



PL690 Generic Interrupt Controller **Software Developer Errata Notice**

This document contains all known errata since the r0p2 release of the product.

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.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. 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 subsidiaries) 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 © 2019 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

Confidentiality Status

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

Web address

<http://www.arm.com/>.

Feedback on this product

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

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

Feedback on this document

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

- The document title.
- The document number: SDEN-892601.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Contents

<i>INTRODUCTION</i>	6
<i>ERRATA SUMMARY TABLE</i>	9
<i>1494863</i> SPI recall failure without subsequent trigger	11
<i>1451068</i> DCHIPR reads 0 from chips that are not the default owner	21
<i>1335020</i> Targeted Cross-Chip SGIs incorrectly generate a SGI_OOR error on the source chip	22
<i>1196726</i> CLEAR command may continue after a SYNC	12
<i>1118145</i> QACTIVE may not be driven HIGH on qreqn_its_<x> transition	13
<i>1109171</i> Possible SPI corruption if GICD_IERRR<n> written to 1 without being read first in multi-chip configs	23
<i>1109146</i> GICD_CTLR.RWP may be stuck at 1 if already 1 when rt_owner changes	14
<i>1104459</i> ITS debug overflow not recorded in trace registers	24
<i>1098957</i> UNPREDICTABLE behavior on failed attempt to connect chips	25
<i>1071611</i> DPG might allow a single 1ofN interrupt to be sent after being set	26
<i>1049857</i> PMU filtering for Deactivate Event incorrect	27
<i>1014611</i> Lower or same priority PPI/SGI may not be sent until after deactivate of previous interrupt	15
<i>1003544</i> PPI does not mask out GRPMOD when GICD_CTLR.DS=1	28
<i>996333</i> Disabled Secure SPIs blocking LPIs	16
<i>990914</i> Issues around PT triggering affecting LPI delivery	17
<i>988378</i> Target cache presenting incorrect priorities when DS is set for filter	18
<i>975678</i> Broadcast GEN_SGI generates interrupt for the source CPU when >10 CPUs	20
<i>963506</i> Deactivate - Activate ordering violation may leave interrupt not active	29
<i>957913</i> Interrupt on Target_Cache Return interface can be missed by ITS commands	19
<i>931156</i> ITS not ignoring corrupted RDATA when ITS receives an ERROR response	21
<i>930020</i> IRQCR accesses trigger out of range SPI block software errors	30
<i>907913</i> PT coarse map may be written back when invalid following sleep completion and wakeup in PT	20

This document covers all errata from the r0p2 release.

Introduction

Scope

This document describes errata categorized by level of severity. Each description includes:

- The current status of the erratum.
- Where the implementation deviates from the specification and the conditions required for erroneous behavior to occur.
- The implications of the erratum with respect to typical applications.
- The application and limitations of a workaround where possible.

Categorization of errata

Errata are split into three levels of severity and further qualified as common or rare:

Category A A critical error. No workaround is available or workarounds are impactful. The error is likely to be common for many systems and applications.

Category A (Rare) A critical error. No workaround is available or workarounds are impactful. The error is likely to be rare for most systems and applications. Rare is determined by analysis, verification and usage.

Category B A significant error or a critical error with an acceptable workaround. The error is likely to be common for many systems and applications.

Category B (Rare) A significant error or a critical error with an acceptable workaround. The error is likely to be rare for most systems and applications. Rare is determined by analysis, verification and usage.

Category C A minor error.

Change control

Errata are listed in this section if they are new to the document, or marked as “updated” if there has been any change to the erratum text. Fixed errata are not shown as updated unless the erratum text has changed. The errata summary table on page 9 identifies errata that have been fixed in each product revision.

