AEROSPACE DATA EXCHANGE PROGRAM TRANSMITTAL

PROBLEM ADVISORY

CAES

1. TITLE UT699 LEON3FT, Missing Nullify of Bus Access for Instruction Following Restarted Single-Cycle LOAD Instruction	**2. DOCUMENT NUMBER** SPO-2013-PA-0003 **3. DATE** ***(Year, Month, Date)*** 2013, Sept, 23
4. MANUFACTURER NAME AND ADDRESS CAES 4350 CENTENNIAL BOULEVARD COLORADO SPRINGS, COLORADO 80907-3486	**5. MANUFACTURER POINT OF CONTACT NAME** Gwen Butler **6. MANUFACTURER POINT OF CONTACT TELEPHONE** (719) 594-8466 **7. MANUFACTURER POINT OF CONTACT EMAIL** Gwen.Butler@cobhamaes.com

8. CAGE CODE 65342	**9. LDC START** All	**10. LDC END** All	**11. PRODUCT IDENTIFICATION CODE** WG07A	**12. BASE PART** UT699
13. BLANK			**14. SMD NUMBER** 5962-08228	**15. DEVICE TYPE DESIGNATOR** 01 & 02
			16. RHA LEVELS R & F	**17. QML LEVEL** Q & V
			18. NON QML LEVEL Proto & HiRel	**19. BLANK**

20. PROBLEM DESCRIPTION / DISCUSSION / EFFECT

The GRLIP IP (rev. 1.0.22-b4080 and previous) contained within the affected devices (See SHEET 2), which are the subject of this advisory, causes an instruction restart on a detected parity error when the integer unit is implemented with register file protection.

The error occurs when a single cycle LOAD instruction encounters a data tag parity error and is followed by a LOAD or a STORE instruction. Under this circumstance the data access from the subsequent instruction, either LOAD or STORE, may appear on the Advanced High-performance Bus (AHB) twice. The additional access occurs because the LEON processor detects the data tag parity error after the access reaches the AHB. This can cause data loss or other undesired side-effects if the access is destructive, such as to/from a register that has been mapped in a cacheable memory area, or a store access to the same memory location as the source address for the restarted load instruction.

See Continuation on SHEET 2

Refer to the attached Errata for a complete discussion of the problem.

21. ACTION TAKEN / PLANNED

1. Create an errata to describe the workaround and mitigation methods to handle the error.
   (Complete – LEON3FT Data Cache Nullify Errata – appended to this Product Advisory)

2. Add a compiler switch that inserts a NOP command after single cycle load instructions.
   (Complete – Reference Errata Work-arounds in section 2.0 of the appended errata)

3. The errata will be corrected in CAES' next LEON3FT (Vendor Generic P.N. UT699E).
   (Prototypes available NOW / QML target availability 2QCY14) The revised UT699E is only offered in a 484 Ceramic Land Grid Array, Ceramic Ball Grid Array, and Cercamic Column Grid Array.

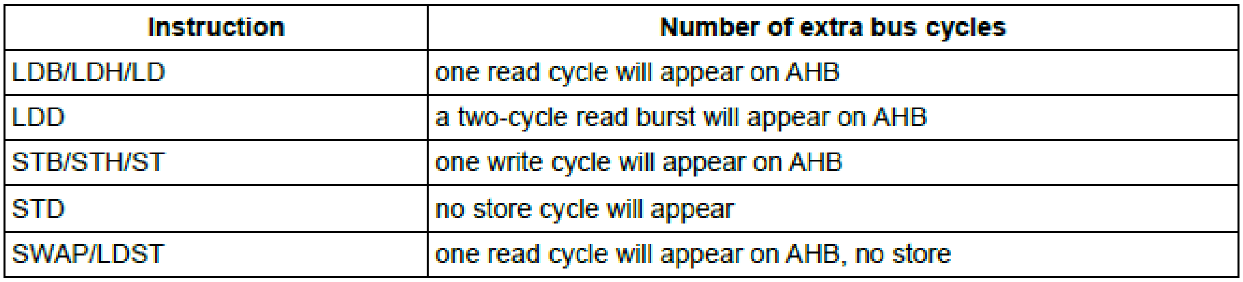
22. DISPOSITIONARY RECOMMENDATION:	**CHECK & USE AS IS** ☐	**CONTACT MANUFACTURER** ☐	**REMOVE & REPLACE** ☐	**CORRECT & USE AS SPECIFIED** ☒

23. ADEPT REPRESENTATIVE **Timothy L. Meade**	**24. SIGNATURE** Timothy L Meade	**25. DATE** **23 September 2013**

Continuation SHEET 1

19. PROBLEM DESCRIPTION / DISCUSSION / EFFECT

Table 1, below, describes the instructions that will exhibit the error when they execute immediately following a single-cycle load instruction with a data tag parity error.



