Efficient Error Detection Architectures for CORDIC through Recomputing with Encoded Operands

Mehran Mozaffari Kermani
Department of Electrical
and Microelectronic Engineering
Rochester Institute of Technology
Rochester, NY 14623
Email: m.mozaffari@rit.edu

Rajkumar Ramadoss
Department of Electrical
and Microelectronic Engineering
Rochester Institute of Technology
Rochester, NY 14623
Email: rxr1052@rit.edu

Reza Azarderakhsh
Department of Computer Engineering
Rochester Institute of Technology
Rochester, NY 14623
Email: rxaeec@rit.edu

***Abstract*—Various optimized coordinate rotation digital computer (CORDIC) designs have been proposed to date. Nonetheless, in the presence of natural faults, such architectures could lead to erroneous outputs. In this paper, we propose error detection schemes for CORDIC architectures used vastly in applications such as complex number multiplication, and singular value decomposition for signal and image processing. To the best of our knowledge, this work is the first in providing reliable architectures for these variants of CORDIC. We present three variants of recomputing with encoded operands to detect both transient and permanent faults. The overheads and effectiveness of the proposed designs are benchmarked through Xilinx FPGA implementations and error simulations. The proposed approaches can be tailored based on overhead tolerance and the reliability constraints to achieve.**

***Index Terms*—Coordinate rotation digital computer, recomputing with encoded operands, reliability.**

I. Introduction

Coordinate rotation digital computer (CORDIC) algorithm has been used in various domains including computation of trigonometric functions [1], multiplication, and division [2]. Low latencies could be achieved for this algorithm; yet, it is mainly attractive because of its low-complexity hardware implementations [3]. This is much preferred in deeply-embedded systems (embedded deeply into human body and objects) which are often battery-powered, e.g., pacemakers for which low-area/power/energy computations are needed.

In [4], [5], [6], [7], [8], CORDIC architectures with angle recording (AR) schemes are presented which reduce the number of iterations for the computations of complex multiplications. This is accomplished by coding the angle of rotation as a linear combination of a set of elementary angle of fine-grained (micro) rotations. Rotation of vectors through known and fixed-angles exhibits wide applications in the field of signal processing, robotics, animations, and graphics [9], [10], and [11]. In the state-of-the-art literature, various CORDIC architectures have been proposed for the rotation of vectors through fixed-angles. In [12], it is proposed to perform an exhaustive search to achieve an optimal elementary-angle-set (EAS) of micro-rotations for the given known angle(s).

Digital circuits are prone to natural errors caused due to alpha particles from cosmic rays creating energetic neutrons, thermal neutrons, and the like, whose detection has been the center of previous works, e.g., for cryptographic applications [13], [14], [15], [16], [17], [18], [19]. It is noted that we refrain from proposing the detection approaches which are not in-line with the implementation objectives of the respective designs. Therefore, it would only be reasonable to use error detection techniques that do not add unacceptable area overheads which could, potentially, result in impractical realizations.

For the proposed error detection schemes through time redundancy, we base the schemes on detecting both transient and permanent faults, i.e., in the first step, the actual operands are used for the operation run and in the second step, the operands are encoded, e.g., shifted (recomputing with shifted operands, RESO [20]) or rotated (recomputing with rotated operands, RERO [21]), such that the original result is achieved. We note that we pinpoint the architectures for which these schemes are utilized and we also propose variants of the RESO scheme through which the hardware overhead is reduced. We implement the proposed fault immune architectures on Xilinx FPGA families. Our results show that the proposed efficient error detection architectures are suitable for the required performance, reliability, and implementation metrics for constrained applications.

II. Preliminaries

We present the preliminaries in the following.

A. CORDIC Algorithm for Fixed-Angles of Rotations

The rotation of a two-dimensional vector $V_o(X_o, Y_o)$ through an angle θ, to obtain a rotated vector $V_n(X_n, Y_n)$ could be performed by the matrix product, $V_n = RV_o$, where $R = [\begin{smallmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{smallmatrix}]$.

