Reliable Hardware Architectures of the CORDIC Algorithm With a Fixed Angle of Rotations

Rajkumar Ramadoss, *Student Member, IEEE*, Mehran Mozaffari Kermani, *Senior Member, IEEE*, and Reza Azarderakhsh, *Member, IEEE*

***Abstract*—Fixed-angle-rotation operation of vectors is widely used in signal processing, robotics, and graphics. Various optimized coordinate rotation digital computer (CORDIC) designs have been proposed for uniform rotation of vectors through known and specified angles. Nevertheless, in the presence of faults, such hardware architectures are potentially vulnerable. In this brief, we propose efficient error detection schemes for cascaded single-rotation CORDIC which hamper the performance of the architectures negligibly. We present signature-based schemes for this CORDIC variant to detect both transient and permanent faults. We further present how such schemes can be applied to other variants of CORDIC. The effectiveness of the proposed designs is assessed through field-programmable gate array implementations and error simulations. The results give confidence for the proposed efficient architectures which can be tailored based on the reliability requirements and overhead tolerance.**

***Index Terms*—Coordinate rotation digital computer (CORDIC), fault detection, reliability.**

I. Introduction

COORDINATE rotation digital computer (CORDIC) algorithm is an efficient iterative approach used for rotating vectors on a plane. Low latencies could be achieved for this algorithm; yet, it is mainly attractive because of its low-complexity hardware implementations. 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. Various CORDIC architectures have been proposed for the rotation of vectors through fixed angles. Meher *et al.* [1] presented several optimized fixed rotation CORDIC designs with reduced number of micro-rotations and reduced complexity of barrel-shifters. In applications where fixed angles of rotation have to be performed, the angles are known beforehand. Therefore, in [1], it is proposed to perform an exhaustive search to achieve an optimal elementary-angle-set of micro-rotations for the given known angle(s).

In this brief, we propose a number of error detection schemes for fixed-angle rotation CORDIC designs. We consider stuck-at fault model in which the nodes in the architectures are stuck-at zero or one, regardless of their correct values. Digital circuits are prone to natural errors caused due to alpha particles from package decay, cosmic rays creating energetic neutrons, protons, and thermal neutrons. There have been a number of works on fault diagnosis in the hardware architectures of different usage models, e.g., for cryptographic applications [2]–[5]. In addition, the work presented in [6] presents an error detection and correction mechanism for CORDIC. In [6], the work proposes a computation-skip scheme to maintain the cycles per instruction (CPI) without throughput penalty at subcritical situations.

Manuscript received June 8, 2016; accepted November 1, 2016. Date of publication November 2, 2016; date of current version July 31, 2017. The work of M. Mozaffari Kermani and R. Azarderakhsh were supported by Texas Instruments Faculty Award. This brief was recommended by Associate Editor F. Lau.

R. Ramadoss and M. Mozaffari Kermani are with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY 14623 USA (e-mail: rxr1052@rit.edu; m.mozaffari@rit.edu).

R. Azarderakhsh is with the Department of Computer and Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL 33431 USA (e-mail: razarderakhsh@fau.edu).

Digital Object Identifier 10.1109/TCSII.2016.2624508

II. Preliminaries

The rotation of a 2-D 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 $R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$. The CORDIC algorithm works in two modes: 1) in the rotating mode, the input vector with coordinates (x, y) is rotated by an input angle θ to achieve the new vector with coordinates (x', y') and 2) in the second mode of operation, the vectoring mode, the input vector is rotated to x-axis while returning the angle θ as output is required to make that rotation.

Since the angle of rotation for 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 in the CORDIC architecture). The rotation through any angle θ, $0 < \theta < 2\pi$, can be mapped into a positive rotation through $0 < \theta < (\pi/4)$ without any extra arithmetic operations [5]. Therefore, the rotation mapping is done so that the rotation angle lies in the range of $0 < \theta < (\pi/4)$.

III. Proposed Error Detection Schemes

In what follows, we present error detection techniques for single rotation CORDIC architecture.

