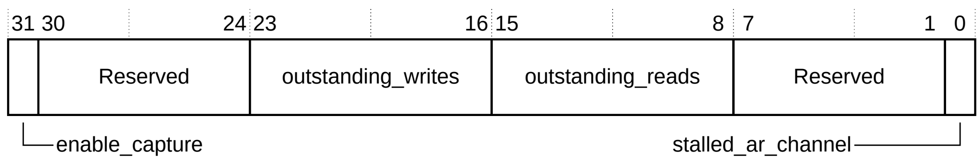
Available in all NI-700 configurations.

Attributes

For more information, see PMNI registers summary.

The following figure shows the bit assignments.

Figure 13-209: PMNI_SILDBG bit assignments



The following table shows the bit descriptions.

Table 13-226: 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	Indicates that the interface has reads that are outstanding.
[7:1]	-	Reserved
[0]	stalled_ar_channel	Indicates stalled read request

Appendix A Signal descriptions

NI-700 components provide a number of external signals. The signal descriptions describe each signal and include information on the signal name and signal direction.

Note

Unless specified otherwise, signals are active-HIGH.

A.1 ASNI external interface types and associated signal groups

You can configure an ASNI to have either an AXI5 or ACE5-Lite external completer interface. The ACE5-Lite interface has an extra set of signal groups compared to the AXI5 interface.

AXI5 external interface signal groups

If your ASNI has an AXI5 interface, see the following sections to find the details of the AXI signals:

- ASNI AXI4 write address channel signals
- ASNI AXI5 extension write address channel signals
- ASNI AXI4 write data channel signals
- ASNI AXI5 extension write data channel signals
- ASNI AXI4 write response channel signals
- ASNI AXI5 extension write response channel signals
- ASNI AXI4 read address channel signals
- ASNI AXI5 extension read address channel signals
- ASNI AXI4 read data channel signals
- ASNI AXI5 extension read data channel signals
- Other ASNI signals

ACE5-Lite external interface signal groups

If your ASNI has an ACE5-Lite interface, see the following sections to find the details of the AXI and ACE-Lite signals:

- ASNI AXI4 write address channel signals
- ASNI AXI5 extension write address channel signals
- ASNI ACE-Lite write address channel signals
- ASNI ACE5-Lite extension write address channel signals

A.1.1 ASNI AXI4 write address channel signals

All ASNI interface configurations contain a set of AXI4 write address channel signals. These signals transport AXI4 write address information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-1: ASNI AXI4 write address channel signals

Signal	Direction	Description
<prefix>_AWID[n:0]	Input	Write address ID. Width is configurable.
<prefix>_AWADDR[n:0]	Input	Write address. Width is configurable from 32 bits to 64 bits.
<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 (QoS) value
<prefix>_AWUSER[n:0]	Input	User-specified extension to AW payload
<prefix>_AWVALID	Input	Write address valid
<prefix>_AWREADY	Output	Write address ready
<prefix>_AWNSAID[3:0]	Input	NSAID signal associated with write address channel

A.1.2 ASNI AXI5 extension write address channel signals

All ASNI interface configurations contain a set of AXI5 extensions to the write address channel signals. These signals transport AXI5 write address information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-2: ASNI AXI5 extension write address channel signals

Signal	Direction	Description
<prefix>_AWATOP	Input	AW atomic operation. Indicates the type and endianness of atomic transactions.
<prefix>_AWTRACE	Input	Trace signals that are associated with the AW write address channel. AXI5 and ACE-Lite only.
<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>_AWTAGOP	Input	Write request tag operation. Encoded as: **00** Invalid **01** Transfer **10** Update **11** Match

A.1.3 ASNI ACE-Lite write address channel signals

ACE5-Lite ASNI interface configurations contain a set of ACE-Lite write address channel signals. These signals transport ACE-Lite write address information between an upstream ACE-Lite device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-3: ASNI ACE-Lite write address channel signals

Signal	Direction	Description
<prefix>_AWSNOOP[3:0]	Input	Transaction type for shareable write transactions
<prefix>_AWDOMAIN[1:0]	Input	Indicates the Shareability domain of a write transaction

A.1.4 ASNI ACE5-Lite extension write address channel signals

ACE5-Lite ASNI interface configurations contain a set of ACE5-Lite extensions to the write address channel signals. These signals transport ACE5-Lite write address information between an upstream ACE-Lite device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-4: ASNI ACE5-Lite extension write address channel signals

Signal	Direction	Description
<prefix>_AWSTASHNID	Input	Indicates the node identifier of the physical interface. This interface is the target interface for the cache stashing 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 stashing operation.
<prefix>_AWSTASHLPIDEN	Input	When asserted, this signal indicates the AWSTASHLPID signal is enabled and must be used.
<prefix>_AWCMO	Input	Indicates the type of CMO.

A.1.5 ASNI AXI4 write data channel signals

All ASNI interface configurations contain a set of AXI4 write data channel signals. These signals transport AXI4 write data information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-5: ASNI AXI4 write data channel signals

Signal	Direction	Description
<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

A.1.6 ASNI AXI5 extension write data channel signals

All ASNI interface configurations contain a set of AXI5 extensions to the write data channel signals. These signals transport AXI5 write data information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-6: ASNI AXI5 extension write data channel signals

Signal	Direction	Description
<prefix>_WTRACE	Input	Trace signals that are associated with the W write data channel
<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.1.7 ASNI AXI4 write response channel signals

All ASNI interface configurations contain a set of AXI4 write response channel signals. These signals transport AXI4 write response information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-7: ASNI AXI4 write response channel signals

Signal	Direction	Description
<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 write response payload
<prefix>_BVALID	Output	Write response valid
<prefix>_BREADY	Input	Write response ready

A.1.8 ASNI AXI5 extension write response channel signals

All ASNI interface configurations contain a set of AXI5 extensions to the write response channel signals. These signals transport AXI5 write response information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-8: ASNI AXI5 extension write response channel signals

Signal	Direction	Description
<prefix>_BTRACE	Output	Trace signals that are associated with the write response channel
<prefix>_BLOOP	Output	LOOP signal associated with the write response channel
<prefix>_BIDUNQ	Output	Write response channel unique ID indicator, active-HIGH
<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
<prefix>_BCOMP	Output	Indicates that the write is observable

A.1.9 ASNI ACE5-Lite extension write response channel signals

ACE5-Lite ASNI interface configurations contain a set of ACE5-Lite extensions to the write response channel signals. These signals transport ACE5-Lite write response information between an upstream ACE5-Lite device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-9: ASNI ACE5-Lite extension write response channel signals

Signal	Direction	Description
<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.

A.1.10 ASNI AXI4 read address channel signals

All ASNI interface configurations contain a set of AXI4 read address channel signals. These signals transport AXI4 read address information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-10: ASNI AXI4 read address channel signals

Signal	Direction	Description
<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	QoS 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>_ARREADY	Output	Indicates that a transfer on the read address channel can be accepted
<prefix>_ARNSAID	Input	NSAID associated with the read address channel

A.1.11 ASNI AXI5 extension read address channel signals

All ASNI interface configurations contain a set of AXI5 extensions to the read address channel signals. These signals transport AXI5 read address information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-11: ASNI AXI5 extension read address channel signals

Signal	Direction	Description
<prefix>_ARTRACE	Input	Trace signal that is associated with the AR read address channel
<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.