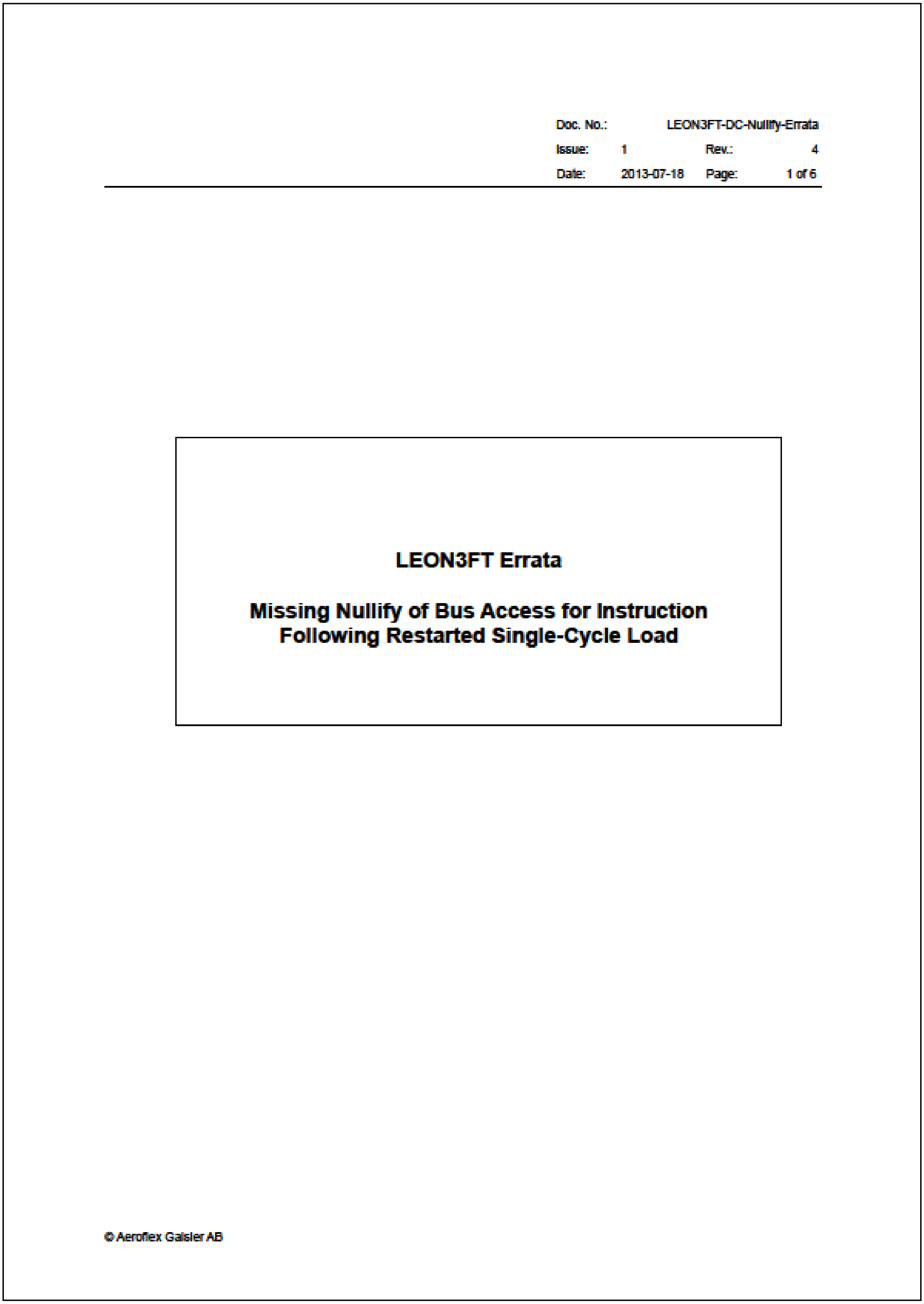
Instruction	Number of extra bus cycles
LDB/LDH/LD	one read cycle will appear on AHB
LDD	a two-cycle read burst will appear on AHB
STB/STH/ST	one write cycle will appear on AHB
STD	no store cycle will appear
SWAP/LDST	one read cycle will appear on AHB, no store

Table 1: Extra bus cycles for instruction following LD that encounters parity error

The SWAP/LDST instructions can also encounter the data tag parity error, which causes the write operation of the instruction to be performed before instruction restart, corrupting the memory location and read result.

