Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded

Revision: r1p1

Technical Reference Manual

arm

101526_0101_03_en

Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded

Technical Reference Manual

Release Information

Document History

Issue	Date	Confidentiality	Change
0000-00	12 March 2019	Non-Confidential	First beta release for r0p0.
0000-01	06 June 2019	Non-Confidential	First limited access release for r0p0.
0100-02	15 August 2019	Non-Confidential	First early access release for r1p0.
0101-03	17 July 2020	Non-Confidential	First full release for r1p1.

Non-Confidential Proprietary Notice

(LES-PRE-20349)

Confidentiality Status

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

Unrestricted Access is an Arm internal classification.

Product Status

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

Web Address

www.arm.com

Contents
Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual

Preface

This preface introduces the *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Technical Reference Manual*.

It contains the following:

About this book

This book describes the functionality of the components in the Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded. It also provides the programming information and the signal descriptions.

Product revision status

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

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

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

Intended audience

This book is written for system designers and programmers who are designing or programming a *System on Chip* (SoC) that uses the SIE-300.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction
This chapter introduces the SIE-300 AXI5 System IP for Embedded.

Chapter 2 MSC functional description
This chapter describes the functionality of the *Master Security Controller* (MSC) component.

Chapter 3 MPC functional description
This chapter describes the functionality of the *Memory Protection Controller* (MPC) component.

Chapter 4 PPC functional description
This chapter describes the functionality of the *Peripheral Protection Controller* (PPC) component.

Chapter 5 SMC functional description
This chapter describes the functionality of the *SRAM Memory Controller* (SMC) component.

Chapter 6 ACG, SDB, and SUB functional description
This chapter describes the functionality of the three bridge components, that is, the Access Control Gate, Sync-Down Bridge, and Sync-Up Bridge.

Chapter 7 Programmers model
This chapter describes the memory regions and registers that the *Memory Protection Controller* (MPC) provides.

Appendix A Signal descriptions
This appendix describes the interface signals that are present for each SIE-300 component.

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

Glossary

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

See the *Arm® Glossary* for more information.

Typographic conventions

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

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

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

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

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

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

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

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

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

Timing diagrams

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

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

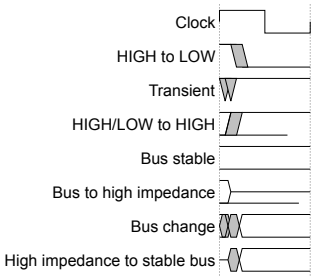

Figure 1 Key to timing diagram conventions

Signals

The signal conventions are:

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

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

Additional reading

This book contains information that is specific to this product. See the following documents for other relevant information.

Arm publications

- *AMBA® AXI and ACE Protocol Specification* (IHI 0022).
- *AMBA® APB Protocol Specification Version 2.0* (IHI 0024).
- *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* (IHI 0068).

The following confidential books are only available to licensees or require registration with Arm:

- *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Configuration and Integration Manual* (101527).

Other publications

- JEDEC, *Standard Manufacturer's Identification Code*, JEP106.

Feedback

Feedback on this product

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

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

Feedback on content

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

- The title *Arm CoreLink SIE-300 AXI5 System IP for Embedded Technical Reference Manual*.
- The number 101526_0101_03_en.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Note

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

Chapter 1
Introduction

This chapter introduces the SIE-300 AXI5 System IP for Embedded.

It contains the following sections:

1.1 About the AXI5 System IP for Embedded

The SIE-300 AXI5 System IP for Embedded provides a set of configurable AXI5 security-aware components. The components can protect peripherals and memories that are unaware of security, so that a peripheral or memory is only accessible to trusted software. The SIE-300 also provides clock synchronizing bridges and an access control gate.

The SIE-300 consists of the following components:

Master Security Controller (MSC)
The MSC acts as security gate for AXI transactions, and it can transform the security attribute.

Memory Protection Controller (MPC)
The MPC acts as security gate for AXI transactions that target a memory interface. The security checks operate on block or page level, and are programmable by using the APB slave interface.

Peripheral Protection Controller (PPC)
The PPC gates AXI5 transactions to, and responses from, peripherals when a security violation occurs.

Access Control Gate (ACG)
The ACG component can be placed on a clock or power domain boundary to pass or block AXI5 transactions whenever the downstream component cannot accept the transaction, or is explicitly asked not to do so. The transaction is latched internally and the ACG generates automatic responses when necessary.

Sync-Down Bridge (SDB)
The SDB synchronizes AXI5 interfaces where the upstream side is faster than the downstream side, and the clocks are synchronous, in phase, and have an N:1 frequency ratio.

Sync-Up Bridge (SUB)
The SUB synchronizes AXI5 interfaces where the upstream side is slower than the downstream side, and the clocks are synchronous, in phase, and have a 1:N frequency ratio.

SRAM Memory Controller (SMC)
The SMC enables on-chip synchronous RAM blocks to attach to an AXI5 interface. The SMC supports 32, 64, 128, or 256-bit SRAM with byte writes.

1.2 Compliance

The SIE-300 AXI5 System IP for Embedded is compliant with Arm specifications and protocols.

The SIE-300 components are compliant with the:

- AMBA 5 AXI protocol. See the *AMBA® AXI and ACE Protocol Specification*.
- AMBA 4 APB protocol. See the *AMBA® APB Protocol Specification Version 2.0*.
- AMBA Low Power Interface specification. See the *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces*.

This *Technical Reference Manual* (TRM) complements the architecture specifications and protocol specifications. The TRM does not duplicate information from these sources.

1.3 Configurable options

The SIE-300 provides design-time configuration options for each component. These options enable the design team to configure a component so that it is suitable for its location in an SoC.

Related concepts

2.2 MSC configuration options on page 2-19
3.2 MPC configuration options on page 3-25
4.2 PPC configuration options on page 4-31
5.2 SMC configuration options on page 5-39
6.2 Bridge configuration options on page 6-46

1.4 Product documentation

Documentation that is provided with this product includes a *Technical Reference Manual* (TRM) and a *Configuration and Integration Manual* (CIM), together with architecture and protocol information.

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

The SIE-300 documentation is as follows:

Technical Reference Manual

The TRM describes the functionality and the effects of functional options on the behavior of SIE-300. It is required at all stages of the design flow. The choices that are made in the design flow can mean that some behaviors that the TRM describes are not relevant. If you are programming SIE-300, contact:

- The implementer to determine:
  - The build configuration of the implementation.
  - What integration, if any, was performed before implementing SIE-300.
- The integrator to determine the signal configuration of the device that you use.

The TRM complements architecture and protocol specifications and relevant external standards. It does not duplicate information from these sources.

Configuration and Integration Manual

The CIM describes:

- The available build configuration options.
- How to configure the RTL with the build configuration options.
- How to integrate SIE-300 into an SoC.
- How to implement SIE-300 into your design.
- The processes to validate the configured design.

The Arm product deliverables include reference scripts and information about using them to implement your design.

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

1.5 Product revisions

This section describes the differences in functionality between product revisions.

r0p0 First release.

r0p0-r1p0 The functional changes are:

- Added the **cfg_ext_gt_err_resp** configuration signal. See *Configuration interface* on page 6-45.
- Changed the SMC arbitration scheme. See *Read and write transaction scheduling* on page 5-37.
- Changed the eviction of an SMC TAG table entry, when all EAMs are occupied. See *Exclusive accesses* on page 5-37.
- Changed when the SMC samples **cfg_gate_resp**. See *Table A-31 SMC external-gating configuration signal* on page Appx-A-94.

r1p0-r1p1 No functional changes.

Chapter 2
MSC functional description

This chapter describes the functionality of the *Master Security Controller* (MSC) component.

It contains the following sections:

2.1 About the MSC

The *Master Security Controller* (MSC) acts as security gate for AXI transactions, and it can transform the security attribute.

The MSC enables AXI masters that are designed for A-class systems to be inserted into M-class systems. Since A-class and M-class systems handle security differently, the MSC can transform the security attributes of a transaction to satisfy the M-class requirements.

The following figure shows the MSC interfaces.

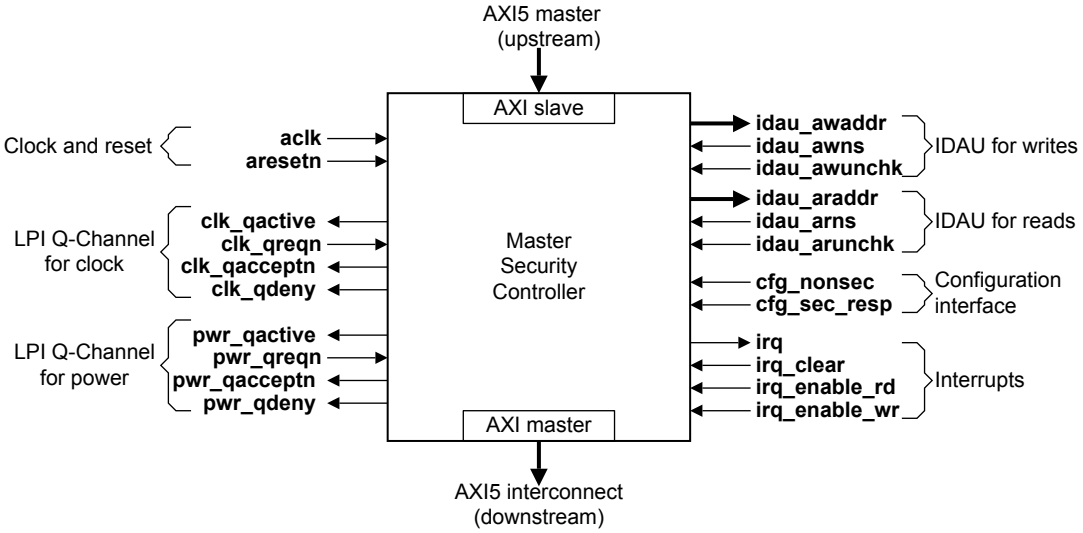

Figure 2-1 MSC interfaces

The AXI slave and AXI master interfaces provide the AXI data path from the AXI master to the interconnect.

To support low-power quiescence, the MSC has two Q-Channel interfaces. One Q-Channel is for clock quiescence and the other Q-Channel is for power quiescence.

Configuration interface

The **cfg_nonsec** input tells the MSC whether the AXI5 master, which connects to its slave interface, is in the Secure state or the Non-secure state. The MSC uses this information to control whether it blocks a transaction from going downstream.

When the MSC blocks a transaction, the **cfg_sec_resp** controls whether the MSC:

- Responds with an AXI slave error (SLVERR).
- Ignores a write transaction or returns zero for a read transaction.

IDAU interfaces

The MSC has two *Implementation Defined Attribution Unit* (IDAU) interfaces that it uses to discover the Security state of an addressed region. One IDAU is for read transactions and the other IDAU is for write transactions.

When the MSC receives an AXI transaction, it accesses the corresponding IDAU and retrieves the Security state for that transaction address. By using the Security state information, the incoming AXI access permissions (**AxPROT**), and the state of **cfg_nonsec**, the MSC can do one of the following:

- Block the transaction from going downstream.
- Forward the transaction.
- Transform the security attributes and then forward the transaction.

2.2 MSC configuration options

The MSC has design options that configure some AXI signal widths and the presence of synchronization logic on the Q-Channel inputs.

When implementing the MSC in an SoC, you can configure:

- The width of the AXI data bus, by using the `DATA_WIDTH` parameter. The possible widths are 32, 64, 128, or 256 bits.
- The width of the ID signals on the AXI interfaces, by using the `ID_WIDTH` parameter. The parameter can be set to a value from 2-32 inclusive.
- The width of the User signals on each AXI channel, by using the `ARUSER_WIDTH`, `AWUSER_WIDTH`, `BUSER_WIDTH`, `RUSER_WIDTH`, and `WUSER_WIDTH` parameters. A parameter can be set to a value from 0-256 inclusive.
- The presence of a synchronizer on the Q-Channel **QREQn** inputs. See *2.4 MSC Q-Channels on page 2-21*.

2.3 MSC interrupts

The MSC has a level-sensitive interrupt output, **irq**, that can indicate the occurrence of a security violation or a faulty security attribute conversion.

For read transactions, the **irq_enable_rd** signal controls whether the MSC can set **irq** HIGH when a security violation or a faulty security attribute conversion occurs during a read transaction.

For write transactions, the **irq_enable_wr** signal controls whether the MSC can set **irq** HIGH when a security violation or a faulty security attribute conversion occurs during a write transaction.

2.4 MSC Q-Channels

The MSC provides two Q-Channel device interfaces. You can use one channel for clock control and the other channel for power control, which enables the MSC to indicate when it requires clock and power.

Both Q-Channels implement the low-power interfaces that the *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* describes.

The presence of a synchronizer on the **clk_qreqn** and **pwr_qreqn** inputs of the Q-Channels is configurable, by using the `QCLK_SYNC_EN` and `QPWR_SYNC_EN` parameters.

2.5 MSC AMBA bus properties

The AMBA protocols define multiple property types that indicate the capabilities of a device.

The following table lists the AXI5 properties of the MSC.

Table 2-1 MSC AXI5 properties

AXI5 property	Value	Comment
Wakeup_Signals	TRUE	Q-Channel activity is generated from the **awakeup** input signal.
Check_Type	FALSE	-
Poison	TRUE	The component forwards the poison data downstream, it does not use the information that the poison bits contain.
Trace_Signals	FALSE	-
QoS_Accept	FALSE	-
Loopback_Signals	FALSE	-
Untranslated_Transactions	FALSE	-
NSAccess_Identifiers	FALSE	-
Atomic_Transactions	FALSE	-

Chapter 3
MPC functional description

This chapter describes the functionality of the *Memory Protection Controller* (MPC) component.

It contains the following sections:

3.1 About the MPC

The *Memory Protection Controller* (MPC) acts as security gate for AXI transactions that target a memory interface. The security checks operate on block or page level, and are programmable by using the APB slave interface.

The following figure shows the MPC interfaces.

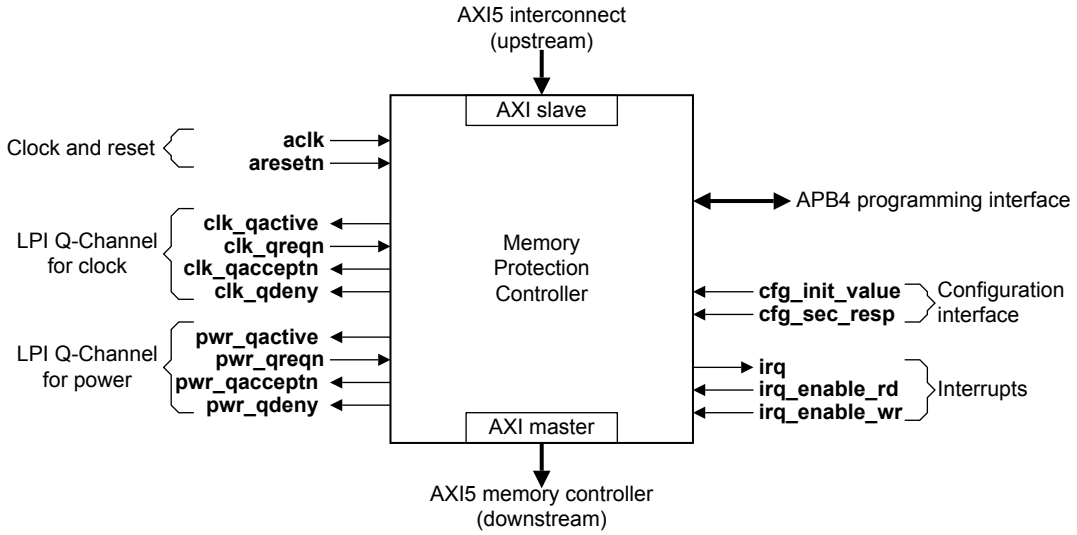

Figure 3-1 MPC interfaces

The AXI slave and AXI master interfaces provide the AXI data path from the interconnect to the memory controller.

To support low-power quiescence, the MPC has two Q-Channel interfaces. One Q-Channel is for clock quiescence and the other Q-Channel is for power quiescence.

Configuration interface

At powerup, the MPC uses the value of the **cfg_init_value** input as the initialization value for the *Look Up Table* (LUT) to be Secure or Non-secure for the entire memory range that the MPC protects.

If a security violation occurs, the MPC generates an interrupt and the **cfg_sec_resp** controls whether the MPC:

- Responds with an AXI slave error (SLVERR).
- Ignores a write transaction or returns zero for a read transaction.

Note

- When accessing all internal registers (except for the PID/CID registers) with a Non-secure APB transaction, the response is either an error or RAZ/WI depending on the value of the **cfg_sec_resp** input signal. See *Table A-8 MSC configuration signals* on page Appx-A-78 for more information.
- When accessing all internal registers (except for the PID/CID registers) with a Secure but unprivileged APB transaction, the response is always RAZ/WI, regardless of the value of the **cfg_sec_resp** input signal.

3.2 MPC configuration options

The MPC has design options that configure the memory block size, the widths of some AXI signals, and the presence of synchronization logic on the Q-Channel inputs. You can also configure the MPC to support the AXI transaction gating feature.

When implementing the MPC in an SoC, you can configure:

- The width of the AXI address bus, by using the `ADDR_WIDTH` parameter. The parameter can be set to a value from 12-32 inclusive.
- The width of the AXI data bus, by using the `DATA_WIDTH` parameter. The possible widths are 32, 64, 128, or 256 bits.
- The width of the ID signals on the AXI interfaces, by using the `ID_WIDTH` parameter. The parameter can be set to a value from 2-32 inclusive.
- The width of the User signals on each AXI channel, by using the `ARUSER_WIDTH`, `AWUSER_WIDTH`, `BUSER_WIDTH`, `RUSER_WIDTH`, and `WUSER_WIDTH` parameters. A parameter can be set to a value from 0-256 inclusive.
- The size of a memory block, by using the `BLK_SIZE` parameter. The block size can be 256 bytes to 1MB. See *7.3.3 Block LUT configuration status register, BLK_CFG* on page 7-58 for the possible values.
- The MPC to support the AXI transaction gating feature, by using the `GATE_PRESENT` parameter. See *7.4 Gating AXI transactions during register updates* on page 7-70 for more information.
- The presence of a synchronizer on the Q-Channel **QREQn** inputs. See *3.4 MPC Q-Channels* on page 3-27.