Arbitrary angles of rotation are obtainable by performing a series of successively smaller elementary rotations. A good conversion method uses an additional adder-subtractor that accumulates the elementary rotation angles at each iteration. The elementary angles can be expressed in any convenient angular unit. Those angular values are supplied by a small look-up table (one entry per iteration) or are hardwired, depending on the implementation. Since the angle of rotation for

978-1-4799-5341-7/16/$31.00 ©2016 IEEE

the fixed rotation case is known beforehand, precomputations can be done and respective values can be stored (these values can be stored in a sign-bit register (SBR) in the CORDIC architecture). The rotation through any angle θ, $0 < \theta < 2\pi$, can be mapped into a positive rotation through $0 < \theta < \frac{\pi}{4}$ without any extra arithmetic operations [22].

B. CORDIC with Interleaved Scaling

In this CORDIC architecture, the micro-rotations and scalings are performed in alternate cycles in an interleaved fashion. This architecture requires an additional line-changer circuitry to alter the unsifted (direct) data input. Apart from the line-changer circuit, this design requires an additional ROM, unlike single-rotation cascade CORDIC circuit.

III. Proposed Error Detection Scheme

In this section, through recomputing with encoded operands approaches, we present schemes in which both transient and permanent faults can be detected.

A. Recomputing with Encoded Operands for Interleaved Scaling CORDIC

In what follows, the CORDIC architecture for fixed-angle of rotation with Interleaved Scaling is designed through recomputing with encoded operands, e.g., RESO, RERO, and variants of RESO, as shown partly in Fig. 1 (for the sake of brevity, the left wing is not shown) with the locations of the error detection modules shaded. Pipeline registers are added to sub-pipeline the design which help in dividing the timing into sub-parts.

1) Variants of Recomputing with Encoded Operands: We propose a number of detection schemes for both transient and permanent faults. We have utilized RESO which performs the recomputation step with shifted operands, i.e., all operands are shifted left or right by k bits. This method is efficient in detecting k consecutive logic errors and $k-1$ arithmetic errors. For the CORDIC architecture, let us assume f is the function performed on operands x and y such that $f(x,y)$ is the result of the operation which is stored in a register. The same operation is performed again with x and y shifted by certain number of bits. This new result $f'(x,y)$ is stored in a new register. The original result $f(x,y)$ can be obtained by shifting $f'(x,y)$ in the opposite direction to that of the operands if and only if there are no defects. If there are any defects, the value stored in $f'(x,y)$ can never be restored back to $f(x,y)$ by shifting the value in the opposite direction. In our design, the RESO method applied is for one shift; yet, the approach is general. The recomputation step takes $(n+k)$ bit operation time in this technique.

For the CORDIC architecture, our utilized RESO requires $(n+k)$ number of bits for its operation in order to preserve the k number of bits moving out. In this paper, we also employ a RESO variant for the CORDIC architecture with Interleaved Scaling. This method is a modified version of the RESO scheme and the modification done is that the bits that are shifted out are not preserved. This signifies that the total number of bits required for operation is only "n" bits and, hence, the architecture becomes less involved. In the modified RESO, only $(n-k)$ LSBs of function is compared with the shifted $(n-k)$ LSBs for error detection. This approach is a compromise between area/power consumption and error coverage, based on the architecture objectives and requirements.

