Approximate FPGA Implementation of CORDIC for Tactile Data Processing using Speculative Adders

M. Franceschi[1†], V. Camus[2†], A. Ibrahim[1], C. Enz[2], M. Valle[1]
[1]Cosmic Lab, DITEN, University of Genova, Italy
[2]Integrated Circuits Laboratory (ICLAB), Swiss Federal Institute of Technology in Lausanne (EPFL), Switzerland
[†]These authors contributed equally to this work:
marta.franceschi@edu.unige.it, vincent.camus@epfl.ch

***Abstract*—In most robotic and biomedical applications, the interest for real-time embedded systems with tactile ability has been growing. For example in prosthetics, a dedicated portable system is needed for developing wearable devices. The main challenges for such systems are low latency, low power consumption and reduced hardware complexity. In order to improve hardware efficiency and reduce power consumption, approximate computing techniques have been assessed. This strategy is suitable for error-tolerant applications involving a large amount of data to be processed, which perfectly fits tactile data processing. This paper presents the first case study of applying Inexact Speculative Adders (ISA) to the FPGA implementation of a Coordinate Rotation Digital Computer (CORDIC) module within the Machine Learning algorithm of a tactile data processing system. The design has been synthesized and implemented on a Xilinx ZYNQ-7000 ZC702 device. Preliminary results have shown dynamic power reduction up to 40 % and delay latency reduction up to 21 % compared to a conventional CORDIC module, at the cost of a negligible average relative error of 0.049 % for sine and 0.003 % for cosine computations.**

***Keywords*—Approximate computing, inexact speculative adder, tactile data processing, FPGA, CORDIC.**

I. Introduction

Restoring the sense of touch is one of the biggest challenges in prosthetics. Commercial prostheses can be successfully controlled using the electrical activity of muscles, but they cannot provide sensing information when touching or being touched: the somatosensory feedback from the prosthesis to the patient is still missing [1, 2]. Closing this loop in the prosthesis control would allow better control of grasping and manipulation. In addition, it is hypothesized that this could enhance the utility as well as improve the embodiment of such artificial system as this tactile information could stimulate the psychological and cognitive mechanisms related to body ownership [3].

In [4], an interface prototype integrating both distributed sensing and stimulation has been presented as a new concept towards providing prosthetic devices with tactile sensing capabilities. This prototype system was comprised of an electronic skin (e-skin) made of an array of piezoelectric polymer sensors with an interface electronics and a stimulation system. This global system, which was validated on Matlab and running in real time on PC, was experimentally tested, evaluating the capacity of the brain to correctly interpret the elicited tactile sensations. In this case, low level information has been extracted by only applying basic processing (i.e. spatial fusion and time integration) to raw tactile data. Those encouraging results represent a breakthrough towards the development of an embedded and real-time system for tactile data processing as required by prosthetics.

Nevertheless, the current challenge of such system relies on the efficient implementation and power requirements of the tactile information processing. Machine Learning (ML) paradigms [5]–[10] have been used to retrieve information about object contacts as they are powerful methods for tackling clustering, regression or classification problems. Therefore, real-time implementation of tactile data processing algorithms for extracting high-level information (e.g. shapes, textures) has been taken into account. A ML approach based on tensorial kernel has been chosen as it has proven effectiveness in tactile data processing [11] and as it can preserve the inherent tensorial structure of the signals produced by the sensing array. However, the creation of a real-time embedded data processing unit for e-skin is yet far from achievement. This ML approach requires the use of Singular Value Decomposition (SVD) algorithm, which is a computationally intensive [12] process.

Many techniques and methods from circuit [13]–[16] to system level [17]–[20] have recently emerged to lower hardware complexity or energy consumption of embedded electronic systems. A multitude of applications are intrinsically resilient to approximations and errors. Hence, inexact circuits and approximate computing have risen as one of the most promising techniques to improve power efficiency of those systems as they involve cognitive perceptions of humans, who can easily tolerate indiscernibly variations about touching or being touched.

This work aims at implementing approximate circuit techniques in the FPGA implementation of real-time tactile data processing for e-skin application. It focuses on the implementation of the Coordinate Rotation Digital Computer (CORDIC) [21] algorithm, as it is used for several computing tasks such as SVD, the most computationally expensive algorithm for ML approaches that has ever been considered [11]. This first attempt of approximate CORDIC implementation on FPGA uses Inexact Speculative Adder (ISA) architectures [22], a circuit-level technique optimized for high-speed arithmetic computations. The paper is organized as follows: Sections II and III explain the ISA architecture and CORDIC algorithm, section IV describes the implementation and general architecture of the approximate CORDIC and section V shows the results for a selection of ISA configurations in the approximate CORDIC.