Affected Parts

UT699-ZPC	5962F0822801QXC
UT699-SPA	5962R0822801QXC
UT699-CPA	5962F0822802QXC
UT699-XPC	5962R0822802QXC
UT699-XEC	5962F0822801VXC
UT699-ZEC	5962R0822801VXC
UT699-SEA	5962F0822801QYC
UT699-CEA	5962R0822801QYC
	5962F0822802QYC
	5962R0822802QYC
	5962F0822801VYC
	5962R0822801VYC
	5962F0822801QZA
	5962R0822801QZA
	5962F0822802QZA
	5962R0822802QZA
	5962F0822801VZA
	5962R0822801VZA



LEON3FT Errata

Missing Nullify of Bus Access for Instruction Following Restarted Single-Cycle Load

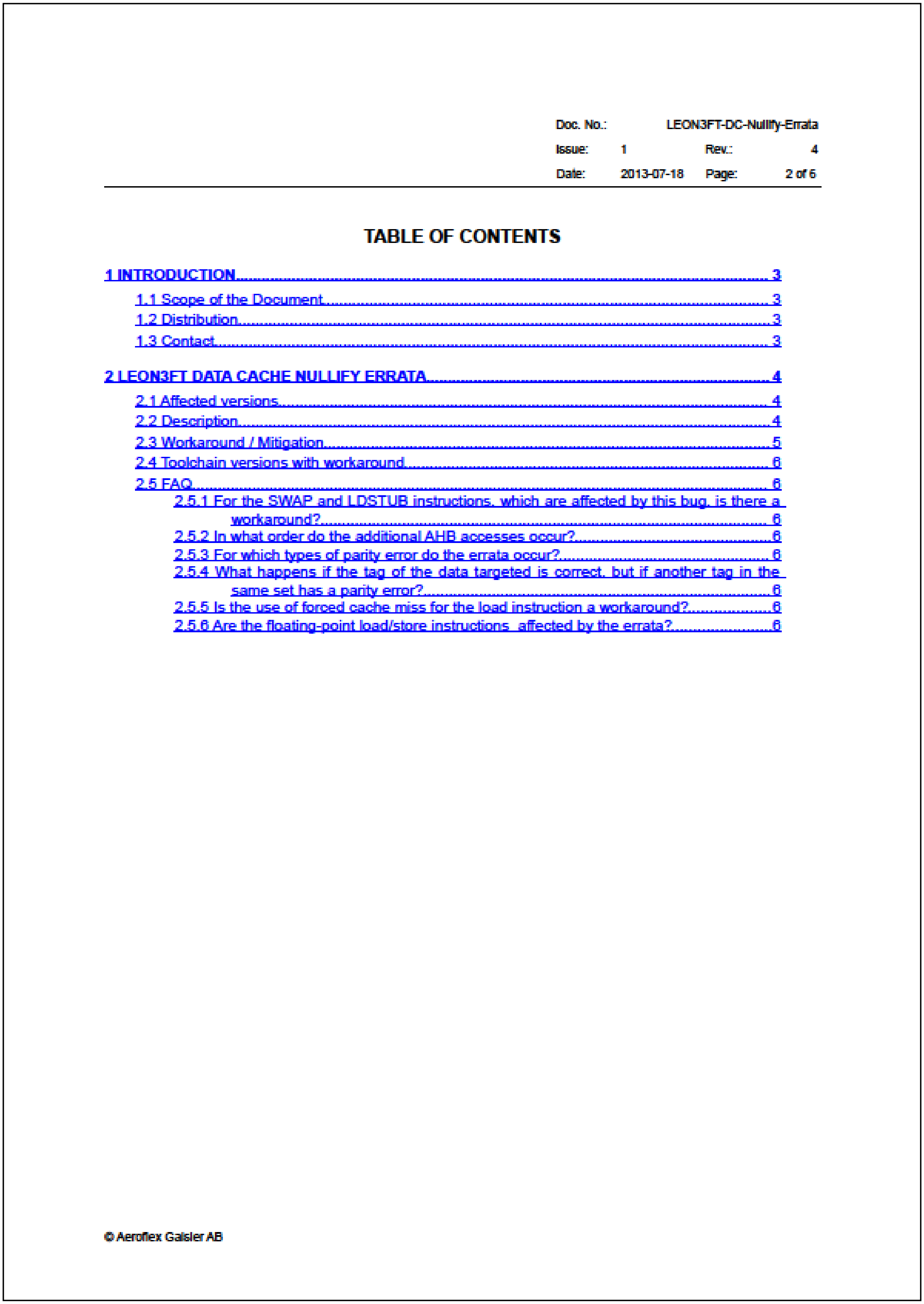
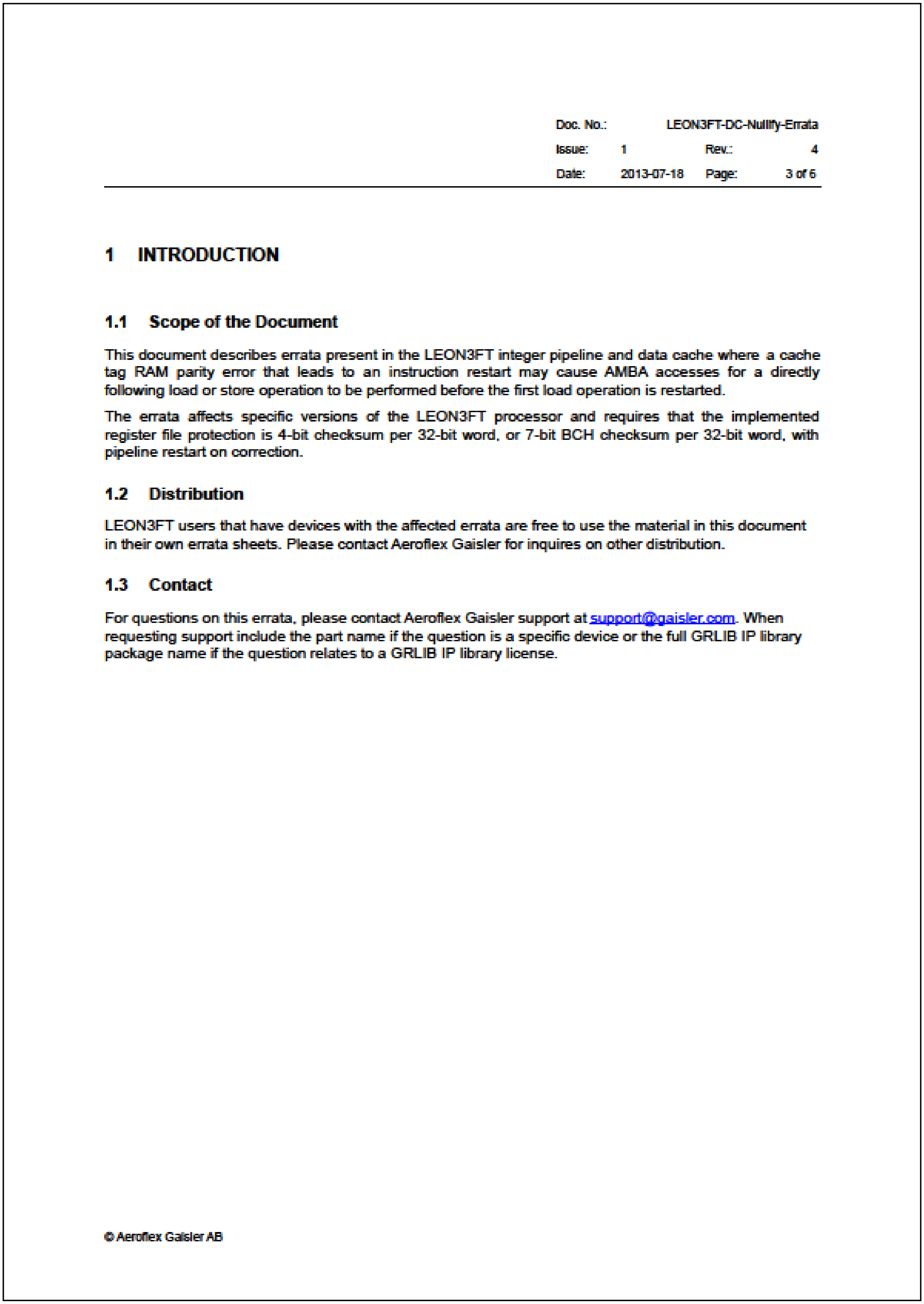


TABLE OF CONTENTS



1 INTRODUCTION

1.1 Scope of the Document

This document describes errata present in the LEON3FT integer pipeline and data cache where a cache tag RAM parity error that leads to an instruction restart may cause AMBA accesses for a directly following load or store operation to be performed before the first load operation is restarted.