02-Aug-2019: Changes in document version 9.0

ID	Status	Area	Cat	Summary of erratum
1494863	New	Programmer	CatB	SPI recall failure without subsequent trigger

25-Apr-2019: Changes in document version 8.0

ID	Status	Area	Cat	Summary of erratum
1451068	New	Programmer	CatC	DCHIPR reads 0 from chips that are not the default owner

08-Feb-2019: Changes in document version 7.0

ID	Status	Area	Cat	Summary of erratum
No new or updated errata in this document version.				

30-Nov-2018: Changes in document version 6.0

ID	Status	Area	Cat	Summary of erratum
No new or updated errata in this document version.				

25-Jul-2018: Changes in document version 5.0

ID	Status	Area	Cat	Summary of erratum
1196726	New	Programmer	CatB	CLEAR command may continue after a SYNC

14-Jun-2018: Changes in document version 4.0

ID	Status	Area	Cat	Summary of erratum
No new or updated errata in this document version.				

23-Apr-2018: Changes in document version 3.0

ID	Status	Area	Cat	Summary of erratum
1118145	New	Programmer	CatB	QACTIVE may not be driven HIGH on qreqn_its_<x> transition
1109171	New	Programmer	CatC	Possible SPI corruption if GICD_IERRR<n> written to 1 without being read first in multi-chip configs
1109146	New	Programmer	CatB	GICD_CTLR.RWP may be stuck at 1 if already 1 when rt_owner changes
1104459	New	Programmer	CatC	ITS debug overflow not recorded in trace registers

1098957	New	Programmer	CatC	UNPREDICTABLE behavior on failed attempt to connect chips
1071611	New	Programmer	CatC	DPG might allow a single 1ofN interrupt to be sent after being set

12-Jan-2018: Changes in document version 2.0

ID	Status	Area	Cat	Summary of erratum
No new or updated errata in this document version.				

11-Jan-2018: Changes in document version 1.0

ID	Status	Area	Cat	Summary of erratum
1049857	New	Programmer	CatC	PMU filtering for Deactivate Event incorrect
1014611	New	Programmer	CatB	Lower or same priority PPI/SGI may not be sent until after deactivate of previous interrupt
1003544	New	Programmer	CatC	PPI does not mask out GRPMOD when GICD_CTLR.DS=1
996333	New	Programmer	CatB	Disabled Secure SPIs blocking LPIs
990914	New	Programmer	CatB	Issues around PT triggering affecting LPI delivery
988378	New	Programmer	CatB	Target cache presenting incorrect priorities when DS is set for filter
975678	New	Programmer	CatB (rare)	Broadcast GEN_SGI generates interrupt for the source CPU when >10 CPUs
963506	New	Programmer	CatC	Deactivate - Activate ordering violation may leave interrupt not active
957913	New	Programmer	CatB	Interrupt on Target_Cache Return interface can be missed by ITS commands
931156	New	Programmer	CatB (rare)	ITS not ignoring corrupted RDATA when ITS receives an ERROR response
930020	New	Programmer	CatC	IRQCR accesses trigger out of range SPI block software errors
907913	New	Programmer	CatB	PT coarse map may be written back when invalid following sleep completion and wakeup in PT

Errata summary table

The errata associated with this product affect product versions as below.

