# A FPGA COPROCESSOR FOR ACCELERATING GEOMETRIC ALGEBRA ALGORITHMS BASED ON THE GAPPCO DESIGN

**S. Franchini**[a], **D. Hildenbrand**[b], **E. Saribatir**[c], and **S. Vitabile**[d]

[a]Institute for High Performance Computing and Networking
National Research Council of Italy (ICAR-CNR)
silvia.franchini@icar.cnr.it

[b]University of Technology Darmstadt, Germany
dietmar.hildenbrand@gmail.com

[c]School of Mathematical and Physical Sciences
University of Technology Sydney
ed.saribatir@student.uts.edu.au

[d]Department of Biomedicine, Neuroscience, Advanced Diagnostics (Bi.N.D.)
University of Palermo, Italy
salvatore.vitabile@unipa.it

## Abstract

The high computational complexity of Geometric Algebra (GA) operations requires dedicated architectures to allow for their practical use in real-world applications. An integrated hardware-software system that exploits both the GAALOPWeb compiler and the GAPPCO coprocessor to accelerate GA algorithms is presented in this paper. GAPPCO is a reconfigurable coprocessor that can be programmed to support different GA algorithms. The GAPPCO coprocessor has been implemented as a System-on-Chip (SoC) on the Avnet Picozed development board containing both an ARM Cortex-A9 processor and a Xilinx Zynq-7000 FPGA device. A basic algorithm for the computation of a 5D vector reflection, which is the basis for all conformal geometric operations in Conformal Geometric Algebra (CGA), namely, rotations, translations, and dilations, has been used to test the effectiveness of the proposed approach. Preliminary results show that the integrated GAALOPWeb-GAPPCO system achieves average speedups of about 5× against the execution of the same algorithm on the standard ARM processor embedded in the Zynq-7000 chip.

## 1 Introduction

Efficient implementations of Geometric Algebra algorithms are required to address their high computational costs and improve performance in real-time applications. Here, we review the existing state-of-the-art implementations of Geometric Algebra subdivided into software implementations, full-hardware implementations and mixed software/hardware implementations.

An approach to overcome the runtime limitations of Geometric Algebra has been achieved

through optimized software solutions. Tools have been developed for high-performance implementations of Geometric Algebra algorithms such as the C++ software library generator Gaigen 2 from Daniel Fontijne and Leo Dorst of the University of Amsterdam [1], GMac from Ahmad Hosney Awad Eid of Suez Canal University [2] and the Versor library [3] from Pablo Colapinto.
The web page BiVector.net gives a good overview over software solutions for Geometric Algebra. It provides code generators for C++, C#, Python, Rust as well as starting points for libraries for Python, C/C++ and Julia.
The GAALOPWeb compiler [4, 5] supports many programming languages as the output language of the optimized code generated by GAALOP. C/C++, OpenCL and CUDA [6] as well as Python, Matlab, Mathematica, Julia or Rust are some of the languages that can be selected. Further details on GAALOPWeb are in Section 2.1.
Another approach is to look for dedicated hardware architectures for the acceleration of Geometric Algebra algorithms. Integrated circuit technology offers a means to achieve high performance with field-programmable gate arrays (FPGAs). The first serious approach was described by Perwass et al. [7] in 2003. A different approach was presented by Gentile et al. [8] in 2005. An update on this work was given by Franchini et al. in a series of papers such as [9, 10]. The first custom-fabricated integrated circuit ASIC implementation was introduced by Mishra and Wilson [11] in 2006.

In this paper, we are using an integrated hardware/software approach based on the GAALOPWeb compiler generating configuration data for the dedicated hardware design GAPPCO. As described in the following sections, the GAPPCO coprocessor has been prototyped as a System-on-Chip on the Xilinx Zynq-7000 FPGA device. Experimental tests on the prototype have shown average 5× speedups against the execution on a standard general-purpose CPU.

# 2 Proposed System

The proposed approach to accelerate GA algorithms is based on a mixed hardware/software system including the software compiler GAALOPWeb and the configurable coprocessor GAPPCO.

## 2.1 GAALOPWeb for GAPPCO

GAALOPWeb [4, 5] is a web interface that allows a Geometric Algebra algorithm to be described using GAALOPScript and entered as text to the web page, and settings for variable assignments and options for output code and/or visualisations to be displayed. It allows a user to generate optimised code for different languages and parallel frameworks such as OpenGL, CUDA, and FGPA. GAALOPWeb also has a web API that allows optimised code to be generated with GAALOPScript code as input.

