CORDIC II: A New Improved CORDIC Algorithm

Mario Garrido, *Member, IEEE*, Petter Källström, Martin Kumm, and Oscar Gustafsson, *Senior Member, IEEE*

***Abstract*—In this brief, we present the CORDIC II algorithm. Like previous CORDIC algorithms, the CORDIC II calculates rotations by breaking down the rotation angle into a series of microrotations. However, the CORDIC II algorithm uses a novel angle set, different from the angles used in previous CORDIC algorithms. The new angle set provides a faster convergence that reduces the number of adders with respect to previous approaches.**

***Index Terms*—CORDIC, friend angles, nanorotation, rotation, uniformly scaled redundant (USR) CORDIC.**

I. Introduction

THE CORDIC algorithm [1] is the algorithm par excellence to calculate rotations in digital systems. Its main principle is simple: It breaks down the rotation angle in a sum of angles and carries out the rotation by a series of the so-called microrotation by these angles. The benefit of the CORDIC algorithm is that the microrotations are calculated by simple shift-and-add operations, which is very efficient in hardware.

Many variations of the CORDIC algorithm have been proposed in the literature. In this brief, we are interested in those approaches that are used to calculate general rotations. This means that they rotate by any angle provided as an input of the rotator. Constant rotators used for specific sets of rotation angles are not considered in this brief but are studied in [2].

Among general rotators, we find numerous versions of the CORDIC algorithm. Some works combine several microrotation stages into a single stage [3], [4] in order to reduce the number of iterations of the CORDIC. The work in [5] is based on skipping and/or repeating microrotations. Some approaches focus on representing the microrotation using a Taylor series approximation [6]–[8]. Other approaches divide the microrotations into coarse and fine parts [9]. Some works focus on reducing the rotation memory [9]–[11]. Scaling-free CORDIC approaches pursue to compensate the scale factor of the CORDIC [6], [7]. Reviews of CORDIC techniques can be found in [12] and [13].

In this brief, we pursue a pipelined CORDIC design with the minimum number of adders. We call it the CORDIC II algorithm. It differs from previous approaches in the used set of microrotations, called *angle set* in the following. The new set of microrotations provides a fast convergence of the rotation angle. This leads to a reduced latency and a smaller number of adders than in previous CORDIC algorithms.

Manuscript received June 13, 2015; accepted August 18, 2015. Date of publication September 28, 2015; date of current version January 28, 2016. This brief was recommended by Associate Editor Z. Zhang.

M. Garrido, P. Källström, and O. Gustafsson are with the Division of Computer Engineering, Department of Electrical Engineering, Linköping University, 58183 Linköping, Sweden (e-mail: mario.garrido.galvez@liu.se; petter.kallstrom@liu.se; oscar.gustafsson@liu.se).

M. Kumm is with the Digital Technology Group, University of Kassel, 34121 Kassel, Germany (e-mail: kumm@uni-kassel.de).

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2015.2483422

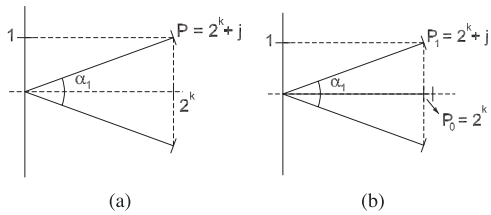

Fig. 1. CORDIC microrotation angles. (a) Conventional CORDIC. (b) Redundant CORDIC.

II. Background

A. Rotations in Digital Systems

This section reviews the key concepts related to rotations in digital systems. Further information can be found in [2] and [14].

In a digital system, a rotation by an angle α can be described as a multiplication by a complex coefficient $P = C + jS$

$$\begin{bmatrix} X_D \\ Y_D \end{bmatrix} = \begin{bmatrix} C & -S \\ S & C \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{1}$$

where $x + jy$ is the input and $X_D + jY_D$ is the result of the rotation. C and S are b-bit integer numbers in 2's complement in the range $[-2^{b-1}, 2^{b-1} - 1]$. They are obtained from the rotation angle as [14]

$$\begin{aligned} C &= R \cdot (\cos\alpha + \epsilon_c) \\ S &= R \cdot (\sin\alpha + \epsilon_s) \end{aligned} \tag{2}$$

where ϵ_c and ϵ_s are the quantization errors of the cosine and sine components, respectively, and R is the scaling factor. The output $X_D + jY_D$ is also scaled by R.