A. Motivating Experiments

In signal processing, CORDIC techniques are used for fixed/adaptive filtering and computation of discrete Hartley,

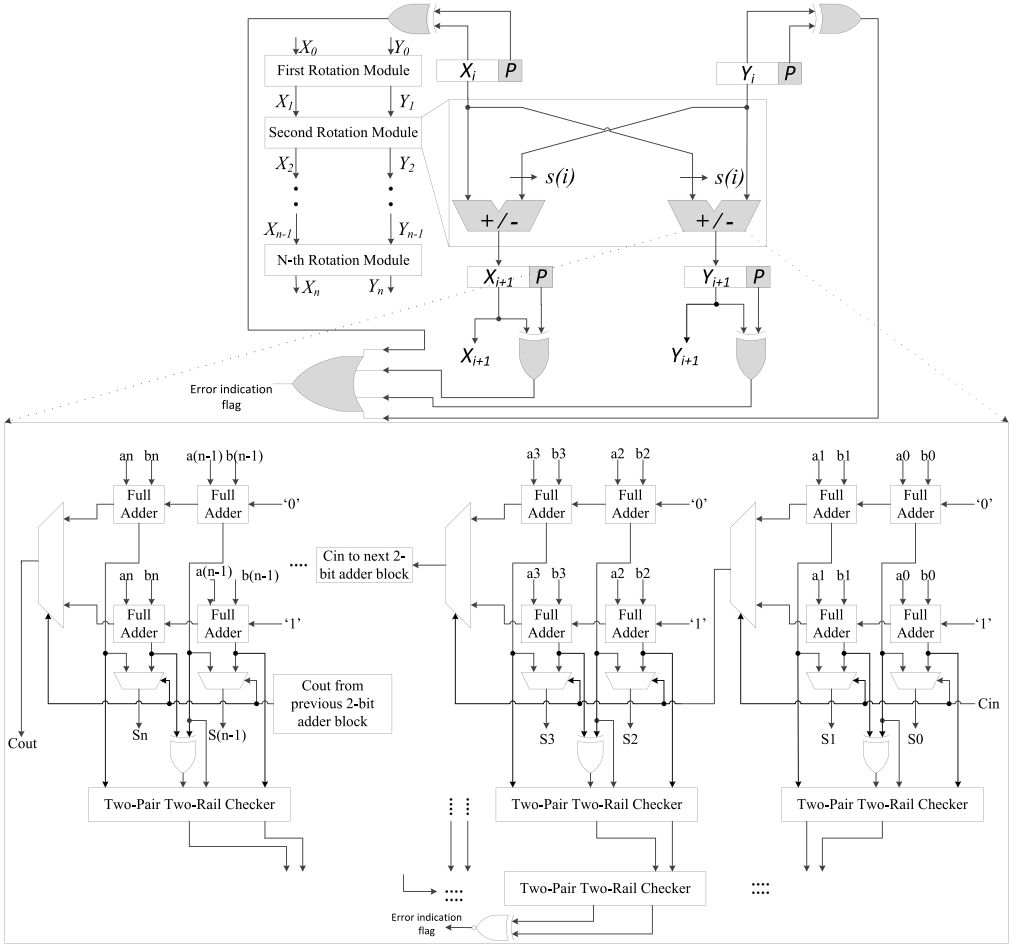

Fig. 1. Error detection in single rotation cascaded CORDIC through unified signature-based scheme.

sinusoidal and cosine transforms. It is also used for direct digital synthesis and digital/analog modulation in the field of communications. Therefore, it is understood that CORDIC algorithm is employed in extremely-fault-sensitive applications, where even a single-bit error is not tolerated. Besides, CORDIC is an iterative approach and hence, even a single stuck-at fault may lead to erroneous (multibit) result as seen in our experiments. In our error simulations, 10 000 random test vectors with single stuck-at faults are injected to the original single cascaded CORDIC design. The test vectors were found to provide faulty results with roughly 60%–70% change in bits, on an average, in the output (for just a single fault injected). As a result, it can be determined that even a single-bit fault in CORDIC is catastrophic and, hence, it is exceedingly important to employ error detection methodologies.

B. Unified Signature-Based Scheme

For low-area applications, it is not suitable to use double/triple modular redundancy as roughly 100% and 200% overhead (including the voters and comparators). The critical elements in single rotation cascaded CORDIC architecture are the datapath registers and adder/subtractor modules. Therefore, signatures (single-bit, multiple-bit, or interleaved parity, cyclic redundancy check, etc.) are employed for all the registers. Moreover, self-checking adders based on dual-rail encoding are included for the adder/subtractor modules.

1) Signature-Based Checking for Datapath Registers: Concurrent error detection through signatures for registers are utilized throughout the architectures. The resulting error indication flags are derived to alert as shown in Fig. 1. The critical compromise here is the error coverage, e.g., single parity-bit error detection technique might have low error coverage for critically-sensitive applications (the merit here is the iterative nature of CORDIC which gives confidence on high error coverage).