ID	Cat	Summary	Found in versions	Fixed in version
1494863	CatB	SPI recall failure without subsequent trigger	r0p2, r0p3, r1p1, r1p2, r1p3, r1p4, r1p5, r1p6	
1451068	CatC	DCHIPR reads 0 from chips that are not the default owner	r1p2, r1p3, r1p4, r1p6	
1335020	CatC	Targeted Cross-Chip SGIs incorrectly generate a SGI_OOR error on the source chip	r1p2, r1p3, r1p4	r1p6
1196726	CatB	CLEAR command may continue after a SYNC	r0p2, r0p3, r1p2, r1p3	r1p4
1118145	CatB	QACTIVE may not be driven HIGH on qreqn_its_<x> transition	r1p2	r1p3
1109171	CatC	Possible SPI corruption if GICD_IERRR<n> written to 1 without being read first in multi-chip configs	r1p2	r1p3
1109146	CatB	GICD_CTLR.RWP may be stuck at 1 if already 1 when rt_owner changes	r1p2	r1p3
1104459	CatC	ITS debug overflow not recorded in trace registers	r0p2, r0p3, r1p2	r1p3
1098957	CatC	UNPREDICTABLE behavior on failed attempt to connect chips	r1p2	r1p3
1071611	CatC	DPG might allow a single 1ofN interrupt to be sent after being set	r0p2, r0p3	r1p2
1049857	CatC	PMU filtering for Deactivate Event incorrect	r0p2, r0p3	r1p2
1014611	CatB	Lower or same priority PPI/SGI may not be sent until after deactivate of previous interrupt	r0p2, r0p3	r1p2
1003544	CatC	PPI does not mask out GRPMOD when GICD_CTLR.DS=1	r0p2	r0p3
996333	CatB	Disabled Secure SPIs blocking LPIs	r0p2	r0p3
990914	CatB	Issues around PT triggering affecting LPI delivery	r0p2	r0p3
988378	CatB	Target cache presenting incorrect priorities when DS is set for filter	r0p2	r0p3
975678	CatB (rare)	Broadcast GEN_SGI generates interrupt for the source CPU when >10 CPUs	r0p2	r0p3
963506	CatC	Deactivate - Activate ordering violation may leave interrupt not active	r0p2	r0p3
957913	CatB	Interrupt on Target_Cache Return interface can be missed by ITS commands	r0p2	r0p3
931156	CatB (rare)	ITS not ignoring corrupted RDATA when ITS receives an ERROR response	r0p2	r0p3
930020	CatC	IRQCR accesses trigger out of range SPI block software errors	r0p2	r0p3

ID	Cat	Summary	Found in versions	Fixed in version
907913	CatB	PT coarse map may be written back when invalid following sleep completion and wakeup in PT	r0p2	r0p3

Errata descriptions

Category A

There are no errata in this category.

Category A (rare)

There are no errata in this category.

Category B

1494863

SPI recall failure without subsequent trigger

Description

If an interrupt is identified as one of the top priority enabled interrupt for a CPU and sent to the target cache then there is a single cycle window where if the interrupt is reprogrammed then the recall requirement is logged but not performed until the next external trigger.

Triggers can be any of:

- 1) activate/release/deactivate of any SPI
- 2) wire toggle on any SPI
- 3) register programming or cpu group enable change

Implication

There are 2 implications:

- 1) If the register being programmed is IROUTER then the ACE-Lite slave interface may not respond until the next trigger occurs or the interrupt is serviced by the CPU.
- 2) If other registers (other than ICENABLER) is being programmed then the time when old programming is used may be extended until after the next trigger. It will not use old programming more than once or go back to old programming once the new has been used.

Workaround

Disable SPIs by writing ICENABLER<n> before reprogramming them especially if rerouting them (i.e. programming GICD_IROUTER)

This ensures that all reprogramming of the SPI is atomic and is the standard behaviour of the linux driver.

1196726**CLEAR command may continue after a SYNC****Description**

A `SYNC` command indicates that all previous ITS commands have taken effect and that current ITS programming will apply to all new incoming GITS_TRANSLATER writes.

To limit the delay to software, the GIC acknowledges all commands as soon as possible.

Sometimes, under heavy load, when executing a `CLEAR` command followed by a `SYNC` command, it is possible for new interrupts to be cleared erroneously.

Implications

The GIC might clear the pending state of interrupts that arrive shortly after the `SYNC` command, for interrupts that match the previous `CLEAR` command.

Workaround

Insert an `INV` command between the `CLEAR` and `SYNC`

```
CLEAR  
INV  
SYNC
```

1118145

QACTIVE may not be driven HIGH on qreqn_its_<n> transition

Description

The errata does not impact monolithic configurations where the ITS and Distributor share a slave port.