II. Inexact Speculative Adder

Additions are the most frequent arithmetic units used in digital systems. Hence, many have tried to improve their speed or power efficiency. For this purpose, some approximate adders have been built using the concept of carry speculation [23]. This is feasible as carry chain propagation typically does not cover the entire length of the adder, allowing to guess relatively accurately an internal carry based on a small number

of preceding stages. As a result, the carry propagation chain, critical path of the circuit, can be sliced in multiple shorter paths executed in parallel, loosing up delay constraints over the whole circuit and enabling performance beyond theoretical bounds of exact adders.

Among numerous speculative adders [24], the Inexact Speculative Adder (ISA) [22] is a general and optimum architecture of speculative addition to improve speed, power efficiency and accuracy management thanks to a short speculative path and to an adaptable double-direction error compensation mechanism. This technique allows to precisely control mean and maximum errors. It has also shown significant benefits compared or combined with other low-power techniques [25]–[27] or successfully integrated within bigger ASIC systems [28]. In the case of FPGA, the ISA could be particularly interesting in order to overcome FPGA's hardware limitations, e.g. fixed number of Look-Up Tables (LUT) and interconnect constraints.

The general block schematic of the ISA is presented on Fig. 1. It slices the carry chain in several speculative sub-paths executed in parallel, each of them consisting of a carry speculation block (SPEC), an addition block (ADD) and an error compensation block (COMP) that overlaps on two ADD blocks. For each of these paths, the functionalities of the blocks are the following:

- The SPEC block produces a speculated internal carry from a very short number of input bits. This is generally done with a carry look-ahead unit. If a carry propagation spans the entire SPEC block, it cannot predict exactly the carry and a wrong guess could lead to a speculative error. Since long propagation sequences are uncommon, the rate of erroneous speculations decreases with larger the SPEC block size.
- The ADD block computes a local sum from the carry speculated in the SPEC block.
- Without compensation, incorrect carry speculations could cause disastrous errors. The COMP block detects those incorrect speculations and compensates erroneous sum either by trying to correct a fixed number of bits in the current sum, or by balancing some bits in the preceding sum to limit its relative arithmetic value.

The achieved adder arithmetic is shown on Fig. 2. Errors only occur in the speculative paths on the right. The COMP is triggered when the speculated carry differs from the carry-out of the prior sub-adder. The COMP's correction technique implements an incrementer or decrementer on a fixed group of LSBs of the current ADD block that fully corrects a missed

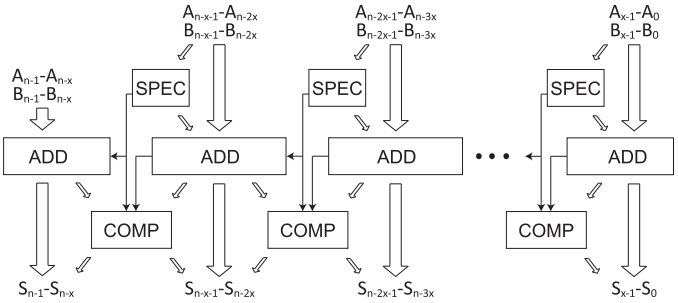

Fig. 1: Block schematic of the Inexact Speculative Adder (ISA) [22]. Every speculative path comprises a carry speculation block (SPEC), an adder block (ADD) and a double-direction error compensation block (COMP).

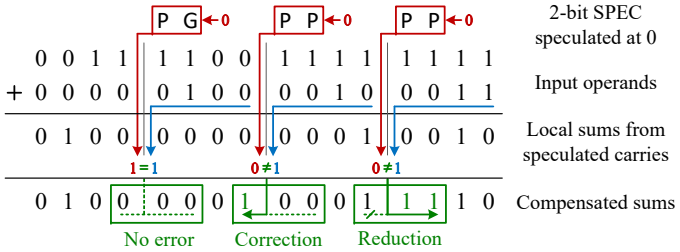

Fig. 2: Example of arithmetic computation in an ISA with 4-bit ADD, 2-bit SPEC, 1-bit COMP correction and 2-bit COMP reduction.

carry. This technique fully resolves most speculative errors, as in the central path of Fig. 2 for which the sum's LSB of the has been corrected. In the cases where the stages above correction bits are all in propagation modes, the sum bits cannot be corrected as it would cause an internal overflow. Thus, the COMP's reduction flips the MSBs of the preceding sum in order to reduce the arithmetic error as in the right path of Fig 2.

