CPU Brainfreeze: The Cortex-A53 Last Level of Cache Conundrum

Authors omitted for review.

***Abstract*—**

***Index Terms*—**

I. Introduction

- In modern Heterogenuous Mutli-core Systems, caches are an angular piece of hardware as they efficiently bridge the gap between the speed of the connected execution units and the main memory.
- With the time, shared last-level of caches architecture has evolved to the point they are now capable of managing transactions to the main memory transparently. A technic known as non-blocking.
- While providing top of the line performances, this complex machinery is unpredictable, a source of concern for safety critical hard real-time systems.
- In addition to well known sources of unpredictability such as the inter-core eviction, recent research have highlighted that internal component such as the Miss-Status-Holding-Register (or MSHR) can introduce substantial inter-core interferences in certain circumstances.
- Previous research put in evidence that such situation occur solely under high write loads.
- The present article shows that the under certain circumstances, a single read transaction is also capable of jeoparding the system predictability and even fully block all the masters connected to the last-level of cache.
- We advocate that, in addition to the inter-core eviction and the management of shared LLC sub-units, a third source of unpredictability exists: the memory target response time.

II. Background

- Cache look-up, hit-miss and eviction mechanism
- Miss-Status-Holding-Register meachnism and shared buffer

III. Related work

IV. Overview

V. Evaluation

A. Experimental Setup

- Jailhouse used to partition the cache (via cache coloring), ensuring that the results will not be stained with inter-core evictions. Furthermore, it isolates the Software stack, ensuring that the observed delays do not come from the operating systems safety mechanisms.
- In the following experiments, only two virtual machines are used. One referred to as the *victim* and the other referred to as the *polluter*. The *victim* virtual machine is a full fledge Linux system tasked to run a givan payload. The *victim* VM features three cores and has half of the LLC allocated as private cache. The *polluter* VM is a lightweight baremetal application in charge of emitting sequential read transactions toward the desired target. The latter runs on one core. We enforce that only one transaction at a time is sent to the target by (1) inserting a *Data Synchronisation Barrier* instruction (or `DSB`) after each read and (2) having one cache partition for the code located in main memory and another cache partition for to store the read transactions mapping the target memory.

B. AXI-Resistor experiment

C. On-Chip Memory experiment

VI. Discussion

VII. Conclusion

- Such system, under strict conditions cannot quarantee QoS nor Mixed-criticality levels.
- In contrast to what has been previously reported, read intensive applications can also become a threat for the system predictability.
- Bus slaves must be designed carefully to provide fast answers. More specifically, SoCs featuring a tightly integrated FPGA must ensure that the FPGA can only be reprogrammed by a trusted actor as simply holding a single transaction can indefinitely stall the whole core cluster.