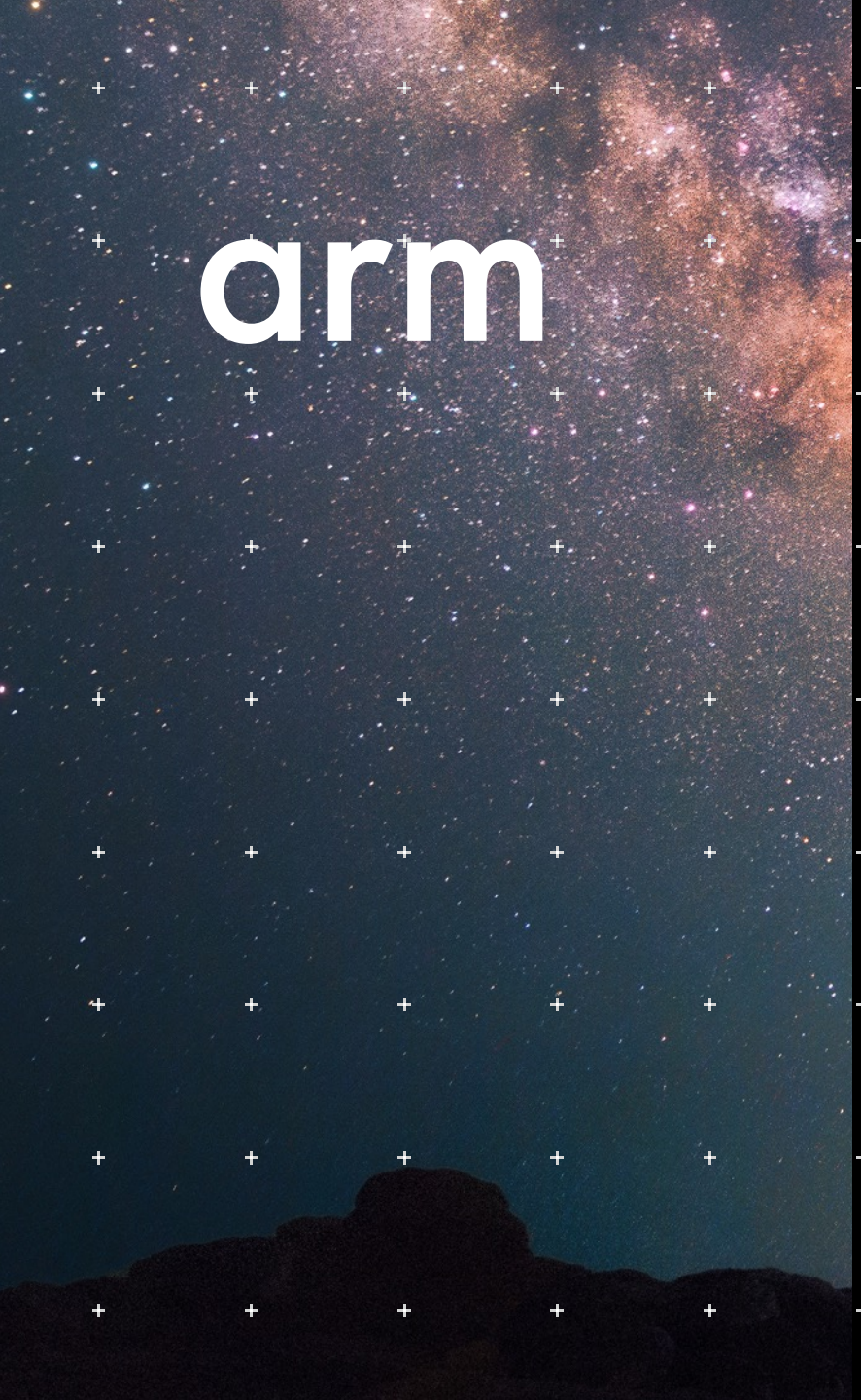