III. CORDIC Algorithm

The CORDIC [21] is an iterative and particularly well parallelizable algorithm extensively used in digital signal processing. It only contains iterative Shift-Add operations to calculate a variety of functions, such as logarithmic, trigonometric and hyperbolic functions. It can be operated in vectoring mode or in rotation mode. The first produces a rotation of the input vector to the x axis while recording the angle needed for that rotation. The second, called *rotation* by Deprettere *et. al* [29], makes a rotation of the input vector by a specified angle. Despite the CORDIC can be operated in both modes, only the latter has been considered in this work.

The CORDIC rotation-mode algorithm starts by initializing the angle accumulator z with the requested rotation angle z_0. Then, depending of the sign of the angle after every iteration, a decision d_i is taken in order to decrease the angle accumulator magnitude. The equations in rotation mode are:

$$\begin{aligned} z_{i+1} &= z_i - d_i \arctan(2^{-i}) \\ x_{i+1} &= x_i - 2^{-i}\, d_i\, y_i \\ y_{i+1} &= y_i - 2^{-i}\, d_i\, x_i \end{aligned} \tag{1}$$

where (i) $i = \{0, N-1\}$, (ii) N the number of iterations, and (iii) $d_i = -1$ if $z_i < 0$ and $+1$ otherwise;
which implies that:

$$\begin{aligned} x_n &= A_n \left(x_0 \cos z_0 - y_0 \sin z_0\right) \\ y_n &= A_n \left(x_0 \sin z_0 + y_0 \cos z_0\right) \\ z_n &= 0 \\ A_n &= \prod \sqrt{1 + 2^{-2i}} \end{aligned} \tag{2}$$

where A_n is a gain depending on the number of iterations.

IV. Hardware Implementation

The CORDIC algorithm architecture uses a single Shift-Add operation for each component: x, y, and z. Each unit consists of a MUX (2:1 multiplexer), a shift register and an adder-subtractor. At the beginning of each CORDIC computation, x_0, y_0 and z_0 values are given as inputs to the MUX. Then the computation proceeds using the values stored in $Xreg$, $Yreg$ and $Zreg$, respectively. In the ROM, the micro-rotation angles $\arctan(2^{-i})$ are stored. The CORDIC algorithm is an

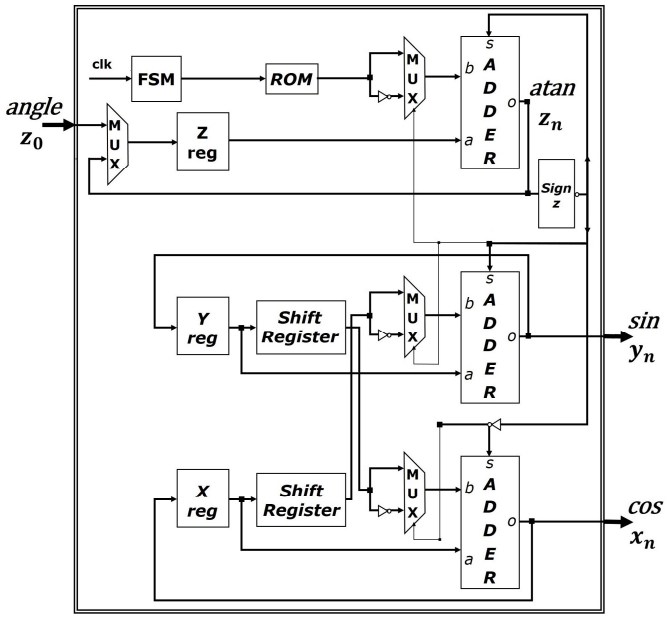

Fig. 3: Architecture of the CORDIC in rotation mode.

iterative process varying according to the ROM input i. In the considered case, the assigned values of the variable i are from 0 to 29. To control the ROM addresses, the FSM tracks the shifting distance and enables the multiplexer signals.

