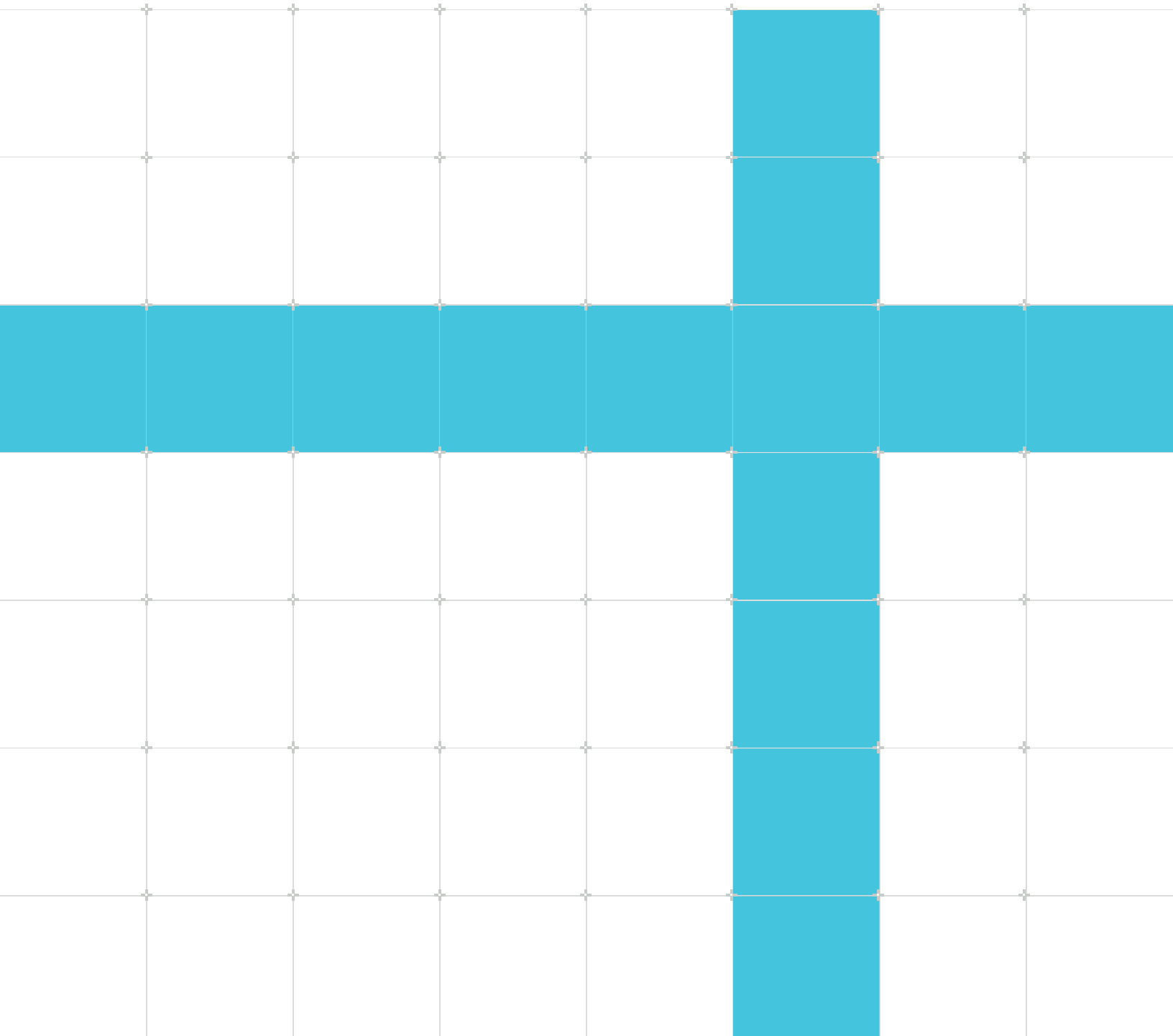
Arm® Neoverse™ MMU S3 System Memory Management Unit

Revision r1p1

Technical Reference Manual

Issue 06
102754_0101_06_en



Arm® Neoverse™ MMU S3 System Memory Management Unit Technical Reference Manual

This document (102754_0101_06_en) was issued on 2025-02-13. There might be a later issue at https://developer.arm.com/documentation/102754

The product revision is r1p1.

See also: Proprietary Notice | Product and document information | Useful resources

Start Reading

If you prefer, you can skip to the start of the content.

Intended audience

This book is written for system designers, system integrators, and programmers who are designing or programming a System-on-Chip (SoC) that uses the Arm® Neoverse™ MMU S3 System Memory Management Unit.

Inclusive language commitment

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

We believe that this document contains no offensive language. To report offensive language in this document, email terms@arm.com.

Feedback

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

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

Contents

1. About the MMU S3 System Memory Management Unit

MMU S3 is a System-level Memory Management Unit (SMMU) that translates an input address to an output address. The address translation is based on address mapping and memory attribute information that is available in the MMU S3 internal registers and translation tables.

MMU S3 implements the following features that the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* defines:

- Arm® SMMU architecture version 3.3, SMMUv3.3
- Realm Management Extension for Device Assignment (RME-DA)

An address translation from an input address to an output address is described as a stage of address translation. MMU S3 can perform:

- Stage 1 translations that translate an input virtual address (VA) to an output physical address (PA) or intermediate physical address (IPA)
- Stage 2 translations that translate an input IPA to an output PA
- Combined stage 1 and stage 2 translations that translate an input VA to an IPA, and then translate that IPA to an output PA.

MMU S3 performs translation table walks for each stage of the translation.

In addition to translating an input address to an output address, a stage of address translation also defines the memory attributes of the output address. With a two-stage translation, the stage 2 translation can modify the attributes that the stage 1 translation defines. You can disable or bypass a stage of address translation, and MMU S3 can define memory attributes for disabled and bypassed stages of translation.

MMU S3 uses inputs from the requester to identify a context. Configuration tables in memory define how MMU S3 is to translate each context, such as which translation tables to use.

MMU S3 can also perform Granule Protection Checks (GPCs). If enabled, these checks must be performed on both:

- SMMU-originated transaction
- Client-originated transactions

The checks ensure that the Physical Address Space of the transaction matches the Physical Address Space that the Granule Protection Tables describe.

MMU S3 can cache the result of a translation table lookup in a Translation Lookaside Buffer (TLB). It can also cache configuration tables in a configuration cache.

MMU S3 contains the following key components:

- Translation Buffer Units (TBUs) that use a TLB to cache translation tables
- A Translation Control Unit (TCU) that controls and manages address translations
- Bi-directional AXI Stream (BAS) components that connect multiple TBUs to the TCU and carry Distributed Translation Interface (DTI) translations

1.1 Compliance

MMU S3 complies with, or implements, the specifications that this section describes.

This Technical Reference Manual (TRM) complements architecture reference manuals, architecture specifications, protocol specifications, and relevant external standards. It does not duplicate information from these sources.

Arm architecture

MMU S3 implements parts of the Arm v8.6 and v9.1 Virtual Memory System Architecture (VMSA), as the *Arm® Architecture Reference Manual for A-profile architecture* defines. The SMMUv3.3 architecture describes the parts of VMSA that apply to MMU S3 and describes support for Realm Management Extension (RME).

SMMU architecture

MMU S3 implements the SMMUv3.3 architecture. See the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

AMBA Distributed Translation Interface protocol

MMU S3 implements v3 of the Distributed Translation Interface (DTI) protocol, as the *AMBA® DTI Protocol Specification* defines. The DTI interfaces use the AXI5-Stream transport layer interface, with Wakeup_Signal enabled and Check_Type not enabled, as the *AMBA® AXI-Stream Protocol Specification* defines.

AMBA ACE5-Lite and AMBA AXI5 protocol

MMU S3 complies with the AMBA® ACE5-Lite protocol. For more information, see the *AMBA® AXI Protocol Specification*.

AMBA® APB protocol

MMU S3 complies with the AMBA® APB5 protocol, as the *AMBA® APB Protocol Specification* defines.

AMBA® LTI protocol

MMU S3 complies with the AMBA® LTI protocol, as the *AMBA® LTI Protocol Specification* defines.

LPI Q-Channel protocol

MMU S3 complies with the LPI Q-Channel, as the *AMBA® Low Power Interface Specification* defines.

1.2 Features

MMU S3 provides features such as compliance with architectures, support for architecture extensions, support for flexible integration, high speed translation, and trace debugging.

The MMU S3 features are as follows:

Compliance with the SMMUv3.3 architecture

- Support for stage 1 translation, stage 2 translation, and stage 1 followed by stage 2 translation
- Support for Armv8 AArch64 translation table format
- Support for 4 KB, 16 KB, and 64 KB granule sizes in AArch64 format
- Support for PCI Express (PCIe) integration, including the following:
  - Address Translation Services (ATS), including full and split-stage ATS
  - Process Address Space IDs (PASIDs)
  - Access Control Services (ACS)
- Support for Page Request Interface (PRI), as SMMUv3 defines. PRI is an optional PCIe ATS extension that enables support for unpinned memory in PCIe
- Support for Memory System Resource Partitioning and Monitoring (MPAM)
- Support for the Secure-EL2 translation regime
- Requesters can be stalled while a processor handles translation faults, enabling software support for on-demand paging.
- Configuration tables in memory
- Queues in memory that perform MMU S3 management. There is no requirement to stall a processor when it accesses MMU S3.
- A Performance Monitoring Unit (PMU) in each TBU and TCU that enables MMU S3 performance to be investigated
- Reliability, Serviceability, and Availability (RAS) features for RAM corruption detection and correction
- E0PD, PTWNNC, ATSRECERR, and MPAM_NS features from SMMUv3.3

Support for Realm Management Extension

- Granule Protection Checks (GPCs) for SMMU-originated and client-originated accesses
- Root PAS registers for GPC
- Reporting of GPC faults
- Support for GPC invalidations

Support for Realm Management Extension Device Assignment support

- Realm translation regime
- Realm queues

- Realm programmers model
- Memory Encryption Context (MEC) support
- DPT checks and caching of DPT information

Support for AMBA® interfaces

- ACE5-Lite TBU transaction interfaces that support cache stash transactions, deallocating transactions, and Cache Maintenance Operations (CMOs)
- Support for AXI5 untranslated transactions version 3
- An ACE5-Lite+Distributed Virtual Memory (DVM) TCU table walk interface that enables Armv8.6 and Realm Management Extension (RME)-capable processors to perform shared TLB invalidate operations without accessing MMU S3 directly.
- An ACE5-Lite TCU table walk interface that enables Armv8.6 and RME-capable processors to perform shared TLB invalidate operations without accessing MMU S3 directly
- An ACE5 Low-Power extension that enables the TCU to subscribe to DVM TLB invalidate requests on powerup and powerdown
- AMBA® DTI communication between the TCU and TBUs, enabling requesters to request translations and implement TBU functionality internally
- Support for the AMBA® Low-Power Interface (LPI) Q-Channel so that standard controllers can control power and clock gating
- WAKEUP signaling on all interfaces, including DTI and APB interfaces
- Support for ACE5-Lite atomic transactions in the ACE-Lite TBU
- Support for Local Translation Interface (LTI). LTI is a point-to-point protocol that enables devices to directly request a translation for each transaction while leaving the TBU to manage the Translation Lookaside Buffer (TLB)
- Support for a dedicated Generic Interrupt Controller (GIC) integration, with Message Signaled Interrupts (MSIs) supported for common interrupt types

Support for flexible integration

- You can place a configurable number of TBUs close to the requesters being translated
- Communication between TBU and TCU over AXI5-Stream is supported using the supplied DTI interconnect components, or any other AXI5-Stream interconnect, with Wakeup_Signal enabled and Check_Type not enabled
- DTI interconnect components support hierarchical topologies and control the tradeoff between the number of wires and the DTI bandwidth

Support for high-performance translation

- Scalable configurable MicroTLB and Main TLB (MTLB) in the TBU can reduce the number of translation requests to the TCU
- TBU direct indexing and MTLB partitioning enable the use of MTLB entries to be managed outside the TBU, improving real-time translation performance

- Optimization enables storage of all architecturally-defined page and block sizes, including contiguous page and block entries, as a single entry in the TBU TLBs and TCU walk caches
- Per-TBU prioritization in the TCU enables high-priority transaction streams to be translated before low-priority streams
- TCU prefetch of translation tables, which can be enabled on a per-context basis, improves translation performance for real-time requesters that access memory linearly
- Hit-Under-Miss (HUM) support in the TBU enables transactions with different AXI IDs to be propagated out of order, when a translation is available. Reordering of transactions is possible only if the transactions are also in different ordering groups.
- TBU detects multiple transactions that require the same translation so that only one TBU request to the TCU is required
- TCU detects multiple translations that require the same table in memory so that only one TCU memory request is required
- Multi-level, multi-stage walk caches in the TCU reduce translation cost by performing only part of the table walk process on a miss
- Granule Protection Table (GPT) caches which cache GPT information, covering both SMMU-originated and client-originated accesses
- A configurable number of concurrent translations in the TBU and TCU promotes high translation throughput.

Translation logging using an Embedded Logic Analyzer

- Using a CoreSight™ ELA-600 Embedded Logic Analyzer to log port activities

1.3 Interfaces

TCUs and TBUs support some common interfaces.

Both TCUs and TBUs support the following common interfaces:

- Clocks and resets
- Distributed Translation Interface (DTI)
- Tie-offs
- Interrupts
- Performance Monitoring Unit (PMU) snapshot
- Test and debug
- Low Power Interface (LPI) clock gating
- LPI powerdown
- ELA observation interface

The TCU also supports the following interfaces:

- Programming interface, APB5 PROG
- Primary ACE-Lite+DVM Queue and Table Walk (QTW)/DVM interface
- Secondary ACE-Lite Page Table Walk (PTW) interface, in dual AXI mode only
- Generic Interrupt Controller (GIC) Message Signaled Interrupt (MSI) interface
- Event interface, eventoreq and eventoack
- System Coherency interface (SYSCO)

The ACE-Lite TBU also supports the following interfaces:

- Transaction subordinate (TBS)
- Transaction manager (TBM)

The LTI TBU also supports the Local Translation Interface (LTI).

See also 2.4 Interfaces on page 31.

1.4 Configurable options

MMU S3 is a highly configurable System Memory Management Unit that provides configuration options for each of the main components.

For the TCU, you can configure the following:

- Size and structure of each cache
- Data width of the Queue and Table Walk (QTW)/DVM and Page Table Walk (PTW) interface
- Number of translations that can be performed at the same time
- Number of Granule Protection Checks (GPCs) that can be performed at the same time for client-originated accesses. You cannot configure the number of parallel GPCs for SMMU-originated accesses. The number of possible SMMU-originated accesses determines the number of parallel GPCs.
- Number of translation requests that can be accepted from all DTI requesters including TBU and ATS requesters
- Type of some RAMs

For the TBU, you can configure the following:

- Size of each cache
- Number of transactions that can be translated at the same time
- Register slices
- Type of some RAMs

For the ACE-Lite TBU, you can configure the following:

- Write data buffer depth

- Number of outstanding read and write transactions that the TBM interface supports
- Width of data, ID, User, StreamID, and SubstreamID signals on the TBS and TBM interfaces


Note

Depths are specified as a discrete number of entries.

You can also configure the DTI interconnect components to meet your system requirements.

See 2.7 Configuration parameters and methodology on page 120.

1.5 Product documentation and design flow

MMU S3 documents are used during different parts of the design flow.

1.5.1 Documentation

We deliver MMU S3 with a Technical Reference Manual (TRM) and a Configuration and Integration Manual (CIM).

The MMU S3 documentation is as follows:

Technical Reference Manual

The Technical Reference Manual (TRM) describes the functionality and the effects of functional options on the behavior of MMU S3. The TRM is required at all stages of the design flow. The choices you make in the design flow can mean that some behaviors that the TRM describes are not relevant. If you are programming MMU S3, then contact:

- The implementer to determine:
  - The build configuration of the implementation
  - The integration, if any, that you performed before you implemented MMU S3
- The integrator to determine the pin configuration of the device that you are using.

Configuration and Integration Manual

The Configuration and Integration Manual (CIM) describes:

- The available build configuration options and related issues in selecting them
- How to integrate MMU S3 into a SoC. This section describes the pins that the integrator must tie off to configure the macrocells for the required integration.
- The processes to sign off on the configuration, integration, and implementation of the design

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

1.5.2 Design flow

We deliver MMU S3 as synthesizable RTL. Before you can use MMU S3 in a product, you must ensure that you implement the following processes:

Implementation

The implementer configures and synthesizes the RTL to produce a hard macrocell. This process might include integrating RAMs into the design.

Integration

The integrator connects the implemented design into a SoC. Integration includes connecting the design to a memory system and peripherals.

Programming

The system programmer develops the software to configure and initialize MMU S3, and tests the required application software.

Each process is separate. Processes can include the implementation and integration choices that affect MMU S3 behavior and features.

The operation of the final device depends on:

Build configuration

The implementer chooses the options that affect how to pre-process the RTL source files. These options usually include or exclude logic that affects one or more of the following:

- Area
- Maximum frequency
- Features of the resulting macrocell

Configuration inputs

The integrator configures some features of MMU S3 by tying inputs to specific values. These configurations affect the start-up behavior before you configure any software.

Software configuration

The programmer configures MMU S3 by programming particular values into registers. This configuration affects the behavior of MMU S3.

See also:

- 1.1 Compliance on page 12
- 1.4 Configurable options on page 16
- 2.7 Configuration parameters and methodology on page 120

1.6 Product revisions

This section describes the differences in the functionality between product revisions.

r0p0 First release

r0p0-r1p0 The following changes apply to this release:

- Support for Device Permission Table (DPT) checks. See 2.1 Translation Control Unit on page 22.
- Support for 6-port LTI TBU.
- New registers. See 3. Programmers model for MMU S3 on page 129.
- New and updated system discovery registers. See 3.9 TCU system discovery registers on page 187.
- New and updated parameters. See 2.7 Configuration parameters and methodology on page 120.
- New signals. See A.1.12 TCU tie-off signals on page 312.
- New TCU events. See 2.5.2.2 SMMU TCU events on page 58.

r1p0-r1p1 The following changes apply to this release:

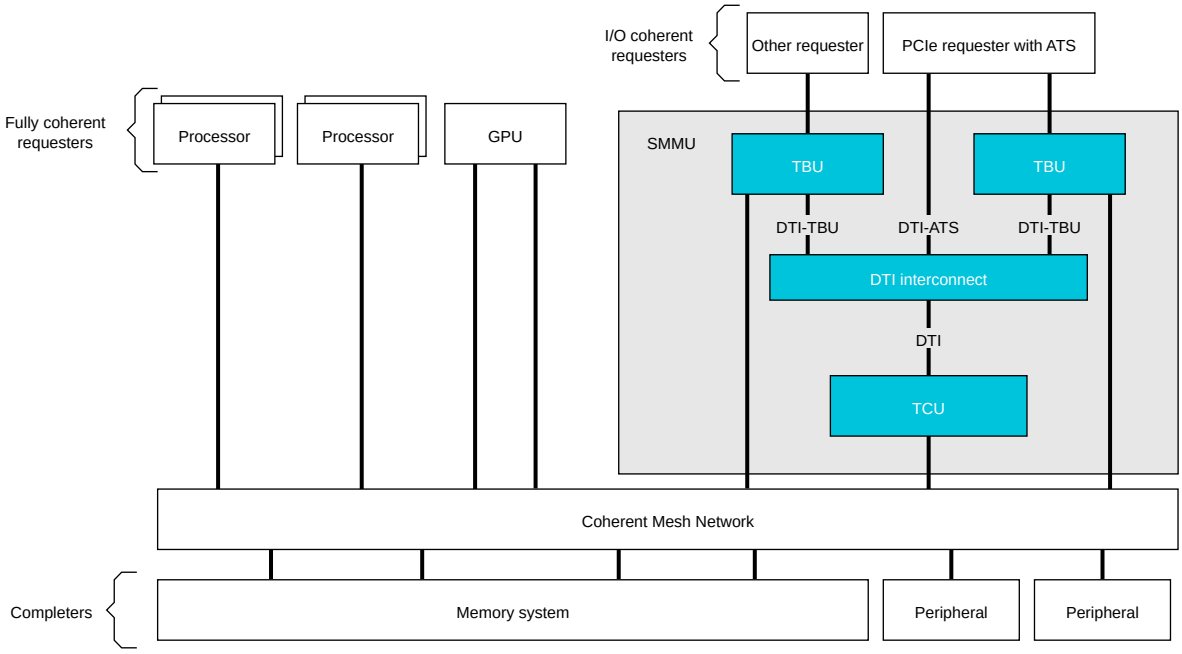
- Errata fixes.

2. Functional description of MMU S3

The major functional blocks of MMU S3 are the Translation Buffer Unit (TBU), Translation Control Unit (TCU), and Distributed Translation Interface (DTI) interconnect.

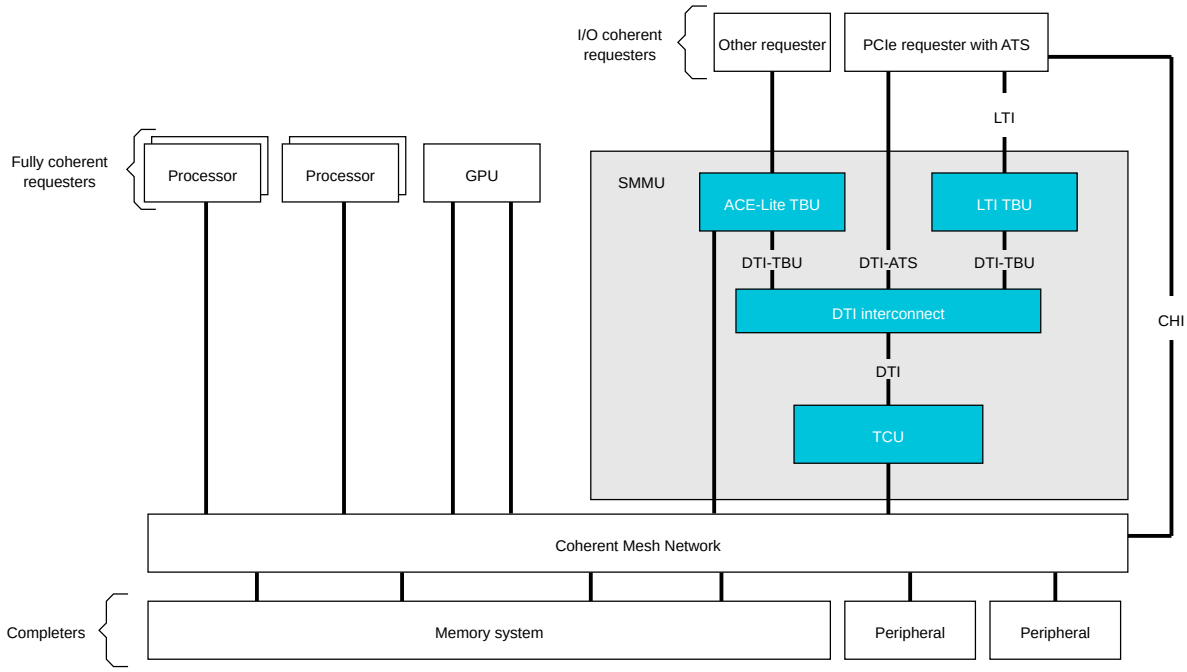
The following figure shows an example system that uses MMU S3.

Figure 2-1: Example system containing MMU S3



The following figure shows an example system that uses MMU S3, and includes a Local Translation Interface (LTI) TBU.

Figure 2-2: Example system with MMU S3 and LTI TBU



MMU S3 contains the following key components:

Translation Buffer Unit (TBU)

The TBU contains Translation Lookaside Buffers (TLBs) that cache translation tables. MMU S3 implements a TBU that can be connected to a single requester or multiple requesters. It is also possible to connect multiple TBUs to a single requester to improve performance. These TBUs are local to the corresponding requester and can be one of the following:

- ACE-Lite TBU.
- LTI TBU.

Translation Control Unit (TCU)

The TCU controls and manages the address translations. MMU S3 implements a single TCU. In MMU S3-based systems, the *AMBA®* DTI protocol defines the standard for communicating with the TCU. See the *AMBA® DTI Protocol Specification*.

DTI interconnect

The DTI interconnect connects multiple TBUs to the TCU.

When an MMU S3 TBU receives a transaction on the TBS or LA channel of the LTI interface, it looks for a matching translation in its TLBs. If it has a matching translation, it uses it to translate the transaction and outputs the transaction on the TBM interface, or LR channel of the LTI interface. If it does not have a matching translation, it requests a new translation from the TCU using the DTI interface.

When the TCU receives a DTI translation request, it uses the QTW/PTW interfaces to perform the following:

- Configuration table walks, that return configuration information for the translation context.
- Translation table walks, that return translation information that is specific to the transaction address.
- Granule Protection Table (GPT) walk, that returns a set of GPT tables that are used to check the physical address space with which each granule is associated.
- Device Permission Table (DPT) walk for Address Translation Services (ATS) on PCIe.

The TCU contains caches that reduce the number of configuration, translation, Granule Protection Table (GPT), and Device Permission Table (DPT) walks that are to be performed. Sometimes no walks are required.

When the TBU receives the translation from the TCU, it stores it in its TLBs. If the translation was successful, the TBU uses it to translate the transaction, otherwise it terminates it.

A processor controls the TCU by performing the following:

- Writing commands to a Command queue in memory.
- Receiving events from an Event queue in memory.
- Writing to its configuration registers using the programming interface.

The *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* describes the following:

- Translation.
- Translation for Realm devices.
- How software communicates with the TCU.
- Granule protection checks.
- Device permission checks.
- How software communicates with Granule Protection Check (GPC) related aspects of the TCU.
- Memory Encryption Context (MEC).

2.1 Translation Control Unit

A typical SMMUv3-based system includes a single Translation Control Unit (TCU). The TCU is usually the largest block in the system, and performs several roles.

The TCU performs the following roles:

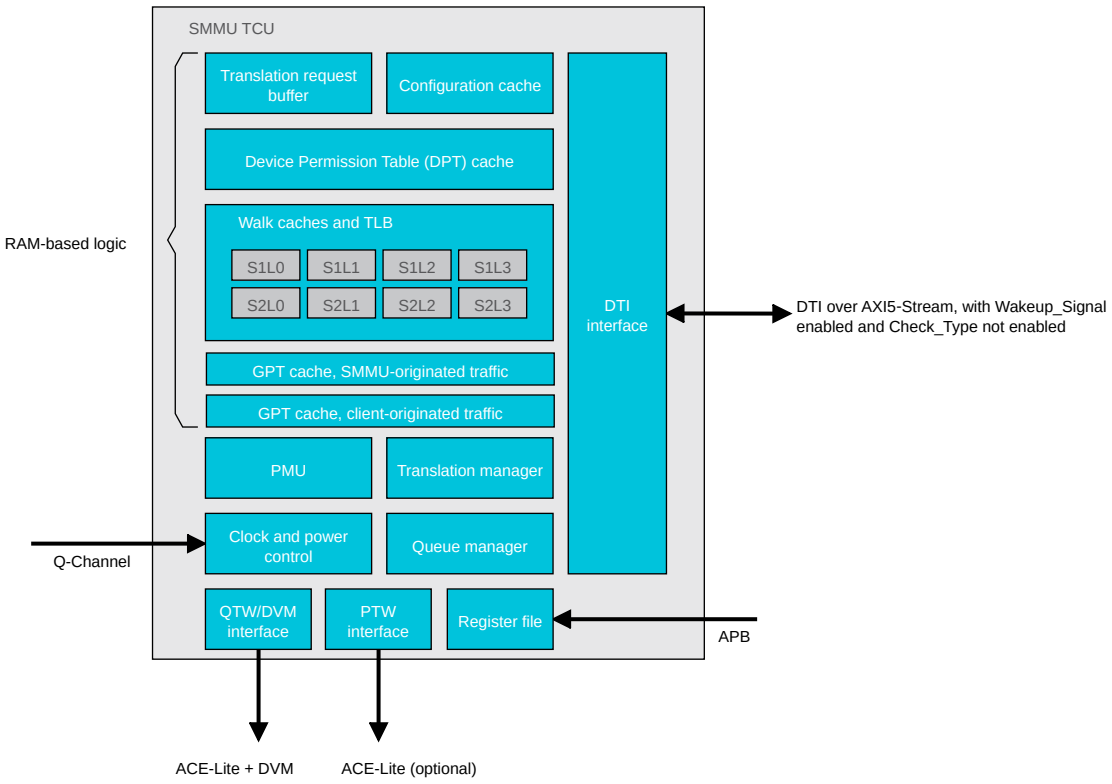
- Manages the memory queues
- Performs the following:
  - Translation table walks
  - Configuration table walks
  - Granule Protection Table (GPT) walks

- Device Permission Table (DPT) walks

- Implements the following:
  - Backup caching structures
  - The SMMU programmers model

The following figure shows the TCU with its main interfaces. The figure is illustrative of the functions that the block performs but might not match exactly with the block names inside the RTL.

Figure 2-3: MMU S3 TCU



Note

The ACE-Lite PTW interface is optional.

The TCU consists of:

Walk cache

The walk cache is a set-associative walk cache that has a configurable number of banks and ways and stores the results of translation table walks. During runtime, the cache line entries can be programmatically split to create separate walk caches that are reserved for:

- Stage 1 level 0 table entries.
- Stage 1 level 1 table and block entries.
- Stage 1 level 2 table and block entries.
- Stage 1 level 3 block entries.
- Stage 2 level 0 table entries.
- Stage 2 level 1 table and block entries.
- Stage 2 level 2 table and block entries.
- Stage 2 level 3 block entries.

To enable and disable the walk cache for a particular stage and level of translation, use the 3.7.1 TCU_CTRL register on page 156. If an error occurs for a cache line entry, the 3.8.3 TCU_ERRSTATUS register on page 177 identifies the affected entry.

The walk cache is useful in cases where a translation request results in a miss in other TCU caches. A subsequent hit in the walk cache requires only a single memory access to complete the translation table walk and fetch the required descriptor.

Configuration cache

The configuration caches are set-associative cache structures that store configuration information. Each entry stores the Context Descriptor (CD) and Stream Table Entry (STE) contents for a translation context.


Note

The configuration cache does not cache the contents of intermediate configuration tables.

Granule Protection Table (GPT) cache

The GPT cache is a set-associative granule protection walk cache that has a configurable number of banks and ways and stores the results of GPT walks. Two GPT caches exist in the TCU for the following:

- Client-originated AXI Granule Protection Check (GPC) checks, DTI GPC Wrapper (DGW).
- SMMU-originated GPC checks, AXI GPC Wrapper (AGW).

During runtime, the cache line entries can be programmatically split to create separate walk caches that are reserved for the following:

- L0 block and table entries.
- L1 contiguous and granule entries.

To enable and disable the GPT cache for a particular stage and level of translation, use the 3.7.7 TCU_ROOT_CTRL register on page 163. If an error occurs for a cache line entry, use the 3.8.3 TCU_ERRSTATUS register on page 177 to identify the affected entry.

Device Permission Table (DPT) cache

The DPT cache is a set-associative device permission walk cache that has a configurable number of banks and ways. The DPT cache stores device permission results based on the following:

- Virtual Memory System Architecture (VMSA) translations performed as part of responding to Address Translation Services (ATS) Translation Requests, that is, successful ATS Translation Completion.
- Successful DPT walks.

During runtime, the cache line entries can be programmatically split to create separate walk caches that are reserved for the following:

- Level 0 table and block entries.
- Level 1 granule and contiguous entries.


Note

Level 1 contiguous entries with a size equal to the size of SMMU_R_DPT_BASE_CFG.L0DPTSZ are converted to L0 block entries internally to the DPT cache.

To enable and disable the DPT cache for a particular level, use the 3.7.9 TCU_R_CTRL register on page 165. If an error occurs for a cache line entry, use the 3.8.3 TCU_ERRSTATUS register on page 177 to identify the affected entry.

Translation manager

The translation manager manages translation requests that are in progress.

Translation request buffer

The translation request buffer stores translation requests from TBUs when all translation manager slots are full. The translation request buffer supports more slots than the translation manager. When correctly configured, this buffer has enough space to store all translation requests that TBUs can issue simultaneously. This buffer therefore prevents the DTI interface from becoming blocked.

PMU

The PMU counts TCU performance-related events and has a configurable number of counters to count the events.

Clock and power control

The TCU has its own clock and power control, that the Q-Channels provide.

Queue manager

The queue manager manages all SMMUv3 queues that are stored in memory as follows:

- Command queues, Secure and Non-secure.
- Event queues, Secure and Non-secure.
- PRI queue, Non-secure.

QTW/DVM and PTW interfaces

The Queue and Table Walk (QTW)/Distributed Virtual Memory (DVM) is an ACE-Lite+DVM interface. The Page Table Walk (PTW) interface is an ACE-Lite interface.

Register file

The register file implements the SMMUv3 programmers model, as the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* defines.

DTI interface

The completer (TCU) DTI interface uses the DTI protocol, over AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled, to enable the TCU to communicate with a requester (TBU) component. For MMU S3, the requester (TBU) component is either a TBU or a PCIe requester.

See also:

- 2.4 Interfaces on page 31
- 2.5 Operation on page 54
- 2.5.8 TCU transaction handling on page 79
- 2.5.9 TCU prefetch on page 81
- 3.2 SMMU architectural registers on page 131

2.2 Translation Buffer Unit

A typical SMMUv3-based system includes multiple Translation Buffer Units (TBUs). Each TBU is located close to the component for which it provides address translation.

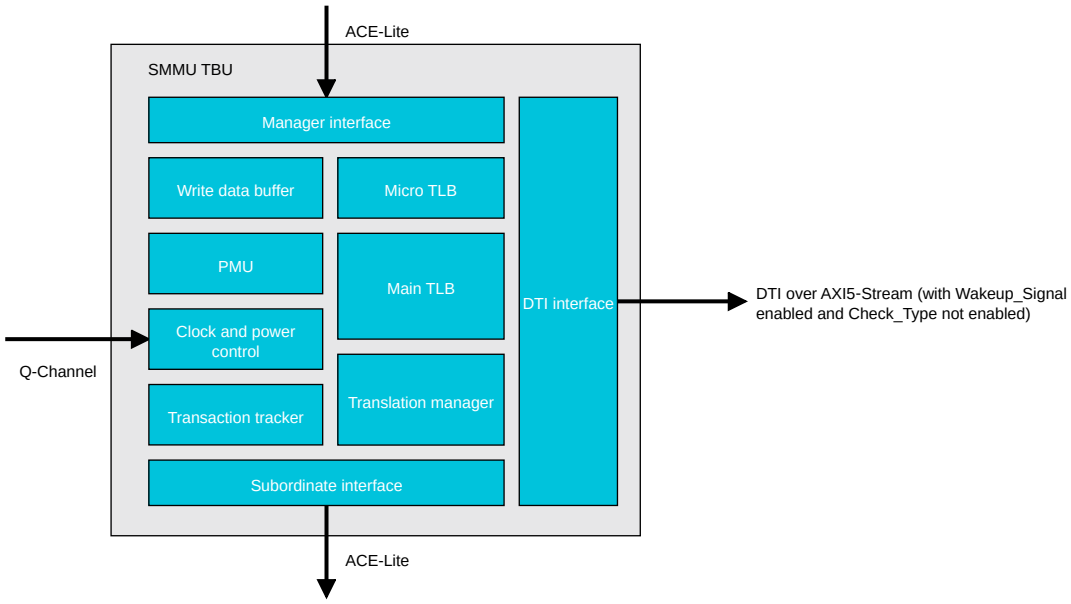
A TBU can be one of the following:

- ACE-Lite TBU
- Local Translation Interface (LTI) TBU

A TBU intercepts transactions and provides the required translation from a Translation Lookaside Buffer (TLB) if possible. If a TLB does not contain the required translation, the TBU requests translations from the TCU, and then caches the translation in one of the TLBs.

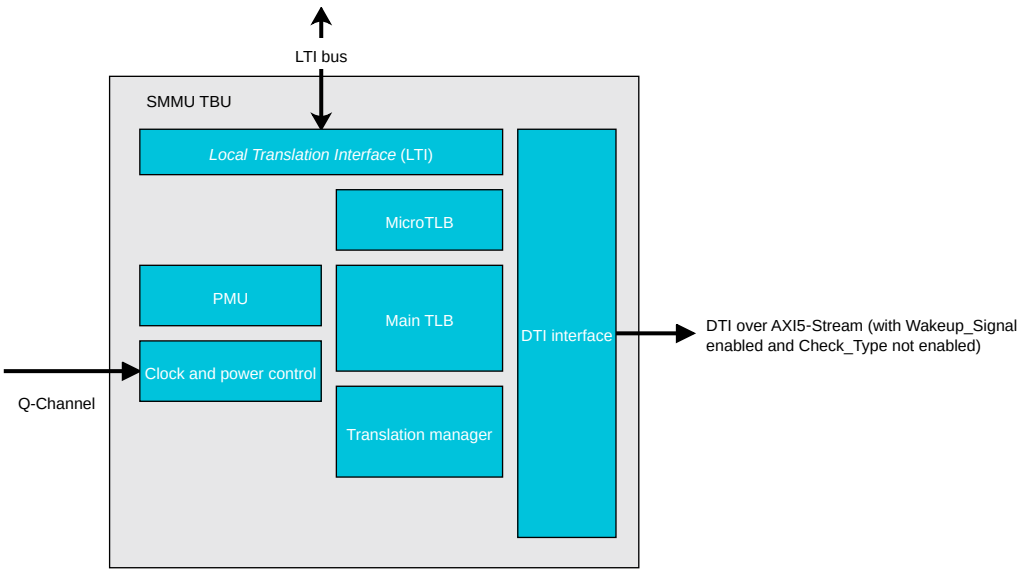
The following figure shows the ACE-Lite TBU. The figure is illustrative of the functions that the block performs but might not match exactly with the block names inside the RTL.

Figure 2-4: MMU S3 ACE-Lite TBU



The following figure shows the LTI TBU. The figure is illustrative of the functions that the block performs but might not match exactly with the block names inside the RTL.

Figure 2-5: MMU S3 LTI TBU



The TBU consists of:

Manager and subordinate interfaces

The TBU manager and subordinate interfaces can contain the following ACE-Lite TBU and LTI TBU interfaces:

ACE-Lite TBU

The ACE-Lite TBU contains the following interfaces:

TBS interface

For receiving untranslated transactions from the requesting manager

TBM interface

For issuing translated transactions to the rest of the system.

LTI TBU

For LTI translation requests and responses.

MicroTLB

The TBU compares incoming transactions with translations that are cached in the MicroTLB before looking in the Main TLB (MTLB). The MicroTLB provides end-to-end translation from an input address to an output address in addition to Granule Protection Check (GPC) information. You can use a tie-off signal to configure the cache replacement policy as either round-robin or Pseudo Least Recently Used (PLRU).

Main TLB

Each TBU includes an optional Main TLB (MTLB) that caches translation table walk entries from:

- Stage 1 translations
- Stage 2 translations
- Stage 1 combined with stage 2 translations

GPC information is also cached in each Main TLB entry.

The MTLB is a set associative cache structure with a configurable number of ways and banks.

If multiple translation sizes are in use, a single transaction might require multiple lookups.

TBU direct indexing enables MMU S3 to manage MTLB entries externally to the TBU. Direct indexing improves the predictability of TBU performance, for bus managers that have real-time performance requirements.

TBU hazarding

If the MicroTLB lookup results in a miss, the transaction checks whether there are any pending transactions from which it can use the translation. This method is called forming a hazard. If the transaction hazards on any pending transactions, then the transaction waits until the response is available for the hazarded transaction and uses that response. The number of unique addresses that form a hazard is limited. The `TBUCFG_HZRD_ENTRIES` parameter controls the number of unique addresses. For information about `TBUCFG_HZRD_ENTRIES`, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124 and 2.7.6 LTI TBU configuration parameters on page 125.

For more information about TBU hazarding, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

Translation manager

The translation manager manages translation requests that are in progress. Each transaction occupies a translation slot until it is propagated downstream through the ACE-Lite TBM, or an LTI translation response is returned. All transactions are hazard-checked to reduce the possibility of duplicate translation requests being sent to the TCU.

Restrictions can exist on the ordering of transactions with different AXI IDs/LTI Order Groups (OGs), if different AXI IDs map to the same ordering group. Transactions with different AXI IDs can be propagated downstream out-of-order.

All transactions with a given AXI ID/LTI OG value must remain ordered. The translation manager propagates such transactions when the translation is ready, provided no other transaction with the same AXI ID/LTI OG is already waiting.

For more information about AXI transaction identifiers, see the *AMBA® AXI Protocol Specification*.

For more information about LTI OGs, see the *AMBA® LTI Protocol Specification*.

Write data buffer

The write data buffer is available in the ACE-Lite TBU only. The optional write data buffer enables write transactions with different AXI IDs to progress through the TBU out-of-order. It reorders the data to match the downstream transaction order.

Performance Monitoring Unit (PMU)

The PMU counts TBU performance-related events.

Clock and power control

The TBU has its own clock and power control, that the Q-Channels provide.

DTI interface

The manager DTI interface uses the DTI protocol over AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled, to enable the TBU to communicate with a subordinate component. For MMU S3, the subordinate component is the TCU. Although you can implement DTI over different transport protocols, the MMU S3 interfaces use AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled.

Transaction tracker

The transaction trackers manage outstanding read and write transactions, permitting invalidation and synchronization to take place without stalling the AXI interfaces.


Note

The transaction tracker is available in the ACE-Lite TBU only.

See also:

- 2.5 Operation on page 54
- 2.5.4 Main TLB direct indexing and main TLB direct partitioning on page 69
- 3.2 SMMU architectural registers on page 131

2.3 Distributed Translation Interface interconnect

The TCU and TBUs use a Distributed Translation Interface (DTI) to communicate. The DTI interconnect enables the DTI interface to use the AXI5-Stream transport protocol, with Wakeup_Signal enabled and Check_Type not enabled.

The DTI interconnect can connect any components that conform to the AXI5-Stream protocol, with Wakeup_Signal enabled and Check_Type not enabled, as the *AMBA® DTI Protocol Specification* defines.

The DTI interconnect contains internal components that are hierarchically composable, that is, they can be connected in different ways to suit your system requirements. For example, within an MMU S3 system, you can use the switch component to combine the DTI interfaces of multiple TBUs into a single DTI interface. You can then connect the combined DTI interface to another DTI interconnect that is closer to the TCU.

The DTI interconnect includes switch, sizer, and register slice components.

Switch

The switch connects multiple DTI TBUs, to a DTI TCU. The switch implements the following parallel networks:

TBU to TCU traffic

Network connecting multiple AXI5-Stream TBUs to a single AXI5-Stream TCU, with Wakeup_Signal enabled and Check_Type not enabled

TCU to TBU traffic

Network connecting a single AXI5-Stream TCU to multiple AXI5-Stream TBUs, with Wakeup_Signal enabled and Check_Type not enabled


Note

The switch does not store any data, and therefore does not require a Q-Channel clock gating interface.

Sizer

The sizer connects channels that have different data widths, enabling different tradeoffs of bandwidth to area. The sizer supports conversion between the supported AXI5-Stream data widths, with Wakeup_Signal enabled and Check_Type not enabled:

- 1 byte
- 4 bytes
- 10 bytes

- 20 bytes
- 24 bytes

The sizer includes a Q-Channel interface to provide clock gating control.

Register slice

Use the register slice to improve timing. The register slice includes a Q-Channel interface to provide clock gating control.

Tip

MMU S3 DTI interconnect components do not include a component to connect different clock and power domains. You can connect DTI interfaces in different clock and power domains by using the Bidirectional AXI5-Stream (BAS) configuration of ADB-400 AMBA® Domain Bridge.

See also 2.5 Operation on page 54.

2.4 Interfaces

MMU S3 includes interfaces for each of the TCU, TBU, and DTI interconnect components.

The DTI interconnect consists of switch, sizer, and register slice components that you can connect separately, and these components therefore have their own interfaces.

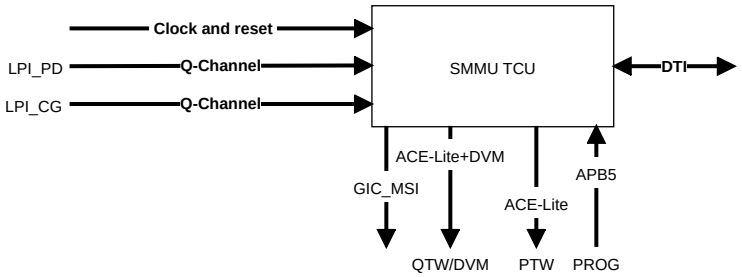
The PMU snapshot interface is common to both TCU and TBU.

2.4.1 TCU interfaces

The MMU S3 TCU includes several requester and completer interfaces.

The following figure shows the TCU interfaces.

Figure 2-6: TCU interfaces



Note

The ACE-Lite PTW interface is optional.

See also:

- 2.5.7 Distributed Virtual Memory messages on page 78
- 2.5.10 Error responses on page 84
- 2.6.2 AMBA implementation on page 94
- A.1.2 TCU QTW/DVM interface signals on page 300

2.4.1.1 TCU Queue and Table Walk/Distributed Virtual Memory interface

The Queue and Table Walk/Distributed Virtual Memory (QTW/DVM) interface is an ACE-Lite +DVM manager interface.

The QTW/DVM interface can issue the following transaction types:

- ReadNoSnoop
- WriteNoSnoop
- ReadOnce
- WriteUnique
- DVM Complete
- AtomicCompare, but only if only a single port is present, that is, TCU x1

The QTW/DVM interface uses the write address transaction ID signal awid_qtw, and the read address transaction ID signal, arid_qtw.

The AXI ID width on this interface is calculated as follows:

(ceil($\log_2$(max(`TCUCFG_PTW_SLOTS`, `TCUCFG_DGW_SLOTS`, 8)) + 3)

The minimum possible TCU ID width value is 6.

For information about the possible values of awid_qtw and arid_qtw, see 2.5.8 TCU transaction handling on page 79.

To support GPT TLBI, the interface provides DVM_9.2 support.

The interface does not issue cache maintenance operations or exclusive accesses.

2.4.1.2 TCU Page Table Walk interface

In the `tcu_x2` variant of the TCU, the Page Table Walk (PTW) interface is present. The dedicated PTW interface is required for high-performance applications where a single AXI interface cannot provide enough bandwidth to handle the combined table walks.

If the SMMU is configured to be in legacy mode, by setting the `TCUCFG_LEGACY_TZ_EN` parameter is set to 1 or setting the legacy_tz_en tie-off signal to 1, the single QTW/DVM interface is usually sufficient. The PTW interface is used only for PTW traffic, including HTTU updates, and DPT traffic, and these are not performed on the QTW/DVM interface. The PTW interface can issue the following transaction types:

- ReadNoSnoop
- ReadOnce
- AtomicCompare

The external ID width is calculated as follows:

ceil($\log_2$(max(`TCUCFG_DGW_SLOTS`, `TCUCFG_PTW_SLOTS`, 8))) + 3.

External ID Width defines the width of the awid_ptw, arid_ptw, bid_ptw, and rid_ptw ports on this interface.

2.4.1.3 TCU PROG interface

The programming, PROG interface is an AMBA APB5 completer interface. It enables software to program the MMU S3 internal registers and read the Performance Monitoring Unit (PMU) registers and the Debug registers.

This interface runs synchronously with the other TCU interfaces.

The applicable address width for this interface is 25 bits.

Transactions are Read-As-Zero, Writes Ignored (RAZ/WI) when any of the following apply:

- An unimplemented register is accessed
- PSTRB[3:0] is not `0b1111` for write transfers
- {PNSE, PPROT[1]} does not encode a PAS which has access to the register being addressed
- In Legacy mode when PNSE == 1

For more information, see:

- A.1.4 TCU programming interface signals on page 306
- *AMBA® APB Protocol Specification*

2.4.1.4 TCU LPI_PD interface

The TCU LPI_PD Q-Channel completer interface manages Low Power Interface (LPI) powerdown (PD) for the TCU.

See also:

- A.1.6 TCU LPI_PD interface signals on page 307
- *AMBA® Low Power Interface Specification*

2.4.1.5 TCU LPI_CG interface

The TCU LPI_CG Q-Channel requester interface enables Low Power Interface (LPI) Clock Gating (CG) for the TCU.

See also:

- A.1.7 TCU LPI_CG interface signals on page 307
- *AMBA® Low Power Interface Specification*

2.4.1.6 TCU DTI interface

The DTI interface manages communication between the TBUs and the TCU, using the DTI protocol. The DTI protocol can be conveyed over different transport layer mediums, including AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled.

The TCU includes a DTI interface, in the same way as each TBU does. To permit bidirectional communication, each DTI interface includes:

Downstream interface
: One AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled, interface

Upstream interface
: One AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled, interface

See also:

- 2.5.1 Distributed Translation Interface on page 54
- A.1.8 TCU DTI interface signals on page 308
- *AMBA® DTI Protocol Specification*
- *AMBA® AXI-Stream Protocol Specification*

2.4.1.7 TCU MSI interface

The Message Signaled Interrupt (MSI) interface provides global, per-context, and performance interrupts. A direct MSI connection to a Generic Interrupt Controller (GIC) is supported, to avoid complex dependencies in the system.

For more information, see A.1.10 TCU Message Signaled Interrupt interface signals on page 311.

2.4.1.8 TCU SYSCO signaling

MMU S3 provides a hardware system coherency interface. The SYSCO requester interface permits the TCU to remove itself from a coherency domain in response to an LPI request.

The SYSCO signals include the syscoreq_qtw and syscoack_qtw handshake signals to enter or exit a coherency domain.

If the sup_btm signal is tied HIGH, the TCU does not assert ACREADY until SYSCOACK has been received and the interface has entered the CONNECTED state.

Note

To prevent a deadlock, the connected interconnect must be capable of driving SYSCOACK without requiring progress on the AC channel.

If the sup_btm signal is tied LOW, the syscoreq_qtw signal is always driven LOW and the syscoack_qtw signal is ignored.

For more information, see A.1.5 TCU SYSCO interface signals on page 307.

2.4.1.9 TCU tie-off signals

The TCU tie-off signals enable you to initialize various operating parameters on exit from reset state.

For more information, see A.1.12 TCU tie-off signals on page 312.

2.4.1.10 TCU ELA observation interface

This Embedded Logic Analyzer (ELA) observation requester interface drives the signal group, signal qualifier, and signal clock enable ELA signals to the on-chip ELA module, if present.

When `TCUCFG_USE_ELA_DEBUG` is 0, these signals are tied to 0. See 2.7.2 TCU debug configuration parameters on page 122.

See also:

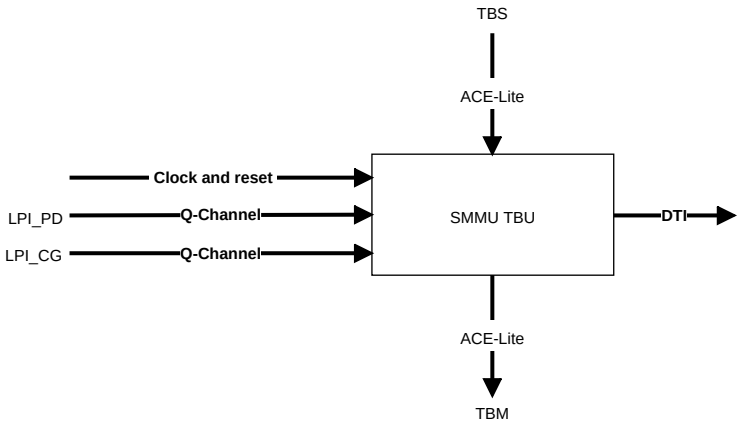
- B.1 TCU observation interfaces on page 346
- *Arm® CoreSight™ ELA-600 Embedded Logic Analyzer Technical Reference Manual*

2.4.2 TBU interfaces

MMU S3 can contain both ACE-Lite and LTI Translation Buffer Units (TBUs). Each MMU S3 TBU includes several requester and completer interfaces.

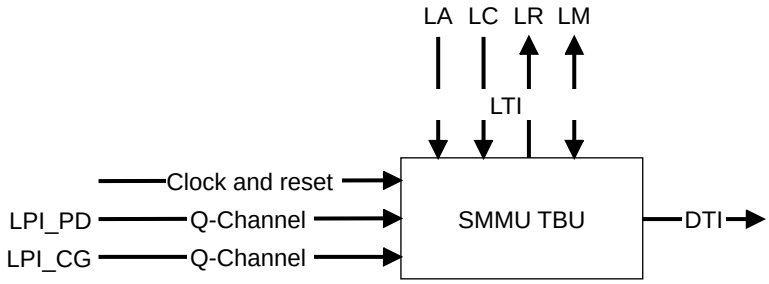
The following figure shows the ACE-Lite TBU interfaces.

Figure 2-7: ACE-Lite TBU interfaces



The following figure shows the LTI TBU interfaces.

Figure 2-8: LTI TBU interfaces



Note

LTI TBUs can have variants with 1, 2, 4, 6, and 8 LTI interfaces. The figure shows a TBU with one LTI interface.

2.4.2.1 ACE-Lite TBU TBS interface

The transaction subordinate interface, TBS, is an ACE5-Lite interface on which the ACE-Lite TBU receives incoming untranslated memory accesses.

This interface supports a 64-bit address width.

The interface implements optional signals to support the following AXI5 extensions:

- Atomic_Transactions

- Cache_Stash_Transactions
- CMO_On_Read, Persist_CMO
- DeAllocation_Transactions
- Exclusive_Accesses
- InvalidateHint_Transaction
- Loopback_Signals
- MEC_Support
- MTE_Support basic
- PBHA_Support
- Poison
- Read_Data_Chunking
- RME_Support
- Shareable_Transactions
- Unique_ID_Support
- Untranslated_Transactions v3
- Wakeup_Signals

The TBS interface supports ACE-Lite Exclusive accesses.

If a transaction is terminated in the TBU, the transaction tracker returns the transaction with the user-defined AXI RUSER and BUSER bits set to 0.

See also:

- 2.5.10 Error responses on page 84
- 2.6.2 AMBA implementation on page 94
- A.2.2 TBU TBS interface signals on page 317

2.4.2.2 ACE-Lite TBU TBM interface

The transaction manager, TBM, interface is an ACE5-Lite interface on which the ACE-Lite TBU sends outgoing translated memory accesses.

The AXI ID of a transaction on this interface is the same as the AXI ID of the corresponding transaction on the transaction subordinate, TBS interface.

This interface supports a 52-bit address width, and `TBUCFG_DATA_WIDTH` defines the data width. See:

- 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124
- 2.7.6 LTI TBU configuration parameters on page 125

This interface can issue read and write transactions until the outstanding transaction limit is reached. MMU S3 provides parameters that permit you to configure:

- The outstanding read transactions limit
- The outstanding write transactions limit
- The total outstanding read and write transactions limit

The interface implements optional signals to support the following AXI5 extensions:

- Atomic_Transactions
- Cache_Stash_Transactions
- CMO_On_Read, Persist_CMO
- DeAllocation_Transactions
- Exclusive_Accesses
- InvalidateHint_Transaction
- Loopback_Signals
- MEC_Support
- MPAM_12_1
- MTE_Support basic
- Ordered Write Observation
- PBHA_Support
- Poison
- Read_Data_Chunking
- Read_Interleaving_Disabled


Note

If terminated transaction responses are not interleaved, the BIU supports the Read_Interleaving_Disabled property.

- RME_Support
- Shareable_Transactions
- Unique_ID_Support
- Untranslated_Transactions v3


Note

The TBM interface does not support the Untranslated_Transactions property. The TBM contains the axmmusecsid and axmmusid signals for backward-compatibility with other SMMU products. These signals are not required for normal operation of MMU S3 and you can ignore them.

- Wakeup_Signals

When receiving an SLVERR or DECERR response to a downstream transaction, the TBM interface propagates the same response to the TBS interface.

See also:

- 2.5.10 Error responses on page 84
- 2.6.2 AMBA implementation on page 94
- A.2.3 TBU TBM interface signals on page 322

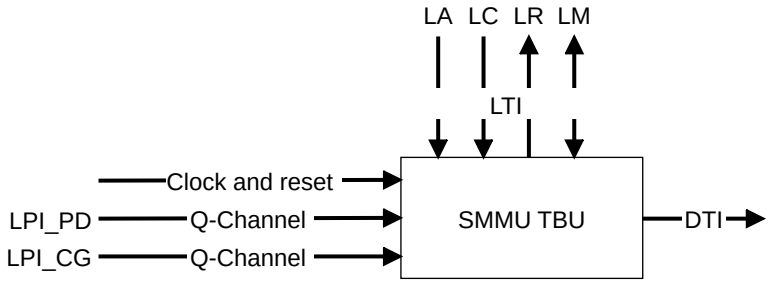
2.4.2.3 LTI TBU LTI interface

There are multiple LTI TBU variants, with 1, 2, 4, 6, and 8 LTI interfaces. Each LTI interface is a complete interface. Most of the parameters that you can use to configure an LTI interface are shared between all of them on the same TBU.

An exception to this sharing is the register slice modes on the LA and LR channels. For more information, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

The following figure shows the LTI TBU interfaces.

Figure 2-9: LTI TBU interfaces



Note

LTI TBUs can have variants with 1, 2, 4, 6, and 8 LTI interfaces. The figure shows a TBU with one LTI interface.

The interface contains the following channels:

LA Request channel. Address and attributes that require translation are sent to the TBU.

LR Response channel. Provides the translated address and attributes to the LTI device.

LC Completion channel. LTI devices must provide information about completion to the TBU.

LM Link Management channel. Contains:

- LMOPENREQ
- LMOPENACK
- LMASKCLOSE
- LMACTIVE

See also:

- 2.6.3 Local Translation Interface implementation on page 108
- 2.7 Configuration parameters and methodology on page 120
- A.2.7 TBU LTI interface signals on page 330
- *AMBA® LTI Protocol Specification*

2.4.2.4 TBU LPI_PD interface

The TBU LPI_PD Q-Channel completer interface manages Low Power Interface (LPI) powerdown (PD) for the TBU.

See also:

- A.2.4 TBU LPI_PD interface signals on page 328
- *AMBA® Low Power Interface Specification*

2.4.2.5 TBU LPI_CG interface

The TBU LPI_CG Q-Channel completer interface enables Low Power Interface (LPI) clock gating (CG) for the TBU.

See also:

- A.2.5 TBU LPI_CG interface signals on page 328
- *AMBA® Low Power Interface Specification*

2.4.2.6 TBU DTI interface

The TBU DTI interface enables the TBU to request translations from the TCU. This interface uses the DTI-TBU protocol for communication between the TBU and the TCU.

The TCU includes a TCU DTI interface and each TBU includes a TBU DTI interface. To permit bidirectional communication, each DTI interface includes:

Downstream interface

One AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled, interface

Upstream interface

One AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled, interface.

See also:

- 2.5.1 Distributed Translation Interface on page 54
- A.2.6 TBU DTI interface signals on page 328
- *AMBA® DTI Protocol Specification*
- *AMBA® AXI-Stream Protocol Specification*

2.4.2.7 TBU interrupt interface

The TBU interrupt interface provides edge-triggered and level-triggered global, per-context, and performance interrupt signals.

See also:

- 2.5.1 Distributed Translation Interface on page 54
- A.2.9 TBU tie-off signals on page 331
- A.2.8 TBU interrupt signals on page 330

2.4.2.8 TBU tie-off signals

The TBU tie-off signals enable you to initialize various operating parameters on exit from reset state.

See also:

- A.2.9 TBU tie-off signals on page 331
- A.2.8 TBU interrupt signals on page 330

2.4.2.9 TBU ELA observation interface

This Embedded Logic Analyzer (ELA) observation requester interface drives the signal group, signal qualifier, and signal clock enable ELA signals to the on-chip ELA module, if present.

When `TBUCFG_USE_ELA_DEBUG` is 0, these signals are tied to 0. See 2.7.7 Common TBU debug configuration parameters on page 126.

See also:

- B.2 ACE-Lite TBU observation interfaces on page 350
- B.3 LTI TBU observation interfaces on page 353
- *Arm® CoreSight™ ELA-600 Embedded Logic Analyzer Technical Reference Manual*

2.4.3 Dual TBU

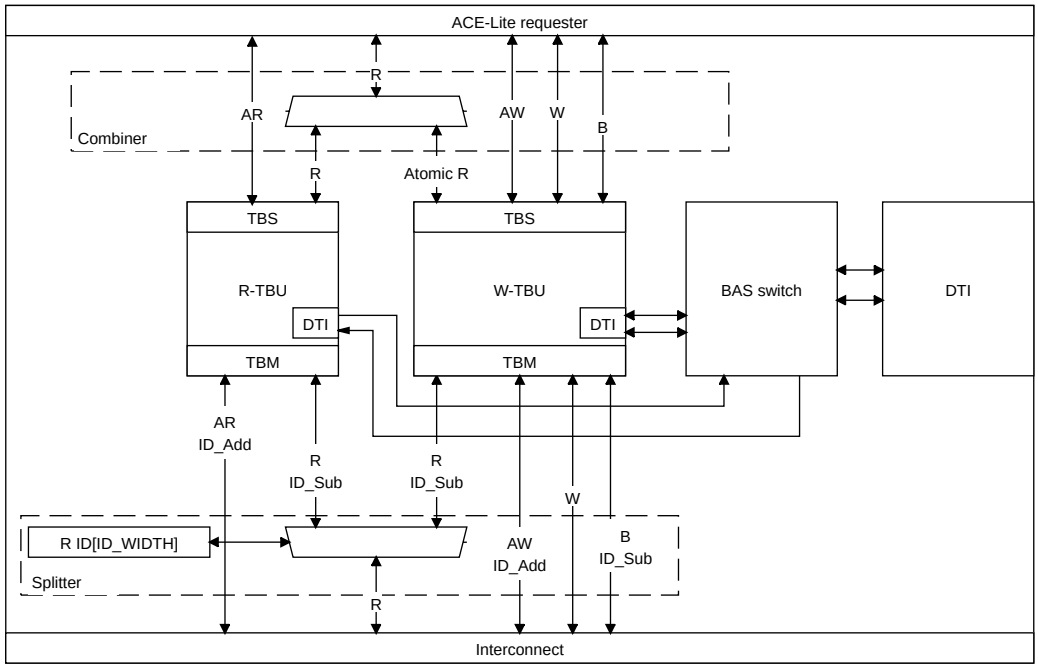
In the ACE-Lite TBU, the translation bandwidth of the TBU is shared between the two AXI address channels, AW and AR meaning that it is not possible to achieve full bandwidth through both channels if both channels are sending an address every cycle, that is, one beat bursts at maximum throughput. The Dual TBU module enables you to achieve greater bandwidth by implementing

separate MMU S3 TBU ACE-Lite instances for read transactions (R-TBU) and write transactions (W-TBU).

Atomic transactions, which can have responses on both the B and R channels, are routed through the W-TBU.

The following figure shows how the ACE-Lite and DTI channels are connected in the Dual TBU.

Figure 2-10: ACE-Lite and DTI channel connections in the Dual TBU



The figure is not intended to convey detailed schematic information. In particular, it omits information on the LPD-500 instances required for clock and power management and the handling of the PMU and RAS signaling.

2.4.3.1 ACE-Lite transactions

The Dual TBU has a single ACE-Lite subordinate interface, and a single ACE-Lite manager interface between these interfaces.

The behavior is as follows:

- AR channel is routed through the R-TBU
- AW, W, and B channels are routed through the W-TBU
- R channel might be routed through the R-TBU or W-TBU, depending on whether the transaction is atomic or not.

The Dual TBU issues translated ACE-Lite transactions downstream as normal. An extra bit is appended to each AxID value to signify whether the transaction is atomic or not.

However, responses on the B channel are routed through the W-TBU, responses on the R channel can be destined for either the R-TBU or the W-TBU. This routing is because AtomicLoad, AtomicSwap, and AtomicCompare transactions return responses on both the B and the R channels.

The R responses for these atomic transactions must therefore be routed through the W-TBU, alongside the corresponding B response. R responses for other transactions must be routed through the R-TBU. The extra bit of the RID value indicates whether a response on the R channel is routed through the W-TBU or the R-TBU.

2.4.3.2 DTI transactions

Both the R-TBU and W-TBU can issue and receive DTI transactions.

Therefore, an MMU S3 BAS Switch arbitrates between the R-TBU and W-TBU to provide a single DTI interface on the MMU S3 Dual TBU.

2.4.3.3 Interrupts and PMU snapshot interface

Both the R-TBU and W-TBU have their own RAS interrupts, PMU interrupts, and PMU snapshot interfaces.

The Dual TBU includes logic to combine these signals to form a single RAS interrupt, PMU interrupt, and PMU snapshot interface on the Dual TBU.

2.4.3.4 Signal descriptions

The following sections list the module interfaces on the block and their purpose.

2.4.3.4.1 Clock and reset

The Dual TBU implements a single clock and a single reset domain. These signals follow the standard Arm® clock and reset guidelines.

The following table shows the clock and reset signals.

Table 2-1: Clock and reset signals

Name	Direction	Description
clk	Input	Global clock. Width is 1-bit.
resetn	Input	Global reset, active-LOW. Width is 1-bit.

Note

Clock and reset might logically form part of the buses used for the interfaces in other sections. Because they are physically shared signals, they are not included in the signal tables.

2.4.3.4.2 LPI PD interface

The Dual TBU implements a single LPI Q-Channel for power control. Because the two TBUs each have an LPI Q-Channel for power control, an LPD-500 is used in expander mode to control the power of the two TBUs from the single LPI Q-Channel interface on the Dual TBU.

The following table shows the interface directions and the agents.

Table 2-2: Interface directions and agents

Interface direction	Agent
Producer	External power controller
Consumer	Dual TBU LPD-500 Q-Channel Expander PD

This interface implements the LPI Q-Channel protocol. See the *AMBA® Low Power Interface Specification*.

The following table shows the LPI PD interface signals.

Table 2-3: LPI PD interface signals

Name	Direction	Description
qreqn	Input	Active-LOW, controller requesting quiescence. Width is 1-bit.
qacceptn	Output	Active-LOW, device accepting quiescence. Width is 1-bit.
qdeny	Output	Active-HIGH, device denying quiescence. Width is 1-bit.
qactive	Output	Active-HIGH, hint that power is required. Width is 1-bit.

2.4.3.4.3 LPI CG interface

The Dual TBU implements a single LPI Q-Channel for clock control. Because the two TBUs each have an LPI Q-Channel for clock control, an LPD-500 is used in expander mode to control the two clocks of the two TBUs from the single LPI Q-Channel interface on the Dual TBU.

The following table shows the interface directions and the agents.

Table 2-4: Interface directions and agents

Interface direction	Agent
Producer	External power controller
Consumer	Dual TBU LPD-500 Q-Channel Expander CG

This interface implements the LPI Q-Channel protocol. See the *AMBA® Low Power Interface Specification*.

The following table shows the LPI CG interface signals.

Table 2-5: LPI CG interface signals

Name	Direction	Description
qreqn	Input	Active-LOW, controller requesting quiescence. Width is 1-bit.
qacceptn	Output	Active-LOW, device accepting quiescence. Width is 1-bit.
qdeny	Output	Active-HIGH, device denying quiescence. Width is 1-bit.
qactive	Output	Active-HIGH, hint that power is required. Width is 1-bit.

2.4.3.4.4 Subordinate interface

The Dual TBU implements an AMBA ACE5-Lite interface which receives both read and write transactions from an upstream client device requiring translation services.

See the *AMBA® AXI Protocol Specification*.

The following table shows the interface directions and the agents.

Table 2-6: Interface directions and agents

Interface direction	Agent
Producer	Upstream client device
Consumer	Dual TBU Combiner

Several ACE-Lite properties are supported as 2.6.2 AMBA implementation on page 94 describes.

2.4.3.4.5 Manager interface

The Dual TBU implements an AMBA ACE5-Lite interface to send the transactions received on the subordinate interface to the downstream memory system after the SMMU has translated the transactions.

See the *AMBA® AXI Protocol Specification*.

The following table shows the interface directions and the agents.

Table 2-7: Interface directions and agents

Interface direction	Agent
Producer	Dual TBU Splitter
Consumer	Downstream memory component

Several ACE-Lite properties are supported as 2.6.2 AMBA implementation on page 94 describes. The following are minor differences to the port listing:

- AWID is (TBUCFG_ID_WIDTH + 1) bits wide
- ARID is (TBUCFG_ID_WIDTH + 1) bits wide
- BID is (TBUCFG_ID_WIDTH + 1) bits wide
- RID is (TBUCFG_ID_WIDTH + 1) bits wide

See A. Signal descriptions on page 300.

2.4.3.4.6 DTI interface

The Dual TBU implements a single DTI interface to enable communication with the MMU S3 TCU. As both TBUs each have a DTI interface, a BAS Switch is used to multiplex between the two DTI interfaces of the TBU.

See the *AMBA® DTI Protocol Specification*.

The following table shows the interface directions and the agents.

Table 2-8: Interface directions and agents

Interface direction	Agent
Producer	Dual TBU BAS Switch
Consumer	TCU

The following table shows the DTI interface signals.

Table 2-9: DTI interface signals

Name	Direction	Description
tvalid_dti_up	Input	UP payload contains valid data. Width is 1-bit.
tready_dti_up	Output	UP consumer is ready to receive payload. Width is 1-bit.
tdata_dti_up	Input	UP interface payload. Width is 192-bit. **Note:** If the TBUCFG_MECID_WIDTHparameter is set to 0 or if the TBUCFG_LEGACY_TZ_EN parameter is set to 1, then width is 160-bit
tkeep_dti_up	Input	Indicates which bytes of tdata contain valid data. Width is 24-bit. If the TBUCFG_MECID_WIDTH parameter is set to 0 or if the TBUCFG_LEGACY_TZ_EN parameter is set to 1, then width is 20-bit
tlast_dti_up	Input	UP payload transfer is complete. Width is 1-bit.
tdest_dti_up	Input	Indicates the TBU for which the transaction is intended. Width is 6-bit.
twakeup_dti_up	Input	DTI TCU is sending, or attempting to send, something. Width is 1-bit.
tvalid_dti_dn	Output	DN payload contains valid data. Width is 1-bit.
tready_dti_dn	Input	DN consumer is ready to receive payload. Width is 1-bit.
tdata_dti_dn	Output	DN interface payload. Width is 160-bit.
tkeep_dti_dn	Output	Indicates which bytes of tdata contain valid data. Width is 20-bit.

Name	Direction	Description
tlast_dti_dn	Output	DN payload transfer is complete. Width is 1-bit.
tid_dti_dn	Output	Indicates the TBU from which the DTI message originates. Width is 6-bit.
twakeup_dti_dn	Output	A TBU wants to, or is sending, a DTI DN transaction. Width is 1-bit.

The Dual TBU contains only two TBUs, but the tdest/tid width on the Dual TBU DTI interface is fixed at 6 bits.

2.4.3.4.7 Interrupts

The Dual TBU implements positive edge-triggered or level-triggered interrupts.

The following table shows the interface directions and the agents.

Table 2-10: Interface directions and agents

Interface direction	Agent
Producer	R-TBU, W-TBU
Consumer	Dual TBU

Both the R-TBU and the W-TBU produce their own set of interrupts. These interrupts are OR'd together to produce the external interrupt signals that the following table shows.

Table 2-11: Interrupt signals

Name	Direction	Description
ras_et_fhi	Output	Edge-triggered fault handling RAS interrupt for a contained error from either R-TBU or W-TBU. Width is 1-bit.
ras_et_eri	Output	Edge-triggered error recovery RAS interrupt for an uncontained error from either R-TBU or W-TBU. Width is 1-bit.
ras_et_cri	Output	Edge-triggered critical error RAS interrupt for an uncontained uncorrected error from either R-TBU or W-TBU. Width is 1-bit.
ras_lt_fhi	Output	Level-triggered fault handling RAS interrupt for a contained error from either R-TBU or W-TBU. Width is 1-bit.
ras_lt_eri	Output	Level-triggered error recovery RAS interrupt for an uncontained error from either R-TBU or W-TBU. Width is 1-bit.
ras_lt_cri	Output	Level-triggered critical error RAS interrupt for an uncontained uncorrected error from either R-TBU or W-TBU. Width is 1-bit.
ras_lt_irpt_v	Output	Level-triggered valid output for connection to System RAS agents. Asserted when the RAS record contains at least one valid error. Connect the ras_lt_irpt_v signal, together with level-triggered versions of RAS interrupts, to a System RAS agent. Width is 1-bit.
pmu_irpt	Output	Edge-triggered PMU counter overflow interrupt from either R-TBU or W-TBU. Width is 1-bit.

Name	Direction	Description
crit_err	Output	Critical error, cannot connect to the TCU because of a deny response, insufficient tokens, or max_tok_trans is smaller than required for the TBU to function. **Note:** max_tok_trans must be >= (2 × `NUM_LTI_PORTS`). **Note:** The crit_err interrupt is level-based to make clock domain crossings easy to manage. Width is 1-bit.

2.4.3.4.8 Tie-offs

The Dual TBU has some configuration options that the static tie-off signals determine. The values of these signals are sampled after reset of the Dual TBU, and so providing the configuration state.

The following table shows the interface directions and the agents.

Table 2-12: Interface directions and agents

Interface direction	Agent
Producer	System integration layer
Consumer	R-TBU, W-TBU

Signals that are appended with *_r* are directed to R-TBU and signals appended with *_w* are directed to W-TBU. Signals without either *_r* or *_w* appended are shared between R-TBU and W-TBU.

The following table shows the tie-off signals.

Table 2-13: Tie-off signals

Dual TBU signal name	Direction	Corresponding R-TBU/W-TBU signal name
ns_sid_high_r	Input	ns_sid_high The width of ns_sid_high_r is (31 - `TBUCFG_SID_WIDTH`)
ns_sid_high_w	Input	ns_sid_high The width of ns_sid_high_w is (31 - `TBUCFG_SID_WIDTH`)
s_sid_high_r	Input	s_sid_high The width of s_sid_high_r is (31 - `TBUCFG_SID_WIDTH`)
s_sid_high_w	Input	s_sid_high The width of s_sid_high_w is (31 - `TBUCFG_SID_WIDTH`)
r_sid_high_r	Input	r_sid_high The width of r_sid_high_r is (31 - `TBUCFG_SID_WIDTH`)

Dual TBU signal name	Direction	Corresponding R-TBU/W-TBU signal name
r_sid_high_w	Input	r_sid_high The width of r_sid_high_w is (31 - TBUCFG_SID_WIDTH)
max_tok_trans_r	Input	max_tok_trans The width of max_tok_trans_r is: $\log_2$(TBUCFG_XLATE_SLOTS_R)
max_tok_trans_w	Input	max_tok_trans The width of max_tok_trans_w is: $\log_2$(TBUCFG_XLATE_SLOTS_W)
sec_override	Input	sec_override. Width is 1-bit.
ecorevnum	Input	ecorevnum. Width is 4-bit.
utlb_roundrobin_r	Input	utlb_roundrobin. Width is 1-bit.
utlb_roundrobin_w	Input	utlb_roundrobin. Width is 1-bit.
pcie_mode	Input	pcie_mode. Width is 1-bit.
poison_support	Input	poison_support. Width is 1-bit.
legacy_tz_en	Input	legacy_tz_en. Width is 1-bit.
same_power_domain	Input	same_power_domain **Note:** If the same_power_domain signal is HIGH, the TBU drives the TBU_DTI_CONDIS_REQ.SPD field HIGH when it attempts to connect to the TCU and causes the TCU to ignore the connection status of this TBU when calculating its PD_QACTIVE output. Tie the same_power_domain signal HIGH when the TBU is in the same power domain as the TCU to which it is connecting to simplify the connection of the power-management logic. Width is 1-bit.

See A.2.9 TBU tie-off signals on page 331.

2.4.3.4.9 PMU snapshot interface

The Dual TBU implements a single PMU snapshot interface. The Dual TBU includes logic to merge together the separate PMU snapshot interfaces of the R-TBU and W-TBU.

The following table shows the interface directions and the agents.

Table 2-14: Interface directions and agents

Interface direction	Agent
Producer	System integration layer
Consumer	R-TBU, W-TBU

The following table shows the PMU snapshot interface signals.

Table 2-15: PMU snapshot interface signals

Name	Direction	Description
pmusnapshot_req	Input	Request to both R-TBU and W-TBU to capture PMU data. Width is 1-bit.
pmusnapshot_ack	Input	Acknowledge that either both R-TBU and W-TBU have captured PMU data or that one of them has. Width is 1-bit.

2.4.3.4.10 DFT interface

The Dual TBU implements a single Design for Test (DFT) interface. No glue logic is required on this interface despite both TBUs having their own interfaces for DFT. Glue logic is not required because the signals used are the same, and when in DFT mode, the test controller tests all logic simultaneously.

The following table shows the interface directions and the agents.

Table 2-16: Interface directions and agents

Interface direction	Agent
Producer	External Test Controller
Consumer	R-TBU and W-TBU, LPD-500 PD, LPD-500 CG

The following table shows the DFT interface signals.

Table 2-17: DFT interface signals

Name	Direction	Description
dftcgen	Input	Scan shift enable, forces on the clock grids during scan shift. Width is 1-bit.
dftrstdisable	Input	Disables internal synchronized reset during scan shift. Width is 1-bit
dftramhold	Input	Disable the RAM chip select during scan testing. Width is 1-bit

2.4.3.4.11 MBIST interface

The Dual TBU R-TBU and W-TBU contain RAM instances. A *Memory Built-In Self Test* (MBIST) interface is provided for in-silicon testing of these RAMs.

The following table shows the interface directions and the agents.

Table 2-18: Interface directions and agents

Interface direction	Agent
Producer	External MBIST controller
Consumer	R-TBU, W-TBU

The following table shows the MBIST interface signals.

Table 2-19: MBIST interface signals

Name	Direction	Description
nMBISTRESET	Input	Resets all MBIST logic. Width is 1-bit
MBISTREQ	Input	Requests MBIST testing. Width is 1-bit

2.4.3.4.12 ELA interface

The Dual TBU separate ELA interfaces for R-TBU and W-TBU and then, unlike other interfaces in this block, has separate external interfaces for R-TBU and W-TBU.

The following table shows the interface directions and the agents.

