Area-energy efficient CORDICs using new elementary-angle-set and base-2 exponent expansions scheme

Yong-sen Wang^{1, 2, a)}, Yu-qun Xue^{1, 2}, Heng You^{1, 2}, and Shu-shan Qiao^{1, 2, b)}

Abstract Area-Energy efficient CORDICs are presented. The proposed CORDICs adopt a new Elementary-Angle-Set, based on which the elementary angle composition of input angle can be directly determined without angle recoding, and the Angle-Set ensures the directions of vector micro-rotation are deterministic anti-clockwise in first quadrant. Based on the Angle-Set, elements sin and cos of rotation matrices are expanded into exponential expressions with base-2, which are convenient to circuit implementation. To improve calculation accuracy, initial values of rotation vectors are added with fixed offsets to pre-compensate the scaling-free expansions errors. Implemented in multiple FPGA devices and CMOS processes, the CORDICs achieve the lowest resource consumption, the least number of stages and the best energy-efficiency compared with other state-of-the-art works.

Keywords: CORDIC, scaling-free, elementary-angle-set, pipeline

Classification: Integrated circuits (memory, logic, analog, RF, sensor)

1. Introduction

COordinate Rotation DIgital Computer (CORDIC) is a kind of numerical approximation algorithm [1, 2, 3, 4]. Its operations only include shift and addition/subtraction, which is convenient to implement in VLSI [5, 6, 7]. It is widely used in the field of digital signal processing (DSP) [1, 2, 8, 9, 10, 11], such as trigonometric function calculation [12], Fast Fourier Transform (FFT) [13, 14], Discrete Cosine Transform (DCT) [15, 16], Direct Digital Synthesis (DDS) [17] and QR decomposition [18]. CORDIC can use pipeline structures to speed up data transmission [4]. In addition, it can be optimized for computing accuracy and speed according to application requirements, enabling it to be applied to low-power, low-complexity, high-throughput designs [9, 19].

Since CORDIC was proposed by Jack E. Volder in 1959, lots of progress has been made in the area of algorithm design and development of architectures [19, 20]. For conventional CORDIC, N -bits CORDIC typically requires $(N + 1)$ iterations [1, 21], which increases latency. In addition, the area and power consumption of CORDIC can be further op-

timized. Angle Recoding (AR) CORDIC can be used to reduce the number of iterations. But AR-CORDICs need additional encoders to pre-code the input angle as a linear combination of a set of selected elementary angles of micro-rotation [19, 22]. In addition, many AR-CORDICs' input angles are not arbitrary but even needs to be known in advance [23, 24, 25, 26]. High-Radix CORDIC can rotate multiple angles at a time but its rotation factor at each stage is a variable value which requires either a large ROM at the final compensation stage or sophisticated technique at each stage to compensate the vector [27, 28]. Conventional scaling-free (SF) CORDIC expands the elements in rotation matrix by Taylor series which ensures the length of vector remains a constant during rotation. But SF-CORDICs usually have narrow convergence range of $[0, \pi/8]$, which require more complex domain folding and post-processing [29, 30, 31]. In this paper a new Elementary-Angle-Set is adopted to determine the elementary angle composition of input angle without angle recoding. Rotation matrix elements also adopt low-complexity expansion scheme. FPGA and ASIC implementation results show the CORDICs features low resource consumption and power.

2. The proposed CORDIC algorithm

For rotation-mode circular-coordinate CORDIC, the rotational matrix for rotating a initialization vector $[X_0, Y_0]^T$ with target angle θ is given by Eq. (1):

$$\begin{bmatrix} X_N \\ Y_N \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix} \quad (1)$$

Where $[X_0, Y_0]^T$ and $[X_N, Y_N]^T$ are input and output vectors respectively for rotation angle θ . In conventional CORDIC, Elementary-Angle-Set (EAS) is $\theta_i = \arctan(2^{-i})$, where $i = 0, 1, 2, \dots, N - 1$. Input angle θ is decomposed as a combination of EAS: $\theta = \sum_{i=0}^{N-1} d_i \cdot \theta_i$, where $d_i \in \{-1, +1\}$ is the direction of i -th micro-rotation. Eq. (2) shows i -th micro-rotation and scaling factor $\cos(\theta_i) = 1/\sqrt{1 + 2^{-2i}}$:

$$\begin{aligned} \begin{bmatrix} X_{i+1} \\ Y_{i+1} \end{bmatrix} &= \begin{bmatrix} \cos(\theta_i) & -d_i \cdot \sin(\theta_i) \\ d_i \cdot \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \end{bmatrix} \\ &= \cos(\theta_i) \cdot \begin{bmatrix} 1 & -d_i \cdot 2^{-i} \\ d_i \cdot 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \end{bmatrix} \end{aligned} \quad (2)$$