Signal	Direction	Description
<prefix>_ARTAGOP	Input	Read request tag operation. Encoded as: **0b00** Invalid **0b01** Transfer **0b10** Reserved **0b11** Fetch

A.1.12 ASNI ACE-Lite read address channel signals

ACE5-Lite ASNI interface configurations contain a set of ACE-Lite read address channel signals. These signals transport ACE-Lite read address information between an upstream ACE5-Lite device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-12: ASNI ACE-Lite read address channel signals

Signal	Direction	Description
<prefix>_ARSNOOP[3:0]	Input	Transaction type for shareable read transactions
<prefix>_ARDOMAIN[1:0]	Input	Shareability domain of a read transaction

A.1.13 ASNI AXI4 read data channel signals

All ASNI interface configurations contain a set of AXI4 read data channel signals. These signals transport AXI read data information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-13: ASNI AXI4 read data channel signals

Signal	Direction	Description
<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 read data payload
<prefix>_RVALID	Output	Read data valid
<prefix>_RREADY	Input	Read data ready

A.1.14 ASNI AXI5 extension read data channel signals

All ASNI interface configurations contain a set of AXI5 extensions to the read data channel signals. These signals transport AXI5 read data information between an upstream AXI device and the downstream ASNI.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-14: ASNI AXI5 extension read data channel signals

Signal	Direction	Description
<prefix>_RTRACE	Output	Trace signal that is associated with the read data channel
<prefix>_RLOOP	Output	LOOP signal associated with the 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.1.15 Other ASNI signals

ASNI configurations have a set of interface signals that are not related to a specific AXI channel.

In this section, <prefix> represents <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-15: Other ASNI signals

Signal	Direction	Description
<prefix>_QOSOVERRIDE	Input	Sample at reset QoS override. For more information, see *QoS value override programmable registers* in the NI-700 Technical Reference Manual.
<prefix>_ORDERED_WRITE_OBSERVATION	Input	Enables OWO on this completer interface if asserted. Refer to the OWO feature in the AMBA® AXI and ACE Protocol Specification. For more information, see *Transaction tracking and ordering* in the NI-700 Technical Reference Manual.

A.2 AMNI external interface types and associated signal groups

You can configure an AMNI to have an AXI5, AXI3, ACE5-Lite, or ACE5-LiteACP external requester interface. Each interface type has a different set of AXI signal groups.

AXI5 external interface signal groups

If your AMNI has an AXI5 interface, see the following sections to find the details of the AXI signals:

- AMNI AXI4 write address channel signals
- AMNI AXI5 extension write address channel signals
- AMNI AXI4 write data channel signals
- AMNI AXI5 extension write data channel signals
- AMNI AXI4 write response channel signals
- AMNI AXI5 extension write response channel signals
- AMNI AXI4 read address channel signals
- AMNI AXI5 extension read address channel signals
- AMNI AXI4 read data channel signals
- AMNI AXI5 extension read data channel signals

AXI3 external interface signal groups

If your AMNI has an AXI3 interface, some of the signals that are also present in the AXI5 configuration have different widths. For information about these changes, see AMNI AXI3 interface configuration signal changes. See the following sections to find the details of the other AXI signals:

- AMNI AXI4 write address channel signals
- AMNI AXI4 write data channel signals
- AMNI AXI4 write response channel signals
- AMNI AXI4 read address channel signals
- AMNI AXI4 read data channel signals

ACE5-Lite and ACE5-LiteACP external interface signal groups

If your AMNI has an ACE5-Lite or ACE5-LiteACP interface, see the following sections to find the details of the AXI and ACE-Lite signals:

- AMNI AXI4 write address channel signals
- AMNI AXI5 extension write address channel signals
- AMNI ACE-Lite write address channel signals
- AMNI ACE5-Lite extension write address channel signals
- AMNI AXI4 write data channel signals

- AMNI AXI5 extension write data channel signals
- AMNI AXI4 write response channel signals
- AMNI AXI5 extension write response channel signals
- AMNI AXI4 read address channel signals
- AMNI AXI5 extension read address channel signals
- AMNI ACE-Lite read address channel signals
- AMNI AXI4 read data channel signals
- AMNI AXI5 extension read data channel signals

A.2.1 AMNI AXI4 write address channel signals

All AMNI interface configurations contain a set of AXI4 write address channel signals. These signals transport AXI4 write address information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-16: AMNI AXI4 write address channel signals

Signal	Direction	Description
<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 QoS value
<prefix>_AWUSER[n:0]	Output	User-specified extension to write address payload
<prefix>_AWVALID	Output	Write address valid
<prefix>_AWREADY	Input	Write address ready
<prefix>_AWNSAID[3:0]	Output	NSAID signal that is associated to the write address channel

A.2.2 AMNI AXI5 extension write address channel signals

All AMNI interface configurations except AXI3 configurations contain a set of AXI5 extensions to the write address channel signals. These signals transport AXI5 write address information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-17: AMNI AXI5 extension write address channel signals

Signal	Direction	Description
<prefix>_AWATOP	Output	AW atomic operation. Indicates the type and endianness of atomic transactions.
<prefix>_AWTRACE	Output	Trace signals that are associated with the write address channel
<prefix>_AWLOOP	Output	LOOP signal that is associated with the write address channel
<prefix>_AWMPAM	Output	Write address channel MPAM information
<prefix>_AWIDUNQ	Output	Write address channel unique ID indicator, active-HIGH
<prefix>_AWTAGOP	Output	Write request tag operation. Encoded as: **00** Invalid **01** Transfer **10** Update **11** Match

A.2.3 AMNI ACE-Lite write address channel signals

ACE5-Lite and ACE5-LiteACP AMNI interface configurations contain a set of ACE-Lite write address channel signals. These signals transport ACE-Lite write address information between the upstream AMNI and the downstream ACE-Lite device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-18: AMNI ACE-Lite write address channel signals

Signal	Direction	Description
<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

A.2.4 AMNI ACE5-Lite extension write address channel signals

ACE5-Lite and ACE5-LiteACP AMNI interface configurations contain a set of ACE5-Lite extensions to the write address channel signals. These signals transport ACE5-Lite write address information between the upstream AMNI and the downstream ACE5-Lite device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-19: AMNI ACE5-Lite extension write address channel signals

Signal	Direction	Description
<prefix>_AWSTASHNID	Output	Indicates the node identifier of the physical interface that is the target interface for the cache stashing 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 stashing operation
<prefix>_AWSTASHLPIDEN	Output	When asserted, this signal indicates that the AWSTASHLPID signal is enabled and must be used.
<prefix>_AWCMO	Output	Indicates the type of CMO

A.2.5 AMNI AXI4 write data channel signals

All AMNI interface configurations contain a set of AXI4 write data channel signals. These signals transport AXI write data information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-20: AMNI AXI4 write data channel signals

Signal	Direction	Description
<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>_WDATA[n:0]	Output	Write data
<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 write data payload
<prefix>_WVALID	Output	Write data valid
<prefix>_WREADY	Input	Write data ready

A.2.6 AMNI AXI5 extension write data channel signals

All AMNI interface configurations except AXI3 configurations contain a set of AXI5 extensions to the write data channel signals. These signals transport AXI5 write data information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-21: AMNI AXI5 extension write data channel signals

Signal	Direction	Description
<prefix>_WTRACE	Output	Trace signals associated with the write data channel
<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.2.7 AMNI AXI4 write response channel signals

All AMNI interface configurations contain a set of AXI4 write response channel signals. These signals transport AXI write data information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-22: AMNI AXI4 write response channel signals

Signal	Direction	Description
<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 write response payload
<prefix>_BVALID	Input	Write response valid
<prefix>_BREADY	Output	Write response ready

A.2.8 AMNI AXI5 extension write response channel signals

All AMNI interface configurations except AXI3 configurations contain a set of AXI5 extensions to the write response channel signals. These signals transport AXI5 write data information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-23: AMNI AXI5 extension write response channel signals

Signal	Direction	Description
<prefix>_BTRACE	Input	Trace signal that is associated with the write response channel
<prefix>_BLOOP	Input	LOOP signal that is associated with the write response channel
<prefix>_BIDUNQ	Input	Write response channel unique ID indicator, active-HIGH
<prefix>_BCOMP	Input	Indicates that the write is observable
<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.2.9 AMNI AXI4 read address channel signals

All AMNI interface configurations contain a set of AXI4 read address channel signals. These signals transport AXI read address information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-24: AMNI AXI4 read address channel signals

Signal	Direction	Description
<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.