Table 2-20: Interface directions and agents

Interface direction	Agent
Producer	External system
Consumer	R-TBU, W-TBU

The following table shows the ELA interface signals.

Table 2-21: ELA interface signals

Name	Direction	Description
ela_enable_wtbu	Input	Width is 1-bit
signalgrp0_wtbu	Output	Width is 128-bit
sigqual0_wtbu	Output	Width is 4-bit
sigclken0_wtbu	Output	Width is 1-bit
signalgrp1_wtbu	Output	Width is 128-bit
sigqual1_wtbu	Output	Width is 4-bit
sigclken1_wtbu	Output	Width is 1-bit
signalgrp2_wtbu	Output	Width is 128-bit
sigqual2_wtbu	Output	Width is 4-bit
sigclken2_wtbu	Output	Width is 1-bit
signalgrp3_wtbu	Output	Width is 128-bit
sigqual3_wtbu	Output	Width is 4-bit
sigclken3_wtbu	Output	Width is 1-bit
signalgrp4_wtbu	Output	Width is 128-bit
sigqual4_wtbu	Output	Width is 4-bit
sigclken4_wtbu	Output	Width is 1-bit
ela_enable_rtbu	Input	Width is 1-bit
signalgrp0_rtbu	Output	Width is 128-bit
sigqual0_rtbu	Output	Width is 4-bit
sigclken0_rtbu	Output	Width is 1-bit
signalgrp1_rtbu	Output	Width is 128-bit

Name	Direction	Description
sigqual1_rtbu	Output	Width is 4-bit
sigclken1_rtbu	Output	Width is 1-bit
signalgrp2_rtbu	Output	Width is 128-bit
sigqual2_rtbu	Output	Width is 4-bit
sigclken2_rtbu	Output	Width is 1-bit
signalgrp3_rtbu	Output	Width is 128-bit
sigqual3_rtbu	Output	Width is 4-bit
sigclken3_rtbu	Output	Width is 1-bit
signalgrp4_rtbu	Output	Width is 128-bit
sigqual4_rtbu	Output	Width is 4-bit
sigclken4_rtbu	Output	Width is 1-bit

2.4.4 DTI interconnect interfaces

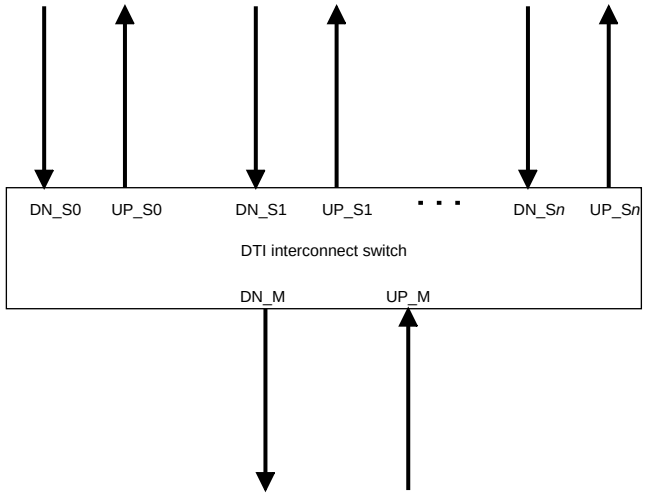
The DTI interconnect includes interfaces for each of the switch, sizer, and register slice components.

2.4.4.1 DTI interconnect switch interfaces

The DTI interconnect switch component includes dedicated interfaces.

The following figure shows the DTI interconnect switch interfaces.

Figure 2-11: DTI interconnect switch interfaces



The following table provides more information about the switch interfaces.

Table 2-22: DTI interconnect switch interfaces

Interface	Data flow	Side	Description
DN_Sn	Downstream	TBU side	TCU downstream interface. One DN_Sn interface is present for each TBU interface.
UP_Sn	Upstream	TBU side	TCU upstream interface. One UP_Sn interface is present for each TBU interface.
DN_M	Downstream	TCU side	TBU downstream interface
UP_M	Upstream	TCU side	TBU upstream interface

Note

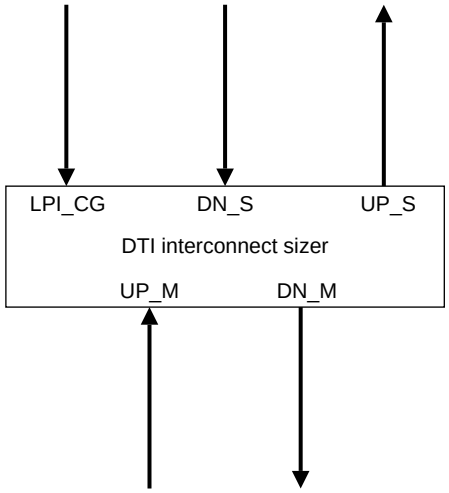
The interconnect switch does not store any data, and therefore does not require a Q-Channel clock gating interface.

2.4.4.2 DTI interconnect sizer interfaces

The DTI interconnect sizer component includes dedicated interfaces.

The following figure shows the DTI interconnect sizer interfaces.

Figure 2-12: DTI interconnect sizer interfaces



The following table provides more information about the sizer interfaces.

Table 2-23: DTI interconnect sizer interfaces

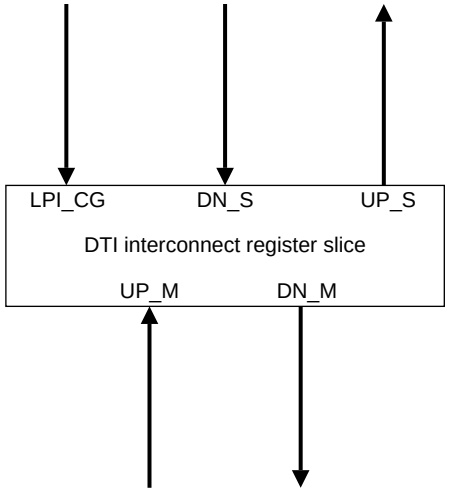
Interface	Data flow	Facing	Description
LPI_CG	-	Facing the clock controller	Clock gating interface
DN_S	Downstream	TBU side	TCU downstream interface
UP_S	Upstream	TBU side	TCU upstream interface
DN_M	Downstream	TCU side	TBU downstream interface
UP_M	Upstream	TCU side	TBU upstream interface

2.4.4.3 DTI interconnect register slice interfaces

The DTI interconnect register slice component includes dedicated interfaces.

The following figure shows the DTI interconnect register slice interfaces.

Figure 2-13: DTI interconnect register slice interfaces



The following table provides more information about the register slice interfaces.

Table 2-24: DTI interconnect register slice interfaces

Interface	Data flow	Facing	Description
LPI_CG	-	Facing the clock controller	Clock gating interface
DN_S	Downstream	TBU side	TCU downstream interface
UP_S	Upstream	TBU side	TCU upstream interface
DN_M	Downstream	TCU side	TBU downstream interface
UP_M	Upstream	TCU side	TBU upstream interface

2.5 Operation

MMU S3 contains components that operate together.

2.5.1 Distributed Translation Interface

In an MMU S3-based system, the AMBA® Distributed Translation Interface (DTI) protocol defines the standard for communicating with a TCU.

The AMBA® DTI protocol includes the following:

DTI-TBU protocol

For communication between a TBU and a TCU.

DTI-ATS protocol

For communication between a PCIe Root Port and a TCU.

The DTI protocol is a point-to-point protocol. Each channel consists of the following:

- Link
- DTI-TBU
- DTI-TCU

The DTI requesters in the respective protocols are as follows:

- The TBU, in the DTI-TBU protocol
- The PCIe Root Port, in the DTI-ATS protocol

The DTI completer, in both DTI-TBU and DTI-ATS is the TCU.

DTI requesters, which are TBUs, and completers, which are TCUs, communicate using defined DTI messages. The DTI protocol defines the following message groups:

- Page request.
- Register access.
- Translation request.
- Connection and disconnection.
- Invalidation and synchronization.

A DTI requester, a TBU, uses a DTI_TBU_CONDIS_REQ or a DTI_ATS_CONDIS_REQ message to initiate a connection handshake. If the requester (TBU) provides a TID value that is greater than the maximum supported TID that `TCUCFG_NUM_TBU` defines, the completer, the TCU, sends a Connect Deny message.

For the TBU, the max_tok_trans signal defines the number of translation tokens that the TBU requests. The TBU must request a minimum of two translation tokens per LTI port.

The TBU uses the TOK_INV_GNT field to grant invalidation tokens. The TBU grants only one invalidation token, and the TCU can only issue one invalidate message at a time.

The DTI_TBU_CONDIS_REQ message initiates a TBU connection or disconnection handshake. The TBU uses this message to connect to the TCU. This handshake process also applies to ATS connections. During connection, the TOK_TRANS_REQ field of this message specifies the number of requested translation tokens.

A translation request to the TCU where StreamID ≥ 2^{32} results in a fault and an SMMUv3 C_BAD_STREAMID event. If the TBU receives an invalidation request where StreamID ≥ 2^{32}, any comparisons with a StreamID value fail. No TLB entries are invalidated, but other effects, that do not consider the supplied StreamID, occur as normal.

DTI connections the TCU accepts

The MMU S3 TCU supports specific DTI-TBU and DTI-ATS versions. The DTI-TBU versions supported are as follows:

- DTI-TBUv4
- DTI-ATS. MMU S3 supports the following versions of DTI-ATS:
  - DTI-ATSv1
  - DTI-ATSv2
  - DTI-ATSv3
  - DTI-ATSv4

A particular TCU can support a mixture of the DTI versions that the list above shows.

Versions lower than the ones listed above are not supported, and DTI_{TBU|ATS}_CONDIS_REQs are denied.

DTI-TBU

If a DTI-TBU with a version greater than v4 sends a DTI_TBU_CONDIS_REQ, and all other connection conditions are satisfied, the connection request is accepted as a DTI-TBUv4 connection.

Legacy mode

If the TCU is in legacy mode, when the `TCUCFG_LEGACY_TZ_EN` parameter is set to 1 or the legacy_tz_en signal is set to 1, the TBU connection request is denied if Granule Protection Checks are requested, that is, DTI_TBU_CONDIS_REQ.STAGES != `2'b00`.

Non-legacy mode

If the TCU is not in legacy mode, when the `TCUCFG_LEGACY_TZ_EN` parameter is set to 0 and the legacy_tz_en signal is set to 0, the TBU connection request is denied if Granule Protection Checks are not requested, that is, DTI_TBU_CONDIS_REQ.STAGES == `2'b00`.

DTI-ATS

If a DTI-ATS with a version greater than v4 sends a DTI_ATS_CONDIS_REQ, and all other connection conditions are satisfied, the connection request is accepted as a DTI-ATSv4 connection.


Note

- The TBU never generates translation requests with StreamID ≥ 2^{32}
- The TCU never generates invalidation requests with StreamID ≥ 2^{32}

For more information, see the *AMBA® DTI Protocol Specification*.

2.5.2 Performance Monitoring Unit

MMU S3 includes a PMU for the TCU and a PMU for each TBU. The PMU events and counters indicate the runtime performance of MMU S3.

MMU S3 includes logic to gather various statistics on the operation of MMU S3 during runtime, using events and counters. These events, which the SMMUv3 architecture defines, provide useful information about the behavior of MMU S3. You can use this information when debugging or profiling traffic.

2.5.2.1 SMMUv3 architectural performance events

Both the TCU and the TBU implement performance events that the SMMUv3 Performance Monitor extension defines.

The SMMUv3 architecture enables the filtering of events so that only events triggered by transactions with a specific StreamID (SID) and/or security state are counted. The event filtering includes the following:

- Speculative transactions and translations
- Transactions and translations that result in a terminated transaction or a translation fault

The events for which this type of filtering can occur are marked as 'Yes' in the 'SID filterable' columns in the following tables. For any events that are not SID filterable, but might reveal information about a security state, they are considered to be 'non-attributable' events by the SMMUv3 architecture. These events are observed only if SMMU_PMCG_SCR and SMMU_PMCG_ROOTCR are set to count these events.

The following table shows the architecturally-defined MMU S3 TCU performance events.

The following table shows the architecturally-defined MMU S3 TCU performance events.

Table 2-25: SMMUv3 performance events for the TCU

Event	Event ID	SID filterable	Non-attributable event	Description
Clock cycle	0x0	No	No	Counts clock cycles. Cycles where the clock is gated after a clock Q-Channel handshake are not counted.
Transaction	0x1	Yes	-	Counts translation requests that originate from a DTI-TBU or DTI-ATS requester.
TLB miss caused by incoming transaction or translation request	0x2	Yes	-	Counts translation requests where the translation walks new translation table entries.
Configuration cache miss caused by transaction or translation request	0x3	Yes	-	Counts translation requests where the translation walks new configuration table entries.
Translation table walk access	0x4	Yes	-	Counts translation table walk accesses.
Configuration structure access	0x5	Yes	-	Counts configuration table walk accesses.
PCIe ATS Translation Request received	0x6	Yes	-	Counts translation requests that originate from a DTI-ATS requester.

The following table shows the architecturally defined MMU S3 TBU performance events.

Table 2-26: SMMUv3 performance events for the TBU

Event	Event ID	SID filterable	Non-attributable event	Description
Clock cycle	0x0	No	No	Counts clock cycles. Cycles where the clock is gated after a clock Q-Channel handshake are not counted.
Transaction	0x1	Yes	-	Counts transactions that are issued on the TBM interface
TLB miss caused by incoming transaction or translation request	0x2	Yes	-	Counts speculative and non-speculative translation requests that are issued to the TCU
PCIe ATS Translation Request received	0x7	Yes	-	Counts ATS-translated transactions that are issued on the TBM interface

For more information, see the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

2.5.2.2 SMMU TCU events

You can configure the MMU S3 PMU to monitor a range of **IMPLEMENTATION DEFINED** TCU performance events.

The SMMUv3 architecture enables the filtering of events so that only events triggered by transactions with a specific StreamID (SID) and/or security state are counted. The event filtering includes the following:

- Speculative transactions and translations
- Transactions and translations that result in a terminated transaction or a translation fault

The events for which this type of filtering can occur are marked as 'Yes' in the 'SID filterable' columns in the following tables. For any events that are not SID filterable, but might reveal information about a security state, they are considered to be 'non-attributable' events by the SMMUv3 architecture. These events are observed only if SMMU_PMCG_SCR and SMMU_PMCG_ROOTCR are set to count these events.

The following table shows the architecturally-defined MMU S3 TCU performance events.

Table 2-27: SMMUv3 performance events for the TCU

Event	Event ID	SID filterable	Non-attributable event	Description
S1L0WC lookup	0x80	Yes	-	Counts translation requests that access S1L0WC entries in the walk cache.
S1L0WC miss	0x81	Yes	-	Counts translation requests that access S1L0WC entries in the walk cache and do not result in a hit.
S1L1WC lookup	0x82	Yes	-	Counts translation requests that access S1L1WC entries in the walk cache.

Event	Event ID	SID filterable	Non-attributable event	Description
S1L1WC miss	0x83	Yes	-	Counts translation requests that access S1L1WC entries in the walk cache and do not result in a hit.
S1L2WC lookup	0x84	Yes	-	Counts translation requests that access S1L2WC entries in the walk cache.
S1L2WC miss	0x85	Yes	-	Counts translation requests that access S1L2WC entries in the walk cache and do not result in a hit.
S1L3WC lookup	0x86	Yes	-	Counts translation requests that access S1L3WC entries in the walk cache.
S1L3WC miss	0x87	Yes	-	Counts translation requests that access S1L3WC entries in the walk cache and do not result in a hit.
S2L0WC lookup	0x88	Yes	-	Counts translation requests that access S2L0WC entries in the walk cache.
S2L0WC miss	0x89	Yes	-	Counts translation requests that access S2L0WC entries in the walk cache and do not result in a hit.
S2L1WC lookup	0x8A	Yes	-	Counts translation requests that access S2L1WC entries in the walk cache.
S2L1WC miss	0x8B	Yes	-	Counts translation requests that access S2L1WC entries in the walk cache and do not result in a hit.
S2L2WC lookup	0x8C	Yes	-	Counts translation requests that access S2L2WC entries in the walk cache.
S2L2WC miss	0x8D	Yes	-	Counts translation requests that access S2L2WC entries in the walk cache and do not result in a hit.
S2L3WC lookup	0x8E	Yes	-	Counts translation requests that access S2L3WC entries in the walk cache.
S2L3WC miss	0x8F	Yes	-	Counts translation requests that access S2L3WC entries in the walk cache and do not result in a hit.
WC read	0x90	Yes	-	Counts reads from the walk cache RAMs, excluding reads that invalidation requests cause. **Note:** A single walk cache lookup might result in multiple RAM reads. This behavior permits contiguous entries to be located.
Buffered translation	0x91	Yes	-	Counts translations that are written to the translation request buffer because either all the configuration table walk slots or all the page table walk slots are occupied.
CC lookup	0x92	Yes	-	Counts lookups into the configuration cache.
CC read	0x93	Yes	-	Counts reads from the Configuration Cache (CC) RAMs, excluding reads that invalidation requests cause. **Note:** A single cache lookup might result in multiple RAM reads. This behavior permits contiguous entries to be located.
CC miss	0x94	Yes	-	Counts lookups into the CC that do not result in a hit.

Event	Event ID	SID filterable	Non-attributable event	Description
WMB is blocked from sending a request to WCB because of backpressure	0x95	No	Yes	Walk Management Block (WMB) is blocked from sending a request to Walk Cache Block (WCB) because of backpressure.
WC loop slots full	0x96	No	Yes	Walk cache loop slots are full.
WC update slots full	0x97	No	Yes	Walk cache RAM update slots are full.
CMB is blocked from sending a request to CCB because of back pressure	0x98	No	Yes	Configuration Management Block (CMB) is blocked from sending a request to the CCB because of back pressure.
PTW slots full	0x99	No	Yes	Page table walk slots are full.
CTW slots full	0x9A	No	Yes	Configuration table walk slots are full.
CC lookup slots full	0x9B	No	Yes	Lookup slots in the CC are full.
CC update slots full	0x9C	No	Yes	Update slots in the CC are full.
Speculative translation	0xA0	Yes	-	Counts translation requests that are marked as speculative.
STE prefetch	0xA1	Yes	-	Stream Table Entry prefetch has occurred.
HTTU	0xA2	Yes	-	The Hardware Translation Table Update (HTTU) has occurred.
AGW GPC Cache (GC) lookup	0xB0	Yes	-	AXI Granule Protection Check (GPC) Wrapper (AGW) GC lookup.
AGW GC miss	0xB1	Yes	-	AGW GC miss.
AGW GC read	0xB2	Yes	-	AGW GC read.
AGW GMB request	0xB5	Yes	-	AGW GPC Management Block (GMB) request.
AGW is blocked from sending a request to GCB because of back pressure	0xB6	No	Yes	AGW is blocked from sending a request to GPT cache block because of back pressure.
AGW GPC Cache Block (GCB) lookup slots full	0xB7	No	Yes	Lookup slots in the AGW GPT cache block are full.
AGW GCB update slots full	0xB8	No	Yes	Update slots in the AGW GPT cache block are full.
DGW GC lookup	0xC0	Yes	-	DTI GPT Wrapper (DGW) GC lookup.
DGW GC miss	0xC1	Yes	-	DGW GC miss.
DGW GC read	0xC2	Yes	-	DGW GC read.
DGW GMB request	0xC5	Yes	-	DGW GMB request.
DGW is blocked from sending a request to GCB because of back pressure	0xC6	No	Yes	DGW is blocked from sending a request to GCB because of back pressure.
DGW GC lookup slots full	0xC7	No	Yes	DGW GC lookup slots full.
DGW GC update slots full	0xC8	No	Yes	DGW GC update slots full.
DGW out of credits	0xC9	No	Yes	All DGW slots are reserved for use.
Primary hazard hit	0xD0	No	No	Primary hazard hit.
Hazard created, new primary hazard entered into primary hazard cache	0xD1	No	No	Hazard created, new primary hazard entered into primary hazard cache.
Primary hazard cache full, all primary hazard slots in use	0xD2	No	No	Primary hazard cache full, all primary hazard slots in use.
Hazard hit in secondary HZU	0xD3	No	No	Hazard hit in secondary HZU.

Event	Event ID	SID filterable	Non-attributable event	Description
Hazard created, new hazard entered into secondary hazard cache	0xD4	No	No	Hazard created, new hazard entered into secondary hazard cache.
Secondary hazard cache full, all secondary hazard slots in use	0xD5	No	No	Secondary hazard cache full, all secondary hazard slots in use.
As event 2 but only counts page table misses	0xE2	Yes	-	As event 2 but only counts page table misses. Page table misses related to stage 2 for CD are not counted.
As event 4 but only counts page table walks, not GPC	0xE4	Yes	-	As event 4 but only counts page table walks, not GPC.
HIT_MAP_1_4	0xE8	Yes	-	CCB update received per category, with SID cont ≤ 2.
HIT_MAP_8_128	0xE9	Yes	-	CCB update received per category, with SID cont >2 and ≤ 7.
HIT_MAP_256_MAX	0xEA	Yes	-	CCB update received per category, with SID cont ≥ 8.
HIT_MAP_04K_L0_C0_S1	0xEB	Yes	-	WCB update received per category, 04K_L0_C0_S1 update.
HIT_MAP_04K_L1_C0_S1	0xEC	Yes	-	WCB update received per category, 04K_L1_C0_S1 update.
HIT_MAP_04K_L1_C1_S1	0xED	Yes	-	WCB update received per category, 04K_L1_C1_S1 update.
HIT_MAP_04K_L2_C0_S1	0xEE	Yes	-	WCB update received per category, 04K_L2_C0_S1 update.
HIT_MAP_04K_L2_C1_S1	0xEF	Yes	-	WCB update received per category, 04K_L2_C1_S1 update.
HIT_MAP_04K_L3_C0_S1	0xF0	Yes	-	WCB Update received per category, 04K_L3_C0_S1 update.
HIT_MAP_04K_L3_C1_S1	0xF1	Yes	-	WCB update received per category, 04K_L3_C1_S1 update.
HIT_MAP_16K_L0_C0_S1	0xF2	Yes	-	WCB update received per category, 16K_L0_C0_S1 update.
HIT_MAP_16K_L1_C0_S1	0xF3	Yes	-	WCB update received per category, 16K_L1_C0_S1 update.
HIT_MAP_16K_L2_C0_S1	0xF4	Yes	-	WCB update received per category, 16K_L2_C0_S1 update.
HIT_MAP_16K_L2_C1_S1	0xF5	Yes	-	WCB update received per category, 16K_L2_C1_S1 update.
HIT_MAP_16K_L3_C0_S1	0xF6	Yes	-	WCB update received per category, 16K_L3_C0_S1 update.
HIT_MAP_16K_L3_C1_S1	0xF7	Yes	-	WCB update received per category, 16K_L3_C1_S1 update.
HIT_MAP_64K_L1_C0_S1	0xF8	Yes	-	WCB update received per category, 64K_L1_C0_S1 update.
HIT_MAP_64K_L2_C0_S1	0xF9	Yes	-	WCB update received per category, 64K_L2_C0_S1 update.
HIT_MAP_64K_L2_C1_S1	0xFA	Yes	-	WCB update received per category, 64K_L2_C1_S1 update.
HIT_MAP_64K_L3_C0_S1	0xFB	Yes	-	WCB update received per category, 64K_L3_C0_S1 update.
HIT_MAP_64K_L3_C1_S1	0xFC	Yes	-	WCB update received per category, 64K_L3_C1_S1 update.
HIT_MAP_04K_L0_C0_S2	0xFD	Yes	-	WCB update received per category, 04K_L0_C0_S2 update.
HIT_MAP_04K_L1_C0_S2	0xFE	Yes	-	WCB update received per category, 04K_L1_C0_S2 update.
HIT_MAP_04K_L1_C1_S2	0xFF	Yes	-	WCB update received per category, 04K_L1_C1_S2 update.
HIT_MAP_04K_L2_C0_S2	0x100	Yes	-	WCB update received per category, 04K_L2_C0_S2 update.
HIT_MAP_04K_L2_C1_S2	0x101	Yes	-	WCB update received per category, 04K_L2_C1_S2 update.
HIT_MAP_04K_L3_C0_S2	0x102	Yes	-	WCB update received per category, 04K_L3_C0_S2 update.
HIT_MAP_04K_L3_C1_S2	0x103	Yes	-	WCB update received per category, 04K_L3_C1_S2 update.
HIT_MAP_16K_L0_C0_S2	0x104	Yes	-	WCB update received per category, 16K_L0_C0_S2 update.
HIT_MAP_16K_L1_C0_S2	0x105	Yes	-	WCB update received per category, 16K_L1_C0_S2 update.
HIT_MAP_16K_L2_C0_S2	0x106	Yes	-	WCB update received per category, 16K_L2_C0_S2 update.
HIT_MAP_16K_L2_C1_S2	0x107	Yes	-	WCB update received per category, 16K_L2_C1_S2 update.

Event	Event ID	SID filterable	Non-attributable event	Description
HIT_MAP_16K_L3_C0_S2	0x108	Yes	-	WCB update received per category, 16K_L3_C0_S2 update.
HIT_MAP_16K_L3_C1_S2	0x109	Yes	-	WCB update received per category, 16K_L3_C1_S2 update.
HIT_MAP_64K_L1_C0_S2	0x10A	Yes	-	WCB update received per category, 64K_L1_C0_S2 update.
HIT_MAP_64K_L2_C0_S2	0x10B	Yes	-	WCB update received per category, 64K_L2_C0_S2 update.
HIT_MAP_64K_L2_C1_S2	0x10C	Yes	-	WCB update received per category, 64K_L2_C1_S2 update.
HIT_MAP_64K_L3_C0_S2	0x10D	Yes	-	WCB update received per category, 64K_L3_C0_S2 update.
HIT_MAP_64K_L3_C1_S2	0x10E	Yes	-	WCB update received per category, 64K_L3_C1_S2 update.
Active Invalidate Mode Active TMU DGW GMB	0x111	No	No	Counts per cycle the instance is in active invalidate mode.
Active Invalidate Mode Active TMU AGW GMB	0x112	No	No	Counts per cycle the instance is in active invalidate mode.
Active Invalidate Mode Active TMU WMB	0x113	No	No	Counts per cycle the instance is in active invalidate mode.
DGW_HM_UPDATE_L1_GRAN	0x116	Yes	-	One GCB update received per category, L1_GRAN update.
DGW_HM_UPDATE_L1_2M	0x117	Yes	-	One GCB update received per category, L1_2M update.
DGW_HM_UPDATE_L1_32M	0x118	Yes	-	One GCB update received per category, L1 32M update.
DGW_HM_UPDATE_L1_512M	0x119	Yes	-	One GCB update received per category, L1 512M update.
DGW_HM_UPDATE_L0_CONT	0x11A	Yes	-	One GCB update received per category; L0 leaf update.
DGW_HM_UPDATE_L0_TBL	0x11B	Yes	-	One GCB update received per category, L0 table update.
AGW_HM_UPDATE_L1_GRAN	0x11E	Yes	-	One GCB update received per category, L1_GRAN update.
AGW_HM_UPDATE_L1_2M	0x11F	Yes	-	One GCB update received per category, L1_2M update.
AGW_HM_UPDATE_L1_32M	0x120	Yes	-	One GCB update received per category, L1 32M update.
AGW_HM_UPDATE_L1_512M	0x121	Yes	-	One GCB update received per category, L1 512M update.
AGW_HM_UPDATE_L0_CONT	0x122	Yes	-	One GCB update received per category L0 leaf update.
AGW_HM_UPDATE_L0_TBL	0x123	Yes	-	One GCB update received per category, L0 table update.
DCB_Lookup_slots_full	0x124	No	Yes	DCB lookup slots are full.
DCB_Update_slots_full	0x125	No	Yes	DCB update slots are full.
DCB_read	0x126	Yes	-	DCB read.
DCB_lookup	0x127	Yes	-	Lookup in the DCB.
DCB_miss	0x128	Yes	-	Miss in the DCB.
DCB_UPDATE_HIT_MAP_DPTGS	0x129	Yes	-	One DCB update received per category, HIT_MAP_DPTGS update.
DCB_UPDATE_HIT_MAP_64K	0x12A	Yes	-	One DCB update received per category, HIT_MAP_64K update.
DCB_UPDATE_HIT_MAP_2M	0x12B	Yes	-	One DCB update received per category, HIT_MAP_2M update.
DCB_UPDATE_HIT_MAP_32M	0x12C	Yes	-	One DCB update received per category, HIT_MAP_32M update.
DCB_UPDATE_HIT_MAP_512M	0x12D	Yes	-	One DCB update received per category, HIT_MAP_512M update.
DCB_UPDATE_HIT_MAP_1G	0x12E	Yes	-	One DCB update received per category, HIT_MAP_1G update.
DCB_UPDATE_HIT_MAP_16G	0x12F	Yes	-	One DCB update received per category, HIT_MAP_16G update.
DCB_UPDATE_HIT_MAP_64G	0x130	Yes	-	One DCB update received per category, HIT_MAP_64G update.

Event	Event ID	SID filterable	Non-attributable event	Description
DCB_UPDATE_HIT_MAP_L0_BLOCK	0x131	Yes	-	One DCB update received per category, HIT_MAP_L0_BLOCK update.
DCB_UPDATE_HIT_MAP_L0_TABLE	0x132	Yes	-	One DCB update received per category, HIT_MAP_L0_TABLE update.
As event 2, but only counts DPT misses	0x133	Yes	-	As event 2, but only counts Device Permission Table (DPT) misses.
As event 4, but only counts DPT walks	0x134	Yes	-	As event 4, but only counts DPT walks.
PTW formed VMSA DPT TLB entry	0x135	No	Yes	Virtual Memory System Architecture (VMSA) formed DPT cache entry inserted after a page table walk.
WC hit formed VMSA DPT TLB entry	0x136	No	Yes	VMSA formed DPT cache entry inserted after a walk cache hit.
DPT cache lookup slots are full causing backpressure to WMB request interface.	0x137	No	Yes	DPT cache lookup slots are full causing backpressure to WMB request interface.
CCB Plim RD	0x138	No	Yes	CCB PLIM Reads.
CCB Plim WR	0x139	No	Yes	CCB PLIM Writes.
CCB Pcnt RD	0x13A	No	Yes	CCB PCNT Reads.
CCB Pcnt WR	0x13B	No	Yes	CCB PCNT Writes.
CCB Repl RD	0x13C	No	Yes	CCB REPL Reads.
CCB Repl WR	0x13D	No	Yes	CCB REPL Writes.
CCB Tag RD	0x13E	No	Yes	CCB TAGS Reads.
CCB Tag WR	0x13F	No	Yes	CCB TAGS Writes.
CCB data RD	0x140	No	Yes	CCB DATA Reads.
CCB data WR	0x141	No	Yes	CCB DATA Writes.
WCB Plim RD	0x142	No	Yes	WCB PLIM Reads.
WCB Plim WR	0x143	No	Yes	WCB PLIM Writes.
WCB Pcnt RD	0x144	No	Yes	WCB PCNT Reads.
WCB Pcnt RD	0x145	No	Yes	WCB PCNT Writes.
WCB Repl RD	0x146	No	Yes	WCB REPL Reads.
WCB Repl WR	0x147	No	Yes	WCB REPL Writes.
WCB Tag RD	0x148	No	Yes	WCB TAGS Reads.
WCB Tag WR	0x149	No	Yes	WCB TAGS Writes.
WCB data RD	0x14A	No	Yes	WCB DATA Reads.
WCB data WR	0x14B	No	Yes	WCB DATA Writes.
AGW GCB Plim RD	0x14C	No	Yes	AGW GCB PLIM Reads.
AGW GCB Plim WR	0x14D	No	Yes	AGW GCB PLIM Writes.
AGW GCB Pcnt RD	0x14E	No	Yes	AGW GCB PCNT Reads.
AGW GCB Pcnt WR	0x14F	No	Yes	AGW GCB PCNT Writes.
AGW GCB Repl RD	0x150	No	Yes	AGW GCB REPL Reads.
AGW GCB Repl WR	0x151	No	Yes	AGW GCB REPL Writes.
AGW GCB Tgdt RD	0x152	No	Yes	AGW GCB TAGS Reads.

Event	Event ID	SID filterable	Non-attributable event	Description
AGW GCB Tgdt WR	0x153	No	Yes	AGW GCB TAGS Writes.
DGW GCB Plim RD	0x154	No	Yes	DGW GCB PLIM Reads.
DGW GCB Plim WR	0x155	No	Yes	DGW GCB PLIM Writes.
DGW GCB Pcnt RD	0x156	No	Yes	DGW GCB PCNT Reads.
DGW GCB Pcnt WR	0x157	No	Yes	DGW GCB PCNT Writes.
DGW GCB Repl RD	0x158	No	Yes	DGW GCB REPL Reads.
DGW GCB Repl WR	0x159	No	Yes	DGW GCB REPL Writes.
DGW GCB Tgdt RD	0x15A	No	Yes	DGW GCB TAGS Reads.
DGW GCB Tgdt WR	0x15B	No	Yes	DGW GCB TAGS Writes.
DCB Plim RD	0x15C	No	Yes	DCB PLIM Reads.
DCB Plim WR	0x15D	No	Yes	DCB PLIM Writes.
DCB Pcnt RD	0x15E	No	Yes	DCB PCNT Reads.
DCB Pcnt WR	0x15F	No	Yes	DCB PCNT Writes.
DCB Repl RD	0x160	No	Yes	DCB REPL Reads.
DCB Repl WR	0x161	No	Yes	DCB REPL Writes.
DCB Tgdt RD	0x162	No	Yes	DCB TAGS Reads.
DCB Tgdt WR	0x163	No	Yes	DCB TAGS Writes.

Note

A single DTI translation request might correspond to multiple translation request events in either of the following circumstances:

- A translation results in a stall fault event and is restarted.
- If a translation results in a stall fault event, because the Event queue is full, the translation is retried when an Event queue slot becomes available.

2.5.2.3 SMMU TBU events

The MMU S3 PMU can be configured to monitor a range of **IMPLEMENTATION DEFINED** TBU performance events.

The SMMUv3 architecture enables the filtering of events so that only events triggered by transactions with a specific StreamID (SID) and/or security state are counted. The event filtering includes the following:

- Speculative transactions and translations
- Transactions and translations that result in a terminated transaction or a translation fault

The events for which this type of filtering can occur are marked as 'Yes' in the 'SID filterable' columns in the following tables. For any events that are not SID filterable, but might reveal information about a security state, they are considered to be 'non-attributable' events by

the SMMUv3 architecture. These events are observed only if SMMU_PMCG_SCR and SMMU_PMCG_ROOTCR are set to count these events.

The following table shows the TBU performance events.

The following table shows the architecturally defined MMU S3 TBU performance events.

Table 2-28: SMMUv3 performance events for the TBU

Event	Event ID	SID filterable	Non-attributable event	Description
Main TLB lookup	0x80	Yes	-	Counts Main TLB lookups.
Main TLB miss	0x81	Yes	-	Counts translation requests that miss in the Main TLB.
Main TLB read	0x82	Yes	-	Counts once per access to the Main TLB RAMs, excluding reads that invalidation requests cause. **Note:** A transaction might access the Main TLB multiple times to look for different page sizes.
MicroTLB lookup	0x83	Yes	-	Counts MicroTLB lookups.
MicroTLB miss	0x84	Yes	-	Counts translation requests that miss in the MicroTLB.
Translation slots full	0x85	No	Yes	Counts once per cycle when all slots are occupied and not ready to issue transactions downstream.
Out of translation tokens	0x86	No	Yes	Counts once per cycle when a translation request cannot be issued because all translation tokens are in use.
Write data buffer full	0x87	No	Yes	Counts once per cycle when a transaction is blocked because the write data buffer is full.
DHCMO FAULT_RAZWI response	0x8A	Yes	-	For the ACE-Lite TBU, counts when R or W permission is not granted, or DRE permission is not granted on DTI for an InvalidateHint transaction. For the LTI TBU, counts when R or W permission is not granted, or DRE permission is not granted on DTI for a DHCMO transaction. This results in the LRRESP being converted from Success to FaultRAZWI.
DCMO downgrade	0x8B	Yes	-	For the ACE-Lite TBU, counts when either: • A MakeInvalid transaction on the TBS interface is output as CleanInvalid on the TBM interface • A ReadOnceMakeInvalid transaction on the TBS interface is output as ReadOnceCleanInvalid on the TBM interface For the LTI TBU, counts once per cycle when an LTI DCMO or R-DCMO transaction on the LA channel is responded to with a downgrade on the LR channel.

Event	Event ID	SID filterable	Non-attributable event	Description
Stash fail	0x8C	Yes	-	For the ACE-Lite TBU, counts when either: • A WriteUniquePtlStash or WriteUniqueFullStash transaction on TBS is output as a WriteNoSnoop or WriteUnique transaction on the TBM interface • A StashOnceShared or StashOnceUnique transaction on the TBS interface has a valid translation, but is terminated in the TBU For the LTI TBU, counts once whenever either an: • LTI WDCP transaction on the LA channel is downgraded as W on the LR channel • LTI DCP transaction on the LA channel that is responded to as FaultRAZWI on the LR channel is counted. This can be because of the following: • Memory attributes or DCP, R, W, or X permission check failure in the Translation Lookaside Buffer Unit (TLBU) • DTI fault response with Non-Abort. The transaction that is responded to with FaultAbort because of DTI StreamDisable or GlobalDisable is not counted. **Note:** A StashOnceShared or StashOnceUnique transaction that is terminated because of a StreamDisable or GlobalDisable translation response does not cause this event to count.
Fastpath taken	0x8D	Yes	-	This event counts each time the fastpath is taken. The fastpath is a lower latency path through the TBU. You can use a fastpath when the following conditions are true: • The TBU is an ACE-Lite or LTI TBU with only one interface • The request is unordered, that is, AxIDUNQ == 1 or LAOGV == 0 • There is a Micro TLB hit • Responses are being accepted • There are no existing transactions currently using the slow path
GPC only request	0x8E	Yes	-	This event counts each GPC only request. A GPC only request occurs when a request arrives with AxMMUVALID == 0 or LAMMUV == 0.
Fixed burst terminated in TBU	0x8F	No	No	This event counts each time a fixed burst is terminated in the TBU. Fixed bursts are terminated in the TBU if the fixed burst would otherwise be output as cacheable and outer shareable. AXI does not support shareable fixed bursts.
MTLB lookup slots full	0x96	No	Yes	This event indicates that the MTLB lookup slots are full.
MTLB update slots full	0x97	No	Yes	This event indicates that the MTLB update slots are full.
LTI port slots full	0xD0 - 0xD7	No	Yes	LTI port event (0xD0 + N) corresponds to LTI port N. Counts once per cycle when the slots that are allocated to the LTI port are all occupied and not ready to issue downstream.
LTI port out of translation tokens	0xE0 - 0xE7	No	Yes	LTI port event (0xD0 + N) corresponds to LTI port N. Counts once per cycle when a translation request cannot be issued for an LTI port because all its allocated translation tokens are in use.

Event	Event ID	SID filterable	Non-attributable event	Description
Hazard hit, request not sent downstream because of hazard	0xF0	Yes	-	This event counts each hazard hit. This event occurs if the hazarding mechanism in the TBU has been able to prevent a translation request going to the TCU because it matches an existing request outstanding to the TCU. When the existing translation response returns, it is also used to satisfy this translation request.
Hazard created, new hazard entered into hazard cache	0xF1	Yes	-	This event counts each time a hazard is created and indicates that a new hazard cache entry has been created. This event means that the translation request has been sent to the TCU and a hazard has been set up to enable future translation requests with the same request information to hazard against it, and potentially share the response.
Hazard cache full, all hazard slots in use	0xF2	Yes	-	This event counts each time a hazard is created when all hazard slots are in use, thereby discarding an existing hazard, and indicates that the maximum number of hazards have been set up in the hazard cache. New hazards are still created, but only by removing existing hazards. This can mean missing opportunities to benefit from hazarding. If hazarding is low, this is expected. However, if the potential for hazarding in your traffic is high, increasing the value of the TBUCFG_HZRD_ENTRIES parameter can be beneficial. See the TBUCFG_HZRD_ENTRIES parameter in the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.
upd_trng_004k	0xF3	Yes	-	MTLB update request with TRANS_RNG = 4k.
upd_trng_016k	0xF4	Yes	-	MTLB Update request with TRANS_RNG = 16k.
upd_trng_064k	0xF5	Yes	-	MTLB Update request with TRANS_RNG = 64k.
upd_trng_002m	0xF6	Yes	-	MTLB Update request with TRANS_RNG = 2M.
upd_trng_032m	0xF7	Yes	-	MTLB Update request with TRANS_RNG = 32M.
upd_trng_512m	0xF8	Yes	-	MTLB Update request with TRANS_RNG = 512M.
upd_trng_001g	0xF9	Yes	-	MTLB Update request with TRANS_RNG = 1G.
upd_trng_016g	0xFA	Yes	-	MTLB Update request with TRANS_RNG = 16G.
upd_trng_064g	0xFB	Yes	-	MTLB Update request with TRANS_RNG = 64G.
upd_trng_512g	0xFC	Yes	-	MTLB Update request with TRANS_RNG = 512G.
upd_trng_004t	0xFD	Yes	-	MTLB Update request with TRANS_RNG = 4T.
upd_trng_full	0xFE	Yes	-	MTLB Update request with TRANS_RNG = FULL.
MTLB Plim RD	0xFF	No	Yes	MTLB PLIM Reads.
MTLB Plim WR	0x100	No	Yes	MTLB PLIM Writes.
MTLB Pcnt RD	0x101	No	Yes	MTLB PCNT Reads.
MTLB Pcnt WR	0x102	No	Yes	MTLB PCNT Writes.
MTLB Repl RD	0x103	No	Yes	MTLB REPL Reads.
MTLB Repl WR	0x104	No	Yes	MTLB REPL Writes.
MTLB Tgdt RD	0x105	No	Yes	MTLB TAGS Reads.
MTLB Tgdt WR	0x106	No	Yes	MTLB TAGS Writes.

2.5.2.4 SMMUv3 PMU register architectural options

The SMMUv3 architecture defines the Performance Monitor Counter Group (PMCG) configuration register, SMMU_PMCG_CFGR. An MMU S3 implementation assumes fixed values for SMMU_PMCG_CFGR, and these values define behavioral aspects of the implementation.

The following table shows the SMMU_PMCG_CFGR register options that the MMU S3 TCU and TBU use.

Table 2-29: MMU S3 SMMU_PMCG_CFGR register architectural options

Field		Default value	Description for default value
SID_FILTER_TYPE		1	A single StreamID filter applies to all PMCG counters
CAPTURE		1	Capture of counter values into SVRn registers is supported
MSI		0	The counter group does not support Message Signaled Interrupts (MSIs)
RELOC_CTRS		1	The PMCG registers are relocated to page 1 of the PMU address map
SIZE		`0x31`	The counter group implements 32-bit counters
MPAM		0	Memory System Resource Partitioning and Monitoring (MPAM)
NCTR	NCTR for the TCU	See Description	The counter group includes `TCUCFG_PMU_COUNTERS` counters. See 2.7.1 TCU I/O and buffer configuration parameters on page 120. The default value is `TCUCFG_PMU_COUNTERS` - 1
	TCTR for the TBU	See Description	The counter group includes `TBUCFG_PMU_COUNTERS` counters. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122. The default value is `TBUCFG_PMU_COUNTERS` - 1

See also 3.3 SMMU memory map on page 137.

2.5.2.5 PMU snapshot interface

The Performance Monitoring Unit (PMU) snapshot interface is included on the TCU and on each TBU. You can use this asynchronous interface to initiate a PMU snapshot. A simultaneous snapshot of each counter register is created and copied to the respective SMMU_PMCG_SVRn register.

The PMU snapshot sequence is a 4-phase handshake. Both pmusnapshot_req and pmusnapshot_ack are LOW after reset. A snapshot occurs on the rising edge of pmusnapshot_req, and is equivalent to writing the value 1 to SMMU_PMCG_CAPR.CAPTURE.

The pmusnapshot_req signal is sampled using synchronizing registers. A register drives pmusnapshot_ack so that the connected component can sample the signal asynchronously.

See also:

- 2.5.5 RAS implementation on page 71
- A.3.2 PMU Snapshot signals on page 336

2.5.3 Multiple LTI interface TBU

There are variants of the LTI TBU that have 1, 2, 4, 6, or 8 LTI interfaces. The traffic through these interfaces is multiplexed together and they share a single translation manager, TLB, and DTI interface.

Registers are provided to enable you to set limits on the maximum proportion of these shared resources any individual LTI interface can use. This limiting applies to entries in the translation managing structure and DTI translation tokens. See the 3. Programmers model for MMU S3 on page 129.

A given LTI interface is not permitted to issue more translation requests into the TBU if it already uses the limit or more than the limit of any of the types of managed resource. If the limits are changed while an interface has outstanding translation requests, and this results in the interface using more than the new permitted proportion, it is unable to issue translation requests until the resource usage returns to below the permitted maximum. To avoid deadlock, each LTI interface must be guaranteed to be able to have at least 2 outstanding DTI requests and 2 outstanding LTI translation requests, one on each virtual channel on the LTI interface.

As a result, when there is more than one LTI interface, the maximum number of translation requests that any individual LTI interface might have outstanding is less than the total number possible across all interfaces. An LTI transaction is considered to be outstanding for these purposes for its full life according to the *AMBA® LTI Protocol Specification*. An LTI transaction is only considered to be using a DTI translation token until the translation manager determines that it is not required to perform a DTI translation request in the future. This determination can be because it has completed a DTI translation request or because it is not necessary to perform one because it gets a hit response to its TLB lookup or it receives a hazard response from another transaction ahead of it.

In all cases, a DTI translation token is required for a transaction to enter the LTI TBU because at the point that it enters the TBU, it cannot be known whether it requires a DTI translation request or not. Because the LTI TBU cannot guarantee deadlock freedom unless it receives at least 2 DTI credits per LTI interface, it fails to exit the Q_STOPPED LPI state if the max_trans_tok tie off signal does not indicate at least that number of tokens should be requested at connection. Similarly, on connection, if the TCU does not grant the minimum number of credits, then the TBU does not unfence its LTI interface, and instead initiates a DTI disconnection when this connection condition is detected.

2.5.4 Main TLB direct indexing and main TLB direct partitioning

Main TLB direct indexing can help your system to meet real-time translation requirements by enabling management of Main TLB (MTLB) entries from outside the TBU.

Note

If you use the Main TLB direct indexing or Main TLB direct partitioning features, the MPAM information is not used to control the allocation of the Main TLB.

Direct indexing enables real-time translation requirements to be met, as follows:

- You can guarantee that prefetched entries are not overwritten.
- The MTLB can be partitioned into different sets of entries that different streams use.

When direct indexing is enabled for a TBU:

- Lookups and updates to the MTLB use the mtlbidx field as the index.
- Updates to the MTLB use the way that mtlbway specifies.
- Lookups to the MTLB operate on all ways simultaneously.

If you configure your system to not use MTLB direct indexing, you can select MTLB direct partitioning. MTLB direct partitioning has similar behavior, but only the most significant TLB index bits must be provided externally, and the other bits are generated internally.

Direct indexing is enabled for a TBU when the `TBUCFG_DIRECT_IDX` parameter is set to 1.

When `TBUCFG_DIRECT_IDX` is set to 1, or when an MTLB is partitioned, the width of the AxUSER signals on the TBS interface is extended to convey the indexing information that is required for MTLB direct indexing or MTLB direct partitioning.

If `TBUCFG_DIRECT_IDX` is set to 0 and `TBUCFG_MTLB_PART` is set to 1 then LATLBLOC is ignored and can be tied to 0.

Indexing information is sent using the LATLBLOC signal for the LTI TBU. See A.5.1 LTI request channel signals on page 343.

Note

The table shows the extended bits in the order MSB first.

Table 2-30: Extended aruser_s and awuser_s bits for MTLB partitioning

Field name	Width	Description
mtlbidx	When direct indexing is enabled, the width of this field is $\log_2$(`TBUCFG_MTLB_DEPTH`) - $\log_2$(`TBUCFG_MTLB_WAYS`). When direct indexing is not enabled, the width of this field is `0`.	MTLB index
mtlbway	When direct indexing is enabled, the width of this field is $\log_2$(`TBUCFG_MTLB_WAYS`). When direct indexing is not enabled, the width of this field is `0`.	MTLB way
mtlbpart	$\log_2$(`TBUCFG_MTLB_PARTS`)	MTLB partition
-	`TBUCFG_AWUSER_WIDTH` for awuser_s. `TBUCFG_ARUSER_WIDTH` for aruser_s.	Regular AxUSER signals

If an MTLB is partitioned:

- The MTLB size is divided into a number of equal segments that the value of the TBUCFG_MTLB_PARTS parameter defines.
- The mtlbpart field defines the $\log_2$(TBUCFG_MTLB_PARTS) most significant index bits.

To maintain system performance, we recommend that you disable DVM invalidation on TBUs on which direct indexing is enabled. Disable DVM invalidation by setting the appropriate TCU_NODE_CTRLn.DIS_DVM bit. See 3.7.11 TCU_NODE_CTRLn register on page 168.

2.5.5 RAS implementation

Reliability, Availability, and Serviceability (RAS) features enable memory, normally SRAM, corruption to be detected and corrected, optionally generating interrupts into the system. All MMU S3 RAM-based caches support either RAS error detection, or RAS error detection and correction.

MMU S3 implements a combination of Single-Error-Correct-Double-Error-Detect (SECDED) and Double-Error-Detect (DED) error correction mechanisms.

SECDED is used in RAMs where a double error usually means that the SMMU cannot contain this error and must raise a Critical Error Interrupt. DED is used in TLB TAGS or DATA, where a single or double error can be recovered by fetching data from System Memory.

The SMMU also raises the following, based on a contained error or an uncontained error respectively:

- Fault Handling Interrupts (FHIs)
- Error Recovery Interrupts (ERIs)
- CRitical Error Interrupts (CRIs)

In addition to detection of errors in memories, MMU S3 includes the following sources of error detection:

Flop parity

The value of the TCUCFG_PARITY_ON parameter controls whether or not parity is enabled for large flop structures and some software-accessible registers in the TCU.

The TBUCFG_PARITY_ON parameter controls the same behavior for structures and some software-accessible registers in the TBU.

All Software-accessible registers, except for the following registers, are unaffected by the TCUCFG_PARITY_ON and TBUCFG_PARITY_ON parameter settings:

- SMMU_PMCG_*
- *_ITEN, that is:
  - 3.11.1 ITEN register for the TCU on page 231
  - 3.19.1 ITEN register for the TBU on page 291
- *_ITEN_ET, that is:

- 3.11.2 ITEN_ET register for the TCU on page 232
- 3.19.2 ITEN_ET register for the TBU on page 292

- *_ITIN, that is:
  - 3.12.2 ITIN_PIU register for the TCU Programmer Interface Unit on page 242
  - 3.13.2 ITIN_TMU register for the TCU Translation Management Unit on page 245
  - 3.19.4 ITIN register for the TBU on page 297
- *_ITOP, that is:
  - 3.12.1 ITOP_PIU register for the TCU Programmer Interface Unit on page 234
  - 3.13.1 ITOP_TMU register for the TCU Translation Management Unit on page 243
  - 3.19.3 ITOP register for the TBU on page 293

Register that are not affected by the `TCUCFG_PARITY_ON` and `TBUCFG_PARITY_ON` parameters are considered to be critical configuration registers and are protected by parity logic regardless of the parameter value settings.

A parity error on one of these sets of flops means that the SMMU cannot contain this error and must raise a CRitical Error Interrupt. In parallel, a Fault Handling Interrupt (FHI) and an Error Recovery Interrupt (ERI) are also raised.

Poison

If the poison_support tie-off signal is tied HIGH, see A.2.9 TBU tie-off signals on page 331, the TCU QTW and PTW interfaces support the poison functionality. The poison function signals that a piece of data received has previously been corrupted and cannot be used. MMU S3 cannot correct a poison error. Software can potentially recover data by re-writing the affected piece of data in memory. If corruption of write data occurs while in the Write Data Buffer, and poison functionality is enabled, the ACE-Lite TBU also generates poison signals.

The following tables show the MMU S3 error sources, and the actions that are taken when errors occur.

In the tables, the following abbreviations apply:

CE

Corrected Error

CRI

Critical Error Interrupt

DE

Deferred Error

DED

Double-Error-Detect

ERI

Error Recovery Interrupt

FHI
Fault Handling Interrupt

SEC
Single-Error-Correct

SECDED
Single-Error-Correct-Double-Error-Detect

UC
Uncontainable error

UE
Uncorrected Error

UER
Recoverable Uncorrected Error

The following table shows the RAM RAS error sources in MMU S3, and actions that are taken when errors occur.

Table 2-31: RAM RAS error sources

RAM name	Error correction and detection mechanism	RAS error triggered		RAS interrupts triggered
BIU WDB ROBUFF D	SEC	CE		FHI
	DED	Poison supported:	DE	FHI
		Poison not supported:	UE (UC)	ERI, FHI, and CRI
BIU WDB ROBUFF C	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
BIU WDB ROBUFF P	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
TLBU TOU OGQ	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
TLBU TOU UOQ	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
TLBU TOU DTIQ	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
TLBU TOU REQ	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
TLBU TOU RSP	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
TLBU TOU LB	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
TLBU TOU HLB ENTRY LEFT	SEC	CE		FHI
	DED	UE (UC)		FHI, ERI, and CRI
TLBU TOU HLB ENTRY RIGHT	SEC	CE		FHI

RAM name	Error correction and detection mechanism	RAS error triggered	RAS interrupts triggered
	DED	UE (UC)	FHI, ERI, and CRI
TLBU TOU HLB PTR LEFT	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TLBU TOU HLB PTR RIGHT	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TLBU DCU MTLB PLIM	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TLBU DCU MTLB PCNT	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TLBU DCU MTLB REPL	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TLBU DCU MTLB TAGS	SED	CE	FHI
	DED	CE	FHI
TMU TWB BSU	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU HZU PTR	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU TWB WMB LKP STATUS	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU TWB WMB WLK STATUS	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU TWB WMB SCRATCH	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU HTTU RAM	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU WCB MWC PLIM	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU WCB MWC PCNT	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU WCB MWC REPL	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU WCB MWC TAGS	SED	CE	FHI
	DED	CE	FHI
TMU WCB MWC DATA	SED	CE	FHI
	DED	CE	FHI
TMU CCB MCC PLIM	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU CCB MCC PCNT	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU CCB MCC REPL	SEC	CE	FHI

RAM name	Error correction and detection mechanism	RAS error triggered	RAS interrupts triggered
	DED	UE (UC)	FHI, ERI, and CRI
TMU CCB MCC TAGS	SED	CE	FHI
	DED	CE	FHI
TMU CCB MCC DATA	SED	CE	FHI
	DED	CE	FHI
TMU HZUSEC PTR	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
PIU AGW GMB WLK STATUS	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
PIU AGW GMB LOOP	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
PIU AGW GCB TAGS+DATA	SED	CE	FHI
	DED	CE	FHI
PIU AGW GCB REPL	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW ID RAM PARITY	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW STATUS RAM PARITY	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW RESPONSE RAM PARITY	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW GMB SCRATCH	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW GMB WLK STATUS	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW GMB LOOP	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU DGW RAM	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU DGW GMB SCRATCH	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU DGW GMB WLK STATUS	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU DGW GMB LOOP	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU DCB TAGS+DATA	SEC	CE	FHI
	DED	CE	FHI
TMU DCB REPL	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI

RAM name	Error correction and detection mechanism	RAS error triggered	RAS interrupts triggered
TMU DCB PCNT	SEC	CE	FHI, ERI, and FHI on DED
	DED	UE (UC)	FHI, ERI, and CRI
TMU DCB PLIM	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW GCB TAGS+DATA	SED	CE	FHI
	DED	CE	FHI
TMU DGW GCB TAGS+DATA	SED	CE	FHI
	DED	CE	FHI
TMU AGW GCB REPL	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW GCB PCNT	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU AGW GCB PLIM	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU DGW GCB REPL	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU DGW GCB PCNT	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI
TMU DGW GCB PLIM	SEC	CE	FHI
	DED	UE (UC)	FHI, ERI, and CRI

The following table shows the non-RAM RAS error sources in MMU S3, and actions that are taken when errors occur.

Table 2-32: Non-RAM RAS error sources

Source name	Error correction and detection mechanism	RAS error triggered	RAS interrupts triggered
PIU CMD RPOISON	RPOISON received, no correction	UE (UER)	FHI and ERI
PIU AGW GMB GPT WLK POISON	RPOISON received, no correction	UE (UER)	FHI and ERI
TMU AGW GMB GPT WLK POISON	RPOISON received, no correction	UE(UER)	FHI and ERI
DGW GMB GPT WLK POISON	RPOISON received, no correction	UE(UER)	FHI and ERI
TMU TWB WMB POISON	RPOISON received, no correction	UE (UER)	FHI and ERI
TMU TWB CMB POISON	RPOISON received, no correction	UE (UER)	FHI and ERI
PIU QTW FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU PIB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU AGW GMB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU AGW GCB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU AGW FF PARITY	FF parity error/FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU INV FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU CMD FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU EVT FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI

Source name	Error correction and detection mechanism	RAS error triggered	RAS interrupts triggered
PIU PRI FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI and CRI
PIU MGB FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU CPB FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU FAR FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU PRG FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PIU RAS FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU DCB FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU HZU FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU PIB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU DMD FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU AGW GMB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU AGW GCB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU AGW FF PARITY	FF parity error/FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU DGW GMB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU DGW GCB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU DGW FF PARITY	FF parity error/FF ctrl parity error r, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU TWB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU WCB FF PARITY	FF parity error/FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU CCB FF PARITY	FF parity error/FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU PRG FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TMU PMU FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
BIU SIBMIB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
BIU WTB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
DIB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
DCU DCB FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
DCU TLB FF PARITY	FF parity error/FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
TOU FF PARITY	FF parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
BIU CPB FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
PMU FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI
RRB FF PARITY	FF ctrl parity error, uncorrectable	UE (UC)	FHI, ERI, and CRI

2.5.6 Quality of Service

You can program the TCU with a priority level for each LTI TBU interface. The priority level is applied to every translation from that TBU interface.

The TCU uses this priority level to:

- Arbitrate between translations that are waiting in the translation request buffer when translation manager slots become available

- Determine the AXI AxQOS value for translation table walks, configuration table walks and Granule Protection Table (GPT) walks, that the TCU issues on the QTW/DVM interface or Page Table Walk (PTW) interface

The arbiters contain starvation avoidance mechanisms to prevent transactions from being stalled indefinitely.

The TBU does not implement any prioritization between transactions. However, if a TBU has multiple LTI ports, the resources that a given port uses can be capped using the 3.16.2 TBU_LTI_PORT_RESOURCE_LIMIT register on page 250. In most use cases, this limiting functionality, combined with the per-LTI interface priority level provides the required QoS management functionality. However, sometimes it might be necessary to use separate TBUs for requesters with different QoS requirements.

See also:

- 3.7.2 TCU_QOS register on page 157
- 3.7.11 TCU_NODE_CTRLn register on page 168

2.5.7 Distributed Virtual Memory messages

The QTW/DVM interface supports Distributed Virtual Memory (DVM) messages. MMU S3 supports DVMv9.2.

The interface supports DVM transactions of message types TLB Invalidate and Synchronization. The interface accepts all other DVM transaction message types, and sends a snoop response, but otherwise ignores such transactions.

Tie the sup_btm input signal HIGH when the system supports Broadcast TLB Maintenance.

When Broadcast TLB Maintenance is supported, you can use SMMU_CR2, SMMU_R_CR2, and SMMU_S_CR2 to control how the SMMU handles TLB Invalidate operations as follows:

SMMU_CR2.PTM = 0
: Non-secure TLB Invalidate operations are applied to the TLBs

SMMU_CR2.PTM = 1
: Non-secure TLB Invalidate operations have no effect

SMMU_S_CR2.PTM = 0
: Secure TLB Invalidate operations are applied to the TLBs

SMMU_S_CR2.PTM = 1
: Secure TLB Invalidate operations have no effect

SMMU_R_CR2.PTM = 0
: Realm TLB Invalidate operations are applied to the TLBs

SMMU_R_CR2.PTM = 1
: Realm TLB Invalidate operations have no effect

Note

Invalidation by PA is applied regardless of the above register values.

Note

When sup_btm is tied HIGH, the reset value of SMMU_CR2.PTM, SMMU_S_CR2.PTM, and SMMU_R_CR2.PTM is 1.

Note

Although TLB Invalidate operations have no effect when PTM = 1, the QTW/DVM interface still returns the appropriate response.

The QTW/DVM interface might receive DVM Sync transactions without receiving a DVM TLB Invalidate transaction, or when the PTM bits have masked a TLB Invalidate. If no DVM TLB Invalidate operations have occurred since the most recent DVM Sync transaction, subsequent DVM Sync transactions result in an immediate DVM Complete transaction. This behavior ensures that the TCU does not affect system DVM performance unless TLB Invalidate operations are performed.

The ACE-Lite interface allocates the access permissions and Shareability of DVM Complete transactions as follows:

- ARPROT = `0b000`, indicating Unprivileged, Secure, Data access
- ARDOMAIN = `0b10`, indicating Outer Shareable

See also 3.2 SMMU architectural registers on page 131.

2.5.8 TCU transaction handling

The transaction width, burst length, and transfer size that the TCU supports depend on the transaction type.

The following table shows the TCU support for read transactions.

Note

In the table, the value of the `TCU_ID_WIDTH` parameter is equal to the external ID width that 2.4.1.1 TCU Queue and Table Walk/Distributed Virtual Memory interface on page 32 defines.

Table 2-33: TCU support for read transactions

Transaction type	arid_qtw[TCU_ID_WIDTH - 1:6]	arid_qtw[5:0]	Interface used
Command Queue read	Any additional bits are 0	6'b00_0000	QTW/DVM
Distributed Virtual Memory (DVM) Complete	Any additional bits are 0	6'b00_1000	QTW/DVM
Granule Protection Table (GPT) walk for CMDQ read	Any additional bits are 0	6'b10_0000	QTW/DVM
GPT walk for PRIQ write	Any additional bits are 0	6'b10_1000	QTW/DVM
GPT walk for EVTQ write	Any additional bits are 0	6'b11_0000	QTW/DVM
GPT walk for Message Signaled Interrupt (MSI) write	Any additional bits are 0	6'b11_1000	QTW/DVM
Device Permission Table (DPT) walk	Indicates the DPT walk, including bits marked as x in range [5:0]	6'bxx_x001 In addition to bits arid_qtw[TCU_ID_WIDTH - 1:6], bits marked as x in arid_qtw[5:0] are also used as part of the DPT walk ID.	PTW if present, otherwise QTW/DVM
GPT walk for CFG fetch	Indicates the configuration table walk slot requesting the page table walk, including bits marked as x in range [5:0]	6'bxx_x010 In addition to bits arid_qtw[TCU_ID_WIDTH - 1:6], bits marked as x in arid_qtw[5:0] are also used as part of the GPT walk ID.	QTW/DVM
Configuration table walk	Indicates the configuration table walk slot requesting the page table walk, including bits marked as x in range [5:0]	6'bxx_x011 In addition to bits arid_qtw[TCU_ID_WIDTH - 1:6], bits marked as x in arid_qtw[5:0] are also used as part of the configuration walk ID.	QTW/DVM
GPT walk for PTE fetch / DPT fetch/ Hardware Translation Table Update (HTTU) write	Indicates the page table walk slot requesting the page table walk, including bits marked as x in range [5:0]	6'bxx_x100 In addition to bits arid_qtw[TCU_ID_WIDTH - 1:6], bits marked as x in arid_qtw[5:0] are also used as part of the GPT walk ID.	QTW/DVM
Page table walk	Indicates the page table walk slot requesting the page table walk, including bits marked as x in range [5:0]	6'bxx_x101 In addition to bits arid_qtw[TCU_ID_WIDTH - 1:6], bits marked as x in arid_qtw[5:0] are also used as part of the page table walk ID.	PTW if present, otherwise QTW/DVM
GPT walk for final response	Indicates the DGW slot number of the final request, including bits marked as x in range [5:0]	6'bxx_x110 In addition to bits arid_qtw[TCU_ID_WIDTH - 1:6], bits marked as x in arid_qtw[5:0] are also used as part of the GPT walk ID.	QTW/DVM
HTTU access R response	Indicates the page table walk slot requesting the page table walk, including bits marked as x in range [5:0]	6'bxx_x111 In addition to bits arid_qtw[TCU_ID_WIDTH - 1:6], bits marked as x in arid_qtw[5:0] are also used as part of the HTTU update ID.	PTW if present, otherwise QTW/DVM

- DVM Complete transactions are always one beat of full data width.
- Command queue reads and DVM Complete transactions are independent of translation slots.

Therefore, the maximum number of read transactions that the TCU can issue at any time as follows:

- When TCUCFG_LEGACY_TZ_EN is set to 0, the maximum number of read transactions is as follows:

  (2 × TCUCFG_PTW_SLOTS) + TCUCFG_DGW_SLOTS + 5.

The PIU can generate an additional five outstanding read transactions as follows:

- One GPC check for a PRIQ queue write
- One GPC check for an EVENTQ write
- One GPC check for a MSI write
- One GPC check for a CMDQ read or one actual CMDQ read, which are serialized so there is only one
- One outstanding DVM Complete message on the AR channel

These extra five outstanding read transactions are low bandwidth, and having all five transactions outstanding at the same time is difficult.

- When `TCUCFG_LEGACY_TZ_EN` is set to 1, the maximum number of read transactions is as follows:

  `TCUCFG_PTW_SLOTS` + 2

See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

The following table shows the TCU support for write transactions.

Note

In the table, `TCU_ID_WIDTH` is equal to the external ID width that 2.4.1.1 TCU Queue and Table Walk/Distributed Virtual Memory interface on page 32 defines.

Table 2-34: TCU support for write transactions

Transaction type	awid_qtw[TCU_ID_WIDTH - 1:5]	awid_qtw[4:0]
Page Request Interface (PRI) Queue Write	0 if present	5'b01000
Event Queue write	0 if present	5'b10000
MSI write	0 if present	5'b11000
HTTU Write	Indicates the page table walk slot requesting the HTTU write, including bits marked as x in range [4:0]	5'bxx111

There can be a maximum of five outstanding writes at one time, which includes:

- Two Hardware Translation Table Updates (HTTUs)
- One Event Queue write
- One Message Signaled Interrupt (MSI) write
- One Page Request Interface (PRI) queue write

All read and write transactions are aligned to the transaction size.

2.5.9 TCU prefetch

The TCU can prefetch translations on a per-context basis to improve translation performance for real-time requesters that access memory linearly. If TCU prefetch is enabled, a second translation request occurs after the original request, and is initiated and terminated entirely within the TCU.

This second translation request is regarded as the prefetch because it is an advance request of the next translation that is expected to be requested. This second request is speculative and is used to allocate into the caches of the TCU.

Software can enable TCU prefetch for a particular translation context by programming the Stream Table Entry (STE). Bits [121:120] are **IMPLEMENTATION DEFINED** in the SMMUv3 architecture. See the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

MMU S3 uses these bits for the PF field as follows:

PF, bits [121:120]

This field determines whether prefetch is enabled or disabled for the translation context that this STE defines as follows:

0b00	Prefetching disabled
0b01	Reserved
0b10	Forward prefetching
0b11	Backward prefetching

Prefetching disabled

TCU prefetch does not occur

Reserved

Reserved values must not be used

Forward prefetching

The address to be prefetched is the first address following the end of the translation range, as DTI_TBU_TRANS_RESP.TRANS_RNG indicates

Backward prefetching

The address to be prefetched is the last address before the beginning of the translation range, as DTI_TBU_TRANS_RESP.TRANS_RNG indicates

Whenever a miss occurs in the MicroTLB and Main TLB of the TBU, the TBU sends a translation request to the TCU. If the STE for the translation is programmed to enable prefetch, each translation request to the TCU can also potentially result in a prefetch that occurs after the original request is complete.

When each incoming translation request completes its translation in the TCU, the STE.PF field indicates whether TCU prefetch is enabled. If TCU prefetch is enabled, a second translation request, the prefetch request, is then issued into the same TCU translation slot. This prefetch request is speculative, and only allocates into the TCU walk caches. A translation response for the prefetch is not returned to the TBU.

When the TCU handles each incoming translation request from the TBU, translation table walks can either occur, or not occur depending on whether there is a hit in each level of walk cache that is looked up. Translation table walks also might or might not occur for the subsequent prefetch request. The number of memory accesses that are performed for this prefetch are unrelated to the number of memory accesses that are performed for the original translation request.

Consider the following examples:

1. An incoming translation request might hit in the lowest level of walk cache, but the subsequent prefetch request might still require at least one translation table walk to memory.
2. The original translation request might require multiple translation table walks, but the subsequent prefetch request might hit in the lowest level of walk cache and not require any memory accesses. If the prefetch request hits in the lowest level of walk cache, then the walk caches are not updated and no memory accesses are performed.


Note

The walk cache uses a Re-Reference Interval Prediction (RRIP) replacement policy.

The prefetch can only occur when the original request is complete irrespective of whether translation table walks were required. Waiting for completion of the original request means that by the time it becomes possible for the prefetch to be initiated, the TCU might have already received a non-speculative request for the next translation and begun to handle this request using a separate translation slot.

Therefore, TCU prefetch results in a performance advantage only if the number of cycles between each sequential translation request from the TBU is greater than the number of cycles that is taken for the TCU to handle the original translation request and to start the subsequent prefetch.

Even if TCU prefetch is enabled, a prefetch does not occur if one of the following caused the original request:

- A speculative translation request, that is, DTI_TBU_TRANS_REQ.PERM[1:0] ;= ;2'b11, because a TBU receives a StashOnceShared, StashOnceUnique, or StashTranslation transaction
- A translation request for an atomic transaction that provides a data response, that is, DTI_TBU_TRANS_REQ.PERM[1:0] ;= ;2'b10, because a TBU receives an AtomicLoad, AtomicSwap, or AtomicCompare transaction

If the original translation request returns one the following, prefetch also does not occur:

- Fault response.
- Global bypass response.
- Stream bypass response.

Note

Prefetches can occur only with non-ATS translation requests because ATS itself is already a prefetch mechanism. Prefetch is not performed for translations that require DPT walks.

2.5.10 Error responses

The AMBA protocol defines external AXI subordinate error, SLVERR, and external AXI DECode ERRor, DECERR, and TRANSFAULT. The MMU S3 error response behavior depends on the interface.

The TCU ACE-Lite interface treats SLVERR and DECERR identically, as an abort.

When terminating a transaction, the TBS interface generates an OKAY, SLVERR, or TRANSFAULT response depending on the reason for the termination and the value of AxMMUFLOW.

If the TBU TBM interface receives a DECERR or SLVERR response to a downstream transaction, it propagates the same abort type to the TBS interface.

2.5.11 Conversion between ACE-Lite and Armv8 attributes

The SMMUv3 architecture defines attributes in terms of the Arm®v8 architecture. The MMU S3 components are therefore required to perform conversion between ACE-Lite and Arm®v8 attributes.

See the *Arm® Architecture Reference Manual for A-profile architecture*.

The TBU must convert:

- ACE-Lite attributes to Arm®v8 attributes when it receives transactions on the Transaction Subordinate (TBS) interface
- Arm®v8 attributes to ACE-Lite attributes when it outputs transactions on the Transaction Manager (TBM) interface

The TCU must convert Arm®v8 attributes to ACE-Lite attributes when it outputs transactions on the QTW/DVM and PTW interfaces.

2.5.11.1 Subordinate interface memory type attribute handling

The AxCACHE and AxDOMAIN signals contain the memory attributes that apply to the TBS interface.

The following table shows the ACE-Lite to Armv8 attribute conversions that the TBU TBS interface performs.

Table 2-35: MMU S3 ACE-Lite to Armv8 memory attribute conversions

AxCACHE attribute	AxDOMAIN attribute	Armv8 memory type	Armv8 Shareability
Device Non-bufferable	System	Device-nGnRnE	Outer Shareable
Device Bufferable	System	Device-nGnRE	Outer Shareable
Normal Non-cacheable Bufferable Normal Non-cacheable Non-bufferable Write-Through No Allocate Write-Through Read-Allocate Write-Through Write-Allocate Write-Through Read and Write-Allocate	Any	Normal Inner Non-cacheable Outer Non-cacheable	Outer Shareable
Write-Back No Allocate Write-Back Read-Allocate Write-Back Write-Allocate Write-Back Read-Allocate Write-Allocate	Non-shareable Inner Shareable Outer Shareable	Normal Inner Write-Back Outer Write-Back	Non-shareable Non-shareable Outer Shareable

Note

- Write-Back transactions are always treated as non-transient.
- The Armv8-A Read-Allocate and Write-Allocate hints are the same as the hints that the AxCACHE Write-Back type provides.
- The TBU TBS interface converts instruction writes into data writes, that is, it treats awprot_s[2] as 0.

2.5.11.2 Manager interface memory type attribute handling

The AxCACHE and AxDOMAIN signals contain the memory attributes that apply to the TBM and the QTW/DVM and PTW interfaces.

The TBU TBM interface can also use the AxLOCK signal to indicate an Exclusive access. The QTW/DVM and PTW interfaces do not use the AxLOCK signal.

On the TBU TBM interface, a bit on AxUSER indicates whether the memory type before the conversion is Outer Cacheable.

The following table shows the Armv8 to ACE-Lite attribute conversions that the manager interfaces perform.

Table 2-36: MMU S3 Armv8 to ACE-Lite memory attribute conversions

Armv8 memory type	AxCACHE attribute	AxDOMAIN attribute	AxLOCK attribute	AxUSER Outer Cacheable
Device-nGnRnE	Device Non-bufferable	System	As Transaction Subordinate (TBS) AxLOCK value	0
Device-GRE Device-nGRE Device-nGnRE	Device Bufferable	System	As TBS AxLOCK value	0
Normal Inner Non-cacheable Outer Non-cacheable Normal Inner Write-Through Outer Non-cacheable Normal Inner Write-Back Outer Non-cacheable	Normal Non-cacheable Bufferable	System	As TBS AxLOCK value	0
Normal Inner Non-cacheable Outer Write-Through Normal Inner Write-Through Outer Write-Through Normal Inner Write-Back Outer Write-Through Normal Inner Non-cacheable Outer Write-Back Normal Inner Write-Through Outer Write-Back	Normal Non-cacheable Bufferable	System	As TBS AxLOCK value	1

Armv8 memory type	AxCACHE attribute	AxDOMAIN attribute	AxLOCK attribute	AxUSER Outer Cacheable
Normal Inner Write-Back Outer Write-Back	Write-Back No Allocate Write-Back Read-Allocate Write-Back Write-Allocate Write-Back Read and Write-Allocate	If AxBURST == FIXED, an abort is generated and the transaction does not propagate downstream. The failure to propagate is because AXI does not support shareable FIXED bursts. If AxBURST != FIXED, the attribute reflects the Armv8 Shareability: • Non-shareable • Outer Shareable An Armv8 shareability attribute of Inner Shareable is always output with an AxDOMAIN value of Outer Shareable because Inner Shareable is deprecated in ACE and ACE-Lite.	0	1

See also 2.5.12 AXI USER bits that the SMMU TBU TBM defines on page 87.

2.5.12 AXI USER bits that the SMMU TBU TBM defines

The TBU TBM interface AxUSER signals, aruser_m, and awuser_m, have 1 bit more than `TBUCFG_AxUSER_WIDTH` defines. This extra bit is output in higher-order bits of aruser_m and awuser_m.

The following table shows the MMU S3-defined aruser_m and awuser_m bits, where `w` represents the AXI USER bus width that `TBUCFG_AxUSER_WIDTH` defines.

Table 2-37: MMU S3-defined aruser_m and awuser_m bits

Bit position	Value
[w]	Outer Cacheable The Outer cacheable is added to the upper most bit of AxUSER on the TBM interface.

See also 2.5.11.2 Manager interface memory type attribute handling on page 85.

2.5.12.1 Page Based Hardware Attribute (PBHA) in SMMUs

The Arm® architecture defines that 4 bits in both stage 1 and stage 2 leaf page table entry formats are reserved for software use. Arm®v8.5 and SMMUv3 define a mechanism where software can declare that it does not require them, on a per bit basis.

See also:

- *Arm® Architecture Reference Manual for A-profile architecture*
- *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*

The Page Based Hardware Attribute (PBHA) mechanism effectively hands over control of those bits to **IMPLEMENTATION DEFINED** hardware purposes. These bits are called PBHA bits.

For more information about PBHA bits, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

2.5.13 NoStreamID

The SMMU can propagate transactions from devices that are not associated with a StreamID. These devices are called NoStreamID devices.

Transactions from NoStreamID devices are not translated, but are still subject to Granule Protection Checks (GPC), if enabled.

ACE-Lite TBU

When using the ACE-Lite TBU, setting AxMMUVALID = 0 indicates a NoStreamID access.

LTI TBU

When using the LTI TBU, setting LAMMUV = 0 indicates a NoStreamID access.

NoStreamID accesses are also called GPC only requests.

The `0x8E` TBU performance event can count GPC only requests. See performance event `0x8E` that the table in 2.5.2.4 SMMUv3 PMU register architectural options on page 67 describes. For more information about NoStreamID, see the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

NoSteamID transactions are typically used in systems that have upstream devices which directly access the Physical Address space without requiring translation using an ACE-Lite TBU or LTI TBU. However, in systems that use Realm Management Extension (RME), these transactions must still go through the SMMU so that GPCs can be performed.

2.6 Constraints and limitations of use

Certain usage constraints and limitations apply to MMU S3.

Unless otherwise specified, an **IMPLEMENTATION DEFINED** field in a structure that MMU S3:

- Generates is 0
- Reads is ignored

2.6.1 SMMUv3 implementation

MMU S3 implements SMMUv3 features.

See also:

- 2.5.11.2 Manager interface memory type attribute handling on page 85
- *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*

2.6.1.1 ID register architectural options

MMU S3 uses ID register architectural options that the SMMUv3 ID registers expose.

The following table shows the architectural options for MMU S3 from the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* that the SMMUv3 ID registers expose.

