



# Arm<sup>®</sup> System Memory Management Unit Architecture Supplement - The Realm Management Extension (RME), for SMMUv3

Document number	ARM IHI 0094
Document quality	EAC
Document version	A.c
Document confidentiality	Non-confidential

*Copyright © 2021-2022 Arm Limited or its affiliates. All rights reserved.*

# Arm® System Memory Management Unit Architecture Supplement - The Realm Management Extension (RME), for SMMUv3

## Release information

Date	Version	Changes
2022/Feb/07	A.c	<ul style="list-style-type: none"><li>• Updated EAC release.</li></ul>
2021/Nov/02	A.b	<ul style="list-style-type: none"><li>• Updated EAC release.</li></ul>
2021/Jun/23	A.a	<ul style="list-style-type: none"><li>• First EAC publication.</li></ul>

## Non-Confidential Proprietary Notice

This document is protected by copyright and other related rights and the practice 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. **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 implementations infringe any third party patents.

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, patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

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.

This document consists solely of commercial items. You shall be responsible for ensuring that any 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 the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its affiliates) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm’s trademark usage guidelines at <http://www.arm.com/company/policies/trademarks>.

Copyright © 2021-2022 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349 version 21.0

## Contents

# Arm® System Memory Management Unit Architecture Supplement - The Realm Management Extension (RME), for SMMUv3

Arm® System Memory Management Unit Architecture Supplement - The Realm Management Extension (RME), for SMMUv3 . . . . .	ii
Release information . . . . .	ii
Non-Confidential Proprietary Notice . . . . .	iii
Conventions . . . . .	vi
Typographical conventions . . . . .	vi
Numbers . . . . .	vi
Pseudocode descriptions . . . . .	vi
Assembler syntax descriptions . . . . .	vi
Additional reading . . . . .	vii
Feedback . . . . .	viii
Feedback on this book . . . . .	viii
Progressive Terminology statement . . . . .	viii

## Chapter 1

### Introduction

1.1 Discovery . . . . .	10
-------------------------	----

## Chapter 2

### Behavior

2.1 Granule protection checks . . . . .	11
2.2 Removal of EL3 StreamWorld . . . . .	12
2.3 Client-originated accesses . . . . .	13
2.3.1 GPC for client devices without a StreamID . . . . .	13
2.3.2 Speculative and hint accesses . . . . .	14
2.4 Interactions with PCIe ATS . . . . .	15
2.5 Interactions with ATOS interfaces . . . . .	16
2.6 SMMU-originated accesses . . . . .	17
2.7 Reporting of GPC faults . . . . .	18
2.8 SMMU behavior if a GPC fault is active . . . . .	19
2.9 Broadcast TLBI interface . . . . .	20
2.10 Observability of GPC faults . . . . .	21
2.11 New registers . . . . .	22
2.11.1 Root Control Page . . . . .	22
2.11.2 SMMU_ROOT_IDR0 . . . . .	23
2.11.3 SMMU_ROOT_IIDR . . . . .	25
2.11.4 SMMU_ROOT_CR0 . . . . .	26
2.11.5 SMMU_ROOT_CR0ACK . . . . .	29
2.11.6 SMMU_ROOT_GPT_BASE . . . . .	30
2.11.7 SMMU_ROOT_GPT_BASE_CFG . . . . .	32
2.11.8 SMMU_ROOT_GPF_FAR . . . . .	36
2.11.9 SMMU_ROOT_GPT_CFG_FAR . . . . .	39
2.11.10 SMMU_ROOT_TLBI . . . . .	43
2.11.11 SMMU_ROOT_TLBI_CTRL . . . . .	46
2.11.12 IMPLEMENTATION DEFINED registers . . . . .	47

## Chapter 3

### Register changes

3.1 SMMU_IDR0 . . . . .	49
-------------------------	----

Contents  
Contents

3.2	SMMU_S_IDR1 . . . . .	50
3.3	SMMU_S_INIT . . . . .	51
3.4	Accessibility of SMMU register pages . . . . .	52
3.5	Confidential information in RAS Error Records . . . . .	53
3.6	Interaction with PMCG events . . . . .	54

**Chapter 4**

**Event record changes**

4.1	F_STE_FETCH, F_CD_FETCH, F_VMS_FETCH and F_WALK_EABT . . . . .	56
-----	--	----

**Glossary**

# Conventions

## Typographical conventions

The typographical conventions are:

*italic*

Introduces special terminology, and denotes citations.

**bold**

Denotes signal names, and is used for terms in descriptive lists, where appropriate.

`monospace`

Used for assembler syntax descriptions, pseudocode, and source code examples.

Also used in the main text for instruction mnemonics and for references to other items appearing in assembler syntax descriptions, pseudocode, and source code examples.

SMALL CAPITALS

Used for some common terms such as IMPLEMENTATION DEFINED.

Used for a few terms that have specific technical meanings, and are included in the Glossary.

Red text

Indicates an open issue.

Blue text

Indicates a link. This can be

- A cross-reference to another location within the document
- A URL, for example <http://developer.arm.com>

## Numbers

Numbers are normally written in decimal. Binary numbers are preceded by 0b, and hexadecimal numbers by 0x. In both cases, the prefix and the associated value are written in a monospace font, for example 0xFFFF0000. To improve readability, long numbers can be written with an underscore separator between every four characters, for example 0xFFFF\_0000\_0000\_0000. Ignore any underscores when interpreting the value of a number.

## Pseudocode descriptions

This book uses a form of pseudocode to provide precise descriptions of the specified functionality. This pseudocode is written in a monospace font. The pseudocode language is described in the Arm Architecture Reference Manual.

## Assembler syntax descriptions

This book contains numerous syntax descriptions for assembler instructions and for components of assembler instructions. These are shown in a `monospace` font.

## Additional reading

This section lists publications by Arm and by third parties.

See Arm Developer (<http://developer.arm.com>) for access to Arm documentation.

[1] *Arm® Architecture Reference Manual Supplement, The Realm Management Extension (RME), for Armv9-A.* (ARM DDI 0615) Arm Ltd.

[2] *Arm® System Memory Management Unit Architecture Specification, SMMU architecture version 3.* (ARM IHI 0070) Arm Ltd.

[3] *Arm<sup>&reg</sup>; Architecture Reference Manual Supplement, Memory System Resource Partitioning and Monitoring (MPAM), for A-profile architecture.* (ARM DDI 0598) Arm Ltd.

[4] *Arm® Reliability, Availability, and Serviceability (RAS) Specification.* (ARM DDI 0587) Arm Ltd.

# Feedback

Arm welcomes feedback on its documentation.

## Feedback on this book

If you have comments on the content of this book, send an e-mail to [errata@arm.com](mailto:errata@arm.com). Give:

- The title (Arm® System Memory Management Unit Architecture Supplement - The Realm Management Extension (RME), for SMMUv3).
- The number (ARM IHI 0094 A.c).
- The page numbers to which your comments apply.
- The rule identifiers to which your comments apply, if applicable.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

---

### Note

Arm tests PDFs only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the appearance or behavior of any document when viewed with any other PDF reader.

---

## Progressive Terminology statement

We believe that this document contains no offensive terms. If you find offensive terms in this document, please contact [terms@arm.com](mailto:terms@arm.com).



# Chapter 1

## Introduction

This document specifies the behavior of new SMMU features to support features introduced in the Realm Management Extension [1]. This includes:

- Applying granule protection checks for all client-originated accesses.
- Applying granule protection checks for SMMU-originated accesses.
- An interface for the Root Security state to configure the *Granule Protection Table* (GPT) and granule protection checks.

The behavior that is specified in this document is in addition to the SMMUv3 behaviors in the *Arm® System Memory Management Unit Architecture Specification*. [2].

The PE behaviors specified in the Realm Management Extension [1] are referred to as FEAT\_RME.

