Arm® CoreLink™ CCI-500 Cache Coherent Interconnect

Software Developer Errata Notice

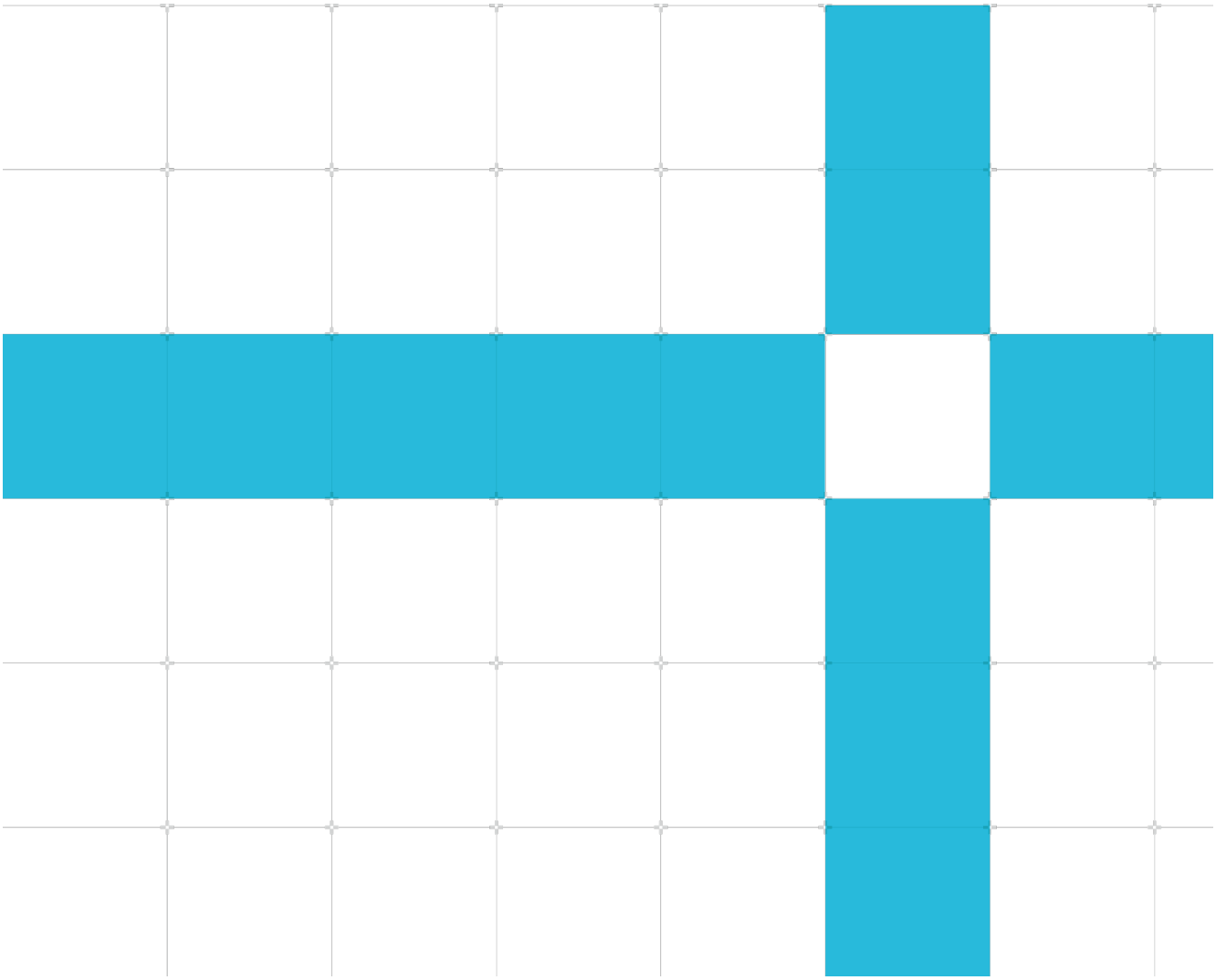
Date of issue: 24-Jun-2022

Non-Confidential

Document version: v5.0

Document ID: SDEN-2287446

This document contains all known errata since the r1p0 release of the product.