Signal	Direction	Description
<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	QoS 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>_ARREADY	Input	Indicates that a transfer on the read address channel can be accepted
<prefix>_ARNSAID[3:0]	Input	NSAID associated with the read address channel

A.2.10 AMNI AXI5 extension read address channel signals

All AMNI interface configurations except AXI3 configurations contain a set of AXI5 extensions to the read address channel signals. These signals transport AXI5 read address information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-25: AMNI AXI5 extension read address channel signals

Signal	Direction	Description
<prefix>_ARTRACE	Output	Trace signal that is associated with the read address channel
<prefix>_ARLOOP	Output	The LOOP signal that is associated with the 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.2.11 AMNI ACE-Lite read address channel signals

ACE5-Lite and ACE5-LiteACP AMNI interface configurations contain a set of ACE-Lite read address channel signals. These signals transport ACE-Lite read address information between the upstream AMNI and the downstream ACE5-Lite device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-26: AMNI ACE-Lite read address channel signals

Signal	Direction	Description
<prefix>_ARSNOOP[3:0]	Output	Transaction type for shareable read transactions
<prefix>_ARDOMAIN[1:0]	Output	Shareability domain of a read transaction

A.2.12 AMNI AXI4 read data channel signals

All AMNI interface configurations contain a set of AXI4 read data channel signals. These signals transport AXI read data information between the upstream AMNI and the downstream AXI device.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-27: AMNI AXI4 read data channel signals

Signal	Direction	Description
<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 read data payload
<prefix>_RVALID	Input	Read data valid
<prefix>_RREADY	Output	Read data ready

A.2.13 AMNI AXI5 extension read data channel signals

All AMNI interface configurations except AXI3 configurations contain a set of AXI5 extensions to the read data channel signals. These signals transport AXI5 read data information between the upstream AMNI and the downstream AXI device.

The following table shows the AMNI AXI5 extension read data channel signals. In this table, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-28: AMNI AXI5 extension read data channel signals

Signal	Direction	Description
<prefix>_RTRACE	Input	Trace signal that is associated with the read data channel
<prefix>_RLOOP	Input	LOOP signal associated with the read data channel
<prefix>_RIDUNQ	Input	Read data channel unique ID indicator, active-HIGH
<prefix>_RCHUNKV	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.2.14 AMNI AXI3 interface configuration signal changes

Configuring the external interface type of the AMNI to AXI3 changes the width of some of the AXI signals. This configuration affects the read address, write address, and write data channels.

In this section, <prefix> represents <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>.

For comparison with the AXI4 interface configuration, see the following sections:

- AMNI AXI4 read address channel signals
- AMNI AXI4 write address channel signals
- AMNI AXI4 write data channel signals

Signal definitions

Table A-29: AMNI AXI3 interface configuration signal changes

Signal	Direction	Description
<prefix>_ARLEN[3:0]	Output	Length. The exact number of data transfers in a read transaction.
<prefix>_ARLOCK[1:0]	Output	Information about the atomic characteristics of a read transaction
<prefix>_AWLEN[3:0]	Output	Write burst length
<prefix>_AWLOCK[1:0]	Output	Write lock type
<prefix>_WID[n:0]	Output	WID pin

A.3 HSNI external interface types and associated signal groups

You can configure a HSNI to have either an AHB5 or AHB5 mirrored requester interface. The AHB5 interface has an extra group of signals compared to the AHB5 mirrored requester interface.

AHB5 external interface signal groups

If your HSNI has an AHB5 interface, see the following sections to find the details of the AHB signals:

- HSNI AHB-Lite request signals
- HSNI AHB5 extension request signals
- HSNI AHB-Lite response signals
- HSNI AHB5 extension response signals
- Other HSNI AHB signals

AHB5 mirrored requester external interface signal groups

If your HSNI has an AHB5 mirrored requester interface, see the following sections to find the details of the AHB signals:

- HSNI AHB-Lite request signals
- HSNI AHB5 extension request signals
- HSNI AHB-Lite response signals
- HSNI AHB5 extension response signals

A.3.1 HSNI AHB-Lite request signals

All HSNI interface configurations have a set of AHB-Lite request signals. These signals transport AHB-Lite request information between an upstream AHB requester device and the downstream HSNI.

In this section, <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-30: HSNI AHB-Lite request signals

Signal	Direction	Description
<prefix>_HADDR	Input	AHB address bus
<prefix>_HBURST	Input	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

Signal	Direction	Description
<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.3.2 HSNI AHB5 extension request signals

All HSNI interface configurations have a set of AHB5 extensions to the request signals. These signals transport AHB5 request information between the upstream AHB requester device and the downstream HSNI.

In this section, <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-31: HSNI AHB5 extension request signals

Signal	Direction	Description
<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 requester identifier which is only used for exclusive transfer

A.3.3 HSNI AHB-Lite response signals

All HSNI interface configurations have a set of AHB-Lite response signals. These signals transport AHB-Lite response information between the upstream AHB requester device and the downstream HSNI.

In this section, <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-32: HSNI AHB-Lite response signals

Signal	Direction	Description
<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.3.4 HSNI AHB5 extension response signals

All HSNI interface configurations have a set of AHB5 extensions to the response signals. These signals transport AHB5 response information between the upstream AHB requester device and the downstream HSNI.

In this section, <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-33: HSNI AHB5 extension response signals

Signal	Direction	Description
<prefix>_HEXOKAY	Output	Exclusive Okay response

A.3.5 Other HSNI AHB signals

If you configure a HSNI to have a full AHB interface, instead of a requester mirror interface, the interface has an extra set of signals. These signals transport control information between the upstream AHB requester device and the downstream HSNI.

In this section, <prefix> represents AHB_SLAVE_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-34: Other HSNI AHB signals

Signal	Direction	Description
<prefix>_HREADY	Input	The HREADY from the multiplexor going to all requesters and completers
<prefix>_HSEL	Input	The completer select signal from the decoder

A.4 HMNI external interface types and associated signal groups

You can configure a HMNI to have either an AHB5 or AHB5 mirrored completer requester interface. The AHB5 interface has an extra group of signals compared to the AHB5 mirrored completer interface.

AHB5 external interface signal groups

If your HMNI has an AHB5 interface, see the following sections to find the details of the AHB signals:

- HMNI AHB-Lite request signals
- HMNI AHB5 extension request signals
- HMNI AHB-Lite response signals
- HMNI AHB5 extension response signals

- Other HMNI AHB signals

AHB5 mirrored completer external interface signal groups

If your HMNI has an AHB5 mirrored completer interface, see the following sections to find the details of the AHB signals:

- HMNI AHB-Lite request signals
- HMNI AHB5 extension request signals
- HMNI AHB-Lite response signals
- HMNI AHB5 extension response signals

A.4.1 HMNI AHB-Lite request signals

All HMNI configurations have a set of AHB-Lite request signals. These signals transmit AHB-Lite request information between the upstream HMNI and the downstream AHB completer.

In this section, <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-35: HMNI AHB-Lite request signals

Signal	Direction	Description
<prefix>_HADDR	Output	AHB address bus
<prefix>_HBURST	Output	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
<prefix>_HWDATA	Output	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.4.2 HMNI AHB5 extension request signals

All HMNI configurations have a set of AHB5 extensions to the request signals. These signals transmit AHB5 request information between the upstream HMNI and the downstream AHB completer.

In this section, <prefix> represents AHB_Manager_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-36: HMNI AHB5 extension request signals

Signal	Direction	Description
<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	Requester identifier which is only used for Exclusive transfer

A.4.3 HMNI AHB-Lite response signals

All HMNI configurations have a set of AHB-Lite response signals. These signals transmit AHB-Lite response information between the upstream HMNI and the downstream AHB completer.