In order to apply ISA within the CORDIC architecture, modifications have been needed. The new architecture is illustrated in Fig. 3. As ISA normally works with unsigned numbers, the adder-subtractors for x, y and z components have been substituted with conventional adders. Adding a MUX that considers the positive or the negative signal depending on a control signal makes possible to construct a circuit performing both addition and subtraction at the same time. This expedient has been used to have the right working principles of the CORDIC algorithm as both addition and subtraction operations can occur depending on the sign of $Zreg$.

Since only the rotation mode of the CORDIC has been taken into account, only the MUX selecting the angle accumulator z or the initialization input value z_0 (the desired angle of rotation) has been implemented. Fig. 3 shows the implementation example underling the independence of x_0 and y_0 on the initial CORDIC operations.

According to (1) and (2), the results of the rotation-mode CORDIC computation are the cosine x_n and sine y_n of the input angle z_0. In particular, by setting:

$$y_0 = 0 \tag{3}$$

the equations in (2) are reduced to:

$$\begin{aligned} x_n &= A_n \cos z_0 \\ y_n &= A_n \sin z_0 \end{aligned} \tag{4}$$

and by setting:

$$x_0 = \frac{1}{A_n} \tag{5}$$

where $A_n = 0.6073$, the rotation produces the unscaled cosine and sine of z_0.

V. Experimental Results

The proposed CORDIC architecture has been modeled in VHDL and simulated using Xilinx Vivado (Fig. 3). It has then been synthesized and implemented on a Xilinx ZYNQ-7000 ZC702 device.

This work aims at reducing resource utilization (e.g. hardware complexity, power consumption and latency) and analyzing computations accuracy of CORDIC and approximate CORDIC architectures. Although it would be advantageous to optimize the bigger number of speculative paths, this study solely considers limited cases of speculative paths with regular speculative structures (i.e. identical speculative paths). As first validation of the use of speculative arithmetic within an FPGA platform, over a hundred ISA architectures have been considered and eight of them have been selected. Table I lists the different ISA configurations of the approximate CORDIC implementations. The parameters choice is based on the considered CORDIC data width of 32 bit.

TABLE I: ISA configurations of the approximate CORDIC implementations

#	ISA configuration details
1	2 paths, ADD = 16, SPEC = 0, COR = 6, RED = 0
2	2 paths, ADD = 16, SPEC = 1, COR = 2, RED = 4
3	2 paths, ADD = 16, SPEC = 2, COR = 1, RED = 3
4	2 paths, ADD = 16, SPEC = 3, COR = 0, RED = 2
5	4 paths, ADD = 8, SPEC = 0, COR = 1, RED = 2
6	4 paths, ADD = 8, SPEC = 1, COR = 3, RED = 0
7	4 paths, ADD = 8, SPEC = 2, COR = 5, RED = 0
8	4 paths, ADD = 8, SPEC = 3, COR = 0, RED = 0

In order to quantify the computation accuracy of the approximate designs, two metrics have been considered as in [22]. Both are built with the relative error (*RE*), defined as:

$$RE = \left| \frac{v_{approx} - v_{correct}}{v_{correct}} \right| \tag{6}$$

where v_{approx} and $v_{correct}$ are the approximate and correct values of CORDIC computation, respectively. The two metrics used are the Root Mean Square (RMS) of the relative error (RE_{RMS}), which is a well-known accuracy estimator, and the maximum relative error (RE_{MAX}), that defines the worst-case accuracy.

Fig. 4 shows the error characteristics and normalized costs of each approximate CORDIC implementation for sine and cosine outputs. Hardware costs, normalized to the conventional FPGA design, are expressed in terms of dynamic power and Power-Delay-Area Product (PDAP). Despite different RE_{RMS} and RE_{MAX} values between sine and cosine computations, the error characteristics follow the same trends. The persistent gap between cosine and sine errors is expected to come from the stimuli angles, ranging from 0 to 45° as required by the tactile data processing application.

Approximate circuits in the CORDIC design allow a dynamic power consumption saving of up to 40 % and a general PDAP improvement of 58 % at the cost of low errors with RE_{RMS} of 0.049 % for sine and of 0.003 % for cosine computations. Though, some designs as #5 and 6 display both bad accuracy and hardware characteristics, showing that small 8-bit ADD combined with low SPEC size do not provide a good enough addition for the convergence of the CORDIC.