## 1.1 Discovery

There are two aspects to RME support for SMMU:

- Whether the SMMU has the Root programming interface and can perform granule protection checks. This is advertised with `SMMU_ROOT_IDR0.ROOT_IMPL == 1`.
- Whether the SMMU has RME-related changes exposed to the Secure and Non-secure programming interfaces. This is advertised with `SMMU_IDR0.RME_IMPL == 1`.

The new SMMU behaviors specified in this document are referred to as applying to an “SMMU with RME”.

An SMMU with RME must have `SMMU_ROOT_IDR0.ROOT_IMPL == 1`. It is permitted for an SMMU with RME to have `SMMU_IDR0.RME_IMPL == 0`.

See also:

- [2.11.2 SMMU\\_ROOT\\_IDR0](#)
- [3.2 SMMU\\_S\\_IDR1](#)

## Chapter 2

### Behavior

#### 2.1 Granule protection checks

The Realm Management Extension, FEAT\_RME, [1] specifies the behavior of granule protection checks. An SMMU with RME performs the same checks for non-PE Requesters.

The format and meaning of the GPT is the same in an SMMU with RME as it is in FEAT\_RME.

The invalidation and synchronization mechanisms for updates to the GPT are the same as for FEAT\_RME.

The SMMU configuration registers for the GPT base address and format are equivalent to those in FEAT\_RME.

Any GPT access is made with the MPAM attributes of the client- or SMMU-originated access that caused the GPT access. When a GPT access is made to check the output address of a translation request, the GPT access uses the MPAM attributes that would be used for an equivalent Untranslated transaction for the same StreamID and SubstreamID.

## 2.2 Removal of EL3 StreamWorld

An SMMU with SMMU\_IDR0.RME\_IMPL == 1 does not support the EL3 StreamWorld. This means that:

- An STE with STRW configured for EL3 is ILLEGAL and results in C\_BAD\_STE.
- The commands CMD\_TLBI\_EL3\_ALL, CMD\_TLBI\_EL3\_VA result in CERROR\_ILL.
- The SMMU is not required to perform any invalidation on receipt of a broadcast TLBI for EL3.

All Secure state specified in SMMUv3 applies only to Secure-EL2 and below.

See also:

- [3.1 SMMU\\_IDR0](#)
- [3.2 SMMU\\_S\\_IDR1](#)

## 2.3 Client-originated accesses

In the same manner as SMMUv3, SEC\_SID identifies only Secure and Non-secure client devices. There are no Root or Realm client devices.

Accesses from client devices undergo translation according to the configuration registers, configuration tables, and translation tables that are specified in the SMMUv3 architecture.

Accesses to all physical addresses, except for fetches of GPT information, are subject to granule protection checks.

A client-originated access that experiences a GPC fault is signaled to the client device in the same manner as an External abort.

A client-originated access that experiences a GPC fault on the output address of the access is not reported in the Event queue.

### 2.3.1 GPC for client devices without a StreamID

Granule protection checks can also apply to accesses from client devices that are not associated with a StreamID. These devices are referred to as NoStreamID devices.

NoStreamID devices only access physical address space, and are not associated with any stage 1 or stage 2 translation configuration.

The GPC fault reporting behavior for accesses from NoStreamID devices is the same as for regular client-originated accesses.

NoStreamID devices are not associated with a SEC\_SID value.

Transactions issued by a NoStreamID device include both a physical address and a Physical Address Space, PAS.

An access from a NoStreamID device with a physical address that exceeds the implemented output address size, advertised in SMMU\_IDR5.OAS, is terminated with an abort and no Event record or fault is recorded.

The SMMU does not perform any architectural transformations or overrides on NoStreamID accesses, but the SMMU may apply protocol-specific normalisation on transaction attributes.

The MPAM PARTID and PMG values used for NoStreamID accesses is an IMPLEMENTATION DEFINED choice between:

- Values provided by the device.
- Zero.

The MPAM PARTID and PMG values used for GPT walks caused by NoStreamID accesses is an IMPLEMENTATION DEFINED choice between:

- Values provided by the device.
- Zero.

The MPAM\_NS or MPAM\_SP values used for NoStreamID accesses, and GPT walks caused by NoStreamID accesses, are independently an IMPLEMENTATION DEFINED choice between:

- The value provided by the device.
- The MPAM\_NS or MPAM\_SP value corresponding to the target PA space of the access.

For an SMMU that does not support MPAM\_SP, then it is permitted to map MPAM\_SP to MPAM\_NS as follows:

- NoStreamID accesses to Root PA space use the Secure MPAM\_NS value of 0.
- NoStreamID accesses to Realm PA space use the Non-secure MPAM\_NS value of 1.

Any IMPLEMENTATION DEFINED determination of MPAM attributes must comply with the requirements of the MPAM architecture [3].

Note: The address and physical address space of an access are sufficient to determine the success of the granule protection check for that access.

Note: The design and integration of any NoStreamID device must guarantee that the client device can only issue accesses to physical address spaces appropriate to its Security state.

Support for checking accesses from NoStreamID devices is IMPLEMENTATION DEFINED. Note: Granule protection checks for NoStreamID devices are anticipated to be used for checking accesses from devices that do not undergo translation, but must not have unlimited access to physical address spaces. This includes accesses from a Debug Access Port and accesses made by a GIC to ITS tables.

### 2.3.2 Speculative and hint accesses

In SMMUv3 [2], the SMMU does not report faults encountered during a speculative translation request, translation of transactions marked as speculative, or for NW-DCP transactions.

For an SMMU with RME, granule protection check faults encountered during a speculative translation request, translation of transactions marked as speculative, or for NW-DCP transactions, are reported as follows:

- No event record is generated.
- If `SMMU_IDR0.RME_IMPL = 0`, it is `CONSTRAINED UNPREDICTABLE` whether the granule protection check fault is reported or not reported. If it is reported, then it is reported in the appropriate `SMMU_ROOT_GPF_FAR` or `SMMU_ROOT_GPT_CFG_FAR` register, if that register does not already contain an active fault.
- If `SMMU_IDR0.RME_IMPL = 1`, the granule protection check fault is not reported.

For speculative translation requests, then:

- If `SMMU_IDR0.RME_IMPL = 0`, it is `CONSTRAINED UNPREDICTABLE` whether the granule protection check on the output address of a translation request is applied at the time of the translation or only when a transaction using the translation is issued.
- If `SMMU_IDR0.RME_IMPL = 1`, the granule protection check on the output address of a translation request is applied at the time of the translation.

## 2.4 Interactions with PCIe ATS

Consistent with the rules for all accesses, all PCIe client transactions are subject to granule protection checks.

The SMMU\_CR0.ATSCHK bit has no effect on granule protection checks.

If an SMMU-originated access experiences a GPC fault while servicing an ATS Translation Request, the SMMU responds to the ATS Translation Request as Completer Abort.

If an ATS Translation Request is completed with Success and R=W=0, the address in the Translation Completion is not valid and it is not subject to granule protection checks.

The SMMU is permitted but not required to perform a granule protection check on the output address for the result of an ATS Translation Request.

If the output address for an ATS Translation Request fails a granule protection check, the SMMU responds to the ATS Translation Request as Completer Abort.

ATS Translated transactions are subject to granule protection checks.

An ATS Translated transaction that fails the granule protection check is treated as Completer Abort.

## 2.5 Interactions with ATOS interfaces

Consistent with the rules for SMMU-originated accesses, configuration structure fetches, and translation table walks arising from use of the ATOS registers are subject to granule protection checks.

If a configuration structure fetch or translation table walk fails with a granule protection check fault, this is reported both:

- In the appropriate SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPT\_CFG\_FAR register, if that register does not already contain an active fault.
- In the appropriate ATOS\_PAR register, as though the failed access experienced an External abort.

When performing an ATOS operation, the SMMU does not perform a granule protection check on the final output address of a successful translation.



## 2.6 SMMU-originated accesses

An SMMU-originated access that experiences a GPC fault is reported as though it had experienced an External abort as specified in the SMMUv3 architecture.

