UNIVERSITÀ DEGLI STUDI DI PALERMO

DIPARTIMENTO DI BIOMEDICINA, NEUROSCIENZE E DIAGNOSTICA AVANZATA (Bi.N.D.)

Embedded Coprocessors for Native Execution of Geometric Algebra Operations

Salvatore Vitabile

University of Palermo - Italy

Geometric Algebra (GA)

- Unifying mathematical language
- Simple and intuitive modeling of geometric objects and their transformations
- Coordinate-free representation obtained by using extended objects in high-dimensional vector spaces
- Homogeneous model → Four-dimensional Geometric Algebra
- Conformal model → Five-dimensional Geometric Algebra

Computational issues

- The generic element of an n-dimensional Geometric Algebra has 2^n coordinates
- The generic element of 5D Geometric Algebra has 32 coefficients requiring more than a thousand of multiply-adds between coefficient combinations
- Significant computational costs demand for dedicated software and/or hardware architectures

State-of-the-art implementations (1)

- Software implementations
  - Software libraries (Gaigen, GluCat)
  - Packages for symbolic and numerical environments
    - Clifford and GA packages for Maple
    - Gable and Clifford Multivector Toolbox packages for Matlab
    - Grassmann Algebra package for Mathematica
  - Stand-alone programs (CluCalc)

State-of-the-art implementations (2)

- Mixed software-hardware implementations
  - Gaalop (Geometric Algebra ALgorithm OPtimizer)
    - A pre-compiler converts GA algorithms to an intermediate representation that can be further compiled to run on different software/hardware platforms
    - GAPPCO (GAPP coprocessor)
- Fully hardware implementations
  - FPGA-based coprocessor by Perwass et al.
    - Geometric products for algebras of dimension up to 8
    - 24-bit integer numbers to represent blade coefficients
  - ASIC-based coprocessor by Mishra et al.
    - Color edge detection applications
    - 3D vector rotations
    - 64-bit floating point numbers

Our contribution

- Design space exploration of hardware implementations that directly support native Geometric Algebra operations
  - Resulting family of embedded coprocessors for the native execution of up to 5D GA operations
- Hardware-oriented representations of GA elements and operators properly conceived to obtain fast performing implementations
- FPGA-based prototypes showing speedups of about one order of magnitude with respect to the baseline software library Gaigen
  - Execution analysis of a suite of GA-based applications
- GAPPCO (GAPP coprocessor)

Design space exploration

- The design space of hardware architectures that natively support GA operations has been explored along several axes:
  - Clifford number representation (homogeneous elements vs quadruples)
  - Execution flow (sequential vs parallel, scalar vs pipeline)
  - Number of multiply-add units
  - Coefficient precision
  - Datapath width
  - Instruction word length
- Several alternative architectures based on different sets of architectural parameters have been explored and different design points have been implemented and prototyped

Coprocessor family

Coprocessor	GA operations	Clifford number representation	Execution flow	Parallelism techniques	N. of multipliers	Precision	Instruction word length
S-CliffoSor	4D products, sums, diff.	Homogeneous elements	Sequential	-	-	32-bit integers	15x32-bit words = 480 bits
CliffoSor	4D products, sums, diff. 3D rotations (Operations on bivectors not implemented)	Homogeneous elements	Parallel	-	24	32-bit integers	15x32-bit words = 480 bits
Quad-CliffoSor	4D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	16-bit floating point	9x16-bit words = 144 bits
CliffordALU5	4D/5D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	32-bit floating point	9x32-bit words = 288 bits
ConformalALU	5D conformal geometric operations (reflections, rotations, translations, dilations)	5D vectors	Parallel	Pipeline	5	32-bit floating point	16x32-bit words = 512 bits

Coprocessor family

Coprocessor	GA operations	Clifford number representation	Execution flow	Parallelism techniques	N. of multipliers	Precision	Instruction word length
S-CliffoSor	**4D products, sums, diff.**	**Homogeneous elements**	**Sequential**	**-**	**-**	**32-bit integers**	**15x32-bit words = 480 bits**
CliffoSor	4D products, sums, diff. 3D rotations (Operations on bivectors not implemented)	Homogeneous elements	Parallel	-	24	32-bit integers	15x32-bit words = 480 bits
Quad-CliffoSor	4D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	16-bit floating point	9x16-bit words = 144 bits
CliffordALU5	4D/5D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	32-bit floating point	9x32-bit words = 288 bits
ConformalALU	5D conformal geometric operations (reflections, rotations, translations, dilations)	5D vectors	Parallel	Pipeline	5	32-bit floating point	16x32-bit words = 512 bits

S-CliffoSor (Sliced Clifford coprocesSor)

- 4D Clifford operations between homogeneous elements

Homogeneous element	Grade	N. of blades	Blades					
Scalar (s)	0	1	a_0					
Vector (v)	1	4	a_1e_1	a_2e_2	a_3e_3	a_4e_4		
Bivector (b)	2	6	$a_{12}e_1e_2$	$a_{13}e_1e_3$	$a_{14}e_1e_4$	$a_{23}e_2e_3$	$a_{24}e_2e_4$	$a_{34}e_3e_4$
Trivector (t)	3	4	$a_{123}e_1e_2e_3$	$a_{124}e_1e_2e_4$	$a_{134}e_1e_3e_4$	$a_{234}e_2e_3e_4$		
Pseudoscalar (p)	4	1	$a_{1234}e_1e_2e_3e_4$					

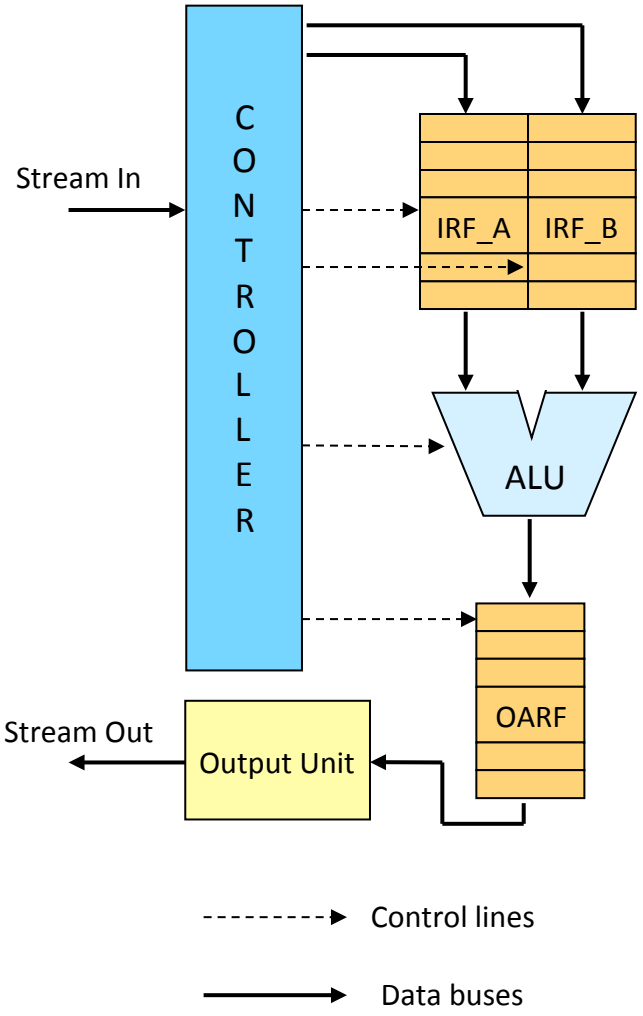
- Each 4D multivector is a tuple of 5 homogeneous elements m = (s, v, b, t, p)
- Each homogeneous element contains a different number of blades (variable-size representation)

Blade representation

Basis blade	Bit mask
1	0000
e_1	0001
e_2	0010
e_3	0100
e_4	1000
e_1e_2	0011
e_1e_3	0101
e_1e_4	1001
e_2e_3	0110
e_2e_4	1010
e_3e_4	1100
$e_1e_2e_3$	0111
$e_1e_2e_4$	1011
$e_1e_3e_4$	1101
$e_2e_3e_4$	1110
$e_1e_2e_3e_4$	1111

- Each blade is a coefficient-basis blade pair
- A 4-bit mask is associated with each basis blade
- The geometric product of two basis blades is the XOR of the associated bit masks

S-CliffoSor architecture



- 32-bit ALU (sum, subtraction, multiplication, logical operations)
- Each Clifford operation is decomposed into the proper sequence of basic operations whose sequential execution is supervised by the control unit
- Long per-operation latency
- The single S-CliffoSor slice can be replicated for parallel execution of multiple Clifford operations