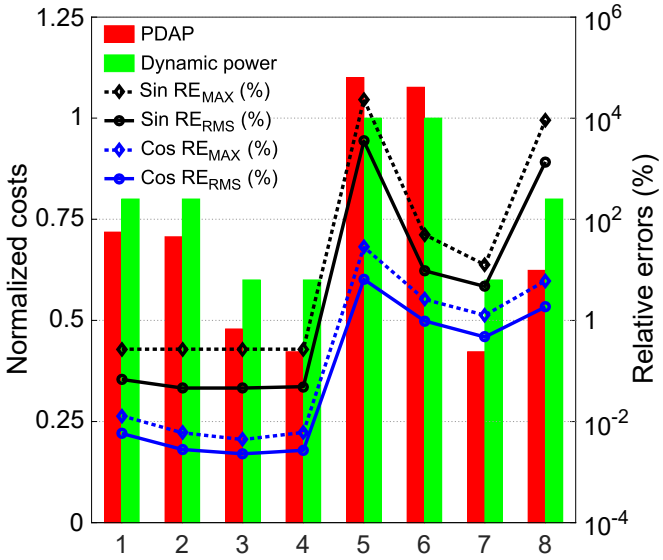

Fig. 4: Normalized costs and relative errors for both sine and cosine computations of the approximate CORDIC implementations.

Configurations with 16-bit ADD blocks have lower costs than 8-bit ones. This contradicts the intuitive ASIC results of [22], it is due to the FPGA's fixed LUT architecture. In effect, some ISA architectures might fit better the LUT configurations and interconnections to minimize their required number or delay.

VI. Conclusion

This paper has proposed a first attempt of approximate Coordinate Rotation Digital Computer (CORDIC) implementation on FPGA using Inexact Speculative Adders (ISA). The use of speculative arithmetic has allowed high performance and efficiency improvements of the CORDIC module, with up to 40 % power consumption reduction and up to 21 % delay reduction, offering overall cost reduction of up to 58 %. The approximate CORDIC has been characterized by its relative arithmetic error, showing negligible average and maximal errors, i.e. RMS relative errors being only 0.003 % for cosine computations and 0.049 % for sine computations.

This first FPGA implementation of approximate CORDIC, sub-task of the computationally expensive SVD required in tactile data processing, represents a successful preliminary investigation of approximate computing for real-time embedded prosthetics. Future work will address the improvement of larger Machine Learning algorithms required for tactile data processing combined with advanced use of approximate arithmetic circuits such as inexact speculative multipliers.

References

[1] N. Jiang, S. Dosen, K. R. Muller, and D. Farina, "Myoelectric control of artificial limbs:is there a need to change focus? [in the spotlight]," *IEEE Signal Processing Magazine*, 2012.

[2] C. Antfolk, M. DAlonzo, B. Rosn, G. Lundborg, F. Sebelius, and C. Cipriani, "Sensory feedback in upper limb prosthetics," *Expert Review of Medical Devices*, 2013.

[3] M. D'Alonzo, F. Clemente, and C. Cipriani, "Vibrotactile stimulation promotes embodiment of an alien hand in amputees with phantom sensations," in *IEEE Trans. Neural Syst. Rehabil. Eng.*, 2015.

[4] M. Franceschi, L. Seminara, S. Dosen, M. Strbac, M. Valle, and D. Farina, "A system for electrotactile feedback using electronic skin and flexible matrix electrodes: Experimental evaluation," *Trans. on Haptics*, 2016.

[5] S. A. Arabshahi and Z. Jiang, "Development of a tactile sensor for braille pattern recognition: sensor design and simulation," *Smart Materials and Structures*, 2005.

[6] D. Silvera Tawil, D. Rye, and M. Velonaki, "Interpretation of the modality of touch on an artificial arm covered with an EIT-based sensitive skin," *International Journal of Robotics Research (IJRR)*, 2012.

[7] S. Decherchi, P. Gastaldo, R. S. Dahiya, M. Valle, and R. Zunino, "Tactile-data classification of contact materials using computational intelligence," *IEEE Trans. on Robotics*, 2011.

[8] D. Goger, N. Gorges, and H. Worn, "Tactile sensing for an anthropomorphic robotic hand: Hardware and signal processing," in *Robotics and Automation (ICRA), IEEE Conference*, 2009.

[9] S.-H. Kim, J. Engel, C. Liu, and D. L. Jones, "Texture classification using a polymer-based mems tactile sensor," *Journal of Micromechanics and Microengineering*, 2005.

[10] H. Iwata and S. Sugano, "Human-robot-contact-state identification based on tactile recognition," *IEEE Trans. on Industrial Electronics*, 2005.