3.3 MPC interrupts

The MPC has a level-sensitive interrupt output, **irq**, that can indicate the occurrence of a security violation.

If a security violation occurs when **irq** is LOW, and the corresponding **irq_enable_*** signal is HIGH, the MPC saves information about the violation in the IRQ_STAT, IRQ_INFO1, and IRQ_INFO2 registers. Also, if the IRQ_EN register bit is set to 1 and the corresponding **irq_enable_*** signal is HIGH, then the MPC sets the **irq** interrupt HIGH. The **irq** signal remains HIGH until the IRQ_CLEAR register is written to.

——— **Note** ———

- If more security violations occur, then the MPC does not update the IRQ_STAT, IRQ_INFO1, and IRQ_INFO2 registers. However, if a security violation occurs while already in interrupt and **irq_enable_*** is set, then the MPC does update the IRQ_INFO2.ERR_MULTI bit.
- When IRQ_EN.IRQ_EN == 0 and **irq** is LOW, if coincident violations occur for a read transaction and a write transaction, the MPC saves information about the read violation in the IRQ_STAT, IRQ_INFO1, and IRQ_INFO2 registers.

———————————

Disabling interrupts for memory accesses from a debugger

If the IRQ_SET register bit is set to 0, a debugger can use the **irq_enable_*** signals to prevent interrupt generation when it accesses memory regions. The **irq_enable_rd** signal controls whether the MPC can set **irq** HIGH when a security violation occurs during a read transaction. The **irq_enable_wr** signal controls whether the MPC can set **irq** HIGH when a security violation occurs during a write transaction. The **irq_enable_*** signals also prevent the saving of the violating transaction to IRQ_INFO1 and IRQ_INFO2 registers.

——— **Note** ———

If the IRQ_SET register bit is set to 1, then the MPC ignores the **irq_enable_*** signals and generates interrupts when IRQ_EN.IRQ_EN == 1.

———————————

3.4 MPC Q-Channels

The MPC provides two Q-Channel interfaces. You can use one channel for clock control and the other channel for power control, which enables the MPC to indicate when it requires clock and power.

Both Q-Channels implement the low-power interfaces that the *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* describes.

The presence of a synchronizer on the **clk_qreqn** and **pwr_qreqn** inputs of the Q-Channels is configurable, by using the `QCLK_SYNC_EN` and `QPWR_SYNC_EN` parameters.

3.5 MPC AMBA bus properties

The AMBA protocols define multiple property types that indicate the capabilities of a device.

The following table lists the AXI5 properties of the MPC.

Table 3-1 MPC AXI5 properties

AXI5 property	Value	Comment
Wakeup_Signals	TRUE	Q-Channel activity is generated from the **awakeup** input signal.
Check_Type	FALSE	-
Poison	TRUE	The component forwards the poison data downstream, it does not use the information that the poison bits contain.
Trace_Signals	FALSE	-
QoS_Accept	FALSE	-
Loopback_Signals	FALSE	-
Untranslated_Transactions	FALSE	-
NSAccess_Identifiers	FALSE	-
Atomic_Transactions	FALSE	-

Chapter 4
PPC functional description

This chapter describes the functionality of the *Peripheral Protection Controller* (PPC) component.

It contains the following sections:

4.1 About the PPC

The *Peripheral Protection Controller* (PPC) provides security checks for AXI peripherals.

The PPC gates AXI transactions towards a peripheral when a security violation occurs. It can be instantiated in the system in connection to any non-security aware AXI5 peripheral. Security checking is performed against the state of the **cfg_ap** and **cfg_nonsec** signals, which indicate the privilege and Security state of the peripheral.

The following figure shows the PPC interfaces.

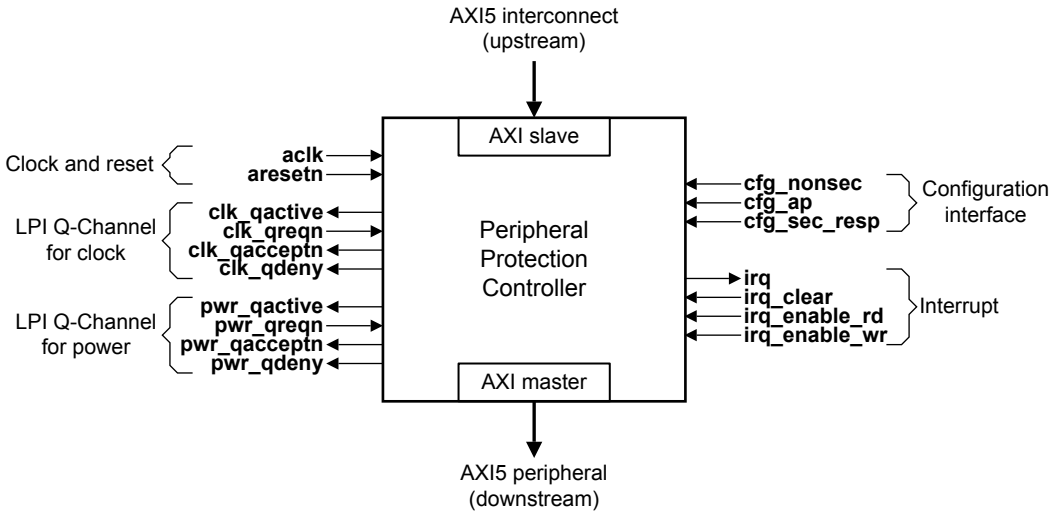

Figure 4-1 PPC interfaces

The AXI slave and AXI master interfaces provide the AXI data path from the AXI master to the attached peripheral.

To support low-power quiescence, the PPC has two Q-Channel interfaces. One Q-Channel is for clock quiescence and the other Q-Channel is for power quiescence.

Configuration interface

The **cfg_nonsec** signal controls the security settings of the attached peripheral:

- If HIGH, only Non-secure accesses to the peripheral are allowed.
- If LOW, only Secure accesses to the peripheral are allowed.

The **cfg_ap** signal controls the privilege settings of the attached peripheral:

- If HIGH, only privileged accesses to the peripheral are allowed.
- If LOW, the privilege attribute is ignored for security checks.

When the PPC blocks a transaction, the **cfg_sec_resp** signal controls whether the PPC:

- Responds with an AXI slave error (SLVERR).
- Ignores a write transaction or returns zero for a read transaction.

4.2 PPC configuration options

The PPC has design options that configure some AXI signal widths and the presence of synchronization logic on the Q-Channel inputs.

When implementing the PPC in an SoC, you can configure:

- The width of the AXI address bus, by using the `ADDR_WIDTH` parameter. The parameter can be set to a value from 12-32 inclusive.
- The width of the AXI data bus, by using the `DATA_WIDTH` parameter. The possible widths are 32, 64, 128, or 256 bits.
- The width of the ID signals on the AXI interfaces, by using the `ID_WIDTH` parameter. The parameter can be set to a value from 2-32 inclusive.
- The width of the User signals on each AXI channel, by using the `ARUSER_WIDTH`, `AWUSER_WIDTH`, `BUSER_WIDTH`, `RUSER_WIDTH`, and `WUSER_WIDTH` parameters. A parameter can be set to a value from 0-256 inclusive.
- The presence of a synchronizer on the Q-Channel **QREQn** inputs. See *4.4 PPC Q-Channels* on page 4-33.

4.3 PPC interrupts

The PPC has a level-sensitive interrupt output, **irq**, that can indicate the occurrence of a security violation.

For read transactions, the **irq_enable_rd** signal controls whether the PPC can set **irq** HIGH when a security violation occurs during a read transaction.

For write transactions, the **irq_enable_wr** signal controls whether the PPC can set **irq** HIGH when a security violation occurs during a write transaction.

4.4 PPC Q-Channels

The PPC provides two Q-Channel interfaces for power and clock control. You can use one channel for clock control and the other channel for power control, which enables the PPC to indicate when it requires clock and power.

Both Q-Channels implement the low-power interfaces that the *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* describes.

The presence of a synchronizer on the **clk_qreqn** and **pwr_qreqn** inputs of the Q-Channels is configurable, by using the `QCLK_SYNC_EN` and `QPWR_SYNC_EN` parameters.

4.5 PPC AMBA bus properties

The AMBA protocols define multiple property types that indicate the capabilities of a device.

The following table lists the AXI5 properties of the PPC.

Table 4-1 PPC AXI5 properties

AXI5 property	Value	Comment
Wakeup_Signals	TRUE	Q-Channel activity is generated from the **awakeup** input signal.
Check_Type	FALSE	-
Poison	TRUE	The component forwards the poison data downstream, it does not use the information that the poison bits contain.
Trace_Signals	FALSE	-
QoS_Accept	FALSE	-
Loopback_Signals	FALSE	-
Untranslated_Transactions	FALSE	-
NSAccess_Identifiers	FALSE	-
Atomic_Transactions	FALSE	-

Chapter 5
SMC functional description

This chapter describes the functionality of the *SRAM Memory Controller* (SMC) component.

It contains the following sections:

5.1 About the SMC

The *SRAM Memory Controller* (SMC) is an AXI5 memory controller for static memory devices.

The SMC has the following features:

- A single clock and reset domain.
- An AXI5 slave interface.
- An SRAM interface.
- Two Q-Channels for clock and power control.
- No data width conversion.
- An external-gating interface that prevents the SMC from issuing new transactions on the SRAM interface.

Note

The SRAM controller primarily supports memory macros that the Arm SRAM Compiler generates.

The following figure shows the SMC interfaces.

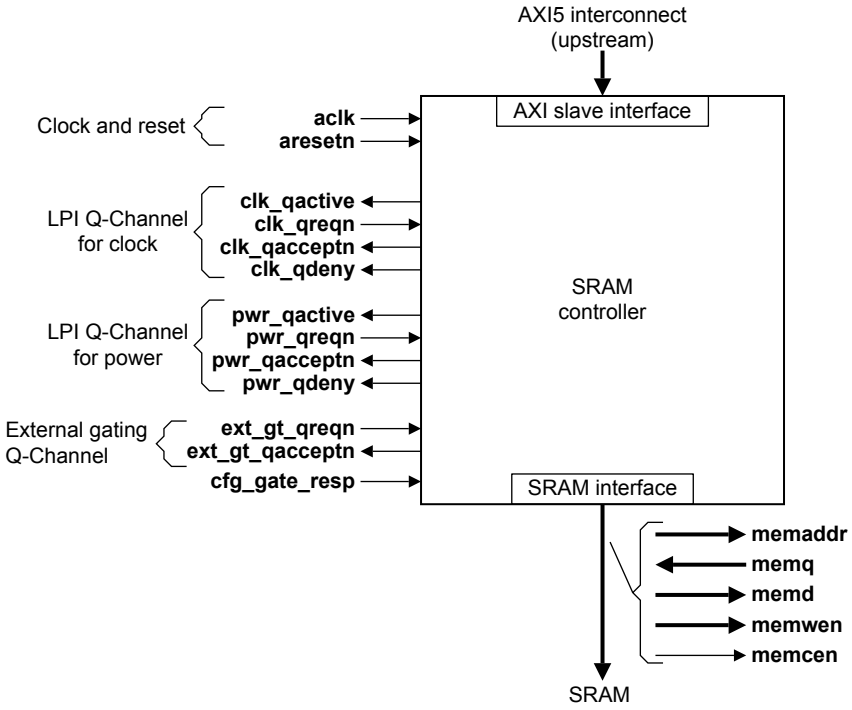

Figure 5-1 SMC interfaces

To support low-power quiescence, the SMC has two Q-Channel device interfaces. One Q-Channel is for clock quiescence and the other Q-Channel is for power quiescence.

The SMC also provides a partial Q-Channel device interface to support external gating, which stops the SMC from starting any new transactions on the SRAM interface. See *5.4 External gating of the SRAM interface* on page 5-41.

Early write response

The SMC buffers the write transactions and for non-exclusive writes it returns an early write response. If the write buffers become full, then the SMC does not return an early write response.

Read and write transaction scheduling

The AXI has separate buses for reads and writes, but the SRAM interface is a single bus for reads and writes. Therefore, the SMC must arbitrate between the AXI channels. The SMC performs arbitration between read and write bursts.

If the write buffers become full, then the arbitration scheme uses the QoS value of the incoming read, **arqos**, and write, **awqos** signal. For example, if the SMC receives a write with a QoS value that is higher than the read QoS, it forwards the oldest transaction from the write queue to the SRAM to allow the new write into the write buffer.

Note

Provided the write buffers (address or data) are not full, the SMC gives priority to read bursts.

To prevent system livelock or starvation issues, the SMC uses a balancing counter to track how many read bursts are granted while a write burst request is present but not granted. If this counter reaches a predefined value, then the SMC grants the write at the next arbitration point, regardless of the QoS values.

Poison

The `AXI5_POISON_EN` configuration parameter controls whether the SMC supports data poisoning.

When data poisoning is enabled, the SMC provides 1 bit of poison information for each 64 bits of data. The SMC uses the MSB of the **memwen** write enable bus on the SRAM memory side to control the writes to the storage element that holds the poison information.

Note

When narrow writes are written to the SRAM, the poison information always gets updated with the new value, regardless of the previously stored content.

The following table lists which signals contain the poison information for different `DATA_WIDTH` configurations.

Table 5-1 Poison bit locations

`DATA_WIDTH`	AXI poison bits	Poison bits on the SRAM interface
32	**wpoison[0]** and **rpoison[0]**	**memd[32]** and **memq[32]**
64	**wpoison[0]** and **rpoison[0]**	**memd[64]** and **memq[64]**
128	**wpoison[1:0]** and **rpoison[1:0]**	**memd[129:128]** and **memq[129:128]**
256	**wpoison[3:0]** and **rpoison[3:0]**	**memd[259:256]** and **memq[259:256]**

Exclusive accesses

The SMC can contain up to 16 *Exclusive Access Monitors* (EAMs), depending on the setting of the `EXCLUSIVE_MONITORS` configuration parameter.

If `EXCLUSIVE_MONITORS` > 0, then Exclusive Load transactions always return an EXOKAY response and the SMC stores the transaction details (address, ID) in an internal TAG buffer.

For Exclusive Store transactions, the SMC checks if the address and ID are present in the TAG buffer, and if so the SMC forwards the write to the SRAM and returns an EXOKAY write response. If the check fails, the SMC ignores the write data and it returns an OKAY response, which indicates an exclusive access failure.

If a non-exclusive write transaction accesses a location that is stored in a TAG buffer, then the SMC clears the TAG.

If all EAMs are occupied, and the SMC receives a new Exclusive Load transaction with an:

- ID that exists in the TAG table, then the new transaction replaces an old entry.
- ID that does not exist in the TAG table, then the SMC overwrites an entry in the TAG table that the round-robin algorithm selects, and returns an EXOKAY response.

If an SMC is configured to contain no EAMs (`EXCLUSIVE_MONITORS` == 0), then exclusive writes always fail. The SMC ignores the write and returns an OKAY response.

5.2 SMC configuration options

The SMC has design options that configure the number of *Exclusive Access Monitors*, the FIFO depths, and whether the first cycle of an AXI read transaction can bypass an AXI channel buffer. There are also options to set the widths of some AXI signals, and the presence of synchronization logic on the Q-Channel inputs. You can also configure the SMC to support AXI data poisoning.

When implementing the SMC in an SoC, you can configure:

- The width of the AXI address bus, by using the `ADDR_WIDTH` parameter. The parameter can be set to a value from 14-24 inclusive.
- The width of the AXI data bus, by using the `DATA_WIDTH` parameter. The possible widths are 32, 64, 128, or 256 bits.
- The width of the ID signals on the AXI interfaces, by using the `ID_WIDTH` parameter. The parameter can be set to a value from 2-32 inclusive.
- The number of exclusive access monitors, which observe and track AXI exclusive access transactions. The `EXCLUSIVE_MONITORS` parameter can be set to a value from 0-16 inclusive. The number of exclusive monitors must be less than $2^{\texttt{ID\_WIDTH}}$, since any given ID is only stored in one of the monitors. If a new exclusive read occurs with the same ID, then the tracked address is updated.
- The FIFO depth on the AR channel. The `AR_BUF_SIZE` parameter can be set to a value from 1-16 inclusive.
- The FIFO depth on the AW channel. The `AW_BUF_SIZE` parameter can be set to a value from 1-16 inclusive.
- The FIFO depth on the W channel. The `W_BUF_SIZE` parameter can be set to a value from 1-16 inclusive.
- The SMC to bypass the AR channel buffer for the first cycle of an AXI read transaction.
- The SMC to bypass the R channel buffer for the first cycle of an AXI read transaction.
- The SMC to support AXI data poisoning, by using the `AXI5_POISON_EN` parameter. See *Poison* on page 5-37.
- The presence of a synchronizer on the Q-Channel **QREQn** inputs. See *5.3 SMC Q-Channels* on page 5-40.

5.3 SMC Q-Channels

The SMC provides two Q-Channel interfaces for power and clock control. You can use one channel for clock control and the other channel for power control, which enables the SMC to indicate when it requires clock and power.

Both Q-Channels implement the low-power interfaces that the *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* describes.

The presence of a synchronizer on the **clk_qreqn** and **pwr_qreqn** inputs of the Q-Channels is configurable, by using the `QCLK_SYNC_EN` and `QPWR_SYNC_EN` parameters.

5.4 External gating of the SRAM interface

The SMC has a partial Q-Channel device interface. An external device can use the Q-Channel interface to gate transactions on the SRAM interface.