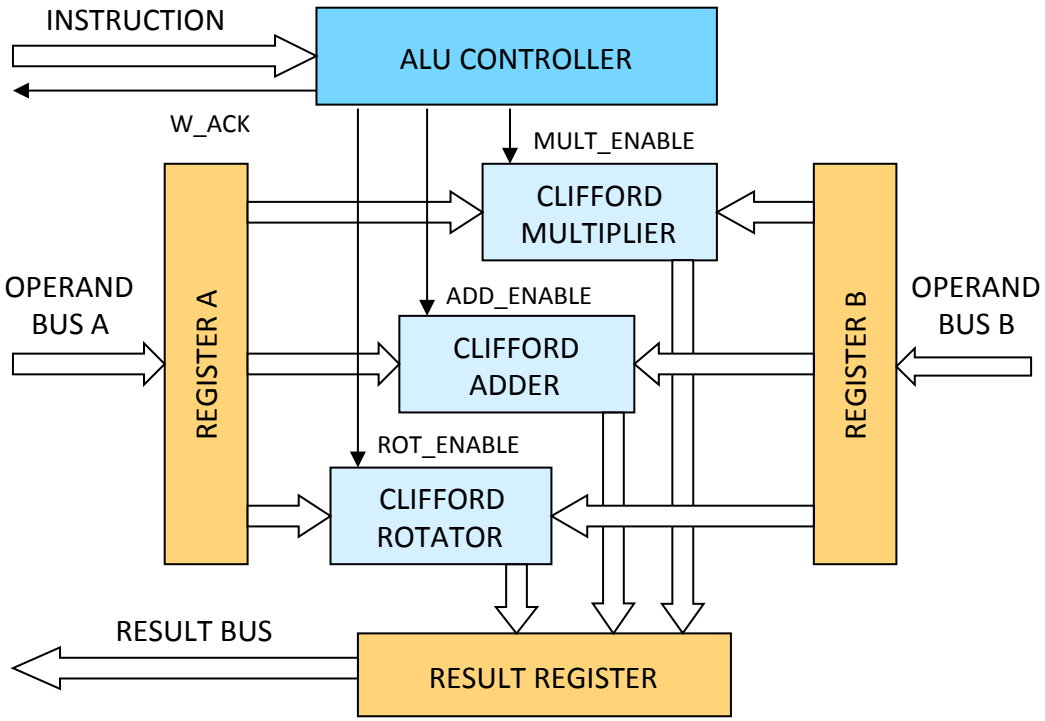
Reference

S. Franchini, A. Gentile, M. Grimaudo, C.A. Hung, S. Impastato, F. Sorbello, G. Vassallo, and S. Vitabile, **A Sliced Coprocessor for Native Clifford Algebra Operations**, in Proc. of *10th IEEE Euromicro Conference on Digital System Design - Architectures, Methods and Tools (DSD 2007)*, Lübeck, Germany, August 29-31, pp. 436-439, IEEE Computer Society Press, **2007**.

Coprocessor family

Coprocessor	GA operations	Clifford number representation	Execution flow	Parallelism techniques	N. of multipliers	Precision	Instruction word length
S-CliffoSor	4D products, sums, diff.	Homogeneous elements	Sequential	-	-	32-bit integers	15x32-bit words = 480 bits
CliffoSor	**4D products, sums, diff. 3D rotations (Operations on bivectors not implemented)**	**Homogeneous elements**	**Parallel**	**-**	**24**	**32-bit integers**	**15x32-bit words = 480 bits**
Quad-CliffoSor	4D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	16-bit floating point	9x16-bit words = 144 bits
CliffordALU5	4D/5D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	32-bit floating point	9x32-bit words = 288 bits
ConformalALU	5D conformal geometric operations (reflections, rotations, translations, dilations)	5D vectors	Parallel	Pipeline	5	32-bit floating point	16x32-bit words = 512 bits

CliffoSor (Clifford coprocesSor)



- 4D Clifford operations between homogeneous elements
- Three functional units for 4D Clifford products, 4D Clifford sums/differences, and 3D Clifford rotations
- Parallel architecture
- The most complex operations on bivectors are not directly supported in hardware (24 parallel multipliers instead of 36)

Reference

S. Franchini, A. Gentile, F. Sorbello, G. Vassallo, and S. Vitabile, **An Embedded, FPGA-based Computer Graphics Coprocessor with Native Geometric Algebra Support**, in *Integration, The VLSI Journal*, Vol. 42, No. 3, pp. 346-355, **2009**

Coprocessor family

Coprocessor	GA operations	Clifford number representation	Execution flow	Parallelism techniques	N. of multipliers	Precision	Instruction word length
S-CliffoSor	4D products, sums, diff.	Homogeneous elements	Sequential	-	-	32-bit integers	15x32-bit words = 480 bits
CliffoSor	4D products, sums, diff. 3D rotations (Operations on bivectors not implemented)	Homogeneous elements	Parallel	-	24	32-bit integers	15x32-bit words = 480 bits
Quad-CliffoSor	**4D products, sums, diff., unary operations**	**Quadruples**	**Parallel**	**Pipeline**	**16**	**16-bit floating point**	**9x16-bit words = 144 bits**
CliffordALU5	4D/5D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	32-bit floating point	9x32-bit words = 288 bits
ConformalALU	5D conformal geometric operations (reflections, rotations, translations, dilations)	5D vectors	Parallel	Pipeline	5	32-bit floating point	16x32-bit words = 512 bits

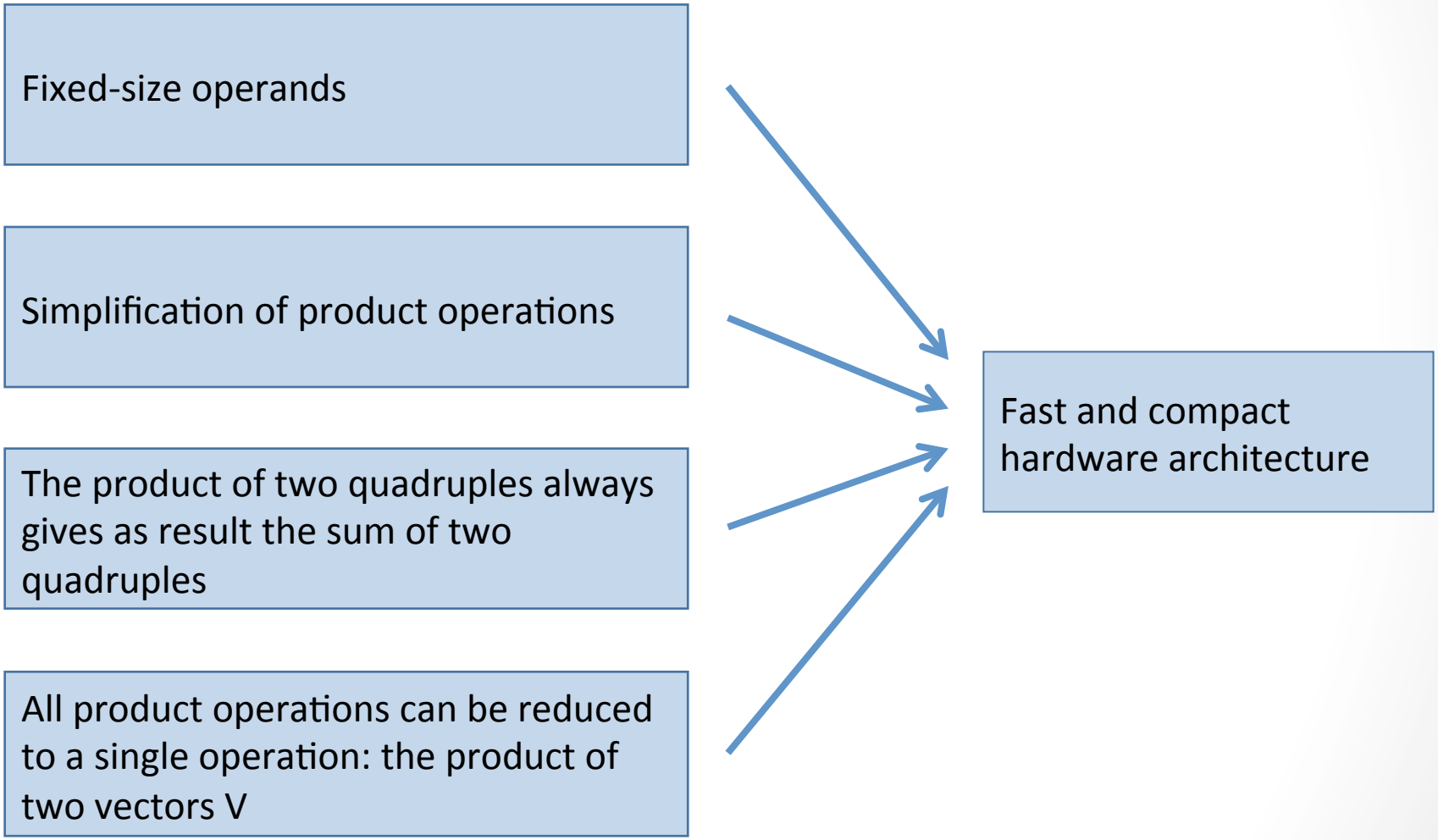
Quad-CliffoSor (Quadruple-based Clifford coprocesSor)

- New fixed-size representation based on quadruples

Quadruple	Blades			
V	a_1e_1	a_2e_2	a_3e_3	a_4e_4
T	$a_{234}e_2e_3e_4$	$a_{134}e_1e_3e_4$	$a_{124}e_1e_2e_4$	$a_{123}e_1e_2e_3$
S	$a_{14}e_1e_4$	$a_{24}e_2e_4$	$a_{34}e_3e_4$	a_0
P	$a_{23}e_2e_3$	$a_{13}e_1e_3$	$a_{12}e_1e_2$	$a_{1234}e_1e_2e_3e_4$