Non-confidential proprietary notice

(LES-PRE-20349)

Confidentiality status

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

Unrestricted Access is an Arm internal classification.

Feedback

Arm welcomes feedback on this product and its documentation. To provide feedback on Arm® CoreLink™ CCI-500 Cache Coherent Interconnect, create a ticket on **https://support.developer.arm.com**.

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

Inclusive language commitment

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

If you find offensive language in this document, please email **terms@arm.com**.

Contents

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** identifies errata that have been fixed in each product revision.

24-Jun-2022: Changes in document version v5.0

ID	Status	Area	Category	Summary
682058	New	Programmer	Category A (rare)	High rate of back-invalidations can cause coherency failure, data corruption and deadlock
466840	New	Programmer	Category B	Possible self-starvation when multiple threads using single slave interface
476523	New	Programmer	Category B	Possible deadlock with small SIx_RW_MAX and large TT_DEPTH parameters
598316	New	Programmer	Category B	Possible starvation of snoop requests
608524	New	Programmer	Category B	Possible starvation of read data within a slave interface
626366	New	Programmer	Category B	Possible starvation of IO-coherent writes to same address
574650	New	Programmer	Category C	QOSACCEPT signaling doesn't work correctly when equal to high priority threshold
2220131	New	Programmer	Category C	Ordered Write Observation is not enforced between shareable and non-shareable writes

24-Jun-2022: Changes in document version v4.0

No new or updated errata in this document version.

24-Jun-2022: Changes in document version v3.0

No new or updated errata in this document version.

24-Jun-2022: Changes in document version v2.0

No new or updated errata in this document version.

18-Oct-2016: Changes in document version v1.0

No errata in this document version.

Errata summary table

The errata associated with this product affect the product versions described in the following table.

ID	Area	Category	Summary	Found in versions	Fixed in version
682058	Programmer	Category A (rare)	High rate of back-invalidations can cause coherency failure, data corruption and deadlock	r0p0, r0p1, r0p2	r1p0
626366	Programmer	Category B	Possible starvation of IO-coherent writes to same address	r0p0, r0p1, r0p2	r1p0
598316	Programmer	Category B	Possible starvation of snoop requests	r0p0, r0p1, r0p2	r1p0
608524	Programmer	Category B	Possible starvation of read data within a slave interface	r0p0, r0p1, r0p2	r1p0
466840	Programmer	Category B	Possible self-starvation when multiple threads using single slave interface	r0p0	r0p1, r0p2, r1p0
476523	Programmer	Category B	Possible deadlock with small SIx_RW_MAX and large TT_DEPTH parameters	r0p0	r0p1, r0p2, r1p0
2220131	Programmer	Category C	Ordered Write Observation is not enforced between shareable and non-shareable writes	r0p0, r0p1, r0p2, r1p0	Open
574650	Programmer	Category C	QOSACCEPT signaling doesn't work correctly when equal to high priority threshold	r0p2, r1p0	Open

Errata descriptions

Category A

There are no errata in this category.

Category A (rare)

682058
High rate of back-invalidations can cause coherency failure, data corruption and deadlock

856571: Old ID for cross-referencing with previous errata documents.

Status

- Affects: CCI-500.
- Fault Type: Programmer, Category A (rare).
- Fault Status: Present in: r0p0, r0p1, r0p2. Fixed in: r1p0.

Description

A rare set of stimulus has been found which can cause a logical failure in the CCI. This can lead to coherency failure, data corruption and deadlock. This was found using directed stimulus which generates a high bandwidth of back-invalidate (BI) requests.

Configurations Affected

All configurations are affected, but the condition requires at least 7 BI requests to be in flight, so is more likely in systems with an undersized snoop filter.

