Breaking the Memory Wall with a Flexible Open-Source L1 Data-Cache

Davy Million
Univ. Grenoble Alpes, CEA, List,
F-38000 Grenoble, France
davy.million@cea.fr

Noelia Oliete-Escuín
Barcelona Supercomputing Center,
Universitat Politècnica de Catalunya,
Barcelona, Spain
noelia.oliete@bsc.es

César Fuguet
Univ. Grenoble Alpes, CEA, List,
F-38000 Grenoble, France
cesar.fuguettortolero@cea.fr

***Abstract*—The lack of concurrency and pipelining in the memory sub-system of recent open-source RISC-V processors, such as the CVA6[1], is increasingly becoming the performance bottleneck. Recent updates to the new RISC-V High Performance L1 Data-cache (HPDcache), now fully integrated with the CVA6, bring significant (up to +234%) speedups in key benchmarks with a negligible 5.92% area impact. In this short paper, we detail these improvements, compare performance with existing caches and highlight the benefits of this new, open-source data-cache.**

***Index Terms*—RISC-V, Non-blocking Data-cache, Open-Source**

I. Introduction

Currently, the open-source computer architecture community is focused on designing CPUs and accelerators, but there is less activity directed at the cache hierarchy which is the main bottleneck in computing systems.

For example, although the CVA6 core can be configured to issue up to 8 outstanding instructions, when there is a single cache read miss, the processor is stalled due to the blocking nature [1] of the existing L1 data-cache. The problem arises if another load miss happens since the core must wait for the previous response before it issues the second load to the cache (miss under miss). This has a significant negative impact on memory-intensive programs, especially those using data-structures with indirection such as linked-lists, sparse matrices, and graphs. Conversely, non-blocking data-caches, such as the High Performance Data-cache (HPDcache) [2], can pipeline multiple miss requests without stalling, considerably improving their average throughput [3]. When the HPDcache was initially released, the performance was not quantified. In this paper, we demonstrate its benefits through application benchmarks.

To improve the performance of the RISC-V processors in the open hardware community, the first version of a new non-blocking data-cache was released on GitHub in 2023 (HPDcache[2]) and recent updates, outlined in this paper, have significantly improved the performance of this new cache.

II. Background

A. HPDcache Core Characteristics

The HPDcache has (1) a three stage pipeline, (2) supports wide memory interfaces (up to 64 bytes, which improves both bandwidth and latency), (3) has a configurable Miss Status Holding Register (MSHR) to support multiple outstanding read miss requests to the memory, and (4) is able to re-order requests from the processor to serve the ones whose data is available first, thus minimizing processor stalls. The cache is write-through (WT), meaning that store requests are forwarded to the next level of the memory hierarchy, simplifying its integration in a multi-core environment [3]. To reduce this write traffic, the HPDcache implements a coalescing write-buffer to merge adjacent store requests into a single transaction.

The HPDcache is flexible as it has over 25 significant SystemVerilog parameters that the user can fine-tune at compilation/synthesis time. For example, the number of requesters is configurable to support high-performance processors with multiple load-store units and tightly-coupled accelerators. Nonetheless, it can also be used in a lean configuration for small, embedded processors.

Since its initial open-source release [2], there have been multiple improvements to the HPDcache, including:

- Ability to index data using virtual addresses (virtually-indexed physically-tagged), which shortens the request latency by 1 cycle ;
- Improvements in the selection policy of the re-ordering mechanism, to prioritize dependent on-hold requests ;
- Improvements in the coalescing policy of the write buffer to virtually increase its size.

These improvements bring a significant performance benefit with respect to the previous version.

B. CVA6 Integration and its Data-Caches

The HPDcache is now publicly available, integrated with the CVA6 and the Sargantana[3] cores, two of the most mature RISC-V 64-bit open-source processors, proving its high-degree of versatility for application cores capable of booting Linux.

In this paper, we compare the performance of the HPDcache to the two existing CVA6 blocking data-caches: the standard write-back data-cache (STDcache) and another WT data-cache (WTDcache) initially implemented to support the OpenPiton multi-core framework [4]. As they share the same writing policy, we focus on comparing the performance between HPD-cache and WTDcache, demonstrating speedups up to 234%.

[1]https://github.com/openhwgroup/cva6
[2]https://github.com/openhwgroup/cv-hpdcache
[3]https://github.com/bsc-loca/sargantana

979-8-3503-4859-0/DATE24/© 2024 EDAA

III. Comparison and Performance Results

We used a hardware simulation platform consisting of a CVA6 core, a typical L1 data-cache size of 32 KB, and a main memory with a fixed access latency of 100 cycles (typical for DDR memory interfaces). For comparison purposes and due to the limitation of the other L1 caches, the width of the HPDcache memory interface is artificially restricted to 8-bytes. We focused on two highly memory-intensive benchmarks:

- Sparse Matrix Vector (SpMV) performs a matrix vector product on a matrix in the Compressed Sparse Row format. We evaluated three matrices, of dimensions 1000, 2000 and 4000, with a density of 1% (uniform distribution of the non-zero elements) and two real world matrices ;
- RaiderSTREAM [5] is a synthetic test which generates memory accesses typical of modern high-performance workloads. It reports achieved bandwidth for kernels with 400 KB arrays. We focus on the Add kernels, but results are similar with the other kernels.

A. Results Discussion

The results for SpMV and RaiderSTREAM are given in Fig. 1 and Fig. 2, respectively.