- The generic 4D multivector can be expressed as a tuple of 4 quadruples
  m = (V, T, S, P)
- Each quadruple is composed of 4 blades (fixed-size representation)

Quadruples – Advantages (1)



Quadruples – Advantages (2)

- The product of two quadruples results in the sum of two quadruples

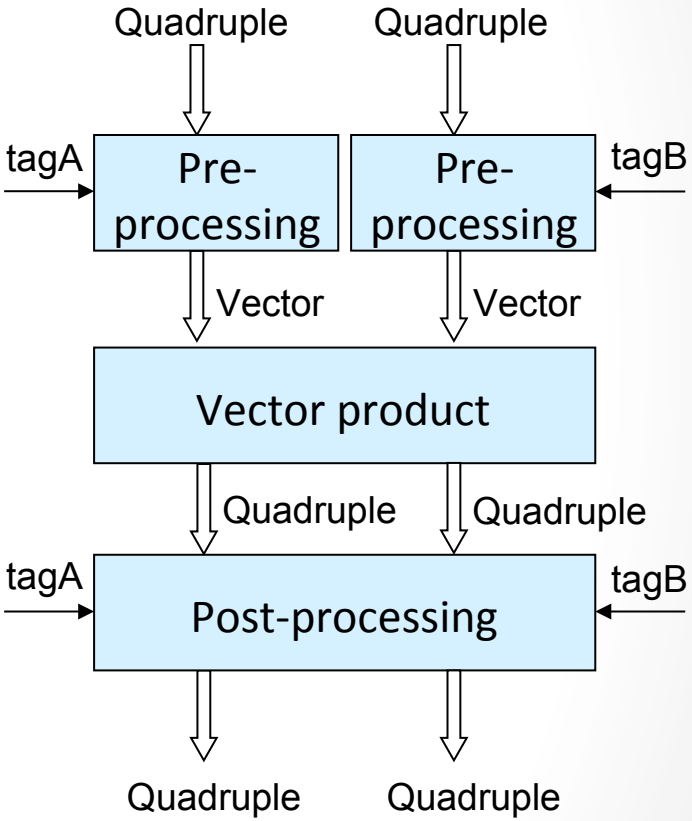
Result type for geometric product (GP) between quadruples

GP	V	T	S	P
V	S+P	S+P	V+T	V+T
T	S+P	S+P	V+T	V+T
S	V+T	V+T	S+P	S+P
P	V+T	V+T	S+P	S+P

- A single fixed format can be used for both input data and output result

Quadruples – Advantages (3)

- Simplified algorithm for fast execution of product operations on quadruples
  - Any quadruple can be reduced to a quadruple of type V (by simple sign changing operations on its coefficients)
  - The product between any couple of quadruples can be reduced to the product of two vectors V by simple pre-processing (sign-changing) and post-processing (sign changing and/ or swapping)
  - Each product operation always requires 16 multiplications and 9 sums
  - Simplified and compact structure of the Quad-CliffoSor multiplier unit



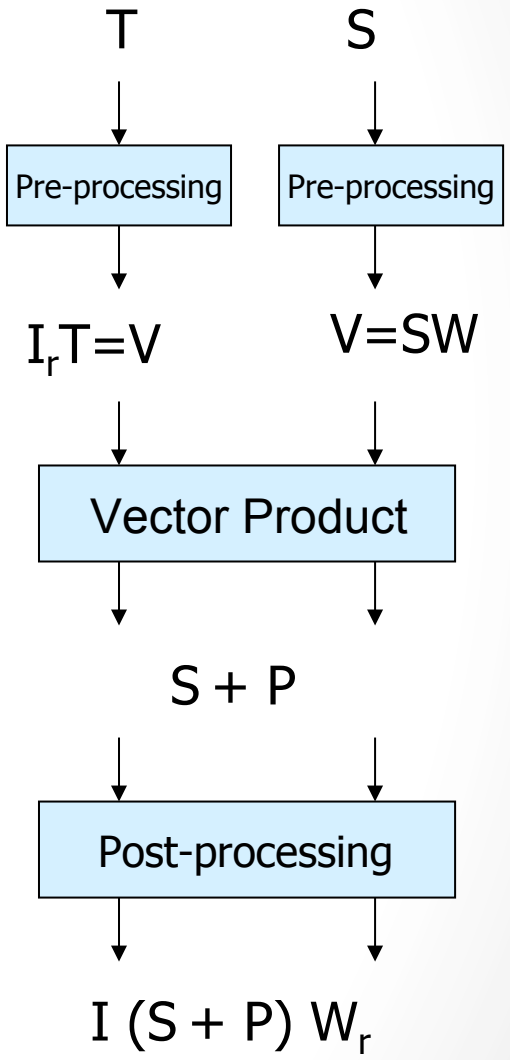
Quadruples – Advantages (4)

- Quadruples T, S and P can be transformed into vectors V by simply multiplying them by one of the following elements:

$I = \boldsymbol{e}_0\boldsymbol{e}_1\boldsymbol{e}_2\boldsymbol{e}_3 = I_r$
$W = \boldsymbol{e}_0 = W_r$
$Z = \boldsymbol{e}_1\boldsymbol{e}_2\boldsymbol{e}_3 = -Z_r$

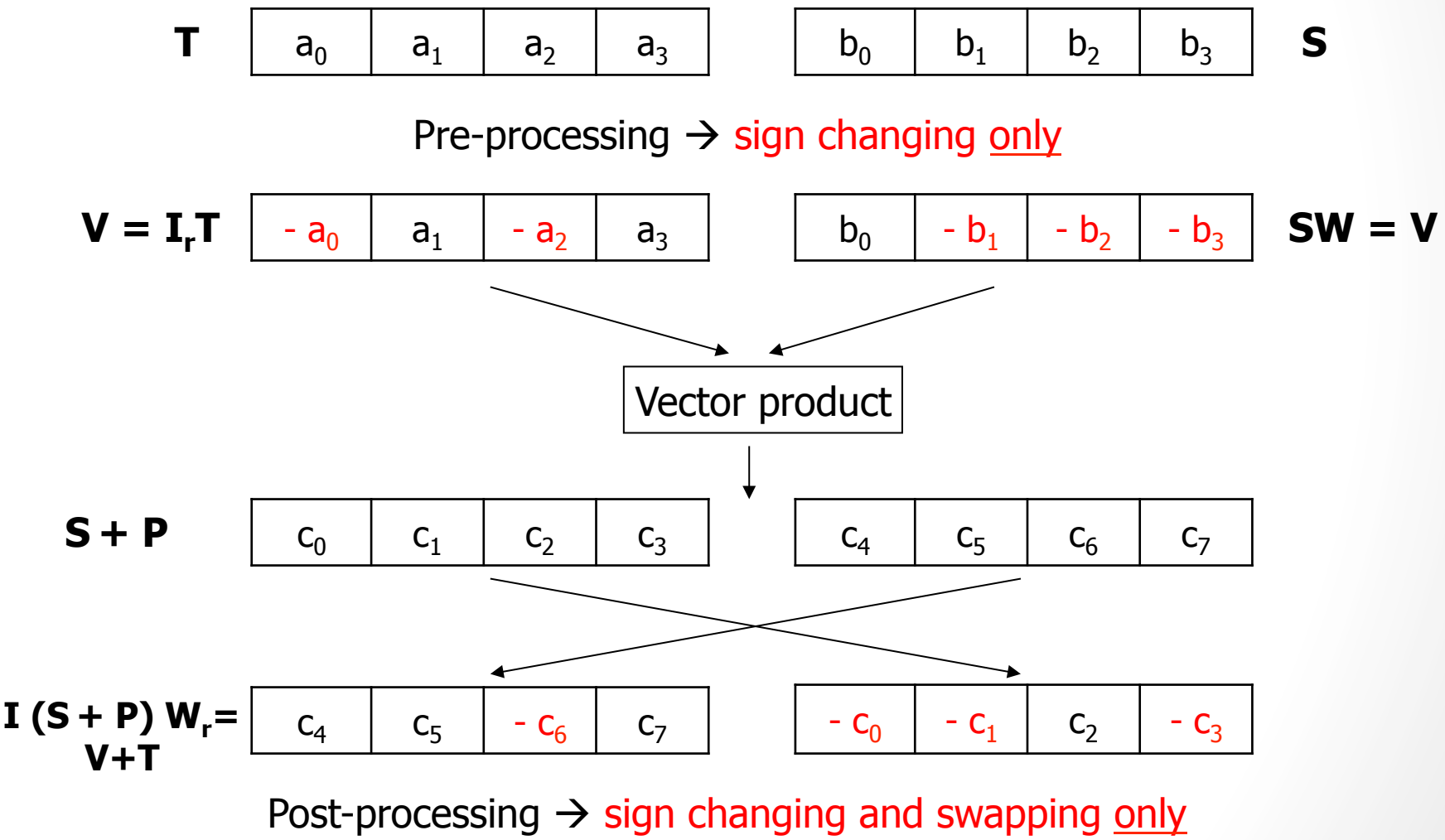
which corresponds to sign changing operations on their coefficients.