Table 2-38: SMMUv3 ID register options

Register	Field	Value	Description
SMMU_ROOT_IDR0	ROOT_IMPL	`1b1`	Root implemented.
	BGPTM	sup_btm	Participates in broadcast TLBI PA operations.
	RGPTM	`1b1`	TLBI registers present.
	REALM_IMPL	`1b1`	Realm programming interface is present.
	BA_REALM	`0x6`	Base address of Realm register page 0 is `0x20000` + (6 × `0x10000`).
SMMU_ROOT_IIDR	Implementer	`0x43B`	Arm implementation.
	Revision	MAX(*p_level*, ecorevnum)	Where *p_level* is: **1** For p1
	Variant	`1b1`	r1.
	ProductID	`0x498`	Third SMMUv3 implementation from Arm.
SMMU_IDR0	S2P	`1b1`	Stage 2 translation supported.
	S1P	`1b1`	Stage 1 translation supported.
	TTF	`2b10`	AArch64 translation supported, required for Realm Management Extension (RME), but does not change in legacy mode.
	COHACC	sup_cohacc or `1`, see the description	Coherent accesses supported, system configuration option. The value is set as follows: **sup_cohacc** If the `TCUCFG_LEGACY_TZ_EN` parameter is set to 1 or the legacy_tz_en tie-off signal is set to 1. **1** If the `TCUCFG_LEGACY_TZ_EN` parameter is set to 0 and the legacy_tz_en tie-off signal is set to 0.
	BTM	sup_btm	Broadcast TLB maintenance, system configuration option. BTM can be set to 0 by writing 1 to TCU_ROOT_CTRL.DIS_DVM.
	HTTU[1:0]	{sup_httu, `1b0`}	Access and dirty flag update supported. The sup_httu tie-off input signal configures the support. as follows: **{sup_httu, `1b0`}** If the `TCUCFG_LEGACY_TZ_EN` parameter is set to 1 or the legacy_tz_en tie-off signal is set to 1

Register	Field	Value	Description
	DORMHINT	1b0	Dormant hint is not supported.
	Hyp	1b1	EL2-E2H is supported.
	ATS	1b1	ATS is supported.
	NS1ATS	1b0	Stage 1-only ATS is supported.
	ASID16	1b1	16-bit ASID is supported.
	MSI	1b1	Message Signaled Interrupts (MSIs) are supported.
	SEV	sup_sev	Send event is supported, system configuration option.
	ATOS	1b0	ATOS is not supported.
	PRI	1b1	PRI is supported.
	VMW	1b1	VMID wildcard matching supported.
	VMID16	1b1	16-bit VMIDs are supported.
	CD2L	1b1	2-level context descriptor tables are supported.
	VATOS	1b0	Virtual ATOS is not supported.
	TTENDIAN	0b00	Mixed-endian translation walks are supported.
	ATSRECERR	1b1	Supports recording some events for ATS translation requests.
	STALL_MODEL	{0b0, SMMU_S_CR0.NSSTALLD}	Stall and terminate models that are supported unless the Secure world disables Non-secure stalling.
	TERM_MODEL	1b0	Terminating a transaction with RAZ/WI is supported.
	ST_LEVEL	2b01	2-level stream table is supported.
	RME_IMPL	~legacy_tz_en && ~TCUCFG_LEGACY_TZ_EN	RME supported unless in legacy TZ mode.
SMMU_IDR1	SIDSIZE	32	32-bit StreamIDs are supported.
	SSIDSIZE	20	20-bit SubstreamIDs are supported.
	PRIQS	5b10011	2^{19} PRI queue entries are supported.
	EVENTQS	5b10011	2^{19} Event queue entries are supported.
	CMDQS	5b10011	2^{19} Command queue entries are supported.
	ATTR_PERMS_OVR	1b1	Incoming permission attributes can be overridden.
	ATTR_TYPES_OVR	1b1	Incoming memory attributes can be overridden.
	REL	1b0	Reported base addresses are absolute addresses.
	QUEUES_PRESET	1b0	Not fixed queue base addresses.
	TABLES_PRESET	1b0	Not fixed table base addresses.
	ECMDQ	1b0	Enhanced Command Queue is not supported.
SMMU_IDR2	BA_VATOS	1b0	No VATOS support.

Register	Field	Value	Description
SMMU_IDR3	HAD	1b1	Hierarchical attribute disable is supported.
	PBHA	1b1	Page-Based Hardware Attributes (PBHA) are supported.
	XNX	1b1	EL0/EL1 stage 2 execute control is supported.
	PPS	1b1	PASID, when present always used in PRI auto-generated response.
	MPAM	1b1	MPAM is supported.
	FWB	1b1	S2 control of memory type is supported.
	STT	1b1	Small translation tables are supported.
	RIL	1b1	Range-based invalidation and Level hint are supported.
	BBML	2	Break before Make level 2 is supported.
	E0PD	1b1	E0PD mechanism is implemented.
	PTWNNC	1b1	Treat stage 1 translation walks mapped as Device memory as Normal Non-cacheable.
	DPT	1b0	Non-secure DPT is not supported.
SMMU_IDR4	IMPDEF	1b0	No **IMPLEMENTATION DEFINED** features.
SMMU_IDR5	OAS	sup_oas	Output address size, system configuration option.
	GRAN4K	1b1	4K translation granule is supported.
	GRAN16K	1b1	16K translation granule is supported.
	GRAN64K	1b1	64K translation granule is supported.
	VAX	2b01	Virtual addresses of 52 bits per CD.TTBx are supported.
	STALL_MAX	TCUCFG_XLATE_SLOTS	Maximum number of outstanding stalled transactions.
SMMU_IIDR	Implementor	0x43B	Arm® implementation.
	Revision	MAX(*p_level*, ecorevnum)	Where *p_level* is 1 for p1.
	Variant	1b1	Product variant, or major revision is r1.
	ProductID	0x498	Arm® ID.
SMMU_AIDR	ArchMinorRev	3	Architectural minor revision is SMMUv3.3.
	ArchMajorRev	1b0	Architectural Major Revision is SMMUv3.
SMMU_S_IDR0	MSI	1b1	Secure MSIs are supported.
	STALL_MODEL	2'b00	Stall and terminate model is supported.
	ECMDQ	1b0	Extended command queue is not supported.
SMMU_S_IDR1	S_SIDSIZE	32	32-bit Secure StreamIDs are supported.
	SEL2	1b1	Secure EL2 is supported.
	SECURE_IMPL	1b1	Two Security states are implemented.
SMMU_S_IDR3	SAMS	1b1	Secure ATS maintenance is not implemented.
SMMU_S_IDR4	IMPDEF	1b0	No **IMPLEMENTATION DEFINED** features.
SMMU_S_IDR6	CMDQ_CONTROL_PAGE_LOG2NUMQ	RAZ/WI	Always 256 queues per page.
	CMDQ_CONTROL_PAGE_LOG2NUMP	RAZ/WI	Number of pages set by parameter.

Register	Field	Value	Description
SMMU_R_IDR0	ATS	1	PCIe ATS supported for Realm.
	MSI	1	Realm MSIs are supported.
	PRI	1	PRI is supported.
	STALL_MODEL	0'b01	Stall not supported for Realm.
	ECMDQ	0	Enhanced Command queue is not supported.
SMMU_R_IDR1	RME_DA_IMPL	1	Realm support is implemented.
SMMU_R_IDR2	Res0	0	RES0.
SMMU_R_IDR3	DPT	0 or 1	DPT support: **0** If the TCUCFG_DPT_SUPPORT parameter is set to 0 **1** If the TCUCFG_DPT_SUPPORT parameter is set to 1
	MEC	TCUCFG_MECID_WIDTH != 0	Memory Encryption Contexts are supported.
SMMU_R_IDR4	Res0	0	RES0.
SMMU_R_MECIDR	MECIDSIZE	TCUCFG_MECID_WIDTH == 0 ? 0 : TCUCFG_MECID_WIDTH - 1;	Number of MECID bits present.

2.6.1.2 Non-implemented commands and events

MMU S3 does not implement all the SMMUv3 commands and events.

Event queue

MMU S3 does not generate the following events:

- F_UUT
- F_TLB_CONFLICT
- F_CFG_CONFLICT
- E_PAGE_REQUEST
- IMPDEF_EVENTn

Command queue

The following commands are accepted but silently ignored:

- CMD_PREFETCH_CONFIG
- CMD_PREFETCH_ADDR
- CMD_CFGI_VMS_PIDM

Only the Non-secure and Realm Command queues support the `CMD_ATC_INV` command. If the TCU encounters the `CMD_ATC_INV` command in the Secure Command queue, a Secure Command queue error with reason code `CERROR_ILL` occurs.

2.6.1.3 IMPLEMENTATION DEFINED fields

MMU S3 contains **IMPLEMENTATION DEFINED** fields.

Memory structures

For memory structure, when:

- Using the IMPDEF field, reset value
- Not using the IMPDEF field, write ignored, treated as zero

Register structures

For register structure, when:

- Using the IMPDEF field, reset value
- Not using the IMPDEF field, write ignored, treated as zero

2.6.1.4 Non-implemented registers

MMU S3 does not implement all registers.

The following optional registers are not implemented and are RAZ/WI:

- SMMU_IDR4
- SMMU_S_IDR4
- SMMU_IDR6
- SMMU_S_IDR6
- SMMU_R_IDR6
- SMMU_STATUSR
- SMMU_CMDQ_CONTROL_*
- SMMU_S_CMDQ_CONTROL_*
- SMMU_R_CMDQ_CONTROL_*
- SMMU_ECMDQ_*
- SMMU_DPT_*
- SMMU_GATOS_*
- SMMU_S_GATOS_*
- SMMU_VATOS_*
- SMMU_S_VATOS_*

The following PMCG registers are not implemented and are RAZ/WI:

- SMMU_PMCG_IIDR
- SMMU_PMCG_IRQ_CFG0
- SMMU_PMCG_IRQ_CFG1
- SMMU_PMCG_IRQ_CFG2
- SMMU_PMCG_IRQ_STATUS
- SMMU_PMCG_GMPAM
- SMMU_PMCG_MPAMIDR
- SMMU_PMCG_S_MPAMIDR

2.6.2 AMBA implementation

MMU S3 implements parts of the AMBA protocols.

2.6.2.1 ACE-Lite property support

MMU S3 supports many ACE-Lite properties.

The following table shows the ACE-Lite properties that MMU S3 supports, and their values for the TBU TBS and TBM interfaces, and the TCU QTW/DVM and PTW interfaces. The table also shows the issue of the *AMBA® AXI Protocol Specification*, that is, E, F, G, H, J, or K, to which the particular property was first added.

Table 2-39: AXI property support

Specification issue	Interface properties	ACE-Lite TBU		TCU	
		TBS	TBM	QTW/DVM	PTW
E	DVM_v8	False	False	True	False
	Multi_Copy_Atomicity	True	True	True	True
	Ordered_Write_Observation	True	True	False	False
F	Atomic_Transactions	True	True	True	True
	Barrier_Transactions	False	False	False	False
	Cache_Stash_Transactions	True, if `TBUCFG_STASH_SUPPORT` is 1	True, if `TBUCFG_STASH_SUPPORT` is 1	False	False
	Check_Type (Odd_Parity_Byte_All)	False	False	False	False
	Coherency_Connection_Signals	False	False	True	False
	DeAllocation_Transactions	True	True	False	False
	DVM_v8.1	False	False	True	False
	Loopback_Signals	True	True	False	False
	NSAccess_Identifiers	False	False	False	False
	Persist_CMO	True	True	False	False

Specification issue	Interface properties	ACE-Lite TBU		TCU	
		TBS	TBM	QTW/DVM	PTW
	Poison	True	True	True	True
	QoS_Accept	False	False	False	False
	Trace_Signals	False	False	False	False
	Untranslated_Transactions	v3	v3 The TBM implements only certain signals of this property	False	False
	Wakeup_Signals	True	True	True	True
	WriteEvict_Transaction	False	False	False	False
G	CMO_On_Read	True	True	False	False
	CMO_On_Write	False	False	False	False
	Read_Data_Chunking	True	True	False	False
	Read_Interleaving_Disabled	False The TBU is agnostic to this property	False The TBU is agnostic to this property	False The TCU is agnostic to this property	False The TCU is agnostic to this property
	Unique_ID_Support	True	True	True	True
H	Consistent_DECERR	False	False	False	False
	DVM_Message_Support	False	False	Receiver	False
	DVM_v8.4	False	False	True	False
	Exclusive_Accesses	True	True	False	False
	Max_Transaction_Bytes	IMPDEF	IMPDEF	64	64
	Prefetch_Transaction	False	False	False	False
	Regular_Transactions_Only	False	False	False	False
	Shareable_Transactions	True	True	True	True
	Write_Plus_CMO	False	False	False	False
	WriteZero_Transaction	False	False	False	False
J	BURST_Present	True	True	True	True
	Busy_Support	False	False	False	False
	CACHE_Present	True	True	True	True
	CMO_On_Write and Write_Plus_CMO (No CleanInvalidPoPA)	False	False	False	False
	DVM_v9.2	False	False	True	False
	InvalidateHint_Transaction	True	True	False	False
	LEN_Present	True	True	True	True
	Memory_Cache_Support	False	False	False	False
	MMUFLOW_Present	True	False	False	False
	PBHA_Support	True	True	True	True
	PROT_Present	True	True	True	True
	REGION_Present	True	True	False	False
	QOS_Present	True	True	True	True

Specification issue	Interface properties	ACE-Lite TBU		TCU	
		TBS	TBM	QTW/DVM	PTW
	RLAST_Present	True	True	True	True
	RME_Support	True, if TBUCFG_LEGACY_TZ_EN is 0	True, if TBUCFG_LEGACY_TZ_EN is 0	True, if TCUCFG_LEGACY_TZ_EN is 0	True, if TCUCFG_LEGACY_TZ_EN is 0
	Shareable_Cache_Support	False	False	False	False
	SIZE_present	True	True	True	True
	STASHLPID_Present	True	True	False	False
	STASHNID_Present	True	True	False	False
	UnstashTranslation_Transaction	False	False	False	False
	WLAST_Present	True	True	True	True
	WSTRB_Present	True	True	True	True
	WriteDeferrable_Transaction	False	False	False	False
K	MEC_Support	True, if TBUCFG_MECID_WIDTH > 0	True, if TBUCFG_MECID_WIDTH > 0	True, if TCUCFG_MECID_WIDTH > 0	True, if TCUCFG_MECID_WIDTH > 0
	Cache_Line_Size	IMPDEF	IMPDEF	64	64
	Device_Normal_Independence	False	False	False	False
	Fixed_Burst_Disable	False	False	False	False
	MPAM_Support	MPAM_12_1	MPAM_12_1	MPAM_12_1	MPAM_12_1
	MTE_Support	Basic	Basic	False	False
	WriteNoSnoopFull_Transaction	False	False	False	False

The following table describes the properties of the AXI widths.

Table 2-40: AXI width property descriptions

Property	Value for interface				Description
	TBS	TBM	TCU QTW/DVM	TCU PTW	
ADDR_WIDTH	64	52	52	52	Width of AxADDR signals.
ARSNOOP_WIDTH	4	4	4	4	Width of ARSNOOP signals.
AWCMO_WIDTH	0	0	0	0	N/A for SMMU S3.
AWSNOOP_WIDTH	5	5	4	4	Width of AWSNOOP signals.
BRESP_WIDTH	3	2	2	2	Width of BRESP signals.
DATA_WIDTH	TBUCFG_DATA_WIDTH	TBUCFG_DATA_WIDTH	TCUCFG_QTW_DATA_WIDTH	TCUCFG_QTW_DATA_WIDTH	Width of xDATA signals.

Property	Value for interface				Description
	TBS	TBM	TCU QTW/DVM	TCU PTW	
ID_R_WIDTH	TBUCFG_ID_WIDTH	TBUCFG_ID_WIDTH	If TCUCFG_DGW_SLOTS > TCUCFG_PTW_SLOTS and TCUCFG_DGW_SLOTS > 8, then $\log_2$(TCUCFG_DGW_SLOTS) + 3. If TCUCFG_PTW_SLOTS > 8, then $\log_2$(TCUCFG_PTW_SLOTS) + 3. Otherwise 6.	If TCUCFG_DGW_SLOTS > TCUCFG_PTW_SLOTS and TCUCFG_DGW_SLOTS > 8, then $\log_2$(TCUCFG_DGW_SLOTS) + 3. If TCUCFG_PTW_SLOTS > 8, then $\log_2$(TCUCFG_PTW_SLOTS) + 3. Otherwise 6.	Width of ARID and RID signals.
ID_W_WIDTH	TBUCFG_ID_WIDTH	TBUCFG_ID_WIDTH	If TCUCFG_DGW_SLOTS > TCUCFG_PTW_SLOTS and TCUCFG_DGW_SLOTS > 8, then $\log_2$(TCUCFG_DGW_SLOTS) + 3. If TCUCFG_PTW_SLOTS > 8, then $\log_2$(TCUCFG_PTW_SLOTS) + 3. Otherwise 6.	If TCUCFG_DGW_SLOTS > TCUCFG_PTW_SLOTS and TCUCFG_DGW_SLOTS > 8, then $\log_2$(TCUCFG_DGW_SLOTS) + 3. If TCUCFG_PTW_SLOTS > 8, then $\log_2$(TCUCFG_PTW_SLOTS) + 3. Otherwise 6.	Width of AWID and BID signals.
LOOP_R_WIDTH	If TBUCFG_OT_TRACKER_TYPE == 1 then TBUCFG_LOOP_WIDTH + 2. Otherwise TBUCFG_LOOP_WIDTH.	If TBUCFG_OT_TRACKER_TYPE == 1 then TBUCFG_LOOP_WIDTH + 2. Otherwise TBUCFG_LOOP_WIDTH.	0	0	Width of ARLOOP and RLOOP signals.
LOOP_W_WIDTH	If TBUCFG_OT_TRACKER_TYPE == 1 then TBUCFG_LOOP_WIDTH + 2. Otherwise TBUCFG_LOOP_WIDTH.	If TBUCFG_OT_TRACKER_TYPE == 1 then TBUCFG_LOOP_WIDTH + 2. Otherwise TBUCFG_LOOP_WIDTH.	0	0	Width of AWLOOP and BLOOP signals.
MECID_WIDTH	If 0 < TBUCFG_MECID_WIDTH < 16 then 16. Otherwise TBUCFG_MECID_WIDTH.	If 0 < TBUCFG_MECID_WIDTH < 16 then 16. Otherwise TBUCFG_MECID_WIDTH.	If 0 < TCUCFG_MECID_WIDTH < 16 then 16. Otherwise TCUCFG_MECID_WIDTH.	If 0 < TCUCFG_MECID_WIDTH < 16 then 16. Otherwise TCUCFG_MECID_WIDTH.	Width of AxMECID signals.

Property	Value for interface				Description
	TBS	TBM	TCU QTW/DVM	TCU PTW	
MPAM_WIDTH	If TBUCFG_LEGACY_TZ_EN == 1, then 14. Otherwise 15.	If TBUCFG_LEGACY_TZ_EN == 1, then 14. Otherwise 15.	If TCUCFG_LEGACY_TZ_EN == 1, then 14. Otherwise 15.	If TCUCFG_LEGACY_TZ_EN == 1, then 14. Otherwise 15.	Width of AxMPAM signals.
RCHUNKNUM_WIDTH	If TBUCFG_DATA_WIDTH == 128, then 8. If TBUCFG_DATA_WIDTH == 256, then 7. If TBUCFG_DATA_WIDTH == 512, then 6. If TBUCFG_DATA_WIDTH == 1024, then 5. Otherwise 1.	If TBUCFG_DATA_WIDTH == 128, then 8. If TBUCFG_DATA_WIDTH == 256, then 7. If TBUCFG_DATA_WIDTH == 512, then 6. If TBUCFG_DATA_WIDTH == 1024, then 5. Otherwise 1.	0	0	Width of RCHUNKNUM signals.
RCHUNKSTRB_WIDTH	If TBUCFG_DATA_WIDTH >= 128, then TBUCFG_DATA_WIDTH / 128. Otherwise 1.	If TBUCFG_DATA_WIDTH >= 128, then TBUCFG_DATA_WIDTH / 128. Otherwise 1.	0	0	Width of RCHUNKSTRB signals.
RRESP_WIDTH	3	2	2	2	Width of RRESP signals.
SECSID_WIDTH	If TBUCFG_LEGACY_TZ_EN ==1, then 1. Otherwise 2.	If TBUCFG_LEGACY_TZ_EN ==1, then 1. Otherwise 2.	0	0	Width of AxMMUSECSID signals.
SID_WIDTH	TBUCFG_SID_WIDTH	TBUCFG_SID_WIDTH	0	0	Width of AxMMUSID signals.
SSID_WIDTH	TBUCFG_SSID_WIDTH	TBUCFG_SSID_WIDTH	0	0	Width of AxMMUSSID signals.
SUBSYD_WIDTH	0	0	0	0	N/A for MMU S3.

Property	Value for interface				Description
	TBS	TBM	TCU QTW/DVM	TCU PTW	

USER_DATA_WIDTH	See description.	See description.	0	0	The following parameters define the width of the xUSER signals in the TBU:
					
					`TBUCFG_ARUSER_WIDTH`
					
					`TBUCFG_AWUSER_WIDTH`
					
					`TBUCFG_RUSER_WIDTH`
					
					`TBUCFG_WUSER_WIDTH`
					
					`TBUCFG_BUSER_WIDTH`
					
					For more information, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.

USER_REQ_WIDTH	See description.	See description.	0	0	The following parameters define the width of the xUSER signals in the TBU:
					
					`TBUCFG_ARUSER_WIDTH`
					
					`TBUCFG_AWUSER_WIDTH`
					
					`TBUCFG_RUSER_WIDTH`
					
					`TBUCFG_WUSER_WIDTH`
					
					`TBUCFG_BUSER_WIDTH`
					
					For more information, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.

Property	Value for interface				Description
	TBS	TBM	TCU QTW/DVM	TCU PTW	

USER_RESP_WIDTH	See description.	See description.	0	0	The following parameters define the width of the xUSER signals in the TBU:
					
					`TBUCFG_ARUSER_WIDTH`
					
					`TBUCFG_AWUSER_WIDTH`
					
					`TBUCFG_RUSER_WIDTH`
					
					`TBUCFG_WUSER_WIDTH`
					
					`TBUCFG_BUSER_WIDTH`
					
					For more information, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.

2.6.2.2 SLVERR and DECERR

MMU S3 uses SLVERR and DECERR responses. The TCU QTW and PTW interfaces treat SLVERR and DECERR identically, as an abort.

The TBU TBS interface generates SLVERR when terminating a transaction that requires an abort response.

If the TBU TBM interface receives an SLVERR or DECERR response to a downstream transaction, the same abort type is propagated to the TBS interface.

2.6.2.3 Poison

The MMU S3 TCU supports poison responses on the QTW and PTW interfaces. These are treated as an abort, but an Uncorrected error, Signaled or Recoverable error (UER) is raised.

2.6.2.4 Attribute handling

MMU S3 handles untranslated or non-ATS transactions, fully-translated or full ATS transactions, and partially-translated or Split-stage ATS transactions.

When translation is enabled and a PCIe Root Port issues transactions to a TBU, the following apply, depending on the type of transaction:

Untranslated (non-ATS) transaction

The SMMU applies attributes that a combination of the input attributes, Stream Table Entry (STE) overrides, and translation table descriptors determine.

Fully-translated (full ATS) transaction

The SMMU does not modify the attributes that are encoded in the fully translated transaction. The unmodified attributes are used as the output attributes.

Partially-translated (Split-stage ATS) transaction

The ATS translation response from the TCU to the PCIe Root Port includes Stage 1 and Stage 2 attributes.

The Stage-1-translated transaction to the TBU encodes these Stage 1 and Stage 2 attributes. The SMMU performs Stage 2 translation and combines the Stage 2 attributes a second time, but this does not affect the output attributes. The output attributes remain the same as the attributes that the TBU receives for the Stage-1-translated transaction.

NoStreamID (AxMMUVALID = 0) transaction

The SMMU does not modify the attributes that are encoded in the NoStreamID transaction. The unmodified attributes are used as output attributes.

For information about the preceding transactions and their attributes, see the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

Note

TBUs that are connected to a PCIe Root Port must have the pcie_mode input signal tied HIGH, as the table in A.2.9 TBU tie-off signals on page 331 describes.

2.6.2.4.1 TBU subordinate interface attribute handling

MMU S3 handles attribute in the subordinate interface. The TBU TBS interface converts the incoming ACE-Lite attributes into Armv8 attributes.

The following table shows how the subordinate interface handles attributes in MMU S3. The following terms are used in the table:

ISH

Inner-shareable

NSH
Non-shareable

OSH
Outer-shareable

SH
Shareable

SYS
System

MMU S3 uses the 0b10 encoding for Shareable on AXI as recommended in Domain signaling in the AMBA® AXI Protocol Specification.

Table 2-41: TBU subordinate interface attribute handling

ACE-Lite AxCACHE	ACE-Lite AxDOMAIN	Armv8 memory type	Armv8 shareability	Description
Device Non-bufferable	SYS	Device-nGnRnE	OSH	-
Device Bufferable	SYS	Device-nGnRE	OSH	-
Normal Non-cacheable Bufferable, Normal Non-cacheable Non-bufferable,	Any	Normal-iNC-oNC	OSH	Normal Non-cacheable Non-bufferable is a deprecated AxCACHE type and is converted to Normal Non-cacheable Bufferable.
Write-Through No-allocate, Write-Through Read-Allocate, Write-Through Write-Allocate, Write-Through Read and Write-Allocate	Any	Normal-iNC-oNC	OSH	Write-Through types are converted to Non-cacheable on input to match the normalization step on output.
Write-back No-allocate, Write-back Read-Allocate, Write-back Write-Allocate, Write-back Read and Write-Allocate	NSH/ISH/ OSH	Normal-iWB-oWB	NSH/OSH	The Armv8 RA and WA hints depend on the Write-Back type. The transaction is always treated as non-transient. All ISH Shareability types are converted to OSH.

Subordinate interface AxPROT handling

Instruction writes are converted into data writes on the TBU TBS interface. In effect, AWPROT[2] is ignored and always treated as 0.

2.6.2.4.2 Manager interface attribute handling

MMU S3 handles attributes in the ACE-Lite manager interfaces.

Normalization

The TBU TBM interface, TCU QTW interface, and TCU PTW interface carry normalized attributes using a standard conversion scheme:

- Memory that is marked as Inner Write-Back Cacheable and Outer Write-Back Cacheable is output as Write-Back Cacheable
- Memory that is marked as Inner Non-cacheable or Write-Through Cacheable, or Outer Non-cacheable or Write-Through Cacheable, is output as Non-cacheable, Outer Shareable

On the TBU TBM interface, a bit on AxUSER indicates whether the output memory type before this conversion is outer cacheable.

The following table shows how the Armv8 transaction types are translated into AMBA ACE-Lite signals.

Table 2-42: Armv8 transaction types translated into AMBA ACE-Lite signals

Armv8 Memory Type	AxCACHE (TBM, QTW)	AxDOMAIN (TBM, QTW)	AxLOCK (TBM)	AxUSER outer cacheable bit (TBM)
Device-nGnRnE	Device No-bufferable	SY	TBS AxLOCK value	0
Device-GRE, Device-nGRE, Device-nGnRE	Device Bufferable	SY	TBS AxLOCK value	0
Normal-iNC-oNC, Normal-iWT-oNC, Normal-iWB-oNC	Normal Non-cacheable Bufferable	SY	TBS AxLOCK value	0

Armv8 Memory Type	AxCACHE (TBM, QTW)	AxDOMAIN (TBM, QTW)	AxLOCK (TBM)	AxUSER outer cacheable bit (TBM)
Normal-iNC-oWT, Normal-iWT-oWT, Normal-iWB-oWT, Normal-iNC-oWB, Normal-iWT-oWB	Normal Non-cacheable Bufferable	SY	TBS AxLOCK value	1
Normal-iWB-oWB	Write-back No-allocate / Write-back Read-Allocate / Write-back Write-Allocate / Write-back Read and Write-Allocate, depending on the Armv8 outer allocate hints.	NSH or OSH, depending on the Armv8 Shareability **Note:** If AxBURST == FIXED and the Armv8 shareability is not NSH, an abort is generated and the transaction does not propagate downstream. The failure to propagate is because AXI does not support shareable FIXED bursts.	0	1

AxCACHE encodings

Where there are multiple legal values for AxCACHE as the *AMBA® AXI Protocol Specification* describes, the canonical, non-bracketed, one is used. Therefore, only the AxCACHE encodings that the following table shows are used.

Table 2-43: AxCACHE encodings

AMBA memory type	ARCACHE	AWCACHE
Device Non-bufferable	0000	0000
Device Bufferable	0001	0001
Normal Non-cacheable Bufferable	0011	0011
Write-back No-allocate	1011	0111
Write-back Read-Allocate	1111	0111
Write-back Write-Allocate	1011	1111
Write-back Read and Write-Allocate	1111	1111

2.6.2.5 AxREGION

MMU S3 does not require AxREGION signals.

The AxREGION signals are not required because on the:

QTW and PTW interfaces

No AxREGION signals exist

TBM interface

The AxREGION signals are passed through from the corresponding TBS interface

2.6.2.6 DVM interface

MMU S3 has a DVM interface that handles DVM operations and DVM complete.

Supported DVM operations

All DVM operations are handled in a protocol-compliant manner because the interconnect does not know that the TCU does not require DVM operations other than TLB invalidate. Any DVM operation with a DVM Message Type in ACADDR[14:12] other than TLB Invalidate or Synchronization is accepted and responded to on the CR channel but otherwise ignored.

DVM complete

DVM Complete messages are presented with:

- ARPROT[2:0] = `0b000`, that is, Unprivileged Secure Data
- ARDOMAIN[1:0] = `0b10`, that is, Outer Shareable
- ARNSE = `0b0`, that is, not Root or Realm
- ARCACHE = `0b0010`, that is, Modifiable and Non-cacheable
- ARQOS is driven by the value written to the QOS_DVMSYNC field of the 3.7.2 TCU_QOS register on page 157

2.6.2.7 Internally terminated transactions

In MMU S3, transactions that are terminated inside the TBU are returned with all RUSER and BUSER bits set to zero.

2.6.2.8 Transaction types

MMU S3 supports several special transaction types, distinguished by a nonzero encoding of AxSNOOP. We describe how each transaction type is handled.

Unless otherwise specified, transactions are propagated on TBM with the same transaction type that was presented on TBS.

An ordinary read or ordinary write is one with AxSNOOP = `0b0000`, that is, depending on AxDOMAIN, and whether it is a read or a write, one of the following:

- ReadNoSnoop
- ReadOnce
- WriteNoSnoop
- WriteUnique

For more information about the mapping of LTI transactions to AXI types and the handling of LTI responses to convert them to AXI responses, see Considerations for AXI5 in the *AMBA® LTI Protocol Specification*.

MMU S3 always treats the following as Normal Write-Back:

- ReadOnceCleanInvalid (ROCI)
- ReadOnceMakeInvalid (ROMI)
- WriteUniquePtlStash
- WriteUniqueFullStash
- StashOnceShared
- StashOnceUnique

If the incoming memory type is not Normal Write-Back then MMU S3 treats the incoming memory type as Normal Write-Back Allocate.

2.6.2.9 Transactions that can result in a translation fault

In an MMU S3 system, some transactions can result in a translation fault, and certain behavior is associated with such transactions.

MMU S3 treats the following transactions as ordinary reads when calculating translation faults:

- CleanShared
- CleanInvalid
- MakeInvalid
- CleanSharedPersist
- ReadOnceMakeInvalid
- ReadOnceCleanInvalid

Therefore, these transactions might require either read permission or execute permission at the appropriate privilege level.

MMU S3 treats the following transactions as ordinary writes when calculating translation faults:

- WriteUniquePtlStash
- WriteUniqueFullStash

Therefore, these transactions require write permission at the appropriate privilege level.

The following transactions do not have a memory type:

- CleanShared
- CleanInvalid
- MakeInvalid

- CleanSharedPersist
- InvalidateHint

For these transactions, the following input and output transaction memory type and allocation hints are used:

- Normal
- Inner Write-Back
- Outer Write-Back
- Read Allocate
- Write Allocate

This behavior means that the ARDOMAIN output on the TBM interface is never System Shareable for these transactions, because they are never Non-cacheable or Device.

MMU S3 treats transactions that pass the translation fault check as follows:

MakeInvalid transactions

MMU S3 converts MakeInvalid transactions to CleanInvalid transactions, unless the translation also grants write permission and Destructive Read Enable (DRE) permission.

InvalidateHint transactions

MMU S3 terminates InvalidateHint transactions with a RAZ/WI response, unless the translation also grants write permission and DRE permission.

ReadOnceMakeInvalid and ReadOnceCleanInvalid transactions

MMU S3 outputs ReadOnceMakeInvalid transactions as ReadOnceCleanInvalid transactions, unless the translation also granted write permission and DRE permission.

If the final transaction attributes on the TBU TBM interface are not Outer Shareable Write-Back, MMU S3 converts ReadOnceMakeInvalid and ReadOnceCleanInvalid transactions into ordinary reads.

WriteUniquePtlStash and WriteUniqueFullStash transactions

If they pass the translation fault check, MMU S3 converts WriteUniquePtlStash and WriteUniqueFullStash transactions to ordinary write transactions if either:

- The translation did not grant Directed Cache Prefetch (DCP) permission
- The final transaction attributes on the TBU TBM interface are not Outer Shareable Write-Back

If such a conversion occurs, AWSTASH* is driven as 0.

2.6.2.10 Transactions that cannot result in a translation fault

In an MMU S3 system, certain transactions cannot result in a translation fault, and certain behavior is associated with such transactions.

The following transactions never result in a translation fault:

- StashOnceShared
- StashOnceUnique
- StashTranslation
- UnstashTranslation
- InvalidateHint

If any of these transactions require a translation request to the TCU, MMU S3 issues a speculative translation request on the DTI interconnect. StashOnceShared and StashOnceUnique transactions are terminated in the TBU, with a BRESP value of OKAY, when any of the following cases apply:

- The translation did not grant Directed Cache Prefetch (DCP) permission
- The final transaction attributes on the TBM interface are not Outer Shareable Write-Back
- The translation did not grant any of read, write, or execute permission at the appropriate privilege level


Note

Only one of these permissions is required for the stash transaction to be permitted.

A BRESP value of OKAY indicates transaction success. MMU S3 always generates this value when a StashOnceShared or a StashOnceUnique transaction is terminated in the TBU. This behavior applies even when a StreamDisable or GlobalDisable translation response causes the transaction to be terminated.

MMU S3 never propagates StashTranslation transactions downstream, and uses StashTranslation only to prefetch Main TLB contents. MMU S3 always terminates StashTranslation transactions with a BRESP value of OKAY, even if no translation could be stored in the Main TLB.

The TBU ignores AWPROT[0] and AWPROT[2] for StashTranslation transactions, because they do not affect speculative translation requests.


Note

A StashTranslation transaction can be used to prefetch translations into the Main TLB of MMU S3. However, for this prefetching to be useful, any subsequent transactions that intend to take advantage of the translations that have been prefetched into the Main TLB must use the same StreamID as the original prefetch. The StreamID identifies a translation context. Using a different StreamID, SECSID, SSV or SubstreamID for a subsequent transaction means that this subsequent transaction uses a different translation context to the translation that has been prefetched into the Main TLB and might lead to a TLB miss.

2.6.3 Local Translation Interface implementation

MMU S3 implements the Local Translation Interface (LTI), with certain properties.

For more information, see the *AMBA® LTI Protocol Specification*.

The following table shows the values of the LTI properties.

Table 2-44: LTI properties

Name	Value	Description
LTI_VC_COUNT	2	Two LTI Virtual channels are chosen, one for read and one for write
LTI_ID_WIDTH	TBUCFG_ID_WIDTH	Equal to ID width of incoming transaction
LTI_MECID_WIDTH	See description	Width of the MECID that is supported. The value of LTI_MEC_WIDTH is calculated as follows: (TBUCFG_MECID_WIDTH == 0) ? 0 : 16 **Note:** When LTI_MECID_WIDTH == 0, the lrmecid signals are 1-bit, unused, and tied-off internally.
LTI_SID_WIDTH	TBUCFG_SID_WIDTH	Equal to width of incoming SID
LTI_SSID_WIDTH	TBUCFG_SSID_WIDTH	Equal to width of incoming SSID
LTI_OG_WIDTH	TBUCFG_LTI_OG_WIDTH	Equal to width of incoming ordering groups
LTI_TLBLOC_WIDTH	See Description	Width of TLB location, in bits. The value of the LTI_TLBLOC_WIDTH parameter, which is a local parameter, is calculated as follows: LTI_TLBLOC_WIDTH == MAX(1, ((TBUCFG_DIRECT_IDX == 1) ? ((TBUCFG_MTLB_DEPTH > 0) ? $\log_2$(TBUCFG_MTLB_DEPTH) : $\log_2$(4)) : $\log_2$(TBUCFG_MTLB_PARTS)))
LTI_LOOP_WIDTH	See Description	Width of the LTI loopback signals, in bits. For the LTI TBU, the value is equal to TBUCFG_LTI_LOOP_WIDTH. For the ACE-Lite TBU, the value is calculated automatically.
LTI_LAUSER_WIDTH	0	LTI user signals are not used
LTI_LRUSER_WIDTH	0	
LTI_LCUSER_WIDTH	0	
LTI_GPC	~TBUCFG_LEGACY_TZ_EN	GPC is supported when not in legacy mode. **Note:** When LTI_GPC == 0, the lanse and lrnsesignals are 1-bit, unused, and tied-off internally.
LTI_MMU	True	Stage 1 and 2 translation always supported.
LTI_LRADDR_WIDTH	52	Width of translated address.

2.6.4 MPAM implementation

MMU S3 implements Memory System Resource Partitioning and Monitoring (MPAM) with some constraints and limitations. MMU S3 implements some MPAM registers. Registers that are not implemented are not described.

For more information, see the *Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification*.

MPAM cache maximum capacity partitioning manages the following:

- TBU MTLB
- TCU configuration cache
- TCU Walk cache
- Granule Protection Table (GPT) caches
- Device Permission Table (DPT) cache

No other mechanism from the *Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification* is implemented.

MPAM and NoStreamID

For NoStreamID transactions, where AxMMUVALID = 0, the following values are used for MPAM:

PARTID 0
PMG 0
MPAM_SP Match the target PAS of the access

2.6.4.1 TCU MPAM

MMU S3 enables you to implement TCU Memory System Resource Partitioning and Monitoring (MPAM). Internally, Resource Instance Selector (RIS) is truncated to different values depending on whether Realm Management Extension (RME) mode is enabled or not, and on the security state. However, externally RIS is the normal 4 bits.

RME mode not enabled

When RME mode is not enabled, the following RIS values are used, and internally, RIS is truncated to 1 bit and the maximum legal RIS value is 1.

The RIS value cache is as follows:

0b0
WCB

0b1
CCB

RME mode enabled

When Realm Management Extension (RME) mode is enabled:

- If Device Permission Table (DPT) checks are supported, that is, the `TCUCFG_DPT_SUPPORT` parameter is set to 1, and Physical Address Space (PAS) is Realm or Non-secure, the value of RIS is 3 bits.
- If DPT is not supported, that is, the `TCUCFG_DPT_SUPPORT` parameter is set to 0, or PAS == Secure, the value of RIS is truncated to 2 bits internally.
- Externally, the value of RIS is 4 bits, but because RIS is truncated internally, no illegal RIS handling occurs.

The *Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification* specifies constrained, predictable behavior of RIS when set above RIS_MAX or to a RIS value of a non-existent Memory-System Component (MSC).

When PAS == Root, then RIS == 0, because DGW GCB is the only resource that is available for Root MPAM.

If DPT checks are supported, the RIS value for the Non-secure and Realm worlds is 3 bits internally and the maximum legal RIS value is 4, as the following table shows.

Table 2-45: DPT supported, for Non-secure and Realm, RIS is 3 bits internally

RIS value	Cache
0b000	Walk Cache Block (WCB).
0b001	Configuration Cache Block (CCB).
0b010	GCB in DGW, that is, GPT Cache Block (GCB) in DTI GPC Wrapper (DGW), where: • GPC is Granule Protection Check (GPC) • GPT is Granule Protection Table (GPT)
0b011	Translation Management Unit (TMU) AXI GPC Wrapper (AGW) GCB.
0b100	Device Protection Table (DPT) Cache Block (DCB).
0b111-0b101	Illegal.

If DPT checks are not supported, the RIS value for the Non-secure and Realm worlds is 2 bits internally and the maximum legal RIS value is 4, as the following table shows.

Table 2-46: DPT not supported, for Non-secure and Realm, RIS is 2 bits internally

RIS value	Cache
0b00	Walk Cache Block (WCB).
0b01	Configuration Cache Block (CCB).
0b10	GPT Cache Block (GCB) in DTI GPC Wrapper (DGW), where: • GPC is Granule Protection Check (GPC). • GPT is Granule Protection Table (GPT).
0b11	TMU AGW GCB.

The RIS value for the Secure world is 2 bits internally and the legal maximum RIS value is 3, as the following table shows.

Table 2-47: For Secure, RIS is 2 bits internally

RIS value	Cache
0b00	WCB
0b01	CCB
0b10	GCB in DGW
0b11	GCB in TMU AGW

TCU MPAM registers

Four copies of these registers exist in the TCU memory map. See the table named TCU PMCG, RAS, and MPAM register allocation to regions of TCU address space in 3.3.1 TCU memory map on page 138. The following tables show the values of these registers as they appear in the Register page relating to each PAS. When the TCUCFG_LEGACY_TZ_EN parameter is set to 1 or the legacy_tie_off signal is set to 1, the Root and Realm copies of these registers are RAZ/WI. The following tables show the TCU MPAM registers and register fields that are implemented. Each table describes a TCU MPAM register and shows the bit fields for each, and the values for Secure, Non-Secure, Realm, and Root.

Table 2-48: MPAMF_IDR_LO register, 0x0000, Shared

Field	Value			Description
	S, NS	Root	Realm	
PARTID_MAX	1, 63, 511, 1023	0	1, 63, 511, 1023	Set to: ($2^{TCUCFG\_PARTID\_WIDTH}$ - 1)
PMG_MAX	1	0	1	Two Non-secure Performance Monitoring groups are supported per PARTID.
HAS_CCAP_PART	1	1	1	Cache maximum capacity partitioning is supported.
HAS_CPOR_PART	0	0	0	Cache portion partitioning is not supported.
HAS_MBW_PART	0	0	0	Memory Bandwidth partitioning is not supported.
HAS_PRI_PART	0	0	0	Priority partitioning is not supported.
EXT	1	1	1	EXTended MPAMF_IDR.
HAS_IMPL_IDR	0	0	0	Does not have **IMPLEMENTATION-SPECIFIC** partitioning features.
HAS_MSMON	1	1	1	Supports performance monitoring by matching a combination of PARTID and PMG.
HAS_PARTID_NRW	0	0	0	Does not have MPAMF_PARTID_NRW_IDR, MPAMCFG_INTPARTID, or intPARTID mapping support.

Table 2-49: MPAMF_IDR_HI register, 0x0004, Shared

Field	Value			Description
	S, NS	Root	Realm	
HAS_RIS	1	0	1	Has Resource Instance Selector (RIS).
HAS_EXTD_ESR	0	0	0	MPAMF_ESR is 64-bits. Not relevant because HAS_ESR is 0.

Field	Value			Description
	S, NS	Root	Realm	
HAS_ESR	0	0	0	MPAMF_ESR and MPAMF_ECR are not implemented.
HAS_NFU	0	0	0	-
HAS_ENDIS	0	0	0	-
RIS_MAX	1, 3, 4	0	3, 4	Maximum RIS value used in the Memory-System Component (MSC).
SP4	0, 1	1	1	We support four security states: **0** If the `TCUCFG_LEGACY_TZ_EN` parameter is set to 1 or the legacy_tz_en tie-off signal is set to 1. **1** Otherwise.

Table 2-50: MPAMF_SIDR register, 0x0008, Secure only

Field	Value for Secure	Description
S_PARTID_MAX	1, 63, 511, 1023	Set to: ($2^{\texttt{TCUCFG\_PARTID\_WIDTH}}$ - 1)
S_PMG_MAX	1	Two Secure Performance Monitoring groups are supported per PARTID.

Table 2-51: MPAMF_IIDR register, 0x0018, Shared

Field	Value			Description
	S, NS	Root	Realm	
All fields	• product ID = 0x498 • variant = 1 • revision = MAX (ecorevnum, P_LEVEL), where P_LEVEL is 1.			Implementation ID Register.

Table 2-52: MPAMF_AIDR register, 0x0020, Shared

Field	Value			Description
	S, NS	Root	Realm	
ArchMajorRev	1	1	1	MPAM architecture v1.1
ArchMinorRev	1	1	1	MPAM architecture v1.1

Table 2-53: MPAMF_CCAP_IDR register, 0x0038, Shared

Field	Value			Description
	S, NS	Root	Realm	
CMAX_WD	min ($\log_2$(Depth/Banks), 8)			Up to 256 fractions are supported.

Table 2-54: MPAMF_MSMON_IDR register, 0x0080, Shared

Field	Value			Description
	S, NS	Root	Realm	
MSMON_CSU	1	1	1	Performance monitor is supported for Cache Storage Usage by PARTID and PMG.
MSMON_MBWU	0	0	0	No performance monitor for Memory Bandwidth Usage by PARTID and PMG.
HAS_LOCAL_CAPT_EVNT	1	1	1	Has the local capture event generator and the MSMON_CAPT_EVNT register.

Field	Value			Description
	S, NS	Root	Realm	
NO_HW_OFLW_INTR	1	1	1	-
HAS_OFLW_MSI	0	0	0	-
HAS_OFLOW_SR	0	0	0	-

Table 2-55: MPAMF_CSUMON_IDR register, 0x0088, Shared

Field	Value			Description
	S, NS	Root	Realm	
NUM_MON	4	1	4	Four monitoring counters are implemented. For the Root world, one monitoring counter is implemented.
HAS_CAPTURE	1	1	1	Has an MSMON_CSU_CAPTURE register for every MSMON_CSU and supports the capture event behavior.
CSU_RO	1	1	1	-

Table 2-56: MPAMCFG_PART_SEL register, 0x0100, Banked

Field	Value			Description
	S, NS	Root	Realm	
PARTID_SEL	[TCUCFG_PARTID_WIDTH - 1:0] bits are valid	This field is ignored	[TCUCFG_PARTID_WIDTH - 1:0] bits are valid	You can select up to 1024 partitions to configure, based on the value of the TCUCFG_PARTID_WIDTH parameter.
RIS	For information about the RIS value, see the information earlier in this section.			Resource Instance Selector (RIS).

Table 2-57: MPAMCFG_CMAX register, 0x0108, Banked

Field	Value			Description
	S, NS	Root	Realm	
CMAX	Width depends on configuration, value is 16-bit aligned. Configure PARTID_SEL before you attempt to update CMAX information for any PARTID.			You can select up to 256 fractions.

Table 2-58: MSMON_CFG_MON_SEL register, 0x0800, Banked

Field	Value			Description
	S, NS	Root	Realm	
MON_SEL	Bits [1:0] are valid, and other bits are ignored.	This field is ignored.	Bits [1:0] are valid, and other bits are ignored.	Selects the monitor to configure.
RIS	For information about the RIS value, see the information earlier in this section.	This field is ignored	For information about the RIS value, see the information earlier in this section.	Resource Instance Selector (RIS).

Table 2-59: MSMON_CFG_CSU_FLT register, 0x0810, Banked

Field	Value			Description
	S, NS	Root	Realm	
PARTID	[TCUCFG_PARTID_WIDTH - 1:0] bits are valid	A value of 0 is supported.	[TCUCFG_PARTID_WIDTH - 1:0] bits are valid	You can select up to 1024 partitions to configure, based on the value of the TCUCFG_PARTID_WIDTH parameter.
PMG	Bit [16] are valid	A value of 0 is supported.	Bit [16] are valid	You can select up to PMG number 1. **Note:** PMG does not exist for the Root world.

Table 2-60: MSMON_CFG_CSU_CTL register, 0x0818, Banked

Field	Value			Description
	S, NS	Root	Realm	
EN	Valid field			The monitor instance is enabled or disabled to collect information.
CAPT_EVNT	0b111			Capture occurs when a MSMON_CAPT_EVNT register is written. No other values are supported.
CAPT_RESET	RES0			There is never a reason to reset a CSU monitor.
OFLOW_STATUS	RES0			Overflow is not possible for a CSU monitor.
OFLOW_INTR	RES0			This MPAM implementation does not support OFLOW_INTR.
OFLOW_FRZ	RES0			Overflow is not possible for a CSU monitor.
SUBTYPE	RES0			This field is reserved for future use.

Table 2-61: MSMON_CSU register, 0x0840, Banked

Field	Value			Description
	S, NS	Root	Realm	
All fields	All fields are valid			Cache storage usage value.

Table 2-62: MSMON_CSU_CAPTURE register, 0x0848, Banked

Field	Value			Description
	S, NS	Root	Realm	
All fields	All fields are valid			Capture cache storage usage.

Table 2-63: MSMON_CAPT_EVNT register, 0x0808, Banked

Field	Value			Description
	S, NS	Root	Realm	
All fields	All fields are valid			Capture event.

2.6.4.2 TBU MPAM

MMU S3 enables you to implement TBU Memory System Resource Partitioning and Monitoring (MPAM). The TBU Memory-System Component (MSC) has only one resource, MTLB, and does not require a Resource Instance Selector (RIS).

If the `TBUCFG_MTLB_DEPTH` parameter is set to 0, the resource is not present, and if the `TBUCFG_DIRECT_IDX` parameter is set to 1, the resource is present but does not have MPAM controls. The associated ID registers must report values of limited control under these circumstances. Therefore, many non-ID control registers are RAZ/WI in such circumstances.

The following tables show the TBU MPAM registers and register fields that are implemented. Each table describes a TBU MPAM register and shows the bit fields and the permitted values for each. Four copies of these registers exist in the 3.3.2 TBU memory map on page 140. The following tables show the values of these registers as they appear in the Register page relating to each Physical Address Space (PAS). If the value is different in a page relating to a certain PAS, the values are listed for each PAS. For registers that have different values depending on whether the resource is present and has MPAM controls, the different values are shown in different columns.

All MPAM registers apart from the following are RAZ/WI if the resource is not present or does not have MPAM controls:

- MPAMF_IDR_LO
- MPAMF_IDR_HI
- MPAMF_SIDR
- MPAMF_AIDR
- MPAMF_IIDR

If the `TBUCFG_LEGACY_TZ_EN` parameter is set to 1 or the legacy_tz_en tie-off signal is set to 1, the register pages containing the copies of these registers, that relate to the Root and Realm worlds, are RAZ/WI.

Table 2-64: MPAMF_IDR_LO, 0x0000, Shared

Field	Value		Description
	Resource present and has MPAM controls	Resource not present or has no MPAM controls	
PARTID_MAX	1, 63, 511, 1023 (0 for Root)	1, 63, 511, 1023 (0 for Root)	Set to: ($2^{\texttt{TBUCFG\_PARTID\_WIDTH}}$ - 1)
PMG_MAX	1 (0 for Root)	1 (0 for Root)	Two Non-secure Performance Monitoring groups are supported for each PARTID.
HAS_CCAP_PART	1	0	Cache maximum capacity partitioning is supported.
HAS_CPOR_PART	0	0	Cache portion partitioning is not supported.
HAS_MBW_PART	0	0	Memory bandwidth partitioning is not supported.
HAS_PRI_PART	0	0	Priority partitioning is not supported.
EXT	1	1	EXTended MPAMF_IDR.

Field	Value		Description
	Resource present and has MPAM controls	Resource not present or has no MPAM controls	
HAS_IMPL_IDR	0	0	Does not have **IMPLEMENTATION-SPECIFIC** partitioning features.
HAS_MSMON	1	0	Supports performance monitoring by matching a combination of PARTID and PMG.
HAS_PARTID_NRW	0	0	Does not have MPAMF_PARTID_NRW_IDR, MPAMCFG_INTPARTID, or intPARTID mapping support.

Table 2-65: MPAMF_IDR_HI, 0x0004, Shared

Field	Value		Description
	Resource present and has MPAM controls	Resource not present or has no MPAM controls	
HAS_RIS	0	0	Does not have a Resource Instance Selector (RIS).
HAS_EXTD_ESR	0	0	MPAMF_ESR is 64-bits. Not relevant because HAS_ESR is 0.
HAS_ESR	0	0	MPAMF_ESR and MPAMF_ECR are not implemented.
HAS_NFU	0	0	-
HAS_ENDIS	0	0	-
SP4	0 or 1 for Secure and Non-secure, 1 for Ream and Root	0 or 1 for Secure and Non-secure, 1 for Realm and Root	For the copies of this register that exist in the register page relating to the Non-Secure and Secure worlds, the value can be 0 or 1 depending on the following: **0** If the TBUCFG_LEGACY_TZ_EN parameter is set to 1 or the legacy_tz_en tie-off signal is set to 1. **1** Otherwise.

Table 2-66: MPAMF_SIDR, 0x0008, Secure only

Field	Value	Description
S_PARTID_MAX	1, 63, 511, 1023	Set to: ($2^{TBUCFG\_PARTID\_WIDTH}$ - 1)
S_PMG_MAX	1	Two Secure Performance Monitoring groups supported per PARTID.

Table 2-67: MPAMF_IIDR, 0x0018, Shared

Field	Value	Description
product ID	0x499	Implementation ID Register field.
variant	1	Implementation ID Register field.
revision	MAX (ecorevnum,P_LEVEL), where P_LEVEL is 1.	Implementation ID Register field.

Table 2-68: MPAMF_AIDR, 0x0020, Shared

Field	Value	Description
ArchMajorRev	1	MPAM architecture v1.1.
ArchMinorRev	1	MPAM architecture v1.1.

The following registers are all RAZ/WI if the resource is not present or does not have MPAM controls.

Table 2-69: MPAMF_CCAP_IDR, 0x0038, Shared

Field	Value	Description
CMAX_WD	min(log_2(Depth/Banks),8)	Implementation ID Register.

Table 2-70: MPAMF_MSMON_IDR, 0x0080, Shared

Field	Value	Description
MSMON_CSU	1	Performance monitor is supported for Cache Storage Usage by PARTID and PMG.
MSMON_MBWU	0	No performance monitor for Memory Bandwidth Usage by PARTID and PMG.
HAS_LOCAL_CAPT_EVNT	1	Has the local capture event generator and the MSMON_CAPT_EVNT register.
NO_HW_OFLW_INTR	1	-
HAS_OFLW_MSI	0	-
HAS_OFLOW_SR	0	-

Table 2-71: MPAMF_CSUMON_IDR, 0x0088, Shared

Field	Value	Description
NUM_MON	4, or 1 for Root	Four monitoring counters are implemented. For the Root world, one monitoring counter is implemented.
HAS_CAPTURE	1	Has an MSMON_CSU_CAPTURE register for every MSMON_CSU and supports the capture event behavior.
CSU_RO	1	-

Table 2-72: MPAMCFG_PART_SEL, 0x0100, Banked

Field	Value	Description
PARTID_SEL	**Non-secure, Secure, and Realm** Bits [(`TBUCFG_PARTID_WIDTH` - 1):0] valid. **Root** This field is ignored.	You can select up to 1024 partitions to configure, based on the value of the `TBUCFG_PARTID_WIDTH` parameter. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.

Table 2-73: MPAMCFG_CMAX, 0x0108, Banked

Field	Value	Description
CMAX	The width depends on the configuration, and the value is 16-bit aligned.	You can select up to 256 fractions. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.

Table 2-74: MSMON_CFG_MON_SEL, 0x0800, Banked

Field	Value	Description
MON_SEL	**Non-secure, Secure, and Realm** Bits [1:0] are valid, and other bits are ignored. **Root** This field is ignored.	Selects the monitor to configure. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.

Table 2-75: MSMON_CAPT_EVNT, 0x0808, Banked

Field	Value	Description
All fields	All fields are valid.	Capture event. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.

Table 2-76: MSMON_CFG_CSU_FLT, 0x0810, Banked

Field	Value			Description
	S, NS	Root	Realm	
PARTID	Bits [(TBUCFG_PARTID_WIDTH - 1):0] are valid.	A value of 0 is supported.	Bits [(TBUCFG_PARTID_WIDTH - 1):0] are valid.	You can select up to 1024 partitions to configure, based on the value of the TBUCFG_PARTID_WIDTH parameter. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.
PMG	Bit [16] is valid.	A value of 0 is supported.	Bit [16] is valid.	You can select up to PMG number 1. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.

Table 2-77: MSMON_CFG_CSU_CTL, 0x0818, Banked

Field	Value	Description
EN	Valid field.	The monitor instance is enabled or disabled to collect information. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.
CAPT_EVNT	3'b111	Capture occurs when a MSMON_CAPT_EVNT register is written. No other values are supported. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.
CAPT_RESET	RES0	There is never a reason to reset a CSU monitor. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.
OFLOW_STATUS	RES0	Overflow is not possible for a CSU monitor. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.
OFLOW_INTR	RES0	This MPAM implementation does not support OFLOW_INTR. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.
OFLOW_FRZ	RES0	Overflow is not possible for a CSU monitor. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.
SUBTYPE	RES0	This field is reserved for future use. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.

Table 2-78: MSMON_CSU, 0x0840, Banked

Field	Value	Description
All fields	All fields are valid.	Cache storage usage value. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.

Table 2-79: MSMON_CSU_CAPTURE, 0x0848, Banked

Field	Value	Description
All fields	All fields are valid.	Capture cache storage usage. When direct indexing or direct partitioning is enabled, this field does not reflect any meaningful value.

2.7 Configuration parameters and methodology

The TBU, TCU, and BAS components in MMU S3 are delivered as SystemVerilog that you can parameterize. Use the *Generate* script to configure these components. There are several versions of the switch component, part of the BAS components that are delivered, to accommodate different numbers of subordinate interfaces.

For more information about the *Generate* script and detailed descriptions of parameters, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

2.7.1 TCU I/O and buffer configuration parameters

You can configure the Translation Control Unit (TCU) I/O and buffer.


Tip

For more detailed descriptions of these configuration parameters, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

The following table shows the TCU buffer configuration parameters.

Table 2-80: TCU buffer configuration parameters

Parameter name	Values	Description
TCUCFG_QTW_DATA_WIDTH	64, 128, 256, 512	ACE-Lite+DVM Queue and Table Walk (QTW) and ACE-Lite Page Table Walk (PTW) interface data width.
TCUCFG_CC_DEPTH	8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096	Configuration cache depth, in entries.
TCUCFG_CC_LKP_SLOTS	2-16	Number of lookup slots that the configuration cache uses.
TCUCFG_WC_DEPTH	8, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536	Walk cache depth, in entries.
TCUCFG_WC_BANKS	1, 2, 4	Number of banks in walk cache.

Parameter name	Values	Description
TCUCFG_WC_WAYS	4, 8, 16	Number of ways in walk cache.
TCUCFG_CC_WAYS	4, 8, 16	Number of ways in configuration cache.
TCUCFG_AGW_GC_DEPTH	8, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536	AXI Granule Protection Check (GPC) Wrapper (AGW) Granule Protection Table (GPT) cache depth, in entries.
TCUCFG_AGW_GC_BANKS	1, 2, 4	Number of banks in AGW GPT cache.
TCUCFG_AGW_GC_WAYS	4, 8, 16	Number of ways in AGW GPT cache.
TCUCFG_DGW_GC_DEPTH	8, 64, 128, 256, 512, 1024,2048, 4096, 8192, 16384, 32768, 65536	DGW GPT cache depth, in entries.
TCUCFG_DGW_GC_BANKS	1, 2, 4	Number of banks in DGW GPT cache.
TCUCFG_DGW_GC_WAYS	4, 8, 16	Number of ways in DGW GPT cache.
TCUCFG_NUM_TBU	14, 62	Maximum number of DTI requesters, that is, DTI-TBU and DTI-ATS requesters, that the TCU supports. The value is two less than 16 or 64 to better fit into system memory maps.
TCUCFG_XLATE_SLOTS	4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096	Total permitted translation requests from all DTI requesters.
TCUCFG_PTW_SLOTS	2, 4, 8, 16, 32, 64, 128, 256, 512	Number of parallel translation table walks.
TCUCFG_CTW_SLOTS	1, 2, 4.	Number of parallel configuration table walks.
TCUCFG_DGW_SLOTS	8, 16, 32, 64, 128, 256, 512	Number of GPC walks slots in DGW.
TCUCFG_WC_LKP_SLOTS	2-32	Number of lookup slots that the walk cache uses.
TCUCFG_AGW_GC_LKP_SLOTS	2-32	AXI Granule Protection Check (GPC) Wrapper (AGW) Granule Protection Table (GPT) cache lookup slots. The number of lookup slots that the AGW GPT cache uses.
TCUCFG_DGW_GC_LKP_SLOTS	2-32	DTI Granule Protection Check (GPC) Wrapper (DGW) Granule Protection Table (GPT) cache lookup slots. The number of lookup slots that the DGW GPT cache uses.
TCUCFG_DC_DEPTH	8, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536	DPT cache depth in entries.
TCUCFG_DC_BANKS	1, 2, 4	Number of banks in the DPT cache.
TCUCFG_DC_WAYS	4, 8, 16	Number of ways in the DPT cache.
TCUCFG_DC_LKP_SLOTS	2-32	Number of lookup slots that the DPT cache uses.
TCUCFG_LEGACY_TZ_EN	0, 1	Disable Realm Management Extension features and switch back to two security states. Secure and Non-secure remains, Realm and Root are removed. **Note:** The value of the TCUCFG_LEGACY_TZ_EN parameter must be the same as the value of TBUCFG_LEGACY_TZ_EN parameter **Note:** If the TCUCFG_DPT_SUPPORT parameter is set to 1, then the TCUCFG_LEGACY_TZ_EN parameter must be set to 0.

Parameter name	Values	Description
TCUCFG_PARITY_ON	0, 1	Enables parity protection on large flop structures and configuration registers.
TCUCFG_MECID_WIDTH	0, 4, 8, 12, 16	Width of the MECID that is supported, which is the number of bits of MECID that are supported internally. **Note:** The value of the TCUCFG_MECID_WIDTH parameter must be the same as the value of TBUCFG_MECID_WIDTH parameter.
TCUCFG_CC_IDXGEN_MODE	0, 1	Index generation mode for the configuration cache.
TCUCFG_DTI_ATS	0, 2, 4, 8, 16	Maximum number of DTI-ATS requesters.
TCUCFG_DTI_ATS_INV_MAX	2, 4, 8, 16, 32	Maximum number of ATS invalidate messages which can be outstanding per manager without a DTI_ATS_INV_COMP response.
TCUCFG_PMU_COUNTERS	4, 16, 32	Number of PMU counters.
TCUCFG_PARTID_WIDTH	1, 6, 9, 10	Width of PARTID that is supported.
TCUCFG_HZU_DEPTH	2, 4, 8, 16, 32, 64.	From MMU S3 version r1p0 onwards, TCU hazarding is no longer supported. Set the value of the TCUCFG_HZU_DEPTH parameter to the minimum value of 2.
TCUCFG_PREFETCH_SUPPORTED	0, 1	Specifies whether prefetch is supported.
TCUCFG_DATARAM_TYPE	0, 1, 2	RAM type for data group of RAMs.
TCUCFG_SLOTRAM_TYPE	0, 1, 2	RAM type for slot group of RAMs.
TCUCFG_CACHERAM_TYPE	0, 1	RAM type for cache group of RAMs.
TCUCFG_DVM_VAS	49 or 53	Virtual address size used by the system. The INV block uses the TCUCFG_DVM_VAS parameter to correctly form the invalidate command. internally.
TCUCFG_DPT_SUPPORT	0, 1	Enables Device Permission Table (DPT) checks. Do not enable this parameter unless you enable the Realm Management Extension (RME) feature.

2.7.2 TCU debug configuration parameters

You can configure the Translation Control Unit (TCU) debug parameters.


Tip

For more detailed descriptions of these configuration parameters, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

The following table shows the TCU debug configuration parameters.

Table 2-81: TCU debug configuration parameters

Parameter name	Values	Description
TCUCFG_USE_ELA_DEBUG	0, 1	Determines whether ELA debug is used or not.

2.7.3 Common ACE-Lite and LTI TBU configuration parameters

You can configure the common Local Translation Interface (LTI) Translation Buffer Unit (TBU) and ACE-Lite TBU parameters.


Tip

For more detailed descriptions of these configuration parameters, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

The following table shows the ACE-Lite TBU and LTI TBU configuration parameters.

Table 2-82: ACE-Lite TBU and LTI TBU configuration parameters

Parameter name	Values	Description
TBUCFG_SID_WIDTH	8, 16, 20, 24	Stream ID width.
TBUCFG_SSID_WIDTH	1, 8, 20	SubstreamID width.
TBUCFG_DIRECT_IDX	0, 1	Direct indexing enabled.
TBUCFG_MTLB_PARTS	1, 2, 4, 8, 16	Number of Main TLB partitions.
TBUCFG_LTI_OG_WIDTH	1-5	LTI ordering group width.
TBUCFG_LA_HNDSHK_MODE	0, 1, 2, 3	Register slice configuration on address channel before TLB lookup.
TBUCFG_LR_HNDSHK_MODE	0, 1, 2, 3	Register slice configuration on translation response path.
TBUCFG_XLATE_SLOTS	2, 4, 8, 16, 32, 64, 128, 256, 512, 1024	Number of translation slots, controlling the Hit-Under-Miss capability of the TBU.
TBUCFG_MTLB_LKP_SLOTS	2-32	Number of MTLB lookup slots.
TBUCFG_MTLB_UPD_SLOTS	2, 4, 8	Number of MTLB update slots.
TBUCFG_UTLB_DEPTH	4, 8, 12, 16, 32, 64	MicroTLB depth, in entries.
TBUCFG_MTLB_DEPTH	0, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536	Main TLB (MTLB) depth, in entries.
TBUCFG_MTLB_WAYS	4, 8, 16	Number of ways in the MTLB.
TBUCFG_MTLB_BANKS	1, 2, 4	Number of banks in the MTLB.
TBUCFG_PMU_COUNTERS	4, 16, 32	Number of PMU counters.
TBUCFG_PARTID_WIDTH	1, 6, 9, 10	Width of PARTID that is supported.
TBUCFG_MECID_WIDTH	0, 4, 8, 12, 16	Width of the MECID that is supported, which is the number of bits of MECID that are supported internally. **Note:** The value of the TBUCFG_MECID_WIDTH parameter must be the same as the value of TCUCFG_MECID_WIDTH parameter
TBUCFG_HZRD_ENTRIES	0, 4, 8, 16, 32, 64	Number of hazard entries.
TBUCFG_SLOTRAM_TYPE	0, 1, 2	RAM type for slot group of RAMs.
TBUCFG_CACHERAM_TYPE	0, 1	RAM type for cache group of RAMs.

Parameter name	Values	Description
TBUCFG_LEGACY_TZ_EN	0, 1	Disable Realm Management Extension (RME) features and switch back to 2-security states. Secure and Non-secure remains, Realm and Root are removed. **Note:** The value of the TBUCFG_LEGACY_TZ_EN parameter must be the same as the value of TCUCFG_LEGACY_TZ_EN parameter
TBUCFG_PARITY_ON	0, 1	Enables parity protection on large flop structures and configuration registers.

2.7.4 ACE-Lite TBU register slice configuration parameters

You can configure the Translation Buffer Unit (TBU) register slices.


Tip

For more detailed descriptions of these configuration parameters, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

The following table shows the ACE-Lite TBU register slice configuration parameters.

Table 2-83: ACE-Lite TBU register slice configuration parameters

Parameter name	Value	Description
TBUCFG_SI_AR_HNDSHK_MODE	0, 1, 2, 3	Register slice configuration parameters. You can configure the register slice to be one of the following: • Fully registered • Registered on the forward path only • Registered on the reverse path only • Bypassed
TBUCFG_SI_R_HNDSHK_MODE	0, 1, 2, 3	
TBUCFG_SI_AW_HNDSHK_MODE	0, 1, 2, 3	
TBUCFG_SI_W_HNDSHK_MODE	0, 1, 2, 3	
TBUCFG_SI_B_HNDSHK_MODE	0, 1, 2, 3	
TBUCFG_MI_AR_HNDSHK_MODE	0, 1, 2, 3	
TBUCFG_MI_R_HNDSHK_MODE	0, 1, 2, 3	
TBUCFG_MI_AW_HNDSHK_MODE	0, 1, 2, 3	
TBUCFG_MI_W_HNDSHK_MODE	0, 1, 2, 3	
TBUCFG_MI_B_HNDSHK_MODE	0, 1, 2, 3	

2.7.5 ACE-Lite TBU I/O configuration parameters

You can configure the ACE-Lite Translation Buffer Unit (TBU) I/O.


Tip

For more detailed descriptions of these configuration parameters, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

The following table shows the ACE-Lite TBU I/O configuration parameters that apply to both the TBS and TBM interfaces.

Table 2-84: ACE-Lite TBU I/O configuration parameters

Parameter name	Values	Description
TBUCFG_ID_WIDTH	1-32	AXI ID width
TBUCFG_DATA_WIDTH	64, 128, 256, 512, 1024	AXI data width
TBUCFG_ARUSER_WIDTH TBUCFG_AWUSER_WIDTH TBUCFG_RUSER_WIDTH TBUCFG_WUSER_WIDTH TBUCFG_BUSER_WIDTH	1-128	AXI USER bus widths
TBUCFG_STASH_SUPPORT	0, 1	Defines whether stash ID signals are used. When configured as 0, stash ID signals on the subordinate interface are ignored, stash ID signals on the manager interface are driven to 0, and AXI Cache_Stash_Transactions are not supported.
TBUCFG_LOOP_WIDTH	1-8	AXI loopback signal width
TBUCFG_WBUF_DEPTH	0, 8, 16, 32, 64, 128, 256, 512, 1024, 2048	Write buffer depth.
TBUCFG_LFIFO_DEPTH	0, 4	Latency FIFO depth.
TBUCFG_OT_TRACKER_TYPE	0, 1	Type of the outstanding transaction tracker that is used.
TBUCFG_ROT_DEPTH	4, 8, 16, 32, 64, 128, 256, 512	Number of outstanding read transactions.
TBUCFG_WOT_DEPTH	4, 8, 16, 32, 64, 128, 256, 512	Number of outstanding write transactions.
TBUCFG_DATARAM_TYPE	0, 1, 2	RAM type for data group of RAMs.

2.7.6 LTI TBU configuration parameters

You can configure the Local Translation Interface (LTI) Translation Buffer Unit (TBU).


Tip

For more detailed descriptions of these configuration parameters, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

The following table shows the LTI TBU configuration parameters.

Table 2-85: LTI TBU configuration parameters

Parameter name	Values	Description
TBUCFG_LTI_ID_WIDTH	1-32	LTI ID width.
TBUCFG_LTI_LOOP_WIDTH	1-256	LTI loop width.
TBUCFG_LAFIFO_EXTENSION	0, 1, 2, 4	Increases the value of the LA_FIFO_DEPTH parameter.

2.7.7 Common TBU debug configuration parameters

You can configure the *Translation Buffer Unit* (TBU) debug parameters.


Tip

For more detailed descriptions of these configuration parameters, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

The following table shows the TBU debug configuration parameters.

Table 2-86: Common TBU debug configuration parameters

Parameter name	Values	Description
TBUCFG_USE_ELA_DEBUG	0, 1	Determines whether ELA debug is used or not

2.7.8 TBU Dual configuration parameters

Because only the R-TBU and the W-TBU are configurable, all the configuration parameters of the MMU S3 Dual TBU are for the R-TBU and W-TBU only.

The following table shows the configuration parameters for the R-TBU and the W-TBU.

Table 2-87: R-TBU and the W-TBU configuration parameters

Dual TBU parameter	TBUs that use parameter		R-TBU/W-TBU parameter
	R-TBU	W-TBU	
TBUCFG_SID_WIDTH	Yes	Yes	TBUCFG_SID_WIDTH
TBUCFG_SSID_WIDTH	Yes	Yes	TBUCFG_SSID_WIDTH
TBUCFG_ID_WIDTH	Yes	Yes	TBUCFG_ID_WIDTH
TBUCFG_LOOP_WIDTH	Yes	Yes	TBUCFG_LOOP_WIDTH
TBUCFG_MECID_WIDTH	Yes	Yes	TBUCFG_MECID_WIDTH
TBUCFG_CACHERAM_TYPE	Yes	Yes	TBUCFG_CACHERAM_TYPE
TBUCFG_SLOTRAM_TYPE	Yes	Yes	TBUCFG_SLOTRAM_TYPE
TBUCFG_DATARAM_TYPE	Yes	Yes	TBUCFG_DATARAM_TYPE
TBUCFG_USE_ELA_DEBUG	Yes	Yes	TBUCFG_USE_ELA_DEBUG
TBUCFG_LTI_OG_WIDTH_R	Yes	No	TBUCFG_LTI_OG_WIDTH
TBUCFG_LTI_OG_WIDTH_W	No	Yes	TBUCFG_LTI_OG_WIDTH

Dual TBU parameter	TBUs that use parameter		R-TBU/W-TBU parameter
	R-TBU	W-TBU	
TBUCFG_XLATE_SLOTS_R	Yes	No	TBUCFG_XLATE_SLOTS
TBUCFG_XLATE_SLOTS_W	No	Yes	TBUCFG_XLATE_SLOTS
TBUCFG_DIRECT_IDX	Yes	Yes	TBUCFG_DIRECT_IDX
TBUCFG_MTLB_PARTS	Yes	Yes	TBUCFG_MTLB_PARTS
TBUCFG_UTLB_DEPTH_R	Yes	No	TBUCFG_UTLB_DEPTH
TBUCFG_UTLB_DEPTH_W	No	Yes	TBUCFG_UTLB_DEPTH
TBUCFG_MTLB_DEPTH	Yes	Yes	TBUCFG_MTLB_DEPTH
TBUCFG_MTLB_WAYS	Yes	Yes	TBUCFG_MTLB_WAYS
TBUCFG_MTLB_BANKS	Yes	Yes	TBUCFG_MTLB_BANKS
TBUCFG_HZRD_ENTRIES_R	Yes	No	TBUCFG_HZRD_ENTRIES
TBUCFG_HZRD_ENTRIES_W	No	Yes	TBUCFG_HZRD_ENTRIES
TBUCFG_PARTID_WIDTH	Yes	Yes	TBUCFG_PARTID_WIDTH
TBUCFG_MTLB_LKP_SLOTS_R	Yes	No	TBUCFG_MTLB_LKP_SLOTS
TBUCFG_MTLB_LKP_SLOTS_W	No	Yes	TBUCFG_MTLB_LKP_SLOTS
TBUCFG_MTLB_UPD_SLOTS_R	Yes	No	TBUCFG_MTLB_UPD_SLOTS
TBUCFG_MTLB_UPD_SLOTS_W	No	Yes	TBUCFG_MTLB_UPD_SLOTS
TBUCFG_PMU_COUNTERS_R	Yes	No	TBUCFG_PMU_COUNTERS
TBUCFG_PMU_COUNTERS_W	No	Yes	TBUCFG_PMU_COUNTERS
TBUCFG_LA_HNDSHK_MODE	Yes	Yes	TBUCFG_LA_HNDSHK_MODE
TBUCFG_LR_HNDSHK_MODE	Yes	Yes	TBUCFG_LR_HNDSHK_MODE
TBUCFG_DATA_WIDTH	Yes	Yes	TBUCFG_DATA_WIDTH
TBUCFG_AWUSER_WIDTH	Yes	Yes	TBUCFG_AWUSER_WIDTH
TBUCFG_WUSER_WIDTH	Yes	Yes	TBUCFG_WUSER_WIDTH
TBUCFG_BUSER_WIDTH	Yes	Yes	TBUCFG_BUSER_WIDTH
TBUCFG_ARUSER_WIDTH	Yes	Yes	TBUCFG_ARUSER_WIDTH
TBUCFG_RUSER_WIDTH	Yes	Yes	TBUCFG_RUSER_WIDTH
TBUCFG_STASH_SUPPORT	No	Yes	TBUCFG_STASH_SUPPORT
TBUCFG_WBUF_DEPTH	No	Yes	TBUCFG_WBUF_DEPTH
TBUCFG_LFIFO_DEPTH	No	Yes	TBUCFG_LFIFO_DEPTH
TBUCFG_WOT_DEPTH	No	Yes	TBUCFG_WOT_DEPTH
TBUCFG_ROT_DEPTH_R	Yes	No	TBUCFG_ROT_DEPTH
TBUCFG_ROT_DEPTH_W	No	Yes	TBUCFG_ROT_DEPTH
TBUCFG_OT_TRACKER_TYPE	Yes	Yes	TBUCFG_OT_TRACKER_TYPE
TBUCFG_SI_AW_HNDSHK_MODE	No	Yes	TBUCFG_SI_AW_HNDSHK_MODE
TBUCFG_SI_W_HNDSHK_MODE	No	Yes	TBUCFG_SI_W_HNDSHK_MODE
TBUCFG_SI_B_HNDSHK_MODE	No	Yes	TBUCFG_SI_B_HNDSHK_MODE
TBUCFG_SI_AR_HNDSHK_MODE	Yes	No	TBUCFG_SI_AR_HNDSHK_MODE
TBUCFG_SI_R_HNDSHK_MODE_R	Yes	No	TBUCFG_SI_R_HNDSHK_MODE
TBUCFG_SI_R_HNDSHK_MODE_W	No	Yes	TBUCFG_SI_R_HNDSHK_MODE

Dual TBU parameter	TBUs that use parameter		R-TBU/W-TBU parameter
	R-TBU	W-TBU	
TBUCFG_MI_AW_HNDSHK_MODE	No	Yes	TBUCFG_MI_AW_HNDSHK_MODE
TBUCFG_MI_W_HNDSHK_MODE	No	Yes	TBUCFG_MI_W_HNDSHK_MODE
TBUCFG_MI_B_HNDSHK_MODE	No	Yes	TBUCFG_MI_B_HNDSHK_MODE
TBUCFG_MI_AR_HNDSK_MODE	Yes	No	TBUCFG_MI_AR_HNDSHK_MODE
TBUCFG_MI_R_HNDSHK_MODE_R	Yes	No	TBUCFG_MI_R_HNDSHK_MODE
TBUCFG_MI_R_HNDSHK_MODE_W	No	Yes	TBUCFG_MI_R_HNDSHK_MODE
TBUCFG_LEGACY_TZ_EN	Yes	Yes	TBUCFG_LEGACY_TZ_EN
TBUCFG_PARITY_ON	Yes	Yes	TBUCFG_PARITY_ON

2.8 Debug capability

The Neoverse™ MMU S3 System Memory Management Unit provides debug functionality using the CoreSight™ ELA-600 Embedded Logic Analyzer.