## 2.2 GAPPCO Coprocessor

GAPPCO has been designed and implemented starting from the proof-of-concept coprocessor presented in [12]. GAPPCO block diagram is reported in Fig. 1.

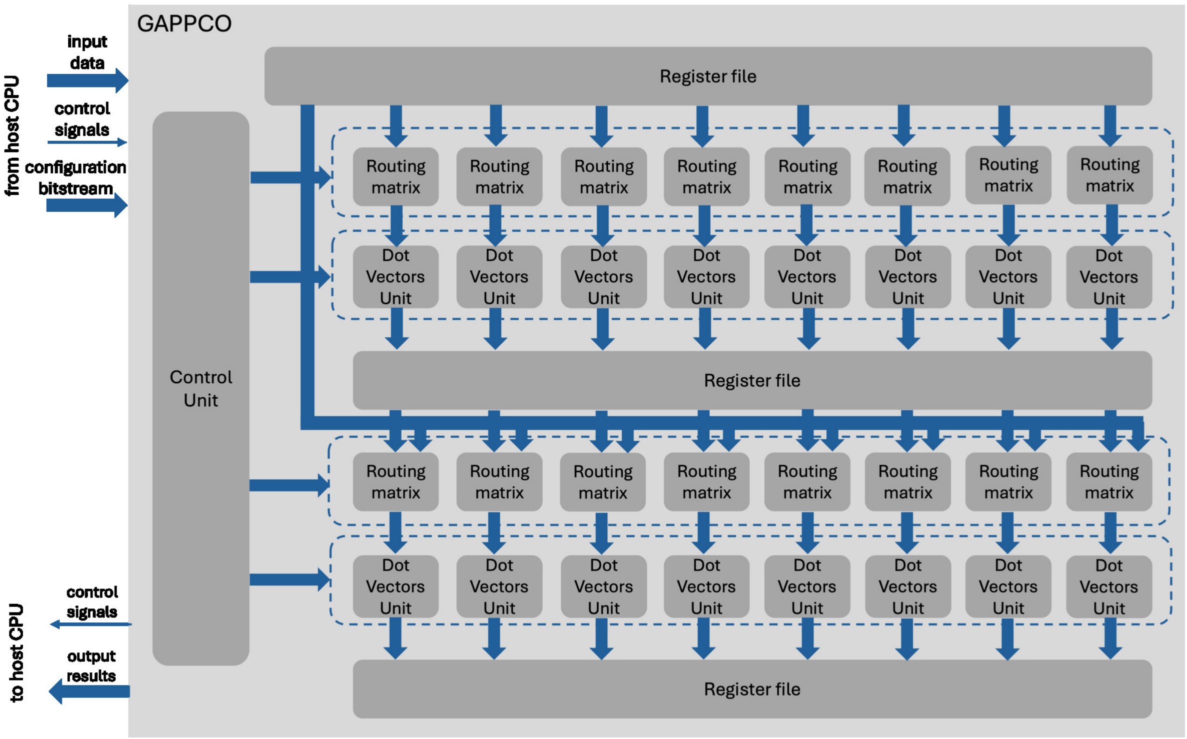

Figure 1: GAPPCO block diagram.

GAPPCO is a configurable coprocessor that can be programmed according to GAALOPWeb configuration data to support different GA algorithms. The coprocessor is composed of the following functional units:

- *DotVectors units* that are the processing units conceived to execute the basic arithmetic operations required after the compilation by GAALOPWeb, namely, sums of products. Each DotVectors unit can be configured to execute two sums of two products or one sum of four products according to the architecture presented in [12]. Several DotVectors units can be linked together to form larger processing units (GAPP units) able to support more complex GA-based computations. Furthermore, different GAPP units can be configured to work in parallel and further accelerate algorithm execution. DotVectors units as well as GAPP units have a pipeline structure so that the GAPPCO architecture is based on a set of pipelines working in parallel;

- *Register files* containing input data, intermediate results, and output results, respectively;

- *Routing matrices* that can be programmed for properly routing data between registers and processing units and connecting different DotVectors units;

- *Control unit* that supervises the coprocessor operations and data exchange with host CPU during both the configuration phase and the processing phase.