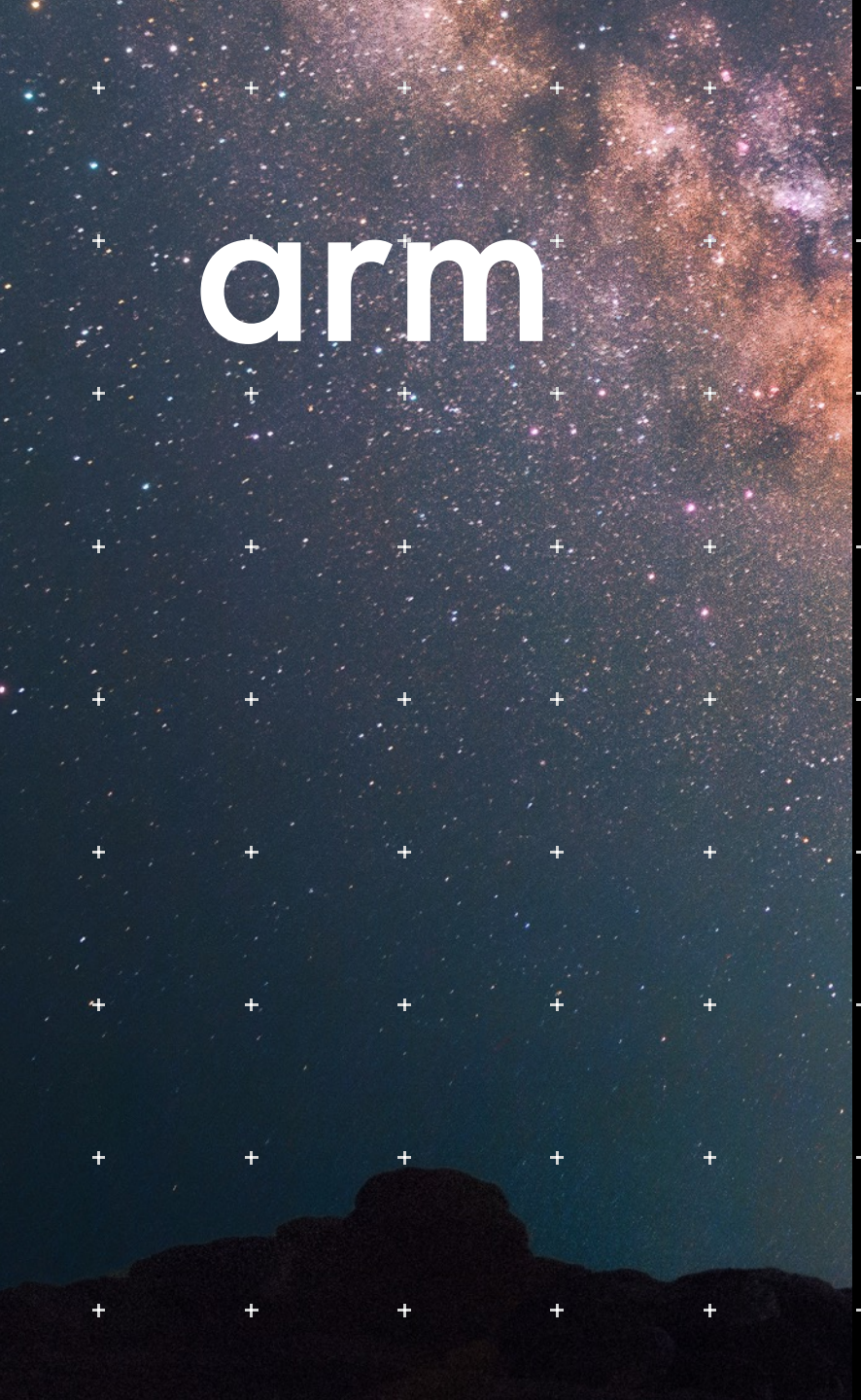