The Distributor of the GIC-600 has a number of Q-Channels.

qreqn is used for hierarchically clock gating the clock to the Distributor.

qreqn_its_<n> are used for power control and disconnecting the interfaces between the Distributor and each ITS.

If a transition occurs on a **qreqn_its_<n>** while the main interface is in low power mode and the clock is stopped x cycles after the **qreqn_its_<n>**, the **qactive** may not be asserted. x is the depth of the implemented CDC synchronizer.

Implications

GIC will not respond to the **qreqn_its_<n>** transition until the clock is restarted and may not raise **qactive**.

Workaround

There are two workarounds:

- 1) Do not stop the clock while completing transitions on **qreqn_its_<n>**.
- 2) Write to GICD_FCTLR.Q_DENY to prevent the main Q-Channel from accepting entry requests until the ITS powerdown sequence is complete.

1109146**GICD_CTLR.RWP may be stuck at 1 if already 1 when rt_owner changes****Description**

This issue only affects multi-chip configurations.

GIC-600 has a Routing Table (RT) that controls the partitioning of SPIs and chip address routing information. A single chip is defined as the owner of that table at any one time (the `rt_owner`).

The GIC supports moving that owner between chips so that the `rt_owner` is the last powered on chip.

If `GICD_CTLR.RWP` is 1 due to an SPI being recalled by the old `rt_owner` or by a `GICD_CTLR` update, when the `rt_owner` is changed then RWP may never drop.

Implications

RWP will remain high and will not drop causing poll timeouts.

Workarounds

There are 2 possible workarounds:

- 1) Never change the `rt_owner` and ensure that the first chip on is the last chip off.
- 2) Ensure that no processor can access the Distributor while the change is taking place and that `RWP == 0` before commencing the change.

1014611**Lower or same priority PPI/SGI may not be sent until after deactivate of previous interrupt****Description**

If there are 2 pending enabled interrupts (PPIs or SGIs) for a CPU then if:

- 1) A search trigger occurs due to a control packet, deactivate or an edge on any **ppi<x>** wire
- 2) An activate occurs from the CPU.
- 3) The second highest priority interrupt may not be sent until the next trigger.

Higher priority interrupts will always be delivered.

Implication

PPIs and SGIs of the same or lower priority may not be delivered until after the deactivate of the activated interrupt.

Workaround

If you want to be certain of receiving PPI and SGIs of the same or lower priority then issue a deactivate to any non-active PPI or SGI on the same CPU with the same security group

996333**Disabled Secure SPIs blocking LPIs****Description**

If the Secure CPU group enables, GICD_CTLR.EnableSG0 or GICD_CTLR.EnableSG1, are disabled while there are pending and enabled Secure SPIs waiting to be delivered to the CPU then the GIC will not accept an LPI into the slot unless its priority exceeds that of the Secure SPI.

If all 5 slots are in this state, then LPIs will not be delivered until an SPI (of any group) arrives and is serviced.

Implications

LPIs get stuck behind higher priority but disabled interrupts.

Workaround

Disable Secure SPIs using GICD_ICENABLER before disabling Secure CPU group enables, GICD_CTLR.EnableSG0 or GICD_CTLR.EnableSG1.

990914

Issues around PT triggering affecting LPI delivery

Description

Under certain conditions a search trigger event may be missed resulting in LPIs in the Pending Table for a CPU that falls outside of the search pointers not to be considered for delivery until the next trigger event occurs.

LPIs in the cache are not affected and will continue to be delivered. They will also cause trigger events which should retrigger the PT and restart the search.

Implications

Interrupts may be left in the PT until the next trigger event occurs.

Some additional odd behavior may be seen in LPI prioritization. However, once interrupts have spilled to the PT the latency of those interrupts will inevitably be increased due to additional latency of searching for and extracting those interrupts from the main memory.