Conditions

The defect is caused by the following sequence of events:

1. A shareable request (A) is received for which there is a conflict for allocation in the snoop filter (SF), the probability of this is dependent on the ratio of snoop filter size to size of tracked caches.
2. The SF generates a BI request to remove any corresponding entry in coherent cache that is monitored by CCI
3. A de-allocating shareable request is received to the same cache line as (A) and is scheduled before the BI. De-allocating requests are: WriteBack, Evict, CleanInvalid, MakeInvalid, WriteUnique, WriteLineUnique. This is expected to be rare.
4. The BI misses in the SF and queues at the BI arbiter.
5. There are at least 6 other BI requests in flight which are granted before the BI miss. This requires a high bandwidth burst of BI requests which is expected to be very rare in systems where the snoop filter is sized according to ARM's recommendation.

Implications

If this condition arises, messages between the sub-blocks in the CCI can be misinterpreted. The implications depend on the exact timing, but range from no effect to protocol violation and deadlock.

Workaround

This defect cannot occur if coherency is disabled, the snoop filter is disabled or there is a low rate of back-invalidates in your system. Disabling coherency or the snoop filter is not practical is most systems. You can count the number of back-invalidates in your implementation using the CCI PMU, global event number [8].

Defect Fix

This defect is fixed in the r1p0 release of the CCI.

Category B

626366
Possible starvation of IO-coherent writes to same address

855071: Old ID for cross-referencing with previous errata documents.

Status

- Affects: CCI-500.
- Fault Type: Programmer, Category B.
- Fault Status: Present in: r0p0, r0p1, r0p2. Fixed in: r1p0.

Description

The CCI-500 has a Point of Serialization (PoS) that ensures that writes to the same memory location by two components are observable in the same order by all components. The PoS compares the cache line address of a coherent transaction request with that of every coherent transaction which has previously been serialized and has not completed. If there is already a transaction in progress to the same cache line, then the request is stalled and re-checked on a later cycle.

The CCI-500 has a central Transaction Tracker (TT) that queues transaction requests and arbitrates between requests that need to retry the PoS check. The arbitration logic permits a request to retry the PoS check only after the completion of at least one other transaction that has previously caused a request to fail the PoS check.

Because of this erratum, WriteUnique (WU) and WriteLineUnique (WLU) requests can retry the PoS check every 3 cycles without dependence on completion of another transaction. This can result in starvation of coherent requests from one master if there is a continuous stream of WU or WLU requests to the same address from at least two other masters.

Configurations Affected

All configurations of the CCI are affected.

Conditions

The starvation can occur under both of the following conditions:

- A steam of shareable writes (WU or WLU) from two masters, where all target the same 64-byte cache line address
- Another coherent request, from any master, to the same 64-byte cache line address

The starvation persists as long as the two masters continue to send shareable writes that target the same cache line address. If one of the requesting masters stalls or stops issuing shareable writes to that address, the starvation desists.

The following ARM processors never issue WU or WLU requests so cannot trigger this defect:

- Cortex-A7
- Cortex-A17
- Cortex-A35
- Cortex-A53
- Artemis

The Cortex-A57 and Cortex-A72 processors are configured by default to not issue WU or WLU transactions. You can use the L2 Auxiliary Control Register (L2ACTLR) to enable these transactions, but ARM does not recommend doing this when these processors are used with a CCI.

ARM Mali GPUs can issue WU and WLU transactions. However, generating a long sequence of these transactions to the same cache line address would require an unusual application.

Implications

The implication of this defect is that some coherent requests might be delayed for a significant time in the presence of heavy snoop traffic. This might have a detrimental effect on the performance of the affected master.

Workaround

There is no workaround, but streams of high-bandwidth coherent writes from more than one master to the same cache line address are expected to be rare.

598316
Possible starvation of snoop requests

854176: Old ID for cross-referencing with previous errata documents.

Status

- Affects: CCI-500.
- Fault Type: Programmer, Category B.
- Fault Status: Present in: r0p0, r0p1, r0p2. Fixed in: r1p0.