I_r , W_r and Z_r are the reverse of I, W and Z, respectively.

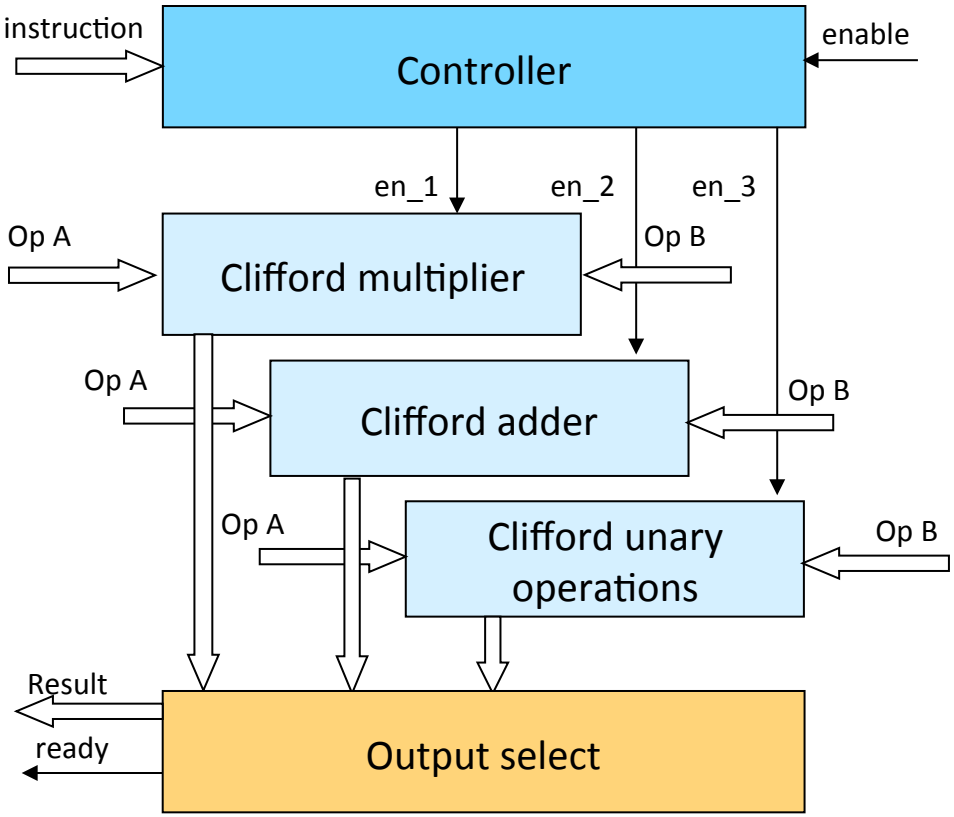


Quadruples – Advantages (5)

Example: T*S



Quad-CliffoSor architecture



- Three functional units for 4D Clifford products, 4D Clifford sums/ differences, and 4D Clifford unary operations
- Parallel architecture
- The multiplier unit uses 16 parallel multipliers and a three-stage pipeline

Reference

S. Franchini, A. Gentile, F. Sorbello, G. Vassallo, and S. Vitabile, **Fixed-size Quadruples for a New, Hardware-Oriented Representation of the 4D Clifford Algebra**, in *Advances in Applied Clifford Algebras*, Vol. 21, No. 2, pp. 315-340, **2011**

Coprocessor family

Coprocessor	GA operations	Clifford number representation	Execution flow	Parallelism techniques	N. of multipliers	Precision	Instruction word length
S-CliffoSor	4D products, sums, diff.	Homogeneous elements	Sequential	-	-	32-bit integers	15x32-bit words = 480 bits
CliffoSor	4D products, sums, diff. 3D rotations (Operations on bivectors not implemented)	Homogeneous elements	Parallel	-	24	32-bit integers	15x32-bit words = 480 bits
Quad-CliffoSor	4D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	16-bit floating point	9x16-bit words = 144 bits
CliffordALU5	**4D/5D products, sums, diff., unary operations**	**Quadruples**	**Parallel**	**Pipeline**	**16**	**32-bit floating point**	**9x32-bit words = 288 bits**
ConformalALU	5D conformal geometric operations (reflections, rotations, translations, dilations)	5D vectors	Parallel	Pipeline	5	32-bit floating point	16x32-bit words = 512 bits

CliffordALU5

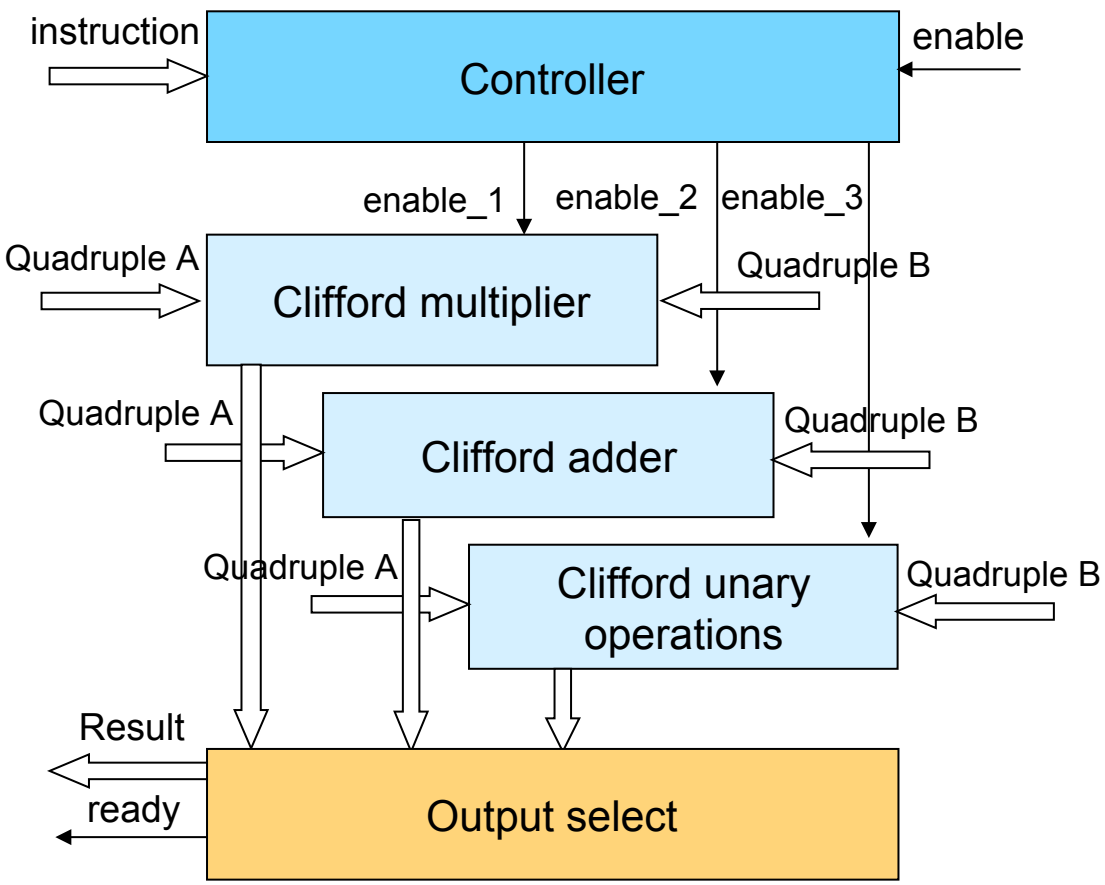
- 5D Clifford operations
- Quadruple-based representation extended to 5D GA elements

5D GA quadruples

Quadruple	Blades			
V	a_1e_1	a_2e_2	a_3e_3	a_4e_4
T	$a_{234}e_2e_3e_4$	$a_{134}e_1e_3e_4$	$a_{124}e_1e_2e_4$	$a_{123}e_1e_2e_3$
S	$a_{14}e_1e_4$	$a_{24}e_2e_4$	$a_{34}e_3e_4$	a_0
P	$a_{23}e_2e_3$	$a_{13}e_1e_3$	$a_{12}e_1e_2$	$a_{1234}e_1e_2e_3e_4$
V’	$a_{15}e_1e_5$	$a_{25}e_2e_5$	$a_{35}e_3e_5$	$a_{45}e_4e_5$
T’	$a_{2345}e_2e_3e_4e_5$	$a_{1345}e_1e_3e_4e_5$	$a_{1245}e_1e_2e_4e_5$	$a_{1235}e_1e_2e_3e_5$
S’	$a_{145}e_1e_4e_5$	$a_{245}e_2e_4e_5$	$a_{345}e_3e_4e_5$	a_5e_5
P’	$a_{235}e_2e_3e_5$	$a_{135}e_1e_3e_5$	$a_{125}e_1e_2e_5$	$a_{12345}e_1e_2e_3e_4e_5$