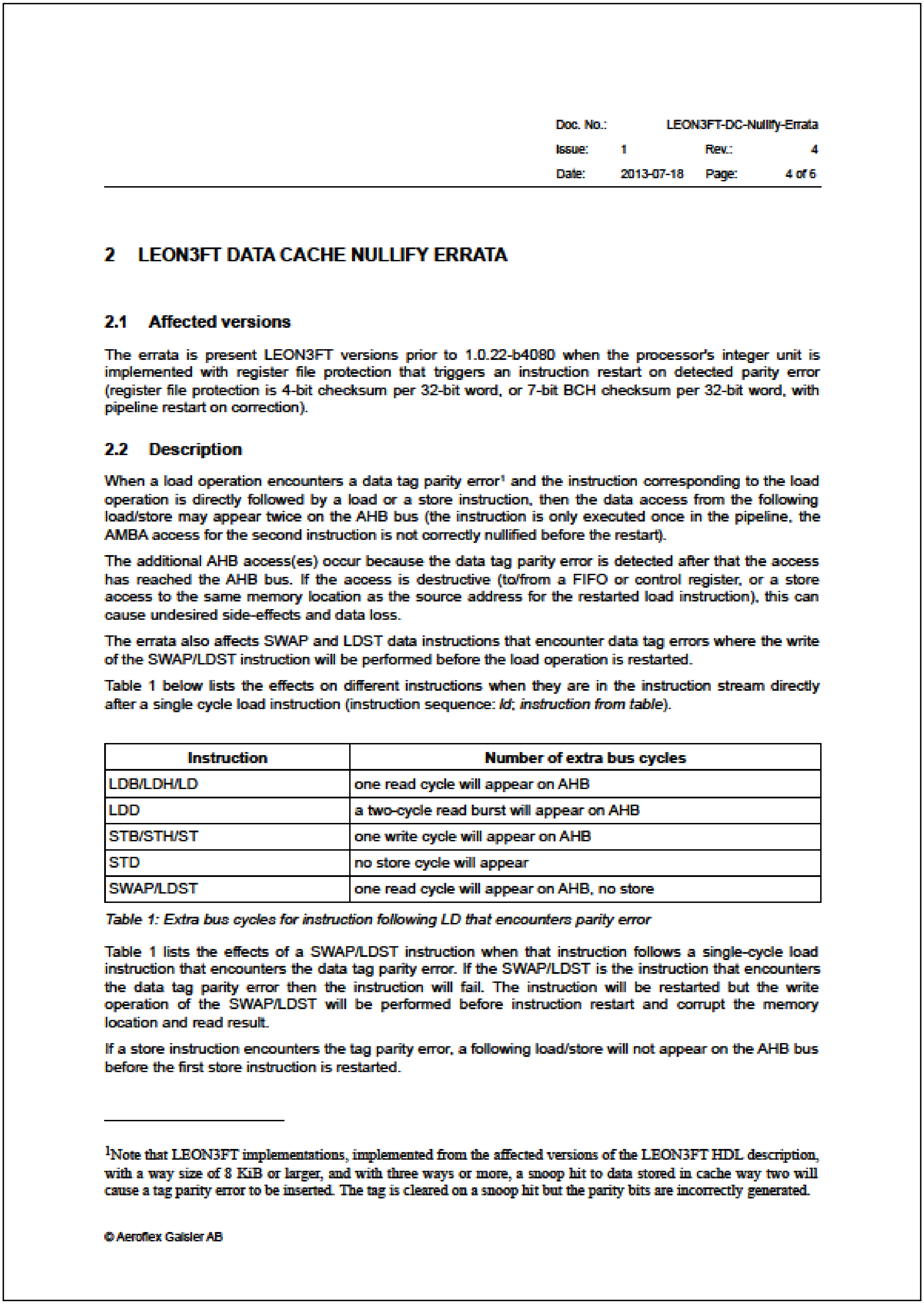
The errata affects specific versions of the LEON3FT processor and requires that the implemented register file protection is 4-bit checksum per 32-bit word, or 7-bit BCH checksum per 32-bit word, with pipeline restart on correction.

1.2 Distribution

LEON3FT users that have devices with the affected errata are free to use the material in this document in their own errata sheets. Please contact Aeroflex Gaisler for inquires on other distribution.

1.3 Contact

For questions on this errata, please contact Aeroflex Gaisler support at support@gaisler.com. When requesting support include the part name if the question is a specific device or the full GRLIB IP library package name if the question relates to a GRLIB IP library license.



2 LEON3FT DATA CACHE NULLIFY ERRATA

2.1 Affected versions

The errata is present LEON3FT versions prior to 1.0.22-b4080 when the processor's integer unit is implemented with register file protection that triggers an instruction restart on detected parity error (register file protection is 4-bit checksum per 32-bit word, or 7-bit BCH checksum per 32-bit word, with pipeline restart on correction).