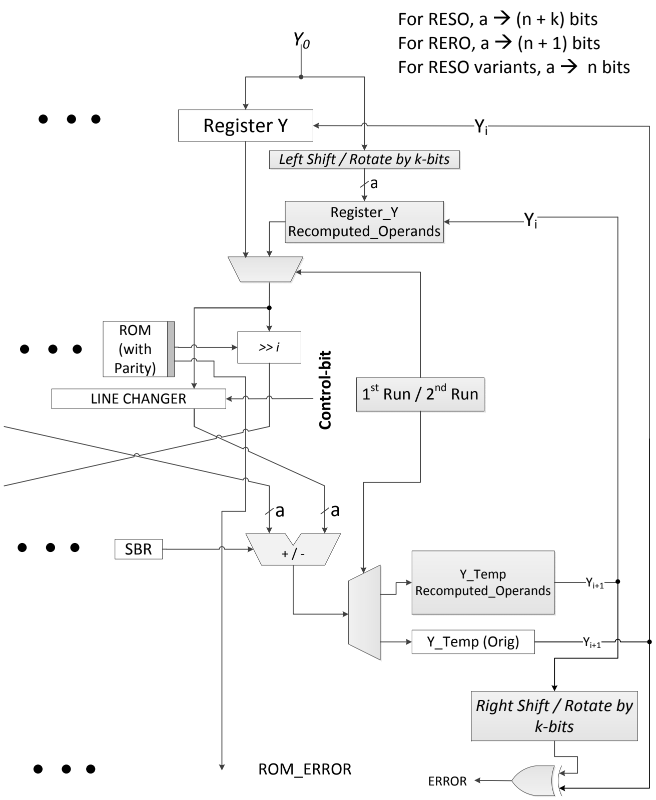

Figure 1. Error detection in fixed-angle of rotation CORDIC design with Interleaved Scaling.

For the CORDIC architecture, in RESO method, when n-bit operand is shifted left by k bits, the operand's leftmost k bits, i.e., the 'k' number of most significant bits, shift out. In order to preserve the moving out bits, all the units in the datapath are required to handle $(n+k)$ bits, i.e., the adders, registers, and shifters. For example, if the original unit has 32 bits and if 'k' is equal to 16, then, the new unit for recomputation should be equal to 48 bits. In RERO, the sizes of the adders, registers, and rotators increase only by one bit, i.e., $n+1$ bits. It is proven that the RERO method effectively detects (k mod n) consecutive logical errors and (k mod $(n+1)$ -1) consecutive faults in arithmetic operations, where "n" is the length of original operands [21]. The first challenge in RERO for the CORDIC architecture is to avoid the interaction between the most significant bit of the original operand and the least significant bit of the original operand during the recomputation operation. This is accomplished by adding an extra bit to the original operands in the most significant position before the rotate operation is performed (the value of this bit will be

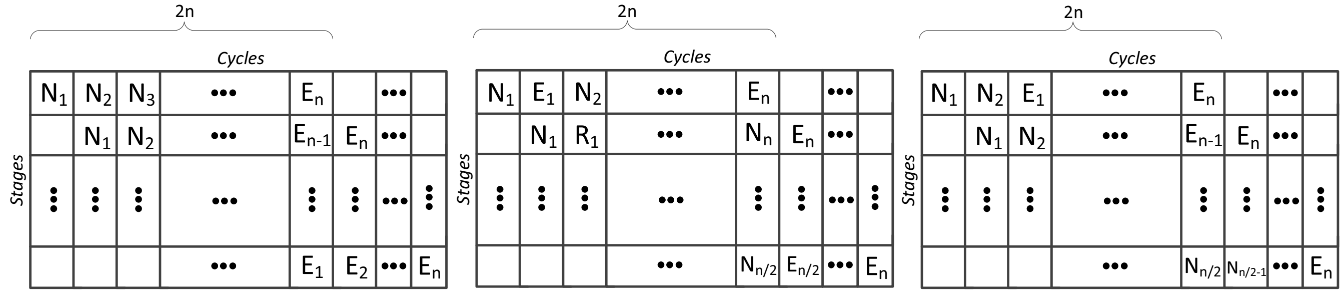

Figure 2. Data reliability and availability compromise through sub-pipelining.

equal to “0”). For the CORDIC architecture, this ensures that there is no carry-out to the LSB from this bit during the recomputation step. The second challenge in RERO for the CORDIC architecture is to ensure that the carry-out from the MSB of the rotated operand is connected with the carry-in of the LSB of the same rotated operand. Finally, we note that we propose performance enhancements through sub-pipelining to increase the frequency and alleviate the throughput overhead as part of the FPGA implementations.