In this section, <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-37: HMNI AHB-Lite response signals

Signal	Direction	Description
<prefix>_HRDATA	Input	The read data from the multiplexor
<prefix>_HREADY/ HREADYOUT	Input	If the interface is an AHB requester interface, this signal is the HREADY signal from the multiplexor. In AHB mirror mode, this signal is the HREADYOUT signal from the completer.
<prefix>_HRESP	Input	The transfer response from the multiplexor
<prefix>_HRUSER	Input	The read data channel User signal from the multiplexor

A.4.4 HMNI AHB5 extension response signals

All HMNI configurations have a set of AHB5 extensions to the response signals. These signals transmit AHB5 response information between the upstream HMNI and the downstream AHB completer.

In the section, <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-38: HMNI AHB5 extension response signals

Signal	Direction	Description
<prefix>_HEXOKAY	Input	Exclusive Okay response

A.4.5 Other HMNI AHB signals

If you configure a HMNI to have a full AHB interface, instead of a requester mirror interface, the interface has an extra set of signals. These signals transport control information between the upstream HMNI and the downstream AHB completer device.

In this section, <prefix> represents AHB_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-39: Other HMNI AHB signals

Signal	Direction	Description
<prefix>_HREADY	Output	The HREADY from the multiplexor, which goes to all requesters and completers
<prefix>_HSEL	Output	The completer select signal from the decoder

A.5 PMNI external interface types and associated signal groups

You can configure a PMNI to have either an APB3 or APB4 external requester interface. The APB4 interface has an extra group of signals compared to the APB3 interface.

APB3 external interface signal groups

If your PMNI has an APB3 interface, see the following sections to find the details of the APB signals:

- PMNI APB signals
- PMNI APB3 signals

APB4 external interface signal groups

If your PMNI has an APB4 interface, see the following sections to find the details of the APB signals:

- PMNI APB signals
- PMNI APB3 signals
- PMNI APB4 signals

A.5.1 PMNI APB signals

You can configure a PMNI to have an APB3 or APB4 completer interface. These APB signals that are always present in the PMNI regardless of the interface configuration.

In this section, <prefix> represents APB_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-40: PMNI APB signals

Signal	Direction	Description
<prefix>_PADDR_{0-15}	Output	APB address bus
<prefix>_PSEL_{0-15}	Output	APB completer device select. PMNI supports up to 16 APB completers.
<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: **0** APB read access **1** APB write access
<prefix>_PWDATA_{0-15}	Output	Write data
<prefix>_PRDATA_{0-15}	Input	APB read data

A.5.2 PMNI APB3 signals

You can configure a PMNI to have an APB3, or APB4 completer interface. These APB signals that are always present in the PMNI regardless of the interface configuration.

In this section, <prefix> represents APB_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-41: PMNI APB3 signals

Signal	Direction	Description
<prefix>_PREADY_{0-15}	Input	Ready. The APB completer 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 the PMNI is tied LOW.

A.5.3 PMNI APB4 signals

You can configure a PMNI to have an APB3 or APB4 completer interface. APB4 introduced new signals that were not present in APB2 or APB3, so these signals are only present in PMNIs with APB4 interfaces.

In this section, <prefix> represents APB_MASTER_<ENDPOINT_INTERFACE_NAME>.

Signal definitions

Table A-42: PMNI APB4 signals

Signal	Direction	Description
<prefix>_PPROT_{0-15}	Output	Protection type **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.6 Power, clock, reset, IDM, and other control signals

You can configure additional protocol control signals as well as NI-700 power domain, clock domain, reset, IDM, and other control signals.

Signal definitions

Table A-43: Power, clock, reset, IDM, and other control signals

Signal	Direction	Description
<AXI>_MASTER_<ENDPOINT_INTERFACE_NAME>_AWAKEUP	Output	Indicates that the requester 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 completer 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	External IDM soft reset.
<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_IDM_SRESET_STRAP	Input	Sample-at-reset input pin at every upstream and downstream 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	Sample-at-reset input pin at every upstream and downstream 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.

Signal	Direction	Description
<ENDPOINT_INTERFACE_NAME>_CONFIG_ACCESS	Input	Sample-at-reset input pin per downstream interface. This signal indicates the downstream interfaces that are permitted to accept new transactions in the CONFIG power state.
<ECOREVNUM>	Input	To track any Engineering Change Order (ECO) fixes in NI-700, you can change part of the Peripheral_ID3 register using the <ECOREVNUM> input pin. The <ECOREVNUM>[3:0] input corresponds to bits[7:4] of the Peripheral_ID3 register. You must tie this input LOW unless you have an ECO from Arm.
<CLKNAME>_CLK	Input	The clock input for that clock domain.
<CLKNAME>_RESETn	Input	Reset signal that is associated with the clock domain. Active-LOW.
<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	The 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 NI-700.
<PDOMAIN>_INTERRUPT	Output	Secure interrupt per power domain that is used to indicate specific conditions (IDM or non-IDM) within upstream or downstream interface. See the *Programmers model* chapter of the NI-700 Technical Reference Manual for the conditions.
<PDOMAIN>_NS_INTERRUPT	Output	Non-secure interrupt per power domain that is used to indicate specific conditions (IDM or non-IDM) within upstream or downstream interface. See the *Programmers model* chapter of the NI-700 Technical Reference Manual for the conditions.

A.7 Design for Test signals

NI-700 contains Design for Test (DFT) signals to disable internal resets and clocks, as well as enable architectural clock gates for CLKNAME clocks.

Signal definitions

Table A-44: Design for Test signals

Signal	Direction	Description
DFTRSTDISABLE[1:0]	Input	Internal resets are disabled.
DFTCGEN	Input	Enables 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.8 PMU and debug signals

In NI-700 each clock domain can count, export, and report performance monitoring events. Each clock domain can report either Secure or Non-secure events.

In the following section, <CLKNAME> represents the name of the clock domain.

Signal definitions

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 level-sensitive interrupt to indicate a counter, event, or cycle has overflowed.

Appendix B Revisions

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

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.	CoreLink NI-700 Network-on-Chip Interconnect
Added Configurable options section.	Configurable options
Updated top-level architecture diagram and associated note (describing top-level PCDC configuration).	Architecture overview
Added information about the configurable options that apply to all functional units.	Functional units
Added description of burst splitting scenarios and low and high-wire modes for ASNI and AMNI units.	ASNI AMNI
Added information about combining Q-Channel LPIs at the top level.	PCDC
Updated ASNI, AMNI, PCDC, and Router configuration options.	ASNI configuration options AMNI configuration options PCDC configuration options Router configuration options
Merged functional description of the NI-700 resets with functional description of power and clock management.	Power, clock, and reset management
Added section describing the NI-700 clock gating hierarchy.	Levels of clock gating
Added External clock controller section.	External Clock Controller
Added Power control section.	Power control
Added Clock and reset control section.	Clock and reset control
Added NodeID mapping and discovery section and moved descriptions of configuration register regions and access to this section.	Discovery
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.	Node configuration register address-mapping overview
Added Security section.	Secure and Non-secure accesses
Updated description of remap configuration and constraints.	Remap
Updated description of security attribute mismatch handling.	TrustZone technology and security
Added Interconnect Device Management section.	Interconnect Device Management
Added footnote to Peripheral ID4 register reset value to indicate that it is partially device dependent.	Global registers summary