Consistent with the behavior of F\_STE\_FETCH, F\_CD\_FETCH, F\_VMS\_FETCH and F\_WALK\_EABT in SMMUv3, signaling and reporting of these failures is not affected by the value of the CD.{A, R, S} bits nor the STE.{S2S, S2R} bits.

For an SMMU with SMMU\_IDR0.RME\_IMPL == 1:

- For each of the F\_STE\_FETCH, F\_CD\_FETCH, F\_VMS\_FETCH and F\_WALK\_EABT event records, there is a new field at bit 80 named GPCF.
- An F\_STE\_FETCH, F\_CD\_FETCH, F\_VMS\_FETCH or F\_WALK\_EABT arising from a GPC fault is reported with GPCF=1.
- An F\_STE\_FETCH, F\_CD\_FETCH, F\_VMS\_FETCH or F\_WALK\_EABT arising for a reason other than a GPC fault is reported with GPCF=0. This is unchanged from SMMUv3.

For example, if the SMMU experiences a GPC fault on an access to an STE, it is reported as F\_STE\_FETCH, with GPCF=1. This is signaled to the client as an External abort.

For example, if the SMMU experiences a GPC fault on an access to the Non-secure Event queue, it is reported through SMMU\_GERROR.EVENTQ\_ABT\_ERR.

See also:

- [4.1 F\\_STE\\_FETCH, F\\_CD\\_FETCH, F\\_VMS\\_FETCH and F\\_WALK\\_EABT](#)

## 2.7 Reporting of GPC faults

The reasons for a GPC fault are categorized into three groups:

- Faults arising from an access to a Location forbidden by the GPT configuration, referred to as a *Granule Protection Fault* (GPF).
- Faults arising from misconfiguration of SMMU\_ROOT\_GPT\_BASE, SMMU\_ROOT\_GPT\_BASE\_CFG or the GPT, referred to as a *GPT lookup error*.
- Faults arising from RAS errors.

The following error conditions represent a GPF, and are reported in SMMU\_ROOT\_GPF\_FAR:

- An access to a physical address space other than Non-secure has a physical address that exceeds the range configured in SMMU\_ROOT\_GPT\_BASE\_CFG.PPS.
- An access was attempted to a location that is forbidden by the configuration in the GPT.

The following error conditions represent a GPT lookup error, and are reported in SMMU\_ROOT\_GPT\_CFG\_FAR:

- Fields in SMMU\_ROOT\_GPT\_BASE\_CFG are configured to reserved values.
- Configuration of SMMU\_ROOT\_GPT\_BASE\_CFG.PPS to exceed SMMU\_IDR5.OAS.
- Configuration of SMMU\_ROOT\_GPT\_BASE\_CFG.{SH, IRGN, ORGN} to an invalid combination.
- SMMU\_ROOT\_GPT\_BASE.ADDR is configured to exceed the size configured in SMMU\_ROOT\_GPT\_BASE\_CFG.PPS.
- The output address of a GPT Table Entry exceeds the size configured in SMMU\_ROOT\_GPT\_BASE\_CFG.PPS.
- The SMMU depends on using values in an invalid GPT Entry.
- The SMMU experienced an External abort while fetching a GPT Entry.
- The SMMU experienced a RAS error while fetching a GPT Entry. This is reported in the same manner as if the SMMU experienced an External abort while fetching a GPT Entry.

See also:

- [2.11.7 SMMU\\_ROOT\\_GPT\\_BASE\\_CFG](#)

## 2.8 SMMU behavior if a GPC fault is active

If a client-originated or SMMU-originated access experiences a GPF reported in SMMU\_ROOT\_GPF\_FAR, then:

- If there is no prior GPF in SMMU\_ROOT\_GPF\_FAR, the appropriate syndrome information is recorded in SMMU\_ROOT\_GPF\_FAR.
- Other accesses that do not experience a GPF or GPT lookup error continue as specified.
- The GPF remains active until software writes 0 to SMMU\_ROOT\_GPF\_FAR.FAULT.

If a client-originated or SMMU-originated access experiences a GPT lookup error reported in SMMU\_ROOT\_GPT\_CFG\_FAR, then:

- If there is no prior GPT lookup error in SMMU\_ROOT\_GPT\_CFG\_FAR, the appropriate syndrome information is recorded in SMMU\_ROOT\_GPT\_CFG\_FAR.
- Other accesses that do not experience a GPF or GPT lookup error continue as specified.
- The GPT lookup error remains active until software writes 0 to SMMU\_ROOT\_GPT\_CFG\_FAR.FAULT.

An SMMU with RME has two additional edge-triggered wired interrupts:

Source	Trigger reason
GPF_FAR	An error becomes active in SMMU_ROOT_GPF_FAR.
GPT_CFG_FAR	An error becomes active in SMMU_ROOT_GPT_CFG_FAR.

## 2.9 Broadcast TLBI interface

An SMMU with RME and `SMMU_ROOT_IDR0.BGPTM == 1` participates in broadcast `TLBI *PA*` instructions from PEs that are executing in EL3.

Consistent with the definition in the RME specification [1], a `TLBI *PA*` to the Outer Shareable shareability domain affects the SMMU.

This applies to all SMMUs with RME and `SMMU_ROOT_IDR0.BGPTM == 1`, regardless of the values of `SMMU_IDR0.BTM` and `SMMU_(S_)CR2.PTM`.

The behavior of `SMMU_IDR0.BTM` and `SMMU_(S_)CR2.PTM` applies only to broadcast invalidations relating to stage 1 and stage 2 translation.

An SMMU with RME does not have to receive the other broadcast TLBI operations. Support for broadcast operations is indicated in `SMMU_IDR0.BTM` and configured in `SMMU_(S_)CR2.PTM`.

The SMMU guarantees the same rules around observability and completion of `TLBI *PA*` and `DSB` instructions as defined in the RME specification.

## 2.10 Observability of GPC faults

If the termination of a client transaction as a result of a GPC fault is observable to the client device, then:

- If the appropriate SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR register already contained an active fault, then it is not updated in this case.
- If the appropriate SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR register did not already contain an active fault, then the related syndrome information is observable in the appropriate register.

If an interrupt indicating the presence of a GPC fault is observable, then the syndrome information is observable in the SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR register as appropriate.

If the termination of a client transaction as a result of a GPC fault has been observable to the client device, completion of a subsequent CMD\_SYNC guarantees observability of any related events in the Event queue or, if the Event queue is unwritable, that the Event will not become observable.

For an SMMU with SMMU\_ROOT\_IDR0.BGPTM == 1, then:

- After completion of a broadcast TLBI \*PA\* and DSB instruction on the PE, completion of a subsequent CMD\_SYNC guarantees that no Events relating to GPT configuration invalidated by that TLBI and DSB will later be made observable in the Event queue.
- After a broadcast TLBI \*PA\* instruction on the PE, completion of a subsequent DSB instruction guarantees that any errors reported in the SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR registers relating to GPT configuration invalidated by that TLBI are already observable.

For an SMMU with SMMU\_ROOT\_IDR0.RGPTM == 1, then:

- After completion of a register-based TLBI by PA, indicated by SMMU\_ROOT\_TLBI\_CTRL.RUN, completion of a subsequent CMD\_SYNC guarantees that no Events relating to GPT configuration invalidated by that TLBI by PA will later be made observable in the Event queue.
- Completion of a register-based TLBI by PA, indicated by SMMU\_ROOT\_TLBI\_CTRL.RUN, guarantees that any errors reported in the SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR registers relating to GPT configuration invalidated by that TLBI by PA are already observable.

If an F\_STE\_FETCH, F\_CD\_FETCH, F\_VMS\_FETCH or F\_WALK\_EABT with GPCF == 1 is observable in the Event queue then either:

- If the appropriate SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR register already contained an active fault, then it is not updated in this case.
- If the appropriate SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR register did not already contain an active fault, then the related syndrome information is observable in the appropriate register.