2) Data Reliability and Availability: Suppose one pipeline-register has been placed to sub-pipeline the structures to break the timing path to approximately equal halves. Let us denote the two halves of pipelined stages by Ω_1 and Ω_2. The original input is first applied to the architecture and in the second cycle, while the second half of the circuit executes the first input, the encoded variant of the first input is fed to the first half of the circuit. This trend (which can be scaled to n stages) is consecutively executed for normal (N) and encoded (E) operands for Ω_n stages. Such approach ensures lower degradation in the throughput (and achieving higher frequencies) at the expense of more area overhead. Fig. 2 shows two possibilities for such a scheme. In the first one, output data availability has precedence over reliability (while both are achieved, the output data are derived first and fault diagnosis is performed after). Nevertheless, in the second approach, error detection is performed for each sub-segment of input data while the entire output is derived after such an order is followed. Depending on the requirements in terms of reliability and availability, one can tailor these approaches to fulfill such constraints, e.g., the third part of Fig. 2 in which an illustrative compromise is seen where two sub-segments are considered.

IV. Fault Injection Simulations and FPGA Implementations

In this section, the results of our fault injection simulations and FPGA implementations are presented.

A. Simulation Results

The proposed error detection schemes are capable of detecting both permanent and transient faults. Through simulations, it is derived that the proposed error detection schemes detect all single stuck-at faults. Nevertheless, in our fault model, the case for which multiple bits are flipped is also considered. The fault model applied for evaluating the proposed error schemes has been realized through linear feedback shift registers (LFSRs) to generate pseudo-random test patterns. LFSRs are used at different parts of the system in the following fashion. 16-bit LFSRs are used for adder/subtractor modules and registers and for the ROM, a 3-bit LFSR is used. The 16-bit LFSR is implemented with the polynomial $x^{16}+x^{13}+x^{11}+1$, whereas the 3-bit LFSR has the polynomial x^3+x^2+1.

In total, 10,000 faults are injected using the above-mentioned test cases. For each injection, error indication flag is observed and the result demonstrates a very high fault coverage of close to 100% (99.99%). The RERO, RESO, and modified variant of RESO provide full fault coverage of 100%, based on our simulations. However, it is noted that in fault checking, modified RESO ignores a specific number of bits (which get shifted during the recomputation step) [with the advantage of low overhead]. Therefore, if error occurs in those bits, the modified RESO is not able to identify the errors. This is a compromise based on reliability objectives and overhead tolerance and in applications where 100% fault coverage is necessary, it would be only ideal to use RERO or RESO.

B. FPGA Implementations

In this section, we present the overheads attained due to the proposed error detection schemes through FPGA implementations. We implement the proposed deigns over two diverse families of Xilinx FPGAs, i.e., Spartan-3A and Virtex-4, and discuss the overhead assessment results. This analysis is performed for the original CORDIC designs and also for the CORDIC designs with the proposed error detection structures using Xilinx ISE for Spartan-3A (XC3SD1800A-4FG676) and Virtex-4 (XC4VSX35-10FF668). CORDIC with Interleaved Scaling has two adder/subtractor modules with a ROM. Each of these parts are designed separately and are port-mapped at the top-level. The overheads are benchmarked for each error scheme as shown in Table I.

The throughput is degraded more for the schemes based on recomputing with encoded operands as two iterations with original and modified (shifted or rotated) operands have to be performed. However, as mentioned before, the performance degradations can be alleviated by the use of sub-pipeline

Table I
PERFORMANCE DEGRADATION COMPARISON FOR THE PROPOSED RECOMPUTING WITH ENCODED OPERANDS SCHEMES.