2.1 The new elementary-angle-set

The New Elementary-Angle-Set (NEAS) directly adopts

¹ Institute of Microelectronics of Chinese Academy of Sciences, Beitucheng West Road, Chaoyang District, Beijing 100029, China

² University of Chinese Academy of Science, No. 19 (A) Yuquan Road, Shijingshan District, Beijing 100049, China

^{a)} wangyongsen@ime.ac.cn

^{b)} qiaoshushan@ime.ac.cn

$\theta_i = 2^{-i}$, where $i = 0, 1, 2, \dots, N-1$. This EAS gives CORDIC two advantages: 1) Elementary angle composition can be determined directly from input angle θ without angle recoding; 2) If input angle is within $[0, \pi/2]$, the vector's micro-rotation direction is deterministic anti-clockwise.

Since target angle θ also adopts binary representation, different from AR-CORDIC, NEAS can be used to directly determine which elementary angles the θ consists of. Take 16-bits input angle as an example, if θ is "0001110000000000", for Q3.13 fixed-point number format, θ is composed of elementary angles $2^{-1}, 2^{-2}, 2^{-3}$. In Table I, $\theta = 2^{-1} + 2^{-2} + 2^{-3}$, which means in order to get $[X_N, Y_N]^T, [X_0, Y_0]^T$ only needs to rotate by three angles $2^{-1}, 2^{-2}$ and 2^{-3} successively. Here, the unit of angle is radian.

For input angle θ within $[0, \pi/2]$, it is represented as $\theta[15:0]$ in hardware description language (HDL). Here, we introduce the marker $S_i \in \{0, 1\}$, $i = 0, 1, 2, \dots, 13$ to indicate that each bit of θ is 0 or 1, namely:

$$S_0 = \theta[13], S_1 = \theta[12], S_2 = \theta[11], \dots, S_{13} = \theta[0] \quad (3)$$

Therefore, θ in Table I can be expressed in a more uniform way as follows:

$$\begin{aligned} \theta &= \theta_0 + \theta_1 + \theta_2 + \dots + \theta_{12} + \theta_{13} \\ &= S_0 \cdot 2^0 + S_1 \cdot 2^{-1} + S_2 \cdot 2^{-2} + \dots \\ &\quad + S_{12} \cdot 2^{-12} + S_{13} \cdot 2^{-13} \end{aligned} \quad (4)$$

Where $S_1, S_2, S_3 = 1$ and other $S_i = 0$.

Table I Angle decomposition example of 16-bits input angle

Bit Position	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NEAS	$\pm$	2^{-1}	1	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}
Input Angle	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
Need to Rotate				2^{-1}	2^{-2}	2^{-3}

Fig. 1 Elementary angle composition of angle "0001110000000000".

Additionally, for target angle θ within $[0, \pi/2]$, since it can be divided into several positive elementary angles, the rotation direction of vector $[X_i, Y_i]^T$ must be anti-clockwise. Therefore, d_i in Eq. (2) can be eliminated, which means that comparators of each stage is not needed in circuits. Fig. 1 illustrates how initial vector $[X_0, Y_0]^T$ rotates by angle θ of Table I. The proposed CORDIC supports all angles in $[-\pi, \pi]$ and they can be folded to first quadrant using symmetry of coordinate system.

2.2 Scaling-free expansion of rotation matrix

By adopting NEAS, Eq. (2) can be rewritten as:

$$\begin{bmatrix} X_{i+1} \\ Y_{i+1} \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \end{bmatrix} \quad (5)$$

For elements $\cos(\theta_i)$ and $\sin(\theta_i)$ in rotation matrix, we adopt exponent expansion. The general expressions of $\cos(\theta_i)$ and $\sin(\theta_i)$ are defined as:

$$\sin(\theta_i) = K_{s0} + K_{s1} \cdot \theta_i + K_{s2} \cdot \theta_i^2 + K_{s3} \cdot \theta_i^3 + K_{s4} \cdot \theta_i^4 \quad (6a)$$

$$\cos(\theta_i) = K_{c0} + K_{c1} \cdot \theta_i + K_{c2} \cdot \theta_i^2 + K_{c3} \cdot \theta_i^3 + K_{c4} \cdot \theta_i^4 \quad (6b)$$