The GAPPCO architecture is easily scalable since the basic processing unit can be upgraded to support other basic operations, while the number of processing units can be increased, as long as the available hardware resources allow it, in order to achieve a higher degree of parallelism.

## 2.3 FPGA Implementation

As depicted in Fig. 2, the GAPPCO coprocessor has been prototyped on the Avnet PicoZed development board equipped with a Xilinx Zynq-7000 xc7z030sbg485-1 FPGA device. The Zynq-7000 chip contains an ARM Cortex-A9 processing system (PS) as well as the programmable logic (PL) fabric to be configured to implement custom hardware units. The GAPPCO prototype has been implemented as an embedded SoC composed of the ARM standard processor and the GAPPCO specialized coprocessor. The coprocessor has been integrated in the PL of the FPGA chip and connected as a custom peripheral (IP-core) to the high-bandwidth AMBA-AXI bus of the ARM processor. The operating frequency is 333 MHz for both the ARM CPU and the GAPPCO coprocessor.

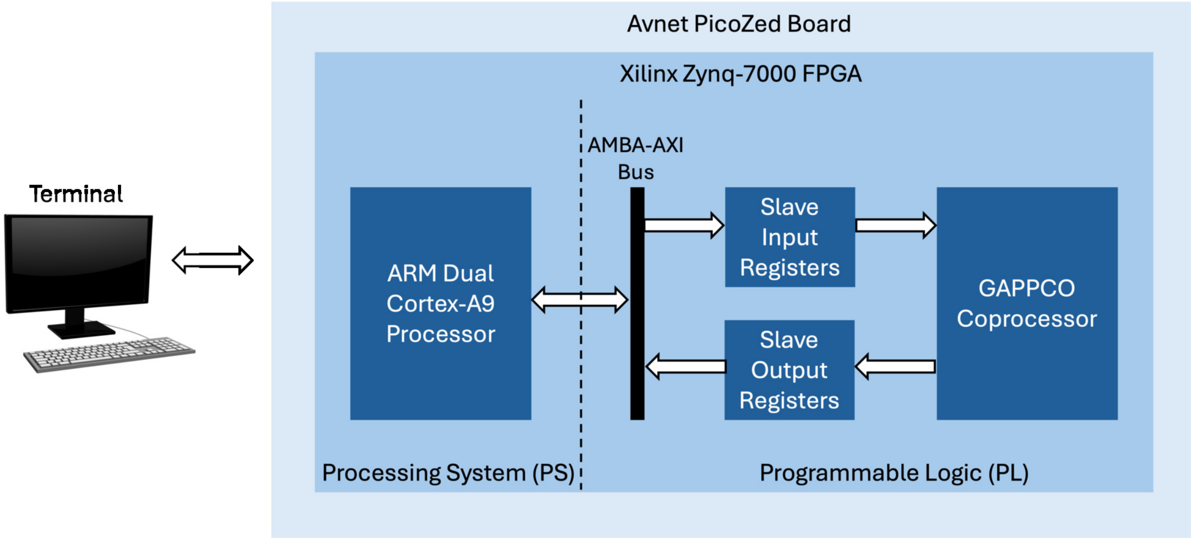

Figure 2: GAPPCO prototype on the PicoZed board.

Before runtime, the configuration bitstream generated by the GAALOPWeb compiler is downloaded from the ARM processor to the peripheral in order to customize GAPPCO for the required GA algorithm. The GAPPCO control unit handles the configuration data distribution to the programmable units of the coprocessor. The system operation at runtime is organized according to the following phases:

1. The ARM processor writes input data to the GAPPCO input registers.
2. GAPPCO reads input data from input registers.
3. GAPPCO pipelines execute the required computations.
4. GAPPCO writes results to the output registers.
5. The ARM processor reads results from the GAPPCO output registers.

Table 1 summarizes the resource utilization of the GAPPCO prototype, composed of 16 DotVectors units, on the Xilinx Zynq-7000 FPGA device. The most used resources are the DSP units exploited for the implementation of the 32-bit floating point multipliers/adders within the DotVectors units.

