



## CoreSight® SoC-600 Software Developer Errata Notice

This document contains all known errata since the r2p0-00eac0 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 © 2017-2019 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.

## Product Status

The information in this document is for a product in development and is not final.

## 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](mailto:errata@arm.com) giving:

- The document title.
- The document number: SDEN-1038867.
- 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>	5
<i>ERRATA SUMMARY TABLE</i>	8
<i>1565895</i> TPIU generates spurious data when stopped	9
<i>1456982</i> TPIU stops accepting trace when FFCR is written	13
<i>1456966</i> TPIU stops accepting trace when the trigger counter is written	15
<i>1434359</i> ATB Replicator DEVID register returns irrelevant field value	16
<i>1416485</i> Flushes might not complete	17
<i>1301623</i> CATU performs unnecessary scatter table reads when memory buffer wraps around after the last valid scatter list entry.	11
<i>1232238</i> Reading ETB+ETF memory via RRD is limited by CBUFLEVEL in Disabled state	12
<i>1116272</i> Flush operation is stalled indefinitely	13

# 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:

- |                          |  |
|--------------------------|--|
| <b>Category A</b>        | A critical error. No workaround is available or workarounds are impactful. The error is likely to be common for many systems and applications.   |
| <b>Category A (Rare)</b> | 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. |
| <b>Category B</b>        | A significant error or a critical error with an acceptable workaround. The error is likely to be common for many systems and applications.   |
| <b>Category B (Rare)</b> | 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.     |
| <b>Category C</b>        | 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 8 identifies errata that have been fixed in each product revision.

### 23-Oct-2019: Changes in document version 8.0

ID	Status	Area	Cat	Summary of erratum
1565895	New	Programmer	CatB	TPIU generates spurious data when stopped

### 03-May-2019: Changes in document version 7.0

ID	Status	Area	Cat	Summary of erratum
1456982	New	Programmer	CatC	TPIU stops accepting trace when FFCR is written
1456966	New	Programmer	CatC	TPIU stops accepting trace when the trigger counter is written
1434359	New	Programmer	CatC	ATB Replicator DEVID register returns irrelevant field value
1301623	New	Programmer	CatB	CATU performs unnecessary scatter table reads when memory buffer wraps around after the last valid scatter list entry.

### 12-Mar-2019: Changes in document version 6.0

ID	Status	Area	Cat	Summary of erratum
1416485	New	Programmer	CatC	Flushes might not complete

### 08-Aug-2018: Changes in document version 5.0

ID	Status	Area	Cat	Summary of erratum
1232238	New	Programmer	CatB	Reading ETB+ETF memory via RRD is limited by CBUFLEVEL in Disabled state
1116272	Updated	Programmer	CatB (rare)	Flush operation is stalled indefinitely

### 18-Jul-2018: Changes in document version 4.0

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

### 30-May-2018: Changes in document version 3.0

ID	Status	Area	Cat	Summary of erratum
1116272	New	Programmer	CatB (rare)	Flush operation is stalled indefinitely

**08-Mar-2018: Changes in document version 2.0**

ID	Status	Area	Cat	Summary of erratum
----	--------	------	-----	--------------------

No new or updated errata in this document version.

**08-Dec-2017: Changes in document version 1.0**

ID	Status	Area	Cat	Summary of erratum
----	--------	------	-----	--------------------

No errata in this document version.

## Errata summary table

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

ID	Cat	Summary	Found in versions	Fixed in version
<a href="#">1565895</a>	CatB	TPIU generates spurious data when stopped	r3p0, r3p1	
<a href="#">1456982</a>	CatC	TPIU stops accepting trace when FFCR is written	r3p0, r3p1	
<a href="#">1456966</a>	CatC	TPIU stops accepting trace when the trigger counter is written	r3p0, r3p1	
<a href="#">1434359</a>	CatC	ATB Replicator DEVID register returns irrelevant field value	r2p0, r3p0	r3p1
<a href="#">1416485</a>	CatC	Flushes might not complete	r2p0, r3p0	r3p1
<a href="#">1301623</a>	CatB	CATU performs unnecessary scatter table reads when memory buffer wraps around after the last valid scatter list entry.	r2p0, r3p0	r3p1
<a href="#">1232238</a>	CatB	Reading ETB+ETF memory via RRD is limited by CBUFLEVEL in Disabled state	r2p0, r3p0	r3p1
<a href="#">1116272</a>	CatB (rare)	Flush operation is stalled indefinitely	r2p0, r3p0	r3p1



# Errata descriptions

## Category A

---

There are no errata in this category.

