Extending RISC-V Datapaths with Coarse-Grained Reconfigurable Architectures

Daniel Vázquez[1]*, Andrés Otero[1], Alfonso Rodríguez[1] and Eduardo de la Torre[1]

[1]Centro de Electrónica Industrial, Universidad Politécnica de Madrid, Madrid, Spain

Abstract

This work proposes extending the datapath of a RISC-V processor with a CGRA, a reconfigurable general-purpose accelerator for computing a wide range of applications with higher energy efficiency and lower execution times, through custom ISA extensions. This accelerator supports the execution of computing-intensive code sections, providing the RISC-V processor with spatially-distributed computing capabilities. The proposed system has been implemented and evaluated on an FPGA device, where experimental results show up to a 13× increase in execution performance for a synthetic benchmark.

Introduction

Thanks to their inherent parallelism, hardware accelerators can boost energy efficiency and execution performance in embedded applications running at the edge of the computing continuum. Besides, integrating hardware acceleration increases the near-sensor computing capabilities, reducing the need for data transmission to the cloud and, consequently, the energy consumption associated with communications. These benefits are particularly important in battery-powered Internet of Things (IoT) or wearable devices. For these reasons, heterogeneous computing fabrics are becoming prominent in the embedded domain. Moreover, the mainstream adoption of open hardware and, in particular, the RISC-V Instruction Set Architecture (ISA), eases the integration of custom hardware within the processor datapath.

Hardware accelerators can be categorised as domain-specific or general-purpose, usually with better performance in the former but at the cost of reduced flexibility. In computing architecture, the latter category usually focuses on exploiting parallelism at different levels without being too specific (e.g., vector processing units to exploit data-level parallelism). Alternatively, this work proposes using a Coarse-Grained Reconfigurable Architecture (CGRA) as a general-purpose hardware accelerator. CGRAs are capable of leveraging the benefits of spatially-distributed computation for offloading computing-intensive code sections while at the same time ensuring the flexibility required for general-purpose acceleration thanks to their reconfiguration capabilities. The work is an evolution of the preliminary solution presented in [1], extended with optimized control and memory mechanisms.

*Corresponding author: daniel.vazquez@upm.es.

This work has been funded by the MCIN/AEI /10.13039/501100011033 and the European Union NextGenerationEU/PRTR under the project COGNITION (ref. TED2021-132475B-I00)

CGRAs

CGRAs are attractive computing platforms providing simultaneous high performance and power efficiency by exploiting word-level operators and special-purpose interconnections. Although initially conceived as stand-alone reconfigurable devices with coarse-grain resources (as an alternative to fine-grain FPGAs), CGRAs can also be deployed as virtualization layers on top of FPGA resources, acting as a matrix of interconnected Processing Elements (PEs) that are programmable at operator level: each PE executes an arithmetic operation and transfers data over an interconnection network [2]. PEs can be programmed each clock cycle in time-multiplexed CGRAs or remain unchanged during complete task executions in spatially-configured architectures.

The proposal in this work relies on a homogeneous spatially-distributed CGRA, forming a computing matrix of size $N \times M$. Each PE has dedicated logic to provide routing with the neighbouring PEs, and computing features, supporting additions, subtractions, multiplications, shifts, and logic operations. The behaviour of the PEs is defined by configuration registers that control multiplexers, logic masks and the operation performed by their internal ALUs.

Two concepts must be considered to exploit this CGRA as a hardware accelerator. First, there is a need to change its functionality at run time to execute the different sections offloaded from the application Data-Flow Graphs (DFGs) onto the computing fabric. Therefore, reconfiguration must be as fast as possible. Second, a high-bandwidth memory interface is essential: once an application DFG is mapped onto the CGRA, it can provide an initiation interval of 1 clock cycle if the system is continuously fed.

CGRA integration with RISC-V

The RISC-V ISA provides predefined encoding ranges to extend the computing capabilities through custom

instructions. The proposal in this work leverages this mechanism to integrate the CGRA to minimize the control overheads during consecutive invocations to the accelerator. Some existing open hardware solutions already provide an instantiation wrapper for custom ISA extensions using interfaces to communicate the accelerator from/to the processor [3, 4]. In this work, the CGRA is integrated as a co-processor of a RISC-V core, using configuration, load, store and synchronization instructions.

Different alternatives exist to load and store data from/to the accelerator, including a scratchpad memory [5]. However, with scratchpad memories, the execution and data transfers cannot be overlapped, and the memory size limits the amount of data that can be processed simultaneously. Alternatively, in this work, dedicated Direct Memory Access (DMA) nodes are used, enabling simultaneous data transfers that benefit the target architecture.