Table 1: GAPPCO resource utilization on Xilinx Zynq-7000 xc7z030sbg485-1 FPGA.

| Resource | Used | Available | Utilization |
|---|---:|---:|---:|
| LUT | 38 496 | 78 600 | 48.98% |
| FF | 29 818 | 157 200 | 18.97% |
| DSP | 288 | 400 | 72.00% |

## 3 Experimental Results

To test the effectiveness of the proposed system, we have used as testbench a basic algorithm for the computation of a 5D vector reflection in the Conformal Geometric Algebra (CGA), which is the basis for all the conformal geometric transformations (rotations, translations, and dilations), as described in [9]. The same algorithm has been executed on both the standard ARM processor and the GAPPCO coprocessor. Table 2 reports the performance comparison in terms of execution times (expressed in clock cycles at 333 MHz) for different numbers of executions. When the number of executions increases, the effects of the parallelism techniques of the coprocessor (parallel processing on multiple pipelines) can be observed. Achieved speedup is 5.5×. However, it has been also observed that, in the GAPPCO execution, most of the time is taken by ARM-GAPPCO data transfers.

Table 2: Average execution times (in clock cycles at 333 MHz).

| Number of executions | ARM CPU ($t_1$) | GAPPCO coprocessor ($t_2$) | Speedup ($t_1/t_2$) |
|---:|---:|---:|---:|
| 1 | 148 | 123 | 1.2× |
| 10 | 1 171 | 281 | 4.2× |
| 100 | 10 891 | 2 015 | 5.4× |
| 1 000 | 108 090 | 19 745 | 5.5× |
| 10 000 | 1 080 113 | 197 045 | 5.5× |

## 4 Conclusions

A hybrid hardware/software system based on the GAALOPWeb compiler and the GAPPCO reconfigurable coprocessor for accelerating Geometric Algebra algorithms has been presented in this paper. Parallelism techniques, such as pipeline and multi-core processing, have been exploited to improve GAPPCO execution speed. An FPGA prototype of the GAPPCO coprocessor showing average 5× speedups against the execution on a standard CPU has been implemented. The GAPPCO architecture has been conceived to be easily scalable so that better speedups could be obtained on larger FPGA devices with faster data transfer interfaces.

# References

[1] Fontijne, Daniel, Efficient Implementation of Geometric Algebra. In *Dissertation University of Amsterdam*, 2007.

[2] Ahmad Hosney Awad Eid, Optimized Automatic Code Generation for Geometric Algebra Based Algorithms with Ray Tracing Application. In *Thesis at Suez Canal University, Port Said*, 2010.

[3] Pablo Colapinto, Articulating Space: Geometric Algebra for Parametric Design-Symmetry, Kinematics, and Curvature. In *Dissertation at UC Santa Barbara*, 2016.

[4] http://www.gaalop.de/gaalopweb
Access date: 07/03/2024

[5] Hildenbrand, Dietmar, The Power of Geometric Algebra Computing: For Engineering and Quantum Computing, 2021.

[6] Hildenbrand, Dietmar, Foundations of Geometric Algebra Computing In *Springer book series Geometry and Computing*, 2013.

[7] Perwass, Christian and Gebken, Christian and Sommer, Gerald, Implementation of a Clifford algebra co-processor design on a field programmable gate array. In *Clifford Algebras: Application to Mathematics, Physics, and Engineering*, 2003.

[8] Gentile, Antonio and Segreto, Salvatore and Sorbello, Filippo and Vassallo, Giorgio and Vitabile, Salvatore and Vullo, Vincenzo, CliffoSor, an Innovative FPGA-based Architecture for Geometric Algebra. In *ERSA 2005*, 2005.

[9] Franchini, S., Gentile, A., Sorbello, F., Vassallo, G., and Vitabile, S., ConformalALU: a Conformal Geometric Algebra Coprocessor for Medical Image Processing. In *IEEE Transactions on Computers*, vol. 64, n. 4, pp. 955-970, 2015.

[10] Franchini, S., Gentile, A., Sorbello, F., Vassallo, G., and Vitabile, S., Design and Implementation of an Embedded Coprocessor with Native Support for 5D, Quadruple-Based Clifford Algebra. In *IEEE Transactions on Computers* vol. 62, n. 12, pp. 2366-2381, 2013.

[11] Mishra, Biswajit and Wilson, Peter R, Color edge detection hardware based on geometric algebra. In *European Conference on Visual Media Production (CVMP)*, 2006.

[12] Hildenbrand, D., Franchini, S., Gentile, A., Vassallo, G., and Vitabile, S., GAPPCO: An Easy to Configure Geometric Algebra Coprocessor Based on GAPP Programs. In *Advances in Applied Clifford Algebras*, volume 27, n. 3, pp. 2115–2135, 2017.