Workaround

There are two parts to the workaround. The first significantly reduces the probability of hitting the preconditions for very low cost. The second ensures that if hit, the GIC will recover:

Part 1

Set GICD_FCTLR.AT (bit 24). This modifies how the TGT\$ triggers and will have a significant impact in reducing the probability of hitting the defect. The impact of setting this is minimal and Arm recommends it is always set.

Part2

The GIC can be poked to re-search on a regular interval. This could be done using either an external timer or by using the PMU to generate an SPI.

Periodically performing any of the trigger events described above will restart the search.

Note that this only needs to be done if any interrupts have been written to the PT and this can also be detected by using the PMU.

988378**Target cache presenting incorrect priorities when DS is set for filter****Description**

In configurations where GICD_CTLR.DS is set then the priority is not correctly shifted when prioritizing LPIs.

Implications

Interrupt prioritization will not be as expected.

Workaround

Use priority 00 for LPIs

957913

Interrupt on Target_Cache Return interface can be missed by ITS commands

Description

When an ITS command occurs, it is supposed to impact all relevant LPIs in the system. However, there is a chance that an interrupt will be missed if the following conditions are met:

1. There are 3 unique LPIs for a particular PE in the Target Cache (TC).
Note for this to happen one of them must be highest priority interrupt for the PE.
2. A higher priority SPI arrives and is sent to the PE and evicts the LPI from the PE.

In these circumstances the TC returns one of the lowest priority LPIs back to the LPI cache.

If there is an ITS command that impacts the LPI being evicted by the TGT cache and the following events are also true:

1. There are no other interrupts for the PE in the cache.
2. There have been at least 4 other unique LPIs that do not hit in the LPI cache and are currently fetching properties as they missed in the LPI cache (that is, they all arrived within the last `t_memory_access` time).
3. The search of the cache (as required by the command) completes before an LFA frees up.

If all of the above conditions align, then the single interrupt on the TC Return path may be missed by the command.

Note that disabled interrupts cannot be affected by this errata so the action of enabling a disabled/masked interrupt via INV cannot fail.

Implications

The implications of missing the command are as follows:

CLR/DISCARD: The interrupt may remain pending on the TGT resulting in a single spurious interrupt to the current PE.

INV: The interrupt may maintain and cache its old properties meaning that future interrupts may remain enabled or take an old priority.

MOV: The interrupt may remain on the old target and not be moved.

Workaround

CLR/DISCARD: The spurious interrupt should be ignored.

INV: Repeating the INV will drastically reduce (but not remove) the probability of hitting this defect. Note that as enables will always work so interrupts should always be delivered eventually.

MOV/MOVALL: The only failsafe mechanism is to insert an INT command after the MOV. This will ensure that interrupts are never missed at the cost of a spurious interrupt.

907913**PT coarse map may be written back when invalid following sleep completion and wakeup in PT****Description**

If GICD_CTLR is updated, in a short window as sleep is completing, then the coarse map is left in dirty state. If a new interrupt arrives via PT set then the coarse map could be overwritten.

Implications

Loss of interrupts.

Workaround

Poll GICR_WAKER.Quiescent before changing GICD_CTLR.
If aborting sleep do INV + SYNC before enabling GICD_CTLR.

Category B (rare)

975678**Broadcast GEN_SGI generates interrupt for the source CPU when >10 CPUs****Description**

Broadcast SGIs for sent from CPUs with MPIDR values of:

0.0.0.10
0.0.0.11
0.0.0.12
0.0.0.13

will send to themselves as well as all other CPUs.

Implications

A spurious SGI will be received.

Workaround

Ignore the spurious interrupt or issue targeted SGIs instead of the broadcast.

931156

ITS not ignoring corrupted RDATA when ITS receives an ERROR response

Description

If the ITS issues a read which is returned with an error from the memory system then it reports the error via the Distributor error registers.

It should also drop the relevant interrupt and allow software to take action based on the reported error data. Instead it uses the data for translation and if the corrupted data happens to map to a valid set of translation data then a spurious interrupt may be created.

Implications

If the corrupted data maps to valid data then it will be used in translation and potentially cached.