Description

The CCI-500 has a central transaction tracker (TT), that queues transaction requests and issues AC requests to the snoop network if necessary for shareable read, write or DVM messages. The TT is split into two partitions, based on an address hash, and DVM requests alternate between partitions. The snoop network can transfer one AC request each cycle, so there is an arbiter to select between TT partitions each cycle.

Because of this erratum, the AC arbiter does not change the grant in the case of contention. This can result in starvation of AC requests from one partition if there is a continuous stream of requests from the other partition.

Configurations Affected

All configurations of the CCI are affected.

Conditions

The starvation is set up by a stream of requests from one or more masters. These could be any of the following:

- Shareable reads that hit in the snoop filter, i.e. they are expected to hit in a processor cache.
- Shareable writes (WriteUnique or WriteLineUnique) that hit in the snoop filter.
- DVM messages that need to be broadcast.
  The starvation persists as long as one partition has a request waiting to send to the snoop network. If the requesting master stalls or stops issuing transactions that need to snoop, then the starvation desists.

Implications

The implication of this defect is that some requests that are expected to hit in a processor might be delayed for a significant time in the presence of heavy snoop traffic. This might have a detrimental effect on the performance of the affected master.

Workaround

There is no workaround, but cases of high bandwidth snoop hit traffic are expected to be rare in most use-cases.

608524
Possible starvation of read data within a slave interface

854524: Old ID for cross-referencing with previous errata documents.

Status

- Affects: CCI-500.
- Fault Type: Programmer, Category B.
- Fault Status: Present in: r0p0, r0p1, r0p2. Fixed in: r1p0.

Description

At a slave interface, read data can be sent from any of the master interfaces which have been sent read requests. Each slave interface has a number of arbiters which determine the order in which data is returned through the slave interface in the case of contention.
Because of this erratum, read data from some master interfaces can prevent the progress of read data from other master interfaces through the same slave interface.

Configurations Affected

This erratum affects all slave interfaces in configurations where there are a total of 2 master interfaces. In this case:

- Read data from M0 can starve read data from M1
- Read data from M1 can starve read data from M0

Conditions

An example of a condition where this erratum can occur with a CCI configured with 2 master interfaces, M0 and M1.

1. Master attached to S0 issues a read request to system through M0
2. Master attached to S0 issues a stream of read requests through M1
3. Read data is returned from M1 in a constant stream, with no idle cycles
4. Read data is returned from M0, but is stalled inside the CCI because data from M1 is continuous
5. The master continues issuing read requests through M1 indefinitely, whilst also waiting for data from M0

Implications

The implication of this defect is that some requests may experience an unacceptably high latency, which might affect the performance of a master.

Workaround

There is no workaround from within the CCI, but the condition of this erratum requires that one flow of read request continues despite the fact that other requests using the same interface are stalled. This is unlikely to happen unless there are multiple master devices sharing a CCI slave interface.

466840
Possible self-starvation when multiple threads using single slave interface

845119: Old ID for cross-referencing with previous errata documents.

Status

- Affects: CCI-500
- Fault Type: Programmer, Category B.
- Fault Status: Present in: r0p0. Fixed in: r0p1.

Description

If a master issues a constant stream of reads using multiple IDs, then one ID stream might not receive a response until the other stream stops issuing new requests.

Conditions

At least 2 streams of reads, which may be:

- Shareable exclusive reads
- Read bursts that exceed 64-bytes in length

To cause starvation the aggressor stream needs to maintain multiple outstanding transactions to different addresses but through the same CCI master interface.

Implications

A loop in Non-secure software might cause denial-of-service of a Secure process.

Workaround

To work around this issue, it must be possible to interrupt a processor which is capable of issuing a never-ending stream of read requests.

476523
Possible deadlock with small SIx_RW_MAX and large TT_DEPTH parameters

845070: Old ID for cross-referencing with previous errata documents.