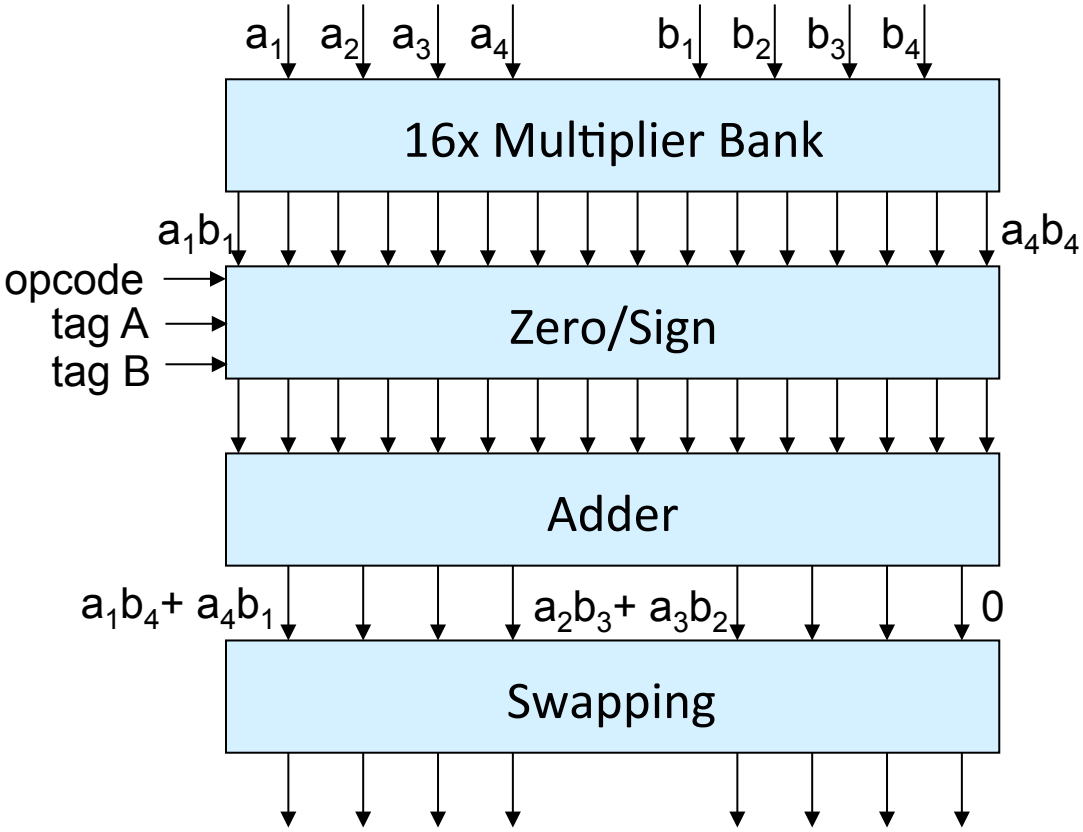
- Simplified algorithm for 4D products on quadruples extended to 5D products

CliffordALU5 architecture



- Universal coprocessor for 4D/5D GA operations
  - Geometric products, outer products, left and right contractions
  - Sums, differences
  - Unary operations (dual, reverse, conjugate, grade involution)
- Three functional units
- Parallel architecture

CliffordALU5 multiplier unit



- Compact architecture of the multiplier unit
- 16 parallel multipliers
- Three-stage pipeline
- 32-bit floating point numbers

Reference

S. Franchini, A. Gentile, F. Sorbello, G. Vassallo, and S. Vitabile, **Design and Implementation of an Embedded Coprocessor with Native Support for 5D, Quadruple-based Clifford Algebra**, in *IEEE Transactions on Computers*, Vol. 62, No. 12, pp. 2366-2381, **2013**

Coprocessor family

Coprocessor	GA operations	Clifford number representation	Execution flow	Parallelism techniques	N. of multipliers	Precision	Instruction word length
S-CliffoSor	4D products, sums, diff.	Homogeneous elements	Sequential	-	-	32-bit integers	15x32-bit words = 480 bits
CliffoSor	4D products, sums, diff. 3D rotations (Operations on bivectors not implemented)	Homogeneous elements	Parallel	-	24	32-bit integers	15x32-bit words = 480 bits
Quad-CliffoSor	4D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	16-bit floating point	9x16-bit words = 144 bits
CliffordALU5	4D/5D products, sums, diff., unary operations	Quadruples	Parallel	Pipeline	16	32-bit floating point	9x32-bit words = 288 bits
ConformalALU	**5D conformal geometric operations (reflections, rotations, translations, dilations)**	**5D vectors**	**Parallel**	**Pipeline**	**5**	**32-bit floating point**	**16x32-bit words = 512 bits**

ConformalALU

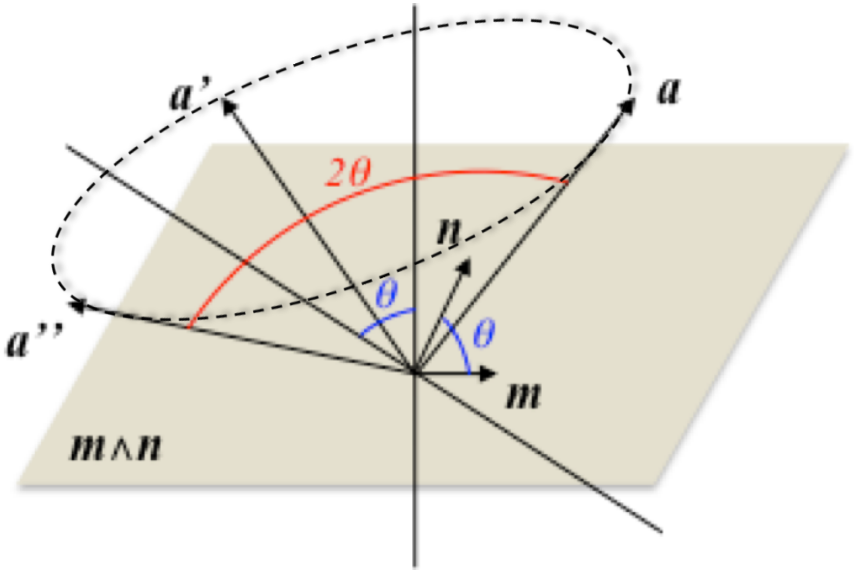
- 5D Conformal Geometric Algebra (CGA) geometric operations (reflections, rotations, translations, dilations)
- Simplified formulation of CGA operations based on two considerations:
  1. A conformal geometric operation on a *k*-blade A_k can be decomposed in operations on vectors:

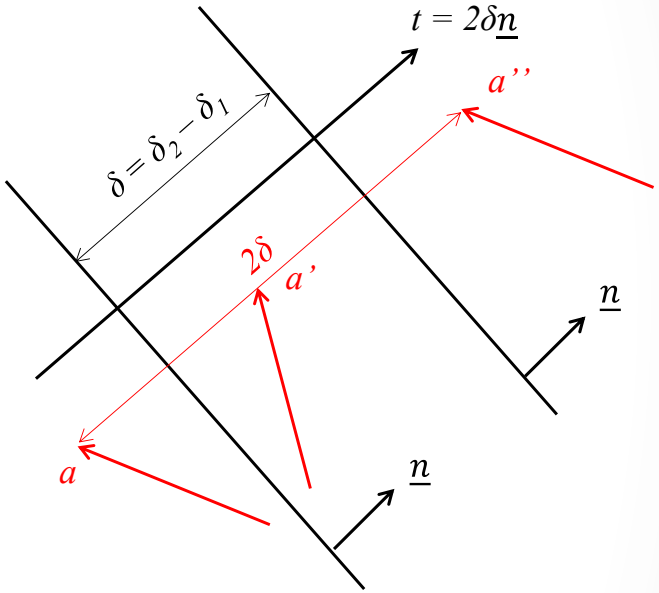
$$XA_k\,\tilde{X} = X(a_1 \wedge a_2 \wedge \ldots \wedge a_k)\,\tilde{X} = Xa_1\,\tilde{X} \wedge Xa_2\,\tilde{X} \wedge \ldots \wedge Xa_k\,\tilde{X}$$

  2. A conformal geometric operation can be obtained by two non-commuting successive reflections

Simplified CGA formulation (1)

- Rotation and translation as two successive reflections





$$a'' = -na'n = -n(-mam)n =$$

$$= (nm)a(mn)$$

$$a'' = Ra\tilde{R}$$

$$a'' = -(\underline{n}+\delta_2 e_\infty)a'(\underline{n}+\delta_2 e_\infty) =$$

$$= -(\underline{n}+\delta_2 e_\infty)[-(\underline{n}+\delta_1 e_\infty)a(\underline{n}+\delta_1 e_\infty)](\underline{n}+\delta_2 e_\infty)$$