If an update to SMMU\_(S\_)GERROR resulting from a GPC fault is observable, then either:

- If the appropriate SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR register already contained an active fault, then it is not updated in this case.
- If the appropriate SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR register did not already contain an active fault, then the related syndrome information is observable in the appropriate register.

If a fault to be reported in SMMU\_(S\_)GERROR is observable in SMMU\_ROOT\_GPF\_FAR or SMMU\_ROOT\_GPF\_CFG\_FAR, then the fault will also be reported in SMMU\_(S\_)GERROR in finite time. The existing mechanisms for ensuring the visibility of errors in SMMU\_(S\_)GERROR also apply in this case. For example, completion of an Update of SMMU\_CR0.EVENTQEN to 0 guarantees observability of any errors to be reported in SMMU\_GERROR.EVENTQ\_ABT\_ERR.

## 2.11 New registers

### 2.11.1 Root Control Page

A new register page is introduced, the SMMU Root Control Page.

Access to the SMMU Root Control Page is bounded by the following rules:

- The base address is IMPLEMENTATION DEFINED.
- The base address is 64KB aligned.
- The page is accessible only in the Root physical address space.
- Accesses in any physical address space other than Root are completed as RAZ/WI.
- The Root Control Page base address is distinct from addresses of registers accessible in other physical address spaces.

Address map:

Offset	Register	Notes
0x0000	SMMU_ROOT_IDR0	32-bit, RO
0x0008	SMMU_ROOT_IIDR	32-bit, RO
0x0020	SMMU_ROOT_CR0	32-bit, R/W
0x0024	SMMU_ROOT_CR0ACK	32-bit, RO
0x0028	SMMU_ROOT_GPT_BASE	64-bit, R/W
0x0030	SMMU_ROOT_GPT_BASE_CFG	64-bit, R/W
0x0038	SMMU_ROOT_GPF_FAR	64-bit, R/W
0x0040	SMMU_ROOT_GPT_CFG_FAR	64-bit, R/W
0x0050	SMMU_ROOT_TLBI	64-bit, R/W, OPTIONAL
0x0058	SMMU_ROOT_TLBI_CTRL	32-bit, R/W, OPTIONAL
0x0E00-0x0EFF	IMPLEMENTATION DEFINED	IMPLEMENTATION DEFINED

Locations in the SMMU Root Control Page that are not associated with a register are RES0.

All unallocated register bits are RES0.

### 2.11.2 SMMU\_ROOT\_IDR0

The SMMU ROOT IDR0 characteristics are:

## Purpose

Feature identification register

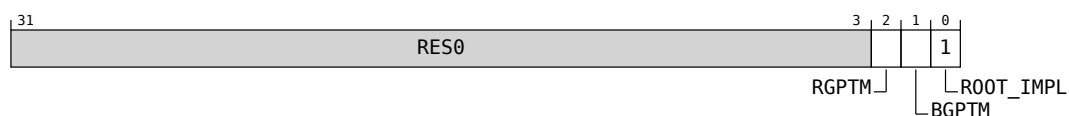
## Attributes

SMMU\_ROOT\_IDR0 is a 32-bit register.

This register is part of the [SMMUv3\\_ROOT](#) block.

## Field descriptions

The SMMU ROOT IDR0 bit assignments are:

**Bits [31:3]**

Reserved, RES0.

***RGPTM, bit [2]***

Register TLBI by PA support.

RGPTM	Meaning
0b0	SMMU_ROOT_TLBI and SMMU_ROOT_TLBI_CTRL are not present.
0b1	SMMU_ROOT_TLBI and SMMU_ROOT_TLBI_CTRL are present.

If BGPTM == 0, then RGPTM == 1.

See also:

- SMMU\_ROOT\_TLBI.
- SMMU\_ROOT\_TLBI\_CTRL.

**BGPTM, bit [1]**

Broadcast TLBI by PA support.

<b>BGPTM</b>	<b>Meaning</b>
0b0	This SMMU does not participate in broadcast TLBI *PA* operations.
0b1	This SMMU does participate in broadcast TLBI *PA* operations.

This field indicates both that:

- The SMMU supports receipt of broadcast `TLBI *PA*` operations issued by PEs.
- The SMMU is integrated in the memory system such that `TLBI *PA*` operations issued to the Outer Shareable shareability domain by PEs correctly reach the SMMU.

The value of this field is independent of the value of `SMMU_IDR0.BTM`, `SMMU_CR2.PTM`, and `SMMU_S_CR2.PTM`.

See also:

- [2.9 Broadcast TLBI interface](#).

#### ***ROOT\_IMPL, bit [0]***

Presence of Root registers.

Reads as 0b1

Access to this field is **RO**.

### **Accessing the SMMU\_ROOT\_IDR0**

`SMMU_ROOT_IDR0` can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0000

Access on this interface is **RO**.



### 2.11.3 SMMU\_ROOT\_IIDR

The SMMU\_ROOT\_IIDR characteristics are:

#### Purpose

Implementation identification register.

#### Attributes

SMMU\_ROOT\_IIDR is a 32-bit register.

This register is part of the [SMMUv3\\_ROOT](#) block.

#### Field descriptions

The SMMU\_ROOT\_IIDR bit assignments are:

31	20	19	16	15	12	11	0
ProductID				Variant	Revision	Implementer	

#### **ProductID, bits [31:20]**

IMPLEMENTATION DEFINED value identifying the SMMU part.

#### **Variant, bits [19:16]**

IMPLEMENTATION DEFINED value used to distinguish product variants, or major revisions of the product.

#### **Revision, bits [15:12]**

IMPLEMENTATION DEFINED value used to distinguish minor revisions of the product.

#### **Implementer, bits [11:0]**

Contains the JEP106 code of the company that implemented the SMMU.

- SMMU\_ROOT\_IIDR[11:8] - The JEP106 continuation code of the implementer.
- SMMU\_ROOT\_IIDR[7] - Always 0.
- SMMU\_ROOT\_IIDR[6:0] - The JEP106 identity code of the implementer.

For an Arm implementation, bits[11:0] are 0x43B.

#### Accessing the SMMU\_ROOT\_IIDR

SMMU\_ROOT\_IIDR can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0008

Access on this interface is **RO**.

#### 2.11.4 SMMU\_ROOT\_CR0

The SMMU\_ROOT\_CR0 characteristics are:

## Purpose

## Root Control Register

## Attributes

SMMU\_ROOT\_CR0 is a 32-bit register.

This register is part of the [SMMUv3\\_ROOT](#) block.

## Field descriptions

The SMMU\_ROOT\_CR0 bit assignments are:

**Bits [31:2]**

Reserved, RES0.

**GPCEN, bit [1]**

### Enable Granule Protection Checks.

GPCEN	Meaning
0b0	All accesses bypass granule protection checks.
0b1	All client and SMMU-originated accesses, except for GPT walks, are subject to granule protection checks.

When this bit is changed, the SMMU updates SMMU\_ROOT\_CR0ACK.GPCEN to match, once the following observability requirements are satisfied. This is referred to as completion of an Update.

Completion of an Update of GPCEN from 0 to 1 guarantees that:

- All future client-originated and SMMU-originated transactions are subject to granule protection checks.
- All previous transactions that were issued without a granule protection check have completed.

Completion of an Update of GPCEN from 1 to 0 guarantees that:

- All future client-originated and SMMU-originated transactions will bypass granule protection checks.
- All prior faults to be reported in SMMU\_ROOT\_GPF\_FAR and SMMU\_ROOT\_GPT\_CFG\_FAR have been reported.
- The SMMU has completed all outstanding fetches of GPT information.

Note: Completion of an Update of GPCEN from 1 to 0 does not guarantee that outstanding accesses using the previous GPT configuration have completed. However, completion of a TLBI by PA with scope PAALL after the completion of the Update of GPCEN does guarantee this, including for Non-secure accesses made to Locations above the range configured in SMMU\_ROOT\_GPT\_BASE.PPS.

The reset behavior of this field is:

- This field resets to 0b0.