The rotation error $\epsilon = \sqrt{\epsilon_c^2 + \epsilon_s^2}$ is the distance between the exact rotation and the actual rotation due to quantization. If the rotator has multiple rotation angles $\alpha_i, i = 1, \ldots, M$, with their corresponding coefficients $P_i = C_i + jS_i$, the rotation error [14] is calculated as

$$\epsilon = \max_i (\epsilon(i)) = \max_i \left(\sqrt{\epsilon_c^2(i) + \epsilon_s^2(i)} \right). \tag{3}$$

Finally, the effective word length is the number of bits of the output that are guaranteed to be accurate [2] and is calculated from the rotation error as

$$WL_E = -\log_2 \frac{\epsilon}{2\sqrt{2}} = -\log_2 \epsilon + \frac{3}{2}. \tag{4}$$

B. CORDIC Algorithm

The CORDIC algorithm considers the coefficients $P = C + jS = 2^k + j\delta_k$, where $\delta_k \in \{-1, 1\}$ and $k = 0, \ldots, M$ is the microrotation stage. The corresponding angles are $\alpha_k = \tan^{-1}(S/C) = \delta_k \tan^{-1}(2^{-k})$. This is shown in Fig. 1(a).

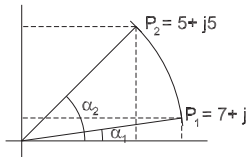

Fig. 2. Example of friend angles for $P_1 = 7 + j$ and $P_2 = 5 + j5$.

The CORDIC algorithm breaks down the rotation angle θ into a sum of microrotations by the angles α_k, i.e.,

$$\theta = \sum_{k=0}^{M} \alpha_k + \epsilon_\phi \tag{5}$$

where ϵ_ϕ is the remaining phase error.

Each microrotation stage calculates

$$\begin{bmatrix} X_D \\ Y_D \end{bmatrix} = \begin{bmatrix} 2^k & -\delta_k \\ \delta_k & 2^k \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{6}$$

where δ_k determines the direction of the rotation and the scaling factor of the stage is $R(k) = \sqrt{2^{2k} + 1}$.

The rotation error at each microrotation stage is $\epsilon = 0$, and the word length is $WL_E = \infty$. This means that the coefficient P_k rotates exactly α_k degrees, and the scaling factor for both angles in each microrotation is the same. The latter is always true, as the coefficients are conjugated.

C. Redundant CORDIC Algorithm

The redundant CORDIC [15] algorithm uses the same set of coefficients $P_1 = C_1 + jS_1 = 2^k + j\delta_k$ as the CORDIC algorithm, with the particularity that $\delta_k \in \{-1, 0, 1\}$. This adds a rotation $P_0 = 2^k$ by 0° in the kernel, as shown in Fig. 1(b). This increases the set of alternative angles per stage, which implies a faster convergence to the rotation angle. However, it has the drawback that the scaling of the angles of the kernel is different. Therefore, redundant CORDIC algorithms need special stages to compensate the scaling and thus reduce the rotation error.

III. Basic Angle Sets

There are three new types of angle sets proposed for CORDIC II, which are described in the following.

A. Friend Angles

We define friend angles as a set of angles α_i for which there exists a set of coefficients $P_i = C_i + jS_i$ with angles α_i, i.e., $\alpha_i = \tan^{-1}(S_i/C_i)$, whose magnitude is the same, i.e., $\forall\, i, j, |P_i| = |P_j|$. As all of the coefficients have the same magnitude, a kernel composed by friend angles α_i does not have any rotation error. This is equivalent to saying that $WL_E = \infty$.