TF-A Tech Forum

August 27, 2020



TF-A Errata Process

August 27, 2020

Agenda

- Bug Review & Categorization of Errata
- Software Developers Errata Notice (SDEN)
- What TF-A Implements
- How TF-A Implements
- Testing

arm

Categorization of Errata

- Bugs
  - raised by partners
  - found by internal tools part of verification
  - critical security findings
- Assigns categories to the Errata and reviews proposed fixes.
- Representative from TF-A present to learn about what is upcoming so we can prepare.

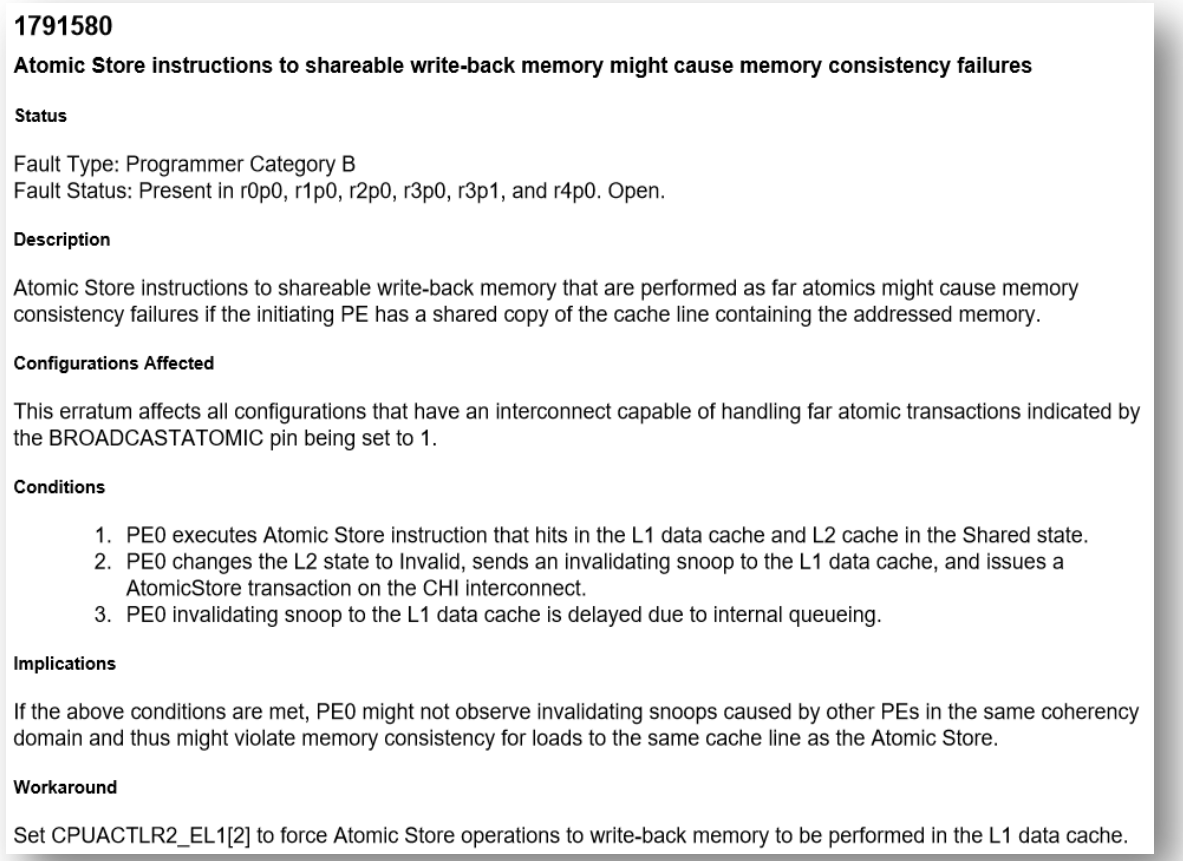
BRC & Categorization of Errata

- Categorization of Errata
  - Errata are split into three levels of severity and further qualified as common or rare:

Category A	Critical, no workaround is available, or workarounds so impactful that we don’t provide workarounds.
Category A (Rare)	Critical, no workaround is available, or workarounds are impactful that we don’t provide workarounds. rare.
Category B	Significant or critical with an acceptable workaround.
Category B (Rare)	Significant or critical with an acceptable workaround, rare.
Category C	Minor.