Accessing this field has the following behavior:

- Access is **RO** if any of the following are true:
  - SMMU\_ROOT\_CR0.ACCESSEN != SMMU\_ROOT\_CR0ACK.ACCESSEN
  - SMMU\_ROOT\_CR0.GPCEN != SMMU\_ROOT\_CR0ACK.GPCEN
- Otherwise, access to this field is **RW**

#### **ACCESSEN, bit [0]**

Enable accesses from the SMMU and client devices.

ACCESSEN	Meaning
0b0	SMMU-originated accesses and client-originated accesses do not become observable.
0b1	SMMU-originated accesses and client-originated accesses are not terminated by this mechanism.

This bit has higher priority than GPCEN.

When this bit is changed, the SMMU updates SMMU\_ROOT\_CR0ACK.ACCESSEN to match, once the observability requirements below are satisfied. This is referred to as completion of an Update.

Completion of an Update of ACCESSEN from 0 to 1 guarantees that:

- Previous accesses that were to be terminated by this mechanism have been marked to be terminated.
- Future client-originated and SMMU-originated accesses might succeed, according to other architectural checks.

Completion of an Update of ACCESSEN from 1 to 0 guarantees that:

- Previous client-originated accesses not terminated by this mechanism are observable to their Shareability domain.
- Previous SMMU-originated accesses, including GPT fetches, have completed.
- The SMMU will not issue GPT fetches.
- Future SMMU-originated accesses will be terminated as though experiencing a GPF as-reported in SMMU\_ROOT\_GPF\_FAR.
- Future client-originated accesses will be processed in a manner consistent with any access to a physical address experiencing a GPF. This includes:
  - An access is terminated with an external abort as a result of a configuration structure or translation table access experiencing a GPF.
  - An access is terminated with an external abort as a result of a GPF on the output address of that access.
  - If cached configuration information and information cached in TLBs would permit the access to be stalled, it is stalled.
  - If cached configuration information and information cached in TLBs would permit the access to be completed as RAZ/WI, it is completed as RAZ/WI.

Note: For an SMMU that supports stall model, advertised in SMMU\_(S\_)IDR0.STALL\_MODEL, there may be outstanding transactions affected by stall configuration if ACCESSEN is programmed to 0 while SMMU\_(S\_)CR0.SMMUEN == 1. Configuration of SMMU\_(S\_)CR0.SMMUEN to 0 guarantees termination of stalled transactions.

The reset behavior of this field is:

- This field resets to 0b0.

Accessing this field has the following behavior:

- Access is **RO** if any of the following are true:

- SMMU\_ROOT\_CR0.ACCESSEN != SMMU\_ROOT\_CR0ACK.ACCESSEN
- SMMU\_ROOT\_CR0.GPCEN != SMMU\_ROOT\_CR0ACK.GPCEN
- Otherwise, access to this field is **RW**

### Accessing the SMMU\_ROOT\_CR0

SMMU\_ROOT\_CR0 can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0020

Access on this interface is **RW**.

### 2.11.5 SMMU\_ROOT\_CR0ACK

The SMMU\_ROOT\_CR0ACK characteristics are:

## Purpose

Provides acknowledgment of changes to configuration in `SMMU_ROOT_CR0`.

## Attributes

SMMU\_ROOT\_CR0ACK is a 32-bit register.

This register is part of the [SMMUv3\\_ROOT](#) block.

## Field descriptions

The SMMU\_ROOT\_CR0ACK bit assignments are:

**Bits [31:2]**

Reserved, RES0.

**GPCEN, bit [1]**

Acknowledgment that granule protection checks are enabled.

See: [SMMU\\_ROOT\\_CR0.GPCEN](#).

The reset behavior of this field is:

- This field resets to 0b0.

**ACCESSEN, bit [0]**

Acknowledgment that SMMU-originated and client-originated accesses are enabled.

See: [SMMU\\_ROOT\\_CR0.ACCESSEN](#).

The reset behavior of this field is:

- This field resets to 0b0.

## Accessing the SMMU\_ROOT\_CR0ACK

SMMU\_ROOT\_CR0ACK can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0024

Access on this interface is **RO**.

### 2.11.6 SMMU\_ROOT\_GPT\_BASE

The SMMU\_ROOT\_GPT\_BASE characteristics are:

## Purpose

Control register for Granule Protection Table base address.

This register is analogous to GPTBR\_EL3.

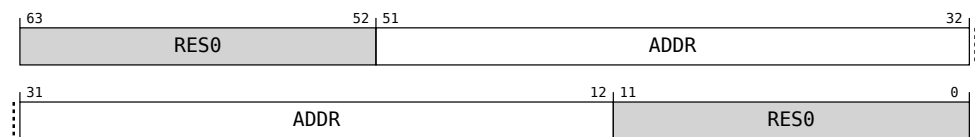
## Attributes

SMMU\_ROOT\_GPT\_BASE is a 64-bit register.

This register is part of the [SMMUv3\\_ROOT](#) block.

## Field descriptions

The SMMU\_ROOT\_GPT\_BASE bit assignments are:



**Bits [63:52]**

Reserved, RES0.

**ADDR, bits [51:12]**

Base address of the L0GPT, bits [51:12].

The SMMU always treats this address as being in the Root physical address space.

This field represents bits [51:12] of the level 0 GPT base address.

Bits that are taken to be zero are RES0 and the SMMU treats them as zero when computing addresses for lookups.

Bits above the implemented output address size, advertised in SMMU\_IDR5.OAS, are RES0.

The level 0 GPT is aligned in memory to the greater of:

- The size of the level 0 GPT in bytes.
- 4KB.

Bits [x:0] of the base address are taken to be zero, where:

- $x = \text{Max}(\text{pps} - \text{lgptsz} + 2, 11)$
- pps is derived from `SMMU_ROOT_GPT_BASE_CFG.PPS` as follows:

PPS	pps
0b000	32
0b001	36
0b010	40
0b011	42
0b100	44
0b101	48

PPS	pps
0b110	52

- l0gptsz is derived from [SMMU\\_ROOT\\_GPT\\_BASE\\_CFG.L0GPTSZ](#) as follows:

L0GPTSZ	l0gptsz
0b0000	30
0b0100	34
0b0110	36
0b1001	39

If x is greater than 11, then BADDR[x - 12:0] are RES0.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

#### **Bits [11:0]**

Reserved, RES0.

### **Accessing the SMMU\_ROOT\_GPT\_BASE**

This register is writeable even when granule protection checks are enabled. After a write of this register, the SMMU is not required to use the new base address value until completion of a subsequent TLBI by PA ALL operation.

Completion of such a TLBI by PA ALL operation also guarantees that the SMMU has completed all outstanding GPT walks that used the old configuration.

SMMU\_ROOT\_GPT\_BASE can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0028

Access on this interface is **RW**.

## 2.11.7 SMMU\_ROOT\_GPT\_BASE\_CFG

The SMMU\_ROOT\_GPT\_BASE\_CFG characteristics are:

### Purpose

Control register for Granule Protection Checks.

The fields in SMMU\_ROOT\_GPT\_BASE\_CFG are the same as for GPCCR\_EL3, except there is no copy of GPCCR\_EL3.GPC. See [SMMU\\_ROOT\\_CR0.GPCEN](#).

Configuration of reserved or invalid values leads to a GPT lookup error, reported as Invalid configuration of GPT configuration registers.

See also:

- [2.7 Reporting of GPC faults](#).

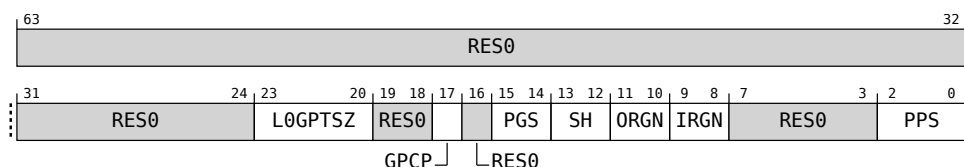
### Attributes

SMMU\_ROOT\_GPT\_BASE\_CFG is a 64-bit register.