The angles $\alpha_1 = 8.13°$ and $\alpha_2 = 45°$ are an example of friend angles. For these angles, there exist the coefficients $P_1 = 7 + j$ and $P_2 = 5 + j5$ whose angles are α_1 and α_2, respectively, and $|P_1| = |P_2| = \sqrt{50}$. This example is shown in Fig. 2.

A property that can be extracted from the definition of friend angles is that any angle α is a friend to itself and also to $-\alpha + n\pi/2$ and $\alpha + n\pi/2$ for any value of n. According to this property, the angles used in the CORDIC for each microrotation stage are friend angles. This happens because each microrotation only considers the pair of angles $\pm\alpha_k$.

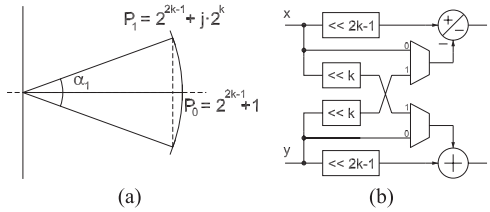

Fig. 3. USR CORDIC. (a) Graphical representation of the coefficients. (b) Hardware circuit.

TABLE I
USR CORDIC Rotations

k	P_0	α_0	P_1	α_1	R	WL_E
1	3	0	2+j2	45	2.91	6.59
2	9	0	8+j4	26.5651	8.97	9.83
3	33	0	32+j8	14.0362	32.99	13.59
4	129	0	128+j16	7.125	128.99	17.52
5	513	0	512+j32	3.5763	512.99	21.51
6	2049	0	2048+j64	1.7899	2048.99	25.50
7	8193	0	8192+j128	0.89517	8193.00	28.50
8	32769	0	32768+j256	0.44761	32769.00	32.50

B. USR CORDIC

The uniformly scaled redundant (USR) CORDIC rotations use the same rotation angles as the redundant CORDIC. However, all of the angles have a similar scaling. The coefficients for the USR CORDIC are

$$\begin{aligned} P_0 &= 2^{2k-1} + 1 \\ P_1 &= 2^{2k-1} + j2^k \end{aligned} \tag{7}$$

and the graphical representation of the USR CORDIC is shown in Fig. 3(a). It can be observed that the magnitudes of P_0 and P_1 are almost the same: From (7), we obtain $|P_0|^2 = |P_1|^2 + 1$.

The angles of the USR CORDIC are

$$\begin{aligned} \alpha_0 &= 0 \\ \alpha_1 &= \tan^{-1}\left(\frac{2^k}{2^{2k-1}}\right) = \tan^{-1}(2^{-k+1}). \end{aligned} \tag{8}$$

Table I shows a list of USR CORDIC rotators for different values of k. The table shows the coefficients P_0 and P_1 with their corresponding angles, the radius, and WL_E, calculated as explained in [2] and [14]. It can be observed that the first kernels have a small WL_E. However, for a large k, WL_E is large enough to use them as rotation stages.

The hardware implementation of the USR CORDIC rotator is shown in Fig. 3(b). The figure shows that the USR CORDIC rotators are implemented using only two adders, like the conventional CORDIC rotations.

C. Nanorotations

Nanorotations refer to the kernel formed by the coefficient set

$$P_k = C + jk, \qquad k = 0, \ldots, N \tag{9}$$

where C is constant and the corresponding angles are

$$\alpha_k = \tan^{-1}\left(\frac{k}{C}\right). \tag{10}$$

In (9), N is considered to be much smaller than C. This makes α_k small and fulfills $\alpha_k \approx \tan(\alpha_k)$. This leads to $\alpha_k \approx k/C$,

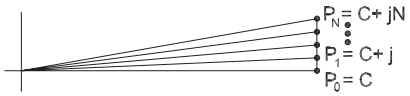

Fig. 4. Kernel to calculate nanorotations.

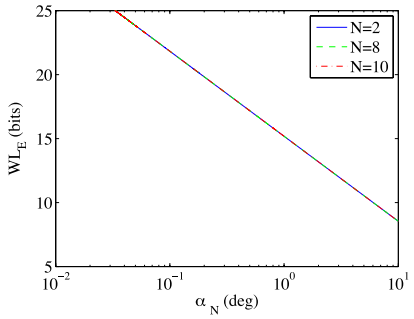

Fig. 5. Nanorotations: WL_E (bits) as a function of the angle α_N (deg).

which is a kernel with equally distributed angles. The fact that $N \ll C$ also makes the scaling of the coefficients very similar.