The external gating Q-Channel interface moves the SMC to a state that forcefully stops all activities towards the SRAM, when **ext_gt_qreqn** goes LOW. The SMC cannot deny an external gating request.

The SMC accepts an external gating request when there is no ongoing transfer and its internal buffers are empty. After the SMC sets **ext_gt_qacceptn** LOW, the system can apply the reset without the risk of data loss.

If the SMC receives a new AXI transaction while **ext_gt_qreqn** and **ext_gt_qacceptn** are LOW (Q_STOPPED state), then the **cfg_gate_resp** signal controls whether the SMC:

- Responds with an AXI slave error (SLVERR).
- Stalls the transaction until the external gating request is released, that is, **ext_gt_qreqn** goes HIGH.

5.5 SMC AMBA bus properties

The AMBA protocols define multiple property types that indicate the capabilities of a device.

The following table lists the AXI5 properties of the SMC.

Table 5-2 SMC AXI5 properties

AXI5 property	Value	Comment
Wakeup_Signals	TRUE	Q-Channel activity is generated from the **awakeup** input signal.
Check_Type	FALSE	-
Poison	TRUE when `AXI5_POISON_EN` == 1.	Optional feature that depends on the `AXI5_POISON_EN` configuration parameter.
Trace_Signals	FALSE	-
QoS_Accept	FALSE	-
Loopback_Signals	FALSE	-
Untranslated_Transactions	FALSE	-
NSAccess_Identifiers	FALSE	-
Atomic_Transactions	FALSE	-

Chapter 6
ACG, SDB, and SUB functional description

This chapter describes the functionality of the three bridge components, that is, the Access Control Gate, Sync-Down Bridge, and Sync-Up Bridge.

It contains the following sections:

6.1 About the bridge components

Bridge components provide low-power management and external gating on boundaries between clock and power domains along the AXI5 data bus. They also have configurable registering options to ease timing on long AXI5 paths.

The following figure shows the interfaces of a bridge component.

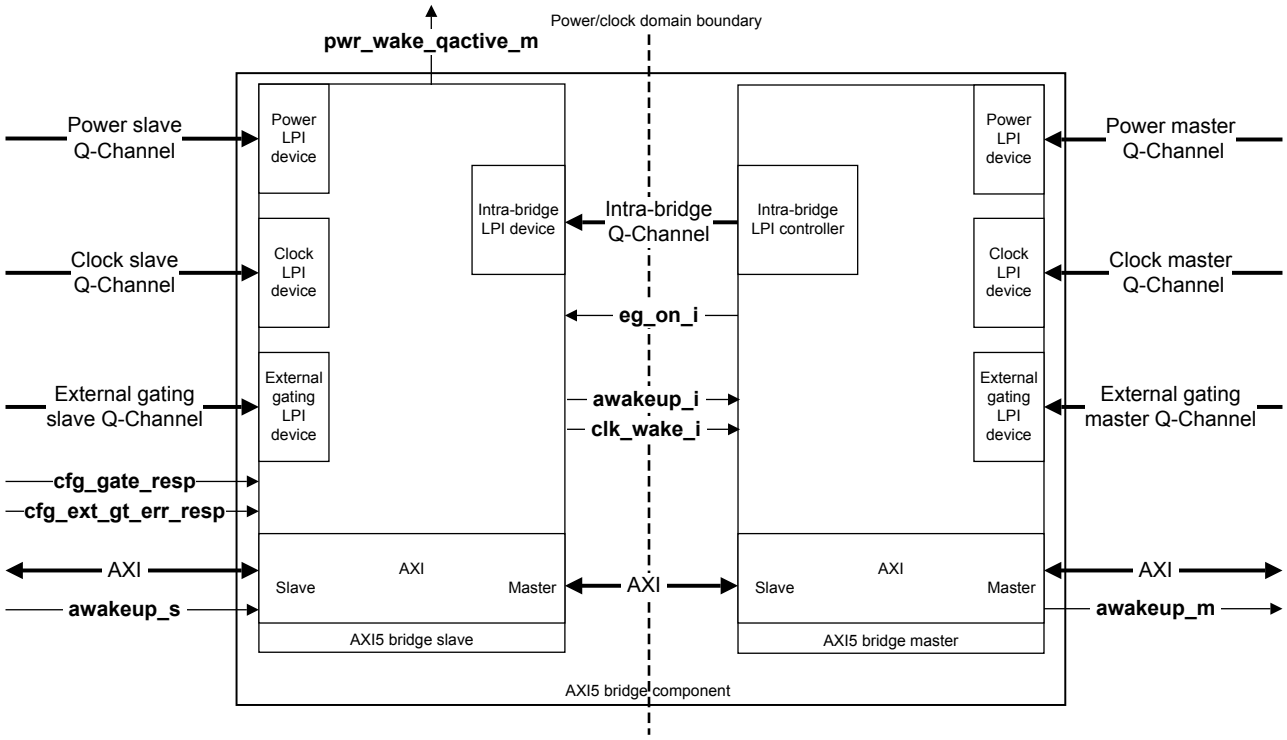

Figure 6-1 Bridge component interfaces

The following table lists the SIE-300 bridge components.

Table 6-1 Supported bridge components

Bridge component	Upstream to downstream clock ratio
Access Control Gate (ACG)	One-to-one
Sync-Down Bridge (SDB)	N-to-one
Sync-Up Bridge (SUB)	One-to-N

Each bridge component consists of an upstream side and a downstream side. To allow communication across clock and power domains, each side of the bridge has one intra-bridge Q-Channel interface and one intra-bridge AXI interface. The **eg_on_i** signal provides the upstream side with information about the state of external gating on the downstream side. The intra-bridge uses a standard Q-Channel LPI interface.

Each half of a bridge component supports the following features:

- A single clock and reset domain.
- An AXI5 slave interface.
- An AXI5 master interface.

- Two Q-Channels for clock and power control to support low-power quiescence.
- An external-gating interface that prevents the bridge component from issuing new transactions on the AXI interface. The external-gating interface is a Q-Channel implementation without the **QDENY** and **QACTIVE** signals.

Configuration interface

The **cfg_gate_resp** controls how the upstream side of the bridge component responds, when the bridge is closed by external gating or downstream power quiescence:

- Responds with an AXI slave error (SLVERR).
- The bridge component sets the relevant AXI **ready** signals LOW, which stalls any AXI transactions, until the bridge is able to forward the transfers to the downstream side.

The **cfg_ext_gt_err_resp** signal controls how the bridge component responds to AXI transactions, when the upstream external gating is in quiescence. However, if the **cfg_gate_resp** is set to error, then the bridge returns an error response. Therefore, when the upstream external gating is in quiescence, the bridge:

- Responds with an AXI slave error (SLVERR), when **cfg_gate_resp** or **cfg_ext_gt_err_resp** are HIGH.
- Stalls the transaction until the external gating request is released, that is, **ext_gt_qreqn_s** goes HIGH. This response behavior requires that **cfg_gate_resp** and **cfg_ext_gt_err_resp** are LOW.

6.2 Bridge configuration options

The bridge components have design options that configure some AXI signal widths and the presence of synchronization logic on the Q-Channel inputs. You can also configure the presence of a register slice on each AXI channel.

When implementing the Access Control Gate, Sync-Down Bridge, or Sync-Up Bridge in an SoC, you can configure:

- The width of the AXI address bus, by using the `ADDR_WIDTH` parameter. The parameter can be set to a value from 12-32 inclusive.
- The width of the AXI data bus, by using the `DATA_WIDTH` parameter. The possible widths are 32, 64, 128, or 256 bits.
- The width of the ID signals on the AXI interfaces, by using the `ID_WIDTH` parameter. The parameter can be set to a value from 2-32 inclusive.
- The width of the User signals on each AXI channel, by using the `ARUSER_WIDTH`, `AWUSER_WIDTH`, `BUSER_WIDTH`, `RUSER_WIDTH`, and `WUSER_WIDTH` parameters. A parameter can be set to a value from 0-256 inclusive.
- The presence of a register slice on each AXI channel, and which direction the registering is applied.
- The presence of a synchronizer on the Q-Channel **QREQn** inputs. See:
  - — *6.3 Bridge upstream Q-Channels* on page 6-47.
  - — *6.4 Bridge downstream Q-Channels* on page 6-48.

6.3 Bridge upstream Q-Channels

The upstream side of each bridge component provides three Q-Channel interfaces for power, clock, and external gate control.

Each Q-Channel implements the low-power interfaces that the *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* describes.

The power Q-Channel and clock Q-Channel implement a full Q-Channel interface. These interfaces enable the upstream side of the bridge to indicate when it requires clock and power.

The external-gating interface implements a partial Q-Channel interface that omits the **QDENY** and **QACTIVE** signals.

The presence of a synchronizer on the **QREQn** input of a Q-Channel is configurable. See the *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Configuration and Integration Manual* for more information.

6.4 Bridge downstream Q-Channels

The downstream side of each bridge component provides three Q-Channel interfaces for power, clock, and external gate control.

Each Q-Channel implements the low-power interfaces that the *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* describes.

The power Q-Channel and clock Q-Channel implement a full Q-Channel interface. These interfaces enable the downstream side of the bridge to indicate when it requires clock and power.

The external-gating interface implements a partial Q-Channel interface that omits the **QDENY** and **QACTIVE** signals.

The presence of a synchronizer on the **QREQn** input of a Q-Channel is configurable. See the *Arm® CoreLink™ SIE-300 AXI5 System IP for Embedded Configuration and Integration Manual* for more information.

6.5 Intra-bridge Q-Channels

The sides of each Bridge component communicate using a Q-Channel interface for power control.

The intra-bridge Q-Channel interface coordinates the powering down of the entire bridge by providing communication between the bridge sides. The power Q-Channel and external gating Q-Channel, on either side of the bridge, use the intra-bridge Q-Channel interface.

The intra-bridge Q-Channel interface also has an extra signal that indicates whether the quiescence request is from the power Q-Channel or the external gating Q-Channel. The bridge can deny a power Q-Channel quiescence request but it cannot deny an external gating Q-Channel quiescence request.

The Q-Channel implements the low-power interfaces that the *AMBA® Low Power Interface Specification, Arm® Q-Channel and P-Channel Interfaces* describes.

6.6 External gating of the AXI interface (upstream)

The upstream side of each bridge component has a Q-Channel device interface. An external device can use the Q-Channel interface to gate transactions on the AXI interface.

The external-gating interface is a Q-Channel implementation without the **QDENY** and **QACTIVE** signals.

If the upstream side of the bridge receives a gating request, that is **ext_gt_qreqn_s** goes LOW, then the bridge component stalls any new incoming AXI transactions.

The bridge component accepts an external gating request when there is no ongoing transfer and its internal buffers are empty. After the bridge component sets **ext_gt_qacceptn_s** LOW, the system can apply the reset to the upstream side of the bridge without the risk of data loss.

If the bridge component receives a new AXI transaction while **ext_gt_qreqn_s** is LOW, then the **cfg_ext_gt_err_resp** and **cfg_gate_resp** signals control whether the bridge component:

- Responds with an AXI slave error (SLVERR).
- Stalls the transaction until the external gating request is released, that is, **ext_gt_qreqn_s** goes HIGH.

6.7 External gating of the AXI interface (downstream)

The downstream side of each bridge component has a Q-Channel device interface. An external device can use the Q-Channel interface to gate transactions on the AXI interface.

The external-gating interface is a Q-Channel implementation without the **QDENY** and **QACTIVE** signals.

If the downstream side of the bridge receives a gating request, that is **ext_gt_qreqn_m** goes LOW, then the bridge sets **eg_on_i** HIGH to notify the upstream side to stall any new incoming AXI transactions.

The bridge component accepts an external gating request when there is no ongoing transfer and its internal buffers are empty. After the bridge component sets **ext_gt_qacceptn_m** LOW, the system can apply the reset to the downstream side of the bridge without the risk of data loss.

If the bridge component receives a new AXI transaction while **ext_gt_qreqn_m** is LOW, then the **cfg_gate_resp** signal controls whether the bridge component:

- Responds with an AXI slave error (SLVERR).
- Stalls the transaction until the external gating request is released, that is, **ext_gt_qreqn_m** goes HIGH.

6.8 AMBA bus properties for bridge components

The AMBA protocols define multiple property types that indicate the capabilities of a device.

The following table lists the AXI5 properties for all AXI interfaces of a bridge component.

Table 6-2 Bridge component AXI5 properties

AXI5 property	Value	Comment
Wakeup_Signals	TRUE	Q-Channel activity is generated from the **awakeup** input signal.
Check_Type	FALSE	-
Poison	TRUE	The component forwards the poison data downstream, it does not use the information that the poison bits contain.
Trace_Signals	FALSE	-
QoS_Accept	FALSE	-
Loopback_Signals	FALSE	-
Untranslated_Transactions	FALSE	-
NSAccess_Identifiers	FALSE	-
Atomic_Transactions	FALSE	-

Chapter 7
Programmers model

This chapter describes the memory regions and registers that the *Memory Protection Controller* (MPC) provides.

——— **Note** ———

The MSC, PPC, ACG, SDB, SUB, and SMC components have no registers that software can program.

———————————

It contains the following sections:

7.1 About the programmers model

The following information applies to all registers:

- 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.
  - Unless otherwise specified, all register bits are reset to a logic 0 by a system or power up reset.
- The following describes the access type:

RW	Read and write.
RO	Read-only.
WO	Write-only.

7.2 Register summary

The *Memory Protection Controller* (MPC) registers occupy a 4KB region.

The following table shows the MPC registers in offset order from the base memory address.

Table 7-1 MPC register summary

Offset	Name	Type	Width	Description
0x000	CTRL	RW	32	*7.3.1 Control register, CTRL* on page 7-56.
0x004-0x00C		-	-	Reserved, RAZ/WI.
0x010	BLK_MAX	RO	32	*7.3.2 Block index maximum value register, BLK_MAX* on page 7-57.
0x014	BLK_CFG	RO	32	*7.3.3 Block LUT configuration status register, BLK_CFG* on page 7-58.
0x018	BLK_IDX	RW	32	*7.3.4 Block LUT index register, BLK_IDX* on page 7-58.
0x01C	BLK_LUT[n]	RW	32	*7.3.5 Block LUT register, BLK_LUT* on page 7-59.
0x020	IRQ_STAT	RO	32	*7.3.6 Interrupt status register, IRQ_STAT* on page 7-60.
0x024	IRQ_CLEAR	WO	32	*7.3.7 Interrupt clear register, IRQ_CLEAR* on page 7-61.
0x028	IRQ_EN	RW	32	*7.3.8 Interrupt signal enable register, IRQ_EN* on page 7-61.
0x02C	IRQ_INFO1	RO	32	*7.3.9 Interrupt information register 1, IRQ_INFO1* on page 7-62.
0x030	IRQ_INFO2	RO	32	*7.3.10 Interrupt information register 2, IRQ_INFO2* on page 7-62.
0x034	IRQ_SET	WO	32	*7.3.11 Interrupt set register, IRQ_SET* on page 7-64.
0x038-0xFCC	-	-	-	Reserved, RAZ/WI.
0xFD0	PIDR4	RO	32	*Peripheral ID register 4, PIDR4* on page 7-64.
0xFD4	PIDR5	RO	32	*Peripheral ID registers 5-7, PIDR5, PIDR6, PIDR7* on page 7-65.
0xFD8	PIDR6	RO	32	
0xFDC	PIDR7	RO	32	
0xFE0	PIDR0	RO	32	*Peripheral ID register 0, PIDR0* on page 7-65.
0xFE4	PIDR1	RO	32	*Peripheral ID register 1, PIDR1* on page 7-66.
0xFE8	PIDR2	RO	32	*Peripheral ID register 2, PIDR2* on page 7-66.
0xFEC	PIDR3	RO	32	*Peripheral ID register 3, PIDR3* on page 7-67.
0xFF0	CIDR0	RO	32	*Component ID register 0, CIDR0* on page 7-68.
0xFF4	CIDR1	RO	32	*Component ID register 1, CIDR1* on page 7-68.
0xFF8	CIDR2	RO	32	*Component ID register 2, CIDR2* on page 7-68.
0xFFC	CIDR3	RO	32	*Component ID register 3, CIDR3* on page 7-69.

7.3 Register descriptions

This section describes the MPC registers.

7.2 Register summary on page 7-55 provides cross references to individual registers.

This section contains the following subsections:

7.3.1 Control register, CTRL

The CTRL register can lock down the programming configuration of the MPC. It also enables the BLK_IDX autoincrement and controls the AXI response when software gates AXI transactions.

Usage constraints Accessible only from Secure state. If SEC_CFG_LOCK == 1, then the register is read-only, which also prevents software from gating AXI transactions. See *7.4 Gating AXI transactions during register updates* on page 7-70.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

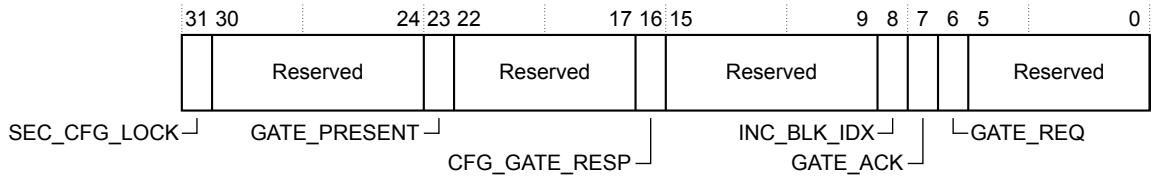

Figure 7-1 CTRL bit assignments

The following table shows the bit assignments.

Table 7-2 CTRL bit assignments

Bits	Name	Default	Description
[31]	SEC_CFG_LOCK	0	Security configuration lockdown. When set to 1, CTRL, BLK_LUT, IRQ_EN and IRQ_SET become read-only registers. To set this bit to 0, the MPC must be reset. See *7.6 Configuration lockdown* on page 7-72.
[30:24]	-	-	Reserved, RAZ/WI.