This register is part of the [SMMUv3\\_ROOT](#) block.

### Field descriptions

The SMMU\_ROOT\_GPT\_BASE\_CFG bit assignments are:



### Bits [63:24]

Reserved, RES0.

### LOGPTSZ, bits [23:20]

Level 0 GPT entry size.

This field advertises the number of least significant address bits protected by each entry in the level 0 GPT.

LOGPTSZ	Meaning
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.



Access to this field is **RO**.

**Bits [19:18]**

Reserved, RES0.

**GPCP, bit [17]**

Granule Protection Check Priority.

GPCP	Meaning
0b0	All GPC faults are reported with a priority consistent with the GPC being performed on any access to physical address space.
0b1	A GPC fault for the fetch of a Table descriptor for a stage 2 translation table walk might not be generated or reported. All other GPC faults are reported with a priority consistent with the GPC being performed on any access to physical address space.

This field is permitted to be cached in a TLB.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

**Bit [16]**

Reserved, RES0.

**PGS, bits [15:14]**

Physical Granule size.

PGS	Meaning
0b00	4KB. Invalid if SMMU_IDR5.GRAN4K == 0.
0b01	64KB. Invalid if SMMU_IDR5.GRAN64K == 0.
0b10	16KB. Invalid if SMMU_IDR5.GRAN16K == 0.
0b11	Reserved.

The value of this field is permitted to be cached in a TLB.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

**SH, bits [13:12]**

GPT fetch Shareability attribute

SH	Meaning
0b00	Non-shareable.
0b01	Reserved.
0b10	Outer Shareable.
0b11	Inner Shareable.

Fetches to the GPT are made to the shareability domain configured in this field.

If both ORGN and IRGN are configured with Non-cacheable attributes, it is invalid to configure this field to values other than Outer Shareable.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

#### **ORGN, bits [11:10]**

GPT fetch Outer cacheability attribute.

ORGN	Meaning
0b00	Normal memory, Outer Non-cacheable.
0b01	Normal memory, Outer Write-Back Read-Allocate Write-Allocate Cacheable.
0b10	Normal memory, Outer Write-Through Read-Allocate No Write-Allocate Cacheable.
0b11	Normal memory, Outer Write-Back Read-Allocate No Write-Allocate Cacheable.

Fetches of GPT information are made with the Outer cacheability attributes configured in this field.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

#### **IRGN, bits [9:8]**

GPT fetch Inner cacheability attribute.

IRGN	Meaning
0b00	Normal memory, Inner Non-cacheable.
0b01	Normal memory, Inner Write-Back Read-Allocate Write-Allocate Cacheable.
0b10	Normal memory, Inner Write-Through Read-Allocate No Write-Allocate Cacheable.
0b11	Normal memory, Inner Write-Back Read-Allocate No Write-Allocate Cacheable.

Fetches of GPT information are made with the Inner cacheability attributes configured in this field.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

#### **Bits [7:3]**

Reserved, RES0.

#### **PPS, bits [2:0]**

Protected physical address size.

The bit width of the memory region protected by the GPT.

PPS	Meaning
0b000	32 bits, 4GB protected address space.
0b001	36 bits, 64GB protected address space.
0b010	40 bits, 1TB protected address space.
0b011	42 bits, 4TB protected address space.
0b100	44 bits, 16TB protected address space.
0b101	48 bits, 256TB protected address space.
0b110	52 bits, 4PB protected address space.
0b111	Reserved.

Values exceeding the implemented physical address size, advertised in SMMU\_IDR5.OAS, are invalid.

This field is permitted to be cached in a TLB or Configuration cache.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

### **Accessing the SMMU\_ROOT\_GPT\_BASE\_CFG**

SMMU\_ROOT\_GPT\_BASE\_CFG can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0030

- When SMMU\_ROOT\_CR0.GPCEN == 1 or SMMU\_ROOT\_CR0ACK.GPCEN == 1, access on this interface is **RO**.
- Otherwise, access on this interface is **RW**.

## 2.11.8 SMMU\_ROOT\_GPF\_FAR

The SMMU\_ROOT\_GPF\_FAR characteristics are:

### Purpose

This register reports details of the originating access that experienced a GPF.

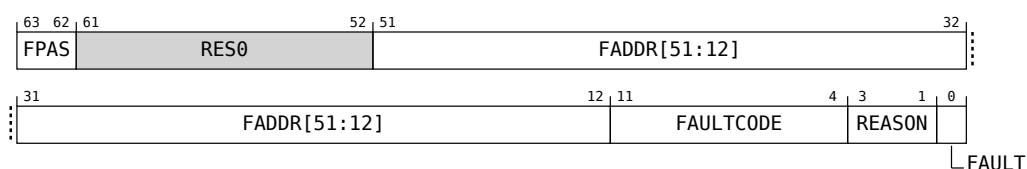
### Attributes

SMMU\_ROOT\_GPF\_FAR is a 64-bit register.

This register is part of the [SMMUv3\\_ROOT](#) block.

### Field descriptions

The SMMU\_ROOT\_GPF\_FAR bit assignments are:



### FPAS, bits [63:62]

The physical address space of the access that failed.

FPAS	Meaning
0b00	Secure
0b01	Non-secure
0b10	Root
0b11	Realm

If FAULT == 0, the value of this field is 0b00.

Note: The encodings for Root and Realm are only applicable for NoStreamID devices.

### Bits [61:52]

Reserved, RES0.

### FADDR, bits [51:12]

The physical address input to the Granule Protection Check that failed.

If FAULT == 0, the value of this field is zero.

### FAULTCODE, bits [11:4]

If REASON == TRANSACTION, then the value of this field is zero.

If REASON == TRANSLATION, then:

Value	Name	Meaning
0x03	GPF_STE_FETCH	STE fetch experienced GPF
0x09	GPF_CD_FETCH	CD fetch experienced GPF
0x0B	GPF_WALK_EABT	Translation Table access experienced GPF
0x25	GPF_VMS_FETCH	VMS fetch experienced GPF

If REASON == GERROR, then:

Value	Name	Meaning
0x00	CMDQ_GPF	Command queue read experienced a GPF
0x02	EVENTQ_GPF	Event queue write experienced a GPF
0x03	PRIQ_GPF	PRI queue write experienced a GPF
0x04	MSI_CMDQ_GPF	Command queue MSI experienced a GPF
0x05	MSI_EVENTQ_GPF	Event queue MSI experienced a GPF
0x06	MSI_PRIQ_GPF	PRI queue MSI experienced a GPF
0x07	MSI_GERROR_GPF	GERROR reporting MSI experienced a GPF
0x10	OTHER_GPF	An unknown SMMU-originated access experienced a GPF.

If FAULT == 0, the value of this field is 0x00.

**REASON, bits [3:1]**

Reports the originator of the access.

REASON	Meaning
0b001	TRANSLATION, GPF on an SMMU-originated access required for translation of a client request.
0b010	GERROR, GPF on an SMMU-originated access not relating to a client translation.
0b011	TRANSACTION, GPF on the output address of a client translation.

If FAULT == 0, the value of this field is 0b000.

**FAULT, bit [0]**

FAULT	Meaning
0b0	There have been zero GPFs since this register was last cleared.

FAULT	Meaning
0b1	There have been one or more GPFs since this register was last cleared.

A write of 1 to this bit is IGNORED, does not trigger the GPF\_FAR interrupt, and does not make this fault active.

The reset behavior of this field is:

- This field resets to 0b0.

#### **Additional information**

An MSI access from a PMCG or RAS record interrupt that experiences a GPF is permitted to be reported as either of:

- REASON = GERROR and FAULTCODE = OTHER\_GPF
- REASON = TRANSACTION

### **Accessing the SMMU\_ROOT\_GPF\_FAR**

All writes to this register are IGNORED unless the write clears the FAULT bit.

When a write clears the FAULT bit, the whole register is cleared to zero.