Fig. 4 shows the kernel to calculate nanorotations. It can be observed that the angle changes by simply changing the value of the imaginary part.

Fig. 5 shows the WL_E as a function of the largest angle of the kernel α_N, where

$$\alpha_N(\text{rad}) = \tan^{-1}\left(\frac{N}{C}\right) \approx \frac{N}{C}. \tag{11}$$

Note that WL_E only depends on α_N independently of the number of angles N. Fig. 5 shows that a WL_E larger than 15 bits is achieved for angles smaller than $\alpha_N = 1^\circ$. Thus, nanorotations will be used when $\alpha_N \leq 1^\circ$.

To design the rotator, the angle α_N must be selected first, according to the range of input angles. Then, N is selected. Finally, the constant C is obtained from (11).

IV. Connecting Rotation Stages

The CORDIC II algorithm consists of several rotation stages connected in series. Each rotation stage can be characterized by an input range $[-\alpha_{\text{in}}, \alpha_{\text{in}}]$ and an output range $[-\alpha_{\text{out}}, \alpha_{\text{out}}]$. For instance, the input angle for the CORDIC microrotation by 7.125° is in the range [−14.25°, 14.25°], and the output angle is in the range [−7.125°, 7.125°].

In general, a rotation stage may include any number of rotation angles. Each input is rotated by one of these angles. We define an N-rotator as a rotator with N different angles to choose from.

If N is even, the rotator includes $N/2$ coefficients and their conjugates. Fig. 6(a) shows this case. The values of δ_i are defined as

$$\begin{aligned} \delta_1 &= \alpha_1 \\ \delta_i &= \frac{(\alpha_i - \alpha_{i-1})}{2} \quad i = 2, \ldots, \frac{N}{2}. \end{aligned} \tag{12}$$

According to this, the input and output angles are

$$\alpha_{\text{out}} = \max_i(\delta_i) \quad i = 1, \ldots, \frac{N}{2} \tag{13}$$

$$\delta_{\frac{N}{2}+1} = \alpha_{\text{out}} \tag{14}$$

$$\alpha_{\text{in}} = \alpha_{\frac{N}{2}} + \delta_{\frac{N}{2}+1} = \alpha_{\frac{N}{2}} + \alpha_{\text{out}}. \tag{15}$$

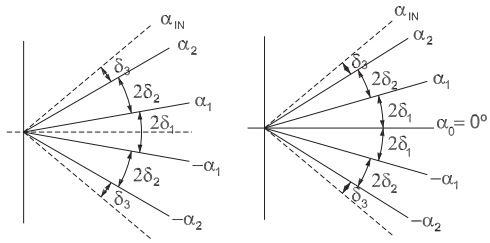

Fig. 6. Rotations of N-rotator. (a) N even ($N = 4$). (b) N odd ($N = 5$).

The best case happens when all of the values of δ_i are equal, i.e., $\delta_i = \phi$, where ϕ is a constant. This minimizes the remaining angle to the next stage α_{out}. Under these conditions, $\alpha_{\text{out}} = \phi = \alpha_{\text{in}}/N$, and the rotation angles are $\alpha_i = (2i-1)\alpha_{\text{in}}/N$.