Table 7-2 CTRL bit assignments (continued)

Bits	Name	Default	Description
[23]	GATE_PRESENT	GATE_PRESENT	This read-only bit returns the state of the GATE_PRESENT parameter: 0 = The GATE_PRESENT parameter is set to 0, so the MPC does not support external gating. 1 = The GATE_PRESENT parameter is set to 1, so the MPC supports external gating.
[22:17]	-	-	Reserved, RAZ/WI.
[16]	CFG_GATE_RESP	0	When CTRL.GATE_PRESENT == 1, this bit enables software to control the AXI response type when the MPC is gating AXI transactions. The AXI response type is either: 0 = The MPC sets the AXI **ready** signals LOW, which stalls any AXI transactions. 1 = The MPC returns an AXI ERROR response. ——— **Note** ——— If gating is active (CTRL.GATE_REQ == 1), then CTRL.CFG_GATE_RESP is read-only. ——— This bit is reserved when CTRL.GATE_PRESENT == 0.
[15:9]	-	-	Reserved, RAZ/WI.
[8]	INC_BLK_IDX	1	BLK_IDX autoincrement enable. This bit is reserved when ADDR_WIDTH − BLK_CFG.BLK_SIZE < 11.
[7]	GATE_ACK	0	This read-only bit indicates if the MPC is gating incoming AXI transactions: 0 = The MPC is not gating incoming AXI transactions. 1 = The MPC is gating incoming AXI transactions. This bit is reserved when CTRL.GATE_PRESENT == 0.
[6]	GATE_REQ	0	Request to gate incoming AXI transactions. Once the MPC enters configuration lockdown (CTRL.SEC_CFG_LOCK == 1), to avoid deadlock occurring the MPC sets GATE_REQ to 0. This bit is reserved when CTRL.GATE_PRESENT == 0.
[5:0]	-	-	Reserved, RAZ/WI.

7.3.2 Block index maximum value register, BLK_MAX

The BLK_MAX register controls the maximum value that can be set in the BLK_IDX register.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

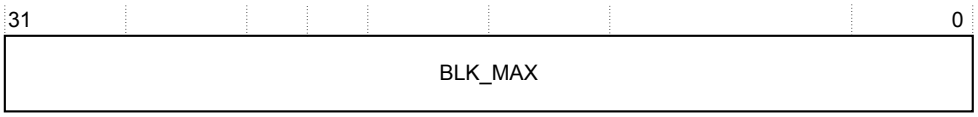

Figure 7-2 BLK_MAX bit assignments

The following table shows the bit assignments.

Table 7-3 BLK_MAX bit assignments

Bits	Name	Default	Description
[31:0]	BLK_MAX	-	Sets the maximum value that the BLK_IDX register can be set to.

Related references

7.3.4 Block LUT index register, BLK_IDX on page 7-58

7.3.3 Block LUT configuration status register, BLK_CFG

The BLK_CFG register returns the block size and whether the block initialization is in progress.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

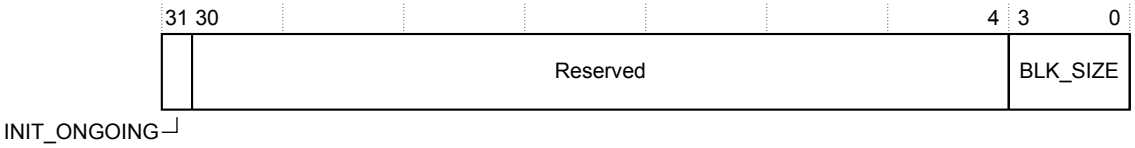

Figure 7-3 BLK_CFG bit assignments

The following table shows the bit assignments.

Table 7-4 BLK_CFG bit assignments

Bits	Name	Default	Description
[31]	INIT_ONGOING	0	When set to 1, it indicates that the block initialization is in progress.
[30:4]	-	-	Reserved, RAZ.
[3:0]	BLK_SIZE	-	This field returns the block size: 0-2 = Reserved. 3 = 256 bytes. 4 = 512 bytes. 5 = 1KB. … 15 = 1MB.

7.3.4 Block LUT index register, BLK_IDX

The BLK_IDX register controls the index pointer that selects each group of 32 blocks.

Usage constraints Accessible only from Secure state. If CTRL.SEC_CFG_LOCK == 1, then BLK_IDX is a read-only register.

Configurations Available in all configurations. However, for MPC configurations where ADDR_WIDTH − BLK_CFG.BLK_SIZE < 11, then this register is reserved.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

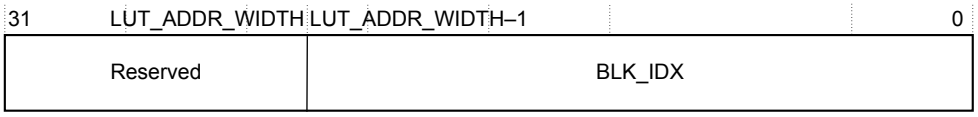

Figure 7-4 BLK_IDX bit assignments

The following table shows the bit assignments.

Table 7-5 BLK_IDX bit assignments

Bits	Name	Default	Description
[31:LUT_ADDR_WIDTH]	-	-	Reserved, RAZ/WI.
[LUT_ADDR_WIDTH – 1:0] **Note** Where: LUT_ADDR_WIDTH = ADDR_WIDTH – BLK_CFG.BLK_SIZE – 10.	BLK_IDX	0	Returns the index value that the MPC uses for accesses to the block-based LUT, using the BLK_LUT register.

Related references

7.3.2 Block index maximum value register, BLK_MAX on page 7-57

7.3.5 Block LUT register, BLK_LUT

The BLK_LUT register controls the Security state of a block. The BLK_LUT register controls up to 32 blocks and the BLK_IDX register sets which 32-block region in the *Look Up Table* (LUT) is accessed.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

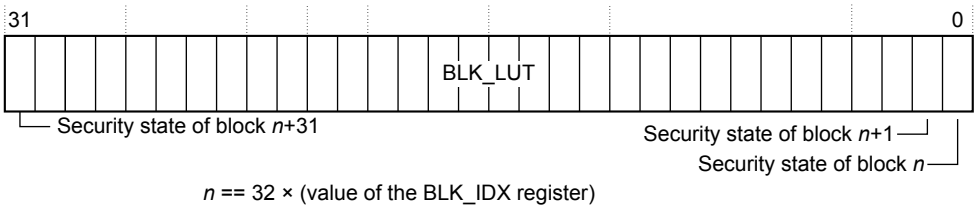

Figure 7-5 BLK_LUT bit assignments

The following table shows the bit assignments.

Table 7-6 BLK_LUT bit assignments

Bits	Name	Default	Description
[31:0]	BLK_LUT	-	Block-based gating *Look Up Table* (LUT). Each bit controls the Security state of a single block, but the block number depends on the value of the index pointer, BLK_IDX. For example: • If BLK_IDX is 0, then bit[0] contains the Security state of block #0 and bit[31] contains the Security state of block #31. • If BLK_IDX is 1, then bit[0] contains the Security state of block #32, bit[31] contains the Security state of block #63. When: • Bit[n] = 0, the block is in the Secure state. • Bit[n] = 1, the block is in the Non-secure state. If auto-increment is enabled, that is CTRL.INC_BLK_IDX == 1, then the MPC increments the BLK_IDX by one, for each word read or word write of the BLK_LUT register. The reset value of this field depends on the implementation. If `ADDR_WIDTH` − BLK_CFG.BLK_SIZE < 10 (LUT contains only 1 register, no LUT address), then the following bits are reserved: • bits[31:16] when `ADDR_WIDTH` − BLK_CFG.BLK_SIZE = 9. • bits[31:8] when `ADDR_WIDTH` − BLK_CFG.BLK_SIZE = 8. • bits[31:4] when `ADDR_WIDTH` − BLK_CFG.BLK_SIZE = 7. • bits[31:2] when `ADDR_WIDTH` − BLK_CFG.BLK_SIZE = 6.

7.3.6 Interrupt status register, IRQ_STAT

The IRQ_STAT register returns the interrupt status.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

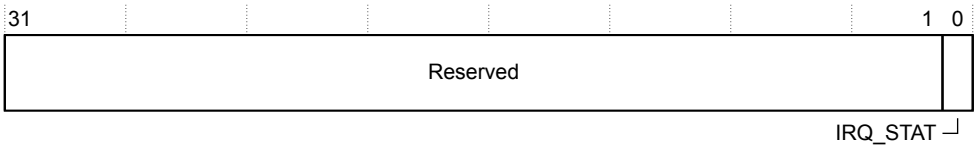

Figure 7-6 IRQ_STAT bit assignments

The following table shows the bit assignments.

Table 7-7 IRQ_STAT bit assignments

Bits	Name	Default	Description
[31:1]	-	-	Reserved, RAZ.
[0]	IRQ_STAT	0	This bit returns the interrupt status: 0 = The security violation interrupt is inactive, so the **irq** signal is LOW. 1 = The security violation interrupt is active. If IRQ_EN.IRQ_EN == 1, then the **irq** signal is HIGH.

Related references

7.3.8 Interrupt signal enable register, IRQ_EN on page 7-61

7.3.7 Interrupt clear register, IRQ_CLEAR

The IRQ_CLEAR register clears the interrupt.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

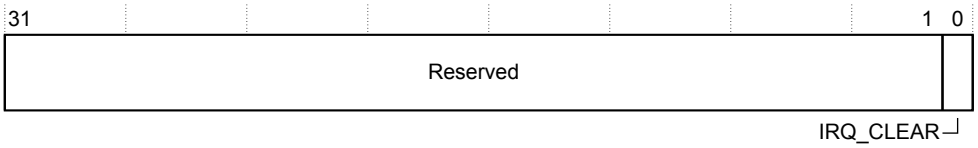

Figure 7-7 IRQ_CLEAR bit assignments

The following table shows the bit assignments.

Table 7-8 IRQ_CLEAR bit assignments

Bits	Name	Default	Description
[31:1]	-	-	Reserved.
[0]	IRQ_CLEAR	-	This bit clears the interrupt: 0 = The MPC ignores the write, so the interrupt state remains unchanged. 1 = The MPC clears the interrupt by: • Setting IRQ_STAT.IRQ_STAT = 0. • Setting IRQ_INFO2.ERR_MULTI = 0. • Setting the **irq** signal LOW.

7.3.8 Interrupt signal enable register, IRQ_EN

The IRQ_EN register controls whether the interrupt output signal, **irq**, is enabled.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

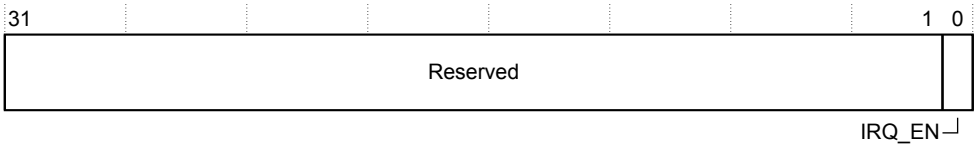

Figure 7-8 IRQ_EN bit assignments

The following table shows the bit assignments.

Table 7-9 IRQ_EN bit assignments

Bits	Name	Default	Description
[31:1]	-	-	Reserved, RAZ/WI.
[0]	IRQ_EN	0	This bit controls whether the MPC can set **irq** HIGH when a security violation occurs: 0 = The interrupt output is disabled, so the **irq** signal is always LOW. Use this setting, if software detects interrupts by polling the IRQ_STAT register. 1 = The interrupt output is enabled. Use this setting, if an interrupt controller notifies the software when an interrupt occurs. **Note** The **irq_enable_rd** and **irq_enable_wr** signals can prevent the MPC from setting the **irq** signal HIGH. See *3.3 MPC interrupts* on page 3-26.

7.3.9 Interrupt information register 1, IRQ_INFO1

The IRQ_INFO1 register returns the address of the AXI transaction that triggered the security violation interrupt.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

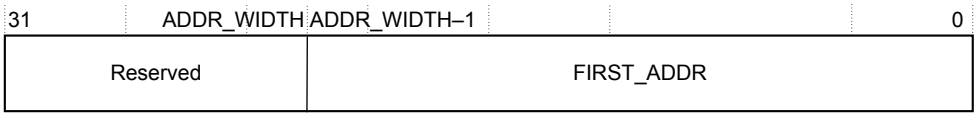

Figure 7-9 IRQ_INFO1 bit assignments

The following table shows the bit assignments.

Table 7-10 IRQ_INFO1 bit assignments

Bits	Name	Default	Description
[31:ADDR_WIDTH]	-	-	Reserved, RAZ. This field is not present when ADDR_WIDTH == 32.
[ADDR_WIDTH – 1:0]	FIRST_ADDR	0	Returns the address of the security violation that triggered the interrupt and set IRQ_STAT.IRQ.STAT = 1. When IRQ_STAT.IRQ.STAT == 1, if subsequent security violations occur, then the MPC does not update this field. **Note** The **irq_enable_rd** and **irq_enable_wr** signals can prevent the capturing of the AXI transaction that triggered the security violation. See *3.3 MPC interrupts* on page 3-26.

7.3.10 Interrupt information register 2, IRQ_INFO2

The IRQ_INFO2 register returns extra information about the AXI transaction that triggered the security violation interrupt. It also indicates the occurrence of multiple violations and coincident read and write security violations.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

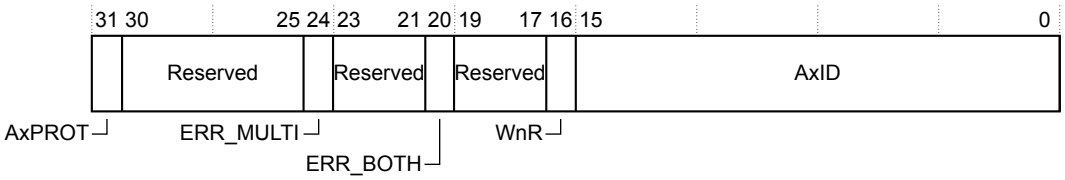

Figure 7-10 IRQ_INFO2 bit assignments

The following table shows the bit assignments.

Table 7-11 IRQ_INFO2 bit assignments

Bits	Name	Default	Description
[31]	AxPROT	0	Returns the **AxPROT[1]** value of the AXI transaction that triggered the security violation interrupt: 0 = A Secure AXI transaction triggered the interrupt. 1 = A Non-secure AXI transaction triggered the interrupt.
[30:25]	-	-	Reserved, RAZ.
[24]	ERR_MULTI	0	Indicates whether multiple security violations have occurred since the MPC set IRQ_STAT.IRQ_STAT = 1: 0 = No more security violations have occurred since the interrupt was triggered. 1 = More security violations have occurred since the interrupt was triggered.
[23:21]	-	-	Reserved, RAZ.
[20]	ERR_BOTH	0	Indicates whether coincident read and write security violations triggered the interrupt: 0 = No coincident read and write security violations have occurred when the interrupt was triggered. 1 = Coincident read and write security violations have occurred when the interrupt was triggered.
[19:17]	-	-	Reserved, RAZ.
[16]	WnR	0	Indicates whether a read or write AXI transaction triggered the security violation interrupt: 0 = A read AXI transaction triggered the interrupt. 1 = A write AXI transaction triggered the interrupt.
[15:0]	AxID	`0x0000`	Returns the ID of the AXI transaction that triggered the security violation interrupt. The field returns either: • The **arid_s[ID_WIDTH−1:0]** value, for a read transaction security violation. • The **awid_s[ID_WIDTH−1:0]** value, for write transaction security violation. If `ID_WIDTH` < 16, then the MPC sets the unallocated upper field bits to zero. **Note** The **irq_enable_rd** and **irq_enable_wr** signals can prevent the capturing of the AXI transaction that triggered the security violation. See *3.3 MPC interrupts* on page 3-26.

7.3.11 Interrupt set register, IRQ_SET

The IRQ_SET register sets the interrupt. This register is intended for debug purposes only.

Usage constraints Accessible only from Secure state.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

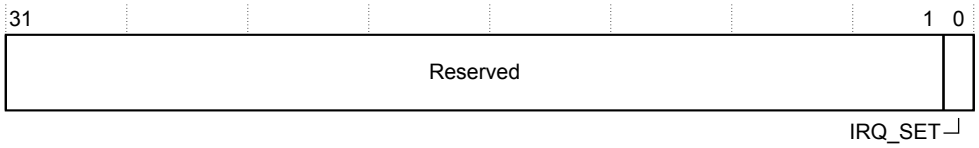

Figure 7-11 IRQ_SET bit assignments

The following table shows the bit assignments.

Table 7-12 IRQ_SET bit assignments

Bits	Name	Default	Description
[31:1]	-	-	Reserved.
[0]	IRQ_SET	-	This bit sets the interrupt: 0 = The MPC ignores the write, so the interrupt state remains unchanged. 1 = The MPC sets the interrupt by: • Setting IRQ_STAT.IRQ_STAT = 1. • Setting the **irq** signal HIGH, if IRQ_EN.IRQ_EN == 1. **Note** When the MPC sets the interrupt, it does not alter the value of the IRQ_INFO1 and IRQ_INFO2 registers.

7.3.12 Identification registers

The MPC has some ID registers that are at the end of the 4KB memory region. Software can use these registers to discover which components are present in an SoC.

Peripheral ID register 4, PIDR4

The PIDR4 register returns byte[4] of the peripheral identifier. A debugger during system discovery can use the peripheral ID to discover which peripherals are in the system and the size of the programming register space.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

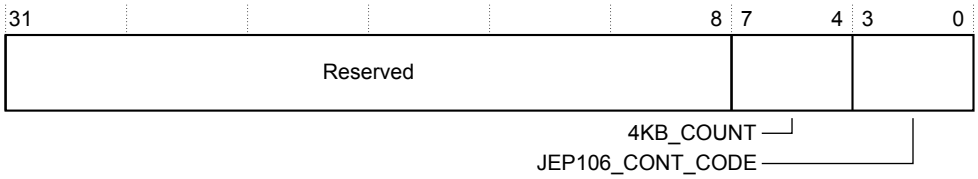

Figure 7-12 PIDR4 register bit assignments

The following table shows the bit assignments.