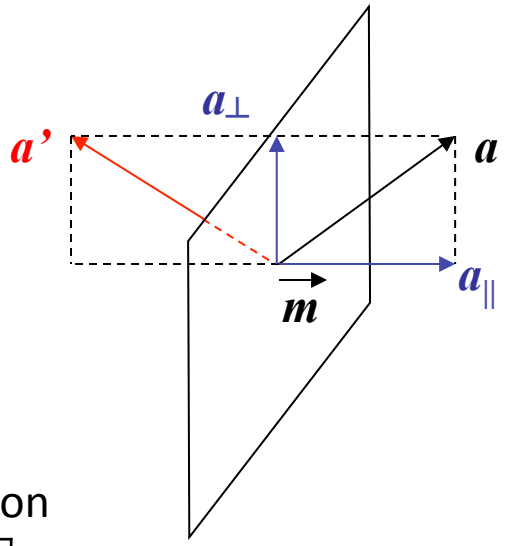
$$a'' = Ta\tilde{T}$$

Simplified CGA formulation (2)

- The basic operation becomes the **reflection of a 5D vector**
- Each vector reflection is obtained by a simplified formula based on the classical dot product, rather than the standard "sandwich" geometric product of CGA

$$a' = a_{\perp} - a_{||} = a - 2a_{||} = a - 2\left|a_{||}\right| m$$

$$a' = a - 2(a \cdot m)m$$



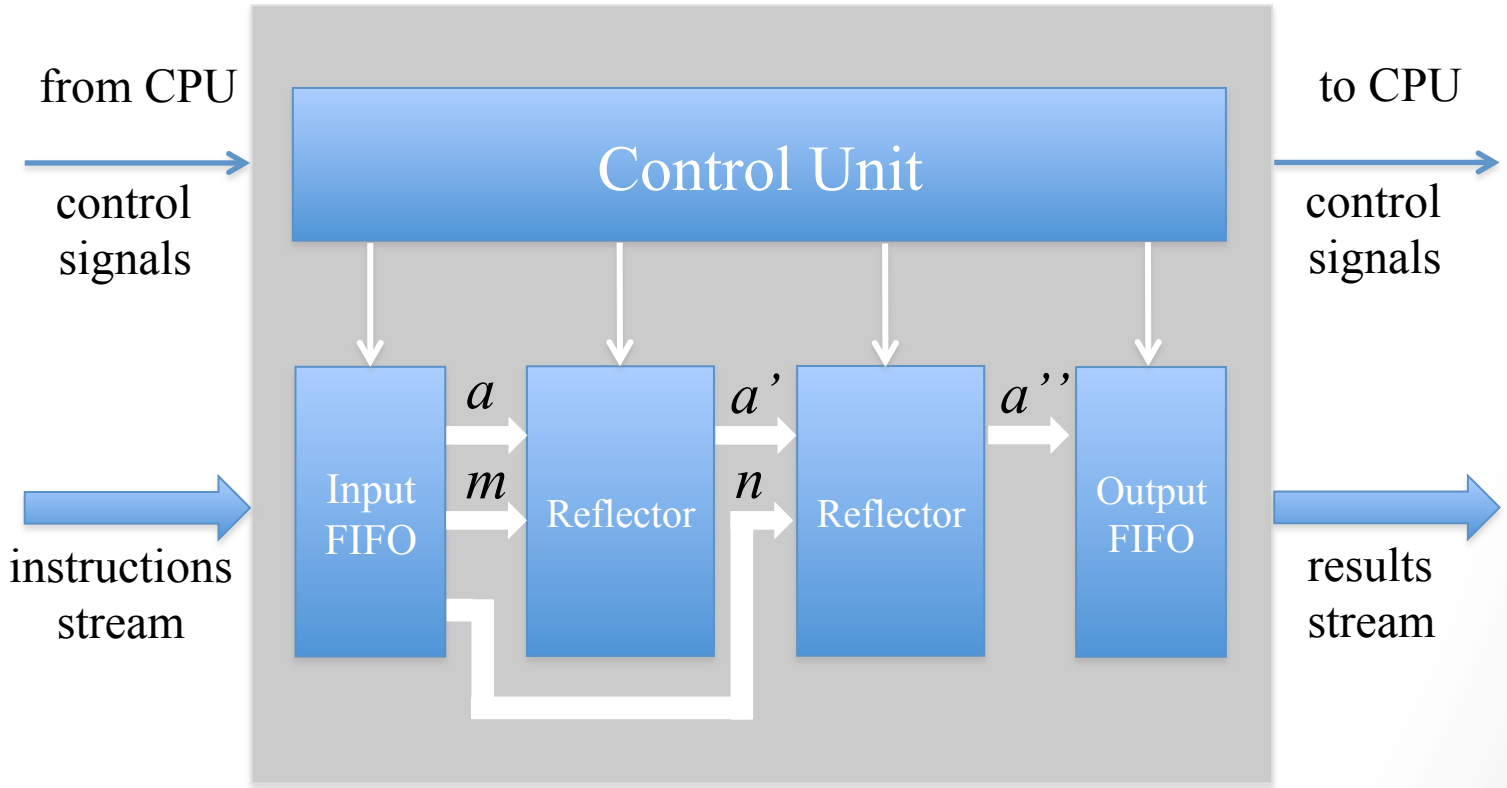
- Computational complexity analysis

N. of basic operations for a CGA transformation

	Operations on vectors
Classical CGA formulation	220 multiplications 189 sums
Our formulation	20 multiplications 8 sums 20 subtractions

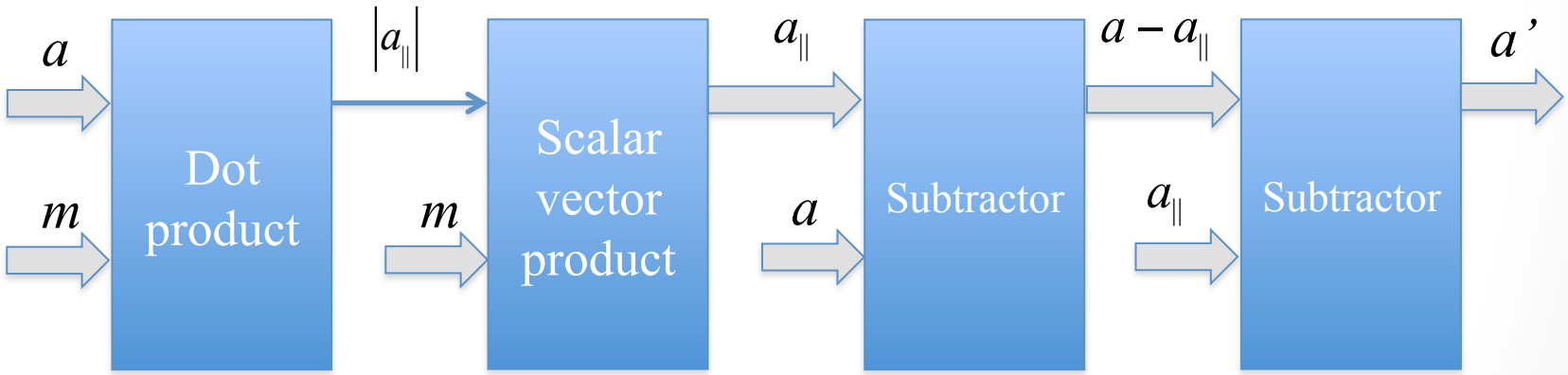
ConformalALU architecture

- The computational simplification leads to a compact and fast hardware architecture
- Two cascade reflector units
- Four-stage pipeline



Reflector unit

- Each reflector unit executes a vector reflection according to the simplified formula that requires one dot product and two subtractions



Reference

S. Franchini, A. Gentile, F. Sorbello, G. Vassallo, and S. Vitabile, **ConformalALU: a Conformal Geometric Algebra Coprocessor for Medical Image Processing**, in *IEEE Transactions on Computers*, Vol. 64, No. 4, pp. 955-970, **2015**

Experimental Results

- FPGA prototypes for all the coprocessors
- CliffordALU5 and ConformalALU implemented as complete embedded Systems-on-Chip (SoC)
- Performance analysis in terms of clock frequency, area cost, relative error, latency per operation, and speedup
- Performance comparison with the baseline software Gaigen

Application suite

- A suite of GA-based algorithms has been chosen from literature as testbench to evaluate the coprocessor effectiveness in specific application domains
- Test applications have been executed using the latest two coprocessors and their performance has been compared with the general-purpose implementation based on Gaigen

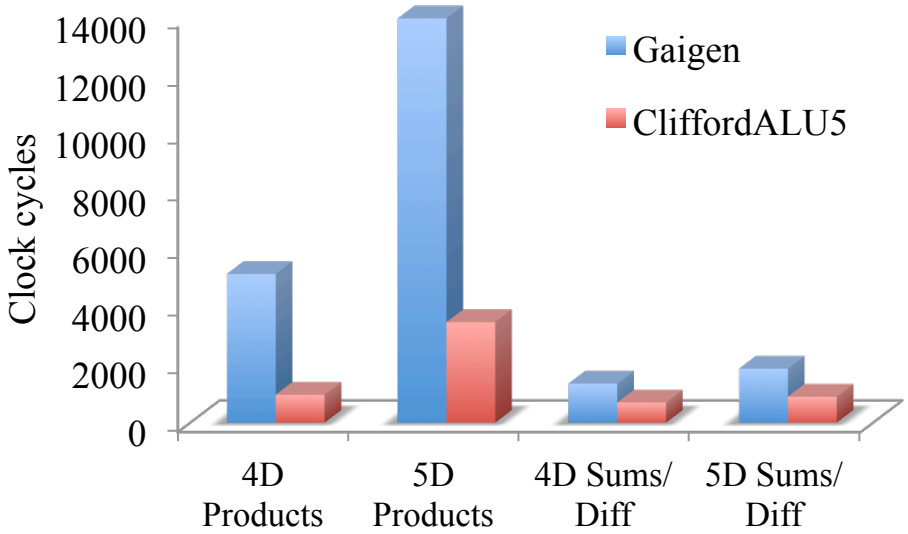
Application	Inverse kinematics	Motion capture	Raytracing	Medical image segmentation	Medical image registration
Observed speedup	3.4x	3.8x	4.8x	46x	43x