## Category A (rare)

---

There are no errata in this category.

## Category B

---

### 1565895

#### TPIU generates spurious data when stopped

##### Status

Affects: CoreSight SoC-600 - Perpetual

Fault Type: Programmer Cat B

Fault Status: Present in: r3p0, r3p1 Fixed in: Open

##### Description

This erratum affects the following components:

- Trace Port Interface Unit (TPIU)
  - css600\_tpiu
  - Component Revisions: r0p0, r1p0

The TPIU generates a **tracectl** output for compatibility with legacy Trace Port Analyzers (TPAs).

The **tracectl** pin is not required when the TPIU operates in continuous mode, which should be supported by all modern TPAs.

Arm recommends that the **tracectl** pin is not exported off chip, thus saving an IO pin.

As a result of this erratum, the TPIU outputs h0001 on **tracedata[]** when stopped.

If the TPA does not see the **tracectl** pin it interprets the static **tracedata[]** value as further trace packets.

##### Conditions

1. The TPIU trace port interface to the TPA does not include the **tracectl** pin and is programmed to operate in Continuous mode.
2. The TPIU is programmed to stop, and enters the Stopped state.

##### Implications

When the TPIU stops the TPA will receive the expected trace data followed by a continuous stream of spurious trace packets.

When CSPSR is configured for 1 bit, **tracedata[0]** is static 1, which is interpreted as ATID=h7F, a reserved value.

When CSPSR is configured for N=2..32 bits, **tracedata[N-1:0]** is h0001.

This can be interpreted as new trace packets, starting with a static or changing trace ID, followed by static or changing trace data, followed by a static or changing auxiliary byte.

- When CSPSR is configured for 1 bit: Spurious trace data with ATID=h7F (reserved value).
- When CSPSR is configured for 2 bits: Spurious trace data with ATID=h00 and ATID=h2A
- When CSPSR is configured for 3 bits: Spurious trace data with ATID=h00 and ATID=h24
- When CSPSR is configured for 4 bits: Spurious trace data with ATID=h00 and ATID=h08
- When CSPSR is configured for 5 bits: Spurious trace data with ATID=h00 and ATID=h10
- When CSPSR is configured for 6 bits: Spurious trace data with ATID=h00 and ATID=h20

- When CSPSR is configured for 7 bits: Spurious trace data with ATID=h00 and ATID=h40
- When CSPSR is configured for 8 to 32 bits: Spurious trace data with ATID=h00

## Workaround

When a trigger would be used to cause the TPIU to stop, debug tools need to program the TPIU:

- To not stop, by ensuring FFCR.StopTrig=0 and FFCR.StopFl=0.
- To generate a flush on a trigger event (FFCR.FOnTrig=1),
- To generate a trigger on flush completion (FFCR.TrigFl=1).

The debug tools need to detect the trigger generated on the trace port and then decide to stop trace capture. The debug tool might need to wait for an amount of time or data to ensure enough data has been captured from the TPIU.

If a trigger packet was inserted on ATB, the debug tool has to deal with two triggers on the trace port. In such a scenario, on detecting the first trigger the debug tool should wait for an amount of time or data to ensure enough data has been captured from the TPIU, although this might not guarantee all flushed data is captured.

When any other condition would be used to stop the TPIU, the debug tool might need to manually stop capture of trace.

## 1301623

**CATU performs unnecessary scatter table reads when memory buffer wraps around after the last valid scatter list entry.**

### Status

Affects: CoreSight SoC-600

Fault Type: Programmer Category B

Fault Status: Present in: r3p0, r2p0 Fixed in: r3p1

### Description

This erratum affects the following components:

- CoreSight Address Translation Unit.
  - css600\_catu
  - Component Revisions: r0p0

The CATU performs address translation to present a view of a contiguous block of memory to a master such as the ETR, when the underlying system memory may be fragmented.