If N is odd, the rotator includes $(N-1)/2$ coefficients with their conjugates plus the coefficient for $\alpha = 0^\circ$. Fig. 6(b) shows this case. The values of δ_i are defined as

$$\delta_i = \frac{(\alpha_i - \alpha_{i-1})}{2} \quad i = 1, \ldots, \frac{(N-1)}{2}. \tag{16}$$

According to this, the input and output angles are

$$\alpha_{\text{out}} = \max_i(\delta_i) \quad i = 1, \ldots, \frac{(N-1)}{2} \tag{17}$$

$$\delta_{\frac{(N+1)}{2}} = \alpha_{\text{out}} \tag{18}$$

$$\alpha_{\text{in}} = \alpha_{\frac{(N-1)}{2}} + \delta_{\frac{(N+1)}{2}} = \alpha_{\frac{(N-1)}{2}} + \alpha_{\text{out}}. \tag{19}$$

The best case also happens when all of the values of δ_i are equal, leading to $\alpha_{\text{out}} = \phi = \alpha_{\text{in}}/N$. In this case, the rotation angles are $\alpha_i = 2i\alpha_{\text{in}}/N$.

From this analysis, we can draw several conclusions. First, in the best cases when the rotation angles are selected carefully, an N-rotator reduces the input range by a factor N because $\alpha_{\text{out}} = \phi = \alpha_{\text{in}}/N$. This fact justifies the efficiency of the CORDIC rotator, where many of the microrotations halve the rotation angle using a 2-rotator. Second, in order to design efficient rotators, we have to aim to rotators for which $\alpha_{\text{out}} \approx \phi = \alpha_{\text{in}}/N$. Finally, in order to connect stages in series, for each stage, α_{in} must be larger than α_{out} of the previous stage. This guarantees the convergence of the rotation angle.

V. CORDIC II Algorithm

Fig. 7 shows the architecture of the CORDIC II rotator, and Table II includes detailed information about each rotation stage. The CORDIC II algorithm consists of six rotation stages in pipeline that use the angle sets described in previous sections.

Stage 1: The first stage calculates trivial rotations by ±180° and ±90° to set the remaining angle in the range of ±45°. The hardware architecture for the trivial rotator is shown in Fig. 8. It uses two negators, which are approximately equivalent to half an adder each, and four 2:1 multiplexers.

Stage 2: The second stage of the CORDIC II algorithm uses friend angles. It consists of the kernel $[25, 24 + j7, 20 + j15]$. The scale factor for all of the coefficients is $R = 25$, as $625 = 25^2 = 24^2 + 7^2 = 20^2 + 15^2$. Thus, there is no rotation error, and $WL_E = \infty$. The friend angles that correspond to

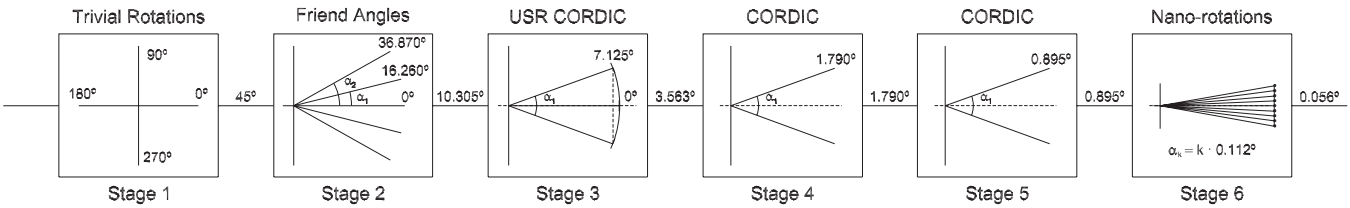

Fig. 7. Architecture of the CORDIC II rotator.