For more information about debug capability, see the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.


Note

The CoreSight™ ELA-600 Embedded Logic Analyzer is a separate licensed product that is not included with the Neoverse™ MMU S3 System Memory Management Unit.

Configuration options

For TCU configuration options, see 2.7.2 TCU debug configuration parameters on page 122.

For TBU configuration options, see 2.7.7 Common TBU debug configuration parameters on page 126.

Signals

For TCU observation signals, see B.1 TCU observation interfaces on page 346.

For ACE-Lite TBU observation signals, see B.2 ACE-Lite TBU observation interfaces on page 350.

For LTI TBU observation signals, see B.3 LTI TBU observation interfaces on page 353.

3. Programmers model for MMU S3

The Programmers model provides general information about the MMU S3 register properties, such as access types, unused, and reserved registers.

The following information applies to the MMU S3 registers:

- The base address is not fixed, and can be different for any particular system implementation. The offset of each register from the base address is fixed.
- Access type is described as follows:

RW	Read and write
RO	Read-only
WO	Write-only
RAZ	Read-As-Zero
WI	Writes ignored
SBZ	Should-Be-Zero

- Do not attempt to access reserved or unused address locations. Reading these locations results in RAZ and writing to these locations results in WI.
- Unless otherwise stated in the accompanying text:
  - Do not modify **UNDEFINED** register bits
  - Ignore **UNDEFINED** register bits on reads
  - All register bits have a reset value of 0
- Bit positions that are described as reserved are:
  - In an RW register, RAZ/WI
  - In an RO register, RAZ
  - In a WO register, WI

MMU S3 registers are accessed using the PROG APB5 completer interface on the TCU, and cannot be accessed directly through any other completer interfaces.

Some registers are 64-bit registers, but the PROG APB4 interface is a 32-bit register. Therefore, software accesses 64-bit registers 32 bits at a time, such accesses are not guaranteed to be 64-bit atomic. This behavior does not cause problems for software, because the SMMUv3 architecture does not require 64-bit atomic access to any registers.

The Programmers model contains separate TBU and TCU regions for internal control, RAS, and identification registers. Writes to unmapped or reserved registers are ignored, and reads *Should-Be-Zero* (SBZ). Non-secure accesses to Secure registers are RAZ/WI. MMU S3 implements the identification register scheme that the SMMUv3 architecture defines.

MMU S3 implements all the Performance Monitor Counter Group (PMCG) registers that the SMMUv3 architecture defines, except for:

- SMMU_PMCG_IIDR

- SMMU_PMCG_IRQ_CFG0
- SMMU_PMCG_IRQ_CFG1
- SMMU_PMCG_IRQ_CFG2
- SMMU_PMCG_IRQ_STATUS
- SMMU_PMCG_GMPAM
- SMMU_PMCG_MPAMIDR
- SMMU_PMCG_S_MPAMIDR

MMU S3 does not implement the following SMMUv3 architectural registers, and accesses to these locations are RAZ/WI:

- SMMU_IDR4
- SMMU_S_IDR4
- SMMU_IDR6
- SMMU_S_IDR6
- SMMU_R_IDR6
- SMMU_STATUSR
- SMMU_CMDQ_CONTROL_*
- SMMU_S_CMDQ_CONTROL_*
- SMMU_R_CMDQ_CONTROL_*
- SMMU_ECMDQ_*
- SMMU_DPT_*
- SMMU_GATOS_*
- SMMU_S_GATOS_*
- SMMU_VATOS_*
- SMMU_S_VATOS_*

For more information about the SMMU architectural registers, see the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

3.1 Clearing ERRSTATUS registers

Software can clear the TCU_ERRSTATUS and TBU_ERRSTATUS registers by writing ones to fields that are set.

For more information about these registers, see the following:

- 3.8.3 TCU_ERRSTATUS register on page 177
- 3.17.3 TBU_ERRSTATUS register on page 260

If both of the following are true, a write to the register is ignored:

- Any of the V, UE, OF, CE, DE, fields are nonzero before the write.
- The write does not clear the nonzero V, UE, OF, CE, DE fields to zero by writing ones to the applicable field or fields.


Note

CE must be cleared by writing `2'b11` to the field.

If a valid clearing write reaches the ERRSTATUS register on the same cycle as a new error, the new record is applied as though no previous error existed.

3.1.1 Realm Management Extension Register Principles

MMU S3 uses principles of Realm Management Extension (RME). Register ownership dictates which PAS is permitted to access the register. Register accesses with the incorrect PAS result in a RAZ/WI response.

The following table shows the permissible PAS for each register ownership.

Table 3-1: Permissible PAS for each register ownership

Register Ownership	Non-secure access	Secure access	Realm access	Root access
Non-secure	Permit	Permit	Permit	Permit
Secure	RAZ/WI	Permit	RAZ/WI	Permit
Realm	RAZ/WI	RAZ/WI	Permit	Permit
Root	RAZ/WI	RAZ/WI	RAZ/WI	Permit

3.2 SMMU architectural registers

MMU S3 implements many of the SMMU architectural registers, that the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* defines.

The following table shows the SMMUv3 architectural registers that MMU S3 implements.


Note

All writable register fields reset to 0 unless the SMMU architecture specifies otherwise.

Table 3-2: SMMUv3 architectural registers

Register	Name	Description
SMMU_S_IDR0 - SMMU_S_IDR3	SMMU Secure feature Identification Registers	Provides information about the Secure features that the SMMU implementation supports
SMMU_S_CR0	Secure global Control Register 0	Provides global configuration of the Secure SMMU
SMMU_S_CR0ACK	Secure global Control Register 0 update Acknowledge	Provides acknowledgment of completion of updates to SMMU_S_CR0
SMMU_S_CR1 SMMU_S_CR2	Secure global Control Registers	Provides the controls for Secure table and queue access attributes
SMMU_S_INIT	Secure Initialization control register	Provides a control to invalidate all Secure SMMU caching on system initialization
SMMU_S_GBPA	Secure Global Bypass Attribute register	Controls the global bypass attributes that are used for transactions from Secure streams when the MMU is disabled
SMMU_S_AGBPA	Secure Alternate Global Bypass Attribute	Allows an implementation to apply additional non-architected attributes or tags to bypassing transactions.
SMMU_S_IRQ_CTRL	Secure Interrupt Control register	Contains enables for SMMU interrupts
SMMU_S_IRQ_CTRLACK	Secure Interrupt Control register update Acknowledge	Provides acknowledgment of the completion of updates to SMMU_S_IRQ_CTRL
SMMU_S_GERROR	Secure Global Error status register	Provides information on Secure global programming interface errors
SMMU_S_GERRORN	Secure Global Error Acknowledgment register	Contains the acknowledgment fields for SMMU_S_GERROR errors
SMMU_S_GERROR_IRQ_CFG0 - SMMU_S_GERROR_IRQ_CFG2	Secure Global Error IRQ Configuration register	Contains the Secure MSI address configuration for the GERROR IRQ
SMMU_S_STRTAB_BASE	Secure Stream Table Base address register	Contains the base address and attributes for the Secure Stream table
SMMU_S_STRTAB_BASE_CFG	Secure Stream Table Base Configuration register	Contains configuration fields for the Secure Stream table
SMMU_S_CMDQ_BASE	Secure Command queue Base address register	Contains the base address and attributes for the Secure Command queue
SMMU_S_CMDQ_PROD	Secure Command queue Producer index register	Contains the Secure Command queue index for writes by the producer
SMMU_S_CMDQ_CONS	Secure Command queue Consumer index register	Contains the Secure Command queue index for reads by the consumer
SMMU_S_EVENTQ_BASE	Secure Event queue Base address register	Contains the base address and attributes for the Secure Event queue
SMMU_S_EVENTQ_PROD	Secure Event queue Producer index register	Contains the Secure Event queue index for writes by the producer
SMMU_S_EVENTQ_CONS	Secure Event queue Consumer index register	Contains the Secure Event queue index for reads by the consumer
SMMU_S_EVENTQ_IRQ_CFG0 - SMMU_S_EVENTQ_IRQ_CFG2	Secure Event queue IRQ Configuration registers	Contains the MSI address configuration for the Secure Event queue IRQ
SMMU_IDR0 - SMMU_IDR3 SMMU_IDR5	SMMU feature Identification Registers	Provides information about the features that the SMMU implementation supports

Register	Name	Description
SMMU_IIDR	Implementation Identification Register	Provides implementer, part, and revision information for the SMMU implementation
SMMU_AIDR	Architecture Identification Register	Identifies the SMMU architecture version to which the implementation conforms
SMMU_CR0	Non-secure global Control Register 0	Provides the controls for the global configuration of the Non-secure SMMU
SMMU_CR0ACK	Non-secure global Control Register 0 update Acknowledge register	Provides acknowledgment of completion of updates to SMMU_CR0
SMMU_CR1	Non-secure global Control Register 1	Provides the controls for Non-secure table and queue access attributes
SMMU_CR2	Non-secure global Control Register 2	Provides the controls for the configuration of the global Non-secure features
SMMU_GBPA	Non-secure Global Bypass Attribute register	Controls the global bypass attributes that are used for transactions from Non-secure streams when the MMU is disabled
SMMU_IRQ_CTRL	Non-secure Interrupt Control register	Provides IRQ enable flags for edge-triggered wired outputs, if implemented, and MSI writes, if implemented
SMMU_IRQ_CTRLACK	Non-secure Interrupt Control register update Acknowledge register	Provides acknowledgment of the completion of updates to SMMU_IRQ_CTRL
SMMU_GERROR	Non-secure Global Error status register	Provides information about Non-secure global programming interface errors
SMMU_GERRORN	Non-secure Global Error acknowledgment register	Contains the acknowledgment fields for SMMU_GERROR errors
SMMU_GERROR_IRQ_CFG0	Non-secure Global Error IRQ Configuration register 0	Contains the MSI address configuration for the GERROR IRQ
SMMU_GERROR_IRQ_CFG1	Non-secure Global Error IRQ Configuration register 1	Contains the MSI payload configuration for the GERROR IRQ
SMMU_GERROR_IRQ_CFG2	Non-secure Global Error IRQ Configuration register 2	Contains the MSI attribute configuration for the GERROR IRQ
SMMU_STRTAB_BASE	Non-secure Stream Table Base address register	Contains the base address and attributes for the Non-secure Stream table
SMMU_STRTAB_BASE_CFG	Non-secure Stream Table Configuration register	Contains configuration fields for the Non-secure Stream table
SMMU_CMDQ_BASE	Non-secure Command queue Base address register	Contains the base address and attributes for the Non-secure Command queue
SMMU_CMDQ_PROD	Non-secure Command queue Producer index register	Contains the Non-secure Command queue index for writes by the producer
SMMU_CMDQ_CONS	Non-secure Command queue Consumer index register	Contains the Non-secure Command queue index for reads by the consumer
SMMU_EVENTQ_BASE	Non-secure Event queue Base address register	Contains the base address and attributes for the Non-secure Event queue
SMMU_EVENTQ_PROD	Non-secure Event queue Producer index register	Contains the Non-secure Event queue index for writes by the producer
SMMU_EVENTQ_CONS	Non-secure Event queue Consumer index register	Contains the Non-secure Event queue index for reads by the consumer

Register	Name	Description
SMMU_EVENTQ_IRQ_CFG0	Non-secure Event queue IRQ Configuration register 0	Contains the MSI address configuration for the Event queue IRQ
SMMU_EVENTQ_IRQ_CFG1	Non-secure Event queue IRQ Configuration register 1	Contains the MSI payload configuration for the Event queue IRQ
SMMU_EVENTQ_IRQ_CFG2	Non-secure Event queue IRQ Configuration register 2	Contains the MSI attribute configuration for the Event queue IRQ
SMMU_PRIQ_BASE	Non-secure PRI queue Base address register	Contains the base address and attributes for the Non-secure PRI queue
SMMU_PRIQ_PROD	Non-secure PRI queue Producer index register	Contains the Non-secure PRI queue index for writes by the producer
SMMU_PRIQ_CONS	Non-secure PRI queue Consumer index register	Contains the Non-secure PRI queue index for reads by the consumer
SMMU_PRIQ_IRQ_CFG0	Non-secure PRI queue IRQ Configuration register 0	Contains the MSI address configuration for the PRI queue IRQ
SMMU_PRIQ_IRQ_CFG1	Non-secure PRI queue IRQ Configuration register 1	Contains the MSI payload configuration for the PRI queue IRQ
SMMU_PRIQ_IRQ_CFG2	Non-secure PRI queue IRQ Configuration register 2	Contains the MSI attribute configuration for the PRI queue IRQ

MMU S3 implements an SMMUv3 Performance Monitor Counter Group (PMCG) in the TCU and in each TBU. The following table lists the registers that MMU S3 implements in each PMCG.

Table 3-3: SMMUv3 PMCG registers

Register	Name	Description
SMMU_PMCG_EVCNTR0 - SMMU_PMCG_EVCNTR3	SMMU PMCG Event Counter registers	Contains the values of the event counters
SMMU_PMCG_EVTYPER0 - SMMU_PMCG_EVTYPER3	SMMU PMCG Event Type configuration registers	Configures the events that the corresponding counter counts
SMMU_PMCG_SVR0 - SMMU_PMCG_SVR3	SMMU PMCG Shadow Value Registers	Contains the shadow value of the corresponding event counter
SMMU_PMCG_SMR0	SMMU PMCG Stream Match filter Register	Configures the stream match filter for the corresponding event counter
SMMU_PMCG_CNTENSET0	SMMU PMCG Counter Enable Set register	Provides the set mechanism for the counter enables
SMMU_PMCG_CNTENCLR0	SMMU PMCG Counter Enable Clear register	Provides the clear mechanism for the counter enables
SMMU_PMCG_INTENSET0	SMMU PMCG Interrupt contribution Enable Set register	Provides the set mechanism for the counter interrupt contribution enables
SMMU_PMCG_INTENCLR0	SMMU PMCG Interrupt contribution Enable Clear register	Provides the clear mechanism for the counter interrupt enables
SMMU_PMCG_OVSCLR0	SMMU PMCG Overflow Status Clear register	Provides the clear mechanism for the overflow status bits and provides read access to the overflow status bit values
SMMU_PMCG_OVSSET0	SMMU PMCG Overflow Status Set register	Provides the set mechanism for the overflow status bits and provides read access to the overflow status bit values
SMMU_PMCG_CAPR	SMMU PMCG Counter shadow value Capture Register	Controls the counter shadow value capture mechanism

Register	Name	Description
SMMU_PMCG_SCR	SMMU PMCG Secure Control Register	Secure Control Register
SMMU_PMCG_SCRA	SMMU PMCG Secure Control Register	Secure Control Register, alias of SMMU_PMCG_SCR register
SMMU_PMCG_CFGR	SMMU PMCG Configuration information Register	Provides information about the PMCG implementation
SMMU_PMCG_CR	SMMU PMCG Control Register	Contains the Performance Monitor control flags
SMMU_PMCG_ROOTCR	SMMU PMCG Root Control Register	Root Control Register
SMMU_PMCG_CEID0 - SMMU_PMCG_CEID1	SMMU PMCG Common Event ID registers	Contains the lower and upper 64 bits of the Common Event identification bitmap
SMMU_PMCG_IRQ_CTRL	SMMU PMCG IRQ enable register	Contains the Performance Monitors IRQ enable
SMMU_PMCG_IRQ_CTRLACK	SMMU PMCG IRQ enable Acknowledge register	Provides acknowledgment of the completion of updates to SMMU_PMCG_IRQ_CTRL
SMMU_PMCG_AIDR	SMMU PMCG Architecture Identification Register	Provides the Performance Monitor Architecture Identification
SMMU_PMCG_ID_REGS	ID registers	**IMPLEMENTATION DEFINED**
SMMU_PMCG_PMAUTHSTATUS	PMU Authentication Status register	Performance Monitor authentication status
SMMU_PMCG_PMDEVARCH	PMU Device Architecture register	Performance Monitor architecture identifier
SMMU_PMCG_PMDEVTYPE	PMU Device Type register	Performance Monitor device type

The following are the SMMU for RME registers:

- SMMU_ROOT_IDR0
- SMMU_ROOT_IIDR
- SMMU_ROOT_CR0
- SMMU_ROOT_CR0ACK
- SMMU_ROOT_GPT_BASE
- SMMU_ROOT_GPT_BASE2
- SMMU_ROOT_GPT_BASE_UPDATE
- SMMU_ROOT_GPT_BASE_CFG
- SMMU_ROOT_GPF_FAR
- SMMU_ROOT_GPT_CFG_FAR
- SMMU_ROOT_TLBI
- SMMU_ROOT_TLBI_CTRL

The following are the SMMU for RME-DA registers:

- SMMU_R_IDR0
- SMMU_R_IDR1

- SMMU_R_IDR2
- SMMU_R_IDR3
- SMMU_R_IDR4
- SMMU_R_AIDR
- SMMU_R_CR0
- SMMU_R_CR0ACK
- SMMU_R_CR1
- SMMU_R_CR2
- SMMU_R_GBPA
- SMMU_R_AGBPA
- SMMU_R_IRQ_CTRL
- SMMU_R_IRQ_CTRLACK
- SMMU_R_GERROR
- SMMU_R_GERRORN
- SMMU_R_GERROR_IRQ_CFG0_LO
- SMMU_R_GERROR_IRQ_CFG0_HI
- SMMU_R_GERROR_IRQ_CFG1
- SMMU_R_GERROR_IRQ_CFG2
- SMMU_R_STRTAB_BASE_LO
- SMMU_R_STRTAB_BASE_HI
- SMMU_R_STRTAB_BASE_CFG
- SMMU_R_CMDQ_BASE_LO
- SMMU_R_CMDQ_BASE_HI
- SMMU_R_CMDQ_PROD
- SMMU_R_CMDQ_CONS
- SMMU_R_EVENTQ_BASE_LO
- SMMU_R_EVENTQ_BASE_HI
- SMMU_R_EVENTQ_IRQ_CFG0_LO
- SMMU_R_EVENTQ_IRQ_CFG0_HI
- SMMU_R_EVENTQ_IRQ_CFG1
- SMMU_R_EVENTQ_IRQ_CFG2
- SMMU_R_PRIQ_BASE_LO
- SMMU_R_PRIQ_BASE_HI
- SMMU_R_PRIQ_IRQ_CFG0_LO

- SMMU_R_PRIQ_IRQ_CFG0_HI
- SMMU_R_PRIQ_IRQ_CFG1
- SMMU_R_PRIQ_IRQ_CFG2
- SMMU_R_MPAMIDR
- SMMU_R_GMPAM
- SMMU_R_MECIDR
- SMMU_R_GMECID
- SMMU_R_EVENTQ_PROD
- SMMU_R_EVENTQ_CONS
- SMMU_R_PRIQ_PROD
- SMMU_R_PRIQ_CONS

The following are the SMMU for DPT registers:

- SMMU_R_DPT_BASE_CFG
- SMMU_R_DPT_BASE_LO
- SMMU_R_DPT_BASE_HI
- SMMU_R_DPT_CFG_FAR_LO
- SMMU_R_DPT_CFG_FAR_HI

The following are the SMMU for PMU registers:

- SMMU_PMCG_SCRA
- SMMU_PMCG_ROOTCR

3.3 SMMU memory map

The MMU S3 memory map contains all the registers and includes the Translation Control Unit (TCU) and all Translation Buffer Units (TBUs), and shows the maximum number of implemented TBUs.

This document describes all TBU and TCU register addresses relative to the base address for that component.


Note

The TBU number is assigned according to the TID value that the TBU provides in its initial Distributed Translation Interface (DTI) connection request. The registers for that TBU are accessed according to the address range for its TID, as the following table shows.

The following table shows the full memory map.

Table 3-4: Main MMU S3 memory map

Address range	Description
0x0000000-0x00AFFFC	TCU registers
0x0100000-0x017FFFC	TBU0 registers. Includes microarchitectural, RAS, ID, MPAM, and PMCG registers.
0x0180000-0x01FFFFC	TBU1 registers. Includes microarchitectural, RAS, ID, MPAM, and PMCG registers.
0x0200000-0x027FFFC	TBU2 registers. Includes microarchitectural, RAS, ID, MPAM, and PMCG registers.
...	...
0x1F00000-0x1F7FFFC	TBU60 registers. Includes microarchitectural, RAS, ID, MPAM, and PMCG registers.
0x1F80000-0x1FFFFFC	TBU61 registers. Includes microarchitectural, RAS, ID, MPAM, and PMCG registers.


Note

Where a multi-port LTI TBU is implemented, it occupies the same address map space as if it were a single-port LTI TBU. The addressing is per TBU, not per initiating interface.

3.3.1 TCU memory map

The TCU memory map contains various categories of registers.

The TCU **IMPLEMENTATION DEFINED** registers include the following:

- 3.6 TCU PMU registers on page 150
- 3.7 TCU microarchitectural registers on page 153 for controlling microarchitectural features
- 3.8 TCU RAS registers on page 175
- 3.9 TCU system discovery registers on page 187

The following registers are also included:

- 3.12 TCU PIU integration registers on page 233.
- Walk cache stage and level Memory System Resource Partitioning and Monitoring (MPAM) maximum capacity registers
- Granule Protection Table (GPT) cache level MPAM maximum capacity registers
- MPAM memory-mapped registers

The following registers are on separate 64K pages to enable mapping to different pieces of software:

- Configuration
- PMU
- RAS
- MPAM

The following table shows the MMU S3 TCU memory map.

Table 3-5: MMU S3 TCU memory map

Address range	Description
0x0000000- 0x000FFFC	TCU SMMUv3 registers page 0, ID registers.
0x0010000- 0x001FFFC	TCU SMMUv3 registers page 1.
0x0020000- 0x002FFFC	TCU **IMPLEMENTATION DEFINED** registers, integration registers, and TCU system discovery registers.
0x0030000- 0x003FFFC	TCU PMCG registers, page 0.
0x0040000- 0x004FFFC	TCU PMCG registers, page 1.
0x0050000- 0x005FFFC	TCU MPAM registers.
0x0060000- 0x006FFFC	TCU RAS registers.
0x0080000- 0x008FFFC	TCU Realm Register Page 0.
0x0090000- 0x009FFFC	TCU Realm Register Page 1.
0x00A0000- 0x00AFFFC	TCU Root Control Page.

The following table shows how the TCU **IMPLEMENTATION DEFINED** PMCG, and MPAM registers are allocated to regions of the TCU address space. Other regions are reserved.

Table 3-6: TCU PMCG, RAS, and MPAM register allocation to regions of TCU address space

Address range	Description
0x00FD0- 0x00FFC	SMMU ID registers
0x0020000- 0x002FFFC	• 3.4.4 TCU microarchitectural registers summary on page 142, 0x0028E00-0x0029910 • 3.4.5 TCU system discovery registers summary on page 143, 0x002A000-0x002A090 • TCU_NODE_CTRL microarchitecture register. This register provides per connected TBU configuration options. See 3.7.11 TCU_NODE_CTRLn register on page 168, 0x0029000-0x00297FC • Walk Cache, 0x0029800-0x002981C for the following: ◦ AGW GPT cache ◦ DGW GPT cache CMAX settings
0x0030000- 0x003FFFC	Performance Monitor, page 0, as the previous table describes.
0x0040000- 0x004FFFC	Performance Monitor, page 1, as the previous table describes.
0x0050000- 0x005FFFC	TCU MPAM registers: • Non-secure MPAM, 0x0050000-0x0050FFC • Secure MPAM, 0x0051000-0x0051FFC • Realm MPAM, 0x0052000-0x0052FFC • Root MPAM, 0x0053000-0x0053FFC
0x0060000- 0x006FFFC	3.4.3 TCU Reliability, Availability, and Service registers summary on page 142

3.3.2 TBU memory map

The TBU memory map contains various categories of registers.

The TBU registers contain the following:

- **IMPLEMENTATION DEFINED** 3.16 TBU microarchitectural registers on page 249 for controlling microarchitectural features
- 3.18 TBU system discovery registers on page 267
- 3.17 TBU RAS registers on page 257
- 3.15 TBU PMU registers on page 246
- Performance Monitor counter registers, on a separate 64KB page to enable it to be paged for direct access from a Guest OS

The following table shows the TBU memory map.

Table 3-7: TBU memory map

Address range	Description	TBU Page
0x00FD0-0x00FFC	ID registers	0
0x08E00-0x08F10	Microarchitectural features	0
0x09000-0x09058	TBU system discovery registers	0
0x10000-0x10040	RAS	1
0x20000-0x20FFC	Performance Monitor page 0	2
0x30000-0x30FFC	Performance Monitor page 1	3
0x40000-0x40FFC	TBU MPAM Non-secure registers	4
0x41000-0x41FFC	TBU MPAM Secure registers	4
0x42000-0x42FFC	TBU MPAM Realm registers	4
0x43000-0x43FFC	TBU MPAM Root registers	4


Note

Any regions that the table does not show are reserved.

3.4 SMMU registers summary

The register summary describes the MMU S3 registers and some key characteristics.

3.4.1 TCU identification registers summary

MMU S3 contains TCU identification registers.

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

Table 3-8: TCU identification registers summary

Offset	Name	Type	Description
0x00FFC	SMMU_CIDR3	RO	3.5 TCU component and peripheral ID registers on page 149
0x00FF8	SMMU_CIDR2	RO	
0x00FF4	SMMU_CIDR1	RO	
0x00FF0	SMMU_CIDR0	RO	
0x00FEC	SMMU_PIDR3	RO	
0x00FE8	SMMU_PIDR2	RO	
0x00FE4	SMMU_PIDR1	RO	
0x00FE0	SMMU_PIDR0	RO	
0x00FDC	SMMU_PIDR7	RO	

Offset	Name	Type	Description
0x00FD8	SMMU_PIDR6	RO	
0x00FD4	SMMU_PIDR5	RO	
0x00FD0	SMMU_PIDR4	RO	

3.4.2 TCU and TBU PMU identification registers summary

The TCU and the TBU use the same PMU identification registers.

The following table shows the TCU and TBU PMU identification registers in offset order from the base memory address.

Table 3-9: TCU and TBU PMU identification registers summary

Offset	Name	Type	Description
0x00FB8	SMMU_PMCG_PMAUTHSTATUS	RO	3.6 TCU PMU registers on page 150 3.15 TBU PMU registers on page 246
0x00FD0	SMMU_PMCG_PIDR4	RO	
0x00FD4	SMMU_PMCG_PIDR5	RO	
0x00FD8	SMMU_PMCG_PIDR6	RO	
0x00FDC	SMMU_PMCG_PIDR7	RO	
0x00FE0	SMMU_PMCG_PIDR0	RO	
0x00FE4	SMMU_PMCG_PIDR1	RO	
0x00FE8	SMMU_PMCG_PIDR2	RO	
0x00FEC	SMMU_PMCG_PIDR3	RO	
0x00FF0	SMMU_PMCG_CIDR0	RO	
0x00FF4	SMMU_PMCG_CIDR1	RO	
0x00FF8	SMMU_PMCG_CIDR2	RO	
0x00FFC	SMMU_PMCG_CIDR3	RO	

3.4.3 TCU Reliability, Availability, and Service registers summary

MMU S3 contains TCU Reliability, Availability, and Service (RAS) registers.

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

Table 3-10: TCU RAS registers summary

Offset	Name	Type	Width	Description
0x60000	TCU_ERRFR	RO, Root	64-bit	3.8.1 TCU_ERRFR register on page 175
0x60008	TCU_ERRCTLR	RW, Root	64-bit	3.8.2 TCU_ERRCTLR register on page 176
0x60010	TCU_ERRSTATUS	RW, Root	64-bit	3.8.3 TCU_ERRSTATUS register on page 177
0x60040	TCU_ERRGEN	RW, Root	64-bit	3.8.4 TCU_ERRGEN register on page 182

3.4.4 TCU microarchitectural registers summary

MMU S3 contains TCU microarchitectural registers.

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

Table 3-11: TCU microarchitectural registers summary

Offset	Name	Type	Width	Description
0x28E00	TCU_CTRL	RW	32-bit	3.7.1 TCU_CTRL register on page 156
0x28E04	TCU_QOS	RW	32-bit	3.7.2 TCU_QOS register on page 157
0x28E08	TCU_CFG	RO	32-bit	3.7.3 TCU_CFG register on page 159
0x28E10	TCU_STATUS	RO	32-bit	3.7.4 TCU_STATUS register on page 160
0x28E18	TCU_SCR	RW, Secure	32-bit	3.7.5 TCU_SCR register on page 161
0x28E20	TCU_ITEN	RW	32-bit	3.11.1 ITEN register for the TCU on page 231
0x28E24	ITOP_PIU	RW	32-bit	3.12.1 ITOP_PIU register for the TCU Programmer Interface Unit on page 234
0x28E28	ITIN_PIU	RW	32-bit	3.12.2 ITIN_PIU register for the TCU Programmer Interface Unit on page 242
0x28E2C	ITOP_TMU	RW	32-bit	3.13.1 ITOP_TMU register for the TCU Translation Management Unit on page 243
0x28E30	ITIN_TMU	RW	32-bit	3.13.2 ITIN_TMU register for the TCU Translation Management Unit on page 245
0x28E34	TCU_ITEN_ET	RW	32-bit	3.11.2 ITEN_ET register for the TCU on page 232
0x28E40	TCU_CTRL2	RW	32-bit	3.7.6 TCU_CTRL2 register on page 162
0x28F00	TCU_ROOT_CTRL	RW	32-bit	3.7.7 TCU_ROOT_CTRL register on page 163
0x28F04	TCU_ROOT_CTRL2	RW	32-bit	3.7.8 TCU_ROOT_CTRL2 register on page 165
0x28F08	TCU_R_CTRL	RW	32-bit	3.7.9 TCU_R_CTRL register on page 165
0x28F10	TCU_RCR	RW	32-bit	3.7.10 TCU_RCR register on page 166
0x29000-0x293FC	TCU_NODE_CTRL*n*	RW	32-bit	3.7.11 TCU_NODE_CTRLn register on page 168
0x29400-0x297FC	TCU_NODE_STATUS*n*	RO	32-bit	3.7.12 TCU_NODE_STATUSn register on page 171
0x29800-0x2981C	TCU_WC_SxLy_CMAX	RW	32-bit	3.7.13 TCU_WC_SxLy_CMAX registers on page 172
0x29900	TCU_AGW_GC_Lx_CMAX	RW	32-bit	3.7.14 TCU_AGW_GC_Lx_CMAX registers on page 173
0x29910	TCU_DGW_GC_Lx_CMAX	RW	32-bit	3.7.15 TCU_DGW_GC_Lx_CMAX registers on page 174
0x29918	TCU_DC_L0_CMAX	RW	32-bit	3.7.16 TCU_DC_L0_CMAX and TCU_DC_L1_CMAX registers on page 174
0x2991C	TCU_DC_L1_CMAX	RW	32-bit	3.7.16 TCU_DC_L0_CMAX and TCU_DC_L1_CMAX registers on page 174

3.4.5 TCU system discovery registers summary

MMU S3 contains TCU system discovery registers.

The following table shows the TCU system discovery registers in offset order from the base memory address.

Table 3-12: TCU system discovery registers summary

Offset	Name	Type	Width	Description
0x2A000	TCU_SYSDISC0	RO	32-bit	3.9.1 TCU_SYSDISC0 system discovery register on page 187
0x2A004	TCU_SYSDISC1	RO	32-bit	3.9.2 TCU_SYSDISC1 system discovery register on page 188
0x2A008	TCU_SYSDISC2	RO	32-bit	3.9.3 TCU_SYSDISC2 system discovery register on page 189
0x2A00C	TCU_SYSDISC3	RO	32-bit	3.9.4 TCU_SYSDISC3 system discovery register on page 190
0x2A010	TCU_SYSDISC4	RO	32-bit	3.9.5 TCU_SYSDISC4 system discovery register on page 191
0x2A014	TCU_SYSDISC5	RO	32-bit	3.9.6 TCU_SYSDISC5 system discovery register on page 192
0x2A018	TCU_SYSDISC6	RO	32-bit	3.9.7 TCU_SYSDISC6 system discovery register on page 193
0x2A01C	TCU_SYSDISC7	RO	32-bit	3.9.8 TCU_SYSDISC7 system discovery register on page 194
0x2A020	TCU_SYSDISC8	RO	32-bit	3.9.9 TCU_SYSDISC8 system discovery register on page 195
0x2A024	TCU_SYSDISC9	RO	32-bit	3.9.10 TCU_SYSDISC9 system discovery register on page 196
0x2A028	TCU_SYSDISC10	RO	32-bit	3.9.11 TCU_SYSDISC10 system discovery register on page 197
0x2A02C	TCU_SYSDISC11	RO	32-bit	3.9.12 TCU_SYSDISC11 system discovery register on page 198
0x2A030	TCU_SYSDISC12	RO	32-bit	3.9.13 TCU_SYSDISC12 system discovery register on page 200
0x2A034	TCU_SYSDISC13	RO	32-bit	3.9.14 TCU_SYSDISC13 system discovery register on page 200
0x2A038	TCU_SYSDISC14	RO	32-bit	3.9.15 TCU_SYSDISC14 system discovery register on page 201
0x2A03C	TCU_SYSDISC15	RO	32-bit	3.9.16 TCU_SYSDISC15 system discovery register on page 202
0x2A040	TCU_SYSDISC16	RO	32-bit	3.9.17 TCU_SYSDISC16 system discovery register on page 203
0x2A044	TCU_SYSDISC17	RO	32-bit	3.9.18 TCU_SYSDISC17 system discovery register on page 204
0x2A048	TCU_SYSDISC18	RO	32-bit	3.9.19 TCU_SYSDISC18 system discovery register on page 205
0x2A04C	TCU_SYSDISC19	RO	32-bit	3.9.20 TCU_SYSDISC19 system discovery register on page 206
0x2A050	TCU_SYSDISC20	RO	32-bit	3.9.21 TCU_SYSDISC20 system discovery register on page 207
0x2A054	TCU_SYSDISC21	RO	32-bit	3.9.22 TCU_SYSDISC21 system discovery register on page 208
0x2A058	TCU_SYSDISC22	RO	32-bit	3.9.23 TCU_SYSDISC22 system discovery register on page 209
0x2A05C	TCU_SYSDISC23	RO	32-bit	3.9.24 TCU_SYSDISC23 system discovery register on page 210
0x2A060	TCU_SYSDISC24	RO	32-bit	3.9.25 TCU_SYSDISC24 system discovery register on page 211
0x2A064	TCU_SYSDISC25	RO	32-bit	3.9.26 TCU_SYSDISC25 system discovery register on page 212
0x2A068	TCU_SYSDISC26	RO	32-bit	3.9.27 TCU_SYSDISC26 system discovery register on page 213
0x2A06C	TCU_SYSDISC27	RO	32-bit	3.9.28 TCU_SYSDISC27 system discovery register on page 214
0x2A070	TCU_SYSDISC28	RO	32-bit	3.9.29 TCU_SYSDISC28 system discovery register on page 215
0x2A074	TCU_SYSDISC29	RO	32-bit	3.9.30 TCU_SYSDISC29 system discovery register on page 216
0x2A078	TCU_SYSDISC30	RO	32-bit	3.9.31 TCU_SYSDISC30 system discovery register on page 217
0x2A07C	TCU_SYSDISC31	RO	32-bit	3.9.32 TCU_SYSDISC31 System Discovery Register on page 218
0x2A080	TCU_SYSDISC32	RO	32-bit	3.9.33 TCU_SYSDISC32 System Discovery Register on page 219

Offset	Name	Type	Width	Description
0x2A084	TCU_SYSDISC33	RO	32-bit	3.9.34 TCU_SYSDISC33 system discovery register on page 220
0x2A088	TCU_SYSDISC34	RO	32-bit	3.9.35 TCU_SYSDISC34 System Discovery Register on page 221
0x2A08C	TCU_SYSDISC35	RO	32-bit	3.9.36 TCU_SYSDISC35 system discovery register on page 222
0x2A090	TCU_SYSDISC36	RO	32-bit	3.9.37 TCU_SYSDISC36 system discovery register on page 223
0x2A094	TCU_SYSDISC37	RO	32-bit	3.9.38 TCU_SYSDISC37 system discovery register on page 224
0x2A098	TCU_SYSDISC38	RO	32-bit	3.9.39 TCU_SYSDISC38 system discovery register on page 225
0x2A09C	TCU_SYSDISC39	RO	32-bit	3.9.40 TCU_SYSDISC39 system discovery register on page 226
0x2A0A0	TCU_SYSDISC40	RO	32-bit	3.9.41 TCU_SYSDISC40 system discovery register on page 226
0x2A0A4	TCU_SYSDISC41	RO	32-bit	3.9.42 TCU_SYSDISC41 system discovery register on page 227
0x2A0A8	TCU_SYSDISC42	RO	32-bit	3.9.43 TCU_SYSDISC42 system discovery register on page 228
0x2A0AC	TCU_SYSDISC43	RO	32-bit	3.9.44 TCU_SYSDISC43 system discovery register on page 229

3.4.6 TCU AUX registers summary

MMU S3 contains TCU auxiliary registers.

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

Table 3-13: TCU AUX registers summary

Offset	Name	Type	Width	Description
0x2A100	TCU_CTRL_AUX0	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000000
0x2A104	TCU_CTRL_AUX1	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000001
0x2A108	TCU_CTRL_AUX2	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000002
0x2A10C	TCU_CTRL_AUX3	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000005
0x2A110	TCU_CTRL_AUX4	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000008
0x2A114	TCU_CTRL_AUX5	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000009
0x2A118	TCU_CTRL_AUX6	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000011
0x2A11C	TCU_CTRL_AUX7	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000015
0x2A120	TCU_CTRL_AUX8	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000025
0x2A124	TCU_CTRL_AUX9	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000030
0x2A128	TCU_CTRL_AUX10	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000039
0x2A12C	TCU_CTRL_AUX11	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000031
0x2A130	TCU_CTRL_AUX12	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000038
0x2A134	TCU_CTRL_AUX13	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000020
0x2A138	TCU_CTRL_AUX14	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000021
0x2A13C	TCU_CTRL_AUX15	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000004
0x2A140	TCU_CTRL_AUX16	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000014
0x2A144	TCU_CTRL_AUX17	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000028
0x2A148	TCU_CTRL_AUX18	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000007
0x2A200	TCU_CTRL_AUX19	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000000

Offset	Name	Type	Width	Description
0x2A204	TCU_CTRL_AUX20	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000001
0x2A208	TCU_CTRL_AUX21	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000002
0x2A20C	TCU_CTRL_AUX22	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000005
0x2A210	TCU_CTRL_AUX23	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000008
0x2A214	TCU_CTRL_AUX24	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000009
0x2A218	TCU_CTRL_AUX25	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000011
0x2A21C	TCU_CTRL_AUX26	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000015
0x2A220	TCU_CTRL_AUX27	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000025
0x2A224	TCU_CTRL_AUX28	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000030
0x2A228	TCU_CTRL_AUX29	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000039
0x2A22C	TCU_CTRL_AUX30	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000031
0x2A230	TCU_CTRL_AUX31	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000038
0x2A234	TCU_CTRL_AUX32	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000020
0x2A238	TCU_CTRL_AUX33	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000021
0x2A23C	TCU_CTRL_AUX34	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000004
0x2A240	TCU_CTRL_AUX35	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000014
0x2A244	TCU_CTRL_AUX36	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000028
0x2A300	TCU_CTRL_AUX37	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000000
0x2A304	TCU_CTRL_AUX38	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000001
0x2A308	TCU_CTRL_AUX39	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000002
0x2A30C	TCU_CTRL_AUX40	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000005
0x2A310	TCU_CTRL_AUX41	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000008
0x2A314	TCU_CTRL_AUX42	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000009
0x2A318	TCU_CTRL_AUX43	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000011
0x2A31C	TCU_CTRL_AUX44	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000015
0x2A320	TCU_CTRL_AUX45	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000025
0x2A324	TCU_CTRL_AUX46	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000030
0x2A328	TCU_CTRL_AUX47	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000039
0x2A32C	TCU_CTRL_AUX48	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000031
0x2A330	TCU_CTRL_AUX49	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000038
0x2A334	TCU_CTRL_AUX50	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000020
0x2A338	TCU_CTRL_AUX51	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000021
0x2A33C	TCU_CTRL_AUX52	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000004
0x2A340	TCU_CTRL_AUX53	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000014
0x2A344	TCU_CTRL_AUX54	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000028
0x2A348	TCU_CTRL_AUX55	RW	32-bit	3.10 TCU_CTRL_AUX<n> registers on page 230. Reset is 0x00000007

3.4.7 TBU identification registers summary

MMU S3 contains TBU identification registers.

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

Table 3-14: TBU identification registers summary

Offset	Name	Type	Description
0x00FFC	SMMU_CIDR3	RO	3.14 TBU component and peripheral ID registers on page 246
0x00FF8	SMMU_CIDR2	RO	
0x00FF4	SMMU_CIDR1	RO	
0x00FF0	SMMU_CIDR0	RO	
0x00FEC	SMMU_PIDR3	RO	
0x00FE8	SMMU_PIDR2	RO	
0x00FE4	SMMU_PIDR1	RO	
0x00FE0	SMMU_PIDR0	RO	
0x00FDC	SMMU_PIDR7	RO	
0x00FD8	SMMU_PIDR6	RO	
0x00FD4	SMMU_PIDR5	RO	
0x00FD0	SMMU_PIDR4	RO	

3.4.8 TBU Reliability, Availability, and Serviceability registers summary

MMU S3 contains TBU Reliability, Availability, and Serviceability (RAS) registers.

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

Table 3-15: TBU RAS registers summary

Offset	Name	Width	Type	Description
0x10000	TBU_ERRFR	64-bit	RO, Root	3.17.1 TBU_ERRFR register on page 258
0x10008	TBU_ERRCTLR	64-bit	RW, Root	3.17.2 TBU_ERRCTLR register on page 259
0x10010	TBU_ERRSTATUS	64-bit	RW, Root	3.17.3 TBU_ERRSTATUS register on page 260
0x10040	TBU_ERRGEN	64-bit	RW, Root	3.17.4 TBU_ERRGEN register on page 263

RAS error reporting

When a Correctable Error (CE) occurs:

A CE is reported in 3.17.3 TBU_ERRSTATUS register on page 260.

If TBU_ERRCTLR.FI is set, an interrupt is raised on ras_fhi. See 2.4.2.7 TBU interrupt interface on page 41.

3.4.9 TBU microarchitectural registers summary

MMU S3 contains TBU microarchitectural registers.

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

Table 3-16: TBU microarchitectural registers summary

Offset	Name	Type	Width	Description
0x08E00	TBU_CTRL	RW	32-bit	3.16.1 TBU_CTRL register on page 249
0x08E04	TBU_LTI_PORT_RESOURCE_LIMIT	RW	32-bit	3.16.2 TBU_LTI_PORT_RESOURCE_LIMIT register on page 250
0x08E18	TBU_SCR	RW, Secure	32-bit	3.16.3 TBU_SCR register on page 253
0x08F00	TBU_ROOT_CTRL	RW	32-bit	3.16.4 TBU_ROOT_CTRL register on page 255
0x08F10	TBU_RCR	RW	32-bit	3.16.5 TBU_RCR register on page 256

3.4.10 TBU system discovery registers summary

MMU S3 contains TBU system discovery registers.

The following table shows the TBU system discovery registers in offset order from the base memory address.

Table 3-17: TBU system discovery registers summary

Offset	Name	Type	Width	Description
0x09000	TBU_SYSDISC0	RO	32-bit	3.18.1 TBU_SYSDISC0 system discovery register on page 268
0x09004	TBU_SYSDISC1	RO	32-bit	3.18.2 TBU_SYSDISC1 system discovery register on page 269
0x09008	TBU_SYSDISC2	RO	32-bit	3.18.3 TBU_SYSDISC2 system discovery register on page 270
0x0900C	TBU_SYSDISC3	RO	32-bit	3.18.4 TBU_SYSDISC3 system discovery register on page 271
0x09010	TBU_SYSDISC4	RO	32-bit	3.18.5 TBU_SYSDISC4 system discovery register on page 272
0x09014	TBU_SYSDISC5	RO	32-bit	3.18.6 TBU_SYSDISC5 system discovery register on page 273
0x09018	TBU_SYSDISC6	RO	32-bit	3.18.7 TBU_SYSDISC6 system discovery register on page 274
0x0901C	TBU_SYSDISC7	RO	32-bit	3.18.8 TBU_SYSDISC7 system discovery register on page 275
0x09020	TBU_SYSDISC8	RO	32-bit	3.18.9 TBU_SYSDISC8 system discovery register on page 276
0x09024	TBU_SYSDISC9	RO	32-bit	3.18.10 TBU_SYSDISC9 system discovery register on page 277
0x09028	TBU_SYSDISC10	RO	32-bit	3.18.11 TBU_SYSDISC10 system discovery register on page 278
0x0902C	TBU_SYSDISC11	RO	32-bit	3.18.12 TBU_SYSDISC11 system discovery register on page 279
0x09030	TBU_SYSDISC12	RO	32-bit	3.18.13 TBU_SYSDISC12 system discovery register on page 280
0x09034	TBU_SYSDISC13	RO	32-bit	3.18.14 TBU_SYSDISC13 system discovery register on page 281
0x09038	TBU_SYSDISC14	RO	32-bit	3.18.15 TBU_SYSDISC14 system discovery register on page 282
0x0903C	TBU_SYSDISC15	RO	32-bit	3.18.16 TBU_SYSDISC15 system discovery register on page 283
0x09040	TBU_SYSDISC16	RO	32-bit	3.18.17 TBU_SYSDISC16 system discovery register on page 284
0x09044	TBU_SYSDISC17	RO	32-bit	3.18.18 TBU_SYSDISC17 system discovery register on page 285
0x09048	TBU_SYSDISC18	RO	32-bit	3.18.19 TBU_SYSDISC18 system discovery register on page 286

Offset	Name	Type	Width	Description
0x0904C	TBU_SYSDISC19	RO	32-bit	3.18.20 TBU_SYSDISC19 system discovery register on page 287
0x09050	TBU_SYSDISC20	RO	32-bit	3.18.21 TBU_SYSDISC20 system discovery register on page 288
0x09054	TBU_SYSDISC21	RO	32-bit	3.18.22 TBU_SYSDISC21 system discovery register on page 288
0x09058	TBU_SYSDISC22	RO	32-bit	3.18.23 TBU_SYSDISC22 system discovery register on page 289

3.4.11 TBU integration registers summary

MMU S3 contains TBU integration registers.

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

Table 3-18: TBU integration registers summary

Offset	Name	Type	Width	Description
0x08E20	TBU_ITEN	RW	32-bit	3.19.1 ITEN register for the TBU on page 291
0x08E24	TBU_ITOP	RW	32-bit	3.19.3 ITOP register for the TBU on page 293
0x08E28	TBU_ITIN	RW	32-bit	3.19.4 ITIN register for the TBU on page 297
0x08E2C	TBU_ITEN_ET	RW	32-bit	3.19.2 ITEN_ET register for the TBU on page 292

3.5 TCU component and peripheral ID registers

The MMU S3 TCU contains component and peripheral ID registers.

The following table shows the TCU component and peripheral ID registers.

Table 3-19: TCU component and peripheral ID registers

Offset	Name	Field	Value	Description
0x00FFC	SMMU_CIDR3, Component ID3	[7:0]	0xB1	Preamble
0x00FF8	SMMU_CIDR2, Component ID2	[7:0]	0x05	Preamble
0x00FF4	SMMU_CIDR1, Component ID1	[7:0]	0xF0	Preamble
0x00FF0	SMMU_CIDR0, Component ID0	[7:0]	0x0D	Preamble
0x00FEC	SMMU_PIDR3, Peripheral ID3	[7:4]	MAX(*p_level*, ecorevnum)	REVAND, minor revision, where *p_level* is 1 for p1.
	SMMU_PIDR3, Peripheral ID3	[3:0]	0x00	CMOD
0x00FE8	SMMU_PIDR2, Peripheral ID2	[7:4]	0x00	REVISION, major revision

Offset	Name	Field	Value	Description
	SMMU_PIDR2, Peripheral ID2	[3]	1	JEDEC-assigned value for DES always used
	SMMU_PIDR2, Peripheral ID2	[2:0]	3	DES_1: bits [6:4] bits of the JEP106 Designer code
0x00FE4	SMMU_PIDR1, Peripheral ID1	[7:4]	0xB	DES_0: bits [3:0] of the JEP106 Designer code
	SMMU_PIDR1, Peripheral ID1	[3:0]	0x4	PART_1: bits [11:8] of the Part number
0x00FE0	SMMU_PIDR0, Peripheral ID0	[7:0]	0x98	PART_0: bits [7:0] of the Part number
0x00FDC	SMMU_PIDR7, Peripheral ID7	-	RES0	Reserved
0x00FD8	SMMU_PIDR6, Peripheral ID6			
0x00FD4	SMMU_PIDR5, Peripheral ID5			
0x00FD0	SMMU_PIDR4, Peripheral ID4	[7:4]	0x0	SIZE = 4KB
		[3:0]	0x4	DES_2: JEP106 Designer continuation code

3.6 TCU PMU registers

The MMU S3 TCU contains Performance Monitor Unit (PMU) registers. The Performance Monitor counter register, on a separate 64KB page from the rest of the PMU registers, enables it to be paged for direct access from a Guest OS.

3.6.1 Registers

The TBU and TCU support the same PMCG registers.

These registers follow the register layout that the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* Performance Monitor Extension describes.

The following PMCG registers, that the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* defines, are implemented:

- SMMU_PMCG_EVCNTR{0-(TCUCFG_PMU_COUNTERS-1)}
- SMMU_PMCG_EVTYPER{0-(TCUCFG_PMU_COUNTERS-1)}
- SMMU_PMCG_SVR{0-(TCUCFG_PMU_COUNTERS-1)}
- SMMU_PMCG_SMR0
  - All counters share this mask register

    - The mask is 32 bits because the TCU uses 32-bit StreamIDs
- SMMU_PMCG_CNTENSET0
- SMMU_PMCG_CNTENCLR0
- SMMU_PMCG_INTENSET0
- SMMU_PMCG_INTSENCLR0
- SMMU_PMCG_OVSCLR0
- SMMU_PMCG_OVSSET0
- SMMU_PMCG_CAPR
- SMMU_PMCG_SCR
- SMMU_PMCG_CFGR. See 3.6.3 SMMU_PMCG_CFGR on page 152.
- SMMU_PMCG_CR
- SMMU_PMCG_CEID{0-1}. See 3.6.4 SMMU_PMCG_CEID{0-1} registers on page 152.
- SMMU_PMCG_IRQ_CTRL
- SMMU_PMCG_IRQ_CTRLACK
- SMMU_PMCG_AIDR, indicates SMMUv3.3
- SMMU_PMCG_ID_REGS
- SMMU_PMCG_SCRA
- SMMU_PMCG_ROOTCR

The following registers are not implemented, because the PMCG does not support MSIs:

- SMMU_PMCG_IRQ_CFG0
- SMMU_PMCG_IRQ_CFG1
- SMMU_PMCG_IRQ_CFG2
- SMMU_PMCG_IRQ_STATUS

The following registers are not implemented, because the PMCG implementation does not support MPAM:

- SMMU_PMCG_GMPAM
- SMMU_PMCG_MPAMIDR
- SMMU_PMCG_S_MPAMIDR

3.6.2 Events

In this description, a translation request corresponds to a translation slot allocation.

A single DTI translation request might correspond to multiple translation request events if:

- A translation results in a stall fault event and is restarted

- A translation results in a stall fault event when the Event queue is full, and is later retried when the Event queue becomes non-full

Each event indicates:

- Whether the SMMU_PMCG_SMR0 register can filter it
- For events that cannot be filtered, whether they are only visible when Secure events are visible by:
  - SMMU_PMCG_ROOTCR.NAO&(SMMU_PMCG_SCR.SO|SMMU_PMCG_SCR.NAO) in RME mode
  - SMMU_PMCG_SCR.SO in legacy mode

For more information about the architectural and **IMPLEMENTATION DEFINED** events that are implemented, see 2.5.2.1 SMMUv3 architectural performance events on page 57.

The following events are also counted for prefetch accesses:

0x80-0x90
Walk cache events.

0x92-0x94
Configuration cache events.

3.6.3 SMMU_PMCG_CFGR

An MMU S3 implementation has fixed values for SMMU_PMCG_CFGR, and these values define behavioral aspects of the implementation.

For information about the SMMU_PMCG_CFGR field values, see 2.5.2.4 SMMUv3 PMU register architectural options on page 67.

3.6.4 SMMU_PMCG_CEID{0-1} registers

The SMMU_PMCG_CEID{0-1} registers indicate the architectural events that are supported. They are described as 64-bit registers, but are accessed 32 bits at a time through the 32-bit PROG interface.

The following table shows the SMMU_PMCG_CEID{0-1} registers.

Table 3-20: SMMU_PMCG_CEID{0-1} registers

Offset	Name	Value
0x30E20	SMMU_PMCG_CEID0	0x0000007F
0x30E28	SMMU_PMCG_CEID1	0x00000000

3.6.5 PMU ID registers

The PMU ID registers appear only in Performance Monitor Page 0. Page 1 does not contain any ID registers.

The following table shows the PMU ID registers.

Table 3-21: PMU ID registers

Offset	Name	Field	Value	Description
0x30FB8	SMMU_PMCG_PMAUTHSTATUS	[7:0]	0x00	No authentication interface is implemented
0x30FD0	SMMU_PMCG_PIDR4, Peripheral ID4	[7:4]	0x4	SIZE = 4KB
		[3:0]	0x0	DES_2: JEP106 Designer continuation code
0x30FD4	SMMU_PMCG_PIDR5, Peripheral ID5	-	RES0	Reserved
0x30FD8	SMMU_PMCG_PIDR6, Peripheral ID6	-	RES0	Reserved
0x30FDC	SMMU_PMCG_PIDR7, Peripheral ID7	-	RES0	Reserved
0x30FE0	SMMU_PMCG_PIDR0, Peripheral ID0	[7:0]	0x99 for TLBU PMU 0x98 for TMU PMU	PART_0: bits [7:0] of the Part number
0x30FE4	SMMU_PMCG_PIDR1, Peripheral ID1	[7:4]	0xB	DES_0: bits [3:0] of the JEP106 Designer code
		[3:0]	0x4	PART_1: bits [11:8] of the Part number
0x30FE8	SMMU_PMCG_PIDR2, Peripheral ID2	[7:4]	0x01	REVISION, major revision, 1, for r1
		[3]	1	JEDEC-assigned value for DES always used
		[2:0]	3	DES_1: bits [6:4] bits of the JEP106 Designer code
0x30FEC	SMMU_PMCG_PIDR3, Peripheral ID3	[7:4]	MAX(*p_level*, ecorevnum)	REVAND, minor revision, where *p_level* is: **1** For p1
		[3:0]	0x00	CMOD
0x30FF0	SMMU_PMCG_CIDR0, Component ID0	[7:0]	0x0D	Preamble
0x30FF4	SMMU_PMCG_CIDR1, Component ID1	[7:0]	0x90	Preamble
0x30FF8	SMMU_PMCG_CIDR2, Component ID2	[7:0]	0x05	Preamble
0x30FFC	SMMU_PMCG_CIDR3, Component ID3	[7:0]	0xB1	Preamble

The PMDEVARCH and PMDEVTYPE registers are implemented as the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* defines.

3.7 TCU microarchitectural registers

You can set the TCU microarchitectural registers at boot time to optimize TCU behavior for your system. We recommend that you use the default values for most systems.

If the TCUCFG_LEGACY_TZ_EN parameter is set to 1, or the legacy_tz_en signal is set to 1, all bits in the 3.7.10 TCU_RCR register on page 166 behave as 1 and the 3.7.10 TCU_RCR register on page 166 is RAZ/WI.

The 3.7.10 TCU_RCR register on page 166 is root-only. Non-root access to this register is Read-As-Zero (RAZ)/Write-Ignored (WI), RAZ/WI.

The 3.7.5 TCU_SCR register on page 161 is Secure or root-only. Non-secure or realm access to this register is RAZ/WI.

TCU_RCR.S_UARCH controls non-root access to the registers that this section discusses, except for the following registers:

- 3.7.7 TCU_ROOT_CTRL register on page 163
- 3.7.8 TCU_ROOT_CTRL2 register on page 165
- 3.7.5 TCU_SCR register on page 161
- 3.7.10 TCU_RCR register on page 166

Note

If the `TCUCFG_LEGACY_TZ_EN` parameter is set to 1, or the legacy_tz_en signal is set to 1, the bit TCU_RCR_S.UARCH behaves as 1. Therefore, non-root register access is permitted for microarchitectural registers.

Non-root accesses to the above registers, when TCU_RCR.S_UARCH == 0 are RAZ/WI.

This mechanism permits the root to disable all non-root access to these registers if necessary for security reasons. However, by default, non-root access is granted as the rest of this section describes.

TCU_RCR.S_ROOT_UARCH controls non-root access to the following:

- 3.7.7 TCU_ROOT_CTRL register on page 163
- 3.7.8 TCU_ROOT_CTRL2 register on page 165

Non-root accesses to the TCU_ROOT_CTRL register when TCU_RCR.S_ROOT_UARCH == 0 is RAZ/WI.

TCU_SCR.NS_UARCH controls Non-secure and realm access to registers in this section other than the TCU_SCR and TCU_RCR registers.

Non-secure and realm accesses to these registers, when TCU_SCR.NS_UARCH == 0, are RAZ and WI.

TCU_RCR.S_UARCH and TCU_RCR.S_ROOT_UARCH overrule control. If TCU_RCR.S_UARCH and TCU_RCR.S_ROOT_UARCH are set to 0, they prevent non-root access, regardless of the setting in the TCU_SCR register, as this section describes.

The following table shows how SCR and RCR override the ownership of a register.

Table 3-22: SCR and RCR override of register ownership

SCR.feature	RCR.feature	Non-secure register	Secure register	Realm register	Root register
0	0	Non-secure	Secure	Realm	Root
0	1	Non-secure	Secure	Realm	Secure
1	0	Non-secure	Non-secure	Realm	Root
1	1	Non-secure	Non-secure	Realm	Non-secure

The following registers can be written only when the conditions in the list lower down occur:

- 3.7.1 TCU_CTRL register on page 156
- 3.7.6 TCU_CTRL2 register on page 162
- 3.7.7 TCU_ROOT_CTRL register on page 163
- 3.7.8 TCU_ROOT_CTRL2 register on page 165
- 3.7.2 TCU_QOS register on page 157
- 3.7.11 TCU_NODE_CTRLn register on page 168
- 3.7.13 TCU_WC_SxLy_CMAX registers on page 172
- 3.7.16 TCU_DC_L0_CMAX and TCU_DC_L1_CMAX registers on page 174
- 3.7.9 TCU_R_CTRL register on page 165

The registers in the previous list can be written only when the following conditions occur:

- SMMU_CR0.SMMUEN == 0
- SMMU_CR0ACK.SMMUEN == 0
- SMMU_S_CR0.SMMUEN == 0
- SMMU_S_CR0ACK.SMMUEN == 0
- SMMU_R_CR0.SMMUEN == 0
- SMMU_R_CR0ACK.SMMUEN == 0

In addition to the above conditions, the following registers can be written only when the conditions in the list lower down occur:

- 3.7.7 TCU_ROOT_CTRL register on page 163
- 3.7.8 TCU_ROOT_CTRL2 register on page 165
- 3.7.14 TCU_AGW_GC_Lx_CMAX registers on page 173
- 3.7.15 TCU_DGW_GC_Lx_CMAX registers on page 174

The registers in the previous list can be written only when the following conditions occur:

- SMMU_CR0.SMMUEN == 0
- SMMU_CR0ACK.SMMUEN == 0
- SMMU_S_CR0.SMMUEN == 0
- SMMU_S_CR0ACK.SMMUEN == 0

- SMMU_R_CR0.SMMUEN == 0
- SMMU_R_CR0ACK.SMMUEN == 0
- SMMU_ROOT_CR0.GPCEN == 0
- SMMU_ROOT_CR0ACK.GPCEN == 0

After modifying these registers, software must issue an INV_ALL operation using the SMMU_S_INIT register, before it sets SMMUEN to 1. Failure to issue the operation results in **UNPREDICTABLE** behavior.

We also recommend that for any register fields that relate to caching, you change the register only when there are no ongoing invalidations.

An array of registers, TCU_CTRL_AUX<n>, where <n> is 0-55, exists. Do not modify the values of these registers unless Arm advises otherwise. See:

- 3.10 TCU_CTRL_AUX<n> registers on page 230
- 3.7 TCU microarchitectural registers on page 153

3.7.1 TCU_CTRL register

The TCU Control register disables TCU features. If the hit rate of an individual walk cache is too low, you can disable individual walk caches to improve performance in some systems. Do not modify the AUX bits unless we direct you to do so.

Configurations

This register is available in all configurations.

Attributes

The TCU_CTRL register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E00`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-1: TCU_CTRL register bit assignments

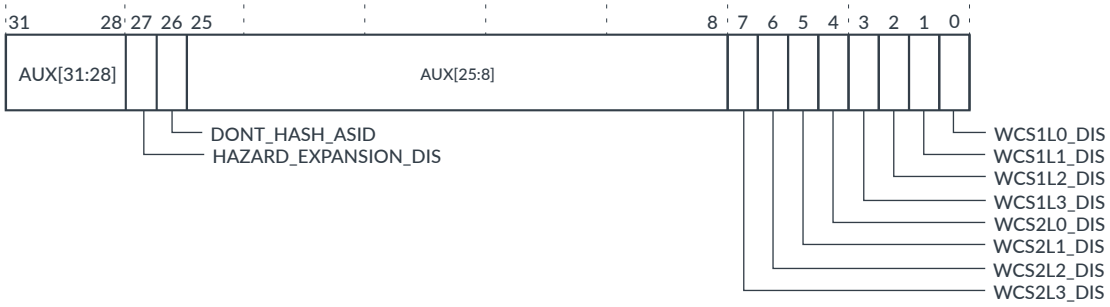


Table 3-23: TCU_CTRL register bit descriptions

Bits	Name	Description
[31:28]	AUX[31:28]	Reserved. Do not change these bits unless Arm advises you to. If writing other bits of this register, ensure that you write the current value for these bits to them, for example, by performing a read-modify-write sequence.
[27]	HAZARD_EXPANSION_DIS	Prevent the Hazard Unit (HZU) from increasing the size of the hazard to match the downstream data width. This effectively overrides the value of the hazard_width tieoff signal to be `3'h3`, 64 bits.
[26]	DONT_HASH_ASID	When set to 1, ASID is not used in the hash to create walk cache indices
[25:8]	AUX[25:8]	Reserved. Do not change these bits unless Arm advises you to. If writing other bits of this register, ensure that you write the current value for these bits to them, for example, by performing a read-modify-write sequence.
[7]	WCS2L3_DIS	Walk cache disable. When a bit of this field is set, it disables the corresponding stage and level of walk cache. WCS2L3_DIS is in bit [7], through to WCS1L0_DIS that is in bit [0].
[6]	WCS2L2_DIS	
[5]	WCS2L1_DIS	
[4]	WCS2L0_DIS	
[3]	WCS1L3_DIS	
[2]	WCS1L2_DIS	
[1]	WCS1L1_DIS	
[0]	WCS1L0_DIS	

3.7.2 TCU_QOS register

The TCU_QOS register controls the value of the AxQOS signal.


Note

The QoS value that is associated with each transaction does not take account of other transactions that are blocked behind it. For example, although translations with a high priority setting in the 3.7.11 TCU_NODE_CTRLn register on page 168 are normally progressed before translations with a lower priority, a low-priority page table walk can block a higher priority page table walk and prevent it from being issued from the TCU.

Configurations

This register is available in all configurations.

Attributes

The TCU_QOS register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E04`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-2: TCU_QOS register bit assignments

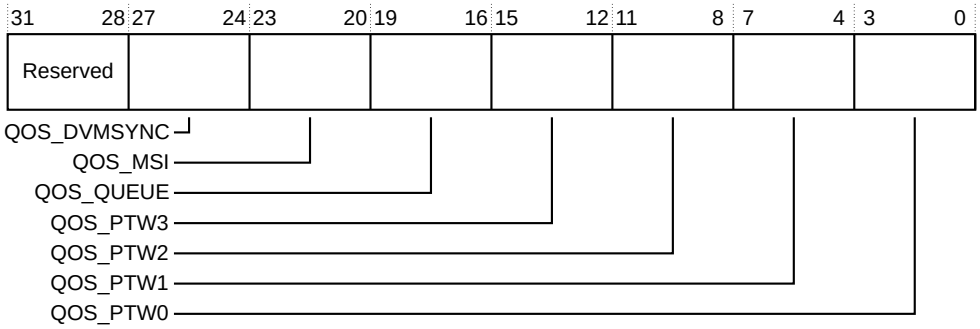


Table 3-24: TCU_QOS register bit descriptions

Bits	Name	Description
[31:28]	-	Reserved.
[27:24]	QOS_DVMSYNC	QoS level to use for DVM Sync Completion messages.
[23:20]	QOS_MSI	QoS level to use for MSIs.
[19:16]	QOS_QUEUE	QoS level to use for queue accesses.
[15:12]	QOS_PTW3	QoS level to use for the following, that are requested from nodes with TCU_NODE_CTRL*n*.PRIORITY = 3: • Translation table walks • Configuration Table Walks (CTWs) • Granule Protection Table (GPT) walks • Device permission Table (DPT) walks

Bits	Name	Description
[11:8]	QOS_PTW2	QoS level to use for the following, that are requested from nodes with TCU_NODE_CTRL*n*.PRIORITY = 2: • Translation table walks • CTWs • GPT walks • DPT walks
[7:4]	QOS_PTW1	QoS level to use for the following, that are requested from nodes with TCU_NODE_CTRL*n*.PRIORITY = 1: • Translation table walks • CTWs • GPT walks • DPT walks
[3:0]	QOS_PTW0	QoS level to use for the following, that are requested from nodes with TCU_NODE_CTRL*n*.PRIORITY = 0: • Translation table walks • CTWs • GPT walks • DPT walks

3.7.3 TCU_CFG register

Use the TCU_CFG register to set configuration information.

Configurations

This register is available in all configurations.

Attributes

The TCU_CFG register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E08`

Type

RO

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-3: TCU_CFG register bit assignments

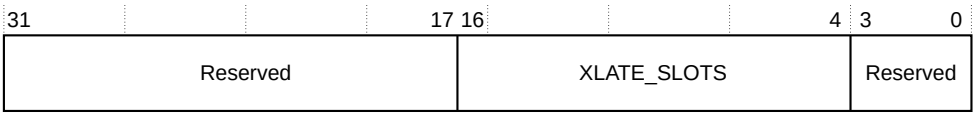


Table 3-25: TCU_CFG register bit descriptions

Bits	Name	Description
[31:17]	-	Reserved
[16:4]	XLATE_SLOTS	Number of translation slots that are available to be shared between all nodes. The value is the value of the `TCUCFG_XLATE_SLOTS`parameter. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
[3:0]	-	Reserved

3.7.4 TCU_STATUS register

Use the TCU_STATUS register to set status Information.

Configurations

This register is available in all configurations.

Attributes

The TCU_STATUS register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E10`

Type

RO

Reset value

0

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-4: TCU_STATUS register bit assignments

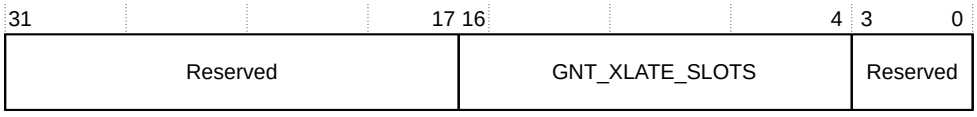


Table 3-26: TCU_STATUS register bit descriptions

Bits	Name	Description
[31:17]	-	Reserved
[16:4]	GNT_XLATE_SLOTS	Total number of translation slots that are currently allocated to all connected nodes. This information can be useful for debugging purposes.
[3:0]	-	Reserved

3.7.5 TCU_SCR register

The TCU Secure Control register controls whether Non-secure and Realm software is permitted to access each TCU register group.

This register does not control Secure access to the Performance Monitor registers. The SMMU_PMCG_SCR register controls Secure access to these registers as the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* defines.

Configurations

This register is available in all configurations.

Attributes

The TCU_SCR register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E18`

Type

Secure, RW

Reset value

sec_override. See A.1.12 TCU tie-off signals on page 312.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-5: TCU_SCR register bit assignments

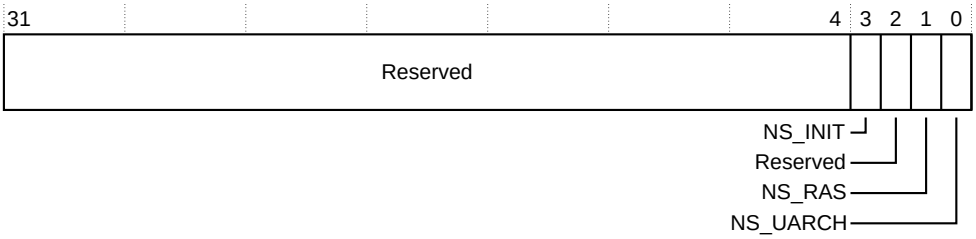


Table 3-27: TCU_SCR register bit descriptions

Bits	Name	Description
[31:4]	–	Reserved
[3]	NS_INIT	Non-secure and Realm register access is permitted to the SMMU_S_INIT register The sec_override input sets the reset value of this signal. See A.1.12 TCU tie-off signals on page 312.
[2]	-	Reserved
[1]	NS_RAS	Non-secure and Realm register access is permitted for RAS registers. When this bit is 0, Non-secure and Realm writes to the following register addresses are ignored, and Non-secure and Realm reads return zero: `0x006_0000-0x006_FFFC`. The sec_override input sets the reset value of this signal. See A.1.12 TCU tie-off signals on page 312.
[0]	NS_UARCH	Non-secure and Realm register access is permitted for microarchitectural registers When this bit is 0, Non-secure and Realm writes to the following register addresses are ignored, and Non-secure and Realm reads return zero: `0x28E00-0x29910` The sec_override input sets the reset value of this signal. See A.1.12 TCU tie-off signals on page 312. If Secure translation might be used, we recommend that software does not set this bit.

3.7.6 TCU_CTRL2 register

The TCU Control 2 register is Reserved. Do not change these bits unless Arm advises you to.

Configurations

This register is available in all configurations.

Attributes

The TCU_CTRL2 register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

0x28E40

Type

RW

Reset value

0x0000_0001

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-6: TCU_CTRL2 register bit assignments

Table 3-28: TCU_CTRL2 register bit descriptions

Bits	Name	Description
[31:0]	AUX[31:0]	Reserved. Do not change these bits unless Arm advises you to.

3.7.7 TCU_ROOT_CTRL register

The TCU_ROOT_CTRL register disables TCU features. In most systems, you can leave the TCU_ROOT_CTRL register unchanged.

The TCU_ROOT_CTRL register is root-only by default. It can be made accessible by Secure or Non-secure by setting various combinations of the 3.7.5 TCU_SCR register on page 161 and the 3.7.10 TCU_RCR register on page 166.

Configurations

This register is available only when either the TBUCFG_LEGACY_TZ_EN parameter is set to 0 or the legacy_tz_en signal is set to 0.

Attributes

The TCU_ROOT_CTRL register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

0x28F00

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-7: TCU_ROOT_CTRL register bit assignments

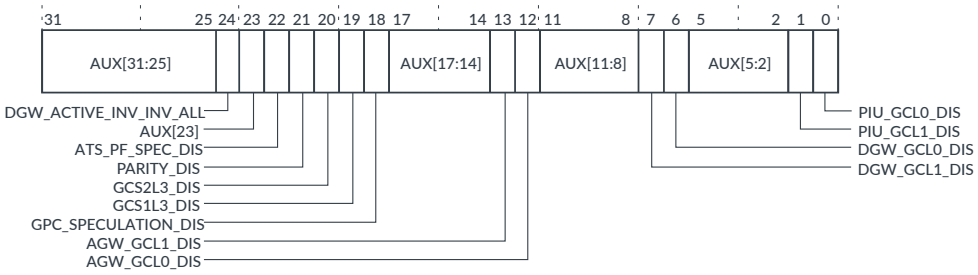


Table 3-29: TCU_ROOT_CTRL register bit descriptions

Bits	Name	Description
[31:25]	AUX[31:25]	Reserved. Do not change these bits unless Arm advises you to. If writing other bits of this register, ensure that you write the current value for these bits to them, for example, by performing a read-modify-write sequence.
[24]	DGW_ACTIVE_INV_INV_ALL	In the DTI GPC Wrapper (DGW), active INV causes everything to be invalidated regardless of matching with an invalidate.
[23]	AUX[23]	Reserved. Do not change these bits unless Arm advises you to. If writing other bits of this register, ensure that you write the current value for these bits to them, for example, by performing a read-modify-write sequence.
[22]	ATS_PF_SPEC_DIS	Split stage ATS IPA message prefetch speculation disable. The GPT cache is not updated for GPT walks at this level.
[21]	PARITY_DIS	Ignore any parity errors from registers with parity bits.
[20]	GCS2L3_DIS	Disable update of GCB resulting from a walk for S2L3.
[19]	GCS1L3_DIS	Disable update of GCB resulting from a walk for S1L3.
[18]	GPC_SPECULATION_DIS	Disables speculation on GPC checks to prevent the TCU from starting a page table walk until the GPC check for the PA of the page table entry has been performed. When set to 0, the TCU starts the GPT walk and the page table walk in parallel for improved performance.
[17:14]	AUX[17:14]	Reserved. Do not change these bits unless Arm advises you to. If writing other bits of this register, ensure that you write the current value for these bits to them, for example, by performing a read-modify-write sequence.
[13]	AGW_GCL1_DIS	GCB disable. When a bit of this field is set, it disables the corresponding level of GCB for the AGW GC inside the TMU.
[12]	AGW_GCL0_DIS	
[11:8]	AUX[11:8]	Set this field to 0 unless we advise otherwise.
[7]	DGW_GCL1_DIS	GCB disable. When a bit of this field is set, it disables the corresponding level of GCB for the DGW GC inside the TMU.
[6]	DGW_GCL0_DIS	
[5:2]	AUX[5:2]	Set this field to 0 unless we advise otherwise.
[1]	PIU_GCL1_DIS	GCB disable. When a bit of this field is set, it disables the corresponding level of GCB for the AGW GC inside the PIU.
[0]	PIU_GCL0_DIS	

3.7.8 TCU_ROOT_CTRL2 register

The TCU_ROOT_CTRL2 register disables TCU features. Do not change the bits in this register unless Arm advises you to.

The TCU_ROOT_CTRL2 register is root-only by default. It can be made accessible by Secure or Non-secure by setting various combinations of the 3.7.5 TCU_SCR register on page 161 and the 3.7.10 TCU_RCR register on page 166.

Configurations

This register is available in all configurations.

Attributes

The TCU_ROOT_CTRL2 register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28F04`

Bit descriptions

Table 3-30: TCU_ROOT_CTRL2 bit descriptions

Bits	Name	Description
[31:0]	AUX31:0	Reserved. Do not change these bits unless Arm advises you to.

3.7.9 TCU_R_CTRL register

The TCU_R_CTRL register disables TCU features. Do not change the bits in this register unless Arm advises you to.

The TCU_R_CTRL register is for the Realm world only.

Configurations

This register is available in configurations where the `TCUCFG_LEGACY_TZ_EN` parameter is set to 0.

Attributes

The TCU_R_CTRL register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

0x28F08

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-8: TCU_R_CTRL register bit assignments

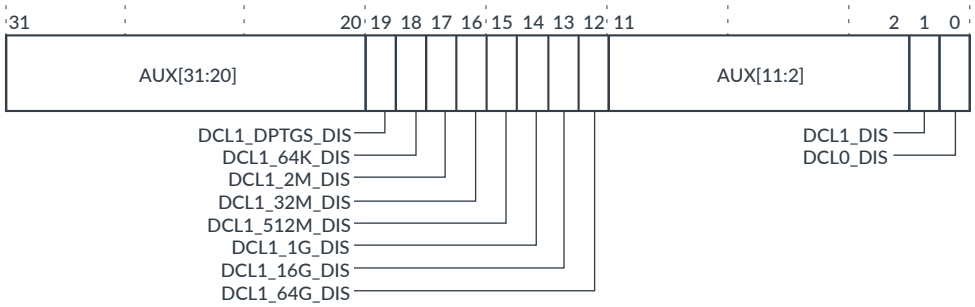


Table 3-31: TCU_R_CTRL register bit descriptions

Bits	Name	Description
[31:20]	AUX[31:20]	Reserved. Do not change these bits unless Arm advises you to.
[19]	DCL1_DPTGS_DIS	Disable lookups and update for L1 DPT at the size defined by SMMU_DPT_BASE_CFG.DPTGS, that is, the granule size
[18]	DCL1_64K_DIS	Disable lookups and update for L1 DPT 64KB size
[17]	DCL1_2M_DIS	Disable lookups and update for L1 DPT 2MB size
[16]	DCL1_32M_DIS	Disable lookups and update for L1 DPT 32MB size
[15]	DCL1_512M_DIS	Disable lookups and update for L1 DPT 512MB size
[14]	DCL1_1G_DIS	Disable lookups and update for L1 DPT 1GB size
[13]	DCL1_16G_DIS	Disable lookups and update for L1 DPT 16GB size
[12]	DCL1_64G_DIS	Disable lookups and update for L1 DPT 64GB size
[11:2]	AUX[11:2]	Reserved. Do not change these bits unless Arm advises you to.
[1]	DCL1_DIS	Disable L1 DPT cache
[0]	DCL0_DIS	Disable L0 DPT cache

3.7.10 TCU_RCR register

This register is accessible only from Root software. Some values are reset to 1 to give access to the Secure world by default.