In typical use cases the master writes to the memory block sequentially as a circular buffer.

With this erratum, when the master wraps from the top of the memory buffer to the bottom, the CATU will perform N scatter-gather translation accesses for a memory buffer of size N MB. For a large buffer, this can stall the AXI slave for longer than expected.

### Conditions

The following conditions must exist:

- There are  $N > 1$  scatter gather pages.
- On the AXI slave side an access is made to an address for the last accessed 4k address region.
- Then on the AXI slave side an access is made to the first 1MB region.

### Implications

The search for the next scatter gather table requires N scatter gather page accesses rather than just one.

Further access on the AXI slave write channel is stalled until the scatter gather list search has completed.

The duration of stalling adds latency to AXI memory which may be too long to be tolerated by the system.

The extra latency to the AXI memory may affect the achievable throughput.

The stalls might cause trace sources to overflow.

### Workaround

None

## 1232238

### Reading ETB+ETF memory via RRD is limited by CBUFLEVEL in Disabled state

#### Status

Affects: CoreSight SoC-600

Fault type: Programmer Category B

Fault status: Present in r2p0, r3p0, Fixed in: r3p1

#### Description

This erratum affects the following components:

- Trace Memory Controller in ETB and ETF configurations.
  - css600\_tmc\_etb, css600\_tmc\_etf
  - Component Revisions: r0p2, r0p3

Reading from memory using RRD might not return all the data from the memory.

In the Stopped and Disabled states, reads of RRD should always return the data from where RRP is pointing to in memory.

This erratum means that the TMC only permits the amount of data to be read out that was captured since the TMC was most recently enabled. This amount is indicated in CBUFLEVEL. If there is more data in the memory than indicated by CBUFLEVEL, then this will not be read out of the memory.

When the data has been read out of the memory, the TMC returns 0xFFFFFFFF on reads of RRD, which indicates there is no more data to read.

#### Conditions

The following sequence must occur:

1. The TMC is enabled and subsequently stopped or disabled for trace capture in a first session.
2. The TMC is enabled and subsequently stopped or disabled for trace capture in a second session, without reading out the trace data for the first session.
3. A debug tool attempts to read out the data captured for both of the sessions.

If all of the captured data is read out of the TMC memory each time the TMC is disabled, then this erratum does not occur.

#### Implications

Reading the entire data is not possible when CBUFLEVEL is less than the amount of data in the memory.

#### Workaround

Under normal circumstances, when using the ETB or ETF, when entering the Stopped or Disabled states only the data which was written to the RAM since the last time the ETB or ETF became enabled can be read out using RRD. This amount of data is indicated in CBUFLEVEL, and after that amount of data has been read out, the ETB or ETF returns 0xFFFFFFFF on reads of RRD.

If a debug tool needs to read out more data, for example data that was written previous times the ETB or ETF was enabled, the following procedure must be used:

1. Choose a location in the RAM from which you do not need to read data. If no such location exists (the RAM is full), then choose the location with the oldest piece of data, and usually this is the location currently pointed to by RWP. This RAM data width sized location will be used as a scratchpad location, and will be overwritten by this workaround and will not be read from the ETB or ETF.
2. Ensure the ETB or ETF is in the Disabled state by setting CTL.TraceCaptEn to 0.
3. Read CBUFLEVEL to determine how much data can currently be read from the RAM.
4. Set RWP to the chosen scratchpad location.
5. Perform N writes to RWD, where N is the RAM data width in bytes divided by 4. E.g. for an 8-byte wide RAM, perform 2 writes. These writes increment CBUFLEVEL and RWP by the amount of data written.
6. Repeat steps 4 and 5 until CBUFLEVEL reaches the desired amount of trace that you need to read from the RAM.

7. Set RRP to the location in the RAM that you wish to read. Usually this will be the location in the memory with the oldest piece of data, unless that data was the location chosen in step 1 and in that scenario this location will be 1 RAM data width further into the buffer.
8. Perform enough reads of RRD to extract the desired amount of data.