TABLE II
CORDIC II ROTATION STAGES

Stage	Rotator type	Micro-rotation		# Add.	# Mux.	WL_E	α_{in}	α_{out}	R_{norm}
1	Trivial rotations	$P_0 = 1$ $P_1 = j$ $P_2 = -1$ $P_3 = -j$	$\alpha_0 = 0°$ $\alpha_1 = 90°$ $\alpha_2 = 180°$ $\alpha_3 = 270°$	1	4	∞	$\pm 180°$	$\pm 45°$	1
2	Friend angles	$P_0 = 25$ $P_1 = 24 + j7$ $P_2 = 20 + j15$	$\alpha_0 = 0°$ $\alpha_1 = 16.260°$ $\alpha_2 = 36.870°$	5	7(+2)	∞	$\pm 47.175°$	$\pm 10.305°$	1.563
3	USR CORDIC	$P_0 = 129$ $P_1 = 128 + j16$	$\alpha_0 = 0°$ $\alpha_1 = 7.125°$	2	2(+2)	17.52	$\pm 10.688°$	$\pm 3.563°$	1.008
4	CORDIC	$P_1 = 32 + j$	$\alpha_1 = 1.790°$	2	0(+2)	∞	$\pm 3.580°$	$\pm 1.790°$	≈ 1
5	CORDIC	$P_1 = 64 + j$	$\alpha_1 = 0.895°$	2	0(+2)	∞	$\pm 1.790°$	$\pm 0.895°$	≈ 1
6	Nano-rotations	$P_k = 512 + jk$ $k = 0, \ldots, 8$	$\alpha_k = k \cdot 0.112°$	4	8(+2)	15.50	$\pm 0.895°$	$\pm 0.056°$	≈ 1
6 bis	CORDIC	$P_1 = 128 + j$	$\alpha_1 = 0.448°$	2	0(+2)	∞	$\pm 0.895°$	$\pm 0.448°$	≈ 1
7 bis	Nano-rotations	$P_k = 1024 + jk$ $k = 0, \ldots, 8$	$\alpha_k = k \cdot 0.056°$	4	8(+2)	17.50	$\pm 0.448°$	$\pm 0.028°$	≈ 1

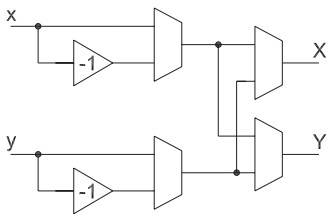

Fig. 8. Architecture of the trivial rotation (stage 1).

the coefficients are 0°, 16.260°, and 36.870°, with normalized scaling $R_{\text{norm}} = 1.563$, according to

$$R_{\text{norm}} = \frac{R}{2^{\lfloor \log_2 R \rfloor}}. \tag{20}$$

The hardware architecture for the friend angle stage is shown in Fig. 9. It consists of five adders and seven 2:1 multiplexers and can calculate all of the rotations of the kernel depending on the configuration of the multiplexers. In Table II, two additional (+2) multiplexers are needed between stages in order to rotate the entire kernel (positive and negative rotations), as in [11].

Stage 3: The third stage of the CORDIC II algorithm uses the USR CORDIC. It consists of the kernel $[129, 128 + j16]$, already shown in Table I. This stage reduces the remaining angle to $\pm 3.563°$. The hardware architecture for the USR CORDIC stage is as shown in Fig. 3(b) for $k = 4$. It consist of two adders and two 2:1 multiplexers.

Stages 4 and 5: The fourth and fifth stages of the CORDIC II use conventional CORDIC rotations by 1.790° and 0.895°.

Stage 6: The sixth stage uses nanorotations. The kernel used is $P_k = 512 + jk$, $k = 0, \ldots, 8$. The rotation angles of the kernel are $\alpha_k = k \cdot 0.112°$. The remaining angle of the CORDIC II is $\pm 0.056°$. The hardware circuit for the nanorotation stage

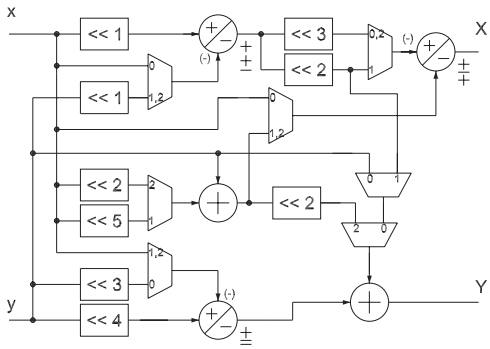

Fig. 9. Architecture of the friend angles (stage 2).