2) Self-Checking Adder and Subtractor Modules Based on Two-Rail Encoding: Many adder designs have been proposed; yet, in our design, we use carry-select type adders for their high speed and low area. We propose using self-checking carry-select adder designs based on a series of cascaded totally self-checking 2-bit adders.

The self-checking adder design used in the single rotation cascaded CORDIC architecture is shown in Fig. 1. XNOR gates and two-pair two-rail (TPTR) checkers are the additional resources utilized for error detection. The checker has two pairs of inputs (x_0, y_0) and (x_1, y_1). The inputs are driven in such a way that in the fault free scenario, $x_0 = y_0'$ and $x_1 = y_1'$. This is performed using XNOR gates and appropriate connections as explained in the following. There are two outputs from

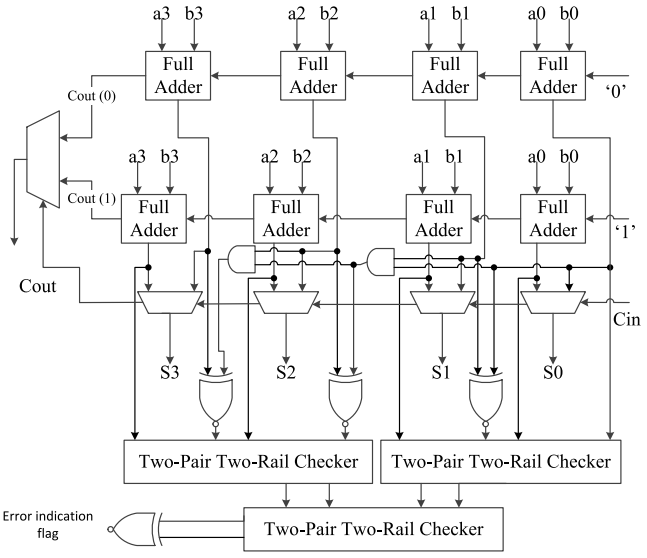

Fig. 2. Variant of a 4-bit self-checking carry-select adder.

Algorithm 1 TPTR Checking

1. if (x_1 = '1') then	2. if (y_1 = '1') then
$z_0 = y_0$;	$z_1 = y_0$;
else	else
$z_0 = x_0$;	$z_1 = x_0$;
end if;	end if;

the checker and the outputs are also in two-rail form as the inputs, i.e., $z_0 = z_1'$. Therefore, two of the output combinations 01 and 10 are considered valid fault-free codewords whereas the remaining two output combinations 00 and 11 indicate presence of faults. Even if one of the inputs of the checker has a fault, the output is not in two-rail form and thus an error indication flag is raised. Let us consider $S0$ and $S01$ are the sum outputs of the two full adders with inputs ($a0$, $b0$) and ($a1$, $b1$) and carry-in equal to "0." The corresponding carry-out signals are $C01$ and $C02$. Similarly, $S10$ and $S11$ are the two sum outputs of the full adders with inputs ($a0$, $b0$) and ($a1$, $b1$) and carry-in equal to "1." The carry-out signals in this case would be $C11$ and $C12$. It can be shown that the sum bits $S00$ and $S10$ are always complementary to each other. Therefore, these two signals are directly fed into the TPTR checker as inputs x_0 and y_0. Furthermore, it is proven that S01 is always complementary to the XNOR of $S00$ and $S11$. Thus, signal $S01$ and signal ($S00$ XNOR $S11$) form the other two complementary input pairs to the checker, i.e., x_1 and y_1. Algorithm 1 describes such process.

A variant of self-checking adder design utilizes two n-bit ripple carry adders to precompute the sum bits with complemented values of C_{in}, i.e., 0 and 1, and the original value of C_{in} is used to select the actual sum bits [7]. We employ this new adder in the single rotation cascaded CORDIC architecture and evaluate its performance and efficiency. Fig. 2 shows the design module of a 4-bit self-checking carry-select adder; an n-bit model of the same design module is employed in place of the adder/subtractor unit in the single rotation cascaded CORDIC circuit. An important modification done in this new adder is that the inputs are given to the TPTR checker. Let us consider $S0n$ is the sum output of the full adder with inputs (an, bn) with the initial C_{in} equal to 0. An XNOR operation is performed between $S0n$ and the product of all the lower sum bits computed in the full adders at the initial C_{in} = "0". The output of the XNOR is given as one of the inputs to the TPTR checker, say x_0. The other input (y_0) is the sum output $S1n$ of the full adder with inputs (an, bn) with the initial C_{in} equal to 1. The outputs of the XNOR and $S1n$ are always complementary to each other and, hence, are chosen as the inputs to the TPTR checker. For example, at bit-position "3," one input to the checker will be equal to $S03$ XNOR ($S00$ AND $S01$ AND $S02$), whereas the other complementary input will be equal to $S13$. As a result, there will be 2 XNORs per TPTR checker as opposed to 1 XNOR per TPTR checker in the previous adder design which is one of the reasons for additional area cost.