Furthermore, each time the ETB or ETF is enabled, RRP must be set to the same value as RWP before enabling.

## Category B (rare)

---

### 1116272

#### Flush operation is stalled indefinitely

##### Status

Affects: CoreSight SoC-600

Fault Type: Programmer Category B (rare)

Fault Status: Present in: r2p0, r3p0, Fixed in: r3p1

##### Description

This erratum affects the following components:

- Trace Memory Controller in ETB, ETR, ETF and ETS configurations.
  - css600\_tmc\_etb, css600\_tmc\_etr, css600\_tmc\_etf, css600\_tmc\_ets
  - Component Revisions: r0p2, r0p3

A flush is requested and more than one trigger sources are generated in one session.

With this erratum, a flush operation will get stuck after the upstream flush.

TMC ignores further trigger, flush and stop requests. It would not be able to enter into STOPPED or DISABLED state.

##### Conditions

The following conditions must exist:

- Both types of trigger sources are received in one session i.e. atid\_s=7D is received on ATB slave interface and a trigin is asserted on the event interface.
- A flush source is received and an upstream flush is requested in this session.
- FFCR.EnTI is set.
- FFCR.TrigOnTrigEvt is set.

This is a rare scenario where the trigger ID on the ATB slave interface is received before an upstream flush (afready\_s = 1) and trigin is received after the upstream flush.

##### Implications

All flush and trigger requests are ignored. The TMC cannot come out of RUNNING/DISABLING state.

##### Workaround

Generate trigger requests only from one source. Use either upstream trigger requests through the ATB slave interface (ATID=7D) or trigin pulses on the event interface but not both.

If this is not possible then set FFCR.TrigOnTrigEvt = 0 or disable trigger insertion FFCR.EnTI = 0.

## Category C

---

### 1456982

#### TPIU stops accepting trace when FFCR is written

##### Status

Affects: CoreSight SoC-600

Fault Type: Programmer Category C

Fault Status: Present in: r3p0, r3p1 Fixed in: Open

**Description**

This erratum affects the following components:

- Trace Port Interface Unit.
  - css600\_tpiu
  - Component Revisions: r0p0

With this erratum the TPIU stalls the ATB slave interface indefinitely.

**Conditions**

The following conditions must exist:

- TPIU register FFCR.EnFCont is 1, which enables Continuous Formatting Mode.
- At least one of the TPIU register bits FFCR.StopTrig or FFCR.StopFI is 1.
- A condition occurs which causes the TPIU to stop capture.
- Before the TPIU stops capture, the debugger clears TPIU register bits FFCR.StopTrig and FFCR.StopFI to 0.

**Implications**

The following implications persist until the TPIU is reset:

- The ATB slave interface is stalled, accepting no more trace.
- The TPIU does not stop, and FFSR.FtStopped is never set to 1.
- FFSR.FillnProg remains at the value 1.

**Workaround**

Debug tools must avoid clearing FFCR.StopTrig and FFCR.StopFI while the TPIU is not stopped, and must only change these fields when FFSR.FtStopped==1.

Alternatively, disable the trigger indication on the trace port by clearing FFCR.TrigEvt and FFCR.TrigIn before clearing FFCR.StopTrig and FFCR.StopFI.

## 1456966

### TPIU stops accepting trace when the trigger counter is written

#### Status

Affects: CoreSight SoC-600

Fault Type: Programmer Category C

Fault Status: Present in: r3p0, r3p1 Fixed in: Open

#### Description

This erratum affects the following components:

- Trace Port Interface Unit.
  - css600\_tpiu
  - Component Revisions: r0p0

With this erratum the TPIU stalls the ATB slave interface indefinitely.