SDEN

- The categorized Errata with workarounds, if a workaround exists, and what CPU versions they appear in are published in the Software Developer Errata Notice (SDEN)
- Since TF-A is an open source project we only implement workarounds published in the publicly available SDEN. (on developer.arm.com)



1791580

Atomic Store instructions to shareable write-back memory might cause memory consistency failures

Status

Fault Type: Programmer Category B
Fault Status: Present in r0p0, r1p0, r2p0, r3p0, r3p1, and r4p0. Open.

Description

Atomic Store instructions to shareable write-back memory that are performed as far atomics might cause memory consistency failures if the initiating PE has a shared copy of the cache line containing the addressed memory.

Configurations Affected

This erratum affects all configurations that have an interconnect capable of handling far atomic transactions indicated by the BROADCASTATOMIC pin being set to 1.

Conditions

1. PE0 executes Atomic Store instruction that hits in the L1 data cache and L2 cache in the Shared state.
2. PE0 changes the L2 state to Invalid, sends an invalidating snoop to the L1 data cache, and issues a AtomicStore transaction on the CHI interconnect.
3. PE0 invalidating snoop to the L1 data cache is delayed due to internal queueing.

Implications

If the above conditions are met, PE0 might not observe invalidating snoops caused by other PEs in the same coherency domain and thus might violate memory consistency for loads to the same cache line as the Atomic Store.

Workaround

Set CPUACTLR2_EL1[2] to force Atomic Store operations to write-back memory to be performed in the L1 data cache.

What TF-A Implements

Rule of thumb for what TF-A implements

a) The erratum is Cat B. Cat C workarounds can be provided if directly requested by partners
b) The erratum exists on CPU revisions post EAC*. Earlier revisions can be provided if directly requested by partners (e.g. if C is true)
c) The erratum affects CPU revisions that are deployed in the field (i.e. there is a customer for it)

- EAC stands for Early Access Release to partners.

How TF-A Implements

- Disabled by default.
- Added to TF-A based on CPU and version.
- Typically the workaround is an instruction sequence patch or setting a chicken bit during boot.
- More complicated workarounds can involve additions during runtime, boot and runtime together, or TF-A and Kernel together.