2.2 Description

When a load operation encounters a data tag parity error[1] and the instruction corresponding to the load operation is directly followed by a load or a store instruction, then the data access from the following load/store may appear twice on the AHB bus (the instruction is only executed once in the pipeline, the AMBA access for the second instruction is not correctly nullified before the restart).

The additional AHB access(es) occur because the data tag parity error is detected after that the access has reached the AHB bus. If the access is destructive (to/from a FIFO or control register, or a store access to the same memory location as the source address for the restarted load instruction), this can cause undesired side-effects and data loss.

The errata also affects SWAP and LDST data instructions that encounter data tag errors where the write of the SWAP/LDST instruction will be performed before the load operation is restarted.

Table 1 below lists the effects on different instructions when they are in the instruction stream directly after a single cycle load instruction (instruction sequence: *ld*; *instruction from table*).

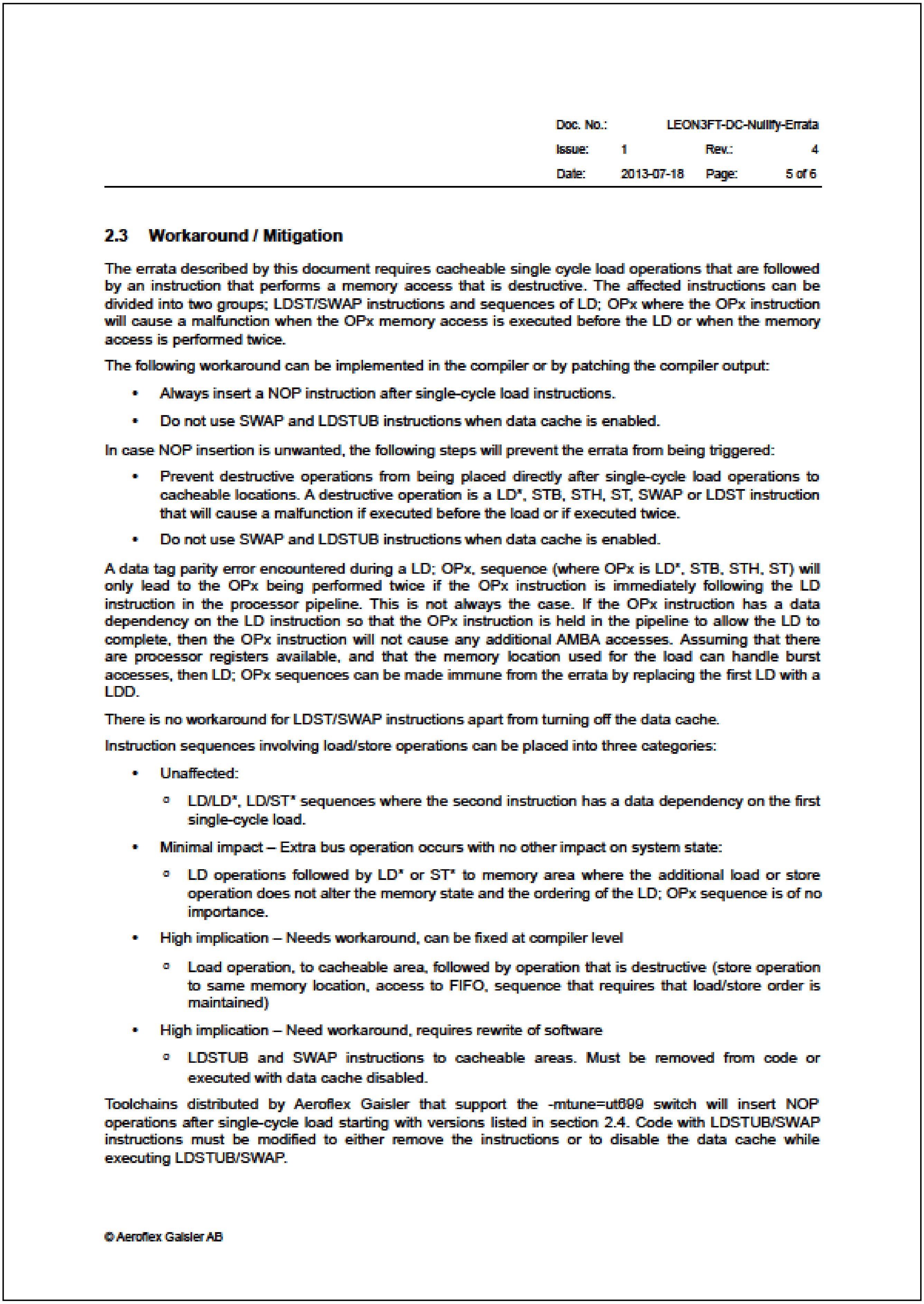
Instruction	Number of extra bus cycles
LDB/LDH/LD	one read cycle will appear on AHB
LDD	a two-cycle read burst will appear on AHB
STB/STH/ST	one write cycle will appear on AHB
STD	no store cycle will appear
SWAP/LDST	one read cycle will appear on AHB, no store