#### Conditions

The following conditions must exist:

- The TPIU register FFCR.EnFCont is 1, which enables Continuous Formatting Mode.
- The TPIU register TCVR was written with the value 1 or higher, which enabled trigger byte counting.
- The TPIU register bit STMR.TrgRun is 1. The trigger byte counter was loaded with the TCVR value and the trigger byte counter decrements depending on the trace traffic on the ATB slave interface.
- The TPIU register bit FFCR.StopTrig is 0.
- The TPIU register bit FFCR.StopFI is 1 or 0.
- The generation of upstream flush requests is enabled by having set at least one of the TPIU register bits FFCR.FonMan, FFCR.FonTrig or FFCR.FOnFIIn, and an upstream flush is initiated by one of these mechanisms.
- The upstream flush is subsequently acknowledged by the system and the TPIU proceeds to drain internal buffers.
- The TPIU register TCVR is written with the value zero before the TPIU has drained its internal buffers.

#### Implications

The following implications persist until the TPIU is reset:

- The ATB slave interface is stalled, accepting no more trace.
- The TPIU does not stop, and FFSR.FtStopped is never set to 1.
- FFSR.FIInProg remains at the value 1.

#### Workaround

Debug tools must not write to TCVR while the TPIU is not stopped, and must only change TCVR when FFSR.FtStopped==1.

## 1434359

### ATB Replicator DEVID register returns irrelevant field value

#### Status

Affects: CoreSight SoC-600

Fault Type: Programmer Category C

Fault Status: Present in: r2p0, r3p0, Fixed in: r3p1

#### Description

This erratum affects the following components:

- ATB Replicator in programmable configuration only
  - css600\_atbreplicator\_prog
  - Component Revisions: r0p0

The ATB Replicator is used to connect a single ATB data source to two ATB data sinks. The programmable replicator has an APB interface and includes the standard CoreSight management registers.

With this erratum the DEVID register has a SCHEME field, bits [7:4], that returns the value 0x3. The TRM states that this value means "Priority scheme implemented.", however, there is no priority scheme implemented in the replicator so this field is redundant and software should ignore it.

#### Conditions

The following conditions must exist:

- The css600\_atbreplicator\_prog is used.
- Software reads the DEVID register and relies upon the value of bits [7:4], documented as DEVID.SCHEME.

#### Implications

No significant implications are expected. Software may break if it relies on a specific value in DEVID bits [7:4].

#### Workaround

Software must ignore DEVID bits [7:4].



## 1416485

### Flushes might not complete

#### Status

Affects: CoreSight SoC-600

Fault Type: Programmer Category C

Fault Status: Present in: r2p0, r3p0, Fixed in: r3p1

#### Description

This erratum affects the following components:

- Trace Memory Controller in ETS configuration.
  - css600\_tmc\_ets
  - Component Revisions: r0p2, r0p3

After a flush completes on ATB slave interface the data is flushed out to the AXI stream master interface. With this erratum, the flush appears to be still in progress after the data is flushed out on the AXI stream interface.

#### Conditions

The following sequence must exist:

1. Register bit FFCR.StopFI is 0.
2. Flush completes on ATB slave interface.
3. Data offered to AXI stream master interface is collected slower than further data is offered from ATB slave interface.

#### Implications

The ETS Flush In Progress bit FFSR.FInProg is set during a flush and only clears when the data throughput on the ATB slave interface drops below the data throughput on the AXI stream master interface. If this condition does not occur, register bit FFSR.FInProg stays set indefinitely.

Despite completing the flush on the ATB slave interface:

- The register bit FFSR.FInProg indicates that the flush is still in progress until the throughput condition is met.
- The **flushcomp** output is not asserted until the throughput condition is met.
- Subsequent requests to flush (e.g. via input **flushin** or by setting register bit FFCR.FlushMan or by observing a trigger event whilst register bit FFCR.FOnTrigEvt is 1) are pended until the throughput condition is met, and are not requested on the ATB slave interface.
  - Setting FFCR.StopOnFI to 1 after the flush completed on ATB slave interface and then attempting to issue another flush request will not cause the formatter to stop.
    - Without the formatter stopping the register bit STS.TMCReady will not get set and ETS stays in the RUNNING state.

#### Workaround

If the ETS fails to enter the DISABLED state, clear the register bit CTL.TraceCaptEn to 0 to force exiting the RUNNING state via the "emergency stop" state (DISABLING).