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

Confidentiality Status

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 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

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

04-Dec-2019: Changes in document version 9.0				
ID	Status	Area	Cat	Summary of erratum
1639613	New	Programmer	CatC	Debug Port has poor data throughput using JTAG protocol
1552163	Updated	Programmer	CatC	Debug Port has poor data throughput using Serial Wire protocol

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
1552163	New	System	CatC	Debug Port has poor data throughput using Serial Wire protocol

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
1639613	CatC	Debug Port has poor data throughput using JTAG protocol	r2p0, r3p0, r3p1	r3p2
1565895	CatB	TPIU generates spurious data when stopped	r3p0, r3p1	
1552163	CatC	Debug Port has poor data throughput using Serial Wire protocol	r2p0, r3p0, r3p1	r3p2
1456982	CatC	TPIU stops accepting trace when FFCR is written	r3p0, r3p1	
1456966	CatC	TPIU stops accepting trace when the trigger counter is written	r3p0, r3p1	
1434359	CatC	ATB Replicator DEVID register returns irrelevant field value	r2p0, r3p0	r3p1
1416485	CatC	Flushes might not complete	r2p0, r3p0	r3p1
1301623	CatB	CATU performs unnecessary scatter table reads when memory buffer wraps around after the last valid scatter list entry.	r2p0, r3p0	r3p1
1232238	CatB	Reading ETB+ETF memory via RRD is limited by CBUFLEVEL in Disabled state	r2p0, r3p0	r3p1
1116272	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

1639613

Debug Port has poor data throughput using JTAG protocol

Status
Affects: CoreSight SoC-600 - Perpetual
Fault Type: Programmer Cat C
Fault Status: Present in: r2p0, r3p0, r3p1 Fixed in: r3p2

Description

This erratum affects the following components:

- Debug Port
    - css600_dp (css600_dpslv, css600_dpmstr)
    - Component Revisions: r0p2, r0p3

The Debug Port, operating in JTAG protocol, exhibits poor data throughput for APACC accesses to the APB master interface. The Debug Port can issue a WAIT response to the external debugger when the DP is not ready to accept an APACC transaction.

As a result of this erratum:

- APACC Writes will always see one additional WAIT response
- APACC Reads will always see one additional WAIT response.

Conditions

1. The DP is operating in JTAG protocol mode.

Implications

Sequences of back-to-back Reads or Writes will result in one or more WAIT responses from the DP, with a potential performance implication on accesses between the DP and the Access Ports beneath it.

The overall system performance impact is system dependent, and will be influenced by the relative clock ratios of the DP APB interface, the AP master interface clock ratios and the access latency to the components or memory being accessed.

Workaround

None

1552163

Debug Port has poor data throughput using Serial Wire protocol

Status
Affects: CoreSight SoC-600 - Perpetual
Fault Type: System Programmer C
Fault Status: Present in: r2p0, r3p0, r3p1 Fixed in: r3p2

Description
This erratum affects the following components:

- Debug Port
    - css600_dp (css600_dpslv, css600_dpmstr)
    - Component Revisions: r0p2, r0p3

The Debug Port, operating in Serial Wire protocol, exhibits poor data throughput for APACC accesses to the APB master interface. The Debug Port can issue a WAIT response to the external debugger when the DP is not ready to accept an APACC transaction.

As a result of this erratum:

- APACC Writes will always see at least one WAIT response
- APACC Reads will see at least one WAIT response when SWCLKTCK:CLK is greater than 1:5.

Conditions

1. The DP is operating in Serial Wire protocol mode.

Implications
Sequences of back-to-back Reads or Writes will result in one or more WAIT responses from the DP, with a potential performance implication on accesses between the DP and the Access Ports beneath it.
The overall system performance impact is system dependent, and will be influenced by the relative clock ratios of the DP APB interface, the AP master interface clock ratios and the access latency to the components or memory being accessed.

Workaround
None

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.StopFl 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.StopFl 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.FlInProg remains at the value 1.

Workaround
Debug tools must avoid clearing FFCR.StopTrig and FFCR.StopFl 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.StopFl.

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.StopFl 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.FOnFlIn, 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.FlInProg 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.StopFl 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.FlInProg 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.FlInProg stays set indefinitely.

Despite completing the flush on the ATB slave interface:

- The register bit FFSR.FlInProg 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.StopOnFl 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).