Table 7-13 PIDR4

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:4]	4KB_COUNT	`0x0`	Indicates that the MPC registers occupy a single 4KB page.
[3:0]	JEP106_CONT_CODE	`0x4`	Indicates how many Continuation Codes (`0x7F`) an Arm device requires. For identifying an Arm device or product, the *Standard Manufacturer's Identification Code* specifies a requirement of four Continuation Codes.

Peripheral ID registers 5-7, PIDR5, PIDR6, PIDR7

The PIDR5, PIDR6, and PIDR7 registers return byte[7:5] of the peripheral identifier. These bytes are unallocated and they all return zero.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

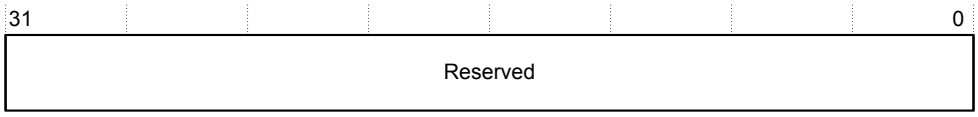

Figure 7-13 PIDR5, PIDR6, PIDR7 bit assignments

The following table shows the bit assignments.

Table 7-14 PIDR5, PIDR6, PIDR7 bit assignments

Bits	Name	Value	Function
[31:0]	-	`0x0`	Reserved.

Peripheral ID register 0, PIDR0

The PIDR0 register returns byte[0] of the peripheral identifier. A debugger during system discovery can use the peripheral ID to discover which peripherals are in the system.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

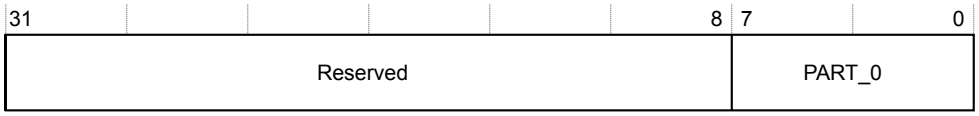

Figure 7-14 PIDR0 register bit assignments

The following table shows the register bit assignments.

Table 7-15 PIDR0

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:0]	PART_0	`0x65`	Part number, bits[7:0], for the MPC. See also PIDR1.PART_1. The MPC part number is `0x865`.

Peripheral ID register 1, PIDR1

The PIDR1 register returns byte[1] of the peripheral identifier. A debugger during system discovery can use the peripheral ID to discover which peripherals are in the system.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

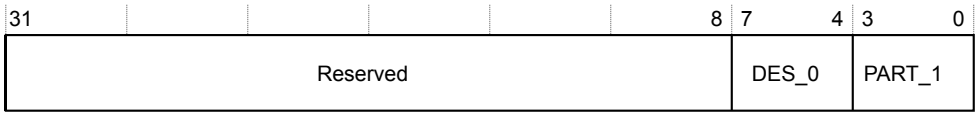

Figure 7-15 PIDR1 register bit assignments

The following table shows the register bit assignments.

Table 7-16 PIDR1

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:4]	DES_0	`0xB`	The JEDEC JEP106 ID code [3:0], which identifies Arm as the designer of the MPC. See also PIDR2.DES_1 and the *Standard Manufacturer's Identification Code*.
[3:0]	PART_1	`0x8`	Part number, bits[11:8], for the MPC. See also PIDR0.PART_0. The MPC part number is `0x865`.

Peripheral ID register 2, PIDR2

The PIDR2 register returns byte[2] of the peripheral identifier. A debugger during system discovery can use the peripheral ID to discover which peripherals are in the system and its `rmpn` revision status.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

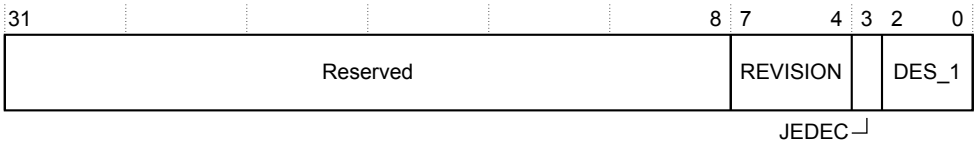

Figure 7-16 PIDR2 register bit assignments

The following table shows the register bit assignments.

Table 7-17 PIDR2

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:4]	REVISION	`0x0`	Revision identifier for the MPC: • `0x0` = r0p0. **Note** The revision status identifier for the MPC might differ from the SIE-300 revision status identifier.
[3]	JEDEC	`0b1`	Returns 1, which indicates the use of a JEDEC-assigned ID value.
[2:0]	DES_1	`0b011`	The JEDEC JEP106 ID code [6:4], which identifies Arm as the designer of the MPC. See also PIDR1.DES_0[3:0] and the *Standard Manufacturer's Identification Code*.

Peripheral ID register 3, PIDR3

The PIDR3 register returns byte[3] of the peripheral identifier. A debugger during system discovery can use the peripheral ID to discover which peripherals are in the system and whether the peripheral has any modifications applied.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

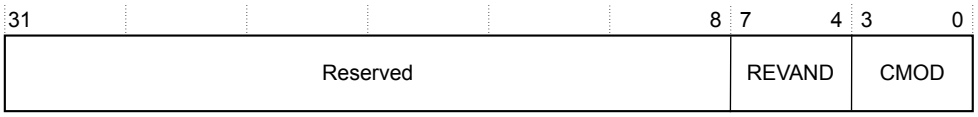

Figure 7-17 PIDR3 register bit assignments

The following table shows the register bit assignments.

Table 7-18 PIDR3

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:4]	REVAND	`0x0`	A nonzero value indicates that Arm has approved the application of a post-manufacture metal layer fix to the MPC silicon.
[3:0]	CMOD	`0x0`	Customer modification number. A nonzero value indicates that the customer has modified the MPC RTL, which might affect its behavior. Do not modify this field unless you have permission from Arm.

Component ID register 0, CIDR0

The CIDR0 register returns byte[0] of the component ID. A debugger during system discovery can use the component ID to discover that the peripheral contains a programmers register block.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

Figure 7-18 CIDR0 register bit assignments

The following table shows the register bit assignments.

Table 7-19 CIDR0

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:0]	PRMBL_0	`0x0D`	Preamble[0]. Returns segment 1 of the component identification code.

Component ID register 1, CIDR1

The CIDR1 register returns byte[1] of the component ID. A debugger during system discovery can use the component ID to discover that the peripheral contains a programmers register block and which component class the MPC belongs to.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

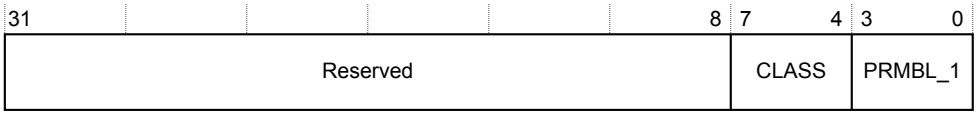

Figure 7-19 CIDR1 register bit assignments

The following table shows the register bit assignments.

Table 7-20 CIDR1

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:4]	CLASS	`0xF`	Component class. Returns `0xF`, which indicates that the MPC belongs to the CoreLink family.
[3:0]	PRMBL_1	`0x0`	Preamble[1]. Returns segment 2 of the component identification code.

Component ID register 2, CIDR2

The CIDR2 register returns byte[2] of the component ID. A debugger during system discovery can use the component ID to discover that the peripheral contains a programmers register block.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

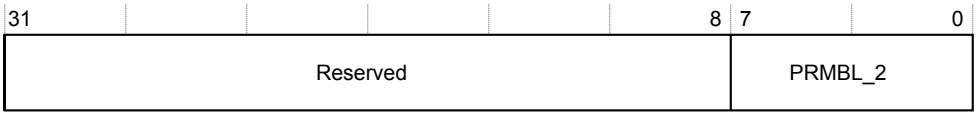

Figure 7-20 CIDR2 register bit assignments

The following table shows the register bit assignments.

Table 7-21 CIDR2

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:0]	PRMBL_2	`0x05`	Preamble[2]. Returns segment 2 of the component identification code.

Component ID register 3, CIDR3

The CIDR3 register returns byte[3] of the component ID. A debugger during system discovery can use the component ID to discover that the peripheral contains a programmers register block.

Usage constraints There are no usage constraints.

Configurations Available in all configurations.

Attributes See *7.2 Register summary* on page 7-55.

The following figure shows the bit assignments.

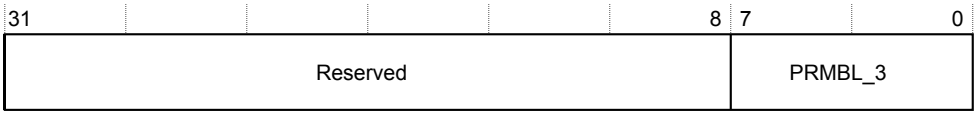

Figure 7-21 CIDR3 register bit assignments

The following table shows the register bit assignments.

Table 7-22 CIDR3

Bits	Name	Value	Function
[31:8]	-	0	Reserved.
[7:0]	PRMBL_3	`0xB1`	Preamble[3]. Returns segment 3 of the component identification code.

7.4 Gating AXI transactions during register updates

If your MPC configuration supports the AXI transaction gating feature, then software can use the CTRL register to gate AXI transactions while it updates the MPC registers. The CTRL.GATE_PRESENT bit indicates whether an MPC configuration supports AXI transaction gating.

To prevent spurious memory protection behavior when software is updating the MPC registers, it can gate AXI transactions so that no transactions are active during the update. However, software cannot gate AXI transactions if it has set configuration lockdown.

The following steps show an example of using the AXI gating feature:

Procedure

1. Set the CTRL.CFG_GATE_RESP bit to either:
   - 1. When the gating is enabled, the MPC returns a bus ERROR for each AXI transaction.
   - 0. When the gating is enabled, the MPC stalls each AXI transaction.
2. Set the CTRL.GATE_REQ bit to 1, to request that the MPC enables AXI gating.
3. Poll the CTRL.GATE_ACK bit until it is set to 1.

   When GATE_ACK == 1, the AXI gating is enabled.
4. Reprogram the internal registers or LUT content.
5. Clear the CTRL.GATE_REQ bit, to request that the MPC disables AXI gating.
6. Poll the CTRL.GATE_ACK bit until it is set to 0.

   When GATE_ACK == 0, the AXI gating is disabled.

Related references

7.3.1 Control register, CTRL on page 7-56

7.5 Programming the LUT

The contents of the *Look Up Table* (LUT) can be accessed in several ways that might require different configurations of the autoincrement function of the BLK_IDX register.

The following sections provide examples for programming the LUT.

To read the entire contents of the LUT

1. Set the autoincrement enable bit, CTRL.INC_BLK_IDX, to `0b1`.
2. Read the BLK_MAX register. This register returns the value `0xN`, which represents the last address in the LUT.
3. Write `0x0` to the BLK_IDX register.
4. Perform `0xN` + 1 reads of the BLK_LUT register, to read the entire LUT.

To write the entire contents of the LUT

1. Set autoincrement enable bit, CTRL.INC_BLK_IDX, to `0b1`.
2. Read the BLK_MAX register. This register returns the value `0xN`, which represents the last address in the LUT.
3. Write `0x0` to the BLK_IDX register.
4. Perform `0xN` + 1 writes to the BLK_LUT register, to fill the entire LUT with your chosen write data. Do not use sparse writes (**pstrb** != `0xF`), otherwise the BLK_IDX does not increment automatically.

To read-modify-write a single location

1. Set autoincrement enable bit, CTRL.INC_BLK_IDX, to `0b0`.
2. Write the required address to the BLK_IDX register.
3. Read the BLK_LUT register, to access the current contents of the LUT.
4. Write to the BLK_LUT register with your chosen write data.

   Byte accesses can be used to update only the required byte of the register without reading the full contents.

Read and write to a 16-block LUT configuration (`ADDR_WIDTH - BLK_SIZE = 9`)

1. Read the contents of the BLK_LUT register, present on bits[15:0].
2. Write the new contents to the BLK_LUT register using bits[15:0] only.

Note

As the number of blocks is less than 32, software cannot use the CTRL.INC_BLK_IDX bit or the BLK_IDX register.

7.6 Configuration lockdown

The MPC provides a configuration lockdown feature that prevents malicious software from changing the security configuration. Writing `0b1` to the security lockdown bit, CTRL.SEC_CFG_LOCK, enables the configuration lockdown feature.

Once the configuration lockdown feature is enabled:

- The following registers are read-only:
  - CTRL.
  - BLK_LUT.
  - IRQ_EN.
  - IRQ_SET.
- Disabling lockdown requires an MPC reset, which resets CTRL.SEC_CFG_LOCK to `0b0`.

Note

Arm recommends that you write `0b1` to the LUT autoincrement bit, CTRL.INC_BLK_IDX before enabling the configuration lockdown feature. When the lockdown feature is enabled, only LUT reading is available which is simpler when BLK_IDX increments automatically during the read sequence.

Related references

7.3.1 Control register, CTRL on page 7-56

Appendix A
Signal descriptions

This appendix describes the interface signals that are present for each SIE-300 component.

It contains the following sections:

A.1 MSC signals

The *Master Security Controller* (MSC) has an AXI5 slave interface and an AXI5 master interface. The MSC also has an *Implementation Defined Attribution Unit* IDAU-Lite interface, interrupts, configuration signals, and two Q-Channel device interfaces.

The following table lists the clock and reset signals.

Table A-1 MSC clock and reset signals

Signal	Direction	Description
aclk	Input	Clock
aresetn	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.

The following table lists the AXI5 slave interface signals.

Table A-2 MSC AXI5 slave interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_s	Input	Write address valid signal.
awaddr_s[ADDR_WIDTH−1:0]	Input	Write address signal.
awburst_s[1:0]	Input	Write burst type signal.
awid_s[ID_WIDTH−1:0]	Input	Write request ID signal.
awlen_s[7:0]	Input	Write burst length signal.
awsize_s[2:0]	Input	Write burst size signal.
awlock_s	Input	Write lock type signal.
awprot_s[2:0]	Input	Write protection type signal.
awready_s	Output	Write address ready signal.
awcache_s[3:0]	Input	Indicates how transactions are required to progress through a system.
awregion_s[3:0]	Input	Permits a single physical interface on a slave to be used for multiple logical interfaces.
awqos_s[3:0]	Input	QoS identifier.
awuser_s[AWUSER_WIDTH−1:0]	Input	User-defined signal.
AR channel signals:		
arvalid_s	Input	Read address valid signal.
araddr_s[ADDR_WIDTH−1:0]	Input	Read address signal.
arburst_s[1:0]	Input	Read burst type signal.
arid_s[ID_WIDTH−1:0]	Input	Read request ID signal.
arlen_s[7:0]	Input	Read address burst length signal.
arsize_s[2:0]	Input	Read burst size signal.
arlock_s	Input	Read lock type signal.
arprot_s[2:0]	Input	Read protection type signal.

Table A-2 MSC AXI5 slave interface signals (continued)

Signal	Direction	Description
arready_s	Output	Read address ready signal.
arcache_s[3:0]	Input	Indicates how transactions are required to progress through a system.
arregion_s[3:0]	Input	Permits a single physical interface on a slave to be used for multiple logical interfaces.
arqos_s[3:0]	Input	QoS identifier.
aruser_s[ARUSER_WIDTH−1:0]	Input	User-defined signal.
W channel signals:		
wvalid_s	Input	Write data valid signal.
wlast_s	Input	Indicates last transfer in a write burst.
wstrb_s[(DATA_WIDTH/8)−1:0]	Input	Write byte lane strobes.
wdata_s[DATA_WIDTH−1:0]	Input	Write data signal.
wpoison_s[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
wuser_s[WUSER_WIDTH−1:0]	Input	User-defined signal.
wready_s	Output	Write data ready signal.
R channel signals:		
rvalid_s	Output	Read data valid signal.
rid_s[ID_WIDTH−1:0]	Output	Read data ID.
rlast_s	Output	Indicates last transfer in read data.
rdata_s[DATA_WIDTH−1:0]	Output	Read data.
ruser_s[RUSER_WIDTH−1:0]	Output	User-defined signal.
rresp_s[1:0]	Output	Read data response.
rready_s	Input	Read data ready signal.
rpoison_s[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.
B channel signals:		
bvalid_s	Output	Write response valid signal.
bid_s[ID_WIDTH−1:0]	Output	Write response ID signal.
bresp_s[1:0]	Output	Write response signal.
bready_s	Input	Write response ready signal.
buser_s[BUSER_WIDTH−1:0]	Output	User-defined signal.

The following table lists the low-power signal on the AXI5 slave interface.

Table A-3 MSC AXI5 slave interface low-power signal

Signal	Direction	Description
awakeup_s	Input	When this signal is HIGH, it indicates that the AXI master is initiating activity on this interface.

The following table shows the AXI5 master interface signals. For more information about the AMBA AXI5 signals, see the *AMBA® AXI and ACE Protocol Specification*.