More in detail, the Chipyard platform [6] is used in this work to generate a System on Chip (SoC) with a Small Rocket Core (*RV32IMAC*). The accelerator, integrated through the RoCC interface, is composed of a control unit that handles the instructions received from the core and controls the memory infrastructure, the input and output memory nodes that load and store the data consumed by the CGRA, the configuration memory node that loads the CGRA configuration, and the $N \times M$ reconfigurable architecture. The block diagram of the system can be seen in 1.

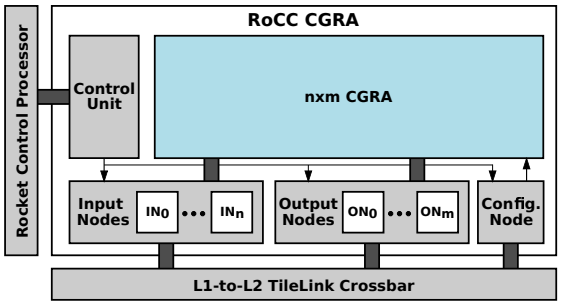

Figure 1: *Block diagram of the RoCC-based CGRA.*

Results

The resulting SoC has a core with the co-processor, cache memories, and an external memory controller to access an off-chip RAM. It has been implemented on a FPGA running at 50 MHz and benchmarked in [1] with applications that execute nested vector operations and reductions, being the *cap* algorithm the most complex one with 17 active PEs hosting 8 operations, using two vector operands and providing one vector result. The obtained speed-ups can be seen in 2, comparing the execution in the co-processor to a pure software execution in the Small Rocket Core.

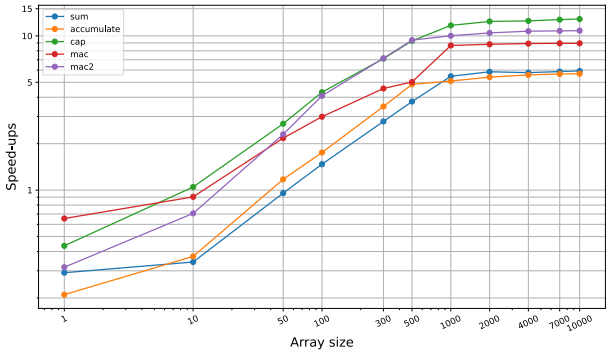

Figure 2: *Results of the RoCC-based CGRA [1].*

Conclusions and future work

The proposed system can leverage spatially-distributed computing to accelerate computing-intensive sections of the application code. By overlapping the memory transfers and execution, speed-ups can be obtained starting from small array sizes, and a maximum acceleration of 13× is obtained using synthetic benchmarks.

As future work regarding the Rocket Custom Coprocessor (RoCC) wrapper, architectural improvements to support wider bus lengths, different element width sizes and multi-frequency designs are proposed. On the CGRA side, a design-space exploration to assess homogeneous and heterogeneous designs is needed. Finally, an ASIC tape-out of the design in a low-power platform will be carried out.

References

[1] Daniel Vázquez et al. "Extending RISC-V Processor Datapaths with Multi-Grain Reconfigurable Overlays". In: *2022 37th Conference on Design of Circuits and Integrated Circuits (DCIS)*. 2022, pp. 01–06. DOI: 10.1109/DCIS55711.2022.9970069.

[2] Anuj Vaishnav, Khoa Dang Pham, and Dirk Koch. "A Survey on FPGA Virtualization". In: *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*. 2018, pp. 131–1317. DOI: 10.1109/FPL.2018.00031.

[3] Krste Asanović et al. *The Rocket Chip Generator*. Tech. rep. UCB/EECS-2016-17. EECS Department, University of California, Berkeley, Apr. 2016. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-17.html.

[4] *cv32e40x*. Tech. rep. OpenHW group. URL: https://docs.openhwgroup.org/projects/cv32e40x-user-manual/en/latest/index.html.

[5] Rafael Zamacola, Andrés Otero, and Eduardo de la Torre. "Multi-grain reconfigurable and scalable overlays for hardware accelerator composition". In: *Journal of Systems Architecture* 121 (2021), p. 102302. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2021.102302.

[6] Alon Amid et al. "Chipyard: Integrated Design, Simulation, and Implementation Framework for Custom SoCs". In: *IEEE Micro* 40.4 (2020), pp. 10–21. ISSN: 1937-4143. DOI: 10.1109/MM.2020.2996616.