[11] A. Ibrahim, P. Gastaldo, H. Chible, and M. Valle, "Real-time digital signal processing based on FPGAs for electronic skin implementation," *Sensors*, 2017.

[12] A. Ibrahim, M. Valle, L. Noli, and H. Chible, "FPGA implementation of fixed point CORDIC-SVD for e-skin systems," in *Ph.D. Research in Microelectronics and Electronics (PRIME), 11th Conference*, 2015.

[13] X. Jiao, Y. Jiang, A. Rahimi, and R. K. Gupta, "SLoT: A supervised learning model to predict dynamic timing errors of functional units," in *Design, Automation & Test in Europe (DATE), IEEE*, 2017.

[14] V. Camus, J. Schlachter, and C. Enz, "A low-power carry cut-back approximate adder with fixed-point implementation and floating-point precision," in *Design Automation Conference (DAC), 53rd ACM/EDAC/IEEE*, 2016.

[15] A. Bonetti, A. Teman, P. Flatresse, and A. Burg, "Multipliers-driven perturbation of coefficients for low-power operation in reconfigurable FIR filters," *IEEE Trans. on Circuits and Systems I (TCAS-I)*, 2017.

[16] J. Schlachter, V. Camus, K. V. Palem, and C. Enz, "Design and applications of approximate circuits by gate-level pruning," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2017.

[17] S. S. Basu, P. G. del Valle, G. Karakonstantis, G. Ansaloni, L. Pozzi, and D. Atienza Alonso, "Inexact-aware architecture design for ultra-low power bio-signal analysis," *IET Computers & Digital Techniques*, 2016.

[18] B. Barrois, K. Parashar, and O. Sentieys, "Leveraging power spectral density for scalable system-level accuracy evaluation," in *Design, Automation & Test in Europe (DATE), IEEE Conference*, 2016.

[19] A. Mercat, J. Bonnot, M. Pelcat, W. Hamidouche, and D. Menard, "Exploiting computation skip to reduce energy consumption by approximate computing, an hevc encoder case study," in *Design, Automation & Test in Europe (DATE), IEEE Conference*, 2017.

[20] O. L. I. Galindez, K. Badami, V. R. Pamula, S. Lauwereins, W. Meert, and M. Verhelst, "Exploiting system configurability towards dynamic accuracy-power trade-offs in sensor front-ends," in *Asilomar Conference on Signals, Systems & Computers (ACSSC), IEEE*, 2016.

[21] R. Andraka, "A survey of CORDIC algorithms for FPGA based computers," in *Field Programmable Gate Arrays (FPGA), ACM/SIGDA Sixth International Symposium*, 1998.

[22] V. Camus, J. Schlachter, and C. Enz, "Energy-Efficient Inexact Speculative Adder with High Performance and Accuracy Control," in *Circuits and Systems (ISCAS), IEEE International Symposium*, 2015.

[23] T. Liu and S.-L. Lu, "Performance Improvement with Circuit-level Speculation," in *Microarchitecture (MICRO-33), IEEE/ACM*, 2000.

[24] H. Jiang, C. Liu, L. Liu, F. Lombardi, and J. Han, "A review, classification and comparative evaluation of approximate arithmetic circuits," in *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 2017.

[25] V. Camus, J. Schlachter, and C. Enz, "Energy-efficient digital design through inexact and approximate arithmetic circuits," in *New Circuits and Systems Conference (NEWCAS), IEEE*, 2015, pp. 1–4.

[26] J. Schlachter, V. Camus, and C. Enz, "Near/sub-threshold circuits and approximate computing: The perfect combination for ultra-low-power systems," in *VLSI (ISVLSI), IEEE Symposium*, 2015.

[27] X. Jiao, V. Camus, M. Cacciotti, Y. Jiang, C. Enz, and R. Gupta, "Combining structural and timing errors in overclocked inexact speculative adders," in *Design, Automation & Test in Europe (DATE), IEEE*, 2017.

[28] V. Camus, J. Schlachter, C. Enz, M. Gautschi, and F. K. Gurkaynak, "Approximate 32-bit floating-point unit design with 53% power-area product reduction," in *European Solid-State Circuits (ESSCIRC), IEEE Conference*, 2016.

[29] E. Deprettere, P. Dewilde, and R. Udo, "Pipelined CORDIC architectures for fast VLSI filtering and array processing," in *Acoustics, Speech, and Signal Processing (ICASSP), IEEE Conference*, 1984.