The TCU_RCR register does not control Root access to the Performance Monitor registers. The SMMU_PMCG_SCR register controls Root access to these registers as the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* defines.

The TCU_RCR register also affects the 3.9 TCU system discovery registers on page 187, 3.12 TCU PIU integration registers on page 233, and 3.13 TCU TMU integration registers on page 243 in the same way.

If the `TCUCFG_LEGACY_TZ_EN` parameter is set to 1 or the legacy_tz_en signal is set to 1, all bits in this register behave as 1 and the TBU_RCR register is RAZ/WI.

Configurations

This register is available in all configurations.

Attributes

The TCU_RCR register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28F10`

Type

RW

Reset value

See 'Reset' column in the following table.

Usage constraints

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-9: TCU_RCR register bit assignments

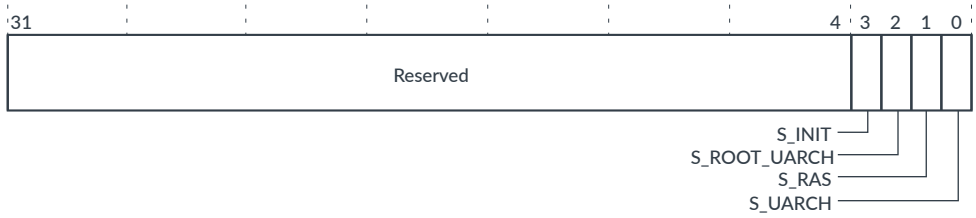


Table 3-32: TCU_RCR register bit descriptions

Bits	Name	Reset	Description
[31:4]	-	-	Reserved
[3]	S_INIT	1	Non-root register access permitted to the SMMU_S_INIT register.

Bits	Name	Reset	Description
[2]	S_ROOT_UARCH	0	Non-root register access permitted for the following Root-specific microarchitectural registers: • 3.11.1 ITEN register for the TCU on page 231 • 3.12.1 ITOP_PIU register for the TCU Programmer Interface Unit on page 234 • 3.12.2 ITIN_PIU register for the TCU Programmer Interface Unit on page 242 • 3.13.1 ITOP_TMU register for the TCU Translation Management Unit on page 243 • 3.13.2 ITIN_TMU register for the TCU Translation Management Unit on page 245 • 3.7.7 TCU_ROOT_CTRL register on page 163 • 3.7.8 TCU_ROOT_CTRL2 register on page 165
[1]	S_RAS	1	Non-root register access permitted for RAS registers. When this bit is 0, Non-root writes to the following register addresses are ignored, and Non-secure reads return zero: `0x006_0000-0x006_FFFC`
[0]	S_UARCH	1	Non-root register access permitted for microarchitectural registers. When this bit is 0, Non-root writes to some register are ignored, and Non-root reads return zero, as 3.1.1 Realm Management Extension Register Principles on page 131 describes. The TCU_RCR register also affects System discovery registers in the same way.

3.7.11 TCU_NODE_CTRLn register

The TCU_NODE_CTRLn register controls how the TCU communicates with a single DTI requester, either a TBU or a PCIe Root Port implementing ATS.

Each DTI requester, for example, a TBU has a node ID, each with TCU_NODE_CTRLn register in sequential addresses:

Node 0

TCU_NODE_CTRL0 at address `0x29000`

Node 1

TCU_NODE_CTRL1 at address `0x29004`

...

...

Node n

TCU_NODE_CTRLn at address `0x29000` + (4 × (`TCUCFG_NUM_TBU` - 1))

The number of registers that are implemented corresponds to the value of `TCUCFG_NUM_TBU`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

All bits [31:16] are implemented and are readable and writable, regardless of the number of LTI ports the attached DTI node has. The following expression determines the priority that is associated with a transaction:

PRIORITY = (TCU_NODE_CTRLn.PRIORITY_SEL ?
TCU_NODE_CTRLn[((DTI_x_TRANS_REQ.QOS[2:0] × 2) + 16) + :2] :
TCU_NODE_CTRLn.DEFAULT_PRIORITY)

If a DTI node sends a translation request with an incorrect QoS value, the programmed value for the LTI port indicated in the QoS field is used. This value is used because it is not possible for the TCU to determine what is a valid or invalid port number.


Note

When the priority level is established, these are translated into QoS values by selection of the appropriate QOS_PTW* field from the TCU_QOS register, which override the value in the DTI *trans_req* message. This means that the transaction has its QoS value overridden but no additional information is required to be associated with the transaction. If the PRI level association is updated, the rest of the mechanism requires no alteration.

Configurations

This register is available in all configurations.

Attributes

The TCU_NODE_CTRL*n* register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x29000-0x293FC`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-10: TCU_NODE_CTRLn register bit assignments

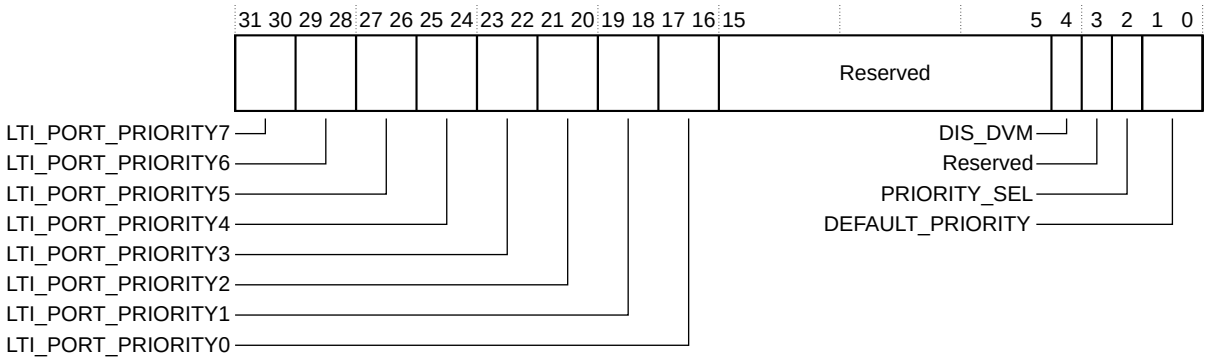


Table 3-33: TCU_NODE_CTRL*n* register bit descriptions

Bits	Name	Description
[31:30]	LTI_PORT_PRIORITY7	Priority level for LTI Port 7 for this node, if the port exists
[29:28]	LTI_PORT_PRIORITY6	Priority level for LTI Port 6 for this node, if the port exists
[27:26]	LTI_PORT_PRIORITY5	Priority level for LTI Port 5 for this node, if the port exists
[25:24]	LTI_PORT_PRIORITY4	Priority level for LTI Port 4 for this node, if the port exists
[23:22]	LTI_PORT_PRIORITY3	Priority level for LTI Port 3 for this node, if the port exists
[21:20]	LTI_PORT_PRIORITY2	Priority level for LTI Port 2 for this node, if the port exists
[19:18]	LTI_PORT_PRIORITY1	Priority level for LTI Port 1 for this node, if the port exists
[17:16]	LTI_PORT_PRIORITY0	Priority level for LTI Port 0 for this node
[15:5]	-	Reserved
[4]	DIS_DVM	Disable DVM. When this bit is set, the node does not participate in DVM invalidation. This setting can improve performance if the node can be slow to respond to invalidations issued over DTI. **Note:** The DIS_DVM setting does not affect invalidation by PA. This bit is only used for TBU nodes and is ignored for ATS nodes.
[3]	-	Reserved
[2]	PRIORITY_SEL	Select the priority between the DTI node ID, default and the LTI port: **0** When this bit is set to 0, the priority of all translation requests from the DTI requester, TBU, are given the priority that the DEFAULT_PRIORITY field specifies. **1** When this bit is set to 1, transactions from the DTI requester, TBU, are given the priority that is set in the LTI_PORT_PRIORITY*m* field, where the QOS[2:0] bits in the DTI translation request provide the value of *m*. **Note:** LTI PORT PRIORITY is not useful for ATS connections. Software is expected to write PRIORITY_SEL to 0 for ATS managers.

Bits	Name	Description
[1:0]	DEFAULT_PRIORITY	Default Priority level for this DTI requester, a TBU. Translation requests from a node with a higher priority level are normally progressed before translation requests from a node with a lower priority level.


Note

The 3.7.2 TCU_QOS register on page 157 contains four QoS level fields for the following:

- Translation table walks
- Configuration Table Walks (CTWs)
- Granule Protection Table (GPT) walks

These fields set the AxQOS value for each of the four levels. The TCU_NODE_CTRLn.LTI_PORT_PRIORITYm and TCU_NODE_CTRLn.DEFAULT_PRIORITY fields do not set the AxQOS value directly. Instead, these fields index into the TCU_QOS register to select which of the four QoS levels to use for walks that are associated with a particular LTI port or node. See 3.7.2 TCU_QOS register on page 157.

3.7.12 TCU_NODE_STATUSn register

The TCU_NODE_STATUSn register provides the status for each node. The TCU_NODE_STATUSn register and the TCU_NODE_CTRL*n* register each have a single status register.

The number of registers that are implemented corresponds to the value of `TCUCFG_NUM_TBU`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Configurations

This register is available in all configurations.

Attributes

The TCU_NODE_STATUS*n* register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x29400-0x297FC`

Type

RO

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-11: TCU_NODE_STATUS*n* register bit assignments

Table 3-34: TCU_NODE_STATUS*n* register bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3]	ATSv4	Indicates whether the node implements DTI-ATSv4. **0** The node is DTI-ATSv3, DTI-ATSv2, or DTI-ATSv1. **1** The node is DTI-ATSv4. This bit is valid only when ATS = 1. When ATS = 0, this bit is 0.
[2]	ATSv3	Indicates whether the node implements DTI-ATSv3. **0** The node is DTI-ATSv2 or DTI-ATSv1. **1** The node is DTI-ATSv3. This bit is valid only when ATS = 1. When ATS = 0, this bit is 0.
[1]	ATS	Indicates whether the node implements ATS: **0** The node is a TBU connected using DTI-TBU **1** The node is a PCIe Root Port supporting ATS, connected using DTI-ATS This bit is only valid when CONNECTED = 1. When CONNECTED = 0, this bit is 0.
[0]	CONNECTED	Indicates whether the DTI link for this node is in the connected state: **0** Node currently not in the connected state, including the states transitioning to and from the connected state **1** Node currently in the connected state When not connected, write accesses to TBU registers are ignored and read accesses return 0. However, the state might change between reading this register and attempting to access the TBU.

3.7.13 TCU_WC_SxLy_CMAX registers

TCU_WC_SxLy_CMAX registers enable you to set maximum capacities for the TCU walk cache RAMs, per stage and level.

The encoding of the TCU_WC_SxLy_CMAX registers is the same as the encoding for the MPAMCFG_CMAX registers that the *Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification* defines. These registers are readable and writable registers.

The following table describes the TCU_WC_SxLy_CMAX registers.

Table 3-35: TCU_WC_SxLy_CMAX registers

Address	Name	Field	Position	Description
0x29800	TCU_WC_S1L0_CMAX	CMAX	[15:0]	Maximum capacity for TCU Walk Cache stage 1 level 0
0x29804	TCU_WC_S1L1_CMAX	CMAX	[15:0]	Maximum capacity for TCU Walk Cache stage 1 level 1
0x29808	TCU_WC_S1L2_CMAX	CMAX	[15:0]	Maximum capacity for TCU Walk Cache stage 1 level 2
0x2980C	TCU_WC_S1L3_CMAX	CMAX	[15:0]	Maximum capacity for TCU Walk Cache stage 1 level 3
0x29810	TCU_WC_S2L0_CMAX	CMAX	[15:0]	Maximum capacity for TCU Walk Cache stage 2 level 0
0x29814	TCU_WC_S2L1_CMAX	CMAX	[15:0]	Maximum capacity for TCU Walk Cache stage 2 level 1
0x29818	TCU_WC_S2L2_CMAX	CMAX	[15:0]	Maximum capacity for TCU Walk Cache stage 2 level 2
0x2981C	TCU_WC_S2L3_CMAX	CMAX	[15:0]	Maximum capacity for TCU Walk Cache stage 2 level 3

Note

The implementation defines how many bits are used, depending on the cache configuration, but not more than 8 bits. Because the value represents a fixed-point binary faction, it is MSB aligned. Therefore, only bits [15:x] have any impact, where x can be 8, 9, 10, 11, 12, or 13.

3.7.14 TCU_AGW_GC_Lx_CMAX registers

TCU_AGW_GC_Lx_CMAX registers enable you to set maximum capacities for the TCU AGW GPTW cache RAMs, per level.

The encoding of the TCU_AGW_GC_Lx_CMAX registers is the same as the encoding for the MPAMCFG_CMAX registers that the *Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification* defines. These registers are readable and writable registers.

The following table describes the TCU_AGW_GC_Lx_CMAX registers.

Table 3-36: TCU_AGW_GC_Lx_CMAX registers

Address	Name	Field	Position	Description
0x29900	TCU_AGW_GC_L0_CMAX	CMAX	[15:0]	Maximum capacity for TCU GPT Cache level 0
0x29904	TCU_AGW_GC_L1_CMAX	CMAX	[15:0]	Maximum capacity for TCU GPT Cache level 1

Note

The implementation defines how many bits are used, depending on cache configuration, but not more than 8 bits. Because the value represents a fixed-point binary faction, it is MSB aligned, so only bits [15:x] have any impact, where x can be 8, 9, 10, 11, 12, or 13.

3.7.15 TCU_DGW_GC_Lx_CMAX registers

TCU_DGW_GC_Lx_CMAX registers enable you to set maximum capacities for the TCU DGW GPTW cache RAMs, per level.

The encoding of the TCU_DGW_GC_Lx_CMAX registers is the same as the encoding for the MPAMCFG_CMAX registers that the *Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification* defines. These registers are readable and writable registers.

The following table describes the TCU_DGW_GC_Lx_CMAX registers.

Table 3-37: TCU_DGW_GC_Lx_CMAX registers

Address	Name	Field	Position	Description
0x29910	TCU_DGW_GC_L0_CMAX	CMAX	[15:0]	Maximum capacity for TCU GPT Cache level 0
0x29914	TCU_DGW_GC_L1_CMAX	CMAX	[15:0]	Maximum capacity for TCU GPT Cache level 1

Note

The implementation defines how many bits are used, depending on cache configuration, but not more than 8 bits. Because the value represents a fixed-point binary faction, it is MSB aligned, so only bits [15:x] have any impact, where x can be 8, 9, 10, 11, 12, or 13.

3.7.16 TCU_DC_L0_CMAX and TCU_DC_L1_CMAX registers

TCU_DC_Lx_CMAX registers enable you to set maximum capacities for the TCU DPT cache, per level.

The encoding of the TCU_DC_Lx_CMAX registers is the same as the encoding for the MPAMCFG_CMAX registers that the *Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification* defines. These registers are readable and writable registers.

The following table describes the TCU_DC_L0_CMAX and TCU_DC_L1_CMAX registers.

Table 3-38: TCU_DC_L0_CMAX and TCU_DC_L1_CMAX registers

Address	Name	Field	Position	Description
0x29918	TCU_DC_L0_CMAX	CMAX	[15:0]	Maximum capacity for TCU DPT Cache level 0
0x2991C	TCU_DC_L1_CMAX	CMAX	[15:0]	Maximum capacity for TCU DPT Cache level 1

Note

The implementation defines how many bits are used, depending on cache configuration, but not more than 8 bits. Because the value represents a fixed-point binary faction, it is MSB-aligned, so only bits [15:x] have any impact, where x can be 8, 9, 10, 11, 12, or 13.

3.8 TCU RAS registers

The MMU S3 TCU contains Reliability, Availability, and Serviceability (RAS) registers.

The RAS registers implement the RAS Extension registers, single record format:

- Non-root access to these registers when TCU_RCR.S_RAS = 0 are RAZ/WI. See 3.7.10 TCU_RCR register on page 166.
- Non-secure accesses to these registers, when TCU_SCR.NS_RAS = 0, are RAZ/WI. See 3.7.5 TCU_SCR register on page 161.

Note

If the `TCUCFG_LEGACY_TZ_EN` parameter is set to 1 or the legacy_tz_en signal is set to 1:

- All bits in the 3.7.10 TCU_RCR register on page 166 behave as 1.
- The 3.16.3 TBU_SCR register on page 253 is RAZ/WI.

The RAS registers enable software to monitor the following classes of error:

- Corrected Errors (CEs) in the RAMs that the configuration cache uses.
- CEs in the RAMs used by the walk caches.

3.8.1 TCU_ERRFR register

To discover how the TCU handles errors, use the TCU Error Feature register.

Configurations

This register is available in all configurations.

Attributes

The TCU_ERRFR register attributes are as follows:

Width

32-bit

Functional group

3.4.3 TCU Reliability, Availability, and Service registers summary on page 142.

Address offset

0x60000

Type

Root, RO

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-12: TCU_ERRFR register bit assignments

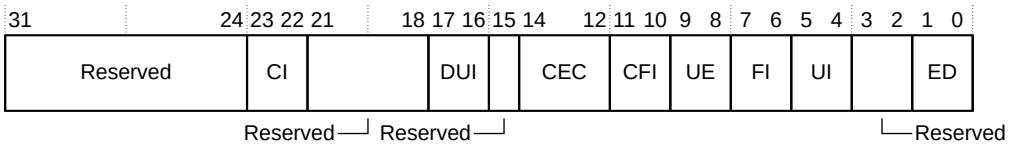


Table 3-39: TCU_ERRFR register bit descriptions

Bits	Name	Description
[31:24]	-	Reserved
[23:22]	CI	Critical Error Interrupt is always enabled. Value is `0b01`.
[21:18]	-	Reserved
[17:16]	DUI	Does not support this feature. Value is `0b00`.
[15]	-	Reserved
[14:12]	CEC	Does not implement the standard corrected error counter model. Value is `0b000`.
[11:10]	CFI	Does not support this feature. Value is `0b00`.
[9:8]	UE	In-band error signaling feature is always enabled. Value is `0b01`.
[7:6]	FI	Fault handling interrupt is controllable. Value is `0b10`.
[5:4]	UI	Error Recovery Interrupt always enabled for UE. Value is `0b01`.
[3:2]	-	Reserved
[1:0]	ED	Error detection is always enabled. Value is `0b01`.

3.8.2 TCU_ERRCTLR register

To enable fault handling interrupts, use the TCU Error Control register.

Configurations

This register is available in all configurations.

Attributes

The TCU_ERRCTLR register attributes are as follows:

Width

32-bit

Functional group

3.4.3 TCU Reliability, Availability, and Service registers summary on page 142.

Address offset

0x60008

Type

Root, RW

Reset value

1

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-13: TCU_ERRCTLR register bit assignments

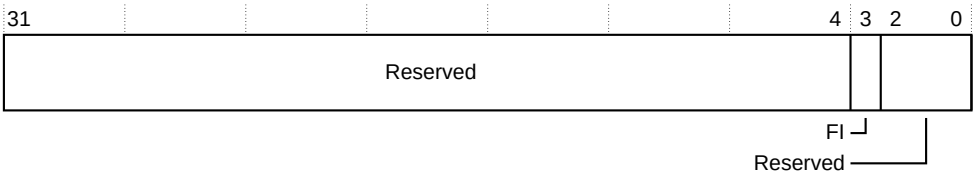


Table 3-40: TCU_ERRCTLR register bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3]	FI	Fault handling interrupt enable
[2:0]	-	Reserved

3.8.3 TCU_ERRSTATUS register

Use the TCU error status register to find out whether different types of error have occurred. Certain bits in this register are cleared by writing a 1 to their bit position. These writes are ignored in certain circumstances to avoid race conditions where a new error has occurred which software has not yet observed.

Configurations

This register is available in all configurations.

Attributes

The TCU_ERRSTATUS register attributes are as follows:

Width

32-bit

Functional group

3.4.3 TCU Reliability, Availability, and Service registers summary on page 142.

Address offset

0x60010

Type

Root, RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-14: TCU_ERRSTATUS register bit assignments

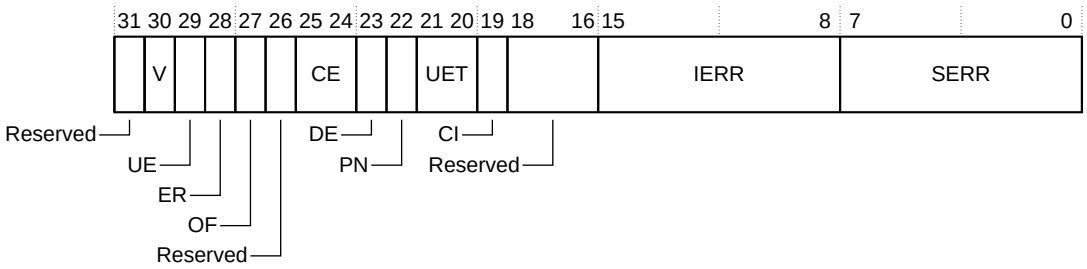


Table 3-41: TCU_ERRSTATUS register bit descriptions

Bits	Name	Description
[31]	-	Reserved
[30]	V	The status Register is valid. The possible values of this bit are as follows: **0** ERRSTATUS is not valid. **1** ERRSTATUS is valid. At least 1 error has been recorded. If any of the UE, DE, or CE bits are set to 1, and are not being cleared to 0 in the same write, direct writes to this bit are ignored. This bit is read/write-one-to-clear. This bit resets to zero on a reset.
[29]	UE	Uncorrected error, or errors. The possible values of this bit are as follows: **0** No errors that could not be corrected or deferred **1** At least one error detected that has not been corrected or deferred If the OF bit is set to 1 and is not being cleared to zero in the same write, direct writes to this bit are ignored. This bit is read/write-one-to-clear.
[28]	ER	Error Reported. The possible values of this bit are as follows: **0** No in-band error (External abort) is reported **1** The node to the requester making the access or other transaction signaled an External abort This bit is read/write-one-to-clear.

Bits	Name	Description
[27]	OF	Overflow. Multiple errors are detected. This bit is set to 1 when: • Multiple errors are detected on the same cycle • A new error occurs when there is already a valid record in the register This bit is read/write-one-to-clear.
[26]	-	Reserved
[25:24]	CE	Correctable Error, or errors. **0b00** No correctable errors recorded **0b10** At least one Corrected error recorded Other values are Reserved. This field is cleared by writing `0b11` to it. If OF is set and not being cleared, the write is ignored. A write of any value other than `0b11` is ignored.
[23]	DE	Deferred error, or errors. The possible values of this bit are as follows: **0** No errors were deferred **1** At least one error was not corrected and deferred If the OF bit is set to 1 and is not being cleared to 0 in the same write, direct writes to this bit are ignored. This bit is read/write-one-to-clear.
[22]	PN	Poison. The possible values of this bit are as follows: **0** Uncorrected error or deferred error is recorded because a corrupt value was detected, for example, by an Error Detection Code (EDC) **1** Uncorrected error or deferred error is recorded because a poison value was detected This bit is read/write-one-to-clear.
[21:20]	UET	Uncorrected Error Type. Describes the state of the component after detecting or consuming an Uncorrected error. The possible values of this field are as follows: **0b00** Uncorrected error, Uncontainable error (UC) **0b11** Uncorrected error, Signaled or Recoverable error (UER) Accessing this field has the following behavior. This field is not valid and reads UNKNOWN if any of the following are true: • TCU_ERRSTATUS.V == `0b0` • TCU_ERRSTATUS.UE == `0b0` Otherwise, this field is read/write-ones-to-clear. Writing a value other than all-zeros or all-ones sets this field to an UNKNOWN value.
[19]	CI	Indicates whether a critical error condition has been recorded. The possible values of this bit are as follows: **0** No critical error condition **1** Critical error condition This field is read/write-one-to-clear.
[18:16]	-	Reserved

Bits	Name	Description
[15:8]	IERR	**IMPLEMENTATION DEFINED** error code. When SERR ≠ 0, the value of this field indicates the source of the error: **80** PIU QTW FF PARITY **79** PIU PIB FF PARITY **78** PIU AGW GCB FF PARITY **77** PIU AGW GMB FF PARITY **76** PIU AGW FF PARITY **75** PIU INV FF PARITY **74** PIU CMD FF PARITY **73** PIU EVT FF PARITY **72** PIU PRI FF PARITY **71** PIU MGB FF PARITY **70** PIU CPB FF PARITY **69** PIU FAR FF PARITY **68** PIU PRG FF PARITY **67** PIU RAS FF PARITY **66** TMU DCB FF PARITY **65** TMU PMD FF PARITY **64** TMU PRG FF PARITY **63** TMU PMU FF PARITY **62** TMU WCB FF PARITY **61** TMU CCB FF PARITY Writes to this field are ignored.
[15:8]	IERR	IERR (continued) **IMPLEMENTATION DEFINED** error code. When SERR ≠ 0, the value of this field indicates the source of the error: **60** TMU HZU FF PARITY **59** TMU PIB FF PARITY **58** TMU DMD FF PARITY **57** TMU AGW GMB FF PARITY **56** TMU AGW GCB FF PARITY **55** TMU AGW FF PARITY **54** TMU DGW GMB FF PARITY **53** TMU DGW GCB FF PARITY **52** TMU DGW FF PARITY **51** TMU TWB FF PARITY **50** TMU DCB TAGS+DATA **49** TMU DCB REPL **48** TMU DCB PCNT **47** TMU DCB PLIM **46-44** Reserved **43** PIU AGW GMB WLK STATUS **42** PIU AGW GMB LOOP/PIU_GMB_POISON **41** PIU AGW GCB TAGS+DATA Writes to this field are ignored.

Bits	Name	Description
[15:8]	IERR	IERR (continued) **IMPLEMENTATION DEFINED** error code. When SERR ≠ 0, the value of this field indicates the source of the error: **40** PIU AGW GCB REPL **39** PIU CMD RPOISON **38** Reserved **37** TMU TWB WMB POISON **36** TMU TWB CMB POISON **35** TMU AGW ID RAM PARITY **34** Reserved **33** TMU AGW RESPONSE RAM PARITY **32** TMU AGW GMB SCRATCH **31** TMU AGW GMB WLK STATUS **30** TMU AGW GMB LOOP/TMU_AGW_GMB_POISON **29** TMU DGW RAM **28** TMU DGW GMB SCRATCH **27** TMU DGW GMB WLK STATUS **26** TMU DGW GMB LOOP/TMU_DGW_GMB_POISON **25** TMU AGW GCB TAGS+DATA **24** TMU DGW GCB TAGS+DATA **23** TMU AGW GCB REPL **22** TMU AGW GCB PCNT **21** TMU AGW GCB PLIM Writes to this field are ignored.
[15:8]	IERR	IERR (continued) **IMPLEMENTATION DEFINED** error code. When SERR ≠ 0, the value of this field indicates the source of the error: **20** TMU DGW GCB REPL **19** TMU DGW GCB PCNT **18** TMU DGW GCB PLIM **17** TMU CCB MCC DATA **16** TMU CCB MCC TAGS **15** TMU WCB MWC DATA **14** TMU WCB MWC TAGS **13** TMU CCB MCC REPL **12** TMU CCB MCC PCNT **11** TMU CCB MCC PLIM **10** TMU WCB MWC REPL **9** TMU WCB MWC PCNT **8** TMU WCB MWC PLIMH **7** TMU HZU SEC PTR **6** TMU HZU PTR **5** TMU HTTU RAM **4** TMU TWB WMB SCRATCH **3** TMU TWB WMB WLK STATUS **2** TMU TWB WMB LKP STATUS **1** Reserved **0** TMU TWB BSU Writes to this field are ignored.

Bits	Name	Description
[7:0]	SERR	The error code provides information about the earliest unacknowledged error. It can contain the following values: **2** Single or Double Error from RAMs that are not *GW, CCB, or WCB TAGS/DATA **8** Single or Double Error from CCB or WCB Data **9** Single or Double Error from CCB or WCB Tags or *GW TAGs or DATA **17** Internal control register FF parity error **21** Poisoned data read from downstream, `0x08` **26** Miscellaneous FF Parity Error All other values are reserved. Writes to this field are ignored.

3.8.4 TCU_ERRGEN register

Use the TCU Error Generation Register to test how the system responds when a RAS error occurs in the RAM.

The field locations are the same as for 3.8.3 TCU_ERRSTATUS register on page 177.

When this register is updated, the TCU_ERRSTATUS register is also updated with the same value, as long as the write data generates a valid error record.

A write to the TCU_ERRGEN register is valid if all the following conditions are met:

- The TCU_ERRGEN.V bit is set.
- At least 1 of the following is true (CE is legal if CE == `0b00` or CE == `0b10`):
  - The TCU_ERRGEN.UE bit is set and CE is legal
  - The TCU_ERRGEN.DE bit is set and CE is legal
  - The TCU_ERRGEN.CE bits are set to `0b10`
- One of the following is true:
  - TCU_ERRGEN.UET == `0b00`
  - TCU_ERRGEN.UET == `0b11` and UE == 1

The following also apply:

- All other fields can take any legal value.
- Any other write is considered to be invalid and the behavior of the SMMU is **UNPREDICTABLE**.
- If a valid error record is written, then the appropriate interrupt, or interrupts, are also generated.


Tip

This register has identical fields to the 3.8.3 TCU_ERRSTATUS register on page 177.

Configurations

This register is available in all configurations.

Attributes

The TCU_ERRGEN register attributes are as follows:

Width

64-bit

Functional group

3.4.3 TCU Reliability, Availability, and Service registers summary on page 142.

Address offset

0x60040

Type

Root, RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-15: TCU_ERRGEN register bit assignments

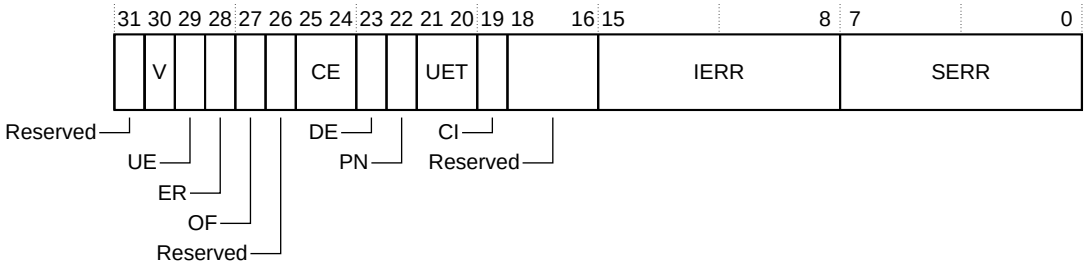


Table 3-42: TCU_ERRGEN register bit descriptions

Bits	Name	Description
[31]	-	Reserved
[30]	V	Status Register valid. The possible values of this bit are as follows: **0** ERRSTATUS is not valid **1** ERRSTATUS is valid. At least one error has been recorded. If any of the UE, DE, or CE bits are set to 1, and are not being cleared to 0 in the same write, direct writes to this bit are ignored. This bit is read/write-one-to-clear. This bit resets to zero on a reset.

Bits	Name	Description
[29]	UE	Uncorrected error, or errors. The possible values of this bit are as follows: **0** No errors that could not be corrected or deferred **1** At least one error detected that has not been corrected or deferred If the OF bit is set to 1 and is not being cleared to zero in the same write, direct writes to this bit are ignored. This bit is read/write-one-to-clear.
[28]	ER	Error Reported. The possible values of this bit are as follows: **0** No in-band error (External abort) is reported **1** The node to the requester making the access or other transaction signaled an External abort This bit is read/write-one-to-clear.
[27]	OF	Overflow. Multiple errors are detected. This bit is set to 1 when: • Multiple errors are detected on the same cycle • A new error occurs when there is already a valid record in the register This bit is read/write-one-to-clear.
[26]	-	Reserved
[25:24]	CE	Correctable Error, or errors. **0b00** No correctable errors recorded **0b10** At least one Corrected error recorded Other values are Reserved. This field is cleared by writing `0b11` to it. If OF is set and not being cleared, the write is ignored. A write of any value other than `0b11` is ignored.
[23]	DE	Deferred error, or errors. The possible values of this bit are as follows: **0** No errors were deferred **1** At least one error was not corrected and deferred If the OF bit is set to 1 and is not being cleared to 0 in the same write, direct writes to this bit are ignored. This bit is read/write-one-to-clear.
[22]	PN	Poison. The possible values of this bit are as follows: **0** Uncorrected error or deferred error is recorded because a corrupt value was detected, for example, by an *Error Detection Code* (EDC) **1** Uncorrected error or deferred error is recorded because a poison value was detected This bit is read/write-one-to-clear.

Bits	Name	Description
[21:20]	UET	Uncorrected Error Type. Describes the state of the component after detecting or consuming an Uncorrected error. The possible values of this field are as follows: **0b00** Uncorrected error, Uncontainable error (UC) **0b11** Uncorrected error, Signaled or Recoverable error (UER) Accessing this field has the following behavior. This field is not valid and reads UNKNOWN if any of the following are true: • TCU_ERRSTATUS.V == 0b0 • TCU_ERRSTATUS.UE == 0b0 Otherwise, this field is read/write-ones-to-clear. Writing a value other than all-zeros or all-ones sets this field to an UNKNOWN value.
[19]	CI	Indicates whether a critical error condition has been recorded. The possible values of this bit are as follows: **0** No critical error condition **1** Critical error condition This field is read/write-one-to-clear.
[18:16]	-	Reserved
[15:8]	IERR	**IMPLEMENTATION DEFINED** error code. When SERR ≠ 0, the value of this field indicates the source of the error: **80** PIU QTW FF PARITY **79** PIU PIB FF PARITY **78** PIU AGW GCB FF PARITY **77** PIU AGW GMB FF PARITY **76** PIU AGW FF PARITY **75** PIU INV FF PARITY **74** PIU CMD FF PARITY **73** PIU EVT FF PARITY **72** PIU PRI FF PARITY **71** PIU MGB FF PARITY **70** PIU CPB FF PARITY **69** PIU FAR FF PARITY **68** PIU PRG FF PARITY **67** PIU RAS FF PARITY **66** TMU DCB FF PARITY **65** TMU PMD FF PARITY **64** TMU PRG FF PARITY **63** TMU PMU FF PARITY **62** TMU WCB FF PARITY **61** TMU CCB FF PARITY Writes to this field are ignored.

Bits	Name	Description
[15:8]	IERR	IERR (continued) **IMPLEMENTATION DEFINED** error code. When SERR ≠ 0, the value of this field indicates the source of the error: **60** TMU HZU FF PARITY **59** TMU PIB FF PARITY **58** TMU DMD FF PARITY **57** TMU AGW GMB FF PARITY **56** TMU AGW GCB FF PARITY **55** TMU AGW FF PARITY **54** TMU DGW GMB FF PARITY **53** TMU DGW GCB FF PARITY **52** TMU DGW FF PARITY **51** TMU TWB FF PARITY **50** TMU DCB TAGS+DATA **49** TMU DCB REPL **48** TMU DCB PCNT **47** TMU DCB PLIM **46-44** Reserved **43** PIU AGW GMB WLK STATUS **42** PIU AGW GMB LOOP/PIU_GMB_POISON **41** PIU AGW GCB TAGS+DATA Writes to this field are ignored.
[15:8]	IERR	IERR (continued) **IMPLEMENTATION DEFINED** error code. When SERR ≠ 0, the value of this field indicates the source of the error: **40** PIU AGW GCB REPL **39** PIU CMD RPOISON **38** Reserved **37** TMU TWB WMB POISON **36** TMU TWB CMB POISON **35** TMU AGW ID RAM PARITY **34** Reserved **33** TMU AGW RESPONSE RAM PARITY **32** TMU AGW GMB SCRATCH **31** TMU AGW GMB WLK STATUS **30** TMU AGW GMB LOOP/TMU_AGW_GMB_POISON **29** TMU DGW RAM **28** TMU DGW GMB SCRATCH **27** TMU DGW GMB WLK STATUS **26** TMU DGW GMB LOOP/TMU_DGW_GMB_POISON **25** TMU AGW GCB TAGS+DATA **24** TMU DGW GCB TAGS+DATA **23** TMU AGW GCB REPL **22** TMU AGW GCB PCNT **21** TMU AGW GCB PLIM Writes to this field are ignored.

Bits	Name	Description
[15:8]	IERR	IERR (continued) **IMPLEMENTATION DEFINED** error code. When SERR ≠ 0, the value of this field indicates the source of the error: **20** TMU DGW GCB REPL **19** TMU DGW GCB PCNT **18** TMU DGW GCB PLIM **17** TMU CCB MCC DATA **16** TMU CCB MCC TAGS **15** TMU WCB MWC DATA **14** TMU WCB MWC TAGS **13** TMU CCB MCC REPL **12** TMU CCB MCC PCNT **11** TMU CCB MCC PLIM **10** TMU WCB MWC REPL **9** TMU WCB MWC PCNT **8** TMU WCB MWC PLIMH **7** TMU HZU SEC PTR **6** TMU HZU PTR **5** TMU HTTU RAM **4** TMU TWB WMB SCRATCH **3** TMU TWB WMB WLK STATUS **2** TMU TWB WMB LKP STATUS **1** Reserved **0** TMU TWB BSU Writes to this field are ignored.
[7:0]	SERR	The error code provides information about the earliest unacknowledged error. It can contain the following values: **2** Single or Double Error from RAMs that are not *GW, CCB, or WCB TAGS/DATA **8** Single or Double Error from CCB or WCB Data **9** Single or Double Error from CCB or WCB Tags or *GW TAGs or DATA **17** Internal control register FF parity error **21** Poisoned data read from downstream, `0x08` **26** Miscellaneous FF Parity Error All other values are reserved. Writes to this field are ignored.

3.9 TCU system discovery registers

The MMU S3 TCU contains system discovery registers.

3.9.1 TCU_SYSDISC0 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC0 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A000`

Type

RO

Reset value

`TCUCFG_WC_DEPTH`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-16: TCU_SYSDISC0 register bit assignments

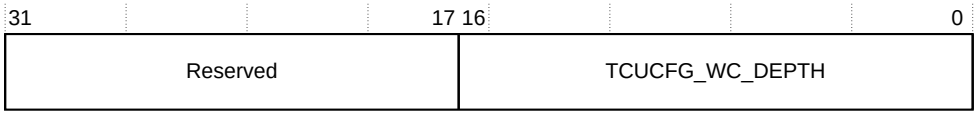


Table 3-43: TCU_SYSDISC0 register bit descriptions

Bits	Name	Description
[31:17]	-	Reserved

[16:0]	TCUCFG_WC_DEPTH	The read data reflects the chosen parameter value, for example:
		`17'h0_0008 : 8`
		....
		`17'h1_0000 : 65536`

3.9.2 TCU_SYSDISC1 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC1 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A004`

Type

RO

Reset value

`TCUCFG_CC_DEPTH`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-17: TCU_SYSDISC1 register bit assignments

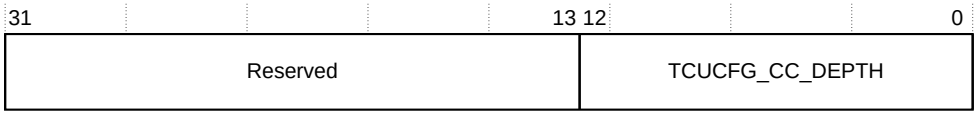


Table 3-44: TCU_SYSDISC1 register bit descriptions

Bits	Name	Description
[31:13]	-	Reserved

[12:0]	TCUCFG_CC_DEPTH	The read data reflects the chosen parameter value, for example:
		`13'h0004 : 4`
		....
		`13'h1000 : 4096`

3.9.3 TCU_SYSDISC2 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC2 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A008`

Type

RO

Reset value

`TCUCFG_WC_WAYS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-18: TCU_SYSDISC2 register bit assignments

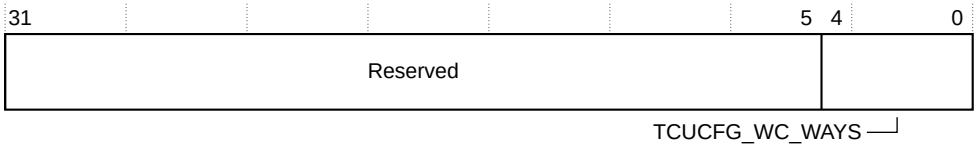


Table 3-45: TCU_SYSDISC2 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved

[4:0]	TCUCFG_WC_WAYS	The read data reflects the chosen parameter value, for example:
		`5'h04 : 4`
		
		....
		
		`5'h10 : 16`

3.9.4 TCU_SYSDISC3 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC3 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A00C`

Type

RO

Reset value

`TCUCFG_WC_BANKS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-19: TCU_SYSDISC3 register bit assignments

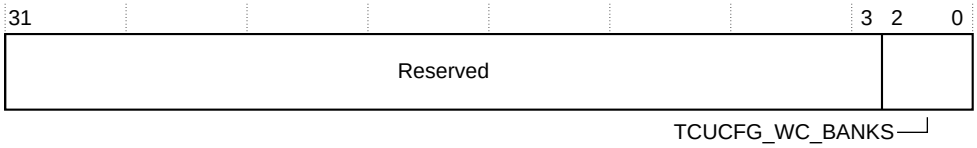


Table 3-46: TCU_SYSDISC3 register bit descriptions

Bits	Name	Description
[31:3]	-	Reserved

[2:0]	TCUCFG_WC_BANKS	The read data reflects the chosen parameter value, for example:
		`3'b001 : 1`
		....
		`3'b100 : 4`

3.9.5 TCU_SYSDISC4 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC4 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A010`

Type

RO

Reset value

`TCUCFG_XLATE_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-20: TCU_SYSDISC4 register bit assignments

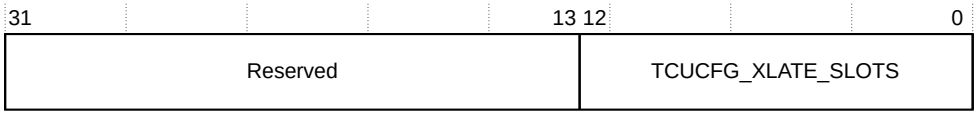


Table 3-47: TCU_SYSDISC4 register bit descriptions

Bits	Name	Description
[31:13]	-	Reserved

[12:0]	TCUCFG_XLATE_SLOTS	The read data reflects the chosen parameter value, for example:
		`13'h0004 : 4`
		
		....
		
		`13'h1000 : 4096`

3.9.6 TCU_SYSDISC5 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC5 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A014`

Type

RO

Reset value

`TCUCFG_PTW_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-21: TCU_SYSDISC5 register bit assignments

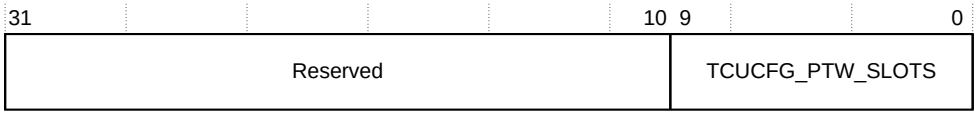


Table 3-48: TCU_SYSDISC5 register bit descriptions

Bits	Name	Description
[31:10]	-	Reserved

[9:0]	TCUCFG_PTW_SLOTS	The read data reflects the chosen parameter value, for example:
		`10'h002 : 2`
		....
		`10'h200 : 512`

3.9.7 TCU_SYSDISC6 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC6 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A018`

Type

RO

Reset value

`TCUCFG_CTW_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-22: TCU_SYSDISC6 register bit assignments

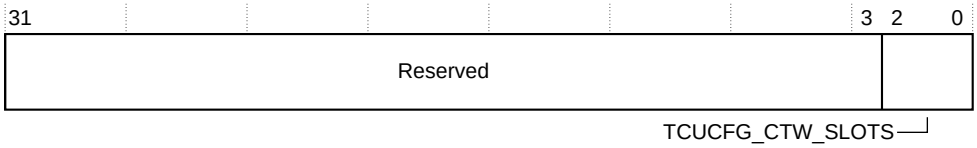


Table 3-49: TCU_SYSDISC6 register bit descriptions

Bits	Name	Description
[31:3]	-	Reserved

[2:0]	TCUCFG_CTW_SLOTS	The read data reflects the chosen parameter value, for example:
		`3'b001 : 1`
		
		....
		
		`3'b100 : 4`

3.9.8 TCU_SYSDISC7 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC7 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A01C`

Type

RO

Reset value

`TCUCFG_CC_IDXGEN_MODE`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-23: TCU_SYSDISC7 register bit assignments

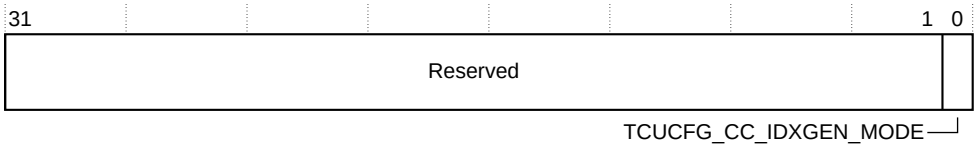


Table 3-50: TCU_SYSDISC7 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved

[0]	TCUCFG_CC_IDXGEN_MODE	The read data reflects the chosen parameter value, for example:
		`1'b0 : 0`
		....
		`1'b1 : 1`

3.9.9 TCU_SYSDISC8 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC8 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A020`

Type

RO

Reset value

`TCUCFG_DTI_ATS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-24: TCU_SYSDISC8 register bit assignments

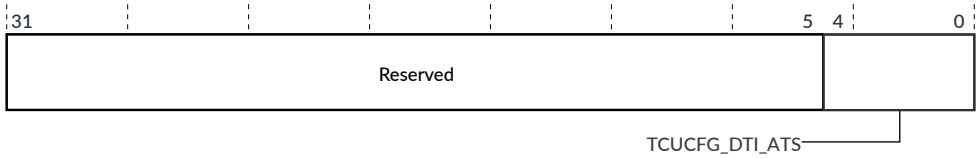


Table 3-51: TCU_SYSDISC8 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved

[4:0]	TCUCFG_DTI_ATS	The read data reflects the chosen parameter value, for example:
		`5'b00000 : 0`
		....
		`5'b10000 : 16`

3.9.10 TCU_SYSDISC9 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC9 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A024`

Type

RO

Reset value

`TCUCFG_NUM_TBU`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-25: TCU_SYSDISC9 register bit assignments

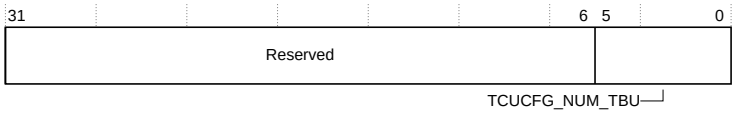


Table 3-52: TCU_SYSDISC9 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved

[5:0]	TCUCFG_NUM_TBU	The read data reflects the chosen parameter value, for example:
		`6'h01 : 1`
		....
		`6'h3E : 62`
		Note:
		Legal values are 14 and 62.

3.9.11 TCU_SYSDISC10 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC10 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A028`

Type

RO

Reset value

`TCUCFG_PMU_COUNTERS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-26: TCU_SYSDISC10 register bit assignments

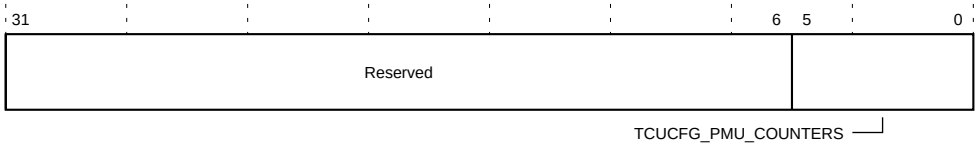


Table 3-53: TCU_SYSDISC10 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved

[5:0]	TCUCFG_PMU_COUNTERS	The read data reflects the chosen parameter value, for example:
		`6'h04 : 4`
		....
		`6'h20 : 32`

3.9.12 TCU_SYSDISC11 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC11 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

0x2A02C

Type

RO

Reset value

TCUCFG_PARTID_WIDTH. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-27: TCU_SYSDISC11 register bit assignments

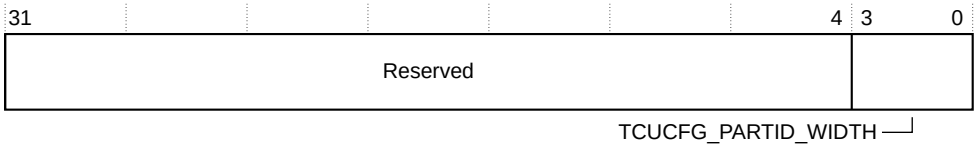


Table 3-54: TCU_SYSDISC11 register bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3:0]	TCUCFG_PARTID_WIDTH	The read data reflects the chosen parameter value, for example: 4'b0001 : 1 4'b0001 : 1 .... 4'b1010 : 10

3.9.13 TCU_SYSDISC12 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC12 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A030`

Type

RO

Reset value

`TCUCFG_HZU_DEPTH`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-28: TCU_SYSDISC12 register bit assignments

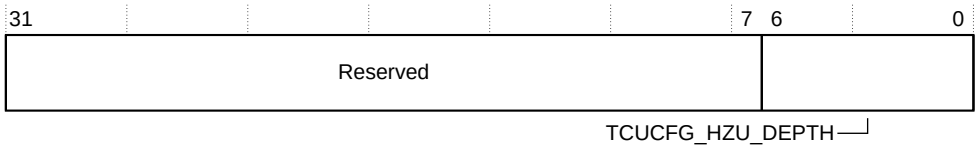


Table 3-55: TCU_SYSDISC12 register bit descriptions

Bits	Name	Description
[31:7]	-	Reserved

[6:0]	TCUCFG_HZU_DEPTH	The read data reflects the chosen parameter value, for example:
		`7'h02 : 2`
		....
		`7'h40 : 64`

3.9.14 TCU_SYSDISC13 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC13 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A034`

Type

RO

Reset value

`TCUCFG_PREFETCH_SUPPORTED`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-29: TCU_SYSDISC13 register bit assignments

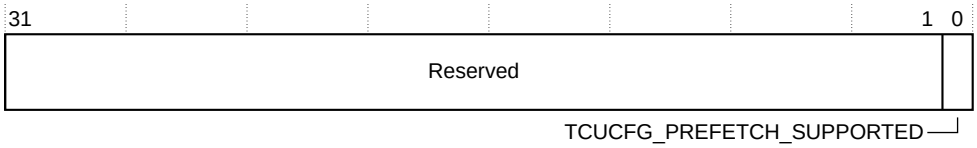


Table 3-56: TCU_SYSDISC13 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved

[0]	TCUCFG_PREFETCH_SUPPORTED	The read data reflects the chosen parameter value, for example:
		`1'b0 : 0`
		....
		`1'b1 : 1`

3.9.15 TCU_SYSDISC14 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC14 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A038`

Type

RO

Reset value

`TCUCFG_DATARAM_TYPE`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-30: TCU_SYSDISC14 register bit assignments

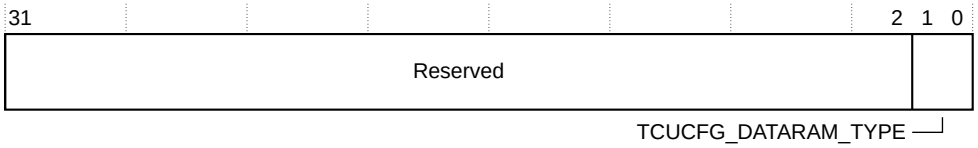


Table 3-57: TCU_SYSDISC14 register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved

[1:0]	TCUCFG_DATARAM_TYPE	The read data reflects the chosen parameter value, for example:
		`2'b00 : 0`
		....
		`2'b10 : 2`

3.9.16 TCU_SYSDISC15 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC15 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

0x2A03C

Type

RO

Reset value

TCUCFG_SLOTRAM_TYPE. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-31: TCU_SYSDISC15 register bit assignments

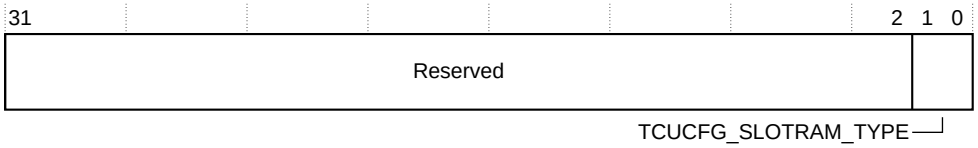


Table 3-58: TCU_SYSDISC15 register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1:0]	TCUCFG_SLOTRAM_TYPE	The read data reflects the chosen parameter value, for example: 2'b00 : 0 .... 2'b01 : 1

3.9.17 TCU_SYSDISC16 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC16 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

0x2A040

Type

RO

Reset value

TCUCFG_CACHERAM_TYPE. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-32: TCU_SYSDISC16 register bit assignments

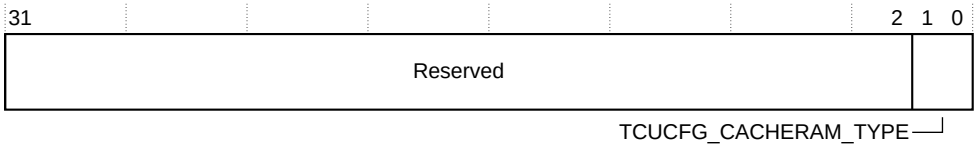


Table 3-59: TCU_SYSDISC16 register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1:0]	TCUCFG_CACHERAM_TYPE	The read data reflects the chosen parameter value, for example: 2'b00 : 0 .... 2'b01 : 1

3.9.18 TCU_SYSDISC17 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC17 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A044`

Type

RO

Reset value

`TCUCFG_QTW_DATA_WIDTH`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-33: TCU_SYSDISC17 register bit assignments

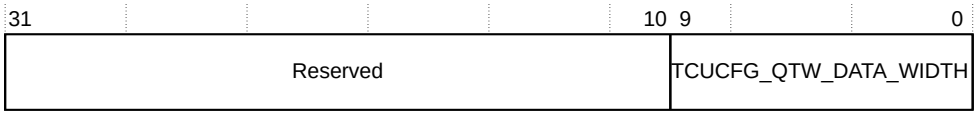


Table 3-60: TCU_SYSDISC17 register bit descriptions

Bits	Name	Description
[31:10]	-	Reserved

[9:0]	TCUCFG_QTW_DATA_WIDTH	The read data reflects the chosen parameter value, for example:
		`10'h040 : 64`
		....
		`10'h200 : 512`

3.9.19 TCU_SYSDISC18 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC18 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A048`

Type

RO

Reset value

`TCUCFG_AGW_GC_DEPTH`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-34: TCU_SYSDISC18 register bit assignments

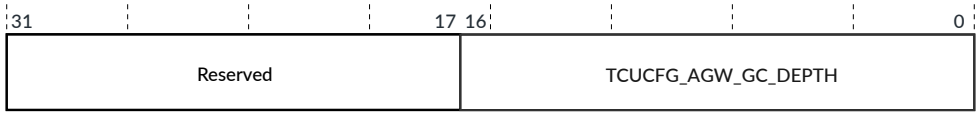


Table 3-61: TCU_SYSDISC18 register bit descriptions

Bits	Name	Description
[31:17]	-	Reserved
[16:0]	TCUCFG_AGW_GC_DEPTH	The read data reflects the chosen parameter value

3.9.20 TCU_SYSDISC19 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC19 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A04C`

Type

RO

Reset value

`TCUCFG_AGW_GC_WAYS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-35: TCU_SYSDISC19 register bit assignments

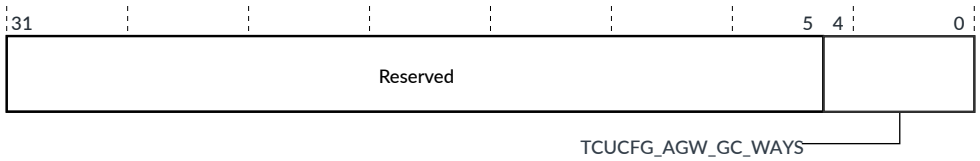


Table 3-62: TCU_SYSDISC19 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved
[4:0]	TCUCFG_AGW_GC_WAYS	The read data reflects the chosen parameter value

3.9.21 TCU_SYSDISC20 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC20 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

0x2A050

Type

RO

Reset value

TCUCFG_AGW_GC_BANKS. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-36: TCU_SYSDISC20 register bit assignments

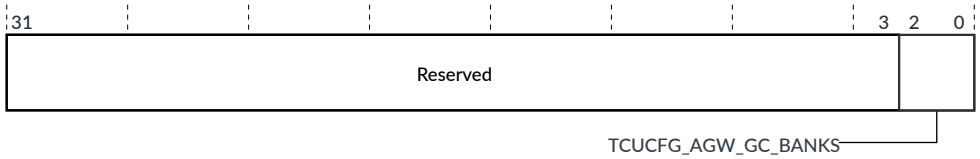


Table 3-63: TCU_SYSDISC20 register bit descriptions

Bits	Name	Description
[31:3]	-	Reserved
[2:0]	TCUCFG_AGW_GC_BANKS	The read data reflects the chosen parameter value

3.9.22 TCU_SYSDISC21 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC21 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

0x2A054

Type

RO

Reset value

`TCUCFG_DGW_GC_DEPTH`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-37: TCU_SYSDISC21 register bit assignments

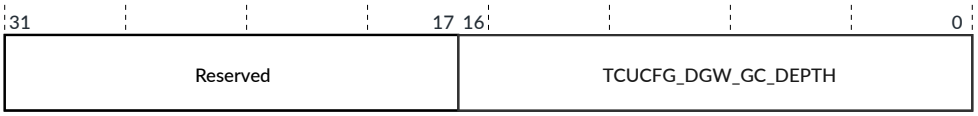


Table 3-64: TCU_SYSDISC21 register bit descriptions

Bits	Name	Description
[31:17]	-	Reserved
[16:0]	TCUCFG_DGW_GC_DEPTH	The read data reflects the chosen parameter value

3.9.23 TCU_SYSDISC22 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC22 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A058`

Type

RO

Reset value

`TCUCFG_DGW_GC_WAYS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-38: TCU_SYSDISC22 register bit assignments

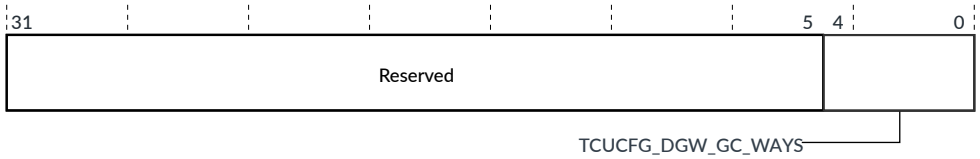


Table 3-65: TCU_SYSDISC22 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved
[4:0]	TCUCFG_DGW_GC_WAYS	The read data reflects the chosen parameter value

3.9.24 TCU_SYSDISC23 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC23 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A05C`

Type

RO

Reset value

`TCUCFG_DGW_GC_BANKS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-39: TCU_SYSDISC23 register bit assignments

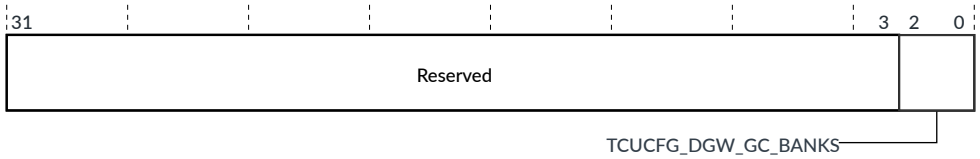


Table 3-66: TCU_SYSDISC23 register bit descriptions

Bits	Name	Description
[31:3]	-	Reserved
[2:0]	TCUCFG_DGW_GC_BANKS	The read data reflects the chosen parameter value

3.9.25 TCU_SYSDISC24 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC24 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A060`

Type

RO

Reset value

`TCUCFG_DGW_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-40: TCU_SYSDISC24 register bit assignments

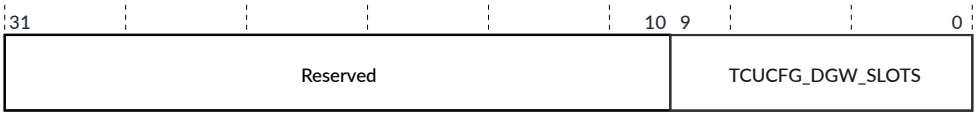


Table 3-67: TCU_SYSDISC24 register bit descriptions

Bits	Name	Description
[31:10]	-	Reserved
[9:0]	TCUCFG_DGW_SLOTS	The read data reflects the chosen parameter value

3.9.26 TCU_SYSDISC25 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC25 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A064`

Type

RO

Reset value

`TCUCFG_DTI_ATS_INV_MAX`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-41: TCU_SYSDISC25 register bit assignments

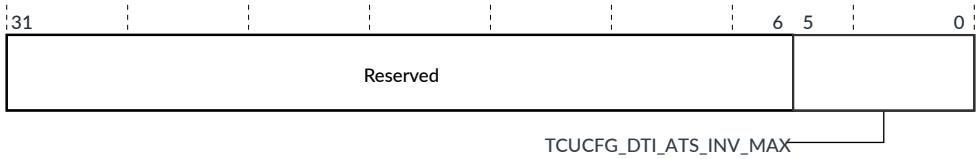


Table 3-68: TCU_SYSDISC25 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	TCUCFG_DTI_ATS_INV_MAX	The read data reflects the chosen parameter value

3.9.27 TCU_SYSDISC26 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC26 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A068`

Type

RO

Reset value

`TCUCFG_WC_LKP_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-42: TCU_SYSDISC26 register bit assignments

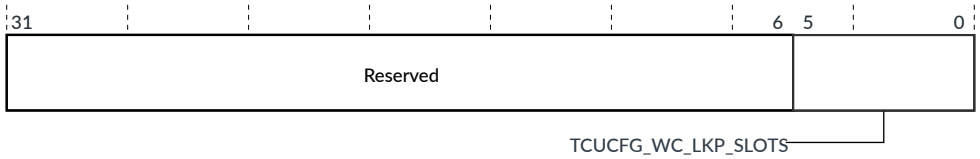


Table 3-69: TCU_SYSDISC26 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	TCUCFG_WC_LKP_SLOTS	The read data reflects the chosen parameter value

3.9.28 TCU_SYSDISC27 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC27 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A06C`

Type

RO

Reset value

`TCUCFG_AGW_GC_LKP_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-43: TCU_SYSDISC27 register bit assignments

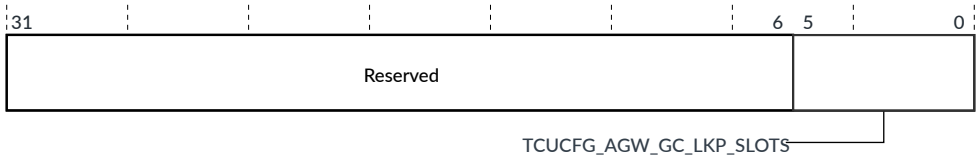


Table 3-70: TCU_SYSDISC27 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	TCUCFG_AGW_GC_LKP_SLOTS	The read data reflects the chosen parameter value

3.9.29 TCU_SYSDISC28 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC28 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A070`

Type

RO

Reset value

`TCUCFG_DGW_GC_LKP_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-44: TCU_SYSDISC28 register bit assignments

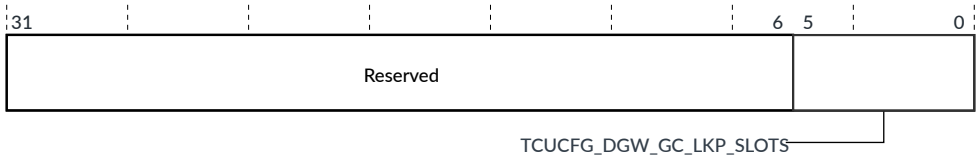


Table 3-71: TCU_SYSDISC28 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	TCUCFG_DGW_GC_LKP_SLOTS	The read data reflects the chosen parameter value

3.9.30 TCU_SYSDISC29 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC29 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A074`

Type

RO

Reset value

`TCUCFG_LEGACY_TZ_EN`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-45: TCU_SYSDISC29 register bit assignments

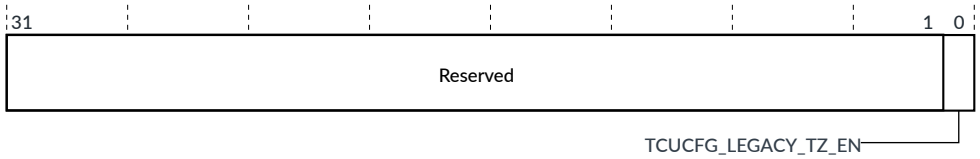


Table 3-72: TCU_SYSDISC29 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	TCUCFG_LEGACY_TZ_EN	The read data reflects the chosen parameter value

3.9.31 TCU_SYSDISC30 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC30 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A078`

Type

RO

Reset value

`TCUCFG_USE_ELA_DEBUG`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-46: TCU_SYSDISC30 register bit assignments

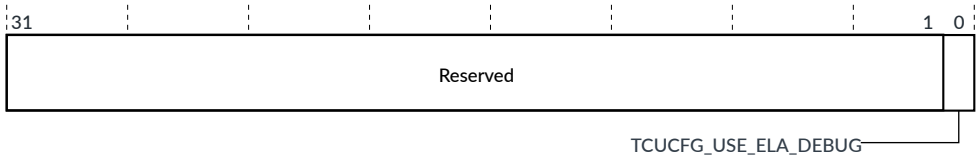


Table 3-73: TCU_SYSDISC30 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	TCUCFG_USE_ELA_DEBUG	The read data reflects the chosen parameter value

3.9.32 TCU_SYSDISC31 System Discovery Register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC31 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A07C`

Type

RO

Reset value

`TCUCFG_PARITY_ON`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-47: TCU_SYSDISC31 register bit assignments

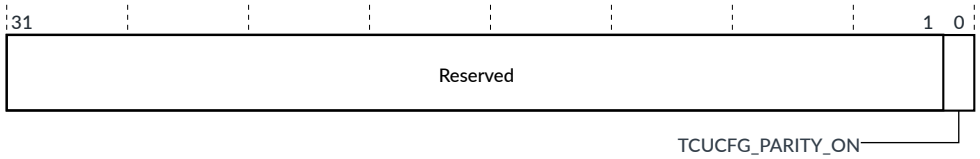


Table 3-74: TCU_SYSDISC31 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	TCUCFG_PARITY_ON	The read data reflects the chosen parameter value

3.9.33 TCU_SYSDISC32 System Discovery Register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC32 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A080`

Type

RO

Reset value

`TCUCFG_CC_WAYS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-48: TCU_SYSDISC32 register bit assignments

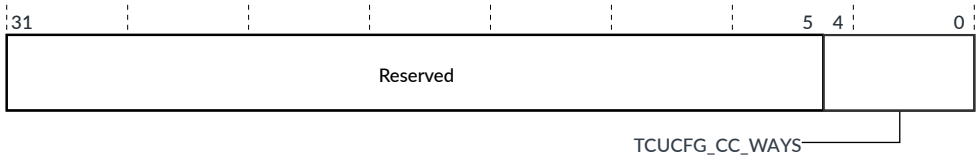


Table 3-75: TCU_SYSDISC32 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved
[4:0]	TCUCFG_CC_WAYS	The read data reflects the chosen parameter value

3.9.34 TCU_SYSDISC33 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC33 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A084`

Type

RO

Reset value

`TCUCFG_DVM_VAS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-49: TCU_SYSDISC33 register bit assignments

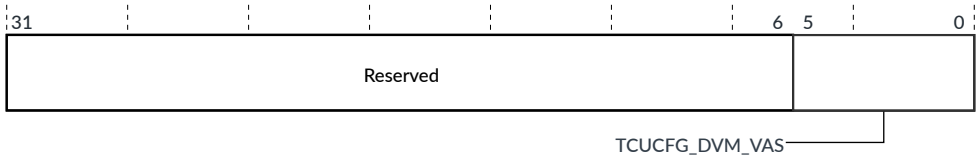


Table 3-76: TCU_SYSDISC33 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	TCUCFG_DVM_VAS	The read data reflects the chosen parameter value

3.9.35 TCU_SYSDISC34 System Discovery Register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC34 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A088`

Type

RO

Reset value

`TCUCFG_MECID_WIDTH`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-50: TCU_SYSDISC34 register bit assignments

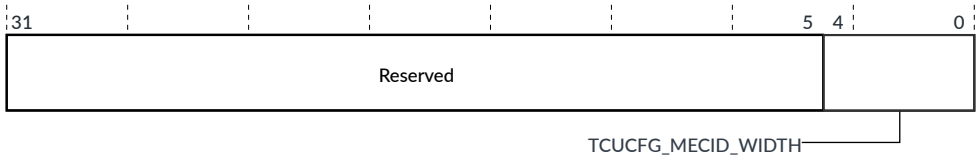


Table 3-77: TCU_SYSDISC34 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved
[4:0]	TCUCFG_MECID_WIDTH	The read data reflects the chosen parameter value

3.9.36 TCU_SYSDISC35 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC35 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A08C`

Type

RO

Reset value

`TCU_X2 (2 port variant).`

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-51: TCU_SYSDISC35 register bit assignments

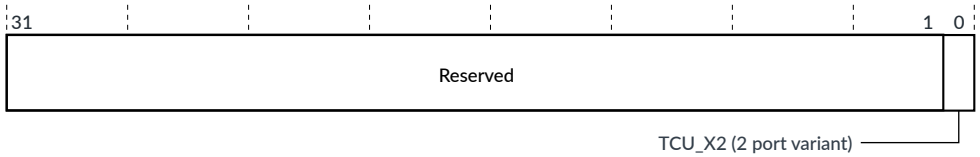


Table 3-78: TCU_SYSDISC35 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	TCU_X2 (2 port variant)	The read data reflects the chosen value

3.9.37 TCU_SYSDISC36 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC36 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A090`

Type

RO

Reset value

msi_addr[31:0].

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-52: TCU_SYSDISC36 register bit assignments

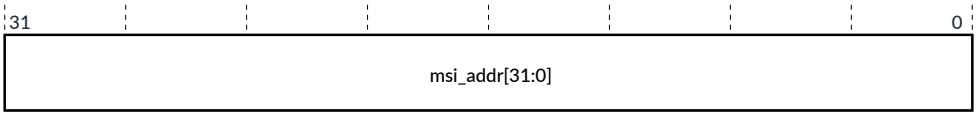


Table 3-79: TCU_SYSDISC36 register bit descriptions

Bits	Name	Description
[31:0]	msi_addr[31:0]	The read data reflects the chosen value.

3.9.38 TCU_SYSDISC37 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC37 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A094`

Type

RO

Reset value

msi_addr[51:32].

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-53: TCU_SYSDISC37 register bit assignments

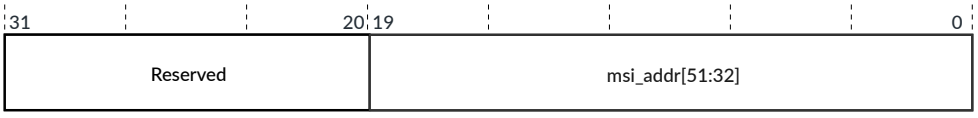


Table 3-80: TCU_SYSDISC37 register bit descriptions

Bits	Name	Description
[31:20]	-	Reserved.
[19:0]	msi_addr[51:32]	The read data reflects the chosen value.

3.9.39 TCU_SYSDISC38 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TCU_SYSDISC38 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A098`

Type

RO

Reset value

`TCUCFG_CC_LKP_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-54: TCU_SYSDISC38 register bit assignments

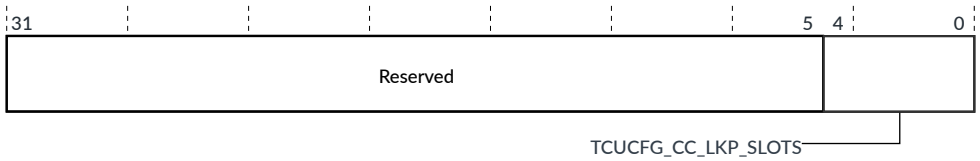


Table 3-81: TCU_SYSDISC38 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved.
[4:0]	TCUCFG_CC_LKP_SLOTS	The read data reflects the chosen parameter value.

3.9.40 TCU_SYSDISC39 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in configurations when the `TCUCFG_DPT_SUPPORT` parameter is set to 1.

Attributes

The TCU_SYSDISC39 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A09C`

Type

RO

Reset value

`TCUCFG_DPT_SUPPORT`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-55: TCU_SYSDISC39 register bit assignments

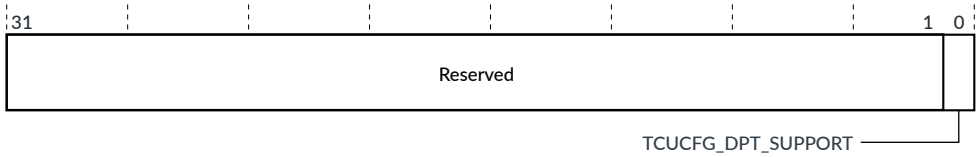


Table 3-82: TCU_SYSDISC39 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved.
[0]	TCUCFG_DPT_SUPPORT	The read data reflects the chosen parameter value.

3.9.41 TCU_SYSDISC40 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in configurations when the TCUCFG_DPT_SUPPORT parameter is set to 1.

Attributes

The TCU_SYSDISC40 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

0x2A0A0

Type

RO

Reset value

TCUCFG_DC_DEPTH. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-56: TCU_SYSDISC40 register bit assignments

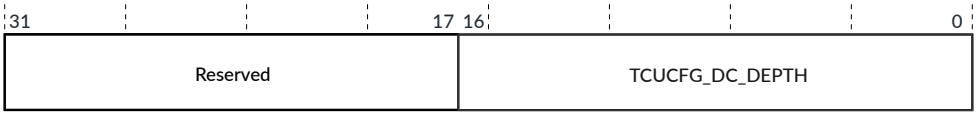


Table 3-83: TCU_SYSDISC40 register bit descriptions

Bits	Name	Description
[31:17]	-	Reserved.
[16:0]	TCUCFG_DC_DEPTH	The read data reflects the chosen value.

3.9.42 TCU_SYSDISC41 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in configurations when the TCUCFG_DPT_SUPPORT parameter is set to 1.

Attributes

The TCU_SYSDISC41 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

`0x2A0A4`

Type

RO

Reset value

`TCUCFG_DC_BANKS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-57: TCU_SYSDISC41 register bit assignments

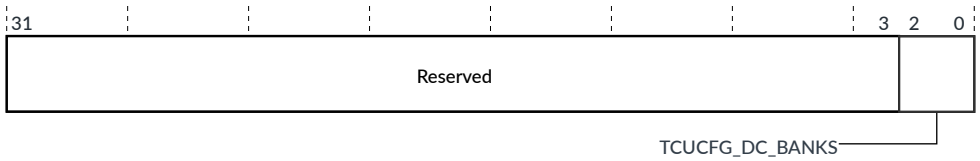


Table 3-84: TCU_SYSDISC41 register bit descriptions

Bits	Name	Description
[31:3]	-	Reserved.
[2:0]	TCUCFG_DC_BANKS	The read data reflects the chosen value.

3.9.43 TCU_SYSDISC42 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in configurations when the `TCUCFG_DPT_SUPPORT` parameter is set to 1.

Attributes

The TCU_SYSDISC42 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

0x2A0A8

Type

RO

Reset value

TCUCFG_DC_WAYS. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-58: TCU_SYSDISC42 register bit assignments

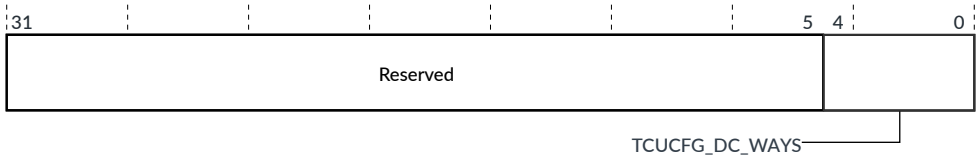


Table 3-85: TCU_SYSDISC42 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved.
[4:0]	TCUCFG_DC_WAYS	The read data reflects the chosen value.

3.9.44 TCU_SYSDISC43 system discovery register

The TCU system discovery registers discover components in the system.

Configurations

This register is available in configurations when the TCUCFG_DPT_SUPPORT parameter is set to 1.

Attributes

The TCU_SYSDISC43 register attributes are as follows:

Width

32-bit

Functional group

3.4.5 TCU system discovery registers summary on page 143.

Address offset

0x2A0AC

Type

RO

Reset value

`TCUCFG_DC_LKP_SLOTS`. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-59: TCU_SYSDISC43 register bit assignments

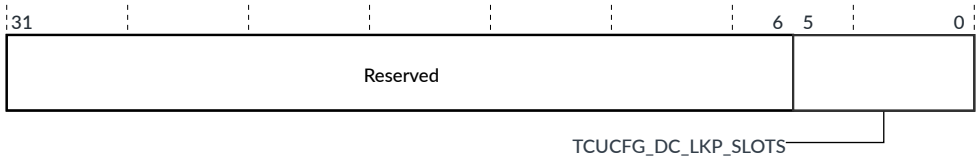


Table 3-86: TCU_SYSDISC43 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved.
[5:0]	TCUCFG_DC_LKP_SLOTS	The read data reflects the chosen value.

3.10 TCU_CTRL_AUX<n> registers

The TCU Control registers, AUX<n>, where <n> can have a value of 0-55, disable TCU features. Do not modify the AUX bits unless we instruct you to do so.

Configurations

This register is available in all configurations.

Attributes

TCU_CTRL_AUX<n> register attributes are as follows:

Width

32-bit

Functional group

3.4.6 TCU AUX registers summary on page 145

Address offset

`0x`
2A100-`0x2A348`

Type

RW

Reset value

See the reset value in the table in 3.4.6 TCU AUX registers summary on page 145.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-60: TCU_CTRL_AUX<n> register bit assignments

Table 3-87: TCU_CTRL_AUX<n> register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	AUX[0]	Reserved. Do not change these bits unless Arm advises you to.

3.11 TCU integration registers

The MMU S3 TCU contains TCU integration registers.

3.11.1 ITEN register for the TCU

Integration mode register for the TCU. When integration mode is enabled, the values of certain TCU input pins are made visible in the ITIN register for the TCU. The values that are written to the ITOP for the TCU control the values of certain TCU output pins. This mechanism helps system integrators to integrate the SMMU into the system and perform basic connectivity checks.