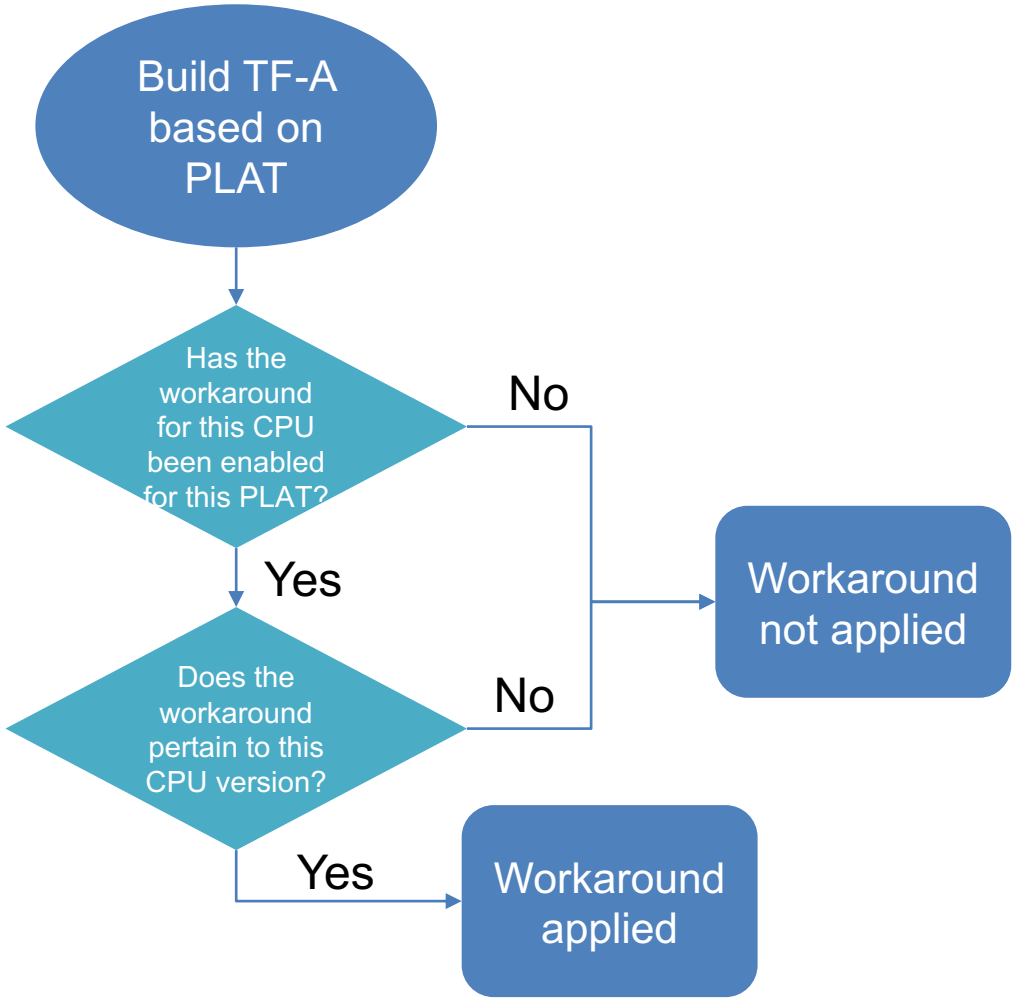
Simple workaround example:

```
/* ---------------------------------------------
 * Errata Workaround for Neoverse N1 Errata #1207823
 * This applies to revision <=r2p0 of Neoverse N1.
 * Inputs:
 * x0: variant[4:7] and revision[0:3] of current cpu.
 * Shall clobber: x0-x17
 * ---------------------------------------------
 */
func errata_n1_1207823_wa
        /* Compare x0 against revision r2p0 */
        mov     x17, x30
        bl      check_errata_1207823
        cbz     x0, 1f
        mrs     x1, NEOVERSE_N1_CPUACTLR2_EL1
        orr     x1, x1, NEOVERSE_N1_CPUACTLR2_EL1_BIT_11
        msr     NEOVERSE_N1_CPUACTLR2_EL1, x1
1:
        ret     x17
endfunc errata_n1_1207823_wa

func check_errata_1207823
        /* Applies to <=r2p0 */
        mov     x1, #0x20
        b       cpu_rev_var_ls
endfunc check_errata_1207823
```


arm

Applying Errata workaround - Flowchart



Testing

- FVP + ArmDS
  - Most workarounds that require an instruction sequence or bit set during boot can be tested using the associated FVP and confirming the registers are properly being set to specification
  - Note : We do not execute the CPU verification suites that find errata, therefore we do not test that the workarounds actually work.

- FPGA
  - More complicated workarounds may require testing on FPGA platform
    - E.g. Errata spread across TF-A & Kernel
  - Unavailability of FVP for any new implementation

arm

Thank You
Danke
Merci
谢谢
ありがとう
Gracias
Kiitos
감사합니다
धन्यवाद
شكرًا
ধন্যবাদ
תודה