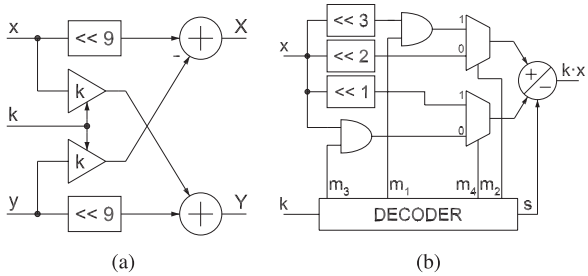

Fig. 10. Architecture of the nanorotator (stage 6). (a) Nanorotator for the angle set $P_k = 512 + jk$, $k = 0, \ldots, 8$. (b) Multiplication by k for the nanorotator, where $k \in \{0, \ldots, 8\}$.

is shown in Fig. 10. Fig. 10(a) shows the nanorotator, and Fig. 10(b) shows how the multiplication by k is implemented. The decoder in Fig. 10(b) consists of a few logic gates.

Stages 6 bis and 7 bis: An alternative to the sixth stage of the CORDIC II is to add one more CORDIC rotation (stage 6 bis) followed by a nanorotator (stage 7 bis), as shown in Table II.

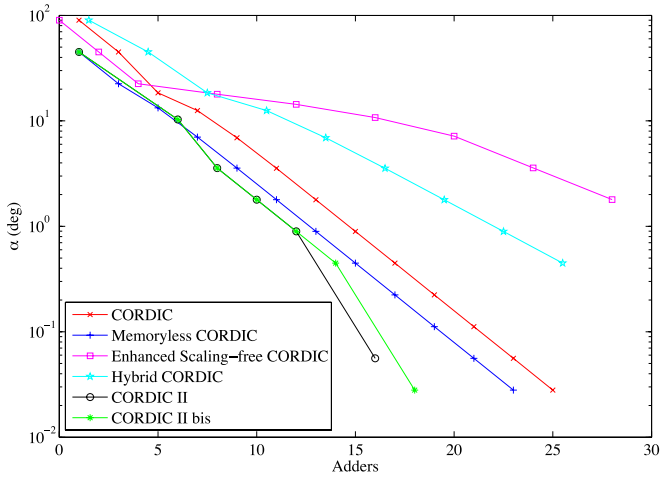

Fig. 11. Resolution of the rotators as a function of the number of adders.

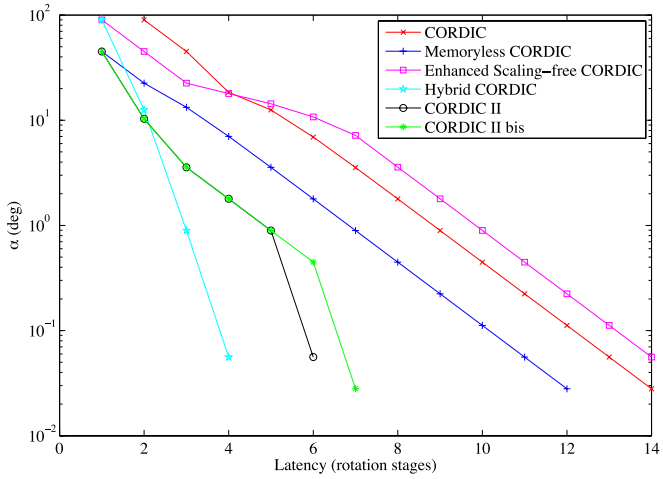

Fig. 12. Resolution of the rotators versus latency.

This increases the WL_E of the nanorotator and reduces the remaining angle of the CORDIC II bis to $\pm 0.028°$.

Note also that the CORDIC II provides convergence for the entire circumference, as α_{in} for each stage is larger than α_{out} of the previous stage.

Finally, the control logic is similar to [11]: By representing the angle in the range [0, 1], the first three bits determine the trivial rotations, stages 2 and 3 use comparators, and the control for the rest of stages is obtained directly by representing the angle proportionally to the minimum rotation angle.

VI. Comparison

Fig. 11 compares the number of adders as a function of the remaining angle for the CORDIC, memoryless CORDIC [11], enhanced scaling-free CORDIC [7], hybrid CORDIC [4], CORDIC II, and CORDIC II bis. For a precision of $\pm 0.056°$, the CORDIC II uses 16 adders. This represents a saving of 30.4% with respect to the CORDIC algorithm and 23.8% with respect to the memoryless CORDIC. For a precision of $\pm 0.028°$, the savings of the CORDIC II bis with respect to the CORDIC are 28%.