Status

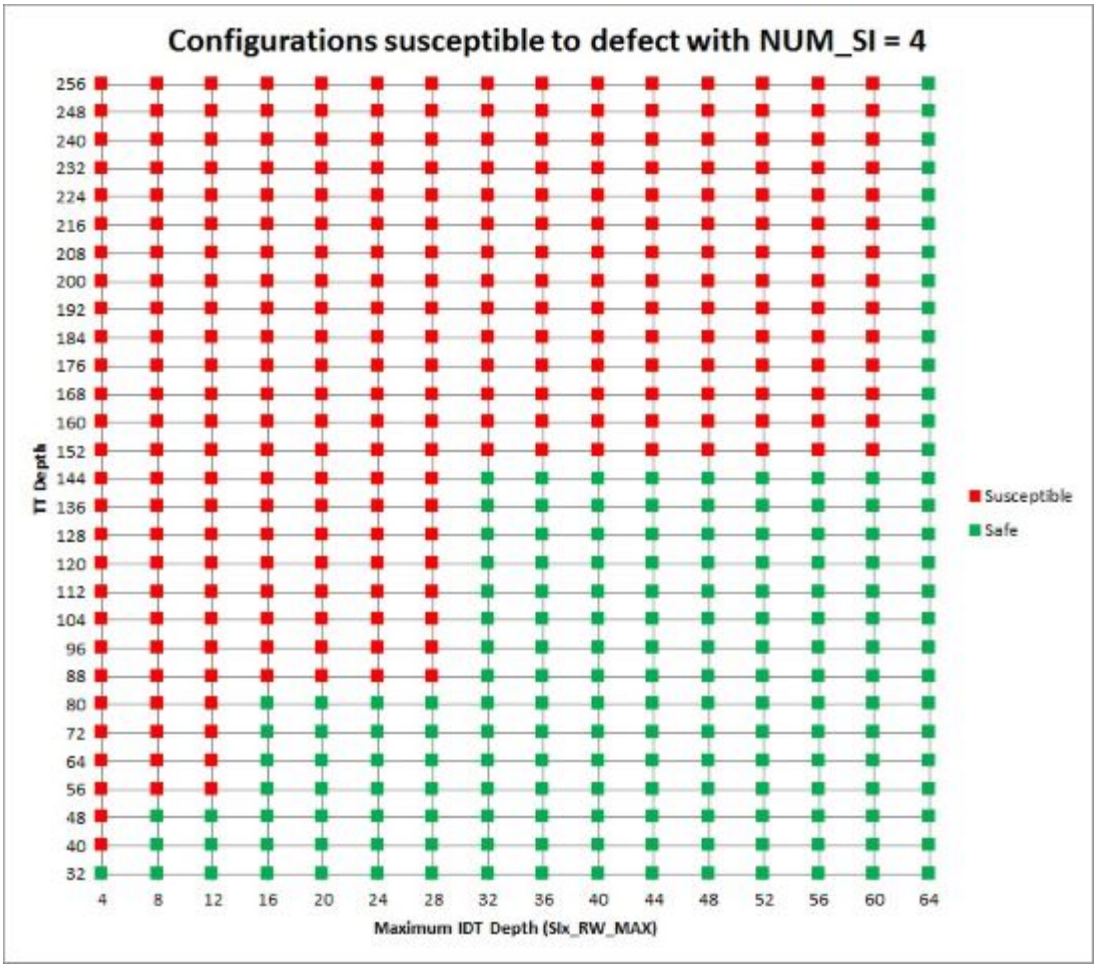
- Affects: CCI-500.
- Fault Type: Programmer, Category B.
- Fault Status: Present in: r0p0. Fixed in: r0p1.

Description

Certain configurations where the maximum ID tracker (IDT) depth is small and transaction tracker (TT) size is relatively large can cause a counter in the request concentrator (cci500_rc.v) to overflow.

Conditions

Affected combinations of maximum IDT depth and TT size are shown below. The X-axis is the maximum value across all interfaces. For example, if you have interfaces with SIx_RW_MAX parameters of 32, 32, 16, 8 then you use a value of 32 to look up which TT depth values are safe.
This example matrix is for 4x slave interfaces but can be provided for other numbers of interfaces on request.