The TPTR checker can be used for error detection in the proposed scheme. One can use error correction techniques as well to correct faults in the architectures of the presented CORDIC. Let us go over a possible correction technique that can be used for carry-select adders within CORDIC, adopted from [8]. The carry-select adder can be used in conjunction with an increment/decrement circuit, one NOT gate, and one multiplexer, i.e., correction-multiplexer. Increment/decrement is fast and also has low area and power consumption and will increment its input when its control line in 0 and will decrement the input when its control line is 1. Two "sum" inputs are used for the correction-multiplexer. The carry-in input is used as the select of this multiplexer which chooses between the two sum inputs to correct the faults. Finally, error correction in [6] is performed through a mixture of algorithm and circuit level schemes: it mitigates the error at a very tight timing window to maintain the constant CPI. When timing violations are detected, the computations in the next cycle are skipped as an approximation. This method proposes an error correction scheme which can be embedded in this brief as well. Skipping trivial operations in CORDIC is a parallel method to this brief which can be combined based on the reliability requirements and overhead tolerance.

Signature-based schemes, e.g., parity or CRC-based approaches or the scheme presented in this brief, add a detection circuitry to the original algorithm implementation to detect faults. In general, such detection components can be prone to faults. Let us consider three cases.

1) When faults happen in the original architecture, the discussions presented in the proposed scheme hold, and can be utilized for detections.
2) If faults happen in the TPTR checker circuitry only, they are detected. We note that in such cases, faults are correctly detected with respect to the union of the original and fault detection components.
3) If faults occur in both the original and the fault detection circuitry they are neither single stuck-at fault (single-event upset) nor burst fault which happen in specific parts of the architecture.

Dealing with distributed multiple faults in such cases, if they happen, can be performed through selective hardening

(one needs to note that not all of such cases is among undetected faults). Such additional circuitry can be hardened when implemented, e.g., through radiation hardening which is the act of making electronic components and systems resistant to damage or malfunctions caused by ionizing radiation (particle radiation and high-energy electromagnetic radiation). We also note that using recomputing with encoded operands, one can detect such cases which can be used as an add-on scheme.

C. Applicability to Other Variants

The proposed architectures can be employed to other variants of fixed-angle-rotation CORDIC, such as CORDIC designs with interleaved scaling and bi-rotation cascaded CORDIC as well as other iterative architectures such as conventional CORDIC architectures with nonfixed-angle rotations. This is due to the fact that these architectures share similar sub-blocks like datapath registers, adder/subtractor, ROM, and barrel shifters. Traditionally, fixed or nonfixed angle CORDIC architectures have common sub-blocks, such as two data registers, two barrel shifters, two adder/subtractor units, and a memory unit. Signature-based error checking can be used for datapath registers, and the memory unit (ROM), while self-checking adder designs with TPTR can be employed for the adders and subtractors. In general, all CORDIC architectures (fixed and nonfixed angle-rotations) are iterative in nature and hence, fast adders/subtractors are crucial. The proposed error detection schemes with self-checking adders/subtractors and TPTR guarantee reliable and fast operations, making them suitable for error detection in CORDIC designs.

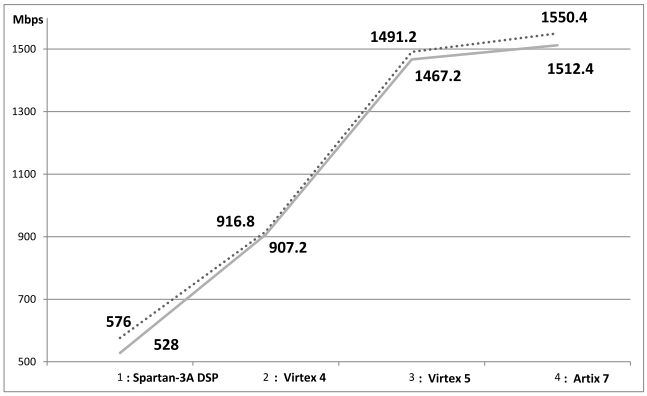

Fig. 3. Throughput for the original (dotted line) and proposed (solid line) architectures for four FPGA families.