Workaround

If a memory access is reported via the GICD software error record, then software should flush the ITS caches via GITS_FCTLR.

Category C

1451068

DCHIPR reads 0 from chips that are not the default owner

Description

The DCHIPR register contains 2 fields

- 1) RTOwner - Identifies the chip that is the current owner of the *Routing Table* (RT), also known as Default Owner or Crown Socket.
- 2) PUP - Indicates that RTOwner is performing an update.

The only reason for writing this register is to change the RTOwner, to allow for the RTOwner Socket to be powered down.

Operations to move the owner will complete successfully, however PUP may incorrectly indicate that the update has completed while it is still in progress.

Implications

The software receives an incorrect indication that PUP has dropped early.

If the software attempts to start a new access, for example chip offline, before PUP has completed, then the new access is rejected.

Rejected writes to update chip state are detectable, as the relevant CHIPR register do not record the updated value.

Workaround

If changing the RT Owner is required, then the software should poll DCHIPR on both the new and old sockets to ensure that PUP is recorded as 0 on both before attempting further RT operations.

1335020

Targeted Cross-Chip SGIs incorrectly generate a SGI_OOR error on the source chip

Description

An targeted (non-broadcast) SGI targeted at a remote chip will generate an SGI OOR error which will be reported in T&D Record 0 of the source chip.

The SGIs is still delivered correctly.

Implication

Error Record 0 will be flooded with false SGI errors. This may mask other real software errors that are generating as only a single error is recorded unless the record is cleared.

If record 0 is programmed to generate a fault_handling or error_recovery interrupt then spurious error interrupts will be generated.

There is no impact on tracking ECC or ITS software errors and no impact on the architectural operation of the GIC.

Workaround

Any of the following workarounds can be applied:

- a) Do not enable the interrupts generated from error record 0 and ignore error record 0. (This is the reset behaviour of the GIC)
- b) If the interrupts are enabled clear error record 0 after every remote SGI (based on receiving the error interrupt)
- c) If interrupts are enabled (or software plans to periodically check the error record status registers) during software debug then trap writes to the SGI generation registers and disable before generating each Cross-Chip targeted SGI.

1109171**Possible SPI corruption if GICD_IERRR<n> written to 1 without being read first in multi-chip configs****Description**

This does not impact single-chip configurations.

GIC-600 contains a register GICD_IERRR<n> which is involved in containment and correction of SPI after an uncorrectable RAM error occurs.

If an attempt is made to clear an error while the GIC is attempting to send the SPI to another chip, then the error clear may be successful when the SPI is not yet in a safe state.

This behavior only occurs if both the following conditions occur:

- 1) The error on the SPI occurs in the small window between deciding to send an SPI to another chip.
- 2) Software does not read GICD_IERRR<n> to observe the error before attempting to clear it using a GICR_IERRR<n> write.

Implications

Behavior of that SPI is UNPREDICTABLE.

Workaround

No software updates should be required because software should not attempt to clear errors not identified by reads to GICD_IERRR<n>.

1104459**ITS debug overflow not recorded in trace registers****Description**

This only impacts configurations with an ITS.

If more than one translation or command error occurs in a single ITS within a very short time window, then the overflow bit may not be correctly recorded.

Implications

The ITS error record may not indicate an overflow when multiple errors are detected.

Workaround

None is required because software should be programming the GIC to prevent errors from occurring.

1098957**UNPREDICTABLE behavior on failed attempt to connect chips****Description**

This does not impact single chip configurations.

GIC-600 has a connection process for connecting chips together under the control of Secure software.

If an attempt is made to connect to a chip in an unsuitable state the GIC is supposed to reject the connection and return to a known good state. However, the GIC fails to store the address to return the response and may respond to the incorrect address.

Unsuitable states include being already connected, out-of-range or GICD_CLTR values are incorrect.