Change	Location
Added voltage domain, power domain, and clock domain Secure Access Registers summaries and descriptions.	Voltage domain registers summary Power domain registers summary Clock domain registers summary
Updated description of global, ASNI, and AMNI Secure Access Registers.	SECR_ACC, Secure access register ASNI_SECR_ACC, Secure access register AMNI_SECR_ACC, Secure access register
Updated requester interface registers summary table.	ASNI registers summary
Added ASNI_IDM_DEVICE_ID and ASNI_IDM_RESET_CONTROL register summary and description.	ASNI registers summary
Updated ASNI Address Remap Vector Register description.	ASNI_ADDR_REMAP, Address remap vector register
Updated Usage constraints of various registers.	ASNI_SILDBG, ASNI silicon debug monitor register AMNI_SILDBG, Silicon debug monitor register
Updated requester interface registers summary table.	AMNI registers summary
Added AMNI_IDM_DEVICE_ID and AMNI_IDM_RESET_CONTROL registers summary and description.	AMNI registers summary
Updated Performance Monitoring Unit registers summary table.	PMU registers summary
Updated PMEVTYPERn, PMSSCR, and PMCFGR Register descriptions.	PMEVTYPERn, Performance monitor event type and filter registers PMSSCR, Performance monitors snapshot capture register PMCFGR, Performance monitors configuration register
Added Performance optimization and monitoring chapter.	Performance monitoring
Updated upstream interface signal tables.	Signal descriptions
Updated downstream interface signal tables.	Signal descriptions
Updated power and clock control signal tables.	Power, clock, reset, IDM, and other control signals
Added PMU and debug signal descriptions.	PMU and debug signals

Table B-3: Differences between issue 0000-01 and issue 0000-02

Change	Location
Updated ASNI configuration options.	ASNI configuration options
Updated AMNI configuration options.	AMNI configuration options
Added description of minimum latency for HSNI requests.	Architecture overview
Added NIs to more than one clock domain.	Configuration register address region calculation
Updated IDM description.	Interconnect Device Management

Change	Location
Updated QoS features.	Quality of Service
Updated programmers model.	Programmers model
Updated signal descriptions.	Signal descriptions

Table B-4: Differences between issue 0000-02 and issue 0000-03

Change	Location
Added information on error handling and interrupts.	Error handling and interrupts
Added information on network interface IDM registers.	Network Interface IDM registers
Added configuration information to functional units.	Functional units
Extended security information.	Secure and Non-secure accesses

Table B-5: Differences between issue 0000-03 and issue 0000-04

Change	Location
Updated protocol information about the NI-700 Interconnect.	Compliance
Updated protocol information about the NI-700 Interconnect.	Architecture overview
Added information on interface configuration options.	Configurable options
Updated the HSNI configuration options.	HSNI configuration options
Updated the HMNI configuration options.	HMNI configuration options
Added configuration data on Pipeline slices to the ASNI functional description.	ASNI
Added configuration data on Pipeline slices to the AMNI functional description.	AMNI
Added configuration data on pipeline slices to the HSNI functional description and scenarios for multi-copy atomicity.	HSNI
Added configuration data on pipeline slices to the HMNI functional description.	HMNI
Added configuration data on pipeline slices to the PMNI functional description.	PMNI
Updated the functional description of AHB address phase buffering in HSNI.	AHB address phase buffering in HSNIs
Added a section on external interfaces and their InterfaceID with an explanatory diagram.	Component and interface identifiers
Updated the APB security configuration options.	Security access permissions of APB requests
Added a note to Interconnect Device Management.	Interconnect Device Management
Added a section on the user signal widths.	User signals
Added a section on the ASNI address decoders.	ASNI address decode
Added a section on the HSNI address decoders.	HSNI address decode
Added a section on the PMNI address decoders.	PMNI address decode
Added information on the Address Hash Function in Address striping.	Address striping
Updated the functional description of AHB address phase buffering in HSNI.	AHB address phase buffering in HSNIs

Change	Location
Updated configuration information for all Secure and Non-secure IDM Power Domain register descriptions.	For Secure IDM_PD registers: IDM_PD_ERROR_STATUS to IDM_PD_ACCESS_CONTROL. For Non-secure IDM_PD registers: IDM_PD_ERROR_STATUS_NS to IDM_PD_ACCESS_CONTROL_NS
Added a section on PMNI_INTERFACEID to configure APB interfaces 0–3.	PMNI_INTERFACEID, Configure APB interface IDs 0-3
Added a section on PMNI_INTERFACEID to configure APB interfaces 4–7.	PMNI_INTERFACEID, Configure APB interface IDs 4-7
Added a section on PMNI_INTERFACEID to configure APB interfaces 8–11.	PMNI_INTERFACEID, Configure APB interface IDs 8-11
Added a section on PMNI_INTERFACEID to configure APB interfaces 12–15.	PMNI_INTERFACEID, Configure APB interface IDs 12-15
Added additional information on Secure Exempt for HSNI performance events `0x25`, `0x2A` and `0x2B`.	HSNI performance events
Updated the PMNI performance events (`0x03`, `0x0A`, `0x0B`, `0x20` and `0x20`).	PMNI performance events
Added additional information on Secure Exempt for HMNI performance events `0x22` and `0x23`.	HMNI performance events
Updated the APB security configuration option.	Security access permissions of APB requests
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
Updated the HSNI_CTRL register to include secure_ctrl field description, values and location in bit assignment diagram.	HSNI_CTRL, HSNI control register
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
Updated the HMNI_CTRL register to include secure_ctrl field description, values and location in bit assignment diagram.	HMNI_CTRL, HMNI control register
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
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
Updated cross reference.	HMNI_INTERRUPT_STATUS_NS, Interrupt status (Non-secure) register
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
Updated the PMNI_NODE_INFO register to include the no_of_enabled_apb_interfaces field.	PMNI_NODE_INFO, Node information for PMNI register
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
Updated the PMNI_CTRL register to include secure_transfers field, values, description and location within the bit assignment diagram.	PMNI_CTRL, PMNI control register
Updated the APB requster request signals to delete the Protection Types table.	PMNI APB signals
Updated the Power and clock control AWAKEUP signals to identify the protocol as AXI.	Power, clock, reset, IDM, and other control signals

Change	Location
Updated the DFTRSTDISABLE and DFTCGEN signal names in Design For Test (DFT) signals.	Design for Test signals

Table B-6: Differences between issue 0000-04 and issue 0000-05

Change	Location
Changed protocols to packets for AXI-H section.	Compliance
Updated the unsupported AMBA features.	Key features
Revised the amount of configurable devices.	Configurable options
Loopback_Signals option updated for ASNI.	ASNI configuration options
Loopback_Signals option updated for AMNI.	AMNI configuration options
Content on clock disable pin added.	Test features
Updated the HMNI_INTERRUPT_STATUS_NS register to New note added on shareable exclusive transactions.	HSNI
Updated supported protocols, removed APB2.	PMNI
Moved content on Address decode and mapping to Functional description section.	Address decode and mapping
Updated Address striping section.	Address striping
Repositioned Interconnect Device Management content within Functional description section.	Interconnect Device Management
New example case for upstream NI IDM block which fails to accept read data beat and write response.	Timeout detection through IDM block
Added new content on IDM error logging interrupts and status flags.	IDM error logging interrupts and status flags
Added new section on Error Handling and interrupt security.	Error handling and interrupt security
New section added on requester network Interface error responses.	Requester network interface error responses
Updated the AHB security access permissions.	Security access permissions of AHB requests
Added text and cross-referencing to register security attribute and security classification and Secure register access.	Security access permissions of AHB requests
Added new content and table to Quality of Service. Moved it to the Functional description section.	Quality of Service
New section added on AHB locked transfers.	AHB locked transfers
Added section on Exclusive and locked accesses.	Exclusive and locked accesses
Updated the User signals content.	User signals
Updated About the programmers model section.	About the programmers model
Updated the Reset value for the NODE_TYPE Global register.	Global registers summary
Updated the Reset value for the NODE_TYPE Voltage domain register.	Voltage domain registers
Updated the Reset value for the NODE_TYPE Power domain register.	Power domain registers
Updated the Reset value for the NODE_TYPE Clock domain register.	Clock domain registers
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
Performance monitor configuration register PMCFGR changed to RO in registers summary.	Performance Monitoring Unit registers summary
Performance monitor control register PMCR updated to reflect Write Only and RW bits.	PMCR, Performance monitors control register