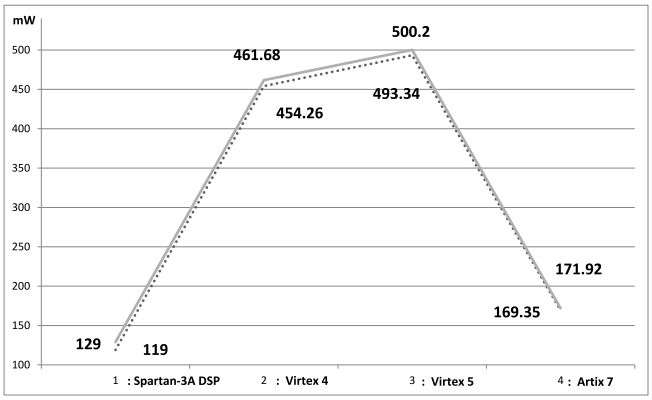

Fig. 4. Power consumption for the original (dotted line) and proposed (solid line) architectures for four FPGA families.

IV. Benchmark

A. Error Simulation Results

We first present the error model.

1) Error Model: The proposed error detection schemes are capable of detecting both *permanent* and *transient* faults. In our simulations, we consider both single and multiple stuck-at fault scenarios. *Single event upset* and *multiple event upset* are both considered. 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, thereby providing a more real-world scenario to the error structures. 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. The LFSR outputs have been either ANDed or ORed with the actual outputs to generate transient fault patterns.

2) Experiment Results: Three different test cases have been used for fault simulations. In the first case, one fault is injected into the entire system either in the output of the register or adder/subtractor or in the output of the ROM. In the second case, two faults are injected in different combinations of adder, register, and ROM. Finally, in the third case, three faults are injected. In total, 10 000 faults are injected for each of the above-mentioned test cases (the detected cases are 9999 instances). For each injection, error indication flag is observed and the result demonstrates a very high fault coverage of close to 100% (99.99%). We note that the signature-based error checking technique provides 100% coverage of single stuck-at faults (10 000 injected and detected cases) and for multiple stuck-at faults, this technique has slightly lower coverage but close to 100%. We note that single stuck-at faults can only happen in one architecture. If faults occur in both the original and the fault detection circuitry, they are neither single stuck-at fault (single-event upset) nor burst fault which happen in specific parts of the architecture. Thus, for multiple faults, the reason for slight degradation in error coverage is that they can happen in both architectures, bypassing the detection mechanism.

B. FPGA Implementations

We implement the proposed deigns for four families of Xilinx field-programmable gate arrays (FPGAs) 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), Virtex-4

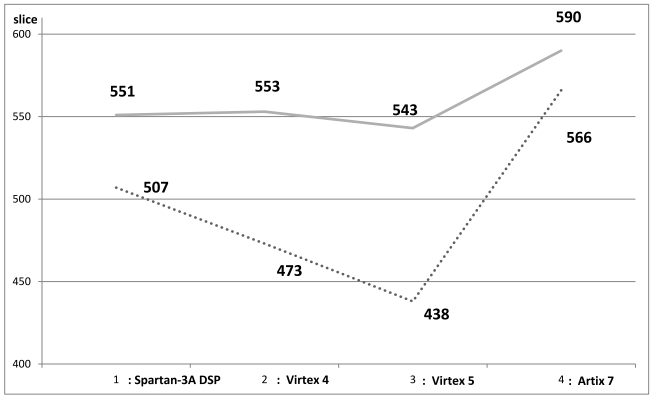

Fig. 5. Area for the original (dotted line) and proposed (solid line) architectures for four FPGA families.