Implications

Behavior is UNPREDICTABLE and can include any of:

- 1) Failure to send a response to the correct address.
- 2) Deadlock.
- 3) Loss of coherence between chips.

Workaround

No software change is expected as software should only ever be attempting to connect chips in the correct state.

1071611**DPG might allow a single 1ofN interrupt to be sent after being set****Description**

A single 1ofN interrupt might ignore GICR_CTLR.DPG settings if the following events occur:

- 1) DPG is programmed while the CPU group enables are enabled.
- 2) There are multiple 1ofN SPIs that have been routed to CPUs that are processing other higher priority interrupts.

Implications

A GIC might deliver an unexpected but valid 1ofN SPI interrupt to a CPU that should not receive it.

Workaround

No workaround is expected to be required as the only impact is a spurious interrupt to a CPU that is able to accept it.

However, the issue can be prevented by only changing DPG when the CPU groups enables are off.

1049857**PMU filtering for Deactivate Event incorrect****Description**

The PMU UP_DEACT Event is intended to allow tracking of SPI deactivates sent to the GIC from the CPUs.

The PMU will correctly count unfiltered deactivates, however, attempting to count deactivates filtered by CPU (or group of CPUs) uses incorrect CPU information and will give an invalid result.

Implications

Filtered PMU count of Deactivates will be incorrect.

Workaround

Use an unfiltered counter or count Deactivates via code added in the ISR.

1003544**PPI does not mask out GRPMOD when GICD_CTLR.DS=1****Description**

This only impact systems with a single security state: GICD_CTLR.DS == 1

The PPI block considers GRPMOD when determining if the CPU group is enabled for a particular interrupt.

Implications

Interrupts with GRPMOD == 1 and GRP == 0 will not be delivered.

There are two mechanisms for GRPMOD getting set:

- 1) Programming of GICR_GRPMODR before GICD_CTLR.DS is set. This requires GICD_CTLR.DS to be configured as programmable.
- 2) Uncorrectable errors corrupting the GRPMOD bit in the PPI RAM cannot be corrected once DS is set. This requires corruption and will slightly decrease the MTBF.

Workaround

The workaround for 1) is to program GICD_GRPMODR to 0 before setting DS. 0 is the reset value of GICD_GRPMOD and there is no known use case for programming the GIC with 2 security levels and then disabling security.

If the RAM is corrupted with an uncorrectable error which results in GICD_GRPMODR == 1 for an interrupt, then there is no way to reprogram it and the only way to recover is to:

- 1) Reprogram and use the interrupt as a Group1 interrupt.
- 2) Reset the PPI block following powerdown sequence and reprogram.

963506

Deactivate - Activate ordering violation may leave interrupt not active

Description

If a Deactivate is issued over the GIC-Stream interface followed by an Activate to the same SGI or PPI within 3 GIC cycles then the Deactivate may be processed after the Activate.

This would leave the interrupt with Active = 0, meaning that it could be delivered again if it is pending or becomes pending.

Writes to GICR<x>_I(S|C)Active are not affected by this issue.

Interrupts cannot remain masked by this issue.

Software should never get into this state because under normal operation it should only deactivate interrupts that it has already activated and cannot, therefore, be activated again.

Implications

The SGI or PPI will remain inactive and may be delivered again. This could result in a spurious interrupt.

Workaround

A DSB after Deactivating an interrupt would avoid the issue but no workaround is believed to be necessary.

930020**IRQCR accesses trigger out of range SPI block software errors****Description**

In configurations with < 8 SPI blocks (256 SPIs) then an OOR block error will be generated when GICP_IRQCR is programmed to internally route the PMU interrupt to an SPI.

and

In configurations with < 7 SPI blocks (224 SPIs) then an OOR block error will be generated when GICT_ERRIRQCR<n> is programmed to internally route the error handling or fault recovery interrupt to an SPI.

Implications

An OOR block error will be generated in RAS error record 0.

Workaround

Clear (and ignore) the incorrectly generated error after programming.