Change	Location
Updated ASNI registers summary to remove repeated occurrence of ASNI_NODE_INFO and include ASNI Interface Ids 0:15.	ASNI registers summary
Added the mpam_input_present bit to the bit assignment diagram.	ASNI_NODE_INFO, Node information for ASNI register
Added section on ASNI Interface IDs 0–3.	ASNI_INTERFACEID, Configure ASNI interface IDs 0-3
Added section on ASNI Interface IDs 4–7.	ASNI_INTERFACEID, Configure ASNI interface IDs 4-7
Added section on ASNI Interface IDs 8–11.	ASNI_INTERFACEID, Configure ASNI interface IDs 8-11
Added section on ASNI Interface IDs 12–15.	ASNI_INTERFACEID, Configure ASNI interface IDs 12-15
Added 'Type' column to ASNI_BURSPLT bit assignment table.	ASNI_BURSPLT, Burst split control register
Added 'Type' column to ASNI_SILDBG bit assignment table.	ASNI_SILDBG, ASNI silicon debug monitor register
Updated the ASNI_ARQOSOVR, Read channel description and the arqos_value bit description.	ASNI_ARQOSOVR, Read channel QoS value override register
Updated the ASNI_AWQOSOVR, Write channel description and the awqos_value bit description.	ASNI_AWQOSOVR, Write channel QoS value override register
Updated AMNI registers summary to include AMNI Interface Ids 0:15.	AMNI registers summary
Added consent required to modify AMNI_QOSACC, QoS Accept Control for AMNI.	AMNI_QOSACC, QoS accept control
Updated AMNI registers summary to reflect AMNI_SILDBG register as RW/RO.	AMNI registers summary
Added section on AMNI Interface IDs 0–3.	AMNI_INTERFACEID, Configure AMNI interface IDs 0-3
Added section on AMNI Interface IDs 4–7.	AMNI_INTERFACEID, Configure AMNI interface IDs 4-7
Added section on AMNI Interface IDs 8–11.	AMNI_INTERFACEID, Configure AMNI interface IDs 8-11
Added section on AMNI Interface IDs 12–15.	AMNI_INTERFACEID, Configure AMNI interface IDs 12-15
Updated HSNI registers summary to include HSNI Interface Ids 0:15.	HSNI registers summary
Updated HSNI registers summary to reflect HSNNI_SILDBG registers as RW/RO.	HSNI registers summary
Added section on HSNI Interface IDs 0–3.	HSNI_INTERFACEID, Configure HSNI interface IDs 0-3
Added section on HSNI Interface IDs 4–7.	HSNI_INTERFACEID, Configure HSNI interface IDs 4-7
Added section on HSNI Interface IDs 8–11.	HSNI_INTERFACEID, Configure HSNI interface IDs 8-11
Added section on HSNI Interface IDs 12–15.	HSNI_INTERFACEID, Configure HSNI interface IDs 12-15
Updated HMNI registers summary to reflect HSNI_CTRL and HMNNI_SILDBG registers as RW/RO.	HMNI registers summary
Updated HMNI registers summary to include HMNI Interface Ids 0:15.	HMNI registers summary
Added section on HMNI Interface IDs 0–3.	HMNI_INTERFACEID, Configure HMNI interface IDs 0-3

Change	Location
Added section on HMNI Interface IDs 4–7.	HMNI_INTERFACEID, Configure HMNI interface IDs 4-7
Added section on HMNI Interface IDs 8–11.	HMNI_INTERFACEID, Configure HMNI interface IDs 8-11
Added section on HMNI Interface IDs 12–15.	HMNI_INTERFACEID, Configure HMNI interface IDs 12-15
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.	Network Interface IDM registers summary
Updated the bit descriptions for IDM_ERRCTL.	IDM_ERRCTLR
Updated IDM_ERRSTATUS and IDM_ERRSTATUS_NS to reflect Reserved fields as RO and SERR field as RO	IDM_ERRSTATUS and IDM_ERRSTATUS_NS
Notes added to isolate bit description.	IDM_ACCESS_CONTROL
Bit descriptions updated for IDM_ACCESS_STATUS	IDM_ACCESS_STATUS
Added new notes to the IDM_RESET_CONTROL descriptions.	IDM_RESET_CONTROL
Updated IDM_RESET_STATUS bit descriptions.	IDM_RESET_STATUS
Updated PMNI register summary to show PMNI_SILDBG register as RW/RO.	PMNI registers summary
Updated AMNI note on Secure Events and table heading, Secure exempt, replaced with Secure Only and relevant bookmarks added.	AMNI performance events
ASNI table heading, Secure exempt, replaced with Secure Only and relevant bookmarks added.	ASNI performance events
HSNI table heading, Secure exempt, replaced with Secure Only and relevant bookmarks added.	HSNI performance events
HMNI table heading, Secure exempt, replaced with Secure Only and relevant bookmarks added.	HMNI performance events
Table heading, Secure exempt, replaced with Secure Only.	PMNI performance events
Updated HSNI request signals.	HSNI external interface types and associated signal groups
Updated Power and clock control signals.	Power, clock, reset, IDM, and other control signals
Updated DFT<CLKNAME>CLKDISABLE description.	Design for Test signals

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.	Key features
Updated ASNI configuration options: Updated the maximum number of completer NIs for ASNIs and HSNIs and the maximum number of upstream 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.	ASNI configuration options
Updated the AMNI configuration options read and write issuing capability from 1–64 transactions to 1–256.	AMNI configuration options
Defined input signals for HMNI mirrored requester interface and HSNI.	HSNI
Removed information on the AMNI output signal AWAKEUP in clock domain wakeup content.	Clock domain wakeup
Added a note to explain what happens when IDM and Read Data Chunking features are enabled together.	Interconnect Device Management

Change	Location
Added new use case examples for upstream and downstream interface soft resets.	Soft reset use case examples for completer and requester network interfaces
Added new content on the write response buffer.	Transaction reorder buffers
Updated title to (Read) reorder buffer, added new content on read reorder buffer allocation and merging partial read responses.	Transaction reorder buffers
Added new content on AXI non-modifiable transactions.	Single completer for each ID
Added new section on Ordered Write Observation (OWO).	Ordered Write Observation
Added a new note on per transaction User bits to the user signals content.	User signals
Updated global register PERIPHERAL_ID2 product_version bit to identify EAC r1p0 and DEV r0p1 product versions.	PERIPHERAL_ID2
Updated ASNI registers summary to reflect changes in width to 10 for the registers ASNI_ATQOSOT, ASNI_ARQOSOT, ASNI_AWQOSOT, and ASNI_AXQOSOT.	ASNI registers summary
Updated the ASNI_SILDBG register bit assignments.	ASNI_SILDBG, ASNI silicon debug monitor register
Updated the max_atomic_ots bit assignment to [9:0] in the ASNI_ATQOSOT register.	ASNI_ATQOSOT, Maximum atomic Outstanding Transactions register
Updated the max_read_ots bit assignment to [9:0] in the ASNI_ARQOSOT register.	ASNI_ARQOSOT, Maximum read Outstanding Transactions register
Updated the max_write_ots bit assignment to [9:0] in the ASNI_AWQOSOT register.	ASNI_AWQOSOT, Maximum write Outstanding Transactions register
Updated the max_ar_aw_ots bit assignment to [9:0] in the ASNI_AXQOSOT register.	ASNI_AXQOSOT, Maximum combined Outstanding Transactions register
Updated the AMNI_SILDBG register bit assignments.	AMNI_SILDBG, Silicon debug monitor register
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
Updated descriptions for the AHB completer Network Interface performance events read request stall on `0x0E` and write request stall on `0x10`.	HSNI performance events
Updated descriptions for several HMNI 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`.	HMNI performance events
<PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_AWID[n:0] width is not configurable.	AMNI AXI4 write address channel signals