See also:

- 3.13.2 ITIN_TMU register for the TCU Translation Management Unit on page 245
- 3.13.1 ITOP_TMU register for the TCU Translation Management Unit on page 243

Configurations

This register is available in all configurations.

Attributes

The ITEN register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

0x28E20

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-61: ITEN register bit assignments

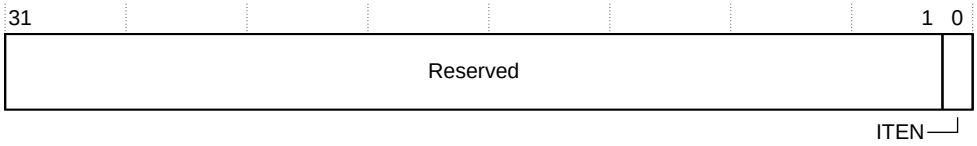


Table 3-88: ITEN register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	ITEN	**0** Disable integration mode **1** Enable integration mode When integration mode is disabled, the value of the ITOP register is cleared and becomes RO.

3.11.2 ITEN_ET register for the TCU

Edge triggered integration mode register for the TCU. When integration mode is enabled, the values of certain TCU input pins are made visible in the ITIN register for the TCU. When integration mode is enabled and triggered, the values that are written to the ITOP register for the TCU are driven to certain TCU output pins for a single cycle before being cleared. The mechanism helps system integrators to integrate the SMMU into the system and perform basic connectivity checks for single cycle interrupts.

See also:

- 3.13.2 ITIN_TMU register for the TCU Translation Management Unit on page 245
- 3.13.1 ITOP_TMU register for the TCU Translation Management Unit on page 243

Configurations

This register is available in all configurations.

Attributes

The ITEN_ET register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E34`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-62: ITEN_ET register bit assignments

Table 3-89: ITEN register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	TRIG	Write-Only A write of 1 to this field, when integration mode is enabled, triggers the values of ITOP to be present on outputs for a single cycle.
[0]	ITEN	Read-Write **0** Disable integration mode **1** Enable integration mode When integration mode is disabled, the value of the 3.12.1 ITOP_PIU register for the TCU Programmer Interface Unit on page 234 is cleared and becomes read only. **Note:** ITEN functionality takes priority over ITEN_ET functionality.

3.12 TCU PIU integration registers

The MMU S3 TCU contains Programmer Interface Unit (PIU) integration registers.

3.12.1 ITOP_PIU register for the TCU Programmer Interface Unit

The ITOP_PIU register is the integration output register for the TCU Programmer Interface Unit (PIU).

Configurations

This register is available in all configurations.

Attributes

The ITOP_PIU register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E24`

Type

RO or RW:

- RW when ITEN_ET.ITEN = 1 or ITEN.ITEN = 1.
- RO otherwise.

Reset value

0

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-63: ITOP_PIU register bit assignments

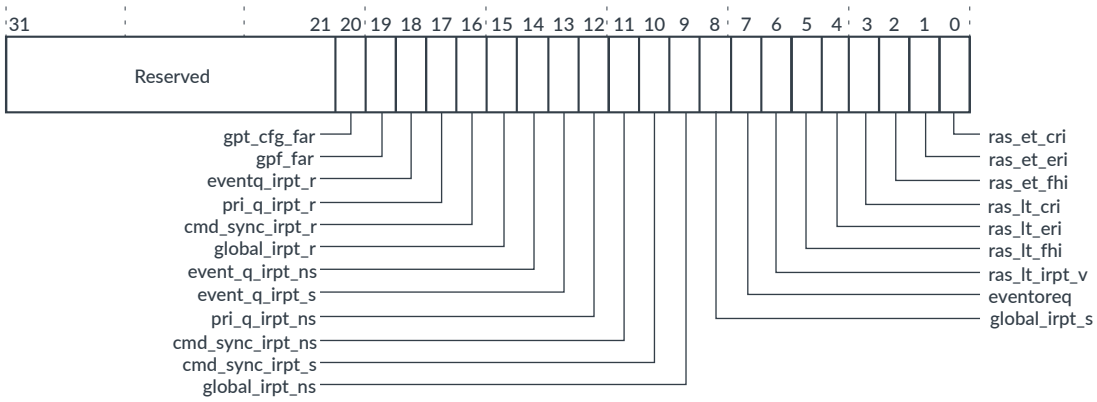


Table 3-90: ITOP_PIU register bit descriptions

Bits	Name	Description
[31:21]	-	Reserved, Should-Be-Zero (SBZ)
[20]	gpt_cfg_far	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[19]	gpf_far	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

Bits	Name	Description
[18]	eventq_irpt_r	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[17]	pri_q_irpt_r	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[16]	cmd_sync_irpt_r	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

Bits	Name	Description
[15]	global_irpt_r	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[14]	event_q_irpt_ns	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[13]	event_q_irpt_s	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

Bits	Name	Description
[12]	pri_q_irpt_ns	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[11]	cmd_sync_irpt_ns	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[10]	cmd_sync_irpt_s	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

Bits	Name	Description
[9]	global_irpt_ns	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[8]	global_irpt_s	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[7]	eventoreq	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

Bits	Name	Description
[6]	ras_lt_irpt_v	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[5]	ras_lt_fhi	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[4]	ras_lt_eri	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

Bits	Name	Description
[3]	ras_lt_cri	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[2]	ras_et_fhi	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
[1]	ras_et_eri	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

Bits	Name	Description
[0]	ras_et_cri	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1, this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

3.12.2 ITIN_PIU register for the TCU Programmer Interface Unit

The ITIN_PIU register is the integration input register for the TCU Programmer Interface Unit (PIU).

Configurations

This register is available in all configurations.

Attributes

The ITIN register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E28`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-64: ITIN_PIU register bit assignments

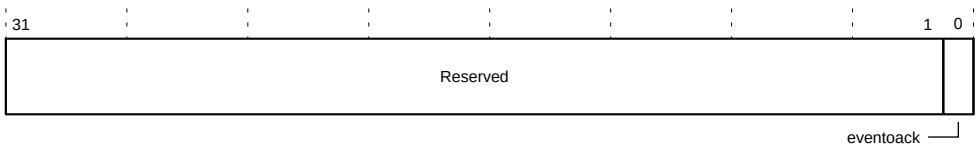


Table 3-91: ITIN_PIU register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, Should-Be-Zero (SBZ)
[0]	eventoack	When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field reflects the value of the eventoack signal. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field reflects the value of the eventoack signal See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312

3.13 TCU TMU integration registers

The MMU S3 TCU contains Translation Management Unit (TMU) integration registers.

3.13.1 ITOP_TMU register for the TCU Translation Management Unit

The MMU S3 TCU contains the integration output, ITOP_TMU register for the TCU Translation Management Unit (TMU).

Configurations

This register is available in all configurations.

Attributes

The ITOP register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E2C`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-65: ITOP_TMU register bit assignments

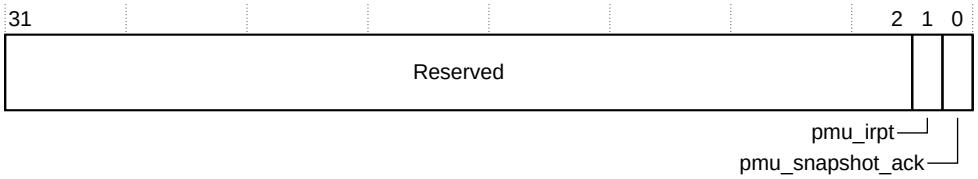


Table 3-92: ITOP_TMU register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved, Should-Be-Zero (SBZ)
[1]	pmu_irpt	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.3.2 PMU Snapshot signals on page 336
[0]	pmu_snapshot_ack	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1, and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.3.2 PMU Snapshot signals on page 336

3.13.2 ITIN_TMU register for the TCU Translation Management Unit

The MMU S3 TCU contains an integration input, ITIN_TMU register for the TCU Translation Management Unit (TMU).

Configurations

This register is available in all configurations.

Attributes

The ITIN register attributes are as follows:

Width

32-bit

Functional group

3.4.4 TCU microarchitectural registers summary on page 142.

Address offset

`0x28E30`

Type

RO

Reset value

0

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-66: ITIN_TMU register bit assignments

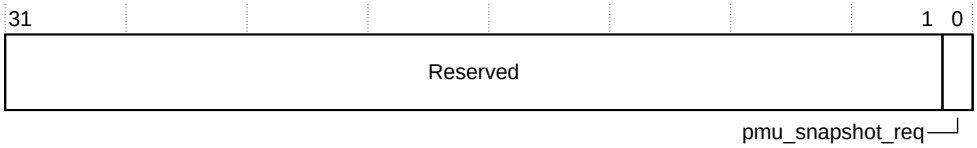


Table 3-93: ITIN_TMU register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, Should-Be-Zero (SBZ)

Bits	Name	Description
[0]	pmu_snapshot_req	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field reflects the value of the input signal specified. See: • 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • A.3.2 PMU Snapshot signals on page 336

3.14 TBU component and peripheral ID registers

The MMU S3 TBU contains component and peripheral ID registers.

The following table shows the TBU component and peripheral ID.

Table 3-94: TBU component and peripheral ID registers

Offset	Name	Field	Value	Description
0x00FFC	SMMU_CIDR3, Component ID3	[7:0]	0xB1	Preamble
0x00FF8	SMMU_CIDR2, Component ID2	[7:0]	0x05	Preamble
0x00FF4	SMMU_CIDR1, Component ID1	[7:0]	0xF0	Preamble
0x00FF0	SMMU_CIDR0, Component ID0	[7:0]	0x0D	Preamble
0x00FEC	SMMU_PIDR3, Peripheral ID3	[7:4]	MAX(*p_level*, ecorevnum)	REVAND, minor revision. Where *p_level* is 1 for p1.
	SMMU_PIDR3, Peripheral ID3	[3:0]	0x00	CMOD
0x00FE8	SMMU_PIDR2, Peripheral ID2	[7:4]	0x01	REVISION, major revision
	SMMU_PIDR2, Peripheral ID2	[3]	1	JEDEC-assigned value for DES always used
	SMMU_PIDR2, Peripheral ID2	[2:0]	3	DES_1: bits [6:4] bits of the JEP106 Designer code
0x00FE4	SMMU_PIDR1, Peripheral ID1	[7:4]	0xB	DES_0: bits [3:0] of the JEP106 Designer code
	SMMU_PIDR1, Peripheral ID1	[3:0]	0x4	PART_1: bits [11:8] of the Part number
0x00FE0	SMMU_PIDR0, Peripheral ID0	[7:0]	0x99	PART_0: bits [7:0] of the Part number
0x00FDC	SMMU_PIDR7, Peripheral ID7	-	RES0	Reserved
0x00FD8	SMMU_PIDR6, Peripheral ID6			
0x00FD4	SMMU_PIDR5, Peripheral ID5			
0x00FD0	SMMU_PIDR4, Peripheral ID4	[7:4]	0x0	SIZE = 4KB
		[3:0]	0x4	DES_2: JEP106 Designer continuation code

3.15 TBU PMU registers

The MMU S3 TBU contains Performance Monitor Unit (PMU) registers.

The TBU PMU registers follow the register layout that the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* Performance Monitor Extension describes.

3.15.1 Registers

The TBU and TCU support the same PMCG registers.

See 3.6 TCU PMU registers on page 150.

3.15.2 Events

Each event indicates whether the SMMU_PMCG_SMR0 register can filter it.

For events that cannot be filtered, you can set whether they are visible only when Secure events are visible by setting the following fields:

In RME mode

SMMU_PMCG_ROOTCR.NAO&(SMMU_PMCG_SCR.SO|SMMU_PMCG_SCR.NAO)

In legacy mode

SMMU_PMCG_SCR.SO

For more information, see the following:

- 2.5.2 Performance Monitoring Unit on page 56
- 2.5.2.2 SMMU TCU events on page 58

3.15.3 SMMU_PMCG_CFGR

An MMU S3 implementation assumes fixed values for SMMU_PMCG_CFGR, and these values define behavioral aspects of the implementation.

For information about the SMMU_PMCG_CFCR fields values, see 2.5.2.4 SMMUv3 PMU register architectural options on page 67.

See also 2.7 Configuration parameters and methodology on page 120.

3.15.4 SMMU_PMCG_CEID{0-1} registers

The SMMU_PMCG_CEID{0-1} registers indicate the architectural events that are supported. They are described as 64-bit registers, but they are accessed 32 bits at a time through the 32-bit DTI register access messages.

The following table shows the SMMU_PMCG_CEID{0-1} registers.

Table 3-95: SMMU_PMCG_CEID{0-1} registers

Offset	Name	Value
0x20E20	SMMU_PMCG_CEID0	0x00000087
0x20E28	SMMU_PMCG_CEID1	0x00000000

3.15.5 PMU ID registers

The PMU ID registers appear only in Performance Monitor Page 0. Page 1 does not contain any ID registers.

The following table shows the PMU ID registers.

Table 3-96: PMU ID registers

Offset	Name	Field	Value	Description
0x20FFC	SMMU_PMCG_CIDR3, Component ID3	[7:0]	0xB1	Preamble
0x20FF8	SMMU_PMCG_CIDR2, Component ID2	[7:0]	0x05	Preamble
0x20FF4	SMMU_PMCG_CIDR1, Component ID1	[7:0]	0x90	Preamble
0x20FF0	SMMU_PMCG_CIDR0, Component ID0	[7:0]	0x0D	Preamble
0x20FDC	SMMU_PMCG_PIDR7, Peripheral ID7	-	RES0	Reserved
0x20FD8	SMMU_PMCG_PIDR6, Peripheral ID6	-	RES0	Reserved
0x20FD4	SMMU_PMCG_PIDR5, Peripheral ID5	-	RES0	Reserved
0x20FD0	SMMU_PMCG_PIDR4, Peripheral ID4	[7:4]	0x0	SIZE = 4KB
		[3:0]	0x4	DES_2: JEP106 Designer continuation code
0x20FEC	SMMU_PMCG_PIDR3, Peripheral ID3	[7:4]	MAX(*p_level*, ecorevnum)	REVAND, minor revision, where *p_level* is 1 for p1.
	SMMU_PMCG_PIDR3, Peripheral ID3	[3:0]	0x00	CMOD
0x20FE8	SMMU_PMCG_PIDR2, Peripheral ID2	[7:4]	0x01	REVISION, major revision
	SMMU_PMCG_PIDR2, Peripheral ID2	[3]	1	JEDEC-assigned value for DES always used
	SMMU_PMCG_PIDR2, Peripheral ID2	[2:0]	3	DES_1: bits [6:4] bits of the JEP106 Designer code
0x20FE4	SMMU_PMCG_PIDR1, Peripheral ID1	[7:4]	0xB	DES_0: bits [3:0] of the JEP106 Designer code
	SMMU_PMCG_PIDR1, Peripheral ID1	[3:0]	0x4	PART_1: bits [11:8] of the Part number
0x20FE0	SMMU_PMCG_PIDR0, Peripheral ID0	[7:0]	0x99	PART_0: bits [7:0] of the Part number
0x20FB8	SMMU_PMCG_PMAUTHSTATUS	[7:0]	0x00	No authentication interface is implemented

The PMAUTHSTATUS, PMDEVARCH, and PMDEVTYPE registers are implemented as the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* defines.

3.16 TBU microarchitectural registers

You can set the microarchitectural registers at boot time to optimize TBU behavior for your system. We recommend that you use the default values for most systems.

Non-secure access to the Secure-only 3.16.3 TBU_SCR register on page 253 is RAZ/WI.

Secure and Non-secure accesses to microarchitectural registers, other than the 3.16.3 TBU_SCR register on page 253 are RAZ/WI except for when affected by fields in the 3.16.5 TBU_RCR register on page 256 and 3.16.3 TBU_SCR register on page 253.

The following table shows how SCR and RCR override the ownership of a register.

Table 3-97: SCR and RCR override of register ownership

SCR.feature	RCR.feature	Non-secure register	Secure register	Realm register	Root register
0	0	Non-secure	Secure	Realm	Root
0	1	Non-secure	Secure	Realm	Secure
1	0	Non-secure	Non-secure	Realm	Root
1	1	Non-secure	Non-secure	Realm	Non-secure

3.16.1 TBU_CTRL register

The TBU_CTRL register disables TBU features. Do not modify the bits in this register unless Arm® instructs you to do so.

Configurations

This register is available in all configurations.

Attributes

The TBU_CTRL register attributes are as follows:

Width

32-bit

Functional group

3.4.9 TBU microarchitectural registers summary on page 147.

Address offset

`0x08E00`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-67: TBU_CTRL register bit assignments

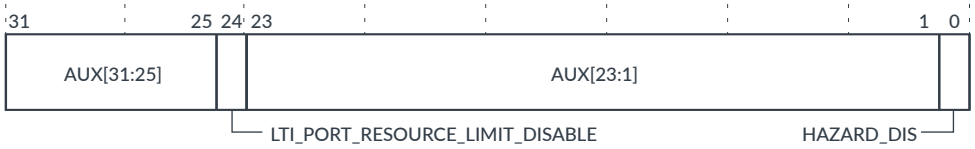


Table 3-98: TBU_CTRL register bit descriptions

Bits	Name	Description
[31:25]	AUX[31:25]	Reserved. Do not change these bits unless Arm advises you to. If writing other bits of this register, ensure that you write the current value for these bits to them, for example, by performing a read-modify-write sequence.
[24]	LTI_PORT_RESOURCE_LIMIT_DISABLE	This bit is used only when more than one LTI port is configured. An ACE-Lite TBU contains only one LTI port internally. **0** When 0, the 3.16.2 TBU_LTI_PORT_RESOURCE_LIMIT register on page 250 is used to control how the LTI ports use translation slots and DTI credits. **1** When 1, all LTI ports are permitted to use the maximum resource. To avoid deadlock and starvation avoidance, some slots and DTI credits are reserved for each LTI port regardless of the value of this register field and the 3.16.2 TBU_LTI_PORT_RESOURCE_LIMIT register on page 250. Therefore, if multiple LTI ports are present, it is not possible for any individual port to use all the slots or DTI credits.
[23:1]	AUX[23:1]	Reserved. Do not change these bits unless Arm advises you to. If writing other bits of this register, ensure that you write the current value for these bits to them, for example, by performing a read-modify-write sequence.
[0]	HAZARD_DIS	**0** When this bit is clear, and multiple outstanding transactions access the same page, the TBU sends a single translation request and uses that for all the affected transactions. **1** When this bit is set, disables hazarding between translation requests from transactions in the same page. Post reset, this bit can be set to 1 once, but cannot be cleared again without a reset.

3.16.2 TBU_LTI_PORT_RESOURCE_LIMIT register

The TBU_LTI_PORT_RESOURCE_LIMIT register limits the resource usage for each LTI port.

Non-secure access to TBU_LTI_PORT_RESOURCE_LIMIT when TBU_SCR.NS_UARCH = 0 is RAZ/WI. See 3.16.3 TBU_SCR register on page 253. Each 4-bit field of this register indicates the resource usage limit fraction for the LTI port number that is indicated.

If the current usage is greater than or equal to the specified fraction of total resource, an LTI port is not permitted to use extra resource.

This allocation affects the resources as follows:

- Translation slots, the total number of which the `TBUCFG_XLATE_SLOTS` parameter provides, that transactions from that LTI port can occupy. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
- DTI translation tokens that the TCU returns in the CONDIS_ACK message. A transaction is considered to use a DTI translation credit from the point that it is accepted into a translation slot until it is known that it no longer requires a DTI request, other than a retry. This includes no longer requiring a translation because it has already performed one.


Note

This is different from occupancy of a translation slot because a translation that has received a cache hit, or otherwise determines that it will not require a DTI translation, is not counted in this fraction. Similarly, a transaction that has received a DTI response, but has not returned the LR is no longer considered to use a DTI credit.

The usage of the port of each of the controlled resources is tracked separately because their usage varies separately, but the limit applies equally to all of them.

The register value expresses the number of sixteenths of the available resource that can be used. Therefore, the values encode to the fractions that the following table shows.

Table 3-99: Value encodings

Register value	Fraction
`4'b0000`	0
`4'b0001`	1/16
`4'b0010`	1/8
`4'b0011`	3/16
`4'b0100`	1/4
`4'b0101`	5/16
`4'b0110`	3/8
`4'b0111`	7/16
`4'b1000`	1/2
`4'b1001`	9/16
`4'b1010`	5/8
`4'b1011`	11/16
`4'b1100`	3/4
`4'b1101`	13/16
`4'b1110`	7/8
`4'b1111`	15/16

The greater-than-or-equal-to constraint permits a fractional credit to be used when the register fraction value multiplied by the total resource available is not a whole number.

If the sum of the allocated resources is more than 1, then a port might not achieve its maximum allocated resources. Resources are allocated on a first case first served basis for that LTI port.

To guarantee freedom from starvation and deadlock, the TLBU must receive at least (2 × NUM_LTI_PORTS) DTI translation request tokens.

The minimum value of tokens that is required is considered to be part of the usage fraction when in use, but those 2 credits are not available to any other port when not in use. Two credits must be reserved for each port to use. The reserved credits are included in the computation of whether extra resource can be used.

If the minimum allocation, 2, is greater than the fractional allocation, the limit is 2.

Combining the fractional maximum and per-port reservation requirements means that the maximum number of translation slots that transactions can occupy from a given port is:

min(max(2,(TBUCFG_XLATE_SLOTS*<lti_port_resource_limit>)), (TBUCFG_XLATE_SLOTS-(2*(NUM_LTI_PORTS-1))))


Note

If the TBUCFG.LTI_PORT_RESOURCE_LIMIT_DISABLE register field is set to 1, this register value does not limit the usage per port. Instead, only the availability of resource once other reservations of ports are considered limits usage.

When only one port is present, this register is RAZ/WI but is treated internally as `4'hF`. This register has no effect on behavior because its effective value permits the single port to use all the available credits.

Configurations

This register is available in all configurations.

Attributes

The TBU_CTRL register attributes are as follows:

Width

32-bit

Functional group

3.4.9 TBU microarchitectural registers summary on page 147.

Address offset

`0x08E04`

Type

RW

Reset value

```
4'hF
```

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-68: TBU_LTI_PORT_RESOURCE_LIMIT register bit assignments

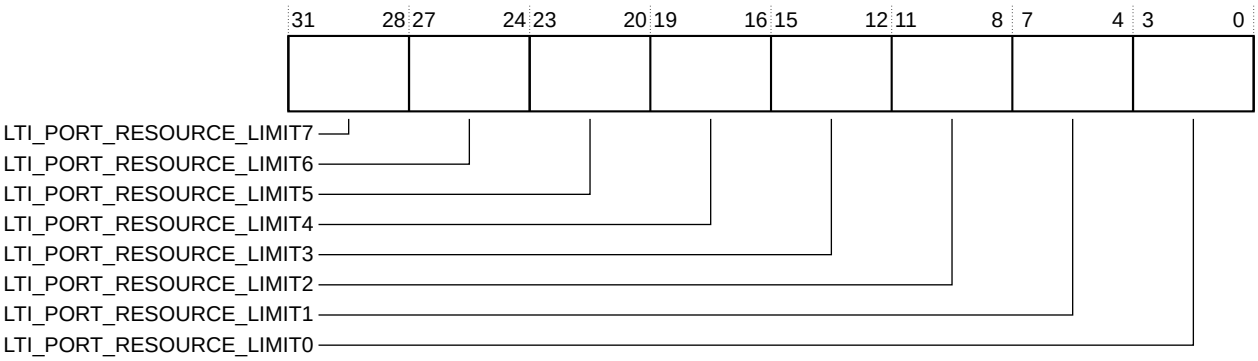


Table 3-100: TBU_LTI_PORT_RESOURCE_LIMIT bit descriptions

Bits	Name	Description
[31:28]	LTI_PORT_RESOURCE_LIMIT7	Resource limit for LTI Port 7
[27:24]	LTI_PORT_RESOURCE_LIMIT6	Resource limit for LTI Port 6
[23:20]	LTI_PORT_RESOURCE_LIMIT5	Resource limit for LTI Port 5
[19:16]	LTI_PORT_RESOURCE_LIMIT4	Resource limit for LTI Port 4
[15:12]	LTI_PORT_RESOURCE_LIMIT3	Resource limit for LTI Port 3
[11:8]	LTI_PORT_RESOURCE_LIMIT2	Resource limit for LTI Port 2
[7:4]	LTI_PORT_RESOURCE_LIMIT1	Resource limit for LTI Port 1
[3:0]	LTI_PORT_RESOURCE_LIMIT0	Resource limit for LTI Port 0

3.16.3 TBU_SCR register

The TBU Secure Control register, TBU_SCR, enables you to control Non-secure and realm register access.

The TBU_SCR register also affects 3.4.10 TBU system discovery registers summary on page 148 in the same way.

Configurations

This register is available in all configurations.

Attributes

The TBU_SCR register attributes are as follows:

Width

32-bit

Functional group

3.4.9 TBU microarchitectural registers summary on page 147.

Address offset

`0x08E18`

Type

RW

Reset value

sec_override signal. See A.2.9 TBU tie-off signals on page 331.

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-69: TBU_SCR register bit assignments

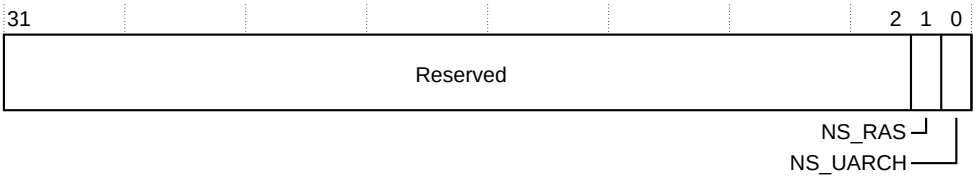


Table 3-101: TBU_SCR register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	NS_RAS	Non-secure and realm register access that is permitted for RAS registers. When this bit is 0, Non-secure writes to the following register addresses are ignored, and Non-secure reads return zero: `0x1_0000-0x1_FFFC` The sec_override input sets the reset value of this signal. See A.2.9 TBU tie-off signals on page 331.

Bits	Name	Description
[0]	NS_UARCH	Non-secure register and realm access that is permitted for the following registers: • 3.16.1 TBU_CTRL register on page 249 • 3.16.2 TBU_LTI_PORT_RESOURCE_LIMIT register on page 250 • 3.19.1 ITEN register for the TBU on page 291 • 3.19.3 ITOP register for the TBU on page 293 • 3.19.4 ITIN register for the TBU on page 297 When this bit is 0, Non-secure writes are ignored, and Non-secure reads return zero for the following registers: • 3.16.1 TBU_CTRL register on page 249 • 3.16.2 TBU_LTI_PORT_RESOURCE_LIMIT register on page 250 • 3.19.1 ITEN register for the TBU on page 291 • 3.19.3 ITOP register for the TBU on page 293 • 3.19.4 ITIN register for the TBU on page 297 The TBU_SCR register also affects the 3.16.4 TBU_ROOT_CTRL register on page 255 register if TBU_RCR.S_ROOT_UARCH = 1. See 3.16.5 TBU_RCR register on page 256. The sec_override input sets the reset value of this signal. See A.2.9 TBU tie-off signals on page 331. If Secure translation might be used, we recommend that software does not set this bit.

3.16.4 TBU_ROOT_CTRL register

The TBU_ROOT_CTRL register disables TBU features. In most systems, you can leave the TBU_ROOT_CTRL register unchanged.

The TBU_ROOT_CTRL register is root only by default. It can be made accessible by Secure or Non-secure by setting various combinations of the 3.16.3 TBU_SCR register on page 253 and the 3.16.5 TBU_RCR register on page 256.

Configurations

This register is available in all configurations.

Attributes

The TBU_ROOT_CTRL register attributes are as follows:

Width

32-bit

Functional group

3.4.9 TBU microarchitectural registers summary on page 147.

Address offset

`0x08F00`

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Table 3-102: TBU_ROOT_CTRL bit descriptions

Bits	Name	Description
[31:1]	AUX[31:1]	Reserved. Do not change these bits unless Arm advises you to. If writing other bits of this register, ensure that you write the current value for these bits to them, for example, by performing a read-modify-write sequence.
[0]	PARITY_DIS	Ignore any parity errors from registers with parity bits.

3.16.5 TBU_RCR register

Use the TBU_RCR register to control security for the TBU.

The TBU_RCR register also affects 3.18 TBU system discovery registers on page 267 and 3.19 TBU integration registers on page 290 in the same way.

If the `TBUCFG_LEGACY_TZ_EN` parameter is set to 1 or the legacy_tz_en signal is set to 1, all bits in this register behave as 1 and the TBU_RCR register is RAZ/WI.

Configurations

This register is available in all configurations.

Attributes

The TBU_RCR register attributes are as follows:

Width

32-bit

Functional group

3.4.9 TBU microarchitectural registers summary on page 147.

Address offset

`0x08F10`

Type

RW

Reset value

sec_override. See A.2.9 TBU tie-off signals on page 331.

Bit descriptions

The following figure and table show the bit assignments and descriptions.

Figure 3-70: TBU_RCR register bit assignments

Table 3-103: TBU_RCR register bit descriptions

Bits	Name	Description
[31:3]	-	Reserved
[2]	S_ROOT_UARCH	Non-root register access permitted for the following root-specific microarchitectural registers: • 3.16.2 TBU_LTI_PORT_RESOURCE_LIMIT register on page 250 • 3.19.1 ITEN register for the TBU on page 291 • 3.19.3 ITOP register for the TBU on page 293 • 3.19.4 ITIN register for the TBU on page 297 • 3.16.4 TBU_ROOT_CTRL register on page 255 Reset is 0.
[1]	S_RAS	Secure register access permitted for microarchitectural registers. When this bit is 0, non-root writes to the following register addresses are ignored, and non-root reads return zero: `0x1_0000-0x1_FFFC` Reset is 1.
[0]	S_UARCH	Secure register access permitted for 3.16.1 TBU_CTRL register on page 249. When this bit is 0, non-root writes to TBU_CTRL are ignored, and non-root reads return zero. Reset is 1. **Note:** The 3.18 TBU system discovery registers on page 267 are also affected

3.17 TBU RAS registers

The MMU S3 TBU contains Reliability, Availability, and Serviceability (RAS) registers. The RAS registers implement the RAS Extension registers, single record format.

Non-secure accesses to these registers, when TBU_SCR.NS_RAS = 0, are RAZ/WI.

The RAS registers enable software to monitor the following classes of error:

- Corrected Errors (CEs) in the RAMs that the Main TLB uses
- CEs in the RAMs, that the Write Data Buffer uses

RAS error reporting

When a CE occurs:

A CE is reported in 3.17.3 TBU_ERRSTATUS register on page 260.

If TBU_ERRCTLR.FI is set, an interrupt is raised on ras_fhi. See 2.4.2.7 TBU interrupt interface on page 41.

3.17.1 TBU_ERRFR register

Error Feature Register.

Configurations

This register is available in all configurations.

Attributes

The TBU_ERRFR register attributes are as follows:

Width

32-bit

Functional group

3.4.8 TBU Reliability, Availability, and Serviceability registers summary on page 147.

Address offset

`0x10000`

Type

Root, RO

Reset value

0

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-71: TBU_ERRFR register bit assignments

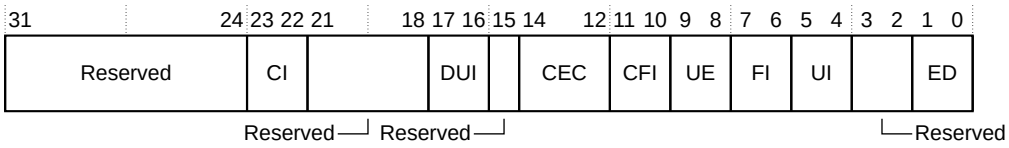


Table 3-104: TBU_ERRFR register bit descriptions

Bits	Name	Description
[31:24]	-	Reserved
[23:22]	CI	Critical Error Interrupt is always enabled. Value is `0b01`.
[21:18]	-	Reserved
[17:16]	DUI	Does not support feature. Value is `0b00`.
[15]	-	Reserved
[14:12]	CEC	Does not implement the standard corrected error counter model. Value is `0b000`.
[11:10]	CFI	Does not support feature. Value is `0b00`.
[9:8]	UE	In-band error signaling feature is not enabled. Value is `0b00`.
[7:6]	FI	Fault handling interrupt is controllable. Value is `0b10`.
[5:4]	UI	Error Recovery Interrupt always enabled for UE. Value is `0b01`.
[3:2]	-	Reserved
[1:0]	ED	Error detection is always enabled. Value is `0b01`.

3.17.2 TBU_ERRCTLR register

To enable fault handling interrupts, use the TBU Error Control register.

Configurations

This register is available in all configurations.

Attributes

The TBU_ERRCTLR register attributes are as follows:

Width

32-bit

Functional group

3.4.8 TBU Reliability, Availability, and Serviceability registers summary on page 147.

Address offset

`0x10008`

Type

Root, RW

Reset value

1

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-72: TBU_ERRCTLR register bit assignments

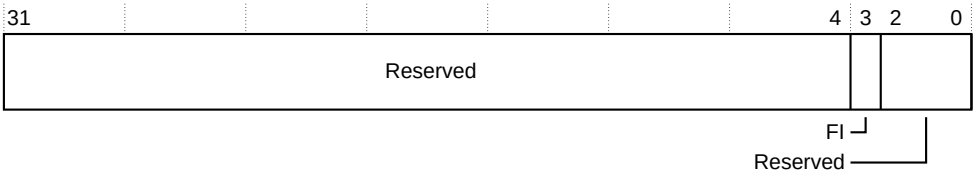


Table 3-105: TBU_ERRCTLR register bit descriptions

Bits	Name	Description
[31:4]	-	Reserved
[3]	FI	Fault handling interrupt enable
[2:0]	-	Reserved

3.17.3 TBU_ERRSTATUS register

To find out whether different types of error have occurred, use the TBU error status register. Certain bits in this register are cleared by writing a 1 to their bit position. These writes are ignored in certain circumstances to avoid race conditions where a new error has occurred that software has not yet observed.

Configurations

This register is available in all configurations.

Attributes

The TBU_ERRSTATUS register attributes are as follows:

Width

32-bit

Functional group

3.4.8 TBU Reliability, Availability, and Serviceability registers summary on page 147.

Address offset

0x10010

Type

Root, RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-73: TBU_ERRSTATUS register bit assignments

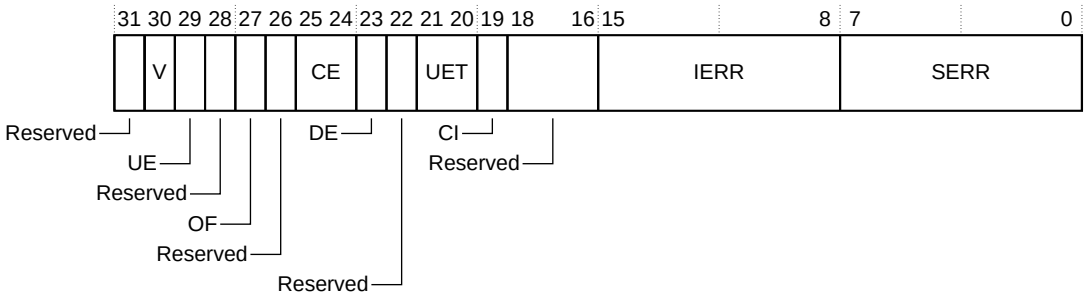


Table 3-106: TBU_ERRSTATUS register bit descriptions

Bits	Name	Description
[31]	-	Reserved
[30]	V	The status Register is valid. The possible values of this bit are as follows: **0** ERRSTATUS is not valid. **1** ERRSTATUS is valid, meaning that at least 1 error has been recorded. This field is read/write-one-to-clear. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130. This bit resets to 0 on a reset.
[29]	UE	Uncorrected errors. The possible values of this bit are: **0** No errors that could not be corrected or deferred **1** At least one error is detected that has not been corrected or deferred This field is read/write-one-to-clear. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130.
[28]	-	Reserved
[27]	OF	Overflow. Multiple errors detected. This bit is set to 1 when: • Any error is received and TBU_ERRSTATUS.V is already set, and not being cleared on the same cycle • Multiple errors are received on the same cycle This field is read/write-one-to-clear. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130.
[26]	-	Reserved

Bits	Name	Description
[25:24]	CE	Correctable Errors: **0b00** No correctable errors recorded **0b10** At least one corrected error recorded Other values are Reserved. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130.
[23]	DE	Deferred errors. The possible values of this bit are as follows: **0** No errors were deferred **1** At least one error was not corrected and deferred This error is raised when the Bus Interface Unit (BIU) sets wpoison. This field is read/write-one-to-clear. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130.
[22]	-	Reserved
[21:20]	UET	Uncorrected Error Type. Describes the state of the component after detecting or consuming an Uncorrected error. The possible values of this field are as follows: **0b00** Uncorrected error, UnContainable error (UC) Writes to this field are ignored.
[19]	CI	Indicates whether a critical error condition has been recorded. The possible values of this bit are as follows: **0** No critical error condition **1** Critical error condition Writes to this field are ignored.
[18:16]	-	Reserved

Bits	Name	Description
[15:8]	IERR	**IMPLEMENTATION DEFINED** error code. The value of this field indicates the source of the error as follows: **32** RRB FF PARITY **31** PRG FF PARITY **30** PMU FF PARITY **29** BIU CPB FF PARITY **28** BIU WTB FF PARITY **27** BIU SIBMIB FF PARITY **26** DIB FF PARITY **25** DCU TLB FF PARITY **24** DCU DCB FF PARITY **23** TOU FF PARITY **22** Reserved **21** Reserved **20** BIU WDB ROBUFF_P **19** BIU WDB ROBUFF_C **18** BIU WDB ROBUFF_D **17** Reserved **16** Reserved **15** TLBU DCU MTLB TAGS **14** TLBU DCU MTLB REPL **13** TLBU DCU MTLB PCNT **12** TLBU DCU MTLB PLIM **11** TLBU TOU HLB_ENTRY RIGHT **10** TLBU TOU HLB_ENTRY LEFT **9** TLBU TOU HLB PTR RIGHT **8** TLBU TOU HLB PTR LEFT **7** Reserved **6** Reserved **5** TLBU TOU DTIQ **4** TLBU TOU UOQ **3** TLBU TOU OGQ **2** TLBU TOU LB **1** TLBU TOU RSP **0** TLBU TOU REQ Writes to this field are ignored.
[7:0]	SERR	Error code. This provides information about the earliest unacknowledged Error. It can contain the following values: **0** No error **2** Single or double error from RAMs that are not MTLB TAGS or DATA **8** Single or double error from MTLB Data **9** Single or double error from MTLB Tags **17** Internal control register FF parity **26** Miscellaneous FF Parity Error All other values are reserved. Writes to this field are ignored.

3.17.4 TBU_ERRGEN register

To test how the system responds when a RAS error occurs, use the Error Generation Register, TBU_ERRGEN.

The field locations are same as for the 3.17.3 TBU_ERRSTATUS register on page 260.

When this register is updated, the 3.17.3 TBU_ERRSTATUS register on page 260 is also updated with the same value, as long as the write data generates a valid error record.

A write to the TBU_ERRGEN register is valid if all the following conditions are met:

- The TBU_ERRGEN.V bit is set.
- At least 1 of the following is true (TBU_ERRGEN.CE is legal if CE == `2'b00` or CE == `2'b10`):
  - The TBU_ERRGEN.UE bit is set and the CE bits have legal values
  - The TBU_ERRGEN.DE bit is set and the CE bits have legal values
  - The TBU_ERRGEN.CE bits are set to `0b10`
- The TBU_ERRGEN.UET bits must be set to `0b00`

The following also apply:

- All other fields can take any legal value.
- Any other write is considered to be invalid and the behavior of the SMMU is **UNPREDICTABLE**.
- If a valid error record is written, then the appropriate interrupt, or interrupts, are also generated.


Tip

This register has identical fields to the 3.17.3 TBU_ERRSTATUS register on page 260.

Configurations

This register is available in all configurations.

Attributes

The TBU_ERRGEN register attributes are as follows:

Width

32-bit

Functional group

3.4.8 TBU Reliability, Availability, and Serviceability registers summary on page 147.

Address offset

`0x10040`

Type

Root, RW

Reset value

0

Bit descriptions

The following figure and table show the bit assignments.

Figure 3-74: TBU_ERRGEN register bit assignments

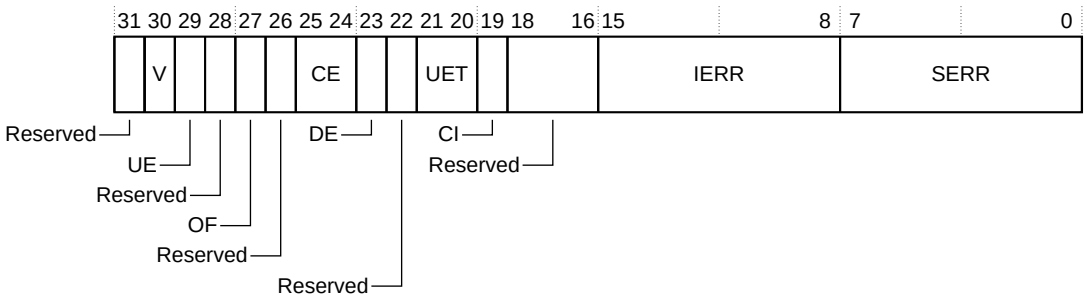


Table 3-107: TBU_ERRGEN register bit descriptions

Bits	Name	Description
[31]	-	Reserved
[30]	V	The status Register is valid. The possible values of this bit are as follows: **0** ERRSTATUS is not valid. **1** ERRSTATUS is valid, meaning that at least 1 error has been recorded. This field is read/write-one-to-clear. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130. This bit resets to 0 on a reset.
[29]	UE	Uncorrected errors. The possible values of this bit are: **0** No errors that could not be corrected or deferred **1** At least one error is detected that has not been corrected or deferred This field is read/write-one-to-clear. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130.
[28]	-	Reserved
[27]	OF	Overflow. Multiple errors detected. This bit is set to 1 when: • Any error is received and TBU_ERRSTATUS.V is already set, and not being cleared on the same cycle • Multiple errors are received on the same cycle This field is read/write-one-to-clear. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130.
[26]	-	Reserved

Bits	Name	Description
[25:24]	CE	Correctable Errors: **0b00** No correctable errors recorded **0b10** At least one corrected error recorded Other values are Reserved. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130.
[23]	DE	Deferred errors. The possible values of this bit are as follows: **0** No errors were deferred **1** At least one error was not corrected and deferred This field is read/write-one-to-clear. Clearing depends on other ERRSTATUS fields. See 3.1 Clearing ERRSTATUS registers on page 130.
[22]	-	Reserved
[21:20]	UET	Uncorrected Error Type. Describes the state of the component after detecting or consuming an Uncorrected error. The possible values of this field are as follows: **0b00** Uncorrected error, UnContainable error (UC) Writes to this field are ignored.
[19]	CI	Indicates whether a critical error condition has been recorded. The possible values of this bit are as follows: **0** No critical error condition **1** Critical error condition Writes to this field are ignored.
[18:16]	-	Reserved

Bits	Name	Description
[15:8]	IERR	**IMPLEMENTATION DEFINED** error code. The value of this field indicates the source of the error as follows: **32** RRB FF PARITY **31** PRG FF PARITY **30** PMU FF PARITY **29** BIU CPB FF PARITY **28** BIU WTB FF PARITY **27** BIU SIBMIB FF PARITY **26** DIB FF PARITY **25** DCU TLB FF PARITY **24** DCU DCB FF PARITY **23** TOU FF PARITY **22** Reserved **21** Reserved **20** BIU WDB ROBUFF_P **19** BIU WDB ROBUFF_C **18** BIU WDB ROBUFF_D **17** Reserved **16** Reserved **15** TLBU DCU MTLB TAGS **14** TLBU DCU MTLB REPL **13** TLBU DCU MTLB PCNT **12** TLBU DCU MTLB PLIM **11** TLBU TOU HLB_ENTRY RIGHT **10** TLBU TOU HLB_ENTRY LEFT **9** TLBU TOU HLB PTR RIGHT **8** TLBU TOU HLB PTR LEFT **7** Reserved **6** Reserved **5** TLBU TOU DTIQ **4** TLBU TOU UOQ **3** TLBU TOU OGQ **2** TLBU TOU LB **1** TLBU TOU RSP **0** TLBU TOU REQ Writes to this field are ignored.
[7:0]	SERR	Error code. The error code provides information about the earliest unacknowledged Error and can contain the following values: **0** No error **2** Single or double error from RAMs that are not MTLB TAGS or DATA **8** Single or double error from MTLB Data **9** Single or double error from MTLB Tags **17** Internal control register FF parity **26** Miscellaneous FF Parity Error All other values are reserved. Writes to this field are ignored.

3.18 TBU system discovery registers

The MMU S3 TBU contains system discovery registers.

3.18.1 TBU_SYSDISC0 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC0 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09000`

Type

RO

Reset value

`TBUCFG_MTLB_DEPTH`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-75: TBU_SYSDISC0 register bit assignments

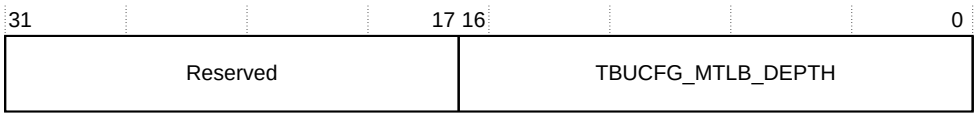


Table 3-108: TBU_SYSDISC0 register bit descriptions

Bits	Name	Description
[31:17]	-	Reserved

Bits	Name	Description

[16:0]	TBUCFG_MTLB_DEPTH	The read data reflects the chosen parameter value, for example:
		`17'h0_0008 : 8`
		
		....
		
		`17'h1_0000 : 65536`

3.18.2 TBU_SYSDISC1 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC1 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09004`

Type

RO

Reset value

`TBUCFG_UTLB_DEPTH`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-76: TBU_SYSDISC1 register bit assignments

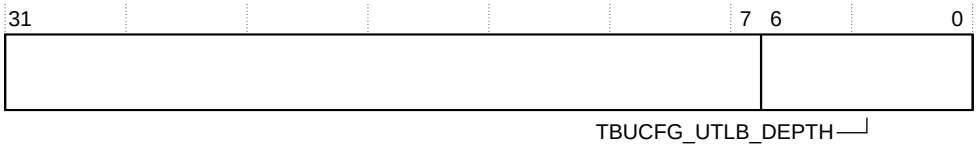


Table 3-109: TBU_SYSDISC1 register bit descriptions

Bits	Name	Description
[31:7]	-	Reserved

Bits	Name	Description

[6:0]	TBUCFG_UTLB_DEPTH	The read data reflects the chosen parameter value, for example:
		`7'h04 : 4`
		
		....
		
		`7'h40 : 64`

3.18.3 TBU_SYSDISC2 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC2 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09008`

Type

RO

Reset value

`TBUCFG_MTLB_WAYS`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-77: TBU_SYSDISC2 register bit assignments

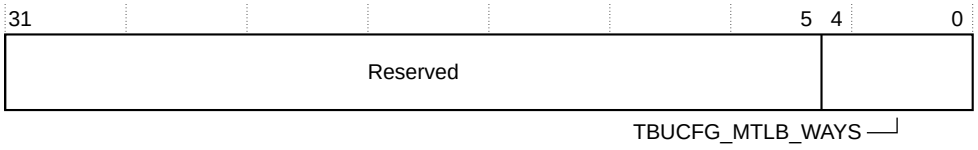


Table 3-110: TBU_SYSDISC2 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved

Bits	Name	Description

[4:0]	TBUCFG_MTLB_WAYS	The read data reflects the chosen parameter value, for example:
		`5'h04 : 4`
		....
		`5'h10 : 16`

3.18.4 TBU_SYSDISC3 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC3 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x0900C`

Type

RO

Reset value

`TBUCFG_MTLB_BANKS`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-78: TBU_SYSDISC3 register bit assignments

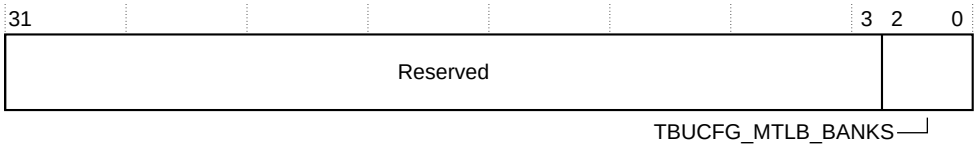


Table 3-111: TBU_SYSDISC3 register bit descriptions

Bits	Name	Description
[31:3]	-	Reserved

[2:0]	TBUCFG_MTLB_BANKS	The read data reflects the chosen parameter value, for example:
		`3'b001 : 1`
		
		....
		
		`3'b100 : 4`

3.18.5 TBU_SYSDISC4 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC4 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09010`

Type

RO

Reset value

`TBUCFG_XLATE_SLOTS`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-79: TBU_SYSDISC4 register bit assignments

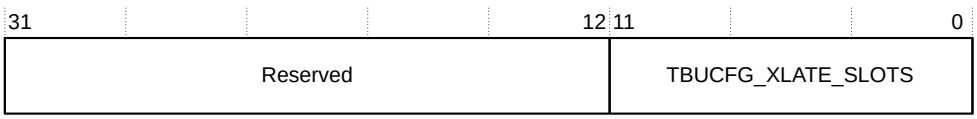


Table 3-112: TBU_SYSDISC4 register bit descriptions

Bits	Name	Description
[31:12]	-	Reserved

Bits	Name	Description

[11:0]	TBUCFG_XLATE_SLOTS	The read data reflects the chosen parameter value, for example:
		`11'h0004 : 4`
		
		....
		
		`11'h400 : 1024`

3.18.6 TBU_SYSDISC5 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC5 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09014`

Type

RO

Reset value

`TBUCFG_PMU_COUNTERS`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-80: TBU_SYSDISC5 register bit assignments

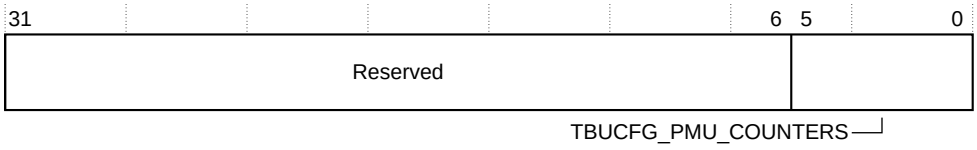


Table 3-113: TBU_SYSDISC5 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved

Bits	Name	Description

[5:0]	TBUCFG_PMU_COUNTERS	The read data reflects the chosen parameter value, for example:
		`6'h04 : 4`
		
		....
		
		`6'h20 : 32`

3.18.7 TBU_SYSDISC6 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC6 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09018`

Type

RO

Reset value

`TBUCFG_SID_WIDTH`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-81: TBU_SYSDISC6 register bit assignments

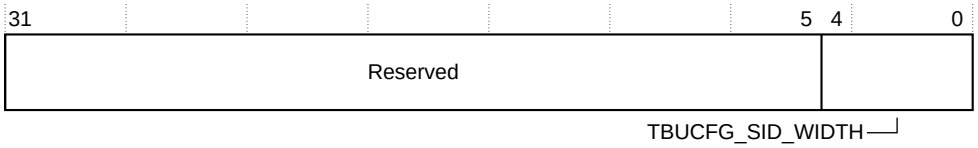


Table 3-114: TBU_SYSDISC6 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved

Bits	Name	Description

[4:0]	TBUCFG_SID_WIDTH	The read data reflects the chosen parameter value, for example:
		`5'h08 : 8`
		
		....
		
		`5'h18 : 24`

3.18.8 TBU_SYSDISC7 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC7 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x0901C`

Type

RO

Reset value

`TBUCFG_SSID_WIDTH`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-82: TBU_SYSDISC7 register bit assignments

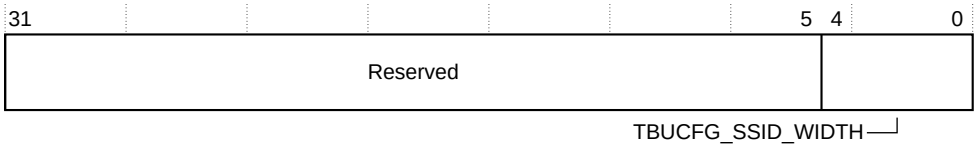


Table 3-115: TBU_SYSDISC7 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved

Bits	Name	Description

[4:0]	TBUCFG_SSID_WIDTH	The read data reflects the chosen parameter value, for example:
		`5'h01 : 1`
		
		....
		
		`5'h14 : 20`

3.18.9 TBU_SYSDISC8 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC8 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09020`

Type

RO

Reset value

`TBUCFG_DIRECT_IDX`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-83: TBU_SYSDISC8 register bit assignments

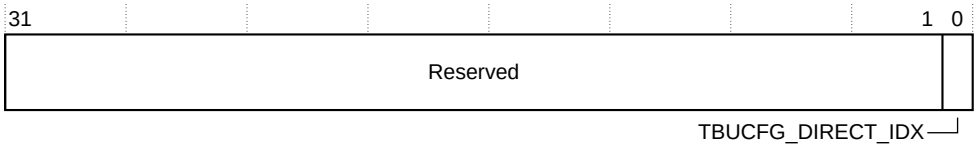


Table 3-116: TBU_SYSDISC8 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved

Bits	Name	Description

[0]	TBUCFG_DIRECT_IDX	The read data reflects the chosen parameter value, for example:
		`1'b0 : 0`
		
		....
		
		`1'b1 : 1`

3.18.10 TBU_SYSDISC9 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC9 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09024`

Type

RO

Reset value

`TBUCFG_MTLB_PARTS`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-84: TBU_SYSDISC9 register bit assignments

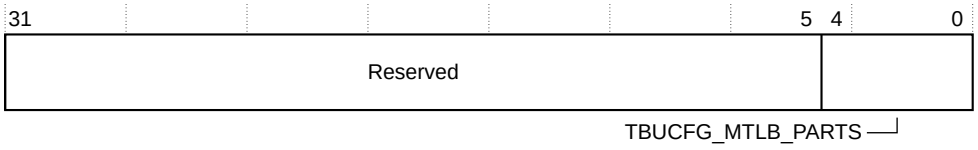


Table 3-117: TBU_SYSDISC9 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved

Bits	Name	Description

[4:0]	TBUCFG_MTLB_PARTS	The read data reflects the chosen parameter value, for example:
		`5'h00 : 0`
		....
		`5'h10 : 16`

3.18.11 TBU_SYSDISC10 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC10 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09028`

Type

RO

Reset value

`TBUCFG_LTI_OG_WIDTH`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-85: TBU_SYSDISC10 register bit assignments

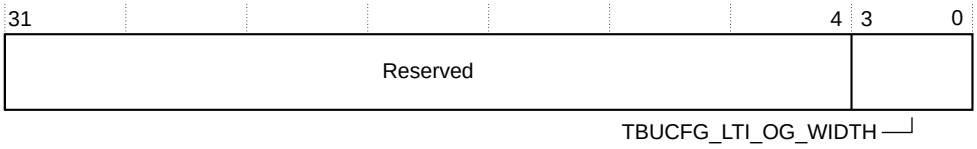


Table 3-118: TBU_SYSDISC10 register bit descriptions

Bits	Name	Description
[31:4]	-	Reserved

Bits	Name	Description

[3:0]	TBUCFG_LTI_OG_WIDTH	The read data reflects the chosen parameter value, for example:
		`3'h00 : 0`
		
		....
		
		`3'h20 : 05`

3.18.12 TBU_SYSDISC11 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC11 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x0902C`

Type

RO

Reset value

`TBUCFG_PARTID_WIDTH`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-86: TBU_SYSDISC11 register bit assignments

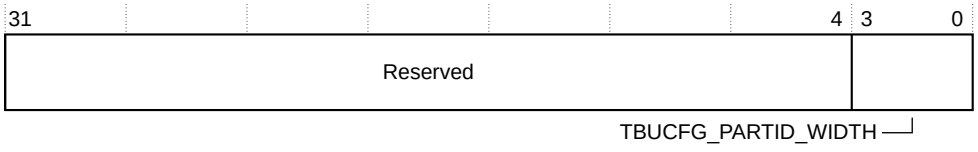


Table 3-119: TBU_SYSDISC11 register bit descriptions

Bits	Name	Description
[31:4]	-	Reserved

Bits	Name	Description

[3:0]	TBUCFG_PARTID_WIDTH	The read data reflects the chosen parameter value, for example:
		`4'b0001 : 1`
		
		....
		
		`4'b1010 : 10`

3.18.13 TBU_SYSDISC12 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC12 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09030`

Type

RO

Reset value

`TBUCFG_HZRD_ENTRIES`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-87: TBU_SYSDISC12 register bit assignments

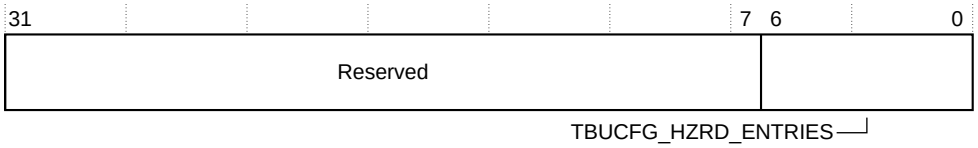


Table 3-120: TBU_SYSDISC12 register bit descriptions

Bits	Name	Description
[31:7]	-	Reserved

Bits	Name	Description

[6:0]	TBUCFG_HZRD_ENTRIES	The read data reflects the chosen parameter value, for example:
		`7'h00 : 0`
		....
		`7'h40 : 64`

3.18.14 TBU_SYSDISC13 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC13 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09034`

Type

RO

Reset value

`TBUCFG_SLOTRAM_TYPE`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-88: TBU_SYSDISC13 register bit assignments

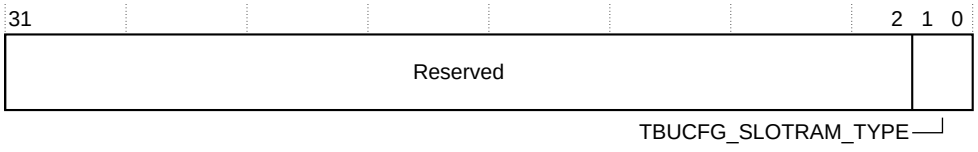


Table 3-121: TBU_SYSDISC13 register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved

Bits	Name	Description

[1:0]	TBUCFG_SLOTRAM_TYPE	The read data reflects the chosen parameter value, for example:
		`2'b00 : 0`
		
		....
		
		`2'b01 : 1`

3.18.15 TBU_SYSDISC14 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC14 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09038`

Type

RO

Reset value

`TBUCFG_CACHERAM_TYPE`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-89: TBU_SYSDISC14 register bit assignments

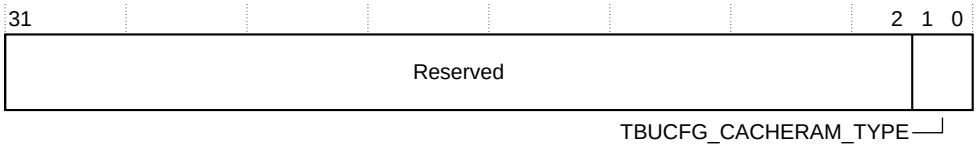


Table 3-122: TBU_SYSDISC14 register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved

Bits	Name	Description

[1:0]	TBUCFG_CACHERAM_TYPE	The read data reflects the chosen parameter value, for example:
		`2'b00 : 0`
		
		....
		
		`2'b01 : 1`

3.18.16 TBU_SYSDISC15 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC15 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x0903C`

Type

RO

Reset value

`TBUCFG_MTLB_LKP_SLOTS`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-90: TBU_SYSDISC15 register bit assignments

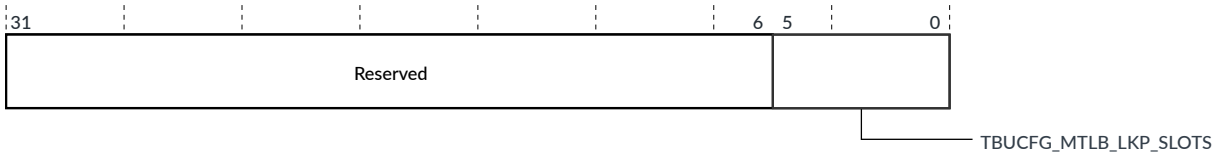


Table 3-123: TBU_SYSDISC15 register bit descriptions

Bits	Name	Description
[31:6]	-	Reserved
[5:0]	TBUCFG_MTLB_LKP_SLOTS	The read data reflects the chosen parameter value

3.18.17 TBU_SYSDISC16 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC16 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09040`

Type

RO

Reset value

`TBUCFG_LA_HNDSHK_MODE`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-91: TBU_SYSDISC16 register bit assignments

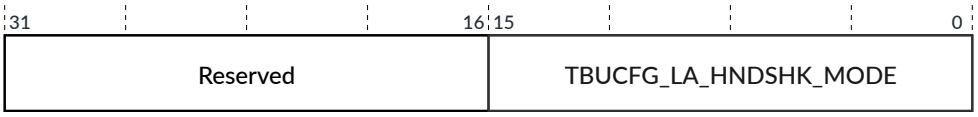


Table 3-124: TBU_SYSDISC16 register bit descriptions

Bits	Name	Description
[31:16]	-	Reserved

Bits	Name	Description
[15:0]	TBUCFG_LA_HNDSHK_MODE	The read data reflects the chosen parameter value. {TBUCFG_LA_HNDSHK_MODE7, ... TBUCFG_LA_HNDSHK_MODE0}

3.18.18 TBU_SYSDISC17 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC17 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

0x09044

Type

RO

Reset value

TBUCFG_LR_HNDSHK_MODE. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-92: TBU_SYSDISC17 register bit assignments

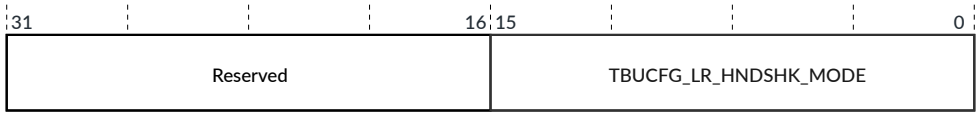


Table 3-125: TBU_SYSDISC17 register bit descriptions

Bits	Name	Description
[31:16]	-	Reserved

Bits	Name	Description
[15:0]	TBUCFG_LR_HNDSHK_MODE	The read data reflects the chosen parameter value. {TBUCFG_LR_HNDSHK_MODE7, ... TBUCFG_LR_HNDSHK_MODE0}

3.18.19 TBU_SYSDISC18 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC18 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

0x09048

Type

RO

Reset value

TBUCFG_LEGACY_TZ_EN. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-93: TBU_SYSDISC18 register bit assignments

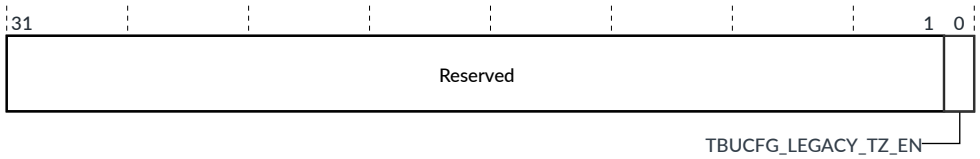


Table 3-126: TBU_SYSDISC18 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	TBUCFG_LEGACY_TZ_EN	The read data reflects the chosen parameter value

3.18.20 TBU_SYSDISC19 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC19 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x0904C`

Type

RO

Reset value

`TCUCFG_USE_ELA_DEBUG`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-94: TBU_SYSDISC19 register bit assignments

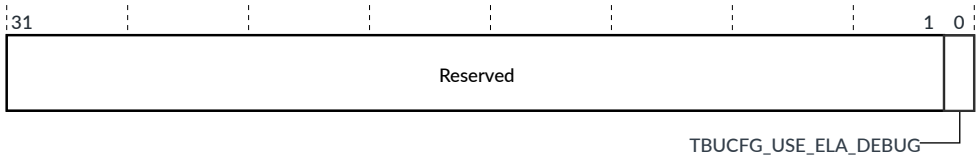


Table 3-127: TBU_SYSDISC19 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	TBUCFG_USE_ELA_DEBUG	The read data reflects the chosen parameter value

3.18.21 TBU_SYSDISC20 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC20 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

0x09050

Type

RO

Reset value

TBUCFG_PARITY_ON. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-95: TBU_SYSDISC20 register bit assignments

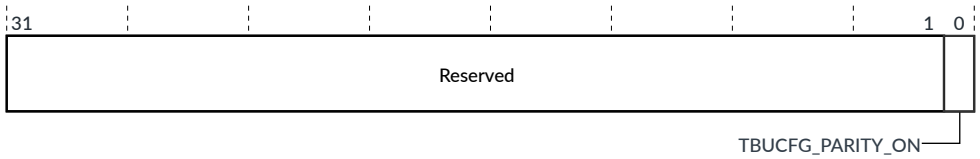


Table 3-128: TBU_SYSDISC20 register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	TBUCFG_PARITY_ON	The read data reflects the chosen parameter value

3.18.22 TBU_SYSDISC21 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC21 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

0x09054

Type

RO

Reset value

TBUCFG_MTLB_UPD_SLOTS. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-96: TBU_SYSDISC21 register bit assignments

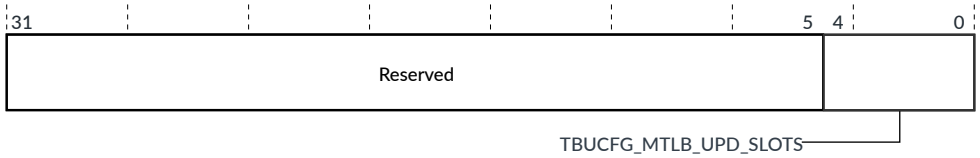


Table 3-129: TBU_SYSDISC21 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved
[4:0]	TBUCFG_MTLB_UPD_SLOTS	The read data reflects the chosen parameter value

3.18.23 TBU_SYSDISC22 system discovery register

The TBU system discovery registers discover components in the system.

Configurations

This register is available in all configurations.

Attributes

The TBU_SYSDISC22 register attributes are as follows:

Width

32-bit

Functional group

3.4.10 TBU system discovery registers summary on page 148.

Address offset

`0x09058`

Type

RO

Reset value

`TBUCFG_MECID_WIDTH`. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-97: TBU_SYSDISC22 register bit assignments

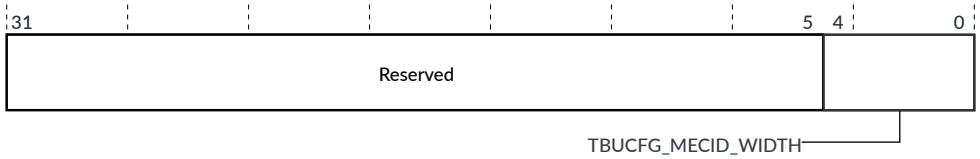


Table 3-130: TBU_SYSDISC22 register bit descriptions

Bits	Name	Description
[31:5]	-	Reserved
[4:0]	TBUCFG_MECID_WIDTH	The read data reflects the chosen parameter value

3.19 TBU integration registers

The MMU S3 TBU contains integration registers.

3.19.1 ITEN register for the TBU

Integration mode enable register for the TBU. When integration mode is enabled, the values of certain TBU input pins are made visible in the ITIN register of the TBU. The values that are written to the ITOP register of the TBU control the values of certain TBU output pins. This mechanism helps system integrators to integrate the SMMU into the system and perform basic connectivity checks.

See also:

- 3.19.2 ITEN_ET register for the TBU on page 292
- 3.19.4 ITIN register for the TBU on page 297
- 3.19.3 ITOP register for the TBU on page 293

Configurations

This register is available in all configurations.

Attributes

The ITEN register attributes are as follows:

Width

32-bit

Functional group

3.4.11 TBU integration registers summary on page 149.

Address offset

`0x08E20`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-98: ITEN register bit assignments

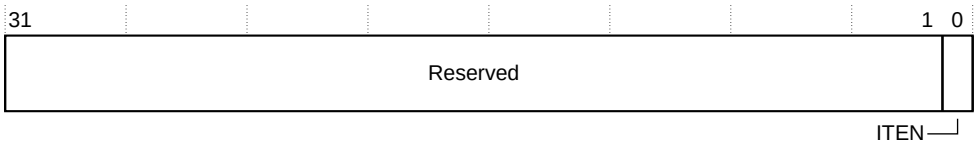


Table 3-131: ITEN register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved
[0]	ITEN	**0** Disable integration mode **1** Enable integration mode

3.19.2 ITEN_ET register for the TBU

Edge triggered integration mode register for the TBU. When integration mode is enabled, the values of certain TBU input pins are made visible in the ITIN register for the TBU. When integration mode is enabled and triggered, the values that are written to the ITOP register for the TBU are driven to certain TBU output pins for a single cycle before being cleared. The mechanism helps system integrators to integrate the SMMU into the system and perform basic connectivity checks for single cycle interrupts.

See also:

- 3.19.1 ITEN register for the TBU on page 291
- 3.19.4 ITIN register for the TBU on page 297
- 3.19.3 ITOP register for the TBU on page 293

Configurations

This register is available in all configurations.

Attributes

The ITEN_ET register attributes are as follows:

Width

32-bit

Functional group

3.4.11 TBU integration registers summary on page 149.

Address offset

`0x08E2C`

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-99: ITEN_ET register bit assignments

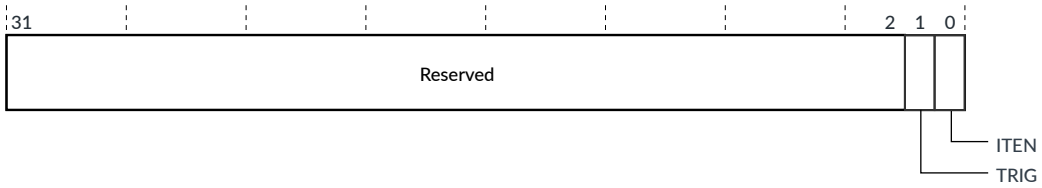


Table 3-132: ITEN register bit descriptions

Bits	Name	Description
[31:2]	-	Reserved
[1]	TRIG	Write-Only A write of 1 to this field, when integration mode is enabled, triggers the values of ITOP to be present on outputs for a single cycle.
[0]	ITEN	Read-Write **0** Disable integration mode **1** Enable integration mode When integration mode is disabled, the value of the 3.19.3 ITOP register for the TBU on page 293 is cleared and becomes read only. **Note:** ITEN functionality takes priority over ITEN_ET functionality.

3.19.3 ITOP register for the TBU

Integration output register for the TBU.

Configurations

This register is available in all configurations.

Attributes

The ITOP_TBU register attributes are as follows:

Width

32-bit

Functional group

3.4.11 TBU integration registers summary on page 149.

Address offset

0x08E24

Type

RW

Reset value

0

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-100: ITOP_TBU register bit assignments

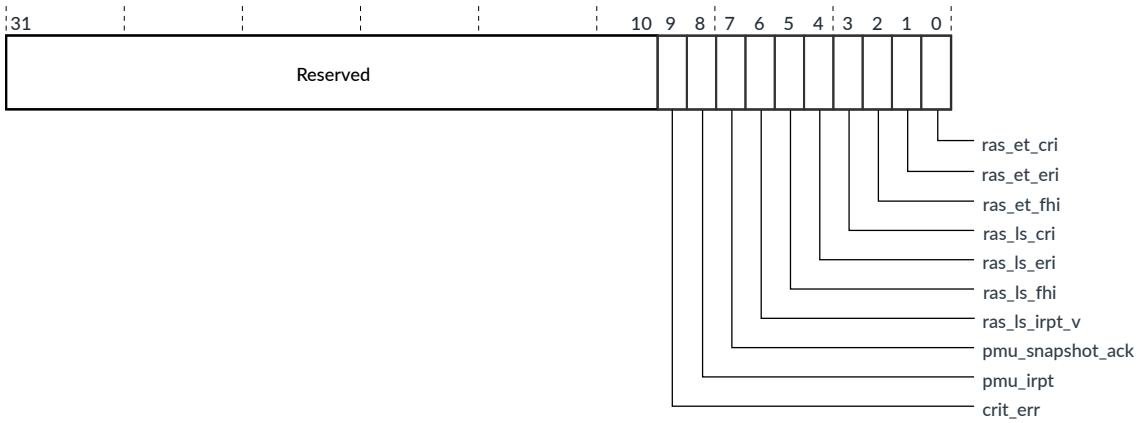


Table 3-133: ITOP_TBU register bit descriptions

Bits	Name	Description
[31:10]	-	Reserved, Should-Be-Zero (SBZ)
[9]	crit_err	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336

Bits	Name	Description
[8]	pmu_irpt	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336
[7]	pmu_snapshot_ack	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336
[6]	ras_ls_irpt_v	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336

Bits	Name	Description
[5]	ras_ls_fhi	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336
[4]	ras_ls_eri	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336
[3]	ras_ls_cri	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336

Bits	Name	Description
[2]	ras_et_fhi	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336
[1]	ras_et_eri	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336
[0]	ras_et_cri	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field value can be 0 or 1. • When ITEN.ITEN == 1 this value is driven to the output specified. • When ITEN.ITEN == 0 and ITEN_ET.ITEN == 1 and the value of 1 is written to ITEN_ET.TRIG, this value is driven to the output specified for a single cycle. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.2.8 TBU interrupt signals on page 330 • A.3.2 PMU Snapshot signals on page 336

3.19.4 ITIN register for the TBU

Integration input register for the TBU.

See A.3.2 PMU Snapshot signals on page 336.

Configurations

This register is available in all configurations.

Attributes

The ITIN_TBU register attributes are as follows:

Width

32-bit

Functional group

3.4.11 TBU integration registers summary on page 149.

Address offset

`0x08E28`

Type

RO

Reset value

0

Bit descriptions

The following figure and table show the register bit assignments and descriptions.

Figure 3-101: ITIN_TBU register bit assignments

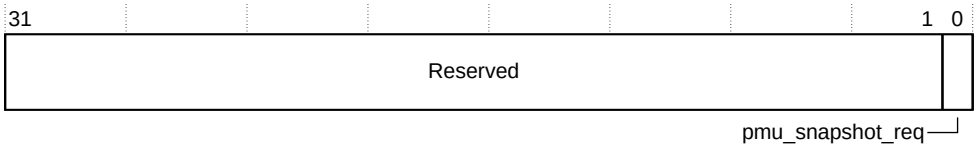


Table 3-134: ITIN_TBU register bit descriptions

Bits	Name	Description
[31:1]	-	Reserved, Should-Be-Zero (SBZ)

Bits	Name	Description
[0]	pmu_snapshot_req	• When ITEN.ITEN == 0 and ITEN_ET.ITEN == 0, this field is RO and set to 0. • When ITEN.ITEN == 1 or ITEN_ET.ITEN == 1, this field reflects the value of the input signal specified. See: • 3.19.1 ITEN register for the TBU on page 291 • 3.19.2 ITEN_ET register for the TBU on page 292 • A.3.2 PMU Snapshot signals on page 336

Appendix A Signal descriptions

MMU S3 uses various external signals for the Translation Control Unit (TCU), the Translation Buffer Unit (TBU), and the Distributed Translation Interface (DTI).

MMU S3 external signal names use the following convention:

_s Secure signal, for example event_q_irpt_s
_ns Non-secure signal, for example event_q_irpt_ns
_r Realm signal, for example event_q_irpt_r

For more information about Secure, Non-secure, and Realm signals, see the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

A.1 TCU signals

MMU S3 contains TCU signals.

A.1.1 TCU clock and reset signals

The TCU uses a single set of standard clock and reset signals.

Signal definitions

Table A-1: Clock and reset signals

Signal	Direction	Description
clk	Input	Global clock. Width is 1-bit.
resetn	Input	Global reset. Width is 1-bit.

A.1.2 TCU QTW/DVM interface signals

The TCU QTW/DVM interface signals are based on the AMBA ACE5-Lite+DVM signals.

Signal definitions

Table A-2: TCU QTW/DVM interface signals

Signal	Direction	Description
acaddr_qtw	Input	Snoop address. Width is 52-bit.
acready_qtw	Output	Snoop address ready. Width is 1-bit.
acvalid_qtw	Input	Snoop address valid. Width is 1-bit.
acwakeup_qtw	Input	Snoop wakeup. Width is 1-bit.
acvmidext_qtw	Input	Snoop Extended Virtual Machine IDentifier (VMID). Width is 4-bit.

Signal	Direction	Description
arid_qtw	Output	Read address ID The width of arid_qtw is equal to `QTW_ID_WIDTH`. `QTW_ID_WIDTH` is calculated as follows: ceil($\log_2$(max(`TCUCFG_DGW_SLOTS`, `TCUCFG_PTW_SLOTS`, 8))) + 3
araddr_qtw	Output	Read address. Width is 52-bit.
arburst_qtw	Output	Burst type. Width is 2-bit.
arcache_qtw	Output	Memory type. Width is 4-bit.
ardomain_qtw	Output	Shareability domain. Width is 2-bit.
arlen_qtw	Output	Burst length. Width is 8-bit.
arlock_qtw	Output	Lock type. Width is 1-bit.
arprot_qtw	Output	Protection type. Width is 3-bit.
arqos_qtw	Output	QoS identifier. Width is 4-bit.
arready_qtw	Input	Read address ready. Width is 1-bit.
arsize_qtw	Output	Transfer size. Width is 3-bit.
arsnoop_qtw	Output	Transaction type. Width is 4-bit.
arvalid_qtw	Output	Read address valid. Width is 1-bit.
aridunq_qtw	Output	Read address channel unique ID indicator. Width is 1-bit.
armpam_qtw	Output	Read address channel MPAM information. Width is (`TCUCFG_LEGACY_TZ_EN`==1) ? 14 : 15 The MPAM signal position is aligned with MPAM_12_1
arnse_qtw	Output	Along with ARPROT[1] indicates whether the physical address space is Secure, Non-Secure, Root, or Realm. Width is 1 bit
arpbha_qtw	Output	Page-based hardware attributes for read requests. Width is 4-bit.
armecid_qtw	Output	MECID for requests on the read request channel Width is (`TCUCFG_MECID_WIDTH` == 0) ? 1 : 16
awid_qtw	Output	Write address ID Width is `QTW_ID_WIDTH`-bit. `QTW_ID_WIDTH` is calculated as follows: ceil($\log_2$(max(`TCUCFG_DGW_SLOTS`, `TCUCFG_PTW_SLOTS`, 8))) + 3
awaddr_qtw	Output	Write address. Width is 52-bit.
awburst_qtw	Output	Burst type. Width is 2-bit.
awcache_qtw	Output	Memory type. Width is 4-bit.