SMMU\_ROOT\_GPF\_FAR can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0038

Access on this interface is **RW**.

### 2.11.9 SMMU\_ROOT\_GPT\_CFG\_FAR

The SMMU\_ROOT\_GPT\_CFG\_FAR characteristics are:

#### Purpose

Reports details of the originating access that experienced a GPT lookup error.

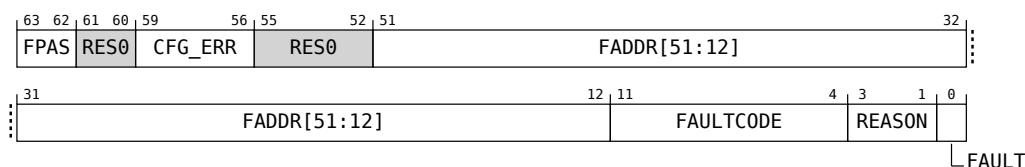
#### Attributes

SMMU\_ROOT\_GPT\_CFG\_FAR is a 64-bit register.

This register is part of the [SMMUv3\\_ROOT](#) block.

#### Field descriptions

The SMMU\_ROOT\_GPT\_CFG\_FAR bit assignments are:



#### FPAS, bits [63:62]

The physical address space of the access that failed.

FPAS	Meaning
0b00	Secure
0b01	Non-secure
0b10	Root
0b11	Realm

If FAULT == 0, the value of this field is 0b00.

Note: The encodings for Root and Realm are only applicable for NoStreamID devices.

#### Bits [61:60]

Reserved, RES0.

#### CFG\_ERR, bits [59:56]

Value	Meaning
0x0	Invalid configuration of GPT configuration registers. This corresponds to GPT walk fault at Level 0 arising from invalid configuration of GPCCR_EL3 in the RME specification.
0x1	SMMU_ROOT_GPT_BASE.ADDR exceeds the address size configured in SMMU_ROOT_GPT_BASE_CFG.PPS. This corresponds to GPT address size fault at Level 0 in the RME specification.
0x2	External abort on GPT entry fetch. This corresponds to Synchronous External abort on GPT fetch in the RME specification.

Value	Meaning
0x3	Invalid configuration of GPT entry. This corresponds to GPT walk fault arising from an invalid GPT entry in the RME specification.
0x4	Next-level address in GPT entry exceeds the address size configured in SMMU_ROOT_GPT_BASE_CFG.PPS. This corresponds to GPT address size fault at Level 0 in the RME specification.

If FAULT == 0, the value of this field is 0x0.

**Bits [55:52]**

Reserved, RES0.

**FADDR, bits [51:12]**

The physical address input to the Granule Protection Check that failed.

If FAULT == 0, the value of this field is zero.

**FAULTCODE, bits [11:4]**

If REASON == TRANSACTION, then the value of this field is zero.

If REASON == TRANSLATION, then:

Value	Name	Meaning
0x03	GPF_STE_FETCH	STE fetch experienced GPF
0x09	GPF_CD_FETCH	CD fetch experienced GPF
0x0B	GPF_WALK_EABT	Translation Table access experienced GPF
0x25	GPF_VMS_FETCH	VMS fetch experienced GPF

If REASON == GERROR, then:

Value	Name	Meaning
0x00	CMDQ_GPF	Command queue read experienced a GPF
0x02	EVENTQ_GPF	Event queue write experienced a GPF
0x03	PRIQ_GPF	PRI queue write experienced a GPF
0x04	MSI_CMDQ_GPF	Command queue MSI experienced a GPF
0x05	MSI_EVENTQ_GPF	Event queue MSI experienced a GPF
0x06	MSI_PRIQ_GPF	PRI queue MSI experienced a GPF
0x07	MSI_GERROR_GPF	GERROR reporting MSI experienced a GPF
0x10	OTHER_GPF	An unknown SMMU-originated access experienced a GPF.



If FAULT == 0, the value of this field is 0x00.

**REASON, bits [3:1]**

Reports the originator of the access.

REASON	Meaning
0b001	TRANSLATION, GPF on an SMMU-originated access required for translation of a client request.
0b010	GERROR, GPF on an SMMU-originated access not relating to a client translation.
0b011	TRANSACTION, GPF on the output address of a client translation.

If FAULT == 0, the value of this field is 0b000.

**FAULT, bit [0]**

FAULT	Meaning
0b0	There have been zero GPT lookup errors since this register was last cleared.
0b1	There have been one or more GPT lookup errors since this register was last cleared.

A write of 1 to this bit is IGNORED, does not trigger the GPT\_CFG\_FAR interrupt, and does not make this fault active.

The reset behavior of this field is:

- This field resets to 0b0.

**Additional information**

An MSI access from a PMCG or RAS record interrupt that experiences a GPF is permitted to be reported as either of:

- REASON = GERROR and FAULTCODE = OTHER\_GPF
- REASON = TRANSACTION

**Accessing the SMMU\_ROOT\_GPT\_CFG\_FAR**

All writes to this register are IGNORED unless the write clears the FAULT bit.

When a write clears the FAULT bit, the whole register is cleared to zero.

SMMU\_ROOT\_GPT\_CFG\_FAR can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0040

Access on this interface is **RW**.

## 2.11.10 SMMU\_ROOT\_TLBI

The SMMU\_ROOT\_TLBI characteristics are:

### Purpose

TLBI by PA attributes register.

### Attributes

SMMU\_ROOT\_TLBI is a 64-bit register.

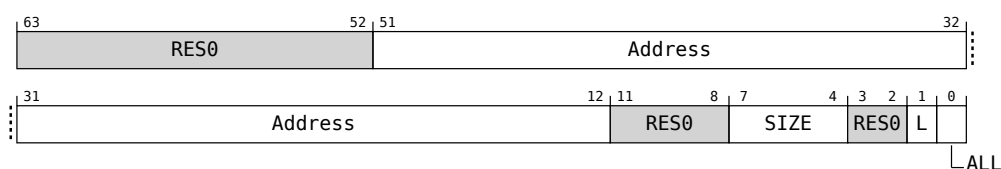
This register is part of the [SMMUv3\\_ROOT](#) block.

### Configuration

This register is present only when SMMU\_RME\_RGPTM is implemented. Otherwise, direct accesses to SMMU\_ROOT\_TLBI are RES0.

### Field descriptions

The SMMU\_ROOT\_TLBI bit assignments are:



#### Bits [63:52]

Reserved, RES0.

#### Address, bits [51:12]

Base address from which to start invalidation.

Bits [11:0] of the base address are taken to be zero.

If ALL == 1, this field is IGNORED.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

#### Bits [11:8]

Reserved, RES0.

#### SIZE, bits [7:4]

Range of addresses to be invalidated.

SIZE	Meaning
0b0000	4KB.
0b0001	16KB.
0b0010	64KB.
0b0011	2MB.

SIZE	Meaning
0b0100	32MB.
0b0101	512MB.
0b0110	1GB.
0b0111	16GB.
0b1000	64GB.
0b1001	512GB.

All other values are reserved.

If ALL == 1, this field is IGNORED.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

#### **Bits [3:2]**

Reserved, RES0.

#### **L, bit [1]**

GPT Last-level only.

L	Meaning
0b0	Invalidate GPT information from all levels of the GPT walk.
0b1	Invalidate GPT information from only the last level of the GPT walk.

If ALL == 1, this field is IGNORED.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

#### **ALL, bit [0]**

GPT All.

ALL	Meaning
0b0	Invalidate GPT information from TLBs based on other fields.
0b1	Invalidate all GPT information from TLBs.

The reset behavior of this field is:

- This field resets to an UNKNOWN value.

## Accessing the SMMU\_ROOT\_TLBI

SMMU\_ROOT\_TLBI can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0050

- When SMMU\_ROOT\_TLBI\_CTRL.RUN == 0, access on this interface is **RW**.
- Otherwise, access on this interface is **RO**.