Design	Xilinx Spartan-3A DSP	Xilinx Virtex-4
	Throughput (Gbps) [deg.]	**Throughput (Gbps) [deg.]**
Original	2.58	5.13
RERO	2.40 [7.5%]	4.39 [16.9%]
RESO	2.53 [2.0%]	4.40 [16.6%]

Table II
OVERHEADS FOR THE PROPOSED RECOMPUTING WITH ENCODED OPERANDS SCHEMES.

Xilinx Spartan-3A DSP (XC3SD1800A-4FG676)			
Design	**Power (mW)**	**Area (# slices)**	**Max. freq.**
Original	120.6	270	165.0
RERO	124.8 (3.4%)	288 (6.6%)	143.5 (15%)
RESO	125.4 (3.9%)	289 (7.0%)	144.0 (14.5%)
M-RESO	124.6 (3.2%)	274 (1.4%)	144.4 (14.2%)
Xilinx Virtex-4 (XC4VSX35-10FF668)			
Original	460.0	265	328.4
RERO	469.4 (2.0%)	299 (12.8%)	264.4 (24.2%)
RESO	488.2 (6.1%)	299 (12.8%)	250.3 (31.2%)
M-RESO	476.0 (3.5%)	283 (6.7%)	234.3 (40.1%)

registers. At the expense of adding additional registers to sub-pipeline the architectures, higher frequencies which substantially increase the overall throughput can be achieved. The area in terms of number of slices, power consumption at 100 MHz frequency, and the maximum operating frequency are shown in Table II (M-RESO is the modified RESO architecture as explained before).

As seen in Tables I and II, the proposed error detection methodologies for CORDIC designs provide high error coverage at the expense of negligible and acceptable overheads on hardware platforms (Xilinx Spartan and Virtex FPGAs), which make the architectures for CORDIC designs with fixed-angle of rotation more reliable. Finally, we would like to emphasize that for application-specific integrated circuit (ASIC) platforms and other FPGA families, we expect similar results as the proposed schemes are platform-oblivious.

V. CONCLUSION

In this paper, efficient error detection schemes for CORDIC designs have been proposed. The Interleaved Scaling CORDIC is optimized for low-area applications; thus, design-space explorations of variants of recomputing with encoded operands have been presented. The simulation results show that high fault coverage (very close to 100%) is achieved for the injected faults through the proposed error detection schemes. Furthermore, the error detection structures have been implemented on different FPGA families, i.e., Xilinx Spartan-3A DSP and Virtex-4. The hardware implementation assessments show that the overheads gained by the error detection structures are acceptable. Thus, the proposed hardware architectures provide reliable and efficient structures which can be tailored based on the reliability requirements and the overhead tolerance.

ACKNOWLEDGMENT

This work has been partly funded by Texas Instruments Faculty award of M. Mozaffari-Kermani and R. Azarderakhsh.

REFERENCES

[1] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electron. Comput.*, vol. EC-8, pp. 330–334, Sep. 1959.

[2] J. S. Walther, "A unified algorithm for elementary functions," in *Proc. Spring Joint Comput. Conf.*, 1971, pp. 379–385.

[3] J. R. Cavallaro and F. T. Luk, "CORDIC arithmetic for a SVD processor," *J. Parallel Distrib. Comput.*, vol. 5, pp. 271–290, 1988.

[4] Y. H. Hu and S. Naganathan, "An angle recoding method for CORDIC algorithm implementation," *IEEE Trans. Comput.*, vol. 42, no. 1, pp. 99–102, Jan. 1993.

[5] Y. H. Hu and H. H.M. Chern, "A novel implementation of CORDIC algorithm using backward angle recoding (BAR)," *IEEE Trans. Comput.*, vol. 45, no. 12, pp. 1370–1378, Dec. 1996.