Signal	Direction	Description
awdomain_qtw	Output	Shareability domain. Width is 2-bit.
awlen_qtw	Output	Burst length. Width is 8-bit.
awlock_qtw	Output	Lock type. Width is 1-bit.
awprot_qtw	Output	Protection type. Width is 3-bit.
awqos_qtw	Output	QoS identifier. Width is 4-bit.
awready_qtw	Input	Write address ready. Width is 1-bit.
awsize_qtw	Output	Burst size. Width is 3-bit.
awsnoop_qtw	Output	Transaction type. Width is 4-bit.
awvalid_qtw	Output	Write address valid. Width is 1-bit.
awatop_qtw	Output	Atomic operation. Width is 6-bit.
awidunq_qtw	Output	Write address channel unique ID indicator. Width is 1-bit.
awmpam_qtw	Output	Write address channel MPAM information. Width is (`TCUCFG_LEGACY_TZ_EN` == 1) ? 14 : 15 The MPAM signal position is aligned with MPAM_12_1
awnse_qtw	Output	Along with ARPROT[1], indicates whether the physical address space is Secure, Non-Secure, Root, or Realm. Width is 1-bit.
awpbha_qtw	Output	Page-based hardware attributes for write requests. Width is 4-bit.
awmecid_qtw	Output	MECID for requests on the write request channel. Width is (`TCUCFG_MECID_WIDTH` == 0) ? 1 : 16
crready_qtw	Input	Snoop response ready. Width is 1-bit.
crvalid_qtw	Output	Snoop response valid. Width is 1-bit.
rdata_qtw	Input	Read data. The width of rdata_qtw is equal to `TCUCFG_QTW_DATA_WIDTH` See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
rid_qtw	Input	Read data ID Width is `QTW_ID_WIDTH`-bit. `QTW_ID_WIDTH` is calculated as follows: ceil($\log_2$ (max(`TCUCFG_DGW_SLOTS`, `TCUCFG_PTW_SLOTS`, 8))) + 3 See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
ridunq_qtw	Input	Read data channel unique ID indicator, active HIGH. Width is 1-bit.
rlast_qtw	Input	Read last. Width is 1-bit.
rpoison_qtw	Input	Read poison input to the TCU. Width is `TCUCFG_QTW_DATA_WIDTH` / 64
rready_qtw	Output	Read ready. Width is 1-bit.
rresp_qtw	Input	Read response. Width is 2-bit.
rvalid_qtw	Input	Read valid. Width is 1-bit.

Signal	Direction	Description
wdata_qtw	Output	Write data The width of wdata_qtw is equal to `TCUCFG_QTW_DATA_WIDTH`-bit. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
wlast_qtw	Output	Write last. Width is 1-bit.
wready_qtw	Input	Write ready. Width is 1-bit.
wstrb_qtw	Output	Write strobe The width of wstrb_qtw is calculated as (`TCUCFG_QTW_DATA_WIDTH`/8)-bit. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
wvalid_qtw	Output	Write valid. Width is 1-bit.
wpoison_qtw	Output	Write data poison Width is `TCUCFG_QTW_DATA_WIDTH` / 64
bid_qtw	Input	Response ID Width is `QTW_ID_WIDTH`-bit. `QTW_ID_WIDTH` is calculated as follows: ceil($\log_2$(max(`TCUCFG_DGW_SLOTS`, `TCUCFG_PTW_SLOTS`, 8))) + 3
bready_qtw	Output	Response ready. Width is 1-bit.
bresp_qtw	Input	Write response. Width is 2-bit.
bvalid_qtw	Input	Write response valid. Width is 1-bit.
bidunq_qtw	Input	Write response channel unique ID indicator. Width is 1-bit.
awakeup_qtw	Output	Wakeup. Width is 1-bit.

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

A.1.3 TCU PTW interface signals

The TCU Page Table Walk (PTW) interface signals are based on the AMBA ACE5-Lite signals.


Note

The TCU PTW interface signals are valid for TCUx2 only.

Signal definitions

Table A-3: TCU PTW interface signals

Signal	Direction	Description
arid_ptw	Output	Read address ID The width of arid_ptw is equal to QTW_ID_WIDTH. QTW_ID_WIDTH is calculated as follows: ceil($\log_2$(max(TCUCFG_DGW_SLOTS, TCUCFG_PTW_SLOTS, 8))) + 3
araddr_ptw	Output	Read address. Width is 52-bit.
arburst_ptw	Output	Burst type. Width is 2-bit.
arcache_ptw	Output	Memory type. Width is 4-bit.
ardomain_ptw	Output	Shareability domain. Width is 2-bit.
arlen_ptw	Output	Burst length. Width is 8-bit.
arlock_ptw	Output	Lock type. Width is 1-bit.
arprot_ptw	Output	Protection type. Width is 3-bit.
arqos_ptw	Output	QoS identifier. Width is 4-bit.
arready_ptw	Input	Read address ready. Width is 1-bit.
arsize_ptw	Output	Burst size. Width is 3-bit.
arsnoop_ptw	Output	Transaction type. Width is 4-bit.
arvalid_ptw	Output	Read address valid. Width is 1-bit.
aridunq_ptw	Output	Read address channel unique ID indicator. Width is 1-bit.
armpam_ptw	Output	Read address channel MPAM information. Width is (TCUCFG_LEGACY_TZ_EN == 1) ? 14 : 15 The MPAM signal position is aligned with MPAM_12_1
arnse_ptw	Output	Along with ARPROT[1] indicates whether the physical address space is Secure, Non-Secure, Root, or Realm. Width is 1 bit
arpbha_ptw	Output	Page-based hardware attributes for write requests. Width is 4-bit.
armecid_ptw	Output	MECID for requests on the write request channel Width is (TCUCFG_MECID_WIDTH == 0) ? 1 : 16
awid_ptw	Output	Write address ID Width is QTW_ID_WIDTH-bit. QTW_ID_WIDTH is calculated as follows: ceil($\log_2$(max(TCUCFG_DGW_SLOTS, TCUCFG_PTW_SLOTS, 8))) + 3
awaddr_ptw	Output	Write address. Width is 52-bit.
awburst_ptw	Output	Burst type. Width is 2-bit.
awcache_ptw	Output	Memory type. Width is 4-bit.
awdomain_ptw	Output	Shareability domain. Width is 2-bit.

Signal	Direction	Description
awlen_ptw	Output	Burst length. Width is 8-bit.
awlock_ptw	Output	Lock type. Width is 1-bit.
awprot_ptw	Output	Protection type. Width is 3-bit.
awqos_ptw	Output	QoS identifier. Width is 4-bit.
awready_ptw	Input	Write address ready. Width is 1-bit.
awsize_ptw	Output	Burst size. Width is 3-bit.
awsnoop_ptw	Output	Transaction type. Width is 4-bit.
awvalid_ptw	Output	Write address valid. Width is 1-bit.
awatop_ptw	Output	Atomic operation. Width is 6-bit.
awidunq_ptw	Output	Write address channel unique ID indicator. Width is 1-bit.
awmpam_ptw	Output	Write address channel MPAM information. Width is (TCUCFG_LEGACY_TZ_EN == 1) ? 14 : 15 The MPAM signal position is aligned with MPAM_12_1
awnse_ptw	Output	Along with ARPROT[1], indicates whether the physical address space is Secure, Non-Secure, Root, or Realm. Width is 1-bit.
awpbha_ptw	Output	Page-based hardware attributes for write requests. Width is 4-bit.
awmecid_ptw	Output	MECID for requests on the write request channel. Width is (TCUCFG_MECID_WIDTH == 0) ? 1 : 16
rdata_ptw	Input	Read data. The width of rdata_ptw is equal to TCUCFG_QTW_DATA_WIDTH See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
rid_ptw	Input	Read data ID Width is QTW_ID_WIDTH-bit. QTW_ID_WIDTH is calculated as follows: ceil($\log_2$ (max(TCUCFG_DGW_SLOTS, TCUCFG_PTW_SLOTS, 8))) + 3 See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
ridunq_ptw	Input	Read data channel unique ID indicator, active HIGH. Width is 1-bit.
rlast_ptw	Input	Last data beat of the read. Width is 1-bit.
rpoison_ptw	Input	Read poison input to the TCU. Width is TCUCFG_QTW_DATA_WIDTH / 64
rready_ptw	Output	Read ready. Width is 1-bit.
rresp_ptw	Input	Read response. Width is 2-bit.
rvalid_ptw	Input	Read valid. Width is 1-bit.
wdata_ptw	Output	Write data The width of wdata_ptw is equal to TCUCFG_QTW_DATA_WIDTH-bit. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
wlast_ptw	Output	Write last. Width is 1-bit.

Signal	Direction	Description
wready_ptw	Input	Write ready. Width is 1-bit.
wstrb_ptw	Output	Write strobe The width of wstrb_ptw is calculated as (TCUCFG_QTW_DATA_WIDTH/8)-bit. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
wvalid_ptw	Output	Write valid. Width is 1-bit.
wpoison_ptw	Output	Write data poison. Width is TCUCFG_QTW_DATA_WIDTH / 64
bid_ptw	Input	Response ID Width is QTW_ID_WIDTH-bit. QTW_ID_WIDTH is calculated as follows: ceil($\log_2$(max(TCUCFG_DGW_SLOTS, TCUCFG_QTW_SLOTS, 8))) + 3
bready_ptw	Output	Response ready. Width is 1-bit.
bresp_ptw	Input	Write response. Width is 2-bit.
bvalid_ptw	Input	Write response valid. Width is 1-bit.
bidunq_ptw	Input	Write response channel unique ID indicator. Width is 1-bit.
awakeup_ptw	Output	Wakeup. Width is 1-bit.

See the *AMBA® AXI Protocol Specification*.

A.1.4 TCU programming interface signals

The TCU programming interface signals are based on the AMBA APB5 signals.

Signal definitions

Table A-4: TCU programming interface signals

Signal	Direction	Description
paddr_prog	Input	Peripheral address. Width is 25-bit.
psel_prog	Input	Peripheral select. Width is 1-bit.
penable_prog	Input	Enable for transfer. Width is 1-bit.
pwrite_prog	Input	Write transaction indicator. Width is 1-bit.
pnse_prog	Input	The pnse_prog signal, in combination with the pprot_prog signal, determines the physical address space of the transaction, that is, Root or Realm. For more information, see the *AMBA® APB Protocol Specification*. Width is 1-bit.
pprot_prog	Input	Protection type. Width is 3-bit.
pwdata_prog	Input	Write data. Width is 32-bit.
pstrb_prog	Input	Write data strobe. Width is 4-bit.
pslverr_prog	Output	Error response. Width is 1-bit.
prdata_prog	Output	Read data. Width is 32-bit.

Signal	Direction	Description
pready_prog	Output	Transfer ready. Width is 1-bit.
pwakeup_prog	Input	Interface wakeup. Width is 1-bit.

For more information, see the *AMBA® APB Protocol Specification*.

A.1.5 TCU SYSCO interface signals

The following table shows the TCU SYSCO interface signals.

Signal definitions

Table A-5: TCU SYSCO interface signals

Signal	Direction	Description
syscoreq_qtw	Output	System coherency request. This output transitions: **HIGH** To indicate that the requester is requesting to enter the coherency domain. **LOW** To indicate that the requester is requesting to exit the coherency domain. Width is 1-bit.
syscoack_qtw	Input	System coherency acknowledge. The syscoack_qtw signal is an asynchronous input port which is synchronized internally. This input transitions to the same level as syscoreq_qtw when the request to enter or exit the coherency domain is complete. Width is 1-bit.

A.1.6 TCU LPI_PD interface signals

The following table shows the TCU LPI_PD interface signals.

Signal definitions

Table A-6: TCU LPI_PD interface signals

Signal	Direction	Description
qactive_pd	Output	Component active. Width is 1-bit.
qreqn_pd	Input	Quiescence request. Width is 1-bit.
qacceptn_pd	Output	Quiescence accept. Width is 1-bit.
qdeny_pd	Output	Quiescence deny. Width is 1-bit.

For more information, see the *AMBA® Low Power Interface Specification*.

A.1.7 TCU LPI_CG interface signals

The following table shows the TCU LPI_CG interface signals.

Signal definitions

Table A-7: TCU LPI_CG interface signals

Signal	Direction	Description
qactive_cg	Output	Component active. Width is 1-bit.
qreqn_cg	Input	Quiescence request. Width is 1-bit.
qacceptn_cg	Output	Quiescence accept. Width is 1-bit.
qdeny_cg	Output	Quiescence deny. Width is 1-bit.

For more information, see the *AMBA® Low Power Interface Specification*.

A.1.8 TCU DTI interface signals

The following table shows the TCU DTI interface signals.

In the following table, the 'Direction' has the following meaning:

Input
TBU to TCU

Output
TCU to TBU

Signal definitions

Table A-8: TCU DTI interface signals

Signal	Direction	Description
tvalid_dti_dn	Input	Flow control signal on the downstream channel. Width is 1-bit.
tready_dti_dn	Output	Flow control signal on the downstream channel. Width is 1-bit.
tdata_dti_dn	Input	Message data signal. Width is 160-bit.
tkeep_dti_dn	Input	This signal indicates valid bytes. Width is 20-bit.
tlast_dti_dn	Input	Indicates the last cycle of a message. Width is 1-bit.
tid_dti_dn	Input	Identifies the requester (TBU) that initiated the message. Width is 4-bit or 6-bit. The width of tid_dti_dn is calculated as follows: If `TCUCFG_NUM_TBU` is 62, the width of tid_dti_dn is 6-bit. Otherwise, the width of tid_dti_dn is 4-bit. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
twakeup_dti_dn	Input	Wake up signal. Width is 1-bit.
tvalid_dti_up	Output	Flow control signal on the upstream channel. Width is 1-bit.
tready_dti_up	Input	Flow control signal on the upstream channel. Width is 1-bit.

Signal	Direction	Description
tdata_dti_up	Output	Message data signal. Width is 160-bit or 192-bit. If `TCUCFG_MECID_WIDTH` is 0 or `TCUCFG_LEGACY_TZ_EN` is 1, then the width of this signal is 160-bit, otherwise it is 192-bit.
tkeep_dti_up	Output	Indicates valid bytes. Width is 20-bit or 24-bit. If `TCUCFG_MECID_WIDTH` is 0 or `TCUCFG_LEGACY_TZ_EN` is 1, then the width of this signal is 20-bit, otherwise it is 24-bit.
tlast_dti_up	Output	Indicates the last cycle of a message. Width is 1-bit.
tdest_dti_up	Output	Identifies the requester, TBU, that is receiving the message. Width is 4-bit or 6-bit. If `TCUCFG_NUM_TBU` is 62, the width of tdest_dti_up is 6-bit. Otherwise, the width of tdest_dti_up is 4-bit. See 2.7.1 TCU I/O and buffer configuration parameters on page 120.
twakeup_dti_up	Output	Wake up signal. Width is 1-bit.

For more information about the DTI signals, see the *AMBA® AXI-Stream Protocol Specification*.

For more information about DTI protocol messages, see the *AMBA® DTI Protocol Specification*.

A.1.9 TCU interrupt signals

The TCU interrupt signals consist of edge-triggered and level-triggered interrupts. For edge triggered interrupts, the interrupt controller must detect the rising edge of these signals.

The TCU can also output the following as Message Signaled Interrupts (MSIs) on the QTW/DVM interface and the dedicated MSI delivery interface:

- Secure, Non-secure, and Realm Event queue
- SYNC complete commands
- Global interrupts
- Non-secure and Realm Page Request Interface (PRI) queue interrupts

If the system supports capturing MSIs from the TCU, there is no requirement to connect the corresponding interrupt signals in this interface.

Note

In some cases it is possible for the SMMU to enter a powered-down state and even reset after issuing an interrupt, removing the evidence of the source of the interrupt. Always ensure that you write the interrupt-handling software to handle such an occurrence.

Signal definitions

Table A-9: TCU interrupt interface signals

Signal	Direction	Description
event_q_irpt_s	Output	Edge-triggered Event queue, Secure interrupt. The event_q_irpt_s signal asserts a Secure interrupt to indicate that the Secure Event queue is not empty. Width is 1-bit.
event_q_irpt_ns	Output	Edge-triggered Event queue, Non-secure interrupt. The event_q_irpt_ns signal asserts a Non-secure interrupt to indicate that the Non-secure Event queue is not empty. Width is 1-bit.
event_q_irpt_r	Output	Edge-triggered Event queue, Realm interrupt. The event_q_irpt_ns signal asserts a Non-secure interrupt to indicate that the Realm Event queue is not empty. Width is 1-bit.
cmd_sync_irpt_ns	Output	Edge-triggered SYNC complete, Non-secure interrupt. The cmd_sync_irpt_ns signal asserts a Non-secure interrupt to indicate that the `CMD_SYNC` command on the Non-secure Command queue is complete. Width is 1-bit.
cmd_sync_irpt_s	Output	Edge-triggered SYNC complete, Secure interrupt. The cmd_sync_irpt_s signal asserts a Secure interrupt to indicate that the `CMD_SYNC` command on the Secure Command queue is complete. Width is 1-bit.
cmd_sync_irpt_r	Output	Edge-triggered SYNC complete, Realm interrupt. The cmd_sync_irpt_r signal asserts a Realm interrupt to indicate that the `CMD_SYNC` command on the Realm Command queue is complete. Width is 1-bit.
global_irpt_ns	Output	The edge-triggered global_irpt_ns signal asserts a global Non-secure interrupt. Width is 1-bit.
global_irpt_s	Output	The edge-triggered global_irpt_s signal asserts a global Secure interrupt. Width is 1-bit.
global_irpt_r	Output	The edge-triggered global_irpt_r signal asserts a global Realm interrupt. Width is 1-bit.
ras_et_fhi	Output	Edge-triggered fault handling RAS interrupt for a contained or an uncontained error. **Note:** TCU_ERRCTLR.FI can also enable or disable ras_fhi. See 3.8.2 TCU_ERRCTLR register on page 176. Width is 1-bit. **Note:** MMU S3 cannot output this interrupt as an MSI. You must connect this output to an interrupt controller.
ras_et_eri	Output	Edge-triggered error recovery RAS interrupt for an uncontained error. Width is 1-bit. **Note:** MMU S3 cannot output this interrupt as an MSI. You must connect this output to an interrupt controller.
ras_et_cri	Output	Edge-triggered critical error interrupt, for an uncontainable uncorrected error. Width is 1-bit. **Note:** MMU S3 cannot output this interrupt as an MSI. You must connect this output to an interrupt controller.
ras_lt_fhi	Output	Level-triggered fault handling RAS interrupt for a contained or an uncontained error. **Note:** TCU_ERRCTLR.FI can also enable or disable ras_fhi. See 3.8.2 TCU_ERRCTLR register on page 176. Width is 1-bit. **Note:** MMU S3 cannot output this interrupt as an MSI. You must connect this output to an interrupt controller.

Signal	Direction	Description
ras_lt_eri	Output	Level-triggered error recovery RAS interrupt for an uncontained error. Width is 1-bit. **Note:** MMU S3 cannot output this interrupt as an MSI. You must connect this output to an interrupt controller.
ras_lt_cri	Output	Level-triggered critical error interrupt, for an uncontainable uncorrected error. Width is 1-bit. **Note:** MMU S3 cannot output this interrupt as an MSI. You must connect this output to an interrupt controller.
ras_lt_irpt_v	Output	Level-triggered valid output for connection to System RAS agents. Asserted when the RAS record contains at least one valid error. Connect the ras_lt_irpt_v signal, together with level-triggered versions of RAS interrupts, to a System RAS agent. Width is 1-bit.
pmu_irpt	Output	Edge-triggered PMU interrupt assertion signal. Width is 1-bit. **Note:** MMU S3 cannot output PMU interrupts as MSIs. You must connect this output to an interrupt controller.
pri_q_irpt_ns	Output	Edge-triggered Non-Secure Page Request Interface (PRI) queue interrupt assertion signal. Width is 1-bit.
pri_q_irpt_r	Output	Edge triggered Realm Page Request Interface (PRI) queue interface assertion signal. Width is 1-bit.
gpf_far	Output	Edge-triggered signal to indicate that an error becomes active in SMMU_ROOT_GPF_FAR. Width is 1-bit.
gpt_cfg_far	Output	Edge-triggered signal to indicate that an error becomes active in SMMU_ROOT_GPT_CFG_FAR. Width is 1-bit.

A.1.10 TCU Message Signaled Interrupt interface signals

The MMU S3 TCU contains a Message Signaled Interrupt (MSI) interface that follows the AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled, protocol and uses signals to send MSIs.

The MSI interface uses the signals in the following table to send MSIs.

Signal definitions

Table A-10: TCU MSI interface signals

Signal	Direction	Description	Connection information
msitvalid	Output	Indicates valid data to the GIC. Width is 1-bit.	AXI-Stream signal is TVALID
msitready	Input	Indicates acceptance by the GIC. Width is 1-bit.	AXI-Stream signal is TREADY
msitdata	Output	Data being passed to the GIC. Width is 64-bit.	AXI-Stream signal is TDATA
msitwakeup	Output	Indicates that a transaction is ongoing. Width is 1-bit.	AXI-Stream signal is TWAKEUP, AMBA extension
msirtvalid	Input	Indicates that the GIC has accepted an MSI. Width is 1-bit.	AXI-Stream signal is TVALID
msirtready	Output	Indicates that the device has accepted the response packet. Width is 1-bit.	AXI-Stream signal is TREADY
msirtwakeup	Input	Indicates that a transaction is ongoing. Width is 1-bit.	AXI-Stream signal is TWAKEUP, AMBA extension

For more information about these signals, see the *GIC MSI Delivery Interface* document.

A.1.11 TCU event interface

The TCU event interface is an event handshake for connection to processors.

Signal definitions

Table A-11: TCU event interface signals

Signal	Direction	Description
eventoreq	Output	The eventoreq signal is asserted until the eventoack signal indicates an event that enables processors to wake up from the Wait For Event (WFE) low-power state. The eventoreq signal is asserted to indicate an event that enables processors to wake up from the Wait For Event (WFE) low-power state. The eventoreq signal remains asserted until the eventoack signal is asserted. The eventoreq signal can then be deasserted followed by the eventoack signal. This is a four-phase handshake. Connect the eventoreq signal of the TCU to the event interface of Arm® processors. Processors that use the DynamIQ Shared Unit (DSU) have a different event handshake mechanism. Arm® processors can use the following event mechanisms: • Some processors have an eventi input which only requires a single pulse. This can be connected to the eventoreq, with eventoack also connected to eventoreq to provide a dummy acknowledgement. • Some processors, including DSU-based systems, have a req/ack handshake mechanism that can be connected directly to the corresponding signals on the TCU. **Note:** You can also route the eventoreq and eventoack signals through other interconnects such as the Neoverse™ CMN-700 Coherent Mesh Network instead of connecting these signals directly to the processor. These interconnects, like the DSU, support only the req/ack handshake mechanism. In both mechanisms, in the signal names: **i** Represents events that are inputs to a particular component **o** Represents events that are outputs from a particular component The eventoreq signal is held HIGH until eventoack is observed. Width is 1-bit.
eventoack	Input	Acknowledge signal for eventoreq. The eventoack signal clears the current eventoreq output. If eventoreq is connected to a pulse interface, also connect eventoreq to eventoack to provide a dummy acknowledgment. Width is 1-bit.

For more information, see your processor or DSU documentation.

A.1.12 TCU tie-off signals

The TCU tie-off signals are sampled between exiting reset and the LPI_PD interface first entering the Q_RUN state. Ensure that the value of these signals does not change when the LPI_PD interface is in the Q_STOPPED or Q_EXIT state for the first time after exiting reset.

Signal definitions

Table A-12: TCU tie-off signals

Signal	Direction	Description
sup_cohacc	Input	This signal indicates whether the QTW interface and the PTW interface, if present, are I/O-coherent. Tie the sup_cohacc signal HIGH when the TCU is connected to a coherent interconnect. Width is 1-bit. **Note:** If RME mode is enabled, the TCU must be connected to a coherent interconnect. Therefore, you must tie the sup_cohacc signal HIGH.
sup_btm	Input	This signal indicates whether the Broadcast TLB Maintenance is supported. Tie the sup_btm signal HIGH when the TCU is connected to an interconnect that supports DVM. Width is 1-bit.
sup_sev	Input	This signal indicates whether the Send Event mechanism is supported. Tie the sup_sev signal HIGH when the eventoreq and eventoack signals are connected. Width is 1-bit.
sup_oas	Input	Output address size supported. The encodings for this input are as follows: **0b000** 32 bits **0b001** 36 bits **0b010** 40 bits **0b011** 42 bits **0b100** 44 bits **0b101** 48 bits **0b110** 52 bits You must not use other encodings. Other encodings are treated as `0b110`. Width is 3-bit.
sec_override	Input	When HIGH, certain registers are accessible to Non-secure accesses from reset, as the 3.7.5 TCU_SCR register on page 161 settings describe. Width is 1-bit.
ecorevnum	Input	Tie the ecorevnum signal to 0 unless directed otherwise by Arm. Width is 4-bit.
msi_addr	Input	If the programmed Message Signaled Interrupt (MSI) address in SMMU_(S/R_)_*_IRQ_CFG0.ADDR matches msi_addr, then an MSI is generated on the TCU MSI interface. Width is 52-bit.
tcu_sid	Input	Used as the DeviceID for TCU-generated MSIs. **Note:** This is only for MSIs that are issued from the dedicated AXI5-Stream, with Wakeup_Signal enabled and Check_Type not enabled, MSI delivery interface. Width is 32-bit.

Signal	Direction	Description
sup_httu	Input	**0** When set to 0, sup_httu indicates that the ACE-Lite interface that is connected to a system that cannot support atomics. The TCU cannot perform Hardware Translation Table Update (HTTU) transactions. **1** When set to 1, the sup_httu signal indicates that the ACE-Lite interface that is connected to a system that can support atomics. The TCU uses atomic transactions to perform HTTU. **Note:** **TCUx1** If you use the TCUx1 type, the HTTU traffic uses the QTW interface. **TCUx2** If you use the TCUx2 type, the HTTU traffic uses the PTW interface. The impact of the sup_httu signal on SMMU_IDR0.HTTU is as follows: **sup_httu is `1'b0`** SMMU_IDR0.HTTU is `2'b00` **sup_httu is `1'b1`** SMMU_IDR0.HTTU is `2'b10` See 2.6.1 SMMUv3 implementation on page 88. Width is 1-bit.
legacy_tz_en	Input	Disable Realm Management Extension (RME) and switch to legacy mode. Width is 1-bit.
l0_gpt_sz	Input	Set the value of Level 0 GPT entry size, L0GPTSZ, to use for RME: **0b0000** 30-bits. Each entry covers 1GB of address space. **0b0100** 34-bits. Each entry covers 16GB of address space. **0b0110** 36-bits. Each entry covers 64GB of address space. **0b1001** 39-bits. Each entry covers 512GB of address space. Width is 4-bit.

Signal	Direction	Description
hazard_width	Input	**Note:** TCU hazarding is disabled in MMU S3 version r1p0 and later. This tie-off signal has no effect when hazarding is disabled. Width to create hazards in the HZU unit. Uses the same encodings as AxSIZE, as follows: **3'h3** 64 **3'h4** 128 **3'h5** 256 **3'h6** 512 **3'h7** Reserved Width is 3-bit. The decoded value must be ≤ `TCUCFG_QTW_DATA_WIDTH`. When you set the hazard_width signal to a larger size, the walk requests from the TCU are performed at a larger size. This larger size can allow the possibility of hazarding more than one page table entry with a single request on the QTW/PTW interfaces. This larger size provides more benefit for sequential workloads which tend to load adjacent pages. We recommend setting this to a size which is natural for the interconnect or memory system to use. For example, if the interconnect transfers all data at 256 bit packets regardless of AxSIZE, setting this to 256 might have very little overhead to the system but provide benefit by reducing PTW traffic.
ns_gbpa_abort_init	Input	Non-secure global bypass. The behavior on reset is as follows: **0** On reset, do not abort all incoming transactions. **1** On reset, abort all incoming transactions. The ns_gbpa_abort_init signal sets the reset value of SMMU_GBPA.ABORT. Width is 1-bit.
s_gbpa_abort_init	Input	Secure global bypass. The behavior on reset is as follows: **0** On reset, do not abort all incoming transactions. **1** On reset, abort all incoming transactions. The s_gbpa_abort_init signal sets the reset value of SMMU_S_GBPA.ABORT. Width is 1-bit.

For more information about the SMMUv3 ID signals, see the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

A.1.13 TCU ELA debug signals

The MMU S3 TCU includes Embedded Logic Analyzer (ELA) debug signals.

Signal definitions

Table A-13: ELA enable signals

Signal	Direction	Description
ela_enable	Input	The ela_enable signal is an asynchronous input port. When `TCUCFG_USE_ELA_DEBUG` is 0, the SMMU ignores the value of the signal. When `TCUCFG_USE_ELA_DEBUG` is 1, ela_enable acts as a clock enable for the TCU ELA observation interface. If ELA debug is required, drive ela_enable HIGH. If ELA debug is not required, drive ela_enable LOW to reduce the dynamic power consumption of the SMMU. Width is 1-bit.
signalgrp0..11_qtw	Output	Group observed signals. The signalgrp0..11_qtw interface is present on TCUx1 and TCUx2. Width is `ELA_GRP_WIDTH`-bit
sigqual0..11_qtw	Output	Group qualification signal. The sigqual0..11_qtw interface is present on TCUx1 and TCUx2. Width is (`ELA_GRP_WIDTH`/32)-bit.
sigclken0..11_qtw	Output	Group enable signal. The sigclken0..11_qtw interface is present on TCUx1 and TCUx2. Width is 1-bit.
signalgrp0..11_ptw	Output	Group observed signals. The signalgrp0..11_ptw interface is present only on TCUx2. Width is `ELA_GRP_WIDTH`-bit
sigqual0..11_ptw	Output	Group qualification signals. The sigqual0..11_ptw interface is present only on TCUx2. Width is (`ELA_GRP_WIDTH` / 32)-bit
sigclken0..11_ptw	Output	Group enable signals. The sigclken0..11_ptw interface is present only on TCUx2. Width is 1-bit

A.2 TBU signals

MMU S3 contains TBU signals.

A.2.1 TBU clock and reset signals

The TBU uses a single set of standard clock and reset signals.

Signal definitions

Table A-14: Clock and reset signals

Signal	Direction	Description
clk	Input	Global clock. Width is 1-bit.
resetn	Input	Global reset. Width is 1-bit.

A.2.2 TBU TBS interface signals

The TBU TBS interface signals are based on the AMBA ACE5-Lite signals. This interface applies to the ACE-Lite TBU componentsand Dual TBU components.

Signal definitions

Table A-15: TBU TBS interface signals

Signal	Direction	Description
araddr_s	Input	Read address. Width is 64-bit.
arburst_s	Input	Burst type. Width is 2-bit.
arcache_s	Input	Memory type. Width is 4-bit.
archunken_s	Input	Read data chunking enable. Width is 1-bit.
arpbha_s	Input	Page-based hardware attributes. Width is 4-bit.
artagop_s	Input	Indicates whether MTE tags are associated with a read transaction. Width is 2-bit.
armmuvalid_s	Input	MMU qualifier signal. When deasserted, the transaction address is a physical address and does not require translation. Width is 1-bit.
ardomain_s	Input	Shareability domain. Width is 2-bit.
arid_s	Input	Read address ID The width of arid_s is `TBUCFG_ID_WIDTH`-bit. For information about how to set the `TBUCFG_ID_WIDTH` parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
aridunq_s	Input	Read address channel unique ID indicator, active-HIGH. Width is 1-bit.
arlen_s	Input	Burst length. Width is 8-bit.
arlock_s	Input	Lock type. Width is 1-bit.
arloop_s	Input	Loopback value for a read transaction. Reflected back on RLOOP. The width of arloop_s is `TBUCFG_LOOP_WIDTH`-bit. For information about how to set the `TBUCFG_LOOP_WIDTH` parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.

Signal	Direction	Description
armecid_s	Input	MECID for requests on the read request channel. **Note:** This value is considered only for NoStreamID transactions. Width is 0, 12 or 16 bits depending on the `TBUCFG_MECID_WIDTH` parameter. For information about how to set the `TBUCFG_MECID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122
armmuflow_s	Input	Indicates the SMMU flow for managing translation faults. Width is 2-bit.
armmussid_s	Input	These signals indicate the StreamID, SubstreamID, and ATS translated status of the originating transaction. The AXI5 Untranslated_Transactions extension defines these signals. For information about how to set these parameters, see 2.4.2.2 ACE-Lite TBU TBM interface on page 37 and 2.6.2 AMBA implementation on page 94. The width of armmusid_s is `TBUCFG_SSID_WIDTH`-bit. For information about how to set the `TBUCFG_SSID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
armmusid_s	Input	These signals indicate the StreamID, SubstreamID, and ATS translated status of the originating transaction. The AXI5 Untranslated_Transactions extension defines these signals. For information about how to set these parameters, see 2.4.2.2 ACE-Lite TBU TBM interface on page 37 and 2.6.2 AMBA implementation on page 94. The width of armmusid_s is `TBUCFG_SSID_WIDTH`-bit. For information about how to set the `TBUCFG_SSID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
armmussidv_s	Input	These signals indicate the StreamID, SubstreamID, and ATS translated status of the originating transaction. The AXI5 Untranslated_Transactions extension defines these signals. For information about how to set these parameters, see 2.4.2.2 ACE-Lite TBU TBM interface on page 37 and 2.6.2 AMBA implementation on page 94. Width is 1-bit.
armmusecsid_s	Input	These signals indicate the StreamID, SubstreamID, and ATS translated status of the originating transaction. The AXI5 Untranslated_Transactions extension defines these signals. For information about how to set these parameters, see 2.4.2.2 ACE-Lite TBU TBM interface on page 37 and 2.6.2 AMBA implementation on page 94. Width is `SECSID_WIDTH`-bit.
arprot_s	Input	Protection type. Width is 3-bit.
arnse_s	Input	Combines with arprot_s to define the PAS. Width is 1-bit. If the `TBUCFG_LEGACY_TZ_EN` parameter is set to 1, the arnse_s signal is 1-bit, unused, and tied-off internally.
arqos_s	Input	Quality of Service (QoS). Width is 4-bit.
arready_s	Output	Read address ready. Width is 1-bit.
arregion_s	Input	Region identifier. Width is 4-bit.
arsize_s	Input	Burst size. Width is 3-bit.
arsnoop_s	Input	Transaction type of read transaction. Width is 4-bit.

Signal	Direction	Description
aruser_s	Input	Read address (AR) channel User signal Calculate the width of aruser_s as follows: (TBUCFG_ARUSER_WIDTH + LTI_TLBLOC_WIDTH_RAW -)-bit. Calculate the LTI_TLBLOC_WIDTH_RAW internal parameter as follows: LTI_TLBLOC_WIDTH_RAW = ((TBUCFG_DIRECT_IDX == 1) ? (TBUCFG_MTLB_DEPTH > 0) ? $\log_2$(TBUCFG_MTLB_DEPTH) : $\log_2$(4) : $\log_2$(TBUCFG_MTLB_PARTS)) For information about how to set these parameters, see 2.7 Configuration parameters and methodology on page 120.
arvalid_s	Input	Read address valid. Width is 1-bit.
rchunknum_s	Output	Read data chunk number. Width is CHUNKNUM_WIDTH-bit.
rchunkstrb_s	Output	Read data chunk strobe. Width is CHUNKSTRB_WIDTH-bit.
rchunkv_s	Output	Valid signal of RCHUNKNUM and RCHUNKSTRB. Width is 1-bit.
rdata_s	Output	Read data. The width of rdata_s is TBUCFG_DATA_WIDTH-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rid_s	Output	Read ID. The width of rid_s is TBUCFG_ID_WIDTH-bit. For information about how to set the TBUCFG_ID_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
ridunq_s	Output	Read data channel unique ID indicator, active-HIGH. Width is 1-bit.
rlast_s	Output	Read last. Width is 1-bit.
rloop_s	Output	Loopback value for a read response The width of rloop_s is TBUCFG_LOOP_WIDTH-bit. For information about how to set the TBUCFG_LOOP_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rpoison_s	Output	Indicates that the read data in this transfer has been corrupted The width of rpoison_s is (TBUCFG_DATA_WIDTH / 64)-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rready_s	Input	Read ready. Width is 1-bit.
rresp_s	Output	Read response. Width is 3-bit.
rtag_s	Output	Memory tag associated with data. Width is ceil(TBUCFG_DATA_WIDTH / 128) × 4.
ruser_s	Output	Read data (R) channel User signal The width if ruser_s is TBUCFG_RUSER_WIDTH-bit. For information about how to set the TBUCFG_RUSER_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rvalid_s	Output	Read valid. Width is 1-bit.
awaddr_s	Input	Write address. Width is 64-bit.

Signal	Direction	Description
awatop_s	Input	Atomic operation. Width is 6-bit.
awburst_s	Input	Burst type. Width is 2-bit.
awcache_s	Input	Memory type. Width is 4-bit.
awdomain_s	Input	Shareability domain. Width is 2-bit.
awid_s	Input	Write address ID The width of awid_s is `TBUCFG_ID_WIDTH`-bit. For information about how to set the `TBUCFG_ID_WIDTH` parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
awlen_s	Input	Burst length. Width is 8-bit.
awlock_s	Input	Lock type. Width is 1-bit.
awmecid_s	Input	MECID for requests on the write request channel. **Note:** This value is considered only for NoStreamID transactions. Width is 0, 12, or 16 bits depending on the `TBUCFG_MECID_WIDTH` parameter. For information about how to set the `TBUCFG_MECID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
awmmuflow_s	Input	Indicates the SMMU flow for managing translation faults. Width is 2-bit.
awmmussid_s	Input	These signals indicate the StreamID, SubstreamID, and ATS translated status of the originating transaction. The AXI5 Untranslated_Transactions extension defines these signals. For information about how to set these parameters, see 2.4.2.2 ACE-Lite TBU TBM interface on page 37 and 2.6.2 AMBA implementation on page 94. The width of awmmusid_s is `TBUCFG_SSID_WIDTH`-bit. For information about how to set the `TBUCFG_SSID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
awmmusid_s	Input	These signals indicate the StreamID, SubstreamID, and ATS translated status of the originating transaction. The AXI5 Untranslated_Transactions extension defines these signals. For information about how to set these parameters, see 2.4.2.2 ACE-Lite TBU TBM interface on page 37 and 2.6.2 AMBA implementation on page 94. The width of awmmusid_s is `TBUCFG_SID_WIDTH`-bit. For information about how to set the `TBUCFG_SID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
awmmussidv_s	Input	These signals indicate the StreamID, SubstreamID, and ATS translated status of the originating transaction. The AXI5 Untranslated_Transactions extension defines these signals. For information about how to set these parameters, see 2.4.2.2 ACE-Lite TBU TBM interface on page 37 and 2.6.2 AMBA implementation on page 94. Width is 1-bit.
awmmusecsid_s	Input	These signals indicate the StreamID, SubstreamID, and ATS translated status of the originating transaction. The AXI5 Untranslated_Transactions extension defines these signals. For information about how to set these parameters, see 2.4.2.2 ACE-Lite TBU TBM interface on page 37 and 2.6.2 AMBA implementation on page 94. Width is SECSID_WIDTH-bit.
awmmuvalid_s	Input	MMU qualifier signal. When deasserted, the transaction address is a physical address and does not require translation. Width is 1-bit.
awpbha_s	Input	Page-based hardware attributes. Width is 4-bit.

Signal	Direction	Description
awtagop_s	Input	Indicates whether MTE tags are associated with a write transaction. Width is 2-bit.
awprot_s	Input	Protection type. Width is 3-bit.
awnse_s	Input	Combines with awprot_s to define the PAS. Width is 1-bit. If the TBUCFG_LEGACY_TZ_EN parameter is set to 1, the awnse_s signal is 1-bit, unused, and tied-off internally.
awqos_s	Input	QoS. Width is 4-bit.
awready_s	Output	Write address ready. Width is 1-bit.
awregion_s	Input	Region identifier. Width is 4-bit.
awsize_s	Input	Burst size. Width is 3-bit.
awuser_s	Input	Write address (AW) channel User signal The width of awuser_s is (TBUCFG_AWUSER_WIDTH + LTI_TLBLOC_WIDTH_RAW)-bit. Calculate the LTI_TLBLOC_WIDTH_RAW internal parameter as follows: LTI_TLBLOC_WIDTH_RAW = ((TBUCFG_DIRECT_IDX == 1) ? (TBUCFG_MTLB_DEPTH > 0) ? $\log_2$(TBUCFG_MTLB_DEPTH) : $\log_2$(4) : $\log_2$(TBUCFG_MTLB_PARTS)) For information about how to set these parameters, see 2.7 Configuration parameters and methodology on page 120.
awvalid_s	Input	Write address valid. Width is 1-bit.
awakeup_s	Input	Wakeup signal. Width is 1-bit.
awsnoop_s	Input	Transaction type of write transaction. Width is 5-bit.
awstashnid_s	Input	The AXI5 Cache_Stash_Transactions extension defines the awstashnid_s signal. If the TBUCFG_STASH_SUPPORT parameter is set to 0, the awstashnid_s[10:0] signal is ignored. Width is 11-bit.
awstashniden_s	Input	The AXI5 Cache_Stash_Transactions extension defines the awstashniden_s signal. If the TBUCFG_STASH_SUPPORT parameter is set to 0, the awstashniden_s signal is ignored. Width is 1-bit.
awstashlpid_s	Input	The AXI5 Cache_Stash_Transactions extension defines the awstashlpid_s[4:0] signal. If the TBUCFG_STASH_SUPPORT parameter is set to 0, the awstashlpid_s[4:0] signal is ignored. Width is 5-bit.
awstashlpiden_s	Input	The AXI5 Cache_Stash_Transactions extension defines the awstashlpiden_s signal. If the TBUCFG_STASH_SUPPORT parameter is set to 0, the awstashlpiden_s signal is ignored. Width is 1-bit.
awidunq_s	Input	Write address channel unique ID indicator, active-HIGH. Width is 1-bit.
awloop_s	Input	Loopback value for a write transaction The width of awloop_s is TBUCFG_LOOP_WIDTH-bit. For information about how to set the TBUCFG_LOOP_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.

Signal	Direction	Description
wdata_s	Input	Write data The width of wdata_s is TBUCFG_DATA_WIDTH-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
wlast_s	Input	Write last. Width is 1-bit.
wpoison_s	Input	Indicates that the write data in this transfer has been corrupted The width of wpoison_s is [(TBUCFG_DATA_WIDTH / 64)-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
wready_s	Output	Write ready. Width is 1-bit.
wstrb_s	Input	Write strobes The width of wstrb_s is (TBUCFG_DATA_WIDTH / 8)-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
wtag_s	Input	Memory tag associated with data. Width is ceil(TBUCFG_DATA_WIDTH / 128) × 4.
wtagupdate_s	Input	Indicates which tags must be written to memory when AWTAGOP is Update. Width is ceil(TBUCFG_DATA_WIDTH / 128).
wuser_s	Input	Write data (W) channel User signal The width of wuser_s is TBUCFG_WUSER_WIDTH-bit. For information about how to set the TBUCFG_WUSER_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
wvalid_s	Input	Write valid. Width is 1-bit.
bid_s	Output	Response ID The width of bid_s is TBUCFG_ID_WIDTH-bit. For information about how to set the TBUCFG_ID_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
bidunq_s	Output	Write response channel unique ID indicator, active-HIGH. Width is 1-bit.
bloop_s	Output	Loopback value for a write response The width of bloop_s is TBUCFG_LOOP_WIDTH-bit. For information about how to set the TBUCFG_LOOP_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
bready_s	Input	Response ready. Width is 1-bit.
bresp_s	Output	Write response. Width is 3-bit.
buser_s	Output	Write response (B) channel User signal The width of buser_s is TBUCFG_BUSER_WIDTH-bit. For information about how to set the TBUCFG_BUSER_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
bvalid_s	Output	Write response valid. Width is 1-bit.

A.2.3 TBU TBM interface signals

The TBU TBM interface signals are based on the AMBA ACE5-Lite signals. This interface applies to the ACE-Lite TBU componentsand Dual TBU components.

Signal definitions

Table A-16: TBU TBM interface signals

Signal	Direction	Description
araddr_m	Output	Read address. Width is 52-bit.
armmuvalid_m	Output	MMU qualifier signal. Width is 1-bit.
arpbha_m	Output	Page-based hardware attributes. Width is 4-bit.
artagop_m	Output	Indicates whether MTE tags are associated with a read transaction. Width is 2-bit.
arburst_m	Output	Burst type. Width is 2-bit.
arcache_m	Output	Memory type. Width is 4-bit.
archunken_m	Output	Read data chunking enable. Width is 1-bit.
ardomain_m	Output	Shareability domain. Width is 2-bit.
arid_m	Output	Read address ID. The width of arid_m is `TBUCFG_ID_WIDTH`-bit. For information about how to set the `TBUCFG_ID_WIDTH` parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
aridunq_m	Output	Read address channel unique ID indicator, active-HIGH. Width is 1-bit.
arlen_m	Output	Burst length. Width is 8-bit.
arlock_m	Output	Lock type. Width is 1-bit.
arloop_m	Output	Loopback value for a read transaction. Reflected back on RLOOP. Calculate the width of arloop_m as follows: If `TBUCFG_OT_TRACKER_TYPE` is 1, the width of arloop_m is (`TBUCFG_LOOP_WIDTH` + 2)-bit. Otherwise, the width of arloop_m is `TBUCFG_LOOP_WIDTH`-bit. For information about how to set these parameters, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
armecid_m	Output	MECID for requests on the read request channel. Width is 0, 12, or 16 bits depending on the `TBUCFG_MECID_WIDTH` parameter. For information about how to set the `TBUCFG_MECID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122
armmusid_m	Output	Indicates the StreamID of the originating transaction. The width of armmusid_m is `TBUCFG_SID_WIDTH`-bit. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
armmusecsid_m	Output	Indicates the Secure StreamID of the originating transaction. Width is 2-bit.
armpam_m	Output	Read address channel MPAM information. If `TBUCFG_LEGACY_TZ_EN` = 1, then the width of armpam_m is 14-bit. Otherwise, the width of armpam_m is 15-bit.
arprot_m	Output	Protection type. Width is 3-bit.

Signal	Direction	Description
arnse_m	Output	Combines with arprot_m to define the PAS. Width is 1-bit. When TBUCFG_LEGACY_TZ_EN is set to 1, the arnse_m signal is 1-bit, unused, and tied-off internally.
arqos_m	Output	Quality of Service (QoS). Width is 4-bit.
arready_m	Input	Read address ready. Width is 1-bit.
arregion_m	Output	Region identifier. Width is 4-bit.
arsize_m	Output	Burst size. Width is 3-bit.
arsnoop_m	Output	Transaction type of read transaction. Width is 4-bit.
aruser_m	Output	Read address (AR) channel User signal. The width of aruser_m is (TBUCFG_ARUSER_WIDTH + 1)-bit. For information about how to set the TBUCFG_ARUSER_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124. For information about the extra aruser bit, see 2.5.12 AXI USER bits that the SMMU TBU TBM defines on page 87.
arvalid_m	Output	Read address valid. Width is 1-bit.
rchunknum_m	Input	Read data chunk number. Width can be 1-bit, 5-bit, 6-bit, 7-bit, or 8-bit. Calculate the width of rchunknum_m as follows: If TBUCFG_DATA_WIDTH is 128, then the width of rchunknum_m is 8-bit. If TBUCFG_DATA_WIDTH is 256, then the width of rchunknum_m is 7-bit. If TBUCFG_DATA_WIDTH is 512, then the width of rchunknum_m is 6-bit. If TBUCFG_DATA_WIDTH is 1024, then the width of rchunknum_m is 5-bit. Otherwise, the width of rchunknum_m is 1-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rchunkstrb_m	Input	Read data chunk strobe. Calculate the width of rchunkstrb_m as follows: If TBUCFG_DATA_WIDTH is 64, then the width of rchunkstrb_m is 1-bit. Otherwise, the width of rchunkstrb_m is (TBUCFG_DATA_WIDTH / 128)-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rchunkv_m	Input	Valid signal of RCHUNKNUM and RCHUNKSTRB. Width is 1-bit.
rdata_m	Input	Read data. The width of rdata_m is TBUCFG_DATA_WIDTH-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rid_m	Input	Read ID. The width of rid_m is TBUCFG_ID_WIDTH-bit. For information about how to set the TBUCFG_ID_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.

Signal	Direction	Description
ridunq_m	Input	Read data channel unique ID indicator, active-HIGH. Width is 1-bit.
rlast_m	Input	Read last. Width is 1-bit.
rloop_m	Input	Loopback value for a read response. The width of rloop_m is calculated as follows: If `TBUCFG_OT_TRACKER_TYPE` is 1, the width of rloop_m is (`TBUCFG_LOOP_WIDTH` + 2)-bit. Otherwise, the width of rloop_m is `TBUCFG_LOOP_WIDTH`-bit. For information about how to set these parameters, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rpoison_m	Input	Indicates that the read data in this transfer has been corrupted. The width of rpoison_m is (`TBUCFG_DATA_WIDTH` / 64)-bit. For information about how to set the `TBUCFG_DATA_WIDTH` parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rready_m	Output	Read ready. Width is 1-bit.
rresp_m	Input	Read response. Width is 2-bit.
rtag_m	Input	Memory tag associated with data. Width is ceil(`TBUCFG_DATA_WIDTH`/128) × 4.
ruser_m	Input	Read data (R) channel User signal The width of ruser_m is `TBUCFG_RUSER_WIDTH`-bit. For information about how to set the `TBUCFG_RUSER_WIDTH` parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
rvalid_m	Input	Read valid. Width is 1-bit.
awmmuvalid_m	Output	MMU qualifier signal. Width is 1-bit.
awpbha_m	Output	Page-based hardware attributes. Width is 4-bit.
awtagop_m	Output	Indicates whether MTE tags are associated with a write transaction. Width is 2-bit.
awaddr_m	Output	Write address. Width is 52-bit.
awatop_m	Output	Atomic operation. Width is 6-bit.
awburst_m	Output	Burst type. Width is 2-bit.
awcache_m	Output	Memory type. Width is 4-bit.
awdomain_m	Output	Shareability domain. Width is 2-bit.
awid_m	Output	Write address ID The width of awid_m is `TBUCFG_ID_WIDTH`-bit. For information about how to set the `TBUCFG_ID_WIDTH` parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
awlen_m	Output	Burst length. Width is 8-bit.
awlock_m	Output	Lock type. Width is 1-bit.
awmecid_m	Output	MECID for requests on the write request channel. Width is 0, 12 or 16 bits depending on the `TBUCFG_MECID_WIDTH` parameter. For information about how to set the `TBUCFG_MECID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122

Signal	Direction	Description
awmmusid_m	Output	Indicates the StreamID of the originating transaction. The Generic Interrupt Controller (GIC) uses these signals to determine the DeviceID of MSIs that originate from upstream managers. The width of awmmusid_m is `TBUCFG_SID_WIDTH`-bit. For information about how to set the `TBUCFG_SID_WIDTH` parameter, see 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
awmmusecsid_m	Output	Indicates the Secure StreamID of the originating transaction. The GIC uses these signals to determine the DeviceID of MSIs that originate from upstream managers. Width is 2-bit.
awprot_m	Output	Protection type. Width is 3-bit.
awnse_m	Output	Combines with awprot_m to define the PAS. Width is 1-bit. When `TBUCFG_LEGACY_TZ_EN` is set to 1, the awnse_m signal is 1-bit, unused, and tied-off internally.
awqos_m	Output	QoS. Width is 4-bit.
awready_m	Input	Write address ready. Width is 1-bit.
awregion_m	Output	Region identifier. Width is 4-bit.
awsize_m	Output	Burst size. Width is 3-bit.
awstashnid_m	Output	The AXI5 Cache_Stash_Transactions extension defines this signal. See *AMBA® AXI Protocol Specification*. If `TBUCFG_STASH_SUPPORT` = 0, these signals are ignored. Width is 11-bit.
awstashniden_m	Output	The AXI5 Cache_Stash_Transactions extension defines this signal. See *AMBA® AXI Protocol Specification*. If `TBUCFG_STASH_SUPPORT` = 0, these signals are ignored. Width is 1-bit.
awstashlpid_m	Output	The AXI5 Cache_Stash_Transactions extension defines this signal. See *AMBA® AXI Protocol Specification*. If `TBUCFG_STASH_SUPPORT` = 0, these signals are ignored. Width is 5-bit.
awstashlpiden_m	Output	The AXI5 Cache_Stash_Transactions extension defines this signal. See *AMBA® AXI Protocol Specification*. If `TBUCFG_STASH_SUPPORT` = 0, these signals are ignored. Width is 1-bit.
awakeup_m	Output	Wakeup signal. Width is 1-bit.
awidunq_m	Output	Write address channel unique ID indicator, active-HIGH. Width is 1-bit.
awloop_m	Output	Loopback value for a write transaction. The width of awloop_m is calculated as follows: If `TBUCFG_OT_TRACKER_TYPE` is 1, the width of awloop_m is (`TBUCFG_LOOP_WIDTH` + 2)-bit. Otherwise, the width of awloop_m is `TBUCFG_LOOP_WIDTH`-bit. For information about how to set these parameters, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
awmpam_m	Output	Write address channel MPAM information. If `TBUCFG_LEGACY_TZ_EN` = 1, then the width of armpam_m is 14-bit. Otherwise, the width of armpam_m is 15-bit.

Signal	Direction	Description
awsnoop_m	Output	Transaction type of write transaction. Width is 5-bit.
awuser_m	Output	Write address (AW) channel User signal. The width of awuser_m is (TBUCFG_AWUSER_WIDTH + 1)-bit. For information about how to set the TBUCFG_AWUSER_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124. For information about the extra awuser bit, see 2.5.12 AXI USER bits that the SMMU TBU TBM defines on page 87.
awvalid_m	Output	Write address valid. Width is 1-bit.
wtag_m	Output	Memory tag associated with data. Width is ceil(TBUCFG_DATA_WIDTH / 128) × 4.
wtagupdate_m	Output	Indicates which tags must be written to memory when AWTAGOP is Update. Width is ceil(TBUCFG_DATA_WIDTH / 128).
wdata_m	Output	Write data The width of wdata_m is TBUCFG_DATA_WIDTH-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
wlast_m	Output	Write last. Width is 1-bit.
wpoison_m	Output	Indicates that the write data in this transfer has been corrupted. The width of wpoison_m is (TBUCFG_DATA_WIDTH / 64)-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
wready_m	Input	Write ready. Width is 1-bit.
wstrb_m	Output	Write strobes. The width of wstrb_m is (TBUCFG_DATA_WIDTH / 8)-bit. For information about how to set the TBUCFG_DATA_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
wvalid_m	Output	Write valid. Width is 1-bit.
wuser_m	Output	Write data (W) channel User signal. The width of wuser_m is TBUCFG_WUSER_WIDTH-bit. For information about how to set the TBUCFG_WUSER_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
bidunq_m	Input	Write response channel unique ID indicator, active-HIGH. Width is 1-bit.
bid_m	Input	Response ID. TBUCFG_ID_WIDTH-bit. For information about how to set the TBUCFG_ID_WIDTH parameter, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
bloop_m	Input	Loopback value for a write response. If TBUCFG_OT_TRACKER_TYPE is 1, the width of awloop_m is (TBUCFG_LOOP_WIDTH + 2)-bit. Otherwise, the width of awloop_m is TBUCFG_LOOP_WIDTH-bit. For information about how to set these parameters, see 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
bready_m	Output	Response ready. Width is 1-bit.
bresp_m	Input	Write response. Width is 2-bit.

Signal	Direction	Description
buser_m	Input	Write response (B) channel User signal. The width of buser_m is `TBUCFG_BUSER_WIDTH`-bit. See 2.7.5 ACE-Lite TBU I/O configuration parameters on page 124.
bvalid_m	Input	Write response valid. Width is 1-bit.

A.2.4 TBU LPI_PD interface signals

The following table shows the TBU LPI_PD interface signals.

Signal definitions

Table A-17: TBU LPI_PD interface signals

Signal	Direction	Description
qactive_pd	Output	Component active. Width is 1-bit.
qreqn_pd	Input	Quiescence request. Width is 1-bit.
qacceptn_pd	Output	Quiescence accept. Width is 1-bit.
qdeny_pd	Output	Quiescence deny. Width is 1-bit.

A.2.5 TBU LPI_CG interface signals

The following table shows the TBU LPI_CG interface signals.

Signal definitions

Table A-18: TBU LPI_CG interface signals

Signal	Direction	Description
qactive_cg	Output	Component active. Width is 1-bit.
qreqn_cg	Input	Quiescence request. Width is 1-bit.
qacceptn_cg	Output	Quiescence accept. Width is 1-bit.
qdeny_cg	Output	Quiescence deny. Width is 1-bit.

A.2.6 TBU DTI interface signals

The following table shows the TBU DTI interface signals.

In the following table, the 'Direction' has the following meaning:

Input
TCU to TBU

Output
TBU to TCU

Signal definitions

Table A-19: TBU DTI interface signals

Signal	Direction	Description
tvalid_dti_dn	Output	Flow control signal. Width is 1-bit.
tready_dti_dn	Input	Flow control signal. Width is 1-bit.
tdata_dti_dn	Output	Message data signal. Width is 160-bit.
tlast_dti_dn	Output	Indicates the last cycle of a message. Width is 1-bit.
tkeep_dti_dn	Output	Indicates valid bytes. Width is 20-bit.
tvalid_dti_up	Input	Flow control signal. Width is 1-bit.
tready_dti_up	Output	Flow control signal. Width is 1-bit.
tdata_dti_up	Input	Message data signal. Width is 160-bit or 192-bit. If `TBUCFG_MECID_WIDTH` is 0 or `TBUCFG_LEGACY_TZ_EN` is 1, then the width of this signal is 160-bit, otherwise it is 192-bit.
tlast_dti_up	Input	Indicates the last cycle of a message. Width is 1-bit.
tkeep_dti_up	Input	Indicates valid bytes. Width is 20-bit or 24-bit. If `TBUCFG_MECID_WIDTH` is 0 or `TBUCFG_LEGACY_TZ_EN` is 1, then the width of this signal is 20-bit, otherwise it is 24-bit.
twakeup_dti_up	Input	Wakeup signal. Width is 1-bit.
twakeup_dti_dn	Output	Wakeup signal. Width is 1-bit.

For more information about the DTI signals, see the *AMBA® AXI-Stream Protocol Specification*.

For more information about DTI protocol messages, see the *AMBA® DTI Protocol Specification*.

A.2.7 TBU LTI interface signals

This interface applies to the TBU LTI components.

This interface implements an LTI interface as the *AMBA® LTI Protocol Specification* defines. The *AMBA LTI Specification* uses several properties to define signal widths.

2.6.3 Local Translation Interface implementation on page 108 describes how MMU S3 defines the LTI interface properties. For TBU LTI components with multiple interfaces, these have 2, 4, 6, or 8 instances of an LTI interface. In these components, each LTI signal has the port number appended, starting from 0.

For the descriptions of the LTI request channel, response channel, and completion channel signals, see A.5 LTI signals on page 343.

A.2.8 TBU interrupt signals

The TBU interrupt signals consist of edge-triggered and level-triggered interrupts. For edge-triggered interrupts, the interrupt controller must detect the rising edge of these signals.

The MMU S3 TBU cannot output these interrupts as Message Signaled Interrupts (MSIs). These signals must be connected to an interrupt controller.


Note

In some cases it is possible for the SMMU to enter a powered-down state and even reset after issuing an interrupt, removing the evidence of the source of the interrupt. Always ensure that you write interrupt-handling software to handle such an occurrence.

Signal definitions

Table A-20: TBU interrupt signals

Signal	Direction	Description
ras_et_fhi	Output	Edge-triggered fault handling RAS interrupt for a contained error. Width is 1-bit.
ras_et_eri	Output	Edge-triggered error recovery RAS interrupt for an uncontained error. Width is 1-bit.
ras_et_cri	Output	Edge-triggered critical error interrupt, for an uncontainable uncorrected error. Width is 1-bit.
ras_lt_fhi	Output	Level-triggered fault handling RAS interrupt for a contained error. Width is 1-bit.
ras_lt_eri	Output	Level-triggered error recovery RAS interrupt for an uncontained error. Width is 1-bit.
ras_lt_cri	Output	Level-triggered critical error interrupt, for an uncontainable uncorrected error. Width is 1-bit.
ras_lt_irpt_v	Output	Level-triggered valid output for connection to System RAS agents. Asserted when the RAS record contains at least one valid error. Connect the ras_lt_irpt_v signal, together with level triggered versions of RAS interrupts, to a System RAS agent Width is 1-bit.
pmu_irpt	Output	Edge-triggered PMU interrupt. Width is 1-bit.

Signal	Direction	Description
crit_err	Output	Level-triggered critical error. Cannot connect to the TCU because of deny response or insufficient tokens, or max_tok_trans is smaller than required for the TBU to function. **Note:** max_tok_trans must be >= (2 × `NUM_LTI_PORTS`). **Note:** The crit_err interrupt is level-based to make clock domain crossings easy to manage. **Note:** Once triggered, software cannot clear the crit_err interrupt signal. Resetting the TBU is the only way to clear the crit_err interrupt signal. Width is 1-bit.

A.2.9 TBU tie-off signals

The TBU tie-off signals are sampled between exiting reset and the LPI_PD interface first entering the Q_RUN state. Ensure that the signal values do not change when the LPI_PD interface is in the Q_STOPPED or Q_EXIT state for the first time after exiting reset.

Signal definitions

Table A-21: TBU tie-off signals

Signal	Direction	Description
ns_sid_high	Input	Provides the high-order StreamID bits for all transactions with a Non-secure StreamID that pass through the TBU. The width of ns_sid_high is (32 - `TBUCFG_SID_WIDTH`)-bit. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
s_sid_high	Input	Provides the high-order StreamID bits for all transactions with a Secure StreamID that pass through the TBU. The width of s_sid_high is (32 - `TBUCFG_SID_WIDTH`)-bit. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
r_sid_high	Input	Provides the high-order StreamID bits for all transactions with a Realm StreamID that pass through the TBU. The width of r_sid_high is (32 - `TBUCFG_SID_WIDTH`)-bit. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
max_tok_trans	Input	Indicates the number of DTI translation tokens to request when connecting to the TCU, minus 1. **Note:** The TBU must request a minimum of two translation tokens per LTI port. However, we recommend one translation token per translation slot for most scenarios. The ACE-Lite TBU has a single internal LTI port, and the LTI TBU can have multiple LTI ports. The width of max_tok_trans is ($\log_2$`TBUCFG_XLATE_SLOTS`)-bit. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.

Signal	Direction	Description
pcie_mode	Input	You must tie this signal HIGH when the TBU is connected to a PCIe interface. When this signal is HIGH, the TBU interprets the input AXI memory types as encoding PCI No Snoop information. For the TBU to provide correct operation, transactions from the PCIe interface must be delivered to the TBU with the following AXI memory types: **Normal Non-cacheable Bufferable** If No Snoop is set **Normal Write-Back Allocate or Non-allocate Outer Shareable** If No Snoop is not set and TH is 0 **Normal Write-Back Allocate Outer Shareable** If No Snoop is not set and TH is 1 It is not recommended that other memory types are used when pcie_mode is set. If this signal is HIGH, the input attribute and shareability override information in the STE fields MTCFG, SHCFG, and ALLOCCFG are ignored. That is, the transaction attributes are always calculated as if the DTI_TBU_TRANS_RESP.STRW field is EL1-S2, regardless of the actual STRW value. If this signal is HIGH, the input attribute and Shareability override information in the ATTR_OVR field of the DTI_TBU_TRANS_RESP message is ignored. pcie_mode is not compatible with the SMMUv3.2-S2FWB feature. It is not compatible because both features attempt to override the memory type of the transaction - FWB based on the page tables and pcie_mode based on the incoming memory type. pcie_mode is applied after FWB. For example, if FWB is attempting to override the memory type to Write-Back cacheable, but the incoming memory type is Non-cacheable, the final memory type will be Non-cacheable despite the FWB setting. If the system requires FWB to always determine the memory type, then pcie_mode must be set to 0 and the PCIe No Snoop feature cannot be supported. Width is 1-bit.
same_power_domain	Input	If the same_power_domain signal is HIGH, the TBU sets the DTI_TBU_CONDIS_REQ.SPD value to 1 when it attempts to connect to the TCU. The TCU then ignores the connection status of this TBU when calculating its PD_QACTIVE output. Tie the same_power_domain signal HIGH when the TBU is in the same power domain as the TCU to which it is connecting. Tying the same_power_domain signal HIGH simplifies the connection of the power management logic.
sec_override	Input	When HIGH, some registers are accessible to Non-secure accesses from reset, as the 3.16.3 TBU_SCR register on page 253 settings describe. Width is 1-bit.
ecorevnum	Input	Tie this signal to 0 unless we recommend otherwise. Width is 4-bit.

Signal	Direction	Description
utlb_roundrobin	Input	Defines the MicroTLB entry replacement policy. When LOW, the MicroTLB uses a Pseudo Least Recently Used (PLRU) replacement policy. This policy typically provides the best average performance. When HIGH, the MicroTLB uses a round-robin replacement policy. With this policy, the oldest entry is evicted when the MicroTLB is full. Tie this signal HIGH if you want to prevent newer translations from being evicted, even if older translations have been used more recently. Otherwise, tie this signal LOW. Width is 1-bit.
poison_support	Input	**Note:** poison_support applies only to the ACE-Lite TBU. Determines how the ACE-Lite TBU handles RAS errors in the write data buffer. When LOW, the ACE-Lite TBU does not drive the wpoison signal HIGH after detecting an uncorrectable error in the write data buffer. The ACE-Lite TBU reports an uncontainable uncorrected RAS error. When HIGH, the ACE-Lite TBU drives the wpoison signal HIGH after detecting an uncorrectable error in the write data buffer and reports a deferred RAS error. wpoison is driven HIGH for all write data beats of the corrupted transaction. There is no effect the pass-through of the wpoison signal from the TBS interface to the TBM interface, or rpoison from the TBM interface to the TBS interface. That is, if poison_support is LOW, and wpoison is driven HIGH on the TBS interface, then the ACE-Lite TBU drives the TBM wpoison HIGH for the related transaction. However, it is expected that poison_support is a system-wide setting. It is expected that if poison_support is LOW, no components in the system can generate poison. Width is 1-bit.
legacy_tz_en	Input	Disable Realm Management Extension (RME) and switch to legacy mode. Width is 1-bit.

A.2.10 TBU ELA debug signals

The MMU S3 TBU includes Embedded Logic Analyzer (ELA) debug signals.

Signal definitions

Table A-22: ELA enable signals

Signal	Direction	Description
ela_enable	Input	The ela_enable signal is an asynchronous input port. When `TBUCFG_USE_ELA_DEBUG` is 0, the SMMU ignores the value of the signal. When `TBUCFG_USE_ELA_DEBUG` is 1, ela_enable acts as a clock enable for the TBU ELA observation interface. If ELA debug is required, drive ela_enable HIGH. If ELA debug is not required, drive ela_enable LOW to reduce the dynamic power consumption of the SMMU. Width is 1-bit.
signalgrp0..11	Output	Group observed signals. Width is `ELA_GRP_WIDTH` bit.
sigqual0..11	Output	Group qualification signal. Width is (`ELA_GRP_WIDTH`/32) bit.
sigclken0..11	Output	Group enable signal. Width is 1-bit.

A.2.11 TBU Dual signals

The interfaces of the TBU Dual are identical to the ACE-Lite TBU interfaces with some exceptions.

The following table shows the signals that differ between the TBU Dual interfaces and the ACE-Lite TBU interfaces.

Table A-23: Differences between TBU Dual interfaces and ACE-Lite TBU interfaces

Interface type	Dual TBU signal	TBUs that use the signal		R-TBU/W-TBU signal
		R-TBU	W-TBU	
Tie-off signals	ns_sid_high_r	Yes	No	ns_sid_high
	ns_sid_high_w	No	Yes	ns_sid_high
	s_sid_high_r	Yes	No	s_sid_high
	s_sid_high_w	No	Yes	s_sid_high
	r_sid_high_r	Yes	No	r_sid_high
	r_sid_high_w	No	Yes	r_sid_high
	max_tok_trans_r	Yes	No	max_tok_trans
	max_tok_trans_w	No	Yes	max_tok_trans
	sec_override	Yes	Yes	sec_override
	ecorevnum	Yes	Yes	ecorevnum
	utlb_roundrobin_r	Yes	No	utlb_roundrobin
	utlb_roundrobin_w	No	Yes	utlb_roundrobin
	pcie_mode	Yes	Yes	pcie_mode
	legacy_tz_en	Yes	Yes	legacy_tz_en
	same_power_domain	Yes	Yes	same_power_domain
	poison_support	Yes	Yes	poison_support
ELA interface signals	ela_enable_wtbu	No	Yes	ela_enable
	signalgrp0_wtbu	No	Yes	signalgrp0
	sigqual0_wtbu	No	Yes	sigqual0
	sigclken0_wtbu	No	Yes	sigclken0
	signalgrp1_wtbu	No	Yes	signalgrp1
	sigqual1_wtbu	No	Yes	sigqual1
	sigclken1_wtbu	No	Yes	sigclken1
	signalgrp2_wtbu	No	Yes	signalgrp2
	sigqual2_wtbu	No	Yes	sigqual2
	sigclken2_wtbu	No	Yes	sigclken2
	signalgrp3_wtbu	No	Yes	signalgrp3
	sigqual3_wtbu	No	Yes	sigqual3
	sigclken3_wtbu	No	Yes	sigclken3
	signalgrp4_wtbu	No	Yes	signalgrp4
	sigqual4_wtbu	No	Yes	sigqual4

Interface type	Dual TBU signal	TBUs that use the signal		R-TBU/W-TBU signal
		R-TBU	W-TBU	
	sigclken4_wtbu	No	Yes	sigclken4
	ela_enable_rtbu	Yes	No	ela_enable
	signalgrp0_rtbu	Yes	No	signalgrp1
	sigqual0_rtbu	Yes	No	sigqual1
	sigclken0_rtbu	Yes	No	sigclken1
	signalgrp1_rtbu	Yes	No	signalgrp2
	sigqual1_rtbu	Yes	No	sigqual2
	sigclken1_rtbu	Yes	No	sigclken2
	signalgrp2_rtbu	Yes	No	signalgrp3
	sigqual2_rtbu	Yes	No	sigqual3
	sigclken2_rtbu	Yes	No	sigclken3
	signalgrp3_rtbu	Yes	No	signalgrp4
	sigqual3_rtbu	Yes	No	sigqual4
	sigclken3_rtbu	Yes	No	sigclken4
	signalgrp4_rtbu	Yes	No	signalgrp5
	sigqual4_rtbu	Yes	No	sigqual5
	sigclken4_rtbu	Yes	No	sigclken5

A.3 TCU and TBU shared signals

Some signals that MMU S3 uses are shared by the TCU and TBU.

A.3.1 TCU and TBU test and debug signals

The test and debug signals are common to the TCU and TBU.

Signal definitions

Table A-24: Test and debug signals

Signal	Direction	Description
dftcgen	Input	Clock gate enable. To enable architectural clock gates for the clock, clk, set this signal HIGH during scan shift. Width is 1-bit.
dftrstdisable	Input	Reset disable. To disable reset, set this signal HIGH during scan shift. Width is 1-bit.
dftramhold	Input	Preserve RAM state. To preserve the state of the RAMs and their connected registers, set this signal HIGH during scan shift. Width is 1-bit.

Signal	Direction	Description
nMBISTRESET	Input	MBIST mode reset. This active-LOW signal is encoded as follows: **0** Reset MBIST functional logic. **1** Normal operation. To prevent unintended reset of the functional logic, keep the MBISTRESETN signal in the inactive state, HIGH, during scan testing. Width is 1-bit.
MBISTREQ	Input	MBIST test request. This signal is encoded as follows: **0** Normal operation. **1** Enable MBIST testing. Width is 1-bit.

A.3.2 PMU Snapshot signals

The PMU snapshot interface is common to both the TCU and TBU. The PMU snapshot interface enables you to capture the values of event counters at a specific times.

Event counters can count the following events:

- Number of transactions
- Number of cache look-ups
- Number of cache misses

The event counter values can provide useful information about the performance of your system. For a full list of countable performance events, see 2.5.2.1 SMMUv3 architectural performance events on page 57.

The *Key performance counter metrics* section in the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual* provides information about how you can use these performance events to obtain useful metrics for assessing the performance of your system.

To use the PMU snapshot interface, you can connect the pmusnapshot_req and pmusnapshot_ack signals to a CoreSight Cross Trigger Interface (CTI). Both signals are LOW after reset. When the pmusnapshot_req input to the TBU or TCU is set HIGH, the rising edge causes a snapshot to occur where all of the event counter values are simultaneously copied from the SMMU_PMCG_EVCNTRn event counter registers and saved to the SMMU_PMCG_SVRn shadow value registers.

Software can then read the counter values that are saved to the SMMU_PMCG_SVRn registers. The pmusnapshot_ack signal is used to acknowledge that the snapshot has occurred and that the event counter values have been saved.

Using the PMU snapshot interface to capture event counter values is optional.

An alternative approach is to use software to write a value of 1 to the CAPTURE bit of the SMMU_PMCG_CFGR register. The hardware and software approaches have the same effect and cause the event counter values to be captured and saved to the SMMU_PMCG_SVRn registers. If you do not use the PMU snapshot interface, then you can tie the pmusnapshot_req input signal to the TBU or TCU to 0. Leave the pmusnapshot_ack output signal unconnected

The ELA debug signals interface can initiate a PMU snapshot. See A.1.13 TCU ELA debug signals on page 316 and A.2.10 TBU ELA debug signals on page 333.

The PMU snapshot interface is asynchronous:

- pmusnapshot_req is sampled via synchronizing registers.
- pmusnapshot_ack is driven from a register.

Signal definitions

Table A-25: PMU snapshot signals

Signal	Direction	Description
pmusnapshot_req	Input	PMU snapshot request. The PMU snapshot occurs on the rising edge of pmusnapshot_req. **Note:** Connect to the debug infrastructure, for example, CoreSight, of your SoC. Width is 1-bit.
pmusnapshot_ack	Output	PMU snapshot acknowledge. The TCU and TBU use this signal to acknowledge that the PMU snapshot has occurred. This signal is LOW after reset. **Note:** Connect to the debug infrastructure, for example, CoreSight, of your SoC. Width is 1-bit.

A.4 DTI signals

MMU S3 uses DTI interconnect switch, sizer, and register slice signals.

A.4.1 DTI interconnect switch DN_S*n* interface signals

The MMU S3 DTI interconnect switch provides one DN_S*n* completer downstream interface per completer interface.

Signal definitions

Table A-26: DTI interconnect switch DN_S*n* interface signals

Signal	Direction	Description
tvalid_dti_dn_s*n*	Input	Flow control signal
tready_dti_dn_s*n*	Output	Flow control signal

Signal	Direction	Description
tdata_dti_dn_s*n*	Input	Message data signal
tid_dti_dn_s*n*	Input	Indicates the DTI manager that initiated the message
tlast_dti_dn_s*n*	Input	Indicates the last cycle of a message
tkeep_dti_dn_s*n*	Input	Indicates valid bytes
twakeup_dti_dn_s*n*	Input	Wakeup signal

A.4.2 DTI interconnect switch UP_S*n* interface signals

The MMU S3 DTI interconnect switch provides one UP_S*n* completer upstream interface per completer interface.

Signal definitions

Table A-27: DTI interconnect switch UP_S*n* interface signals

Signal	Direction	Description
tvalid_dti_up_s*n*	Output	Flow control signal
tready_dti_up_s*n*	Input	Flow control signal
tdata_dti_up_s*n*	Output	Message data signal
tdest_dti_up_s*n*	Output	Indicates the DTI manager that is the target of the message
tlast_dti_up_s*n*	Output	Indicates the last cycle of a message
tkeep_dti_up_s*n*	Output	Indicates valid bytes
twakeup_dti_up_s*n*	Output	Wakeup signal

A.4.3 DTI interconnect switch DN_M interface signals

The MMU S3 DTI interconnect switch provides a DN_M requester downstream interface.

Signal definitions

Table A-28: DTI interconnect switch DN_M interface signals

Signal	Direction	Description
tvalid_dti_dn_m	Output	Flow control signal
tready_dti_dn_m	Input	Flow control signal
tdata_dti_dn_m	Output	Message data signal
tid_dti_dn_m	Output	Indicates the DTI manager that initiated the message
tlast_dti_dn_m	Output	Indicates the last cycle of a message
tkeep_dti_dn_m	Output	Indicates valid bytes
twakeup_dti_dn_m	Output	Wakeup signal

A.4.4 DTI interconnect switch UP_M interface signals

The MMU S3 DTI interconnect switch provides an UP_M requester upstream interface.

Signal definitions

Table A-29: DTI interconnect switch UP_M interface signals

Signal	Direction	Description
tvalid_dti_up_m	Input	Flow control signal
tready_dti_up_m	Output	Flow control signal
tdata_dti_up_m	Input	Message data signal
tdest_dti_up_m	Input	Indicates the DTI manager that is the target of the message
tlast_dti_up_m	Input	Indicates the last cycle of a message
tkeep_dti_up_m	Input	Indicates valid bytes
twakeup_dti_up_m	Input	Wakeup signal

A.4.5 DTI interconnect sizer LPI_CG interface signals

The MMU S3 DTI interconnect sizer provides an LPI_CG clock gating interface.

Signal definitions

Table A-30: DTI interconnect sizer LPI_CG interface signals

Signal	Direction	Description
qactive_cg	Output	Component active
qreqn_cg	Input	Quiescence request
qacceptn_cg	Output	Quiescence accept
qdeny_cg	Output	Quiescence deny

A.4.6 DTI interconnect sizer DN_S interface signals

The MMU S3 DTI interconnect sizer provides a DN_S completer downstream interface.