Table 1: Extra bus cycles for instruction following LD that encounters parity error

Table 1 lists the effects of a SWAP/LDST instruction when that instruction follows a single-cycle load instruction that encounters the data tag parity error. If the SWAP/LDST is the instruction that encounters the data tag parity error then the instruction will fail. The instruction will be restarted but the write operation of the SWAP/LDST will be performed before instruction restart and corrupt the memory location and read result.

If a store instruction encounters the tag parity error, a following load/store will not appear on the AHB bus before the first store instruction is restarted.

[1] Note that LEON3FT implementations, implemented from the affected versions of the LEON3FT HDL description, with a way size of 8 KiB or larger, and with three ways or more, a snoop hit to data stored in cache way two will cause a tag parity error to be inserted. The tag is cleared on a snoop hit but the parity bits are incorrectly generated.



2.3 Workaround / Mitigation

The errata described by this document requires cacheable single cycle load operations that are followed by an instruction that performs a memory access that is destructive. The affected instructions can be divided into two groups; LDST/SWAP instructions and sequences of LD; OPx where the OPx instruction will cause a malfunction when the OPx memory access is executed before the LD or when the memory access is performed twice.

The following workaround can be implemented in the compiler or by patching the compiler output:

- Always insert a NOP instruction after single-cycle load instructions.
- Do not use SWAP and LDSTUB instructions when data cache is enabled.

In case NOP insertion is unwanted, the following steps will prevent the errata from being triggered:

- Prevent destructive operations from being placed directly after single-cycle load operations to cacheable locations. A destructive operation is a LD*, STB, STH, ST, SWAP or LDST instruction that will cause a malfunction if executed before the load or if executed twice.
- Do not use SWAP and LDSTUB instructions when data cache is enabled.

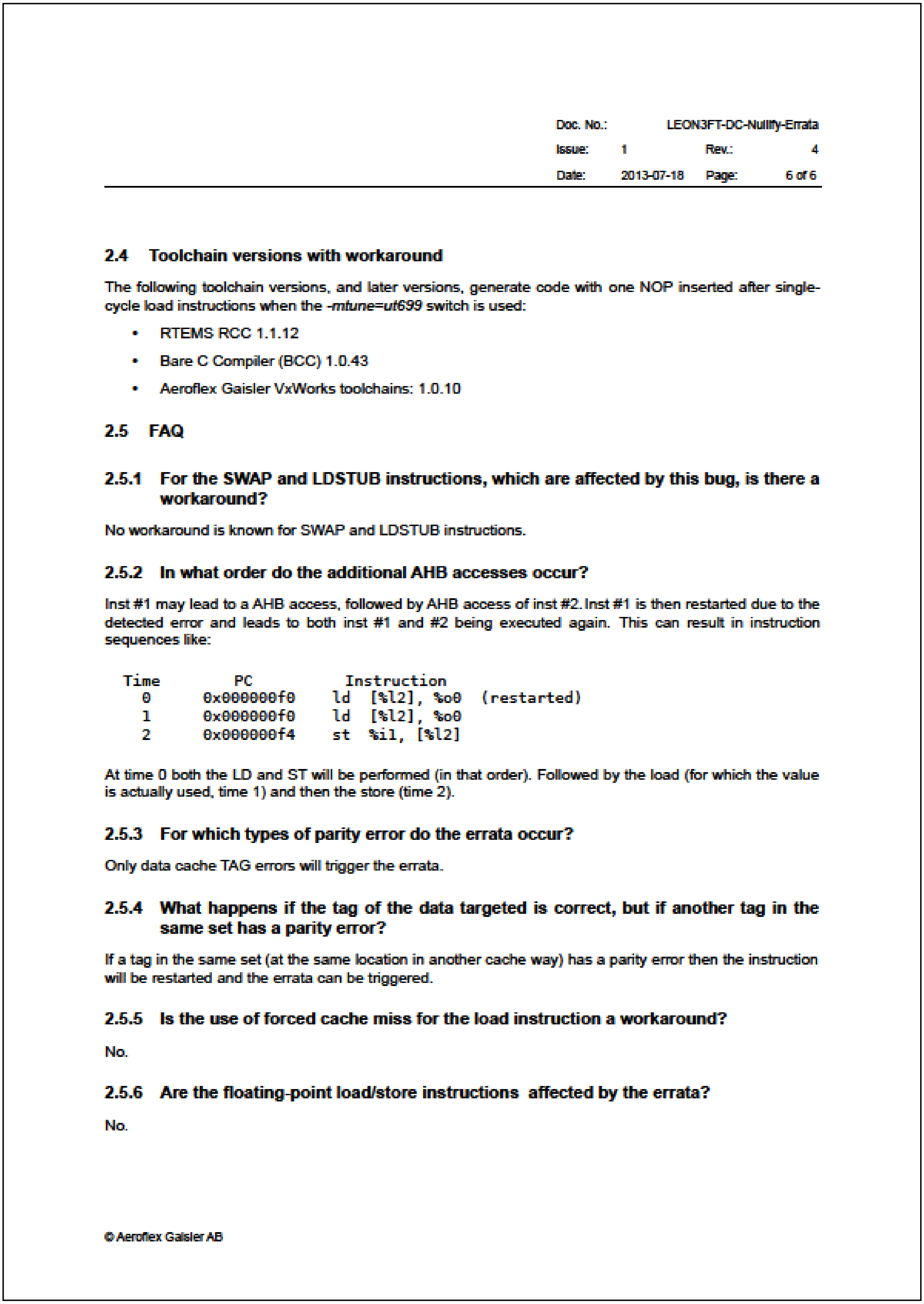
A data tag parity error encountered during a LD; OPx, sequence (where OPx is LD*, STB, STH, ST) will only lead to the OPx being performed twice if the OPx instruction is immediately following the LD instruction in the processor pipeline. This is not always the case. If the OPx instruction has a data dependency on the LD instruction so that the OPx instruction is held in the pipeline to allow the LD to complete, then the OPx instruction will not cause any additional AMBA accesses. Assuming that there are processor registers available, and that the memory location used for the load can handle burst accesses, then LD; OPx sequences can be made immune from the errata by replacing the first LD with a LDD.