Table A-4 MSC AXI5 master interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_m	Output	Write address valid signal.
awaddr_m[ADDR_WIDTH−1:0]	Output	Write address signal.
awburst_m[1:0]	Output	Write burst type signal.
awid_m[ID_WIDTH−1:0]	Output	Write request ID signal.
awlen_m[7:0]	Output	Write burst length signal.
awsize_m[2:0]	Output	Write burst size signal.
awlock_m	Output	Write lock type signal.
awprot_m[2:0]	Output	Write protection type signal.
awready_m	Input	Write address ready signal.
awcache_m[3:0]	Output	Indicates how transactions are required to progress through a system.
awregion_m[3:0]	Output	Permits a single physical interface on a slave to be used for multiple logical interfaces.
awqos_m[3:0]	Output	QoS identifier.
awuser_m[AWUSER_WIDTH−1:0]	Output	Write address channel User signals.
AR channel signals:		
arvalid_m	Output	Read address valid signal.
araddr_m[ADDR_WIDTH−1:0]	Output	Read address signal.
arburst_m[1:0]	Output	Read burst type signal.
arid_m[ID_WIDTH−1:0]	Output	Read request ID signal.
arlen_m[7:0]	Output	Read address burst length signal.
arsize_m[2:0]	Output	Read burst size signal.
arlock_m	Output	Read lock type signal.
arprot_m[2:0]	Output	Read protection type signal.
arready_m	Input	Read address ready signal.
arcache_m[3:0]	Output	Indicates how transactions are required to progress through a system.
arregion_m[3:0]	Output	Permits a single physical interface on a slave to be used for multiple logical interfaces.
arqos_m[3:0]	Output	QoS identifier.
aruser_m[ARUSER_WIDTH−1:0]	Output	Read address channel User signals.
W channel signals:		
wvalid_m	Output	Write data valid signal.
wlast_m	Output	Indicates last transfer in a write burst.
wstrb_m[(DATA_WIDTH/8)−1:0]	Output	Write byte lane strobes.

Table A-4 MSC AXI5 master interface signals (continued)

Signal	Direction	Description
wdata_m[DATA_WIDTH−1:0]	Output	Write data signal.
wpoison_m[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.
wuser_m[WUSER_WIDTH−1:0]	Output	Write channel User signals.
wready_m	Input	Write data ready signal.
R channel signals:		
rvalid_m	Input	Read data valid signal.
rid_m[ID_WIDTH−1:0]	Input	Read data ID.
rlast_m	Input	Indicates last transfer in read data.
rdata_m[DATA_WIDTH−1:0]	Input	Read data.
rpoison_m[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
ruser_m[RUSER_WIDTH−1:0]	Input	Read channel User signals.
rresp_m[1:0]	Input	Read data response.
rready_m	Output	Read data ready signal.
B channel signals:		
bvalid_m	Input	Write response valid signal.
bid_m[ID_WIDTH−1:0]	Input	Write response ID signal.
bresp_m[1:0]	Input	Write response signal.
buser_m[BUSER_WIDTH−1:0]	Input	Write response User signal.
bready_m	Output	Write response ready signal.

The following table lists the sideband signals on the AXI5 master interface.

Table A-5 MSC AXI5 master interface sideband signals

Signal	Direction	Description
awakeup_m	Output	When this signal is HIGH, it indicates that the MSC is initiating activity on this interface.

The following table lists the IDAU-Lite interface.

Table A-6 IDAU-Lite interface signals

Signal	Direction	Description
idau_awaddr[31:12]	Output	Write address.
idau_awns	Input	Non-secure indicator for write address.
idau_awunchk	Input	Uncheck indicator for write address.
idau_araddr[31:12]	Output	Read address.
idau_arns	Input	Non-secure indicator for read address.
idau_arunchk	Input	Uncheck indicator for read address.

The following table lists the interrupt signals.

Table A-7 MSC interrupt signals

Signal	Direction	Description
irq	Output	When HIGH, it indicates a security violation or a faulty attribute conversion. If the MSC sets **irq** HIGH, then the signal remains HIGH until the **irq_clear** signal is set HIGH for at least one **clk** cycle.
irq_clear	Input	This signal clears the interrupt, **irq**, assertion. —————— **Note** —————— **irq_clear** has priority over interrupt generation, so when **irq_clear** is HIGH then no interrupts are generated. Therefore, you might miss a security event while **irq_clear** is HIGH. ————————————
irq_enable_rd	Input	A debugger can use this signal to prevent interrupt generation for AXI read transactions: 0 = For read transactions, disable the generation of interrupts for security violations and faulty attribute conversions. 1 = For read transactions, enable the generation of interrupts for security violations and faulty attribute conversions.
irq_enable_wr	Input	A debugger can use this signal to prevent interrupt generation for AXI write transactions: 0 = For write transactions, disable the generation of interrupts for security violations and faulty attribute conversions. 1 = For write transactions, enable the generation of interrupts for security violations and faulty attribute conversions.

The following table lists the configuration signals.

Table A-8 MSC configuration signals

Signal	Direction	Description
cfg_nonsec	Input	Indicates the Security state of the AXI master that connects to the MSC. When **cfg_nonsec** is: 0 = AXI master is in the Secure state. 1 = AXI master is in the Non-secure state. You can change the value of the configuration inputs during operation. The MSC samples the configuration inputs during the first clock cycle of an incoming transaction, when **arvalid_s** or **awvalid_s** is HIGH.
cfg_sec_resp	Input	Controls how the MSC responds when it detects a security violation. When **cfg_sec_resp** is: 0 = MSC behaves as RAZ/WI, that is, reads return zero and it ignores writes. 1 = MSC responds with an ERROR response. You can change the value of the configuration inputs during operation. The MSC samples the configuration inputs during the first clock cycle of an incoming transaction, when **arvalid_s** or **awvalid_s** is HIGH.

The following table lists the Q-Channel device signals.

Table A-9 Q-Channel signals for the MSC

Signal	Direction	Description
Clock control Q-Channel device signals:		
clk_qreqn	Input	This signal indicates when the controller issues a quiescence entry or exit request to the MSC.
clk_qacceptn	Output	This signal indicates when the MSC accepts the quiescence request.
clk_qdeny	Output	This signal indicates when the MSC denies the quiescence request.
clk_qactive	Output	This signal indicates when the MSC is active and also when it requests to exit from quiescence.
Power control Q-Channel device signals:		
pwr_qreqn	Input	This signal indicates when the controller issues a quiescence entry or exit request to the MSC.
pwr_qacceptn	Output	This signal indicates when the MSC accepts the quiescence request.
pwr_qdeny	Output	This signal indicates when the MSC denies the quiescence request.
pwr_qactive	Output	This signal indicates when the MSC is active and also when it requests to exit from quiescence.

A.2 MPC signals

The *Memory Protection Controller* (MPC) has an AXI5 slave interface and an AXI5 master interface. The MPC also has an APB4 interface, interrupts, configuration signals, and two Q-Channel device interfaces.

The following table lists the clock and reset signals.

Table A-10 MPC clock and reset signals

Signal	Direction	Description
aclk	Input	Clock
aresetn	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.

The following table lists the AXI5 slave interface signals.

Table A-11 MPC AXI5 slave interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_s	Input	Write address valid signal.
awaddr_s[ADDR_WIDTH−1:0]	Input	Write address signal.
awburst_s[1:0]	Input	Write burst type signal.
awid_s[ID_WIDTH−1:0]	Input	Write request ID signal.
awlen_s[7:0]	Input	Write burst length signal.
awsize_s[2:0]	Input	Write burst size signal.
awlock_s	Input	Write lock type signal.
awprot_s[2:0]	Input	Write protection type signal.
awready_s	Output	Write address ready signal.
awcache_s[3:0]	Input	Indicates how transactions are required to progress through a system.
awregion_s[3:0]	Input	Permits a single physical interface on a slave to be used for multiple logical interfaces.
awqos_s[3:0]	Input	QoS identifier.
awuser_s[AWUSER_WIDTH−1:0]	Input	Write address channel User signal.
AR channel signals:		
arvalid_s	Input	Read address valid signal.
araddr_s[ADDR_WIDTH−1:0]	Input	Read address signal.
arburst_s[1:0]	Input	Read burst type signal.
arid_s[ID_WIDTH−1:0]	Input	Read request ID signal.
arlen_s[7:0]	Input	Read address burst length signal.
arsize_s[2:0]	Input	Read burst size signal.
arlock_s	Input	Read lock type signal.
arprot_s[2:0]	Input	Read protection type signal.

Table A-11 MPC AXI5 slave interface signals (continued)

Signal	Direction	Description
arready_s	Output	Read address ready signal.
arcache_s[3:0]	Input	Indicates how transactions are required to progress through a system.
arregion_s[3:0]	Input	Permits a single physical interface on a slave to be used for multiple logical interfaces.
arqos_s[3:0]	Input	QoS identifier.
aruser_s[ARUSER_WIDTH−1:0]	Input	Read address channel User signal.
W channel signals:		
wvalid_s	Input	Write data valid signal.
wlast_s	Input	Indicates last transfer in a write burst.
wstrb_s[(DATA_WIDTH/8)−1:0]	Input	Write byte lane strobes.
wdata_s[DATA_WIDTH−1:0]	Input	Write data signal.
wpoison_s[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
wuser_s[WUSER_WIDTH−1:0]	Input	Write data User signal.
wready_s	Output	Write data ready signal.
R channel signals:		
rvalid_s	Output	Read data valid signal.
rid_s[ID_WIDTH−1:0]	Output	Read data ID.
rlast_s	Output	Indicates last transfer in read data.
rdata_s[DATA_WIDTH−1:0]	Output	Read data.
ruser_s[RUSER_WIDTH−1:0]	Output	Read data User signal.
rresp_s[1:0]	Output	Read data response.
rready_s	Input	Read data ready signal.
rpoison_s[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.
B channel signals:		
bvalid_s	Output	Write response valid signal.
bid_s[ID_WIDTH−1:0]	Output	Write response ID signal.
bresp_s[1:0]	Output	Write response signal.
bready_s	Input	Write response ready signal.
buser_s[BUSER_WIDTH−1:0]	Output	Write response User signal.

The following table lists the low-power signal on the AXI5 slave interface.

Table A-12 MPC AXI5 slave interface low-power signal

Signal	Direction	Description
awakeup_s	Input	When this signal is HIGH, it indicates that the AXI master is initiating activity on this interface.

The following table shows the AXI5 master interface signals. For more information about the AMBA AXI5 signals, see the *AMBA® AXI and ACE Protocol Specification*.

Table A-13 MPC AXI5 master interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_m	Output	Write address valid signal.
awaddr_m[ADDR_WIDTH−1:0]	Output	Write address signal.
awburst_m[1:0]	Output	Write burst type signal.
awid_m[ID_WIDTH−1:0]	Output	Write request ID signal.
awlen_m[7:0]	Output	Write burst length signal.
awsize_m[2:0]	Output	Write burst size signal.
awlock_m	Output	Write lock type signal.
awprot_m[2:0]	Output	Write protection type signal.
awready_m	Input	Write address ready signal.
awcache_m[3:0]	Output	Indicates how transactions are required to progress through a system.
awregion_m[3:0]	Output	Permits a single physical interface on a slave to be used for multiple logical interfaces.
awqos_m[3:0]	Output	QoS identifier.
awuser_m[AWUSER_WIDTH−1:0]	Output	Write address channel User signal.
AR channel signals:		
arvalid_m	Output	Read address valid signal.
araddr_m[ADDR_WIDTH−1:0]	Output	Read address signal.
arburst_m[1:0]	Output	Read burst type signal.
arid_m[ID_WIDTH−1:0]	Output	Read request ID signal.
arlen_m[7:0]	Output	Read address burst length signal.
arsize_m[2:0]	Output	Read burst size signal.
arlock_m	Output	Read lock type signal.
arprot_m[2:0]	Output	Read protection type signal.
arready_m	Input	Read address ready signal.
arcache_m[3:0]	Output	Indicates how transactions are required to progress through a system.
arregion_m[3:0]	Output	Permits a single physical interface on a slave to be used for multiple logical interfaces.
arqos_m[3:0]	Output	QoS identifier.
aruser_m[ARUSER_WIDTH−1:0]	Output	Read address channel User signal.
W channel signals:		
wvalid_m	Output	Write data valid signal.
wlast_m	Output	Indicates last transfer in a write burst.
wstrb_m[(DATA_WIDTH/8)−1:0]	Output	Write byte lane strobes.

Table A-13 MPC AXI5 master interface signals (continued)

Signal	Direction	Description
wdata_m[DATA_WIDTH−1:0]	Output	Write data signal.
wpoison_m[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.
wuser_m[WUSER_WIDTH−1:0]	Output	Write data User signal.
wready_m	Input	Write data ready signal.
R channel signals:		
rvalid_m	Input	Read data valid signal.
rid_m[ID_WIDTH−1:0]	Input	Read data ID.
rlast_m	Input	Indicates last transfer in read data.
rdata_m[DATA_WIDTH−1:0]	Input	Read data.
rpoison_m[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
ruser_m[RUSER_WIDTH−1:0]	Input	Read data User signal.
rresp_m[1:0]	Input	Read data response.
rready_m	Output	Read data ready signal.
B channel signals:		
bvalid_m	Input	Write response valid signal.
bid_m[ID_WIDTH−1:0]	Input	Write response ID signal.
bresp_m[1:0]	Input	Write response signal.
buser_m[BUSER_WIDTH−1:0]	Input	Write response User signal.
bready_m	Output	Write response ready signal.

The following table lists the sideband signals on the AXI5 master interface.

Table A-14 MPC AXI5 master interface sideband signals

Signal	Direction	Description
awakeup_m	Output	When this signal is HIGH, it indicates that the MPC is initiating activity on this interface.

The following table lists the APB4 slave interface signals.

Table A-15 APB slave interface

Signal	Direction	Description
psel	Input	Slave select signal.
penable	Input	Indicates the start of the second cycle of an APB transfer.
paddr[11:0]	Input	Address bus.
pprot[2:0]	Input	Protection type.
pstrb[3:0]	Input	Write byte strobe.
pwrite	Input	APB transfer direction.
pwdata[31:0]	Input	32-bit write data bus.

Table A-15 APB slave interface (continued)

Signal	Direction	Description
pwakeup	Input	The APB bridge sets this signal HIGH when a transfer is in progress.
prdata[31:0]	Output	32-bit read data bus.
pready	Output	Transfer completion indicator.
pslverr	Output	Error response.

The following table lists the interrupt signals.

Table A-16 MPC interrupt signals

Signal	Direction	Description
irq	Output	When HIGH, it indicates a security violation.
irq_enable_rd	Input	A debugger can use this signal to prevent interrupt generation for AXI read transactions: 0 = For read transactions, disable the generation of interrupts for security violations. 1 = For read transactions, enable the generation of interrupts for security violations.
irq_enable_wr	Input	A debugger can use this signal to prevent interrupt generation for AXI write transactions: 0 = For write transactions, disable the generation of interrupts for security violations. 1 = For write transactions, enable the generation of interrupts for security violations.

The following table lists the configuration signals.

Table A-17 MPC configuration signals

Signal	Direction	Description
cfg_init_value	Input	At startup, this signal initializes the *Look Up Table* (LUT) to be Secure or Non-secure, for the entire memory range that the MPC protects. When **cfg_init_value** is: 0 = The LUT content is in the Secure state. 1 = The LUT content is in the Non-secure state.
cfg_sec_resp	Input	Controls how the MPC responds when it detects a security violation. When **cfg_sec_resp** is: 0 = MPC behaves as RAZ/WI, that is, reads return zero and it ignores writes. 1 = MPC responds with an ERROR response. You can change the value of the **cfg_sec_resp** configuration input during operation. The MPC samples the **cfg_sec_resp** input during the first clock cycle of an incoming transaction, when **arvalid_s** or **awvalid_s** is HIGH.

The following table lists the Q-Channel device signals.

Table A-18 Q-Channel signals for the MPC

Signal	Direction	Description
Clock control Q-Channel device signals:		
clk_qreqn	Input	This signal indicates when the controller issues a quiescence entry or exit request to the MPC.

Table A-18 Q-Channel signals for the MPC (continued)

Signal	Direction	Description
clk_qacceptn	Output	This signal indicates when the MPC accepts the quiescence request.
clk_qdeny	Output	This signal indicates when the MPC denies the quiescence request.
clk_qactive	Output	This signal indicates when the MPC is active and also when it requests to exit from quiescence.
Power control Q-Channel device signals:		
pwr_qreqn	Input	This signal indicates when the controller issues a quiescence entry or exit request to the MPC.
pwr_qacceptn	Output	This signal indicates when the MPC accepts the quiescence request.
pwr_qdeny	Output	This signal indicates when the MPC denies the quiescence request.
pwr_qactive	Output	This signal indicates when the MPC is active and also when it requests to exit from quiescence.

A.3 PPC signals

The *Peripheral Protection Controller* (PPC) has an AXI5 slave interface and an AXI5 master interface. The PPC also has interrupts, configuration signals, and two Q-Channel device interfaces.

The following table lists the clock and reset signals.

Table A-19 PPC clock and reset signals

Signal	Direction	Description
aclk	Input	Clock
aresetn	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.

The following table lists the AXI5 slave interface signals.

Table A-20 PPC AXI5 slave interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_s	Input	Write address valid signal.
awaddr_s[ADDR_WIDTH−1:0]	Input	Write address signal.
awburst_s[1:0]	Input	Write burst type signal.
awid_s[ID_WIDTH−1:0]	Input	Write request ID signal.
awlen_s[7:0]	Input	Write burst length signal.
awsize_s[2:0]	Input	Write burst size signal.
awlock_s	Input	Write lock type signal.
awprot_s[2:0]	Input	Write protection type signal.
awready_s	Output	Write address ready signal.
awcache_s[3:0]	Input	Indicates how transactions are required to progress through a system.
awregion_s[3:0]	Input	Permits a single physical interface on a slave to be used for multiple logical interfaces.
awqos_s[3:0]	Input	QoS identifier.
awuser_s[AWUSER_WIDTH−1:0]	Input	Write address channel User signal.
AR channel signals:		
arvalid_s	Input	Read address valid signal.
araddr_s[ADDR_WIDTH−1:0]	Input	Read address signal.
arburst_s[1:0]	Input	Read burst type signal.
arid_s[ID_WIDTH−1:0]	Input	Read request ID signal.
arlen_s[7:0]	Input	Read address burst length signal.
arsize_s[2:0]	Input	Read burst size signal.
arlock_s	Input	Read lock type signal.
arprot_s[2:0]	Input	Read protection type signal.