[6] C.-S. Wu, A.-Y. Wu, and C.-H. Lin, "A high-performance/low-latency vector rotational CORDIC architecture based on extended elementary angle set and trellis-based searching schemes," *IEEE Trans. Circuits Syst. II.*, vol. 50, no. 9, pp. 589–601, Sep. 2003.

[7] T.-B. Juang, S.-F. Hsiao, and M.-Y. Tsai, "Para-CORDIC: Parallel CORDIC rotation algorithm," *IEEE Trans. Circuits Syst. I: Regular Papers*, vol. 51, no. 8, pp. 1515-1524, 2004.

[8] T. Rodrigues and E. Swartzlander, "Adaptive CORDIC: Using parallel angle recoding to accelerate CORDIC rotations," in *Proc. Asilomar Conf.*, Oct.–Nov. 2006, pp. 323–327.

[9] B. G. Blundell, *An Introduction to Computer Graphics and Creative 3-D Environments.* London, U.K.: Springer-Verlag, 2008.

[10] T. Lang and E. Antelo, "High-throughput CORDIC-based geometry operations for 3D computer graphics," *IEEE Trans. Comput.*, vol. 54, no. 3, pp. 347–361, Mar. 2005.

[11] P. Meher, J. Walls, T. Juang, K. Sridharan, and K. Maharatna, "50 years of CORDIC: Algorithms, architectures and applications," *IEEE Trans. Circuits Syst. I*, vol. 56, no. 9, pp. 1893–1907, Sep. 2009.

[12] P. Meher and S. Park, "CORDIC designs for fixed angle of rotation," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 2, pp. 217–228, Feb. 2013.

[13] M. Mozaffari Kermani, R. Azarderakhsh, and A. Aghaie, "Reliable and error detection architectures of Pomaranch for false-alarm-sensitive cryptographic applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 12, pp. 2804-2812, Dec. 2015.

[14] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "Concurrent structure independent fault detection schemes for the Advanced Encryption Standard," *IEEE Trans. Comput.*, vol. 59, no. 5, pp. 608–622, May 2010.

[15] P. Maistri and R. Leveugle, "Double-Data-Rate computation as a countermeasure against fault analysis," *IEEE Trans. Comput.*, vol. 57, no. 11, pp. 1528-1539, Nov. 2008.

[16] M. Mozaffari Kermani, R. Azarderakhsh, and S. Sarmadi, "Fault-resilient lightweight cryptographic block ciphers for secure embedded systems," *IEEE Embedded Sys.*, vol. 6, no. 4, pp. 89-92, Dec. 2014.

[17] S. Bayat-Sarmadi, M. Mozaffari Kermani, and A. Reyhani-Masoleh, "Efficient and concurrent reliable realization of the secure cryptographic SHA-3 algorithm," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 33, no. 7, pp. 1105-1109, Jul. 2014.

[18] M. Mozaffari Kermani and R. Azarderakhsh, "Reliable hash trees for post-quantum stateless cryptographic hash-based signatures," in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems (DFT)*, pp. 103-108, Oct. 2015.

[19] M. Mozaffari Kermani and A. Reyhani-Masoleh, "Reliable Hardware Architectures for the Third-Round SHA-3 Finalist Grostl Benchmarked on FPGA Platform," in *Proc. IEEE Int. Symp. DFT*, pp. 325-331, Vancouver, Canada, Oct. 2011.

[20] J. H. Patel and L. Y. Fung, "Concurrent error detection in ALUs by recomputing with shifted operands," *IEEE Trans. Comput.*, vol. C-31, no. 7, pp. 589–595, Jul. 1982.

[21] J. Li and E. E. Swartzlander, "Concurrent error detection in ALU's by recomputing with rotated operands," in *Proc. IEEE Int. Workshop on Defect and Fault Tolerance in VLSI Systems*, 1992, pp. 109–116.

[22] C. Y. Kang and E. E. Swartzlander, "Digit-pipelined direct digital frequency synthesis based on differential CORDIC," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 5, pp. 1035–1044, May 2006.