Change	Location
Added new signal name and description for <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_WID[n:0].	AMNI AXI4 write data channel signals
Added signal widths to read address channel requester interface signals.	AMNI ACE-Lite read address channel signals
Updated the <PDOMAIN>_INTERRUPT and <PDOMAIN>_NS_INTERRUPT interrupt signal descriptions to state these interrupts are rising edge triggered.	Power, clock, reset, IDM, and other control signals
Updated the signal directions and descriptions for <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_SRESETN and <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>_SRESETN.	Power, clock, reset, IDM, and other control signals
Updated the <CLKNAME>_nPMUINTERRUPT signal description to state this interrupt is rising edge triggered.	PMU and debug signals

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.	Interfaces
Removed content on limitations of the r0p1 DEV release.	
• Updated the supported number of upstream NIs (AMNIs, HMNIs, and PMNIs) to 127 and the supported number of completer NIs (ASNIs and HSNIs) changed to 128. • Added new content on cache line size	Configurable options
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.	ASNI configuration options
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	AMNI configuration options
Updated HSNI configuration option User sideband signal width of 0–64 bits removed and crossreference added to User signals.	HSNI configuration options
Updated HMNI configuration option User sideband signal width of 0–64 bits removed and crossreference added to User signals.	HMNI configuration options
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 HSNIs

<table>
<tr><th>Change</th><th>Location</th></tr>
<tr><td>Updated Access mechanism diagram to remove text reference to Non-coherent interconnect (NCI), and replace with NI-700.</td><td>Node configuration register address-mapping overview</td></tr>
<tr><td>Updated content:
Deleted bullet point: All stripe groups in a memory map that an AXI or AHB completer 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.</td><td>Address striping</td></tr>
<tr><td>Updated content:
Updated text to target interface within the text, updated all remap diagrams with the text target.
Added a new note to the end of the topic on maintaining access to the programmers model and Config target when remapping occurs.</td><td>Remap</td></tr>
<tr><td>Removed content on IDM unsupported AMBA 5 features and reworded existing content.</td><td>Interconnect Device Management</td></tr>
<tr><td>Added content when IDM detects a timeout, software must trigger a soft reset before resuming normal operation.</td><td>Timeout detection through IDM block</td></tr>
<tr><td>Added content on how NI-700 handles outstanding requests and soft reset requests.</td><td>IDM soft reset mode</td></tr>
<tr><td>Added content on how NI-700 handles an isolation request and the difference between isolation and soft reset.</td><td>IDM access control</td></tr>
<tr><td>Added use cases for the soft-reset functionality for upstream and downstream interfaces.</td><td>Soft reset use case examples for completer and requester network interfaces</td></tr>
<tr><td>Added a use case for the access control functionality for a write transaction at a downstream interface.</td><td>Access control use case example for requester and completer network interfaces</td></tr>
<tr><td>Added new content to demonstrate an interrupt handling sequence.</td><td>Example interrupt handling sequence</td></tr>
<tr><td>Added an example to show a fast sequence for placing a downstream device into soft-reset.</td><td>Soft reset sequence</td></tr>
<tr><td>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.</td><td>Requester network interface error responses</td></tr>
<tr><td>Added new content on memory tagging support and relevant behavior in the NI-700.</td><td>Memory tagging support</td></tr>
<tr><td>Added new content on how to calculate TSPEC parameters for traffic.</td><td>Calculating TSPEC parameters for traffic</td></tr>
<tr><td>Added examples on how to calculate TSPEC parameters for traffic.</td><td>TSPEC parameter examples</td></tr>
<tr><td>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.</td><td>TSPEC registers and parameters</td></tr>
<tr><td>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.</td><td>Soft bandwidth regulation</td></tr>
<tr><td>Updated references to the AXI specification from issue G to H and removed Issue F reference from <code>USER_DATA_MODE</code> = 0 in User signals.</td><td>User signals</td></tr>
</table>

Change	Location
Added new content on requirements for configuration register reads and writes.	Requirements of configuration register reads and writes
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
Updated secure_ctrl register bits from [1:0] to [0].	HMNI_CTRL, HMNI control register
Updated IDM register summary to include two new registers IDM_ACCESS_STATUS_NS and IDM_RESET_STATUS_NS.	Network Interface IDM registers summary
Added new register IDM_ACCESS_STATUS_NS.	IDM_ACCESS_STATUS_NS
Added new register IDM_RESET_STATUS_NS.	IDM_RESET_STATUS_NS
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 completer 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.	–
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>	ASNI AXI5 extension write data channel signals
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.	ASNI AXI5 extension write response channel signals
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.	ASNI AXI4 read address channel signals and ASNI AXI5 extension read address channel signals
Updated content: • Added a new signal name <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME>_RTAG. • Updated all signal names with a reference to <prefix>	ASNI AXI5 extension read data channel signals

Change	Location
Updated content: • Removed <PROTOCOL>_SLAVE_<ENDPOINT_INTERFACE_NAME> from the beginning of each AXI signal name. • Added cross reference to QoS value override programmable registers for the QOSOVERRIDE signal. • Added cross reference to Ordered Write Observation for the ORDERED_WRITE_OBSERVATION signal.	Other ASNI signals
Updated content: • Changed `AWSNOOP` value for requester 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 • Added new signal <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE _NAME>_AWTAGOP.	AMNI ACE-Lite write address channel signals
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>	AMNI AXI5 extension write data channel signals
Added a new signal name <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE _NAME>_BTAGMATCH and description to the AMNI AXI5 write response signals.	AMNI AXI5 extension write response channel signals
Updated signal direction for <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE_NAME>_ARREADY to an input in read address channel requester interface signals table.	AMNI AXI4 read address channel signals
Added new signal <PROTOCOL>_MASTER_<ENDPOINT_INTERFACE _NAME>_ARTAGOP to AMNI AXI5 extension read address channel signals.	AMNI AXI5 extension read address channel signals
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.	AMNI AXI5 extension read data channel signals
Removed <PROTOCOL>_<MASTER_>ENDPOINT_INTERFACE_NAME>_PWAKEUP> signal from power and clock control signals.	Power, clock, reset, IDM, and other control signals

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 requester interface connection to downstream completers.	Supported AMBA features

Change	Location
Updated content: • NI-700 supports AXI3 AMBA protocol, but only on NI-700 requester 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.	Key features
Removed AXI3 from the list of unsupported specifications.	Compliance
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 requester interfaces.	Architecture overview
Added new content: • An AMNI can have AXI3 as the requester 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.	AMNI configuration options
Added a new section on support for AXI3 interface types and updated content on write data dependency constraints.	ASNI
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*.	–
Updated references to diagram and power control network diagram.	Power control
Updated HSNI clock gating buffer block diagram to show <CLKNAME>_CLK and <CLKNAME>_RESETn signal direction.	AHB address phase buffering in HSNIs
Added support for address stripe granules, in bytes, 128, 256, 512, 1024, 2048, and 4096.	Address striping
Updated content to describe what happens once IDM detects a timeout and the mode the interface enters	Timeout detection through IDM block
Removed content on when the IDM block detects a bus error on its interface and the software uses the IDM soft reset functionality.	Error logging through IDM block
Added new sections: • Hardware initiated entry based on timeout detection. • Software initiated entry. • IDM_RESET_CONTROL reset initialization input pin.	IDM soft reset mode
Added new content on upstream NI isolation and how an upstream device handles requests and transactions.	IDM access control