Table A-20 PPC AXI5 slave interface signals (continued)

Signal	Direction	Description
arready_s	Output	Read address ready signal.
arcache_s[3:0]	Input	Indicates how transactions are required to progress through a system.
arregion_s[3:0]	Input	Permits a single physical interface on a slave to be used for multiple logical interfaces.
arqos_s[3:0]	Input	QoS identifier.
aruser_s[ARUSER_WIDTH−1:0]	Input	Read address channel User signal.
W channel signals:		
wvalid_s	Input	Write data valid signal.
wlast_s	Input	Indicates last transfer in a write burst.
wstrb_s[(DATA_WIDTH/8)-1:0]	Input	Write byte lane strobes.
wpoison_s[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
wuser_s[WUSER_WIDTH−1:0]	Input	Write data User signal.
wready_s	Output	Write data ready signal.
R channel signals:		
rvalid_s	Output	Read data valid signal.
rid_s[ID_WIDTH−1:0]	Output	Read data ID.
rlast_s	Output	Indicates last transfer in read data.
rdata_s[DATA_WIDTH−1:0]	Output	Read data.
ruser_s[RUSER_WIDTH−1:0]	Output	Read data User signal.
rresp_s[1:0]	Output	Read data response.
rready_s	Input	Read data ready signal.
rpoison_s[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.
B channel signals:		
bvalid_s	Output	Write response valid signal.
bid_s[ID_WIDTH−1:0]	Output	Write response ID signal.
bresp_s[1:0]	Output	Write response signal.
bready_s	Input	Write response ready signal.
buser_s[BUSER_WIDTH−1:0]	Output	Write response User signal.

The following table lists the low-power signal on the AXI5 slave interface.

Table A-21 PPC AXI5 slave interface low-power signal

Signal	Direction	Description
awakeup_s	Input	When this signal is HIGH, it indicates that the AXI master is initiating activity on this interface.

The following table shows the AXI5 master interface signals. For more information about the AMBA AXI5 signals, see the *AMBA® AXI and ACE Protocol Specification*.

Table A-22 PPC AXI5 master interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_m	Output	Write address valid signal.
awaddr_m[ADDR_WIDTH−1:0]	Output	Write address signal.
awburst_m[1:0]	Output	Write burst type signal.
awid_m[ID_WIDTH−1:0]	Output	Write request ID signal.
awlen_m[7:0]	Output	Write burst length signal.
awsize_m[2:0]	Output	Write burst size signal.
awlock_m	Output	Write lock type signal.
awprot_m[2:0]	Output	Write protection type signal.
awready_m	Input	Write address ready signal.
awcache_m[3:0]	Output	Indicates how transactions are required to progress through a system.
awregion_m[3:0]	Output	Permits a single physical interface on a slave to be used for multiple logical interfaces.
awqos_m[3:0]	Output	QoS identifier.
awuser_m[AWUSER_WIDTH−1:0]	Output	Write address channel User signal.
AR channel signals:		
arvalid_m	Output	Read address valid signal.
araddr_m[ADDR_WIDTH−1:0]	Output	Read address signal.
arburst_m[1:0]	Output	Read burst type signal.
arid_m[ID_WIDTH−1:0]	Output	Read request ID signal.
arlen_m[7:0]	Output	Read address burst length signal.
arsize_m[2:0]	Output	Read burst size signal.
arlock_m	Output	Read lock type signal.
arprot_m[2:0]	Output	Read protection type signal.
arready_m	Input	Read address ready signal.
arcache_m[3:0]	Output	Indicates how transactions are required to progress through a system.
arregion_m[3:0]	Output	Permits a single physical interface on a slave to be used for multiple logical interfaces.
arqos_m[3:0]	Output	QoS identifier.
aruser_m[ARUSER_WIDTH−1:0]	Output	Read address channel User signal.
W channel signals:		
wvalid_m	Output	Write data valid signal.
wlast_m	Output	Indicates last transfer in a write burst.
wstrb_m[(DATA_WIDTH/8)−1:0]	Output	Write byte lane strobes.
wdata_m[DATA_WIDTH−1:0]	Output	Write data signal.

Table A-22 PPC AXI5 master interface signals (continued)

Signal	Direction	Description
wpoison_m[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.
wuser_m[WUSER_WIDTH−1:0]	Output	Write data User signal.
wready_m	Input	Write data ready signal.
R channel signals:		
rvalid_m	Input	Read data valid signal.
rid_m[ID_WIDTH−1:0]	Input	Read data ID.
rlast_m	Input	Indicates last transfer in read data.
rdata_m[DATA_WIDTH−1:0]	Input	Read data.
rpoison_m[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
ruser_m[RUSER_WIDTH−1:0]	Input	Read data User signal.
rresp_m[1:0]	Input	Read data response.
rready_m	Output	Read data ready signal.
B channel signals:		
bvalid_m	Input	Write response valid signal.
bid_m[ID_WIDTH−1:0]	Input	Write response ID signal.
bresp_m[1:0]	Input	Write response signal.
buser_m[BUSER_WIDTH−1:0]	Input	Write response User signal.
bready_m	Output	Write response ready signal.

The following table lists the sideband signals on the AXI5 master interface.

Table A-23 PPC AXI5 master interface sideband signals

Signal	Direction	Description
awakeup_m	Output	When this signal is HIGH, it indicates that the PPC is initiating activity on this interface.

The following table lists the interrupt signals.

Table A-24 PPC interrupt signals

Signal	Direction	Description
irq	Output	When HIGH, it indicates a security violation or a faulty attribute conversion. If the PPC sets **irq** HIGH, then the signal remains HIGH until the **irq_clear** signal is set HIGH for at least one **clk** cycle.
irq_clear	Input	This signal clears the interrupt, **irq**, assertion. ——— **Note** ——— **irq_clear** has priority over interrupt generation, so when **irq_clear** is HIGH then no interrupts are generated. Therefore, you might miss a security event while **irq_clear** is HIGH. ———

Table A-24 PPC interrupt signals (continued)

Signal	Direction	Description
irq_enable_rd	Input	A debugger can use this signal to prevent interrupt generation for AXI read transactions: 0 = For read transactions, disable the generation of interrupts for security violations and faulty attribute conversions. 1 = For read transactions, enable the generation of interrupts for security violations and faulty attribute conversions.
irq_enable_wr	Input	A debugger can use this signal to prevent interrupt generation for AXI write transactions: 0 = For write transactions, disable the generation of interrupts for security violations and faulty attribute conversions. 1 = For write transactions, enable the generation of interrupts for security violations and faulty attribute conversions.

The following table lists the configuration signals. You can change the value of the configuration inputs during operation. The PPC samples the configuration inputs during the first clock cycle of an incoming transaction, when **arvalid_s** or **awvalid_s** is HIGH.

Table A-25 PPC configuration signals

Signal	Direction	Description
cfg_ap	Input	Indicates the privilege state of the peripheral that the PPC controls. When **cfg_ap** is: 0 = The peripheral is in the Non-privileged state. 1 = The peripheral is in the Privileged state.
cfg_nonsec	Input	Indicates the Security state of the peripheral that the PPC controls. When **cfg_nonsec** is: 0 = The peripheral is in the Secure state. 1 = The peripheral is in the Non-secure state.
cfg_sec_resp	Input	Controls how the PPC responds when it detects a security violation. When **cfg_sec_resp** is: 0 = PPC behaves as RAZ/WI, that is, reads return zero and it ignores writes. 1 = PPC responds with an ERROR response.

The following table lists the Q-Channel device signals.

Table A-26 Q-Channel signals for the PPC

Signal	Direction	Description
Clock control Q-Channel device signals:		
clk_qreqn	Input	This signal indicates when the controller issues a quiescence entry or exit request to the PPC.
clk_qacceptn	Output	This signal indicates when the PPC accepts the quiescence request.
clk_qdeny	Output	This signal indicates when the PPC denies the quiescence request.
clk_qactive	Output	This signal indicates when the PPC is active and also when it requests to exit from quiescence.
Power control Q-Channel device signals:		
pwr_qreqn	Input	This signal indicates when the controller issues a quiescence entry or exit request to the PPC.

Table A-26 Q-Channel signals for the PPC (continued)

Signal	Direction	Description
pwr_qacceptn	Output	This signal indicates when the PPC accepts the quiescence request.
pwr_qdeny	Output	This signal indicates when the PPC denies the quiescence request.
pwr_qactive	Output	This signal indicates when the PPC is active and also when it requests to exit from quiescence.

A.4 SMC signals

The *SRAM Memory Controller* (SMC) has an AXI5 slave interface and an SRAM interface. The SMC also has an external-gating interface, configuration signal, and two Q-Channel device interfaces.

The following table lists the clock and reset signals.

Table A-27 SMC clock and reset signals

Signal	Direction	Description
aclk	Input	Clock
aresetn	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.

The following table lists the AXI5 slave interface signals.

Table A-28 SMC AXI5 slave interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_s	Input	Write address valid signal.
awaddr_s[ADDR_WIDTH−1:0]	Input	Write address signal.
awburst_s[1:0]	Input	Write burst type signal.
awid_s[ID_WIDTH−1:0]	Input	Write request ID signal.
awlen_s[7:0]	Input	Write burst length signal.
awsize_s[2:0]	Input	Write burst size signal.
awlock_s	Input	Write lock type signal.
awprot_s[2:0]	Input	Write protection type signal.
awready_s	Output	Write address ready signal.
awqos_s[3:0]	Input	QoS identifier.
AR channel signals:		
arvalid_s	Input	Read address valid signal.
araddr_s[ADDR_WIDTH−1:0]	Input	Read address signal.
arburst_s[1:0]	Input	Read burst type signal.
arid_s[ID_WIDTH−1:0]	Input	Read request ID signal.
arlen_s[7:0]	Input	Read address burst length signal.
arsize_s[2:0]	Input	Read burst size signal.
arlock_s	Input	Read lock type signal.
arprot_s[2:0]	Input	Read protection type signal.
arready_s	Output	Read address ready signal.
arqos_s[3:0]	Input	QoS identifier.
W channel signals:		
wvalid_s	Input	Write data valid signal.

Table A-28 SMC AXI5 slave interface signals (continued)

Signal	Direction	Description
wlast_s	Input	Indicates last transfer in a write burst.
wstrb_s[(DATA_WIDTH/8)−1:0]	Input	Write byte lane strobes.
wdata_s[DATA_WIDTH−1:0]	Input	Write data signal.
wpoison_s[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
wready_s	Output	Write data ready signal.
R channel signals:		
rvalid_s	Output	Read data valid signal.
rid_s[ID_WIDTH−1:0]	Output	Read data ID.
rlast_s	Output	Indicates last transfer in read data.
rdata_s[DATA_WIDTH−1:0]	Output	Read data.
rresp_s[1:0]	Output	Read data response.
rready_s	Input	Read data ready signal.
rpoison_s[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.
B channel signals:		
bvalid_s	Output	Write response valid signal.
bid_s[ID_WIDTH−1:0]	Output	Write response ID signal.
bresp_s[1:0]	Output	Write response signal.
bready_s	Input	Write response ready signal.

The following table lists the low-power signal on the AXI5 slave interface.

Table A-29 SMC AXI5 slave interface low-power signal

Signal	Direction	Description
awakeup	Input	When this signal is HIGH, it indicates that the AXI master is initiating activity on this interface.

The following table shows the SRAM master interface signals.

Note

`MDAT_WIDTH` is a local variable rather than a configuration parameter, which represents the data width and the poison information. Therefore, `MDAT_WIDTH` is either:

- `DATA_WIDTH` + (`DATA_WIDTH`−1)/64 + 1, when `AXI5_POISON_EN`=1.
- `DATA_WIDTH`, when `AXI5_POISON_EN`=0.

Table A-30 SRAM master interface signals

Signal	Direction	Description
memaddr[ADDR_WIDTH−1:0]	Output	Address signal.
memd[MDAT_WIDTH−1:0]	Output	Write data signal.
memwen[(MDAT_WIDTH−1)/8:0]	Output	Byte write strobe signal.

Table A-30 SRAM master interface signals (continued)

Signal	Direction	Description
memq[MDAT_WIDTH−1:0]	Input	Read data signal.
memcen	Output	Chip select signal.

The following table lists the external-gating configuration signal.

Table A-31 SMC external-gating configuration signal

Signal	Direction	Description
cfg_gate_resp	Input	Controls how the SMC responds to AXI transactions, when an external device gates transactions on the SRAM interface. When **cfg_gate_resp** is: 0 = SMC stalls the transaction until the external gating is released. 1 = SMC responds with an ERROR response. You can change the value of the **cfg_gate_resp** during operation. The SMC samples **cfg_gate_resp** as it enters the externally gated state.

The following table lists the external gating signals. See *5.4 External gating of the SRAM interface* on page 5-41 for more information.

Table A-32 SMC external gating signals

Signal	Direction	Description
ext_gt_qreqn	Input	This signal indicates when the controller issues an external-gating entry or exit request to the SMC.
ext_gt_qacceptn	Output	This signal indicates when the SMC accepts the external gating request.

The following table lists the Q-Channel device signals.

Table A-33 Q-Channel signals for the SMC

Signal	Direction	Description
Clock control Q-Channel device signals:		
clk_qreqn	Input	This signal indicates when the controller issues a quiescence entry or exit request to the SMC.
clk_qacceptn	Output	This signal indicates when the SMC accepts the quiescence request.
clk_qdeny	Output	This signal indicates when the SMC denies the quiescence request.
clk_qactive	Output	This signal indicates when the SMC is active and also when it requests to exit from quiescence.
Power control Q-Channel device signals:		
pwr_qreqn	Input	This signal indicates when the controller issues a quiescence entry or exit request to the SMC.
pwr_qacceptn	Output	This signal indicates when the SMC accepts the quiescence request.
pwr_qdeny	Output	This signal indicates when the SMC denies the quiescence request.
pwr_qactive	Output	This signal indicates when the SMC is active and also when it requests to exit from quiescence.

A.5 Bridge components signals

The bridge components, that is, the Access Control Gate, Sync-Down Bridge, and the Sync-Up Bridge have similar signal interfaces.

This section contains the following subsections:

A.5.1 Bridge upstream signals

The upstream side of each bridge component (ACG, SDB, or SUB) has an AXI5 slave interface. Bridge components also have an external-gating interface, two configuration signals, and two Q-Channel device interfaces.

The following tables list the clock and reset signals for the upstream side of each bridge component.

Table A-34 ACG upstream clock and reset signals

Signal	Direction	Description
aclk_s	Input	Clock
aresetn_s	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.

Table A-35 SDB upstream clock and reset signals

Signal	Direction	Description
aclk_s	Input	Clock
aresetn_s	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.
aclk_en_s	Input	Clock enable signal that enables the AXI5 slave interface to operate at either: • The **aclk_s** frequency. • A divided integer multiple of **aclk_s** that is aligned to **aclk_s**.

Table A-36 SUB upstream clock and reset signals

Signal	Direction	Description
aclk_s	Input	Clock
aresetn_s	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.

The following table lists the AXI5 slave interface signals.

Table A-37 Bridge upstream AXI5 slave interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_s	Input	Write address valid signal.
awaddr_s[ADDR_WIDTH−1:0]	Input	Write address signal.
awburst_s[1:0]	Input	Write burst type signal.
awid_s[ID_WIDTH−1:0]	Input	Write request ID signal.

Table A-37 Bridge upstream AXI5 slave interface signals (continued)

Signal	Direction	Description
awlen_s[7:0]	Input	Write burst length signal.
awsize_s[2:0]	Input	Write burst size signal.
awlock_s	Input	Write lock type signal.
awprot_s[2:0]	Input	Write protection type signal.
awready_s	Output	Write address ready signal.
awcache_s[3:0]	Input	Indicates how transactions are required to progress through a system.
awqos_s[3:0]	Input	QoS identifier.
awregion_s[3:0]	Input	Permits a single physical interface on a slave to be used for multiple logical interfaces.
awuser_s[AWUSER_WIDTH−1:0]	Input	Write address channel User signal.
AR channel signals:		
arvalid_s	Input	Read address valid signal.
araddr_s[ADDR_WIDTH−1:0]	Input	Read address signal.
arburst_s[1:0]	Input	Read burst type signal.
arid_s[ID_WIDTH−1:0]	Input	Read request ID signal.
arlen_s[7:0]	Input	Read address burst length signal.
arsize_s[2:0]	Input	Read burst size signal.
arlock_s	Input	Read lock type signal.
arprot_s[2:0]	Input	Read protection type signal.
arready_s	Output	Read address ready signal.
arcache_s[3:0]	Input	Indicates how transactions are required to progress through a system.
arqos_s[3:0]	Input	QoS identifier.
arregion_s[3:0]	Input	Permits a single physical interface on a slave to be used for multiple logical interfaces.
aruser_s[ARUSER_WIDTH−1:0]	Input	Read address channel User signal.
W channel signals:		
wvalid_s	Input	Write data valid signal.
wlast_s	Input	Indicates last transfer in a write burst.
wstrb_s[(DATA_WIDTH/8)−1:0]	Input	Write byte lane strobes.
wdata_s[DATA_WIDTH−1:0]	Input	Write data signal.
wpoison_s[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
wuser_s[WUSER_WIDTH−1:0]	Input	Write data User signal.
wready_s	Output	Write data ready signal.
R channel signals:		
rvalid_s	Output	Read data valid signal.

Table A-37 Bridge upstream AXI5 slave interface signals (continued)

Signal	Direction	Description
rid_s[ID_WIDTH−1:0]	Output	Read data ID.
rlast_s	Output	Indicates last transfer in read data.
rdata_s[DATA_WIDTH−1:0]	Output	Read data.
rresp_s[1:0]	Output	Read data response.
rready_s	Input	Read data ready signal.
ruser_s[RUSER_WIDTH−1:0]	Output	Read data User signal.
rpoison_s[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.
B channel signals:		
bvalid_s	Output	Write response valid signal.
bid_s[ID_WIDTH−1:0]	Output	Write response ID signal.
buser_s[BUSER_WIDTH−1:0]	Output	Write response User signal.
bresp_s[1:0]	Output	Write response signal.
bready_s	Input	Write response ready signal.