- 4D operations (motion capture) 5D operations (inverse kinematics, raytracing, medical imaging)

Performance analysis

Coprocessor	Clock frequency	Area (FPGA slices)	Latency (clock cycles)	Speedup over Gaigen
S-CliffoSor	45 MHz	2,295	Products: 91 Sums/diff.: 78	(potential) Products: 4x Sums/diff.: 3x
CliffoSor	50 MHz	8,444	Products: 7 Sums/diff.: 5	(potential) Products: 4x Sums/diff.: 12x
Quad-CliffoSor	50 MHz	3,201	Products: 3 Sums/diff.: 1	(potential) Products: 23x Sums/diff.: 33x
CliffordALU5	100 MHz	6,011	Products: 3 Sums/diff.: 1	(real) 4D Products: 5x 5D Products: 4x Sums/diff.: 2x
ConformalALU	125 MHz	5,876 (scalar) 9,640 (pipelined)	315 (scalar) 88 (pipelined)	(real) Reflections: 56x Rotations: 15x Translations: 46x Dilations: 41x

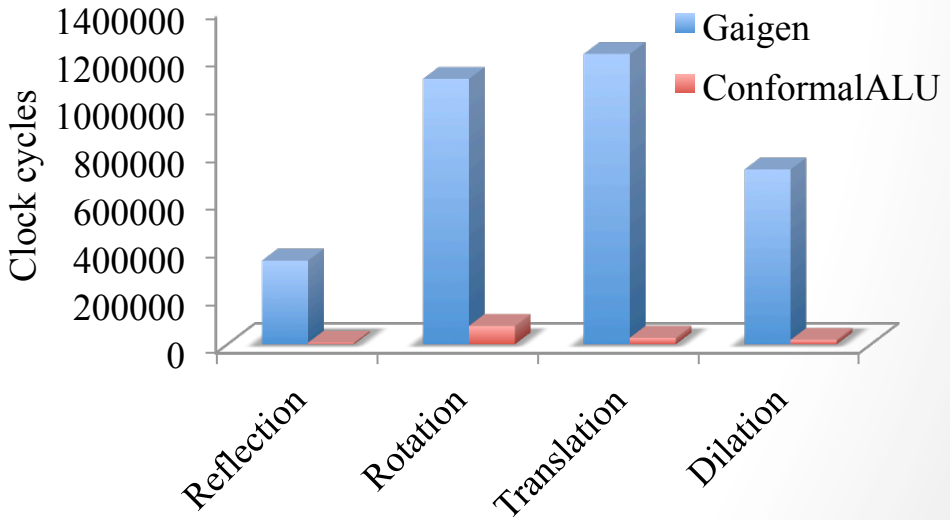
Performance comparison (1)



Gaigen/CliffordALU5 comparison

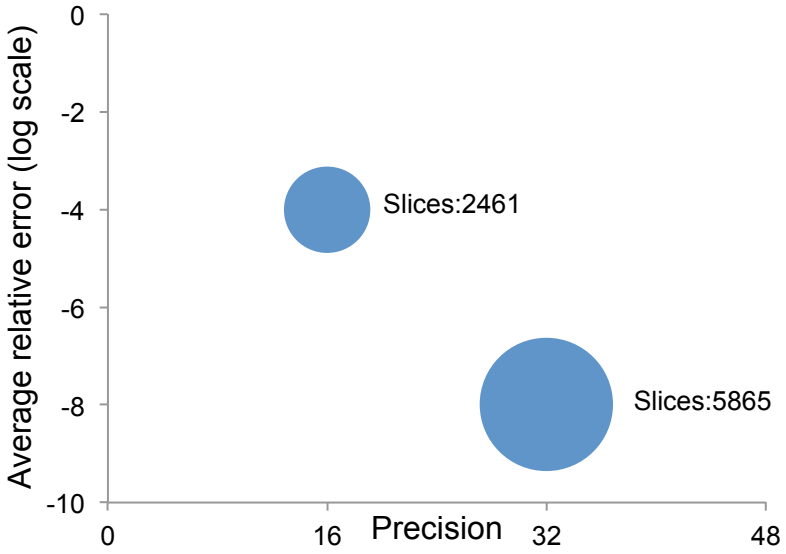
Operation	Speedup
4D Products	5x
5D Products	4x

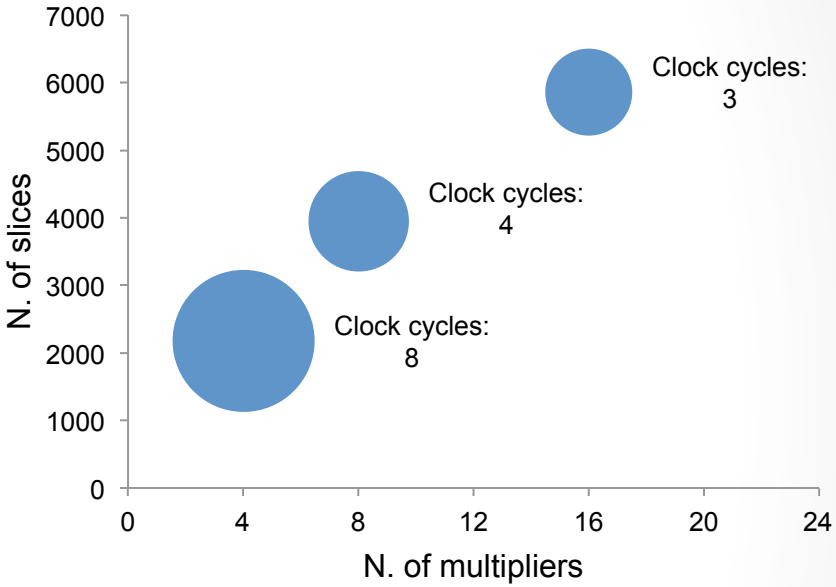
Operation	Speedup
Reflection	56x
Rotation	15x
Translation	46x



Gaigen/ConformalALU comparison

Performance analysis





- Higher-precision architectures: higher area cost, reduced relative errors
- Higher number of multipliers: higher area cost, reduced latency per operation

Performance comparison (2)

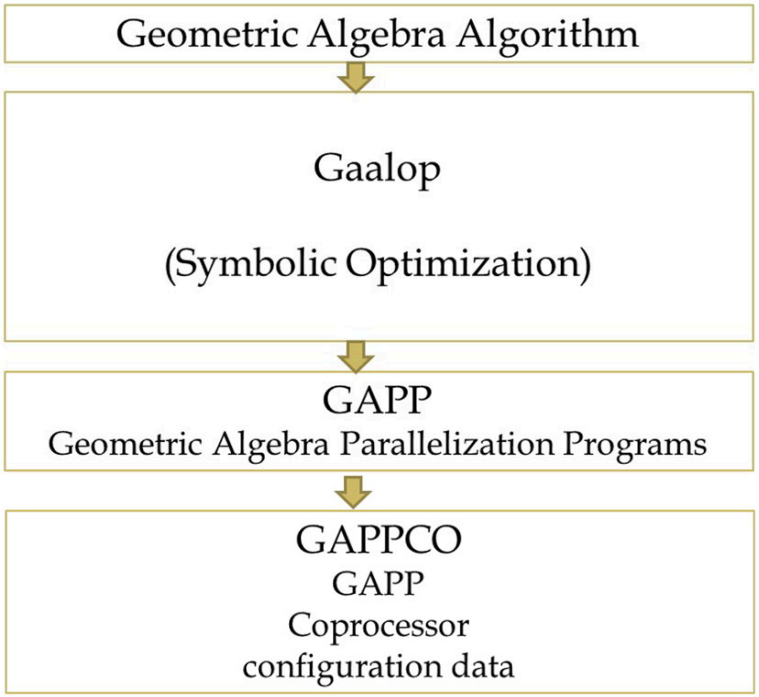
Architecture	Hardware implementation	Parallelism	Target technology	Speedup over conventional CPU
Gaalop (Hildenbrand et al.)	Application-specific	Fine-grained	FPGA/GPU	7x (FPGA) 80x (GPU)
FPGA coprocessor (Perwass et al.)	General-purpose	Coarse-grained	FPGA	1.5x
ASIC coprocessor (Mishra et al.)	Application-specific	Coarse-grained	ASIC	3x
CliffordALU5, 2013	General-purpose	Coarse-grained	FPGA	2x-5x
ConformalALU, 2015	General-purpose	Coarse-grained	FPGA	15x-56x

References

S. Franchini, A. Gentile, F. Sorbello, G. Vassallo, S. Vitabile, **A family of embedded coprocessors with native geometric algebra suppor*t,* David Hestenes Prize Finalist** - 2015 Applied Geometric Algebra in Computer Science and Engineering Conference (**AGACSE 2015**), Barcelona, Spain, **2015**.