Change	Location
Removed the note on hardware must receive a soft reset request to resume normal operation and updated all diagrams and relevant content.	Soft reset use case examples for completer and requester network interfaces
Removed duplicate text.	Example interrupt handling sequence
Added Write response dependency violation to the AMNI non-IDM interrupt conditions per endpoint.	Non-IDM interrupts
Added new content on Memory tagging support (MTE).	Memory tagging support
Added the AMNI_CONFIG_CTL register to the AMNI registers summary.	AMNI registers summary
Updated description for bits [3:0] amni_type, to reflect relevant technical specifications.	AMNI_NODE_INFO, Node information for AMNI register
Added AMNI_CONFIG_CTL register and updated its configuration constraints.	AMNI_CONFIG_CTL, Select response
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
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
Updated HSNI performance events `0x0E` to `0x12` to reflect address and data phases of AHB in the HSNI_SILDBG and HMNI_SILDBG registers.	HSNI performance events
Updated HMNI performance events `0x0E` to `0x12` to reflect address and data phases of AHB in the HSNI_SILDBG and HMNI_SILDBG registers.	HMNI performance events
Corrected signal name from NSAIDW[3:0] to AWNSAID[3:0].	ASNI AXI4 write address channel signals
Added ARNSAID signal name, input type and description.	ASNI AXI4 read address channel signals
Updated signal name from NSAIDW[3:0] to AWNSAID[3:0].	AMNI AXI4 write address channel signals
Added signal ARNSAID[3:0].	AMNI AXI4 read address channel signals
Added new AXI3 requester signals.	AMNI AXI3 interface configuration signal changes
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.	Power, clock, reset, IDM, and other control signals

Table B-10: Differences between issue 0200-08 and issue 0201-09

Change	Location
Added a new bullet to explain support for transporting data parity, ECC, or poison information through the interconnect.	Key features
Updated read reorder depth and permitted read reorder depth, added new note.	ASNI configuration options
Added cross reference to content on calculating output IDs.	AMNI configuration options
Updated text for early write response to state HMASTER width supports 1–16 outstanding writes and removed the ID width bullet point as this is the same as the HMASTER configurable option.	HSNI configuration options
Removed ID width as not configurable on the HMNI.	HMNI configuration options
Updated sections on HSNI and HMNI signals and added new figure on signals.	HSNI
Updated text on the minimum power state that the power domain requires to guarantee forward progress.	Power
Added text on the minimum power state that the power domain requires to guarantee forward progress.	Power state requirements and characteristics

Change	Location
Added relevant PMU registers and offsets to the example configurable design table.	Configuration address space example for design with multiple voltage, power, and clock domains
Updated the example cases where an upstream or downstream network interface IDM block indicates stalled or failed transactions from an upstream or downstream device.	Timeout detection through IDM block
Separated the main IDM soft reset topic into individual topics for better navigation.	IDM soft reset mode
Moved content on IDM soft reset, Hardware initiated entry based on timeout detection, into a new topic.	Hardware initiated entry based on timeout detection
Moved content on IDM soft reset, Software initiated entry, into a new topic.	Software initiated entry
Moved content on IDM soft reset, IDM_RESET_CONTROL reset initialization input pin, into a new topic.	IDM_RESET_CONTROL reset initialization input pin
Added a new topic on transporting data parity, ECC, and poison information.	Transporting data parity, ECC, and poison information
Updated the note in AHB locked transfers.	AHB locked transfers
Updated text on how read reorder reservation functions.	Transaction reorder buffers
Added more content on how to use Resource Planes (RPs).	Resource Planes
Added new topic on calculating output IDs.	Calculation of output IDs
Added new topic on ID reduction.	ID reduction
Updated bit description fields for customer_mod_number and eco_number.	PERIPHERAL_ID3
Updated the reset value for ASNI_BURSPLT.	ASNI registers summary
Updated field descriptions to identify which fields are read only and which are read/write.	ASNI_BURSPLT, Burst split control register
Updated the usage constraints for the AMNI_CONFIG_CTL register.	AMNI_CONFIG_CTL, Select response
Updated the usage constraints for the AMNI_INTERRUPT_STATUS, HSNI_INTERRUPT_STATUS, and HMNI_INTERRUPT_STATUS registers.	AMNI_INTERRUPT_STATUS, Interrupt status register HSNI_INTERRUPT_STATUS, Interrupt status register HMNI_INTERRUPT_STATUS, Interrupt status register
Updated the usage constraints for the AMNI_INTERRUPT_MASK, HSNI_INTERRUPT_MASK, and HMNI_INTERRUPT_MASK registers.	AMNI_INTERRUPT_MASK, Interrupt mask register HSNI_INTERRUPT_MASK, Interrupt mask register HMNI_INTERRUPT_MASK, Interrupt mask register
Updated the reset value for IDM_TIMEOUT_VALUE from `0x0` to `0x4`.	Network Interface IDM registers summary
Added the correct register field names.	IDM_ACCESS_STATUS
Updated the vmaster_id and master_id bit descriptions.	IDM_ACCESS_READID IDM_ACCESS_WRITEID
Added the description of bit 1 to the bit assignment table.	IDM_RESET_CONTROL
Updated the vmaster_id and master_id bit descriptions.	IDM_RESET_READID IDM_RESET_WRITEID

Change	Location
Added the maximum value for the IDM_TIMEOUT register which is 30.	IDM_TIMEOUT_VALUE
Updated references to non-secure registers only not secure registers.	IDM_ERRSTATUS_NS
Updated the vmaster_id and master_id bit descriptions.	IDM_ACCESS_READID_NS IDM_ACCESS_WRITEID_NS
Updated the bit description for bit[0] to active_read.	IDM_RESET_STATUS_NS
Updated the vmaster_id and master_id bit descriptions.	IDM_RESET_READID_NS IDM_RESET_WRITEID_NS
Updated event code descriptions for event codes: `0x02`, `0x03`, `0x04`, `0x09`, `0x0A`, and `0x0B`.	ASNI performance events AMNI performance events
Updated topic title and add new <ECOREVNUM> signal.	Power, clock, reset, IDM, and other control signals

Table B-11: Differences between issue 0201-09 and issue 0203-10

Change	Location
Updated content to reflect progressive language where appropriate	Throughout
Performed text edits and typos	Throughout
Updated packetizing mechanism that enables configurable link widths from 64-512 bits to 32-2048 bits	Architecture Overview
Updated read and write acceptance capability from 1-256 to 1-512 transactions	ASNI configuration options
Updated read and write issuing capability from 1-256 to 1-512 transactions and removed Ordered Write Observation (OWO) from AMNI configuration options	AMNI configuration options
Updated content to reflect 0 is for active-HIGH polarity and 1 is for active-LOW polarity	Power control sequences
Updated the User signal parameter `USER_REQ_WIDTH` supported range from 0-64 bits to 0-256 bits	User signals
Updated sections Requester network interface read transaction timeout leading to soft reset and Requester network interface write transaction timeout leading to soft reset. Text should read After the software writes 1 to the IDM_RESET_CONTROL.reset field, not IDM_RESET_CONTROL.auto field	Soft reset use case examples for completer and requester network interfaces
Updated text on optimizing the transaction size and length for write or read transactions	Upsizing AXI and ACE-Lite data width function
Updated the read data reorder buffer from 1-64 to 1-255 data beats	Transaction reorder buffers
Updated the PERIPHERAL_ID2 register to accomodate this release version	PERIPHERAL_ID2
Changed reference in topic introduction from Non-secure access to Secure access	Voltage domain secure access register
Updated the description of the node_id field of the IDM_PD_ACCESS_STATUS register to access interrupt instead of reset interrupt	IDM_PD_ACCESS_STATUS register
Changed reference in topic introduction from Non-secure access to Secure access	Power domain secure access register
Changed reference in topic introduction from Non-secure access to Secure access	Clock domain secure access register
Signal directions for ARLEN and ARLOCK changed from Input to Output	AMNI AXI3 interface configuration signal changes