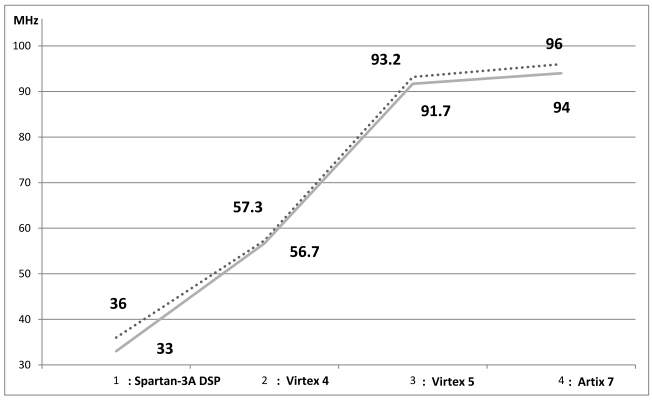

Fig. 6. Frequency for the original (dotted line) and proposed (solid line) architectures for four FPGA families.

(XC4VSX35-10FF668), Virtex-5 (XC5VLX20T-2FF323), and Artix-7 (XC7A100T-2CSG324).

The designs are divided into seven rotation modules and each rotation module has two self-checking or ordinary adder/subtractor modules. The overheads are benchmarked as shown in Figs. 3–6 (power consumption at 100 MHz frequency). In Fig. 3, we have shown the original and proposed architectures and the throughputs which are degraded as 8.1%, 1.0%, 1.6%, and 2.1%, respectively. In Figs. 4–6, power consumptions, area, and frequencies are depicted. Based on Fig. 4, the power consumption overheads are, respectively, 8.6%, 1.6%, 1.4%, and 1.5%. Moreover, the area overheads are 8.7%, 16.9%, 23.9%, and 21.9%, and the frequency degradations are 8.1%, 1.0%, 1.6%, and 2.1%, respectively, based on Figs. 5 and 6. We note that the relation between Figs. 3 and 6 is that higher throughput degradation translates to higher frequency degradation.

An efficient error detection scheme for CORDIC is proposed in [9] with area overheads of 9.5%. However, the scheme in [9] covers only single-bit transient errors. Although, the error detection schemes we propose in this brief have high area cost, it is more efficient due to the high fault coverage, i.e., it covers both single-bit and multiple-bit transient and permanent errors. We have also considered error detection schemes using recomputing with encoded operands which has lower area overheads in the range of 6%–10%. However, such schemes utilize time redundancy which does not provide 100% fault coverage and is quite slower compared to the proposed scheme in this brief.

The FPGA experiments shown above were performed with 16-bit floating point inputs. Thus, the results of CORDIC vector rotations are precise and have an accuracy close to 100%. The obtained overheads correspond to these inputs which are 16 bits in size. However, to test the effectiveness of our design in a larger design space, the same experiments were conducted by employing the proposed error detection schemes for 32-bit size floating point inputs. We have derived that the area and power consumption overheads were increased by roughly 2% and the maximum frequency of operation was reduced by a factor of approximately 1.5%. The changes in the overheads are negligible and our proposed error detection schemes can be applied to large design space of CORDIC architectures.

V. Conclusion

In this brief, efficient error detection schemes for CORDIC designs with a fixed angle of rotation have been proposed. 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 FPGAs. The hardware implementation assessments show that the overheads gained by the error detection structures are acceptable. Thus, the proposed hardware architectures for a fixed angle of rotation CORDIC designs provide reliable and efficient structures which can be tailored based on the reliability requirements and the overhead tolerance.

References

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

[2] 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.

[3] M. Mozaffari-Kermani and R. Azarderakhsh, "Efficient fault diagnosis schemes for reliable lightweight cryptographic ISO/IEC standard CLEFIA benchmarked on ASIC and FPGA," *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5925–5932, Dec. 2013.

[4] 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.

[5] 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.

[6] Y. Huang, M. Li, C. Li, P. Debacker, and L. Van der Perre, "Computation-skip error resilient scheme for recursive CORDIC," in *Proc. Signal Process. Syst.*, Belfast, U.K., 2014, pp. 1–6.

[7] M. A. Akbar and J.-A. Lee, "Comments on 'self-checking carry-select adder design based on two-rail encoding,"' *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 7, pp. 2212–2214, Jul. 2014.

[8] A. Namazi, S. G. Miremadi, and A. Ejlali, "A high speed and low cost error correction technique for the carry select adder," in *Proc. Int. Conf. Availability Rel. Security*, Fukuoka, Japan, 2009, pp. 635–640.

[9] S. Wang, Z. Wen, and L. Yu, "High-performance fault-tolerant CORDIC processor for space applications," in *Proc. Symp. Syst. Control Aerosp.*, Harbin, China, 2006, pp. 360–363.