S. Franchini, A. Gentile, F. Sorbello, G. Vassallo, S. Vitabile, **Embedded Coprocessors for Native Execution of Geometric Algebra Operations** in *Advances in Applied Clifford Algebras*, Vol. 27, No. 1, pp. 559-580, **2017**.

Current Works. Gaalop + GAPPCO: A Mixed software-hardware approach



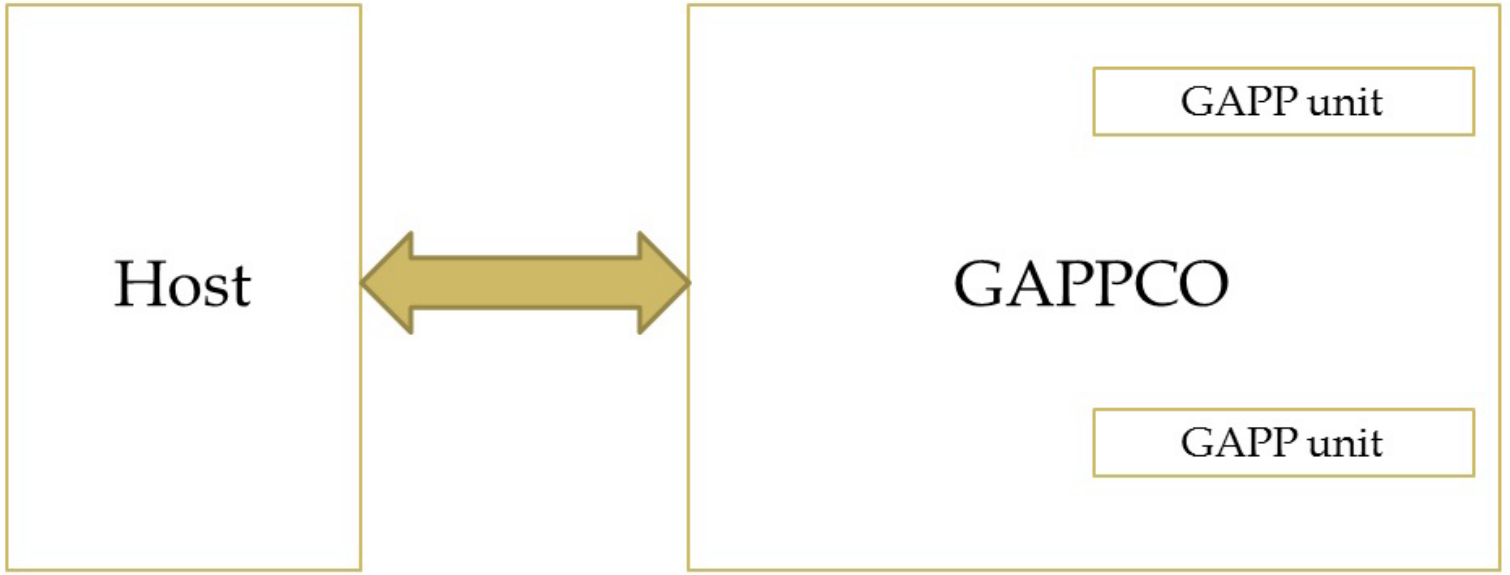
- Gaalop (Geometric Algebra algorithms optimizer) is used to precompile Geometric Algebra algorithms converting them into parallel computations of multivector coefficients each consisting of sums of products to be executed on the GAPPCO coprocessor
- The GAPP compiler generates configuration data for GAPPCO

GAPPCO (GAPP coprocessor)

- Configurable coprocessor based on GAPP (Geometric Algebra Parallelism Programs)
- Direct hardware support of the dot product operations generated after the optimization of the GAPP compiler.
- Configurable coprocessor independent of the algebra dimension that can be configured to accelerate a large set of GA applications.
- GAPPCO consists of a network of configurable DotVectors units each responsible for the computation of one coefficient of one multivector.
- The connections have to be configured before runtime based on the specific configuration data provided by the GAPP compiler.

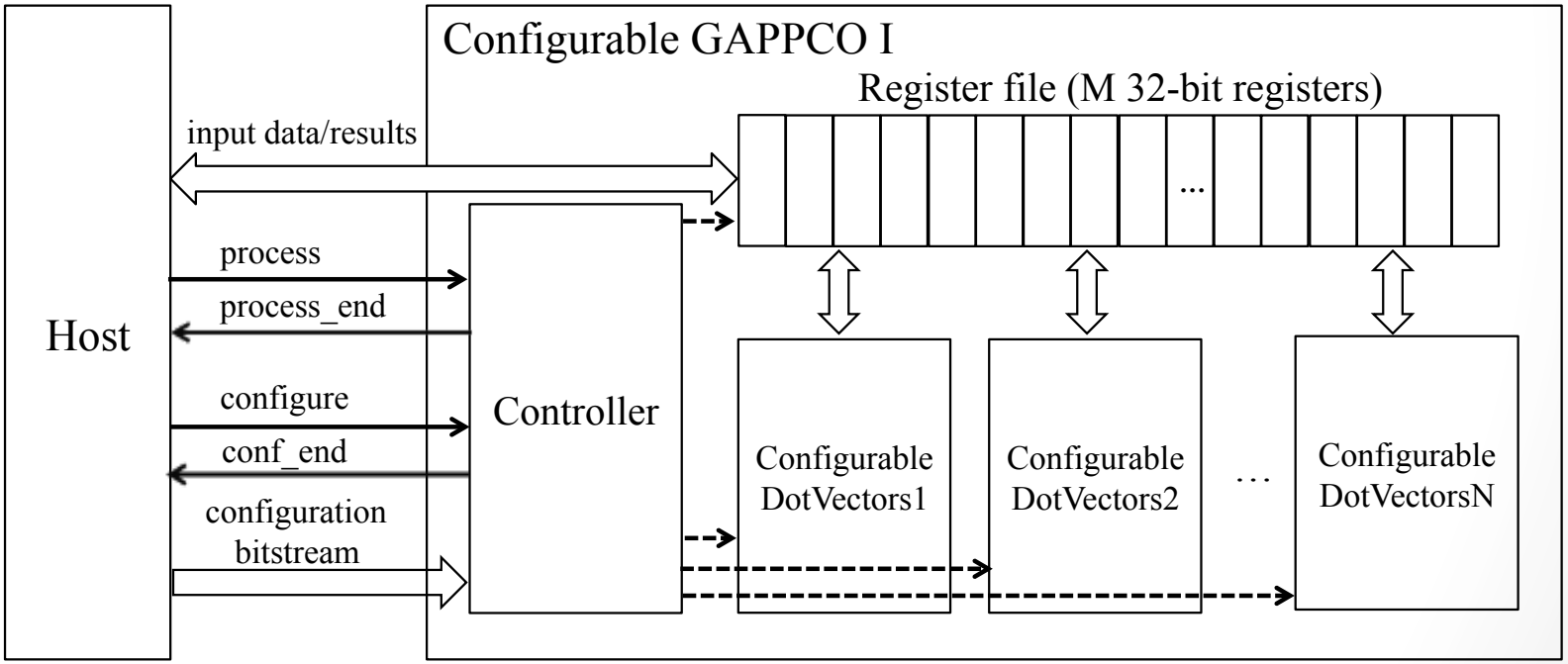
GAPPCO architecture

- GAPPCO can be configured to contain a different number of variable-size parallel processing units (GAPP units). The GAPP compiler on the host side provides the configuration data for GAPPCO.



GAPPCO I concept

- GAPPCO I contains N configurable DotVectors units. Each basic DotVectors unit can calculate the sum of 4 products (4-width DotVectors unit).



Reference

Dietmar Hildenbrand, Silvia Franchini, Antonio Gentile, Giorgio Vassallo, Salvatore Vitabile, **GAPPCO: An Easy to Configure Geometric Algebra Coprocessor Based on GAPP Programs**, in *Advances in Applied Clifford Algebras*, Volume 27, Issue 3, pp. 2115-2132, **2017**.

Conclusions

- Direct hardware support of up to 5D GA operations
- Speedups of about one order of magnitude with respect to the baseline software implementation Gaigen
- Simplified formulation of 5D CGA operations of ConformalALU that allows for a further speedup of about 10x with respect to the execution on CliffordALU5
- An alternative approach: Gaalop+ GAPPCO

Q & A

- Salvatore Vitabile – Department of Biomedicine, Neuroscience and Advanced Diagnostics, University of Palermo, Italy – salvatore.vitabile@unipa.it

- Silvia Franchini - Department of Biomedicine, Neuroscience and Advanced Diagnostics, University of Palermo, Italy silvia.franchini@unipa.it