Fig. 12 shows the latency in terms of rotation stages. The proposed approaches reduce the latency of the CORDIC close to 50% and are only beaten by the hybrid CORDIC [4] at the cost of larger number of adders, as shown in Fig. 11.

Finally, we have obtained synthesis results for 65-nm ASIC technology. For a word length of 16 bits and aiming for $T_{\text{clk}} = 4$ ns, the CORDIC II occupies 7816 μm^2 at 261 MHz. For the same constraints, the CORDIC algorithm occupies 8599 μm^2 at 259 MHz. This represents savings of 10% in area of the CORDIC II with respect to the conventional CORDIC.

VII. Conclusion

The CORDIC II is a new algorithm that substitutes the CORDIC microrotation by a new angle set. This involves three new types of rotators: friend angles, USR CORDIC, and nanorotations. By using the proposed microrotations, the CORDIC II requires the minimum number of adders among CORDIC algorithms so far.

References

[1] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electron. Comput.*, vol. EC-8, no. 3, pp. 330–334, Sep. 1959.
[2] M. Garrido, F. Qureshi, and O. Gustafsson, "Low-complexity multiplierless constant rotators based on combined coefficient selection and shift-and-add implementation (CCSSI)," *IEEE Trans. Circuits Syst. I*, vol. 61, no. 7, pp. 2002–2012, Jul. 2014.
[3] 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.
[4] R. Shukla and K. Ray, "Low latency hybrid CORDIC algorithm," *IEEE Trans. Comput.*, vol. 63, no. 12, pp. 3066–3078, Dec. 2014.
[5] C.-S. Wu and A.-Y. Wu, "Modified vector rotational CORDIC (MVR-CORDIC) algorithm and architecture," *IEEE Trans. Circuits Syst. II*, vol. 48, no. 6, pp. 548–561, Jun. 2001.
[6] S. Aggarwal, P. K. Meher, and K. Khare, "Area-time efficient scaling-free CORDIC using generalized micro-rotation selection," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 8, pp. 1542–1546, Aug. 2012.
[7] F. Jaime, M. Sánchez, J. Hormigo, J. Villalba, and E. Zapata, "Enhanced scaling-free CORDIC," *IEEE Trans. Circuits Syst. I*, vol. 57, no. 7, pp. 1654–1662, Jul. 2010.
[8] Y. Liu, L. Fan, and T. Ma, "A modified CORDIC FPGA implementation for wave generation," *Circuits Syst. Signal Process.*, vol. 33, no. 1, pp. 321–329, Jan. 2014.
[9] C.-Y. Yu, S.-G. Chen, and J.-C. Chih, "Efficient CORDIC designs for multi-mode OFDM FFT," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, May 2006, vol. 3, pp. 1036–1039.
[10] C.-Y. Chen and C.-Y. Lin, "High-resolution architecture for CORDIC algorithm realization," in *Proc. Int. Conf. Commun. Circuits Syst.*, Jun. 2006, vol. 1, pp. 579–582.
[11] M. Garrido and J. Grajal, "Efficient memoryless CORDIC for FFT computation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Apr. 2007, vol. 2, pp. 113–116.
[12] R. Andraka, "A survey of CORDIC algorithms for FPGA based computers," in *Proc. ACM/SIGDA Int. Symp. FPGA*, Feb. 1998, pp. 191–200.
[13] 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*, vol. 56, no. 9, pp. 1893–1907, Sep. 2009.
[14] M. Garrido, O. Gustafsson, and J. Grajal, "Accurate rotations based on coefficient scaling," *IEEE Trans. Circuits Syst. II*, vol. 58, no. 10, pp. 662–666, Oct. 2011.
[15] N. Takagi, T. Asada, and S. Yajima, "Redundant CORDIC methods with a constant scale factor for sine and cosine computation," *IEEE Trans. Comput.*, vol. 40, no. 9, pp. 989–995, Sep. 1991.