### 2.11.11 SMMU\_ROOT\_TLBI\_CTRL

The SMMU\_ROOT\_TLBI\_CTRL characteristics are:

#### Purpose

TLBI by PA control register.

#### Attributes

SMMU\_ROOT\_TLBI\_CTRL is a 32-bit register.

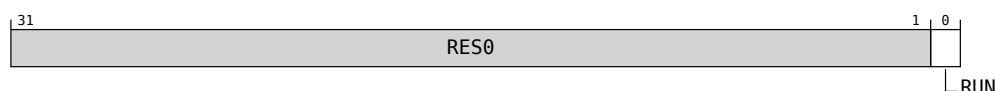
This register is part of the [SMMUv3\\_ROOT](#) block.

#### Configuration

This register is present only when SMMU\_RME\_RGPTM is implemented. Otherwise, direct accesses to SMMU\_ROOT\_TLBI\_CTRL are RES0.

#### Field descriptions

The SMMU\_ROOT\_TLBI\_CTRL bit assignments are:



#### Bits [31:1]

Reserved, RES0.

#### RUN, bit [0]

RUN	Meaning
0b0	Invalidation not in progress.
0b1	Invalidation in progress.

Writes to this register only have an effect if both of the following are true:

- The value of RUN in the register before the write is 0.
- The value of RUN in the write data payload is 1.

Any write that does not satisfy both these conditions is IGNORED.

When the requirements for RUN are met for a given write, the following all apply:

- The values provided for ALL, L, SIZE, and Address are taken from SMMU\_ROOT\_TLBI.
- The SMMU performs the TLBI by PA operation, interpreted as follows:
  - If ALL == 1, the operation behaves as TLBI PAALL as issued on a PE.
  - If ALL == 0 and L == 0, the operation behaves as TLBI RPAOS as issued on a PE, with SIZE and Address interpreted in the same manner as for TLBI RPAOS.
  - If ALL == 0 and L == 1, the operation behaves as TLBI RPALOS as issued on a PE, with SIZE and Address interpreted in the same manner as for TLBI RPALOS.
- A TLBI by PA operation is complete when all the following requirements are met:

- All matching GPT information in TLB entries has been invalidated.
- All SMMU-originated and client-originated accesses that were in progress to physical addresses targeted by the TLBI by PA operation, are globally observable to their Shareability domain.
- Once the TLBI by PA operation is complete, the SMMU clears the whole register to 0.

The reset behavior of this field is:

- This field resets to 0b0.

### Accessing the SMMU\_ROOT\_TLBI\_CTRL

SMMU\_ROOT\_TLBI\_CTRL can be accessed through the [SMMUv3\\_ROOT](#) block, as follows:

Frame	Offset
SMMUv3_ROOT	0x0058

Access on this interface is **RW**.

#### 2.11.12 IMPLEMENTATION DEFINED registers

All offsets from 0x0E00 to 0x0EFF in the SMMU Root Control Page are IMPLEMENTATION DEFINED.

Note: Consistent with the requirements for the SMMU Root Control Page as a whole, any such registers are only accessible in Root physical address space.

See also:

- [2.11 New registers](#)

## Chapter 3

# Register changes

This section describes changes to the registers specified in SMMUv3.



## 3.1 SMMU\_IDR0

A new bit is introduced:

**Bit [30]** RME\_IMPL

---

Value	Meaning
0b0	RME features not supported for Non-secure or Secure programming interfaces.
0b1	RME features supported for Non-secure programming interface, and for Secure programming interface if implemented.

---

If this bit is 1, it is possible to report F\_STE\_FETCH, F\_CD\_FETCH, F\_VMS\_FETCH and F\_WALK\_EABT with GPCF == 1 in the Non-secure Event queue.

See also:

- [4.1 F\\_STE\\_FETCH, F\\_CD\\_FETCH, F\\_VMS\\_FETCH and F\\_WALK\\_EABT](#)

## 3.2 SMMU\_S\_IDR1

The definition of SMMU\_S\_IDR1.SECURE\_IMPL is updated to say:

**Bit [31] SECURE\_IMPL**

Value	Meaning
0b0	The SMMU implements a single Security state. All SMMU_S_* Secure registers are RAZ/WI.
0b1	The SMMU implements two Security states.

When SECURE\_IMPL == 1, stage 1 must be supported, that is SMMU\_IDR0.S1P == 1.

If SECURE\_IMPL == 1 and SMMU\_IDR0.RME\_IMPL == 1, then all the following apply:

- SMMU\_S\_IDR1.SEL2 == 1.
- The EL3 StreamWorld is not supported.
- It is possible to report F\_STE\_FETCH, F\_CD\_FETCH, F\_VMS\_FETCH and F\_WALK\_EABT with GPCF == 1 in the Secure Event queue.

See also:

- [2.2 Removal of EL3 StreamWorld](#)
- [4.1 F\\_STE\\_FETCH, F\\_CD\\_FETCH, F\\_VMS\\_FETCH and F\\_WALK\\_EABT](#)

## 3.3 SMMU\_S\_INIT

Note: As GPT information is permitted to be cached in a TLB, the SMMU\_S\_INIT.INV\_ALL mechanism also invalidates GPT information cached in TLBs.

## 3.4 Accessibility of SMMU register pages

All SMMU registers that were specified to be accessible only in Secure physical address space in SMMUv3 are additionally accessible in Root physical address space in an SMMU with RME.

All SMMU registers that were specified to be accessible in Non-secure physical address space in SMMUv3 are additionally accessible in Root and Realm physical address spaces in an SMMU with RME.

## 3.5 Confidential information in RAS Error Records

The RME specification [1] provides implementation requirements for RAS in PEs and system components using the Arm RAS architecture [4].

The RME specification [1] describes the concept of *confidential information* and provides the requirements on the content that can be recorded in RAS error record registers.

In an SMMU with RME, the same requirements apply to RAS Error Record registers in that SMMU.

## 3.6 Interaction with PMCG events

The following architected PMCG events are permitted to count accesses to the GPT:

Event ID	Description
2	TLB miss caused by incoming transaction or translation request
4	Translation table walk accesses

The other architected PMCG events do not count accesses to the GPT.

Counters must only increment if a GPT access is required for the Security state configured in `SMMU_PMCGR_SCR.SO` and `SMMU_PMCGR_EVTYPEN<n>.FILTER_SEC_SID`.

IMPLEMENTATION DEFINED events are permitted to count GPT accesses, GPT-related TLB hits or GPT-related TLB misses for the appropriate Security state.

No PMCG events are counted for accesses from NoStreamID devices to Root or Realm physical address spaces.

## Chapter 4

### **Event record changes**

This section describes changes to the event records specified in SMMUv3 [\[2\]](#).

## 4.1 F\_STE\_FETCH, F\_CD\_FETCH, F\_VMS\_FETCH and F\_WALK\_EABT

### Bit 80 GPCF

Granule Protection Check fault

This field indicates whether the event record is the result of a GPC fault.

Value	Meaning
0b0	This event record is not the result of a GPC fault.
0b1	This event record is the result of a GPC fault.

If SMMU\_IDR0.RME\_IMPL == 0, this bit is RES0.

See also:

- [2.6 SMMU-originated accesses](#)



# Glossary

## **ATS Translated transaction**

A memory transaction input to the SMMU, in which the supplied address has been translated. In PCIe this is indicated with the AT TLP field value 0b10.

## **GPC**

Granule Protection Check

## **GPC fault**

A granule protection check fault, arising either because a GPT lookup could not be completed, or because the lookup was successful and the access being checked failed the check.

## **GPF, Granule Protection Fault**

The fault reported when a GPT lookup is successful, but the access being checked fails the granule protection check.

## **GPT, Granule Protection Table**

An in-memory structure that describes the association of a Location and a physical address space.

## **NoStreamID device**

An SMMU client device that is not associated with a StreamID.

## **PAS**

Physical address space

## **Untranslated transaction.**

A memory transaction input to the SMMU, in which the supplied address has not been translated. In PCIe this is indicated with the AT TLP field value 0b00.