There is no workaround for LDST/SWAP instructions apart from turning off the data cache.

Instruction sequences involving load/store operations can be placed into three categories:

- Unaffected:
  - LD/LD*, LD/ST* sequences where the second instruction has a data dependency on the first single-cycle load.
- Minimal impact – Extra bus operation occurs with no other impact on system state:
  - LD operations followed by LD* or ST* to memory area where the additional load or store operation does not alter the memory state and the ordering of the LD; OPx sequence is of no importance.
- High implication – Needs workaround, can be fixed at compiler level
  - Load operation, to cacheable area, followed by operation that is destructive (store operation to same memory location, access to FIFO, sequence that requires that load/store order is maintained)
- High implication – Need workaround, requires rewrite of software
  - LDSTUB and SWAP instructions to cacheable areas. Must be removed from code or executed with data cache disabled.

Toolchains distributed by Aeroflex Gaisler that support the -mtune=ut699 switch will insert NOP operations after single-cycle load starting with versions listed in section 2.4. Code with LDSTUB/SWAP instructions must be modified to either remove the instructions or to disable the data cache while executing LDSTUB/SWAP.



2.4 Toolchain versions with workaround

The following toolchain versions, and later versions, generate code with one NOP inserted after single-cycle load instructions when the *-mtune=ut699* switch is used:

- RTEMS RCC 1.1.12
- Bare C Compiler (BCC) 1.0.43
- Aeroflex Gaisler VxWorks toolchains: 1.0.10

2.5 FAQ

2.5.1 For the SWAP and LDSTUB instructions, which are affected by this bug, is there a workaround?

No workaround is known for SWAP and LDSTUB instructions.

2.5.2 In what order do the additional AHB accesses occur?

Inst #1 may lead to a AHB access, followed by AHB access of inst #2. Inst #1 is then restarted due to the detected error and leads to both inst #1 and #2 being executed again. This can result in instruction sequences like:

```
  Time         PC           Instruction
    0      0x000000f0    ld  [%l2], %o0  (restarted)
    1      0x000000f0    ld  [%l2], %o0
    2      0x000000f4    st  %i1, [%l2]
```

At time 0 both the LD and ST will be performed (in that order). Followed by the load (for which the value is actually used, time 1) and then the store (time 2).

2.5.3 For which types of parity error do the errata occur?

Only data cache TAG errors will trigger the errata.

2.5.4 What happens if the tag of the data targeted is correct, but if another tag in the same set has a parity error?

If a tag in the same set (at the same location in another cache way) has a parity error then the instruction will be restarted and the errata can be triggered.

2.5.5 Is the use of forced cache miss for the load instruction a workaround?

No.

2.5.6 Are the floating-point load/store instructions affected by the errata?

No.