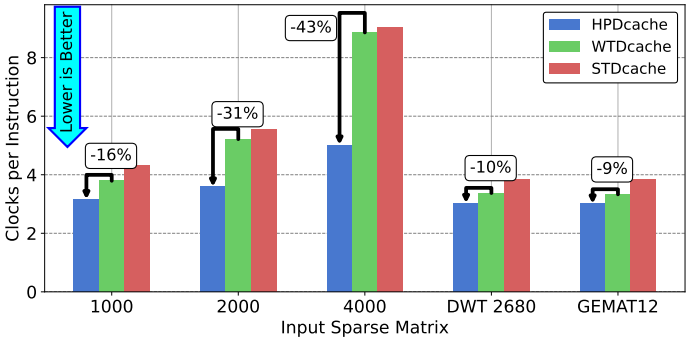

Fig. 1. Performance Comparison Based on Clocks per Instruction for SpMV

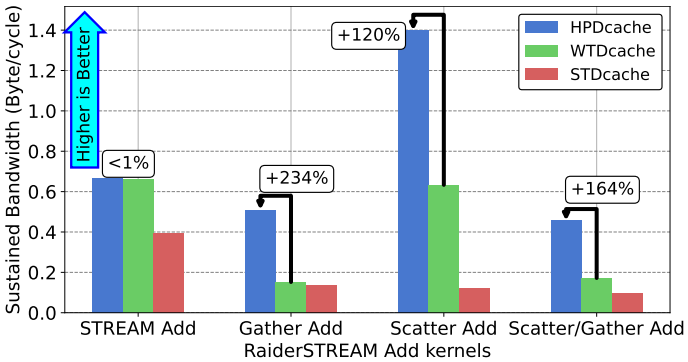

Fig. 2. Bandwidth Measurements using Add Kernels from RaiderSTREAM

On SpMV, taking the original WTDcache as a baseline, the HPDcache delivers speedups ranging from 9% to 43% (gmean=18%), which shows that the HPDcache is more suitable for memory-intensive scientific applications than the other caches. On RaiderSTREAM, the HPDcache outperforms (up to +234%) the original data-caches on the three memory intensive variants (Gather, Scatter and Scatter/Gather) of the Add kernels. These tests provoke frequent read misses and the results show that the HPDcache can simultaneously serve these misses, and thus hide the memory latency. Regarding the STREAM variant, where the accesses are less random, the HPDcache exhibits the same performance as the original WT data-cache.

B. Area and Timing Evaluation

We compared the post-synthesis silicon area and the maximum clock frequency of the HPDcache to the ones of the WTDcache in the GF22FDX technology (SSG 0.72V/125C corner). We considered the area and frequency of the CVA6 core plus the cache subsystem. The CVA6-HPDcache is 5.92% larger but has a 2.06% faster clock frequency than the CVA6-WTDcache (0.386mm^2@950MHz vs 0.364mm^2@931MHz). The width of the memory interface of the HPDcache can easily be increased up to 512 bits with virtually no area impact (<1% based on synthesis). This is because the sizing of internal buffers is independent of the memory bus width. This 512-bit memory interface improves the performance of the HPDcache up to 10% (RaiderSTREAM) and 6% (SpMV).

Conclusion and Future Work

We have demonstrated that memory-intensive applications exhibiting irregular memory access patterns benefit significantly from the HPDcache, on an in-order core, and have observed speedups of up to 234% with a small area increase of 5.92%.

Modern computer systems are multi-core and the HPDcache is now successfully integrated into OpenPiton [4], a widely adopted multi-core cache coherency system. This contribution has been validated by booting Linux in a 16-core configuration and the code is already available as open-source[4]. Future work will focus on evaluating the performance benefit of HPDcache for large, multi-core applications.

Acknowledgment

This work has received funding from the Key Digital Technologies Joint Undertaking (KDT JU) TRISTAN project under Grant Agreement nr. 10109594. It has been also partially supported by the European HiPEAC Network of Excellence, by the Spanish Ministry of Science and Innovation MCIN/AEI/10.13039/501100011033 (contracts PID2019-107255GB-C21 and PID2019-105660RB-C22), by the Generalitat de Catalunya (contract 2021-SGR-00763), by the Ministry of Economic Affairs and Digital Transformation, and by the European Union - Next Generation EU.

References

[1] D. Kroft, "Lockup-free Instruction Fetch/Prefetch Cache Organization," in *Proc. 8th Ann. Int'l Symp. Comput. Architect.*, 1981.

[2] C. Fuguet, "HPDcache: Open-Source High-Performance L1 Data Cache for RISC-V Cores," in *Proceedings of the 20th ACM International Conference on Computing Frontiers*, Association for Computing Machinery, 2023.

[3] J. L. Hennessy and D. A. Patterson, *Computer Architecture, Sixth Edition: A Quantitative Approach*. Morgan Kaufmann Publishers Inc., 6th ed., 2017.

[4] J. Balkind *et al.*, "OpenPiton+Ariane: The first Open-Source, SMP Linux-booting RISC-V System Scaling From One to Many Cores," in *Workshop on Computer Architecture Research with RISC-V (CARRV)*, 2019.

[5] M. Beebe *et al.*, "RaiderSTREAM: Adapting the STREAM Benchmark to Modern HPC Systems," in *IEEE High Performance Extreme Computing Conference*, 2022.

[4]https://github.com/PrincetonUniversity/openpiton/pull/136