The following table lists the wakeup signals.

Table A-38 Bridge upstream wakeup signals

Signal	Direction	Description
awakeup_s	Input	When this signal is HIGH, it indicates that the AXI master is initiating activity on this interface.
pwr_wake_qactive_m	Output	A wakeup indication towards the downstream PPU. The PPU can then wake up the downstream side of the bridge by driving the **pwr_qreqn_m** signal.

The following table lists the external gating configuration signals.

Table A-39 Bridge upstream external gating configuration signals

Signal	Direction	Description
cfg_gate_resp	Input	Controls how the bridge component responds to AXI transactions, when the downstream side of the AXI5 bridge is in external gating or power quiescence. When **cfg_gate_resp** is: 0 = the bridge component stalls the transaction until the external gating and power quiescence on each side of the bridge is released. 1 = the bridge component responds with an ERROR response. You can change the value of **cfg_gate_resp** during operation. The bridge component samples **cfg_gate_resp** during the first clock cycle of an incoming transaction, when **arvalid_s** or **awvalid_s** is HIGH. ——— **Note** ——— Do not tie **cfg_gate_resp** HIGH, otherwise the bridge becomes stuck in a closed state. ———
cfg_ext_gt_err_resp	Input	Controls how the bridge component responds to AXI transactions, when the upstream external gating is in quiescence. When **cfg_ext_gt_err_resp** is: 0 = no effect. The **cfg_gate_resp** value controls the response behavior. 1 = forced error response. The bridge component accepts all currently stalled transactions and responds with an ERROR response. You can change the value of **cfg_ext_gt_err_resp** during operation only when upstream external gating request is not asserted. The bridge component samples **cfg_ext_gt_err_resp** at the first clock cycle of upstream external gating quiescence, and is in effect only during upstream external gating quiescence.

The following table lists the external-gating signals. See *6.6 External gating of the AXI interface (upstream)* on page 6-50 for more information.

Table A-40 Bridge upstream external gating signals

Signal	Direction	Description
ext_gt_qreqn_s	Input	This signal indicates when a controller issues an external-gating entry or exit request to the bridge component.
ext_gt_qacceptn_s	Output	This signal indicates when the bridge component accepts the external gating request.

The following table lists the Q-Channel device signals.

Table A-41 Bridge upstream Q-Channel signals

Signal	Direction	Description
Clock control Q-Channel device signals:		
clk_qreqn_s	Input	This signal indicates when the controller issues a quiescence entry or exit request to the bridge component.
clk_qacceptn_s	Output	This signal indicates when the bridge component accepts the quiescence request.
clk_qdeny_s	Output	This signal indicates when the bridge component denies the quiescence request.

Table A-41 Bridge upstream Q-Channel signals (continued)

Signal	Direction	Description
clk_qactive_s	Output	This signal indicates when the bridge component is active and also when it requests to exit from quiescence.
Power control Q-Channel device signals:		
pwr_qreqn_s	Input	This signal indicates when the controller issues a quiescence entry or exit request to the bridge component.
pwr_qacceptn_s	Output	This signal indicates when the bridge component accepts the quiescence request.
pwr_qdeny_s	Output	This signal indicates when the bridge component denies the quiescence request.
pwr_qactive_s	Output	This signal indicates when the bridge component is active and also when it requests to exit from quiescence.

A.5.2 Bridge downstream signals

The downstream side of each bridge component (ACG, SDB, or SUB) has an AXI5 master interface and an SRAM interface. Bridge components also have an external-gating interface, a configuration signal, and two Q-Channel device interfaces.

The following tables list the clock and reset signals for the downstream side of each bridge component.

Table A-42 ACG downstream clock and reset signals

Signal	Direction	Description
aclk_m	Input	Clock
aresetn_m	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.

Table A-43 SDB downstream clock and reset signals

Signal	Direction	Description
aclk_m	Input	Clock
aresetn_m	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.

Table A-44 SUB downstream clock and reset signals

Signal	Direction	Description
aclk_m	Input	Clock
aresetn_m	Input	Active-LOW reset. Reset can go LOW asynchronously but must go HIGH synchronously.
aclk_en_m	Input	Clock enable signal that enables the AXI5 slave interface to operate at either: • The **aclk_m** frequency. • A divided integer multiple of **aclk_m** that is aligned to **aclk_m**.

The following table lists the AXI5 master interface signals.

Table A-45 Bridge downstream AXI5 master interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_m	Output	Write address valid signal.
awaddr_m[ADDR_WIDTH−1:0]	Output	Write address signal.
awburst_m[1:0]	Output	Write burst type signal.
awid_m[ID_WIDTH−1:0]	Output	Write request ID signal.
awlen_m[7:0]	Output	Write burst length signal.
awsize_m[2:0]	Output	Write burst size signal.
awlock_m	Output	Write lock type signal.
awprot_m[2:0]	Output	Write protection type signal.
awready_m	Input	Write address ready signal.
awcache_m[3:0]	Output	Indicates how transactions are required to progress through a system.
awqos_m[3:0]	Output	QoS identifier.
awregion_m[3:0]	Output	Permits a single physical interface on a slave to be used for multiple logical interfaces.
awuser_m[AWUSER_WIDTH−1:0]	Output	Write address channel User signal.
AR channel signals:		
arvalid_m	Output	Read address valid signal.
araddr_m[ADDR_WIDTH−1:0]	Output	Read address signal.
arburst_m[1:0]	Output	Read burst type signal.
arid_m[ID_WIDTH−1:0]	Output	Read request ID signal.
arlen_m[7:0]	Output	Read address burst length signal.
arsize_m[2:0]	Output	Read burst size signal.
arlock_m	Output	Read lock type signal.
arprot_m[2:0]	Output	Read protection type signal.
arcache_m[3:0]	Output	Indicates how transactions are required to progress through a system.
arqos_m[3:0]	Output	QoS identifier.
arregion_m[3:0]	Output	Permits a single physical interface on a slave to be used for multiple logical interfaces.
aruser_m[ARUSER_WIDTH−1:0]	Output	Read address channel User signal.
W channel signals:		
wvalid_m	Output	Write data valid signal.
wlast_m	Output	Indicates last transfer in a write burst.
wstrb_m[(DATA_WIDTH/8)−1:0]	Output	Write byte lane strobes.
wdata_m[DATA_WIDTH−1:0]	Output	Write data signal.
wpoison_m[(DATA_WIDTH−1)/64:0]	Output	64-bit data granule corruption indicator.

Table A-45 Bridge downstream AXI5 master interface signals (continued)

Signal	Direction	Description
wuser_m[WUSER_WIDTH−1:0]	Output	User-defined signal.
wready_m	Input	Write data ready signal.
R channel signals:		
rvalid_m	Output	Read data valid signal.
rid_m[ID_WIDTH−1:0]	Output	Read data ID.
rlast_m	Output	Indicates last transfer in read data.
rdata_m[DATA_WIDTH−1:0]	Output	Read data.
rresp_m[1:0]	Output	Read data response.
rready_m	Input	Read data ready signal.
ruser_m[RUSER_WIDTH−1:0]	Output	User-defined signal.
rpoison_m[(DATA_WIDTH−1)/64:0]	Input	64-bit data granule corruption indicator.
B channel signals:		
bvalid_m	Input	Write response valid signal.
bid_m[ID_WIDTH−1:0]	Input	Write response ID signal.
buser_m[BUSER_WIDTH−1:0]	Input	Write response User signal.
bresp_m[1:0]	Input	Write response signal.
bready_m	Output	Write response ready signal.

The following table lists the low-power signal on the AXI5 slave interface.

Table A-46 Bridge downstream AXI5 master interface low-power signal

Signal	Direction	Description
awakeup_m	Output	When this signal is HIGH, it indicates that the bridge component is initiating activity on this interface.

The following table lists the external gating signals. See *6.7 External gating of the AXI interface (downstream)* on page 6-51 for more information.

Table A-47 Bridge downstream external gating signals

Signal	Direction	Description
ext_gt_qreqn_m	Input	This signal indicates when the controller issues an external-gating entry or exit request to the bridge component.
ext_gt_qacceptn_m	Output	This signal indicates when the bridge component accepts the external gating request.

The following table lists the Q-Channel device signals.

Table A-48 Bridge downstream Q-Channel signals

Signal	Direction	Description
Clock control Q-Channel device signals:		
clk_qreqn_m	Input	This signal indicates when the controller issues a quiescence entry or exit request to the bridge component.
clk_qacceptn_m	Output	This signal indicates when the bridge component accepts the quiescence request.
clk_qdeny_m	Output	This signal indicates when the bridge component denies the quiescence request.
clk_qactive_m	Output	This signal indicates when the bridge component is active and also when it requests to exit from quiescence.
Power control Q-Channel device signals:		
pwr_qreqn_m	Input	This signal indicates when the controller issues a quiescence entry or exit request to the bridge component.
pwr_qacceptn_m	Output	This signal indicates when the bridge component accepts the quiescence request.
pwr_qdeny_m	Output	This signal indicates when the bridge component denies the quiescence request.
pwr_qactive_m	Output	This signal indicates when the bridge component is active and also when it requests to exit from quiescence.

A.5.3 Intra-bridge signals

The upstream and downstream sides of each bridge component (ACG, SDB, or SUB) communicate using intra-bridge interfaces. Each intra-bridge interface has an AXI5 interface, a Q-Channel device interface, two wakeup signals, and a gating signal.

The following table lists the AXI5 interface signals that connect the upstream side to the downstream side of the bridge component.

Table A-49 Intra-bridge AXI5 interface signals

Signal	Direction	Description
AW channel signals:		
awvalid_i	Upstream to downstream	Write address valid signal.
awaddr_i[ADDR_WIDTH−1:0]	Upstream to downstream	Write address signal.
awburst_i[1:0]	Upstream to downstream	Write burst type signal.
awid_i[ID_WIDTH−1:0]	Upstream to downstream	Write request ID signal.
awlen_i[7:0]	Upstream to downstream	Write burst length signal.
awsize_i[2:0]	Upstream to downstream	Write burst size signal.
awlock_i	Upstream to downstream	Write lock type signal.
awprot_i[2:0]	Upstream to downstream	Write protection type signal.
awready_i	Downstream to upstream	Write address ready signal.
awcache_i[3:0]	Upstream to downstream	Indicates how transactions are required to progress through a system.
awqos_i[3:0]	Upstream to downstream	QoS identifier.
awregion_i[3:0]	Upstream to downstream	Permits a single physical interface on a slave to be used for multiple logical interfaces.

Table A-49 Intra-bridge AXI5 interface signals (continued)

Signal	Direction	Description
awuser_i[AWUSER_WIDTH−1:0]	Upstream to downstream	Write address channel User signal.
AR channel signals:		
arvalid_i	Upstream to downstream	Read address valid signal.
araddr_i[ADDR_WIDTH−1:0]	Upstream to downstream	Read address signal.
arburst_i[1:0]	Upstream to downstream	Read burst type signal.
arid_i[ID_WIDTH−1:0]	Upstream to downstream	Read request ID signal.
arlen_i[7:0]	Upstream to downstream	Read address burst length signal.
arsize_i[2:0]	Upstream to downstream	Read burst size signal.
arlock_i	Upstream to downstream	Read lock type signal.
arprot_i[2:0]	Upstream to downstream	Read protection type signal.
arready_i	Downstream to upstream	Read address ready signal.
arcache_i[3:0]	Upstream to downstream	Indicates how transactions are required to progress through a system.
arqos_i[3:0]	Upstream to downstream	QoS identifier.
arregion_i[3:0]	Upstream to downstream	Permits a single physical interface on a slave to be used for multiple logical interfaces.
aruser_i[ARUSER_WIDTH−1:0]	Upstream to downstream	Read address channel User signal.
W channel signals:		
wvalid_i	Upstream to downstream	Write data valid signal.
wlast_i	Upstream to downstream	Indicates last transfer in a write burst.
wstrb_i[(DATA_WIDTH/8)−1:0]	Upstream to downstream	Write byte lane strobes.
wdata_i[DATA_WIDTH−1:0]	Upstream to downstream	Write data signal.
wpoison_i[(DATA_WIDTH−1)/64:0]	Upstream to downstream	64-bit data granule corruption indicator.
wuser_i[WUSER_WIDTH−1:0]	Upstream to downstream	Write data User signal.
wready_i	Downstream to upstream	Write data ready signal.
R channel signals:		
rvalid_i	Downstream to upstream	Read data valid signal.
rid_i[ID_WIDTH−1:0]	Downstream to upstream	Read data ID.
rlast_i	Downstream to upstream	Indicates last transfer in read data.
rdata_i[DATA_WIDTH−1:0]	Downstream to upstream	Read data.
rresp_i[1:0]	Downstream to upstream	Read data response.
rready_i	Upstream to downstream	Read data ready signal.
ruser_i[RUSER_WIDTH−1:0]	Downstream to upstream	Read data User signal.
rpoison_i[(DATA_WIDTH−1)/64:0]	Downstream to upstream	64-bit data granule corruption indicator.
B channel signals:		

Table A-49 Intra-bridge AXI5 interface signals (continued)

Signal	Direction	Description
bvalid_i	Downstream to upstream	Write response valid signal.
bid_i[ID_WIDTH−1:0]	Downstream to upstream	Write response ID signal.
buser_i[BUSER_WIDTH−1:0]	Downstream to upstream	Write response User signal.
bresp_i[1:0]	Downstream to upstream	Write response signal.
bready_i	Upstream to downstream	Write response ready signal.

The following table lists the intra-bridge wakeup signals.

Table A-50 Intra-bridge wakeup signals

Signal	Direction	Description
awakeup_i	Upstream to downstream	When this signal is HIGH, it indicates that the bridge is initiating activity on this interface.
clk_wake_i	Upstream to downstream	Forwards a wakeup request for the clock to be active for the downstream side of the bridge.

The following table lists the intra-bridge gating signals.

Table A-51 Intra-bridge gating signal

Signal	Direction	Description
eg_on_i	Downstream to upstream	This signal indicates when the downstream side of the bridge component receives an external gating request: 1 = The source of the intra-bridge quiescence request is an external gating request. The bridge cannot deny this request. 0 = The source of the intra-bridge quiescence request is not an external gating request. The bridge can deny the request.

The following table lists the intra-bridge Q-Channel device signals.

Table A-52 Intra-bridge Q-Channel signals

Signal	Direction	Description
ib_qreqn_i	Downstream to upstream	This signal indicates when the downstream side of the bridge component issues a quiescence entry or exit request to the upstream side of the bridge component.
ib_qacceptn_i	Upstream to downstream	This signal indicates when the bridge component accepts the quiescence request.
ib_qdeny_i	Upstream to downstream	This signal indicates when the bridge component denies the quiescence request.
ib_qactive_i	Upstream to downstream	This signal indicates when the bridge component is active and also when it requests to exit from quiescence.

Appendix B
Revisions

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

It contains the following section:

- *B.1 Revisions* on page Appx-B-106.

B.1 Revisions

This appendix describes changes between released issues of this book.

Table B-1 Issue 0000-00

Change	Location	Affects
First release.	-	-

Table B-2 Differences between issue 0000-00 and issue 0000-01

Change	Location	Affects
Added configuration information.	• *1.3 Configurable options* on page 1-14. • *2.2 MSC configuration options* on page 2-19. • *3.2 MPC configuration options* on page 3-25. • *4.2 PPC configuration options* on page 4-31. • *5.2 SMC configuration options* on page 5-39. • *6.2 Bridge configuration options* on page 6-46.	All revisions
Added IRQ_SET to the SEC_CFG_LOCK bit description. Updated the INC_BLK_IDX bit description.	*7.3.1 Control register, CTRL* on page 7-56	All revisions
Updated the formulas that contain BLK_CFG.BLK_SIZE.	• *7.3.4 Block LUT index register, BLK_IDX* on page 7-58 • *7.3.5 Block LUT register, BLK_LUT* on page 7-59	All revisions
Deleted the INT_EN register and added the IRQ_EN and IRQ_SET registers.	*7.6 Configuration lockdown* on page 7-72	All revisions

Table B-3 Differences between issue 0000-01 and issue 0100-02

Change	Location	Affects
Changed the arbitration scheme.	*Read and write transaction scheduling* on page 5-37	r1p0
Changed the eviction of a TAG table entry, when all EAMs are occupied.	*Exclusive accesses* on page 5-37	r1p0
Changed configuration signal name.	*6.6 External gating of the AXI interface (upstream)* on page 6-50	r1p0
Changed when the SMC samples **cfg_gate_resp**.	*Table A-31 SMC external-gating configuration signal* on page Appx-A-94	r1p0
Added the **cfg_ext_gt_err_resp** signal.	• *6.1 About the bridge components* on page 6-44 • *6.6 External gating of the AXI interface (upstream)* on page 6-50 • *Table A-39 Bridge upstream external gating configuration signals* on page Appx-A-98	r1p0
Updated the **cfg_gate_resp** description.	*Table A-39 Bridge upstream external gating configuration signals* on page Appx-A-98	r1p0

Table B-4 Differences between issue 0100-02 and issue 0101-03

Change	Location	Affects
Added note in the *Configuration interface* section.	*3.1 About the MPC* on page 3-24	All revisions
Updated explanation of data poisoning.	*5.1 About the SMC* on page 5-36	All revisions

Table B-4 Differences between issue 0100-02 and issue 0101-03 (continued)

Change	Location	Affects
Updated explanation of the GATE_ACK field.	*7.3.1 Control register, CTRL* on page 7-56	All revisions
Added note in FIRST_ADDR field.	*7.3.9 Interrupt information register 1, IRQ_INFO1* on page 7-62	All revisions
Added note in AxID field.	*7.3.10 Interrupt information register 2, IRQ_INFO2* on page 7-62	All revisions