Implications

If the counter overflows, tracking of available slots in the TT is impaired which can lead to performance degradation and deadlock.

Workaround

The only known safe run-time workaround is to disable the snoop filter, using bit[2] of the Control Override Register.
It is better to work around at design-time by choosing IDT and TT sizes that cannot trigger this defect.

Category B (rare)

There are no errata in this category.

Category C

2220131
Ordered Write Observation is not always enforced between shareable and non-shareable writes

Status

- Affects: CCI-500.
- Fault Type: Programmer, Category C.
- Fault Status: Present in: r0p0, r0p1, r0p2, r1p0. Fixed in: Open.

Description

The Ordered Write Observation AXI property requires that a write W1 is guaranteed to be observed by a write W2, where W2 is issued after W1, from the same master, with the same ID.

When Ordered Write Observation is enabled for a CCI slave interface, this property is not guaranteed if the writes do not have the same shareability and are sent to different CCI master interfaces.

Conditions

This errata can occur when the writes are issued to different CCI master interfaces and the first write is non-shareable or system shareable and the second write is inner or outer shareable.

Configuration Affected

All configurations where the **ORDERED_WRITE_OBSERVATION** input signal is asserted for one or more interfaces.

Implications

The second write to be issued can be observed before the first write is observable.

Managers using a producer/consumer ordering model cannot guarantee that the data is observable before the flag is observable.

Workaround

You can avoid this issue by doing either of the following:

- Ensuring that both writes use the same shareability, by either:

    - The manager that is issuing the write setting it
    - Using the CCI Shareable Override Register
- The manager can wait for a response from the first write before issuing the second write

574650
QOSACCEPT signaling does not work correctly when equal to high priority threshold

854525: Old ID for cross-referencing with previous errata documents.

Status

- Affects: CCI-500.
- Fault Type: Programmer, Category C.
- Fault Status: Present in: r0p2, r1p0. Fixed in: Open.

Description

The CCI-500 has a pair of inputs on each master interface, **VARQOSACCEPT[3:0]** and **VAWQOSACCEPT[3:0]**. These can be used to recommend the minimum request QOS value that the CCI sends downstream.

The CCI has programmable thresholds to determine which requests are classed as high priority. If a request has a QOS value that is greater than or equal to the threshold value, then it is treated as high priority.

The intended behavior is that if **VAxQOSACCEPT** is lower than the threshold, then the CCI sends both high and low priority requests, otherwise it sends only high priority requests.

Because of this erratum, the actual behavior is that if **VAxQOSACCEPT** is lower than or equal to the threshold, then the CCI sends both high and low priority requests, otherwise it sends only high priority requests.

The expected behavior is:

- If **VAxQOSACCEPT** is lower than the threshold, the CCI sends transactions with any QoS value
- If **VAxQOSACCEPT** is equal to the threshold, the CCI sends requests that have a QoS value equal to or greater than the threshold
- If **VAxQOSACCEPT** is greater than the threshold, the CCI sends requests that have a QoS value equal to or greater than the threshold

The actual behavior is:

- If **VAxQOSACCEPT** is lower than the threshold, the CCI sends transactions with any QoS value
- If **VAxQOSACCEPT** is equal to the threshold, the CCI sends transactions with any QoS value
- If **VAxQOSACCEPT** is greater than the threshold, the CCI sends requests that have a QoS value equal to or greater than the threshold

Configurations Affected

All configurations of the CCI are affected.

Conditions

This erratum affects the behavior whenever the **VAxQOSACCEPT** inputs are used to control the issuing of requests.
Systems where the **VAxQOSACCEPT** inputs are tied to 0 are unaffected.

Implications

The implication is that low priority transactions are issued from the CCI when they might not be expected. This could affect the service provided to high priority requests in the system.

Workaround

The workaround is to program the threshold values one value higher than you would expect.