Signal definitions

Table A-31: DTI interconnect sizer DN_S interface signals

Signal	Direction	Description
tvalid_dti_dn_s	Input	Flow control signal
tready_dti_dn_s	Output	Flow control signal
tdata_dti_dn_s	Input	Message data signal
tid_dti_dn_s	Input	Indicates the DTI manager that initiated the message
tlast_dti_dn_s	Input	Indicates the last cycle of a message
tkeep_dti_dn_s	Input	Indicates valid bytes
twakeup_dti_dn_s	Input	Wakeup signal

A.4.7 DTI interconnect sizer UP_S interface signals

The MMU S3 DTI interconnect sizer provides an UP_S completer upstream interface.

Signal definitions

Table A-32: DTI interconnect sizer UP_S interface signals

Signal	Direction	Description
tvalid_dti_up_s	Output	Flow control signal
tready_dti_up_s	Input	Flow control signal
tdata_dti_up_s	Output	Message data signal
tdest_dti_up_s	Output	Indicates the DTI manager that is the target of the message
tlast_dti_up_s	Output	Indicates the last cycle of a message
tkeep_dti_up_s	Output	Indicates valid bytes
twakeup_dti_up_s	Output	Wakeup signal

A.4.8 DTI interconnect sizer DN_M interface signals

The MMU S3 DTI interconnect sizer provides a DN_M requester downstream interface.

Signal definitions

Table A-33: DTI interconnect sizer DN_M interface signals

Signal	Direction	Description
tvalid_dti_dn_m	Output	Flow control signal
tready_dti_dn_m	Input	Flow control signal
tdata_dti_dn_m	Output	Message data signal
tid_dti_dn_m	Output	Indicates the DTI manager that initiated the message
tlast_dti_dn_m	Output	Indicates the last cycle of a message
tkeep_dti_dn_m	Output	Indicates valid bytes
twakeup_dti_dn_m	Output	Wakeup signal

A.4.9 DTI interconnect sizer UP_M interface signals

The MMU S3 DTI interconnect sizer provides an UP_M requester upstream interface.

Signal definitions

Table A-34: DTI interconnect sizer UP_M interface signals

Signal	Direction	Description
tvalid_dti_up_m	Input	Flow control signal
tready_dti_up_m	Output	Flow control signal
tdata_dti_up_m	Input	Message data signal

Signal	Direction	Description
tdest_dti_up_m	Input	Indicates the DTI manager that is the target of the message
tlast_dti_up_m	Input	Indicates the last cycle of a message
tkeep_dti_up_m	Input	Indicates valid bytes
twakeup_dti_up_m	Input	Wakeup signal

A.4.10 DTI interconnect register slice LPI_CG interface signals

The MMU S3 DTI interconnect register slice provides an LPI_CG clock gating interface.

Signal definitions

Table A-35: DTI interconnect register slice LPI_CG interface signals

Signal	Direction	Description
qactive_cg	Output	Component active
qreqn_cg	Input	Quiescence request
qacceptn_cg	Output	Quiescence accept
qdeny_cg	Output	Quiescence deny

A.4.11 DTI interconnect register slice DN_S interface signals

The MMU S3 DTI interconnect register slice provides a DN_S completer downstream interface.

Signal definitions

Table A-36: DTI interconnect register slice DN_S interface signals

Signal	Direction	Description
tvalid_dti_dn_s	Input	Flow control signal
tready_dti_dn_s	Output	Flow control signal
tdata_dti_dn_s	Input	Message data signal
tid_dti_dn_s	Input	Indicates the DTI manager that initiated the message
tlast_dti_dn_s	Input	Indicates the last cycle of a message
tkeep_dti_dn_s	Input	Indicates valid bytes
twakeup_dti_dn_s	Input	Wakeup signal

A.4.12 DTI interconnect register slice UP_S interface signals

The MMU S3 DTI interconnect register slice provides an UP_S completer upstream interface.

Signal definitions

Table A-37: DTI interconnect register slice UP_S interface signals

Signal	Direction	Description
tvalid_dti_up_s	Output	Flow control signal
tready_dti_up_s	Input	Flow control signal
tdata_dti_up_s	Output	Message data signal
tdest_dti_up_s	Output	Indicates the requester that initiated the message
tlast_dti_up_s	Output	Indicates the last cycle of a message
tkeep_dti_up_s	Output	Indicates valid bytes
twakeup_dti_up_s	Output	Wakeup signal

A.4.13 DTI interconnect register slice DN_M interface signals

The MMU S3 DTI interconnect register slice provides a DN_M requester downstream interface.

Signal definitions

Table A-38: DTI interconnect register slice DN_M interface signals

Signal	Direction	Description
tvalid_dti_dn_m	Output	Flow control signal
tready_dti_dn_m	Input	Flow control signal
tdata_dti_dn_m	Output	Message data signal
tid_dti_dn_m	Output	Indicates the DTI manager that initiated the message
tlast_dti_dn_m	Output	Indicates the last cycle of a message
tkeep_dti_dn_m	Output	Indicates valid bytes
twakeup_dti_dn_m	Output	Wakeup signal

A.4.14 DTI interconnect register slice UP_M interface signals

The MMU S3 DTI interconnect register slice provides an UP_M requester upstream interface.

Signal definitions

Table A-39: DTI interconnect register slice UP_M interface signals

Signal	Direction	Description
tvalid_dti_up_m	Input	Flow control signal
tready_dti_up_m	Output	Flow control signal
tdata_dti_up_m	Input	Message data signal
tdest_dti_up_m	Input	Indicates the requester that initiated the message

Signal	Direction	Description
tlast_dti_up_m	Input	Indicates the last cycle of a message
tkeep_dti_up_m	Input	Indicates valid bytes
twakeup_dti_up_m	Input	Wakeup signal

A.5 LTI signals

MMU S3 uses LTI request channel, response channel, and completion channel signals.

A.5.1 LTI request channel signals

The MMU S3 LTI signals include LTI request channel signals.

Signal definitions

Table A-40: LTI request channel signals

Signal	Direction	Category	Description
lavalid	Input	Transport	Channel valid. Width is 1-bit.
lavc	Input	Transport	VC number. This is fixed to 2 virtual channels so the width of the lavc signal is 1-bit.
lacredit	Output	Transport	Channel credit grant. The width of lacredit is fixed to 2-bit because we have 2 virtual channels.
laid	Input	Flow	Translation request ID. You can configure the width of this signal using the `TBUCFG_LTI_ID_WIDTH` configuration parameter. See 2.7.6 LTI TBU configuration parameters on page 125.
laogv	Input	Flow	Order group valid. Width is 1-bit.
laog	Input	Flow	Order group. You can configure the width of this signal using the `TBUCFG_LTI_OG_WIDTH` configuration parameter. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
laflow	Input	Context	Indicates the translation flow required. Width is 2-bit.
lammuv	Input	Context	Indicates whether SMMU translation is required. Width is 1-bit.
lasecsid	Input	Context	StreamID security level. The width is calculated as follows: LTI_GPC == True ? 2:1
lasid	Input	Context	StreamID. You can configure the width of this signal using the `TBUCFG_SID_WIDTH` configuration parameter. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
lassidv	Input	Context	SubstreamID valid. The width is calculated as follows: `LTI_SSID_WIDTH` > 0 ? 1:0
lassid	Input	Context	SubstreamID. You can configure the width of this signal using the `TBUCFG_SSID_WIDTH` configuration parameter. See 2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122.
laprot	Input	Transaction	Protection information. laprot uses the same encoding as the AXI AxPROT signals. The width is calculated as follows: LTI_MMU == True ? 3:1
lanse	Input	Transaction	Used in conjunction with laprot[1] to define the PAS of the transaction. Width is 1-bit.

Signal	Direction	Category	Description
laaddr	Input	Transaction	Address. The width is calculated as follows: LTI_MMU == True: 64-bit LTI_MMU == False: `LTI_LRADDR_WIDTH`-bit
latrans	Input	Transaction	Type of the transaction that the LTI request is translating. Width is 4-bit.
laattr	Input	Transaction	Attributes for the untranslated transaction. Width is 4-bit.
laloop	Input	Impdef	**IMPLEMENTATION DEFINED** loopback signaling. You can configure the width of this signal using the `TBUCFG_LTI_LOOP_WIDTH` configuration parameter. See 2.7.6 LTI TBU configuration parameters on page 125.
latlbloc	Input	Impdef	Location to access in the TLB. The meaning of this signal is **IMPLEMENTATION DEFINED**. When you add TBU_LTI interface signals, you must size the latlbloc signal. Calculate the width of latlbloc as follows: `LTI_TLBLOC_WIDTH` = (`LTI_TLBLOC_WIDTH_RAW` < 1) ? 1 : `LTI_TLBLOC_WIDTH_RAW` For information about how to set these parameters, see 2.7 Configuration parameters and methodology on page 120.
laident	Input	Context	Indicates an identity translation is required. Width is 1-bit.

For more detailed descriptions of these signals, see the *AMBA® LTI Protocol Specification*.

For detailed descriptions of the parameters that this section mentions, see the *Configuration parameters* section in the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

A.5.2 LTI response channel signals

The MMU S3 LTI signals include LTI response channel signals.

Signal definitions

Table A-41: LTI response channel signals

Signal	Direction	Category	Description
lrvalid	Output	Transport	Channel valid. Width is 1-bit.
lrvc	Output	Transport	VC number. This is fixed to 2 virtual channels so the lrvc signal is 1-bit wide.
lrcredit	Input	Transport	Channel credit grant. The width is fixed to 2 because we have 2 virtual channels.
lrid	Output	Flow	Translation request ID. You can configure the width of this signal using the `TBUCFG_LTI_ID_WIDTH` configuration parameter. See 2.7.6 LTI TBU configuration parameters on page 125.
lrctag	Output	Flow	Translation completion tag. Width is 1-bit.
lrresp	Output	Translation	Translation response. Width is 3-bit.
lrprot	Output	Translation	Translated protection information. lrprot uses the same encoding as laprot. Width is 3-bit.
lrnse	Output	Translation	Used in conjunction with lrprot[1] to define the PAS of the translation. This signal is always output as 0 when not in RME mode. Width is 1-bit.
lraddr	Output	Translation	Translated address. Width is `LTI_LRADDR_WIDTH`-bit.

Signal	Direction	Category	Description
lrattr	Output	Translation	Translated transaction attributes. Width is 4-bit.
lrhwattr	Output	Translation	**IMPLEMENTATION DEFINED** hardware attributes. See the *PBHA for translated transactions* section in the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*. Width is 4-bit.
lrmpam	Output	Translation	MPAM information. Width is calculated as follows: • When `LTI_MPAM_SUPPORT` is MPAM_9_1, width is (LTI_GPC == True ? 12 : 11) • When `LTI_MPAM_SUPPORT` is MPAM_12_1, width is (LTI_GPC == True ? 15 : 14)
lrmecid	Output	Translation	Memory Encryption Context Identifier (MECID). Width is `LTI_MECID_WIDTH`-bit.
lrloop	Output	Translation	Loopback signaling. You can configure the width of this signal using the `TBUCFG_LTI_LOOP_WIDTH` configuration parameter. See 2.7.6 LTI TBU configuration parameters on page 125.

For more detailed descriptions of these signals, see the *AMBA® LTI Protocol Specification*.

For detailed descriptions of the parameters that this section mentions, see the *Configuration parameters* section in the *Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual*.

A.5.3 LTI completion channel signals

The MMU S3 LTI signals include LTI completion channel signals.

Signal definitions

Table A-42: LTI completion channel signals

Signal	Direction	Category	Description
lcvalid	Input	Transport	Channel valid. Width is 1-bit.
lccredit	Output	Transport	Channel credit grant. Width is 1-bit.
lcctag	Input	Flow	Translation completion tag. Width is 1-bit.

For detailed descriptions of these signals, see the *AMBA® LTI Protocol Specification*.

Appendix B ELA signal descriptions

You can use the SIGNALGRP<n>, SIGQUAL<n>, and SIGCLKEN<n> signals of the TCU and TBU components to interface with an external Embedded Logic Analyzer (ELA).

The following folder contains machine-readable ELA mapping files:

```
mmu_s3/logical/testbench/ela_mapping
```

B.1 TCU observation interfaces

The TCU observation interfaces use the SIGNALGRP<n>, SIGQUAL<n>, and SIGCLKEN<n> signals, where <n> represents the number in the signal name, to interface to an external CoreSight™ ELA-600 Embedded Logic Analyzer.

Signal group output ports are present on each component. However, only a subset is used.

The SIGCLKEN<n> signal is set to 1 for the signal groups in the 'Enabled signal groups' column in the following table. Groups that are not enabled have their SIGCLKEN<n> signals set to 0. If ela_enable is driven LOW, all SIGCLKEN<n> signals are set to 0.

The following table shows the signal group output ports that are valid for the TCU x1 and TCU x2.

Table B-1: Number of signal groups per module for the TCU

Component	SIGNALGRP Width	_qtw observation signals	_ptw observation signals	Parameter	Enabled signal groups	Total
TCU x1	128	Yes	No	TCUCFG_QTW_DATA_WIDTH <= 128	0, 1, 2, 3, 4, 5, 6, 10	8
				TCUCFG_QTW_DATA_WIDTH == 256	0, 1, 2, 3, 4, 5, 6, 7, 10, 11	10
				TCUCFG_QTW_DATA_WIDTH == 512	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11	12
TCU x2	128	Yes	Yes	TCUCFG_QTW_DATA_WIDTH <= 128	0, 1, 2, 3, 4, 5, 6, 10	8
				TCUCFG_QTW_DATA_WIDTH == 256	0, 1, 2, 3, 4, 5, 6, 7, 10, 11	10
				TCUCFG_QTW_DATA_WIDTH == 512	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11	12

The following tables show the SIGNALGRP<n> bits for the signal groups of the TCU x1 and TCU x2.

Some buses, if configured to be larger than the 128-bit signal group width, are spread across multiple groups. MMU S3 delays sections of the signal by a cycle so that the ELA can sample 128-bit chunks of the data one cycle after another. The Number of cycles of delay column in the

table indicates the number of cycles, from when the signal is observable on an MMU S3 interface, to when the signal is observable on the ELA observation interface.

The TCU ELA observation groups are split into two sets of 12 signal groups:

_qtw

ELA observation signals with the _qtw suffix capture the DTI and the QTW interface signals. The *_qtw observation signals are present in TCU x1 and TCU x2.

_ptw

ELA observation signals with the _ptw suffix capture the PTW interface signals. The *_ptw observation signals are present only in TCU x2.

The following table shows the TCU x1 observation interface signals.

Table B-2: TCU x1 observation interface signals

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
0	[127: 0]	tdata_dti_dn[159:32]	{1'b0, (tready_dti_dn & tvalid_dti_dn) , tready_dti_dn, tvalid_dti_dn}	1
1	[127: 0]	tdata_dti_up[175:48]	{1'b0, (tready_dti_up & tvalid_dti_up) , tready_dti_up, tvalid_dti_up}	1
2	[127:100]	Unused	-	-
	[99: 68]	tdata_dti_dn[31:0]	{tvalid_dti_up, (tready_dti_up & tvalid_dti_up), tvalid_dti_dn, (tready_dti_dn & tvalid_dti_dn)}	0
	[67: 67]	tlast_dti_dn		
	[66: 66]	twakeup_dti_dn		
	[65: 65]	tready_dti_dn		
	[64: 64]	tvalid_dti_dn		
	[63: 58]	tid_dti_dn		
	[57: 10]	tdata_dti_up[47:0]		
	[9: 9]	tlast_dti_up		
	[8: 8]	twakeup_dti_up		
	[7: 7]	tready_dti_up		
	[6: 6]	tvalid_dti_up		
	[5: 0]	tdest_dti_up		
3	[127:126]	Unused	-	-
	[125:125]	rpoison_qtw	{rvalid_qtw, (rready_qtw & rvalid_qtw), arvalid_qtw, (arready_qtw & arvalid_qtw)}	1
	[124:124]	rlast_qtw		
	[123:123]	rready_qtw		
	[122:122]	rvalid_qtw		
	[121:110]	rid_qtw		
	[109: 94]	armecid_qtw		
	[93: 93]	armpam_qtw.pmg		
	[92: 83]	armpam_qtw.part_id[9:0]		
	[82: 81]	armpam_qtw.mpam_sp		

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
	[80: 79]	ardomain_qtw		
	[78: 75]	arpbha_qtw		
	[74: 71]	arqos_qtw		
	[70: 70]	arnse_qtw		
	[69: 69]	arprot_qtw		
	[68: 65]	arcache_qtw		
	[64: 62]	arsize_qtw		
	[61: 14]	araddr_qtw[50:3]		
	[13: 13]	arready_qtw		
	[12: 12]	arvalid_qtw		
	[11: 0]	arid_qtw		
4	[127:127]	crready_qtw	{1'b0, (crready_qtw & crvalid_qtw), (bready_qtw & bvalid_qtw), (awready_qtw & awvalid_qtw)}	1
	[126:126]	crvalid_qtw		
	[125:125]	bready_qtw		
	[124:124]	bvalid_qtw		
	[123:112]	bid_qtw		
	[111: 96]	awmecid_qtw		
	[95: 95]	awakeup_qtw		
	[94: 94]	awatop_qtw		
	[93: 93]	awmpam_qtw.pmg		
	[92: 83]	awmpam_qtw.part_id[9:0]		
	[82: 81]	awmpam_qtw.mpam_sp		
	[80: 79]	awdomain_qtw		
	[78: 75]	awpbha_qtw		
	[74: 71]	awqos_qtw		
	[70: 70]	awnse_qtw		
	[69: 69]	awprot_qtw		
	[68: 65]	awcache_qtw		
	[64: 62]	awsize_qtw		
	[61: 14]	awaddr_qtw[50:3]		
	[13: 13]	awready_qtw		
	[12: 12]	awvalid_qtw		
	[11: 0]	awid_qtw		
5	[127:127]	syscoack_qtw	{2'b00, (wready_qtw & wvalid_qtw), (acready_qtw & acvalid_qtw)}	1
	[126:126]	syscoreq_qtw		
	[125: 62]	wstrb_qtw		
	[61: 61]	wlast_qtw		
	[60: 60]	wready_qtw		
	[59: 59]	wvalid_qtw		
	[58: 58]	acwakeup_qtw		

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
	[57: 6]	acaddr_qtw		
	[5: 2]	acvmidext_qtw		
	[1: 1]	acready_qtw		
	[0: 0]	acvalid_qtw		
6	[127: 0]	rdata_qtw[127:0]	{1'b0, (rready_qtw & rvalid_qtw), rready_qtw, rvalid_qtw}	1
7	[127: 0]	rdata_qtw[255:128]	{1'b0, (rready_qtw & rvalid_qtw), rready_qtw, rvalid_qtw}	2
8	[127: 0]	rdata_qtw[383:256]	{1'b0, (rready_qtw & rvalid_qtw), rready_qtw, rvalid_qtw}	3
9	[127: 0]	rdata_qtw[511:384]	{1'b0, (rready_qtw & rvalid_qtw), rready_qtw, rvalid_qtw}	4
10	[127: 0]	wdata_qtw[127:0]	{1'b0, (wready_qtw & wvalid_qtw), wready_qtw, wvalid_qtw}	1
11	[127: 0]	wdata_qtw[255:128]	{1'b0, (wready_qtw & wvalid_qtw), wready_qtw, wvalid_qtw}	2

All the observation signals that are present for TCU x1 are also present for TCU x2. The following table shows additional secondary observation interface signals that are only present for TCU x2.

Table B-3: TCU x2 secondary observation interface signals

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
0	[127:0]	Unused	{4{1'b0}}	None
1	[127:0]	Unused	{4{1'b0}}	None
2	[127:0]	Unused	{4{1'b0}}	None
3	[127:126]	Unused	-	-
	[125:125]	rpoison_ptw	{rvalid_ptw, (rready_ptw & rvalid_ptw), arvalid_ptw, (arready_ptw & arvalid_ptw)}	1
	[124:124]	rlast_ptw		
	[123:123]	rready_ptw		
	[122:122]	rvalid_ptw		
	[121:110]	rid_ptw		
	[109: 94]	armecid_ptw		
	[93: 93]	armpam_ptw.pmg		
	[92: 83]	armpam_ptw.part_id[9:0]		
	[82: 81]	armpam_ptw.mpam_sp		
	[80: 79]	ardomain_ptw		
	[78: 75]	arpbha_ptw		
	[74: 71]	arqos_ptw		
	[70: 70]	arnse_ptw		
	[69: 69]	arprot_ptw		
	[68: 65]	arcache_ptw		
	[64: 62]	arsize_ptw		
	[61: 14]	araddr_ptw[50:3]		
	[13: 13]	arready_ptw		

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
	[12: 12]	arvalid_ptw		
	[11: 0]	arid_ptw		
4	[127:126]	Unused	-	-
	[125:125]	bready_ptw	{1'b0, 1'b0, (bready_ptw & bvalid_ptw), (awready_ptw & awvalid_ptw)}	1
	[124:124]	bvalid_ptw		
	[123:112]	bid_ptw		
	[111: 96]	awmecid_ptw		
	[95: 95]	awakeup_ptw		
	[94: 94]	awatop_ptw		
	[93: 93]	awmpam_ptw.pmg		
	[92: 83]	awmpam_ptw.part_id[9:0]		
	[82: 81]	awmpam_ptw.mpam_sp		
	[80: 79]	awdomain_ptw		
	[78: 75]	awpbha_ptw		
	[74: 71]	awqos_ptw		
	[70: 70]	awnse_ptw		
	[69: 69]	awprot_ptw		
	[68: 65]	awcache_ptw		
	[64: 62]	awsize_ptw		
	[61: 14]	awaddr_ptw[50:3]		
	[13: 13]	awready_ptw		
	[12: 12]	awvalid_ptw		
	[11: 0]	awid_ptw		
5	[127: 67]	Unused	-	-
	[66: 3]	wstrb_ptw	{2'b00, (wready_ptw & wvalid_ptw), 1'b0}	1
	[2: 2]	wlast_ptw		
	[1: 1]	wready_ptw		
	[0: 0]	wvalid_ptw		
6	[127: 0]	rdata_ptw[127:0]	{1'b0, (rready_ptw & rvalid_ptw), rready_ptw, rvalid_ptw}	1
7	[127: 0]	rdata_ptw[255:128]	{1'b0, (rready_ptw & rvalid_ptw), rready_ptw, rvalid_ptw}	2
8	[127: 0]	rdata_ptw[383:256]	{1'b0, (rready_ptw & rvalid_ptw), rready_ptw, rvalid_ptw}	3
9	[127: 0]	rdata_ptw[511:384]	{1'b0, (rready_ptw & rvalid_ptw), rready_ptw, rvalid_ptw}	4
10	[127: 0]	wdata_ptw[127:0]	{1'b0, (wready_ptw & wvalid_ptw), wready_ptw, wvalid_ptw}	1
11	[127: 0]	wdata_ptw[255:128]	{1'b0, (wready_ptw & wvalid_ptw), wready_ptw, wvalid_ptw}	2

B.2 ACE-Lite TBU observation interfaces

The ACE-Lite TBU observation interface uses the SIGNALGRP<n>, SIGQUAL<n>, and SIGCLKEN<n> signals, where <n> represents the number in the signal name, to interface to an external CoreSight™ ELA-600 Embedded Logic Analyzer.

Signal group output ports are present on each component. However, only a subset is used.

The SIGCLKEN<n> signal is set to 1 for the signal groups in the 'Enabled signal groups' column in the following table. Groups that are not enabled have their SIGCLKEN<n> signals set to 0. If ela_enable is driven LOW, all SIGCLKEN<n> signals are set to 0.

The following table shows the signal group output ports that are valid for the ACE-Lite TBU.

Table B-4: Number of signal groups per module for the ACE-Lite TBU

Component	Parameter	Enabled signal groups	Total
ACE-Lite TBU		0, 1, 2, 3, 4	5

The following table shows the SIGNALGRP<n> bits for the signal groups of the ACE-Lite TBU.

Some buses, if configured to be larger than the 128-bit signal group width, are spread across multiple groups. MMU S3 delays sections of the signal by a cycle so that the ELA can sample 128-bit chunks of the data one cycle after another. The *Number of cycles of delay* column in the table indicates the number of cycles, from when the signal is observable on an MMU S3 interface, to when the signal is observable on the ELA observation interface.

Table B-5: ACE-Lite TBU observation interface signals

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
0	[127:119]	Unused	-	-
	[118:117]	awtagop_m	{1'b0, (aw_ready_m & aw_valid_m), aw_ready_m, aw_valid_m}	1
	[116:113]	awpbha_m		
	[112:112]	awuser_m		
	[111:111]	awidunq_m		
	[110:105]	awatop_m		
	[104:103]	awdomain_m		
	[102:99]	awqos_m		
	[98:98]	awnse_m		
	[97:95]	awprot_m		
	[94:91]	awcache_m		
	[90:89]	awburst_m		
	[88:86]	awsize_m		
	[85:78]	awlen_m		
	[77:74]	awregion_m		
	[73:34]	awaddr_m		

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
	[33:33]	awready_m		
	[32:32]	awvalid_m		
	[31:0]	awid_m		
1	[127:120]	Unused	-	-
	[119:119]	awstashlpiden_m	{1'b0, (aw_ready_m & aw_valid_m), aw_ready_m, aw_valid_m}	2
	[118:114]	awstashlpid_m		
	[113:113]	awstashniden_m		
	[112:102]	awstashnid_m		
	[101:86]	awmecid_m		
	[85:85]	awmpam_m.pmg		
	[84:73]	awmpam_m.part_id		
	[72:71]	awmpam_m.mpam_sp		
	[70:47]	awmmusid_m		
	[46:45]	awmmusecsid_m		
	[44:44]	awmmuvalid_m		
	[43:34]	awloop_m		
	[33:33]	awready_m		
	[32:32]	awvalid_m		
	[31:0]	awid_m		
2	[127:113]	Unused	-	-
	[112:111]	artagop_m	{1'b0, (ar_ready_m & ar_valid_m), ar_ready_m, ar_valid_m}	1
	[110:107]	arpbha_m		
	[106:106]	aruser_m		
	[105:105]	aridunq_m		
	[104:103]	ardomain_m		
	[102:99]	arqos_m		
	[98:98]	arnse_m		
	[97:95]	arprot_m		
	[94:91]	arcache_m		
	[90:89]	arburst_m		
	[88:86]	arsize_m		
	[85:78]	arlen_m		
	[77:74]	arregion_m		
	[73:34]	araddr_m		
	[33:33]	arready_m		
	[32:32]	arvalid_m		
	[31:0]	arid_m		
3	[127:103]	Unused	-	-
	[102:87]	armecid_m	{1'b0, (ar_ready_m & ar_valid_m), ar_ready_m, ar_valid_m}	2
	[86:86]	archunken_m		

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
	[85:85]	armpam_m.pmg		
	[84:73]	armpam_m.part_id		
	[72:71]	armpam_m.mpam_sp		
	[70:47]	armmusid_m		
	[46:45]	armmusecsid_m		
	[44:44]	armmuvalid_m		
	[43:34]	arloop_m		
	[33:33]	arready_m		
	[32:32]	arvalid_m		
	[31:0]	arid_m		
4	[127:106]	Unused	-	-
	[105:98]	wpoison_m	{1'b0, (w_ready_m & w_valid_m), (b_ready_m & b_valid_m), (r_ready_m & r_valid_m)}	1
	[97:97]	wlast_m		
	[96:96]	wready_m		
	[95:95]	wvalid_m		
	[94:85]	bloop_m		
	[84:84]	bidunq_m		
	[83:82]	bresp_m		
	[81:81]	bready_m		
	[80:80]	bvalid_m		
	[79:48]	bid_m		
	[47:47]	ridunq_m		
	[46:37]	rloop_m		
	[36:36]	rlast_m		
	[35:34]	rresp_m		
	[33:33]	rready_m		
	[32:32]	rvalid_m		
	[31:0]	rid_m		

B.3 LTI TBU observation interfaces

The LTI TBU observation interface uses the SIGNALGRP<n>, SIGQUAL<n>, and SIGCLKEN<n> signals, where <n> represents the number in the signal name, to interface to an external CoreSight™ ELA-600 Embedded Logic Analyzer.

Signal group output ports are present on each component. However, only a subset is used.


Note

For multi-LTI port TBUs, the signals in groups 1-6 that this interface reports are after multiple LTI interfaces are multiplexed together. The signals captured reflect the transaction that the TBU is processing in that clock cycle, regardless of which LTI port is responsible for that transaction. The LTI port to which that transaction belongs is not captured.

The SIGCLKEN<n> signal is set to 1 for the signal groups in the 'Enabled signal groups' column in the following table. Groups that are not enabled have their SIGCLKEN<n> signals set to 0. If ela_enable is driven LOW, all SIGCLKEN<n> signals are set to 0.

The following table shows the signal group output ports that are valid for the LTI TBU.

Table B-6: Number of SignalGroups per module for the LTI TBU

Component	Parameter	Enabled signal groups	Total
LTI TBU	When `TBUCFG_LTI_LOOP_WIDTH` <= 128 bits	0, 1, 2, 3, 5, 7, 8	7
	When `TBUCFG_LTI_LOOP_WIDTH` > 128 bit	0, 1, 2, 3, 4, 5, 6, 7, 8	9

The following table shows the SIGNALGRP<n> bits for the signal groups of the LTI TBU.

Some buses, if configured to be larger than the 128-bit signal group width, are spread across multiple groups. MMU S3 delays sections of the signal by a cycle so that the ELA can sample 128-bit chunks of the data one cycle after another. The *Number of cycles of delay* column in the table indicates the number of cycles, from when the signal is observable on an MMU S3 interface, to when the signal is observable on the ELA observation interface.

Table B-7: LTI TBU observation interface signals

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> `4'b{MSB..LSB}`	Number of cycles of delay
0	[127:112]	Unused	-	-
	[111:96]	latlbloc	{`3'b000`, lavalid}	1
	[95:95]	laident		
	[94:93]	laflow		
	[92:89]	laattr		
	[88:85]	latrans		
	[84:33]	laaddr[51:0]		
	[32:32]	lavalid		
	[31:0]	laid		
1	[127:119]	Unused	-	-
	[118:114]	laog	{`3'b000`, lavalid}	1
	[113:113]	laogv		
	[112:93]	lassid		
	[92:92]	lassidv		
	[91:68]	lasid		
	[67:66]	lasecsid		

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
	[65:65]	lammuv		
	[64:61]	lavc		
	[60:57]	latrans		
	[56:56]	lanse		
	[55:53]	laprot		
	[52:1]	laaddr[51:0]		
	[0:0]	lavalid		
2	[127:112]	lrmecid	{3'b000, lrvalid}	1
	[111:111]	lrmpam.pmg		
	[110:101]	lrmpam.partid		
	[100:99]	lrmpam.mpam_sp		
	[98:95]	lrhwattr		
	[94:91]	lrattr		
	[90:45]	lraddr[45:0]		
	[44:44]	lrnse		
	[43:41]	lrprot		
	[40:38]	lrresp		
	[37:37]	lrctag		
	[36:33]	lrvc		
	[32:32]	lrvalid		
	[31:0]	lrid		
3	[127:0]	laloop[127:0]	{3'b000, lavalid}	1
4	[127:0]	laloop[255:128]	{3'b000, lavalid}	2
5	[127:0]	lrloop[127:0]	{3'b000, lrvalid}	1
6	[127:0]	lrloop[255:128]	{3'b000, lrvalid}	2
7	[127:16]	Unused	-	-
	[15:15]	lcctag_7	{3'b000, |lcvalid_m}	0
	[14:14]	lcvalid_7		
	[13:13]	lcctag_6		
	[12:12]	lcvalid_6		
	[11:11]	lcctag_5		
	[10:10]	lcvalid_5		
	[9:9]	lcctag_4		
	[8:8]	lcvalid_4		
	[7:7]	lcctag_3		
	[6:6]	lcvalid_3		
	[5:5]	lcctag_2		
	[4:4]	lcvalid_2		
	[3:3]	lcctag_1		
	[2:2]	lcvalid_1		

SIGNALGRP<n>	Bits	Signal name	SIGQUAL<n> 4'b{MSB..LSB}	Number of cycles of delay
	[1:1]	lcctag_0		
	[0:0]	lcvalid_0		
8	[127:32]	Unused	-	-
	[31:31]	lmaskclose_7	{1'b0, |lmopenreq, |lmopenack, |lmaskclose}	0
	[30:30]	lmactive_7		
	[29:29]	lmopenack_7		
	[28:28]	lmopenreq_7		
	[27:27]	lmaskclose_6		
	[26:26]	lmactive_6		
	[25:25]	lmopenack_6		
	[24:24]	lmopenreq_6		
	[23:23]	lmaskclose_5		
	[22:22]	lmactive_5		
	[21:21]	lmopenack_5		
	[20:20]	lmopenreq_5		
	[19:19]	lmaskclose_4		
	[18:18]	lmactive_4		
	[17:17]	lmopenack_4		
	[16:16]	lmopenreq_4		
	[15:15]	lmaskclose_3		
	[14:14]	lmactive_3		
	[13:13]	lmopenack_3		
	[12:12]	lmopenreq_3		
	[11:11]	lmaskclose_2		
	[10:10]	lmactive_2		
	[9:9]	lmopenack_2		
	[8:8]	lmopenreq_2		
	[7:7]	lmaskclose_1		
	[6:6]	lmactive_1		
	[5:5]	lmopenack_1		
	[4:4]	lmopenreq_1		
	[3:3]	lmaskclose_0		
	[2:2]	lmactive_0		
	[1:1]	lmopenack_0		
	[0:0]	lmopenreq_0		

Appendix C Software initialization examples

We provide examples of how software can initialize and enable MMU S3.

C.1 Initializing the SMMU

Software must initialize MMU S3 before you can use it.

MMU S3 supports Secure, Realm, and Non-secure translation worlds. We define how to initialize Non-secure translation. The procedures for initializing Secure and Realm translation are similar, and require you to access the corresponding MMU S3 Secure and Realm registers.


Note

We do not describe how to create translation tables. For more information, see the *Arm® Architecture Reference Manual for A-profile architecture*.

For more information about MMU S3 initialization, see the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

C.1.1 Initializing Granule Protection Checks

By default, the ACCESSEN control in SMMU_ROOT_CR0 disables all access. Before any transactions can pass through the SMMU, Granule Protection Checks (GPCs) must be configured.


Note

Initialization of the GPCs is not required if the SMMU is configured to operate in legacy mode by setting the `TBUCFG_LEGAZY_TZ_EN` and `TCUCFG_LEGACY_TZ_EN` parameters to 1, or by setting the legacy_tz_en tie-off signal to 1.

To configure GPCs, use the following procedure:

1. Configure the SMMU_ROOT_GPT_BASE register to point to the Physical Address of the L0 Granule Protection Table (GPT) tables
2. Configure SMMU_ROOT_GPT_BASE_CFG register with the appropriate settings for the GPT tables in the system, including setting the following:
   - The granule size for GPC, that is, PGS.
   - The shareability to use when accessing GPT, that is, SH.
   - The outer cacheability to use when accessing GPT, that is, ORGN.
   - The inner cacheability to use when accessing GPT, that is, IRGN.

- The range of memory protected by GPC, that is, PPS.

3. Write 1 to SMMU_S_INIT.INV_ALL and poll until the value returns to 0 to invalidate anything in the SMMU
4. Set SMMU_ROOT_CR0.GPCEN to 1
5. Poll SMMU_ROOT_CR0ACK until GPCEN is 1
6. Set SMMU_ROOT_CR0.ACCESSEN to 1
7. Poll SMMU_ROOT_CR0ACK until ACCESSEN is 1

C.1.2 Allocating the Command queue

MMU S3 uses the Command queue to receive commands. Software must allocate memory for the Command queue and configure the appropriate registers in the SMMU.

About this task

To allocate the Command queue, ensure that your software performs the following steps:

Procedure

1. Allocate memory for the Command queue.
2. Configure the Command queue size and base address by writing to the SMMU_CMDQ_BASE register.


Note

The queue size can affect how many bits of the SMMU_CMDQ_CONS and SMMU_CMDQ_PROD indices are writeable. It is therefore important that you perform this step before writing to SMMU_CMDQ_CONS and SMMU_CMDQ_PROD.

3. Set the queue read index in SMMU_CMDQ_CONS and the queue write index in SMMU_CMDQ_PROD to 0.


Note

Setting the queue read index and the queue write index to the same value indicates that the queue is empty.

The same process can be performed to enable the Realm and Secure event queues using the appropriate Realm or Secure equivalent registers.

C.1.3 Allocating the Event queue

MMU S3 uses the Event queue to signal events. Software must allocate memory for the Event queue and configure the appropriate registers in the MMU.

About this task

To allocate the Event queue, ensure that your software performs the following steps:

Procedure

1. Allocate memory for the Event queue.
2. Configure the Event queue size and base address by writing to the SMMU_EVENTQ_BASE register.


Note

The queue size can affect how many bits of the SMMU_EVENTQ_CONS and SMMU_EVENTQ_PROD indices are writeable. It is therefore important that you perform this step before writing to SMMU_EVENTQ_CONS and SMMU_EVENTQ_PROD.

3. Set the queue read index in SMMU_EVENTQ_CONS and the queue write index in SMMU_EVENTQ_PROD to 0.


Note

Setting the queue read index and the queue write index to the same value indicates that the queue is empty.

The same process can be performed to enable the Realm and Secure event queues using the appropriate Realm or Secure equivalent registers.

C.1.4 Allocating the PRI queue

The PRI queue (PRIQ) can be Non-secure or Realm. PRIQ cannot be Secure. To enable the PRIQ, use SMMU_(R_)CR0.PRIQEN.

If SMMU_(R_)CR0.SMMUEN is 0, then SMMU_(R_)CR0.PRIQEN is RES0

The PRIQ works in a similar way to C.1.3 Allocating the Event queue on page 358.

C.1.5 Configuring the Stream table

The Stream table is a configuration structure in memory that uses a Context Descriptor (CD) to locate translation data for a transaction. Software must allocate memory for the Stream table, configure the table format, and populate the table with Stream Table Entries (STEs).

About this task

To configure the Stream table, ensure that your software performs the following steps:

Procedure

1. Allocate memory for the Stream table.
2. Configure the format and size of the Stream table by writing to SMMU_STRTAB_BASE_CFG.
3. Configure the base address for the Stream table by writing to SMMU_STRTAB_BASE.
4. Prevent uninitialized memory being interpreted as a valid configuration by setting STE.V = 0 for each STE to mark it as invalid.

5. Ensure that written data is observable to the SMMU by performing a Data Synchronization Barrier (DSB) operation.
   If SMMU_IDR0.COHACC = 0, the system does not support coherent access to memory for the TCU. In such cases, you might require extra steps to ensure that the SMMU can observe the written data.

C.1.6 Initializing the Command queue

Software must initialize the Command queue by enabling it and checking that the enable operation is complete.

About this task

To initialize the Command queue, ensure that your software performs the following steps:

Procedure

1. Enable the Command queue by setting the SMMU_CR0.CMDQEN bit to 1.
2. Check that the enable operation is complete by polling SMMU_CR0ACK until CMDQEN reads as 1.

C.1.7 Initializing the Event queue

Software must initialize the Event queue by enabling it and checking that the enable operation is complete.

About this task

To initialize the Event queue, ensure that your software performs the following steps:

Procedure

1. Enable the Event queue by setting the SMMU_CR0.EVENTQEN bit to 1.
2. Check that the enable operation is complete by polling SMMU_CR0ACK until EVENTQEN reads as 1.

C.1.8 Invalidating TLBs and configuration caches

Before use, the MMU S3 TLBs and configuration cache structures must be invalidated by issuing commands to the Command queue. Alternatively, Secure software can invalidate all TLBs and caches with a single write.

To invalidate TLB entries, ensure that your software issues the appropriate command for the translation context.

To invalidate TLB entries for:

Non-secure EL1 contexts

Issue `CMD_TLBI_NSNH_ALL` to the appropriate command queue

Secure or Realm EL1 contexts

Issue CMD_TLBI_NH_ALL to the appropriate command queue

EL2 contexts

Issue CMD_TLBI_EL2_ALL to the appropriate command queue

EL3 contexts

Issue CMD_TLBI_EL3_ALL to the appropriate command queue, if in legacy mode


Note

In non-legacy mode, where RME is enabled, the EL3 stream world is not supported. The CMD_TLBI_EL3_ALL and CMD_TLBI_EL3_VA commands result in CERROR_ILL. The SMMU is not required to perform any invalidation on receipt of a broadcast TLBI for EL3.

GPC-only contexts


Note

Initialization of the GPCs is not required if the SMMU is configured to operate in legacy mode by setting the TBUCFG_LEGAZY_TZ_EN and TCUCFG_LEGACY_TZ_EN parameters to 1, or by setting the legacy_tz_en tie-off signal to 1.

1. Read SMMU_ROOT_TLBI_CTRL to ensure that the RUN field reads as 0.

   This indicates that no invalidates that this mechanism issues are outstanding.
2. Write 1 to the ALL field of SMMU_ROOT_TLBI and 0 to all other fields.

   This issues a PA ALL invalidate into the SMMU.
3. Poll SMMU_ROOT_TLBI_CTRL until the RUN field reads as 0.

   This indicates that the invalidate is complete.


Note

Commands to invalidate Secure TLB entries can only be issued through the Secure Command queue. For a system that implements two Security states, Secure software must issue the appropriate command to the Secure Command queue for the first TLB invalidation. If your system does not use Secure software, you can permit Non-secure software to access SMMU_S_INIT by using either sec_override or the 3.7.5 TCU_SCR register on page 161.

To invalidate both the TCU configuration cache and the TBU combined configuration cache and TLB, issue the CMD_CFGI_ALL command.

To force all previous commands to complete, issue CMD_SYNC.

To invalidate all configuration caches and TLB entries for all translation regimes and Security states, ensure that Secure software:

1. Sets SMMU_S_INIT.INV_ALL to 1. The SMMU sets SMMU_S_INIT.INV_ALL to 0 after the invalidation completes.
2. Polls SMMU_S_INIT.INV_ALL to check it is set to 0 before continuing the SMMU configuration.

For more information about issuing commands to the Command queue, see the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3*.

C.1.9 Creating a basic Context Descriptor

A Context Descriptor (CD) is a data structure in system memory. A CD defines how Stage 1 translation is performed. The SubstreamID is used to select the CD.

To create a CD, ensure that your software performs the following steps:

1. Allocate 64 bytes of memory for the CD.
2. Configure the CD fields according to the information in the following table.

Table C-1: Configuring the CD

Field	Description
AA64	Translation table format: **1** AArch64
EPD0	Enable translations for TTB0 by setting EPD0 to 0.
TTB0	Base address of translation table 0.
TG0	Translation granule size for TTB0 when CD.AA64 = 1.
IR0 OR0	Cacheability attribute to use for translation table walks to TTB0: **00** Non-cacheable. **01** Write-Back Cacheable, Read-Allocate Write-Allocate. **10** Write-through Cacheable, Read-Allocate.
SH0	Shareability of translation table walks to TTB0: **00** Non-shareable. **01** Outer Shareable. **10** Inner Shareable.
EPD1	If the StreamWorld supports split address spaces, enable table walks for TTB1.
ENDI	The endianness for the translation tables.
IPS	The IPA size when CD.AA64 = 1.
ASET	Defines whether the ASID values are shared with the ASID values of an Arm processor. **Note:** If you expect this context to receive broadcast TLB invalidation commands from a PE, set ASET to 0.
V	Valid CD. This field must be set to 1.

C.1.10 Creating a Stream Table Entry

Each Stream Table Entry (STE) configures how Stage 2 translation is performed, and how the Context Descriptor (CD) table can be found. The StreamID is used to select an STE.

To create an STE, ensure that your software performs the following steps:

1. Allocate 64 bytes of memory for the STE.
2. Set the STE.Config field as required for Stage 1 translation, Stage 2 translation, or translation bypass:

0b000	No traffic can pass through the MMU. An abort is returned.
0b100	Stage 1 and Stage 2 bypass.
0b101	Stage 1 translation Stage 2 bypass.
0b110	Stage 1 bypass Stage 2 translation.
0b111	Stage 1 and Stage 2 translation.

3. If Stage 1 translation is enabled, you can set the following fields:

STE.S1CDMax	Controls whether STE.S1ContextPtr points to a single CD or a CD table.
STE.S1Fmt	If STE.S1CDMax > 0, configures the format of the CD table.
STE.S1ContextPtr	Contains a pointer to either a CD or a CD table. If Stage 2 translation is enabled, this pointer is an intermediate physical address (IPA), otherwise it is an untranslated physical address PA.

4. If Stage 2 translation is enabled, you can set the following fields:

STE.S2TTB	Points to the Stage 2 translation table base address.
STE.S2PS	Contains the PA size of the stage 2 PA range.
STE.S2AA64	Indicates whether the Stage 2 tables are AArch32 or AArch64 format. MMU S3 supports only AArch64 format.
STE.S3ENDI	Set this field to the required endianness for the stage 2 translation tables.
STE.S2AFFD	Disable Access Flag faults for Stage 2 translation.
STE.S2TG	`0b00`: 4KB.
STE.S2IR0 and STE.S2OR0 STE.S2SH0	`0b00`: Non-cacheable.
STE.S2VMID	Contains the VMID associated with these translations.

C.2 Enabling the SMMU

Software can enable the SMMU by writing to SMMU_CR0 after the Stream table is populated.

About this task

To enable the SMMU, carry out the following procedure.

Procedure

1. Ensure that all Stream table entries are populated in memory.
2. Set the SMMU_CR0.SMMUEN bit to 1.
3. Check that the enable operation is complete by polling SMMU_CR0ACK until SMMUEN reads as 1.

Next steps

The same procedure can be used to enable the SMMU for Realm and Secure security levels, when the associated configuration of STE and CD has been performed and by using the appropriate Realm or Secure equivalent registers.

Proprietary Notice

This document is protected by copyright and other related rights and the use or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm Limited ("Arm"). No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether the subject matter of this document infringes any third party patents.

The content of this document is informational only. Any solutions presented herein are subject to changing conditions, information, scope, and data. This document was produced using reasonable efforts based on information available as of the date of issue of this document. The scope of information in this document may exceed that which Arm is required to provide, and such additional information is merely intended to further assist the recipient and does not represent Arm's view of the scope of its obligations. You acknowledge and agree that you possess the necessary expertise in system security and functional safety and that you shall be solely responsible for compliance with all legal, regulatory, safety and security related requirements concerning your products, notwithstanding any information or support that may be provided by Arm herein. In addition, you are responsible for any applications which are used in conjunction with any Arm technology described in this document, and to minimize risks, adequate design and operating safeguards should be provided for by you.

This document may include technical inaccuracies or typographical errors. THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, any patents, copyrights, trade secrets, trademarks, or other rights.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Reference by Arm to any third party's products or services within this document is not an express or implied approval or endorsement of the use thereof.

This document consists solely of commercial items. You shall be responsible for ensuring that any permitted use, duplication, or disclosure of this document complies fully with any relevant

export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of this document shall prevail.

The validity, construction and performance of this notice shall be governed by English Law.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

PRE-1121-V1.0

Product and document information

Read the information in these sections to understand the release status of the product and documentation, and the conventions used in the Arm documents.

Product status

All products and Services provided by Arm require deliverables to be prepared and made available at different levels of completeness. The information in this document indicates the appropriate level of completeness for the associated deliverables.

Product completeness status

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

Revision history

These sections can help you understand how the document has changed over time.

Document release information

The Document history table gives the issue number and the released date for each released issue of this document.

Document history

Issue	Date	Confidentiality	Change
0101-06	13 February 2025	Non-Confidential	First release for r1p1 EAC
0100-05	31 July 2024	Non-Confidential	First release for r1p0 EAC
0100-04-01	29 March 2024	Confidential - Draft	Draft DEV release for r1p0 EAC
0000-03	1 November 2023	Confidential	First release for r0p0 EAC
0000-02-02	31 August 2023	Confidential - Draft	Draft DEV release for r0p0 EAC
0000-02-01	27 April 2023	Confidential - Draft	Draft DEV release for r0p0 EAC
0000-01	21 October 2022	Confidential	First release for r0p0 BET

Note

The following releases were draft DEV releases:

- 0000-02-01
- 0000-02-02
- 0100-04-01

The Change history tables describe the technical changes between released issues of this document in reverse order. Issue numbers match the revision history in Document release information on page 367.

Table 2: Differences between issue 0100-05 and issue 0101-06

Change	Location
Improved descriptions	Throughout the document
Improved some feature descriptions and moved some features into different categories	1.2 Features on page 12
Corrected the descriptions of various caches	2.1 Translation Control Unit on page 22
Improved descriptions	• 2.2 Translation Buffer Unit on page 26 • 2.4.1.2 TCU Page Table Walk interface on page 32
Added more information	2.4.1.8 TCU SYSCO signaling on page 35
Added more information about the DTI versions in the *DTI connections the TCU accepts* section	2.5.1 Distributed Translation Interface on page 54
Corrected descriptions	• 2.5.2.2 SMMU TCU events on page 58 • 2.5.2.3 SMMU TBU events on page 64
Added to the description relating to the `TBUCFG_DIRECT_IDX` configuration parameter	2.5.4 Main TLB direct indexing and main TLB direct partitioning on page 69
Added a new *Interface used* column to the *TCU support for read transactions* table	2.5.8 TCU transaction handling on page 79
Changed various field descriptions	2.6.1.1 ID register architectural options on page 88
Added the following properties to the 'Issue J' section of the 'AXI property support' table: • REGION_Present • WLAST_Present • WSTRB_Present	2.6.2.1 ACE-Lite property support on page 94
Added a new *NoStreamID (AxMMUVALID = 0) transaction* subsection	2.6.2.4 Attribute handling on page 100
Updated some property descriptions	2.6.3 Local Translation Interface implementation on page 108
Added a new *MPAM and NoStreamID section*	2.6.4 MPAM implementation on page 110

Change	Location
Changed various field descriptions	• 2.6.4.1 TCU MPAM on page 110 • 2.6.4.2 TBU MPAM on page 115
Changed the description of the `TCUCFG_HZU_DEPTH` parameter	2.7.1 TCU I/O and buffer configuration parameters on page 120
Rearranged the order of some register summary categories so that everything appears in address offset order and there is no duplication with registers appearing in more than one summary table.	3. Programmers model for MMU S3 on page 129
Changed various field descriptions	3.6.5 PMU ID registers on page 152
• Corrected some values in the 'Realm register' column of the 'SCR and RCR override of register ownership' table • Improved the descriptions of the behavior of Secure and Non-secure accesses to the TCU_RCR and TCU_SCR registers.	3.7 TCU microarchitectural registers on page 153
Corrected various register bit field descriptions	• 3.7.2 TCU_QOS register on page 157 • 3.7.9 TCU_R_CTRL register on page 165
Corrected the address offsets	3.10 TCU_CTRL_AUX<n> registers on page 230
Corrected various register bit field descriptions	• 3.14 TBU component and peripheral ID registers on page 246 • 3.15.5 PMU ID registers on page 248
• Improved the descriptions of the behavior of Secure and Non-secure accesses to the TBU_RCR and TBU_SCR registers. • Added a table showing the 'SCR and RCR override of register ownership'.	3.16 TBU microarchitectural registers on page 249
Corrected various signal names and their descriptions	• A.1.3 TCU PTW interface signals on page 303 • A.1.4 TCU programming interface signals on page 306 • A.1.11 TCU event interface on page 312 • A.1.12 TCU tie-off signals on page 312
• Updated the descriptions of the arnse_s and awnse_s signals • Removed some information from the aruser_s and awuser_s signal descriptions and put them in the LATLBLOC signal description. • Made other small corrections and improvements	A.2.2 TBU TBS interface signals on page 317
Corrected descriptions and removed duplicate signals	A.2.3 TBU TBM interface signals on page 322
Corrected various signal names and their descriptions	A.2.9 TBU tie-off signals on page 331
Added new LTI signals section	A.5 LTI signals on page 343
Corrected the steps	C.1.1 Initializing Granule Protection Checks on page 357

Table 3: Differences between issue 0100-04-01 and issue 0100-05

Change	Location
Improved descriptions	Throughout the document
Simplified some feature descriptions	1.2 Features on page 12
Updated the descriptions to mention Granule Protection Checks (GPC) and Device Permission Table (DPT)	2. Functional description of MMU S3 on page 20
Improved the descriptions and modified the figure	2.1 Translation Control Unit on page 22
• Added new information • Improved descriptions • Changed structure of subsections	• 2.4 Interfaces on page 31 • 2.4.1.2 TCU Page Table Walk interface on page 32 • 2.4.1.6 TCU DTI interface on page 34 • 2.4.2 TBU interfaces on page 35
Updated some hyperlinks	2.4.2.7 TBU interrupt interface on page 41
Added a new *DTI connections the TCU accepts* subsection	2.5.1 Distributed Translation Interface on page 54
Added new events	• 2.5.2 Performance Monitoring Unit on page 56: ◦ 2.5.2.1 SMMUv3 architectural performance events on page 57 ◦ 2.5.2.2 SMMU TCU events on page 58 ◦ 2.5.2.3 SMMU TBU events on page 64
Improved descriptions	• 2.5.3 Multiple LTI interface TBU on page 68 • 2.5.4 Main TLB direct indexing and main TLB direct partitioning on page 69
• Added new information • Added a new figure • Improved the descriptions	• 2.5.5 RAS implementation on page 71 • 2.5.8 TCU transaction handling on page 79
Added a new section	2.5.13 NoStreamID on page 88

Change	Location
• Updated descriptions • Changed values of fields	2.6.1.1 ID register architectural options on page 88
• Aligned the list of features with the *AMBA® AXI Protocol Specification*. • Added a new column for the TCU to describe both ACE-Lite interfaces, QTW/DVM and PTW • Added a new table, *AXI width property descriptions*	2.6.2.1 ACE-Lite property support on page 94
Changed the layout and descriptions	2.6.2.4 Attribute handling on page 100
• Changed the layout • Corrected and improved the descriptions	• 2.6.4 MPAM implementation on page 110: ◦ 2.6.4.1 TCU MPAM on page 110 ◦ 2.6.4.2 TBU MPAM on page 115
• Added new parameters • Changed value ranges • Improved descriptions • Updated calculations in existing parameters	• 2.7.1 TCU I/O and buffer configuration parameters on page 120 • 2.7.4 ACE-Lite TBU register slice configuration parameters on page 124 • 2.7.6 LTI TBU configuration parameters on page 125 • 2.7.8 TBU Dual configuration parameters on page 126
Changed the layout and descriptions	3.3 SMMU memory map on page 137

Change	Location
• Updated the layout of sections • Changed some register descriptions	• 3.7 TCU microarchitectural registers on page 153 • 3.8 TCU RAS registers on page 175 • 3.9 TCU system discovery registers on page 187 • 3.11 TCU integration registers on page 231 • 3.12 TCU PIU integration registers on page 233 • 3.13 TCU TMU integration registers on page 243 • 3.16 TBU microarchitectural registers on page 249 • 3.17 TBU RAS registers on page 257 • 3.18 TBU system discovery registers on page 267 • 3.19 TBU integration registers on page 290
Updated some signal descriptions	• A.1.4 TCU programming interface signals on page 306 • A.1.5 TCU SYSCO interface signals on page 307 • A.1.8 TCU DTI interface signals on page 308 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312
• Improved descriptions of some signals. • Changed the names and descriptions of the following signals: ◦ sup_ns_gbpa_abort_rst ◦ sup_s_gbpa_abort_rst Now named as follows: ◦ ns_gbpa_abort_init ◦ s_gbpa_abort_init	A.1.12 TCU tie-off signals on page 312

Change	Location
Updated some signal descriptions	• A.2.2 TBU TBS interface signals on page 317 • A.2.8 TBU interrupt signals on page 330
Added a new *TBU Dual signals* section	A.2.11 TBU Dual signals on page 334
Moved the *PMU snapshot signals* section into the *TCU and TBU shared signals* section	• A.3 TCU and TBU shared signals on page 335 • A.3.2 PMU Snapshot signals on page 336
Updated the tables that show the observation interfaces for each module	• B.1 TCU observation interfaces on page 346 • B.2 ACE-Lite TBU observation interfaces on page 350 • B.3 LTI TBU observation interfaces on page 353
Updated the descriptions	• C.1.1 Initializing Granule Protection Checks on page 357 • C.1.8 Invalidating TLBs and configuration caches on page 360
Moved the Proprietary Notice, Product and document information, and Useful resources sections to the end of the document. These sections were previously at the beginning of the document.	• Proprietary Notice on page 365 • Product and document information on page 367: ◦ Product status on page 367 ◦ Revision history on page 367 ◦ Conventions on page 385 • Useful resources on page 387

Table 4: Differences between issue 0000-03 and issue 0100-04-01

Change	Location
Added information about LTI x6	Throughout the document
Improved descriptions	Throughout the document
Updated the Hit-Under-Miss (HUM) description	1.2 Features on page 12
Added information about the Device Permission Table (DPT) functionality	2.1 Translation Control Unit on page 22
Removed 'MPAM_Support' bullet	2.4.2.1 ACE-Lite TBU TBS interface on page 36

Change	Location
Added 'MPAM_12_1' bullet	2.4.2.2 ACE-Lite TBU TBM interface on page 37
Added new TCU events	2.5.2.2 SMMU TCU events on page 58
Added new RAM and non-RAM RAS error sources	2.5.5 RAS implementation on page 71
Changed the equation for the maximum number of read transactions that the TCU can issue	2.5.8 TCU transaction handling on page 79
Updated multiple descriptions and values	2.6.1.1 ID register architectural options on page 88
Removed 'SMMU_R_DPT_*'	2.6.1.4 Non-implemented registers on page 93
Made multiple changes to descriptions and values	2.6.3 Local Translation Interface implementation on page 108
Added information about the Device Permission Table (DPT) functionality	2.6.4 MPAM implementation on page 110
Made extensive changes to the description of TCU MPAM depending on whether Realm Management Extension (RME) mode is enabled or not, and whether DPT checks are enabled or not	2.6.4.1 TCU MPAM on page 110
Updated descriptions and values in the 'TCU MPAM registers implemented' table	
Updated descriptions and values in the 'TBU MPAM registers implemented' table	2.6.4.2 TBU MPAM on page 115
Added the following new parameters: • `TCUCFG_DPT_SUPPORT` • `TCUCFG_CC_LKP_SLOTS` • `TCUCFG_DC_DEPTH` • `TCUCFG_DC_BANKS` • `TCUCFG_DC_WAYS` • `TCUCFG_DC_LKP_SLOTS`	2.7.1 TCU I/O and buffer configuration parameters on page 120
Updated some parameter descriptions and values	
Updated some parameter values	2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122
Added a new `TBUCFG_LAFIFO_EXTENSION` parameter	2.7.6 LTI TBU configuration parameters on page 125
Moved 'TBU Dual configuration parameters' section and corrected descriptions of some parameters	2.7.8 TBU Dual configuration parameters on page 126
Updated the descriptions of the armpam_m, arnse_m, aruser_m, and awnse_m, signals	2.7.8 TBU Dual configuration parameters on page 126
Removed 'SMMU_R_DPT_*'	3. Programmers model for MMU S3 on page 129
Added new architectural registers: • SMMU_S_AGBPA, Secure Alternate Global ByPass Attribute • SMMU_AGBPA, Alternate Global ByPass Attribute	3.2 SMMU architectural registers on page 131
Added new DPT registers	

Change	Location
Added new registers: • TCU_ITEN_ET • TCU_R_CTRL • TCU_DC_L0_CMAX • TCU_DC_L1_CMAX	3.4.4 TCU microarchitectural registers summary on page 142: • 3.7.9 TCU_R_CTRL register on page 165 • 3.7.16 TCU_DC_L0_CMAX and TCU_DC_L1_CMAX registers on page 174 • 3.11.2 ITEN_ET register for the TCU on page 232
Added new TCU system discovery registers:	3.4.5 TCU system discovery registers summary on page 143
Added new TCU ITEN_ET register	3.4.4 TCU microarchitectural registers summary on page 142
Added new TBU ITEN_ET register	3.4.11 TBU integration registers summary on page 149
Added the following new registers to the list: • TCU_AGW_GC_L0_CMAX • TCU_AGW_GC_L1_CMAX • TCU_DGW_GC_L0_CMAX • TCU_DGW_GC_L1_CMAX • TCU_DC_L0_CMAX • TCU_DC_L1_CMAX • TCU_R_CTRL	3.7 TCU microarchitectural registers on page 153
Added new TCU_R_CTRL register	3.7.9 TCU_R_CTRL register on page 165
Added new 'TCU_DC_L0_CMAX and TCU_DC_L1_CMAX' section	3.7.16 TCU_DC_L0_CMAX and TCU_DC_L1_CMAX registers on page 174
Added and changed values for error codes	• 3.8.3 TCU_ERRSTATUS register on page 177 • 3.8.4 TCU_ERRGEN register on page 182
Changed bit field ranges for various TCU system discovery registers	3.9 TCU system discovery registers on page 187: • 3.9.12 TCU_SYSDISC11 system discovery register on page 198 • 3.9.27 TCU_SYSDISC26 system discovery register on page 213 • 3.9.28 TCU_SYSDISC27 system discovery register on page 214 • 3.9.29 TCU_SYSDISC28 system discovery register on page 215

Change	Location
Added new TCU system discovery registers	3.9 TCU system discovery registers on page 187: • 3.9.39 TCU_SYSDISC38 system discovery register on page 225 • 3.9.40 TCU_SYSDISC39 system discovery register on page 226 • 3.9.41 TCU_SYSDISC40 system discovery register on page 226 • 3.9.42 TCU_SYSDISC41 system discovery register on page 227 • 3.9.43 TCU_SYSDISC42 system discovery register on page 228 • 3.9.44 TCU_SYSDISC43 system discovery register on page 229
Updated descriptions	• 3.11.1 ITEN register for the TCU on page 231 • 3.11.2 ITEN_ET register for the TCU on page 232 • 3.12.1 ITOP_PIU register for the TCU Programmer Interface Unit on page 234 • 3.12.2 ITIN_PIU register for the TCU Programmer Interface Unit on page 242 • 3.13.1 ITOP_TMU register for the TCU Translation Management Unit on page 243 • 3.13.2 ITIN_TMU register for the TCU Translation Management Unit on page 245
Updated some values	3.15.5 PMU ID registers on page 248
Changed bit field ranges for various TBU system discovery registers	3.18 TBU system discovery registers on page 267: • 3.18.12 TBU_SYSDISC11 system discovery register on page 279 • 3.18.16 TBU_SYSDISC15 system discovery register on page 283 • 3.18.17 TBU_SYSDISC16 system discovery register on page 284 • 3.18.18 TBU_SYSDISC17 system discovery register on page 285
Added a new register	3.19.2 ITEN_ET register for the TBU on page 292
Updated descriptions and values	• 3.19.3 ITOP register for the TBU on page 293 • 3.19.4 ITIN register for the TBU on page 297

Change	Location
Updated the widths of some signals	• A.1.2 TCU QTW/DVM interface signals on page 300 • A.1.3 TCU PTW interface signals on page 303
Added the following signals: • sup_ns_gbpa_abort_rst • sup_s_gbpa_abort_rst Subsequently renamed as follows: • ns_gbpa_abort_init • s_gbpa_abort_init	A.1.12 TCU tie-off signals on page 312
Updated the descriptions of various signals	A.1.13 TCU ELA debug signals on page 316
Updated the descriptions of the arnse_s and awnse_s signals	A.2.2 TBU TBS interface signals on page 317
Added LTI x6	A.2.7 TBU LTI interface signals on page 330
Updated the observation interface signals	• B.1 TCU observation interfaces on page 346 • B.2 ACE-Lite TBU observation interfaces on page 350 • B.3 LTI TBU observation interfaces on page 353

Table 5: Differences between issue 0000-02-02 and issue 0000-03

Change	Location
Improved descriptions	Throughout the document
Updated sections to describe new functionality	• 1. About the MMU S3 System Memory Management Unit on page 11 • 1.2 Features on page 12 • 2.2 Translation Buffer Unit on page 26 • 2.1 Translation Control Unit on page 22 • 2.4.1 TCU interfaces on page 31 • 2.4.2 TBU interfaces on page 35
Added eventoreq or eventoack to event interface for TCU features	1.3 Interfaces on page 15
Updated the Number of Granule Protection Checks (GPCs) Updated the Type of some RAMs configurable option definitions	1.4 Configurable options on page 16
Updated the QTW/DVM and PTW interfaces definition	2.1 Translation Control Unit on page 22
Updated explanation of the external ID width calculation	2.4.1.2 TCU Page Table Walk interface on page 32
Added information about the Outer cacheable value for the aruser_m and awuser_m bits	2.5.12 AXI USER bits that the SMMU TBU TBM defines on page 87

Change	Location
Updated description of the SMMU_ROOT_IDR0 register	2.6.1.1 ID register architectural options on page 88
Updated Cache_Stash_Transactions interface properties for specification issue E Updated Read_Interleaving_Disabled interface property for specification issue H Updated RME_Support interface properties for specification issue J Renamed interface property Unstash_Translation to UnstashTranslation_Transaction for specification issue J Updated MEC_Support interface properties for specification issue K	2.6.2.1 ACE-Lite property support on page 94
Added the PTW interface because it also carries normalized attributes using the standard Cortex Armv8 scheme	2.6.2.4.2 Manager interface attribute handling on page 103
Updated transaction descriptions	• 2.6.2.8 Transaction types on page 105 • 2.6.2.9 Transactions that can result in a translation fault on page 106 • 2.6.2.10 Transactions that cannot result in a translation fault on page 107
Added new LTI property `LTI_SSID_WIDTH` Updated the `TBUCFG_MECID_WIDTH` values for `LTI_MEC_WIDTH`	2.6.3 Local Translation Interface implementation on page 108
Updated the registers MPAMF_IDR_LO and MPAMF_IDR_HI for `TCUCFG_LEGACY_TZ_EN` in SP4 and legacy mode Updated the MPAMCFG_CMAX value when resource is used to state you must configure PARTID_SEL before you attempt to update CMAX information for any PARTID Updated the MPAMF_AIDR field values ArchMajorRev and ArchMinorRev to 1 Updated the revision calculation in the register MPAMF_IIDR (0018, Shared)	2.6.4.1 TCU MPAM on page 110
Added calculation to MPAMF_CCAP_IDR CMAX_ID field Added more information about SP4 values to identify legacy and non-legacy mode Updated the revision calculation and values when resource is used and not used, for all fields in the MPAMF_IIDR `0x0018`, Shared) register	2.6.4.2 TBU MPAM on page 115
Updated the descriptions for the following parameters: • `TCUCFG_AGW_GC_LKP_SLOTS` • `TCUCFG_DGW_GC_LKP_SLOTS` • `TCUCFG_DTI_ATS_INV_MAX`	2.7.1 TCU I/O and buffer configuration parameters on page 120
Removed `TBUCFG_DATARAM_TYPE` parameter Updated the descriptions for the following parameters: • `TBUCFG_STASH_SUPPORT` • `TBUCFG_MTLB_LKP_SLOTS`	2.7.3 Common ACE-Lite and LTI TBU configuration parameters on page 122

Change	Location
Updated the lists of non-implemented registers	3. Programmers model for MMU S3 on page 129
Updated the descriptions for the following values in the table, TCU PMCG, RAS, and MPAM register allocation to regions of TCU address space: • `0x0029000 - 0x00297FC` • `0x0029800 - 0x002981C`	3.3.1 TCU memory map on page 138
Removed bullet point, Direct access to cache state	3.3.2 TBU memory map on page 140
Added new registers for new functionality	• 3. Programmers model for MMU S3 on page 129 • 3.4.4 TCU microarchitectural registers summary on page 142 • 3.4.11 TBU integration registers summary on page 149 • 3.10 TCU_CTRL_AUX<n> registers on page 230
Updated register descriptions	• 3.8.3 TCU_ERRSTATUS register on page 177 • 3.8.4 TCU_ERRGEN register on page 182 • 3.17.3 TBU_ERRSTATUS register on page 260 • 3.17.4 TBU_ERRGEN register on page 263

Change	Location
Updated signal descriptions	• A. Signal descriptions on page 300 • A.1.2 TCU QTW/DVM interface signals on page 300 • A.1.3 TCU PTW interface signals on page 303 • A.1.4 TCU programming interface signals on page 306 • A.1.8 TCU DTI interface signals on page 308 • A.1.9 TCU interrupt signals on page 309 • A.1.11 TCU event interface on page 312 • A.1.12 TCU tie-off signals on page 312 • A.1.13 TCU ELA debug signals on page 316 • A.2.2 TBU TBS interface signals on page 317 • A.2.3 TBU TBM interface signals on page 322 • A.2.6 TBU DTI interface signals on page 328 • A.2.9 TBU tie-off signals on page 331 • A.4.1 DTI interconnect switch DN_Sn interface signals on page 337 • B.1 TCU observation interfaces on page 346 • B.3 LTI TBU observation interfaces on page 353 • A.2.10 TBU ELA debug signals on page 333
Replaced the original DTI signals with an updated set of signals	A.4 DTI signals on page 337

Table 6: Differences between issue 0000-02-01 and issue 0000-02-02

Change	Location
Improved descriptions	Throughout the document
Changed order and categorization of some features	1.2 Features on page 12

Change	Location
Updated descriptions	• 2. Functional description of MMU S3 on page 20 • 2.2 Translation Buffer Unit on page 26 • 2.1 Translation Control Unit on page 22 • 2.4.1 TCU interfaces on page 31 • 2.4.1.1 TCU Queue and Table Walk/ Distributed Virtual Memory interface on page 32 • 2.4.1.2 TCU Page Table Walk interface on page 32 • 2.4.1.3 TCU PROG interface on page 33 • 2.4.1.6 TCU DTI interface on page 34 • 2.4.2.1 ACE-Lite TBU TBS interface on page 36 • 2.4.2.2 ACE-Lite TBU TBM interface on page 37 • 2.4.2.6 TBU DTI interface on page 40 • 2.4.3 Dual TBU on page 41
Updated the width values for some Dual TBU DTI interface signals, including equations to calculate the width of some signals	2.4.3.4.6 DTI interface on page 46
Updated descriptions	2.5.1 Distributed Translation Interface on page 54
Added new events	• 2.5.2.2 SMMU TCU events on page 58 • 2.5.2.3 SMMU TBU events on page 64
Updated descriptions	• 2.5.8 TCU transaction handling on page 79 • 2.5.10 Error responses on page 84
Added some extra conditions to some fields	2.6.1.1 ID register architectural options on page 88
Changed whether or not some features are supported for some components	2.6.2.1 ACE-Lite property support on page 94
Updated descriptions	• 2.6.2.6 DVM interface on page 105 • 2.6.2.10 Transactions that cannot result in a translation fault on page 107
Changed descriptions and values	• 2.6.4.1 TCU MPAM on page 110 • 2.6.4.2 TBU MPAM on page 115
• Add parameters • Removed parameters • Changed parameter values	2.7 Configuration parameters and methodology on page 120

Change	Location
Updated descriptions	• 3.3 SMMU memory map on page 137 • 3.3.1 TCU memory map on page 138 • 3.3.2 TBU memory map on page 140 • 3.4.4 TCU microarchitectural registers summary on page 142 • 3.6.2 Events on page 151 • 3.7 TCU microarchitectural registers on page 153
Updated register descriptions	• 3.7.1 TCU_CTRL register on page 156 • 3.7.7 TCU_ROOT_CTRL register on page 163 • 3.8.3 TCU_ERRSTATUS register on page 177 • 3.8.4 TCU_ERRGEN register on page 182 • 3.12.1 ITOP_PIU register for the TCU Programmer Interface Unit on page 234 • 3.12.2 ITIN_PIU register for the TCU Programmer Interface Unit on page 242 • 3.13.1 ITOP_TMU register for the TCU Translation Management Unit on page 243
Updated descriptions	3.15.2 Events on page 247
Updated register descriptions	• 3.17.3 TBU_ERRSTATUS register on page 260 • 3.17.4 TBU_ERRGEN register on page 263 • 3.18.5 TBU_SYSDISC4 system discovery register on page 272 • 3.19.3 ITOP register for the TBU on page 293

Change	Location
• Updated signal descriptions • Added descriptions for signals that did not have descriptions • Changed width values, sometimes adding equations to calculate the width	• A.1.2 TCU QTW/DVM interface signals on page 300 • A.1.3 TCU PTW interface signals on page 303 • A.1.8 TCU DTI interface signals on page 308 • A.1.9 TCU interrupt signals on page 309 • A.1.12 TCU tie-off signals on page 312 • A.1.13 TCU ELA debug signals on page 316 • A.2.2 TBU TBS interface signals on page 317 • A.2.3 TBU TBM interface signals on page 322 • A.2.6 TBU DTI interface signals on page 328 • A.4.1 DTI interconnect switch DN_Sn interface signals on page 337
Changed some signal names and ranges	B. ELA signal descriptions on page 346
Added information to describe usage with Realm and Secure command queues	• C.1.3 Allocating the Event queue on page 358 • C.1.4 Allocating the PRI queue on page 359 • C.2 Enabling the SMMU on page 363

Table 7: Differences between issue 0000-01 and issue 0000-02-01

Change	Location
Removed references to *The Realm Management Extension (RME), for SMMUv3 Arm® System Memory Management Unit Architecture Supplement* because the *Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3* now contains that information.	Throughout the document
Add a *Support for Realm Management Extension Device Assignment* section	1.2 Features on page 12
Added new information	2. Functional description of MMU S3 on page 20
• Corrected the figure • Added a new *TCU PTW interface section* • Updated descriptions	2.4.1 TCU interfaces on page 31
Corrected the LTI TBU interfaces figure	2.4.2 TBU interfaces on page 35
• Added parameters • Renamed parameters • Changed parameter values for DTI interface • Renamed some interrupt signals	2.4.3 Dual TBU on page 41

Change	Location
Added new events	• 2.5.2.2 SMMU TCU events on page 58 • 2.5.2.3 SMMU TBU events on page 64
Added RAS RAM error sources to the tables	2.5.5 RAS implementation on page 71
Changed descriptions and values of registers and conditions under which they operate	2.6.1.1 ID register architectural options on page 88
Added a new section for Issue K to reflect the latest version of the *AMBA® AXI Protocol Specification*	2.6.2.1 ACE-Lite property support on page 94
Added conditions to cater for Root and Realm and changed some setting values	• 2.6.4.1 TCU MPAM on page 110 • 2.6.4.2 TBU MPAM on page 115
• Add parameters • Removed parameters • Changed parameter values	2.7 Configuration parameters and methodology on page 120
• Added registers • Renamed registers • Changed register values • Updated descriptions and added information to sections related to programming registers • Corrected register offset values • Changed some register types • Changed some individual register bit field ranges and descriptions	3. Programmers model for MMU S3 on page 129
• Added new signals • Changed some signal directions • Changed some signal widths	A. Signal descriptions on page 300
Updated the observation interface signals details for the following: • TCU x1 • TCU x2 • ACE-Lite TBU • LTI TBU	B. ELA signal descriptions on page 346
• Added a new C.1.1 Initializing Granule Protection Checks on page 357 subsection • Updated descriptions in some subsections • Added information in some subsections to mention Realm and Realm Management Extension (RME)	C. Software initialization examples on page 357

Table 8: Issue 0000-01

Change	Location
First release	-

Conventions

The following subsections describe conventions used in Arm documents.

Glossary

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

See the Arm Glossary for more information: developer.arm.com/glossary.

Typographic conventions

Arm documentation uses typographical conventions to convey specific meaning.

Convention	Use
italic	Citations.
bold	Interface elements, such as menu names. Terms in descriptive lists, where appropriate.
`monospace`	Text that you can enter at the keyboard, such as commands, file and program names, and source code.
`monospace underline`	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
`<and>`	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: `MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>`
SMALL CAPITALS	Terms that have specific technical meanings as defined in the *Arm® Glossary*. For example, **IMPLEMENTATION DEFINED**, **IMPLEMENTATION SPECIFIC**, **UNKNOWN**, and **UNPREDICTABLE**.

We recommend the following. If you do not follow these recommendations your system might not work.

Your system requires the following. If you do not follow these requirements your system will not work.

You are at risk of causing permanent damage to your system or your equipment, or of harming yourself.

This information is important and needs your attention.

This information might help you perform a task in an easier, better, or faster way.

This information reminds you of something important relating to the current content.

Useful resources

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

Access to Arm documents depends on their confidentiality:

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

Arm product resources	Document ID	Confidentiality
Arm® Neoverse™ MMU S3 System Memory Management Unit Configuration and Integration Manual	102755	Confidential
Arm® Neoverse™ MMU S3 System Memory Management Unit Release Note	107743	Confidential
Arm® CoreLink™ ADB-400 AMBA® Domain Bridge User Guide	DUI 0615	Confidential
Arm® CoreLink™ LPD-500 Low Power Distributor Technical Reference Manual	100361	Non-Confidential
Arm® CoreSight™ System-on-Chip SoC-600 Technical Reference Manual	100806	Non-Confidential
Arm® CoreSight™ ELA-600 Embedded Logic Analyzer Technical Reference Manual	101088	Non-Confidential
Arm® Neoverse™ CMN-700 Coherent Mesh Network Technical Reference Manual	102308	Non-Confidential

Arm architecture and specifications	Document ID	Confidentiality
Arm® System Memory Management Unit Architecture Specification - SMMU architecture version 3	IHI 0070G.a	Non-Confidential
Learn the Architecture - SMMU Software Guide	109242	Non-Confidential
AMBA® DTI Protocol Specification	IHI 0088G	Non-Confidential
AMBA® LTI Protocol Specification	IHI 0089C	Non-Confidential
AMBA® Low Power Interface Specification	IHI 0068D	Non-Confidential
Arm® Architecture Reference Manual for A-profile architecture	DDI 0487K.a	Non-Confidential
Arm® Memory System Resource Partitioning and Monitoring (MPAM) System Component Specification	IHI 0099A.a	Non-Confidential
AMBA® AXI Protocol Specification	IHI 0022K	Non-Confidential
AMBA® AXI-Stream Protocol Specification	IHI 0051B	Non-Confidential
AMBA® APB Protocol Specification	IHI 0024E	Non-Confidential
Arm® Server Base System Architecture 7.1	DEN 0029H	Non-Confidential
GIC MSI Delivery Interface	AES 0019A	Confidential