LEON3/FT AHB Lock Release during Atomic Operation

Technical note 2018-04-12

Doc. No GRLIB-TN-0011

Issue 1.5

CHANGE RECORD

Issue	Date	Section / Page	Description
1.2	2017-11-20		First public release
1.3	2017-11-20	1.1	Remove redundant text
1.4	2017-12-21	Section 3.2	Overview of compiler workaround and list of toolchains with the workaround available.
1.5	2018-04-12	Section 2	Conditions that can affect the execution flow

TABLE OF CONTENTS

1 OVERVIEW

1.1 Scope of the Document

This document describes a corner case that affects specific versions of the LEON3/FT core that can cause the LEON3/FT core to lose the bus lock before making the store operation of an atomic instruction. The corner case can only be triggered when several additional conditions, including the use of the processor's memory management unit, are satisfied during execution.

1.2 Affected versions

The issue is present in all LEON3/FT cores implemented with memory management unit (MMU) up to GRLIB build b4204.

1.3 Affected components

Cobham and Aeroflex components that are **affected** are:

- GR712RC
- UT699, UT699E, UT700*
- LEON3FT-RTAX*

*The condition described in this document is present in the UT699, UT699E, UT700 and LEON3FT-RTAX components but they are practically unaffected due to including only a single LEON3/FT core, and the issue affects atomic instructions when used for synchronization in a multiprocessor system. However, some applications may have implemented memory scrubbing in software using atomic instructions to be able to read-modify-write external memory to scrub correctable EDAC errors. The issues describe in this document may affect these memory scrubbing routines when scrubbing memory areas that are concurrently written by other peripherals in the system.

Cobham and Aeroflex components that are **unaffected** are:

- GR740 – LEON4/FT is unaffected

1.4 How to check if a LEON3/FT design is affected

The GRLIB build ID is present in the AMBA plug&play area. The build ID is also reported by the GRMON debug monitor when connecting to the device.

If you have licensed GRLIB for use in your own FPGA or ASIC design, the GRLIB version can be seen in the file name of the downloaded release package, in the directory name after unpacking the release, and in the file lib/grlib/stdlib/version.vhd in the release file tree (constant grlib_build).

If the GRLIB build ID is smaller or equal to b4204 then the system is affected.

1.5 Distribution

Users that have affected devices described in this document are free to use the material in this document in their own documentation. Contact Cobham Gaisler for inquires on other distribution.

1.6 Contact

For questions on this document, please contact Cobham 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 TECHNICAL DESCRIPTION

The corner case described in this document can cause a LEON3/FT core to lose the bus lock before completing the store operation required by an atomic instruction (swap, ldstub, casa). The atomic instruction will complete atomically in terms of instruction order but due to losing bus lock before store, the atomicity on the bus level is not guaranteed. As a result, if before being able to make the store required by the atomic instruction, if another master gains access to the bus and makes a store to the same address (used by the atomic instruction) it may cause erroneous behaviour since atomicity is lost.

In order for the corner case described in this document to affect the execution following conditions must be met:

1. The processor's memory management unit (MMU) is enabled.
2. The second instruction after an atomic instruction must cause an instruction translate lookaside buffer (ITLB) miss. It should be noted that this condition can be satisfied with two different instruction flows. Either two consecutive instructions after the atomic instruction can cause an ITLB miss or an atomic instruction is placed in the delay slot of a PC-relative control-transfer instruction and the consecutive instruction of the branch target instruction causes an ITLB miss. It should be noted that, when executing consecutive instructions an ITLB miss can occur even though a page boundary is not crossed due to virtually indexed and virtually tagged LEON3/FT instruction cache.

   Example1:

   ```
   PC            INST
   …
   0x...0AD8     swap [%l1],%l2
   0x...0ADC     sub %l3,%l4,%l3
   0x...0AE0     add %l1,0x4,%l1  (this instruction must cause an ITLB miss)
   …
   ```

   Example 2:

   ```
            PC            INST
            …
   $L0:     0x...0B1C     or %l4,%l5,%l4
            0x...0B20     sub %l3,%l4,%l3 (this instruction must cause an ITLB miss when
                                  branched from 0x...2AD8)
            …
            …
            0x...2AD4     bne $L0
            0x...2AD8     swap [%l1],%l2
   ```

Note:

- A PC-relative control-transfer instruction includes branch on integer condition codes instructions (see SPARC Architecture Manual Version 8, section B.21), Branch on floating-point condition code instructions (see SPARC Architecture Manual Version 8, section B.22) and CALL instruction (see SPARC Architecture Manual Version 8, section B.24).
- The CALL instruction mentioned in the previous point only includes the real CALL instruction (CALL label). The pseudo CALL instruction (call with register relative addressing) do not cause the behaviour to be triggered which is explained in example 2.

If the two previously described conditions are met, the page walk initiated by the ITLB miss will interfere with the atomic instruction operation just before the store operation arrives. When the page walk operation interferes, the store operation will be delayed and the processor's lock on the bus will be released. While performing the page walk operation there will be one or more (depends on the page table) cycles on the bus in which another master can be granted. If during one of those cycles another master, competing for the same address with the atomic instruction, is granted to the bus then the atomicity will be lost. This will cause erroneous behaviour because the store of the atomic instruction of the other master will arrive after the second master stores its data hence it will overwrite it.

3 WORKAROUND / MITIGATION

3.1 Workaround

The erroneous behaviour described in this document can be avoided with the following modifications.

1. Insert “.align 16” directive before atomic instructions.
2. If an atomic instruction is placed at a branch delay slot, move it before the branch, replace the delay slot with a “nop” and put “.align 16” directive before the moved atomic instruction.

It should be noted that this workaround is only needed if the software going to run when MMU is enabled and in a multi-master configuration in which atomic instructions are used to access shared memory areas.

3.2 Toolchain versions with workaround

Errata workarounds are available for BCC, RCC, and VxWorks. The workarounds include compiler fixes as well as patched assembly code for the system libraries. The following release versions and later will include the workarounds in toolchain and OS/libraries:

- BCC 1.0.50

- BCC 2.0.2
- RCC 1.2.22
- RCC 1.3-RC3
- VxWorks 6.7 1.0.21 – Toolchain 1.0.15
- VxWorks 6.9 2.0.8 – Toolchain 1.0.4
- VxWorks 7 – Toolchain 7.2.0.0
- MKPROM 2.0.63

The compiler workaround will insert the ".align 16" directive before any atomic instruction. An atomic instruction will by default never be placed in a delay slot by the compiler, so this case is not an issue. The workaround is thus in accordance with the method described in the previous section. The VxWorks 6.7 and BCC 1 compilers does not generate atomic instructions and does not need the compiler workaround.

Note that the compiler workaround will not make any modifications to inline assembly code in the user application. Any sensitive assembly instruction sequence will have to be modified manually. Note also that BCC and RCC by default does not use the MMU, which means that the errata can not trigger unless the MMU is explicitly enabled.

The compiler workaround is activated by compiling with the "-mfix-gr712rc" flag.

The patches for the compiler workaround are listed below. They are included in the official GCC distribution for version 7.3.0 and later.

[SPARC] Errata workaround for GRLIB-TN-0011
https://gcc.gnu.org/viewcvs/gcc?view=revision&revision=255235

3.3 Scan script

Scan scripts are provided allow scans of disassembled programs, object files and libraries for code that can trigger the issue described in this document. The scripts are available from:

http://www.gaisler.com/notes

Please see the headers in the script files for usage instructions.