Where $(K_{s0}, K_{s1}, K_{s2}, K_{s3}, K_{s4})$ and $(K_{c0}, K_{c1}, K_{c2}, K_{c3}, K_{c4})$ are undetermined coefficients. According to the definition of NEAS in Section 2.1, $\theta_i = S_i \cdot 2^{-i}$, $S_i \in \{0, 1\}$, substitute θ_i into Eq. (6):

$$\begin{aligned} \sin(\theta_i) &= K_{s0} + K_{s1} \cdot (S_i \cdot 2^{-i}) + K_{s2} \cdot (S_i \cdot 2^{-i})^2 \\ &\quad + K_{s3} \cdot (S_i \cdot 2^{-i})^3 + K_{s4} \cdot (S_i \cdot 2^{-i})^4 \end{aligned} \quad (7a)$$

$$\begin{aligned} &= K_{s0} + K_{s1} \cdot S_i \cdot 2^{-i} + K_{s2} \cdot S_i \cdot 2^{-2i} \\ &\quad + K_{s3} \cdot S_i \cdot 2^{-3i} + K_{s4} \cdot S_i \cdot 2^{-4i} \end{aligned}$$

$$\begin{aligned} \cos(\theta_i) &= K_{c0} + K_{c1} \cdot (S_i \cdot 2^{-i}) + K_{c2} \cdot (S_i \cdot 2^{-i})^2 \\ &\quad + K_{c3} \cdot (S_i \cdot 2^{-i})^3 + K_{c4} \cdot (S_i \cdot 2^{-i})^4 \end{aligned} \quad (7b)$$

$$\begin{aligned} &= K_{c0} + K_{c1} \cdot S_i \cdot 2^{-i} + K_{c2} \cdot S_i \cdot 2^{-2i} \\ &\quad + K_{c3} \cdot S_i \cdot 2^{-3i} + K_{c4} \cdot S_i \cdot 2^{-4i} \end{aligned}$$

Since $S_i \in \{0, 1\}$, $(S_i)^N = S_i$. For Eq. (7), if $S_i = 0$, then the rotation matrix in Eq. (5) is identity matrix, and there is $[X_{i+1}, Y_{i+1}]^T = [X_i, Y_i]^T$. $2^{-i}, 2^{-2i}, 2^{-3i}$ and 2^{-4i} in above equations can be conveniently achieved by truncating X_i and Y_i . Furthermore, if undetermined coefficients $(K_{s0}, K_{s1}, K_{s2}, K_{s3}, K_{s4})$ and $(K_{c0}, K_{c1}, K_{c2}, K_{c3}, K_{c4})$ are also in exponent form with base-2, Eq. (7) will be very convenient for circuit implementation. Here we define that $(K_{s0}, K_{s1}, K_{s2}, K_{s3}, K_{s4})$ and $(K_{c0}, K_{c1}, K_{c2}, K_{c3}, K_{c4})$ belong to the following set:

$$\begin{aligned} &K_{s0}, K_{s1}, K_{s2}, K_{s3}, K_{s4}, K_{c0}, K_{c1}, K_{c2}, K_{c3}, K_{c4} \in \\ &\{0, \pm 2^{-32}, \pm 2^{-31}, \pm 2^{-30}, \pm 2^{-29}, \dots, \\ &\quad \pm 2^{-29}, \pm 2^{-30}, \pm 2^{-31}, \pm 2^{-32}\} \end{aligned} \quad (8)$$

The optimal values of undetermined coefficients are to minimize Mean Square Error (MSE) of $\cos(\theta_i)$ and $\sin(\theta_i)$ in Eq. (6) with the truth-values of sin and cos functions. Here we use the library functions sin and cos in Matlab as truth-functions, and in order to distinguish, use $\sin_T$ and $\cos_T$ to represent. Therefore, for angles 0 and 2^{-i} , search criterions for undetermined coefficients are:

$$\min \left\{ \frac{1}{2} [(\sin_T(0) - \sin(0))^2 + (\sin_T(2^{-i}) - \sin(2^{-i}))^2] \right\} \quad (9a)$$

$$\min \left\{ \frac{1}{2} [(\cos_T(0) - \cos(0))^2 + (\cos_T(2^{-i}) - \cos(2^{-i}))^2] \right\} \quad (9b)$$

Algorithm-1 is the pseudo-code implementation example of Eq. (9a). Each coefficient has 131 possible values, and

the optimal coefficient combination satisfying Eq. (9a) is searched from 131^5 different coefficient combinations. For different angles, the optimal coefficient combination is different. Specifically, taking angles $S_5 \cdot 2^{-5}$ as example, for angle $S_5 \cdot 2^{-5}$, the minimum MSE values of Eq. (9a) and (9b) that Algorithm-1 can find is $1.6\text{E-}15$ and $1.9\text{E-}19$, and the corresponding coefficient combination are $(K_{s0}, K_{s1}, K_{s2}, K_{s3}, K_{s4}) = (-2^{-25}, -2^{-13}, -2^{-10}, -2^{-7}, 2^{15})$ and $(K_{c0}, K_{c1}, K_{c2}, K_{c3}, K_{c4}) = (1, -2^{-6}, 2^{-15}, 2^{-12}, 2^{-9})$. Substitute these coefficients and $i = 5$ into Eq. (7), so Eq. (7) can be rewritten as:

$$\sin(\theta_5) = -2^{-25} - S_5 \cdot 2^{-18} - S_5 \cdot 2^{-20} - S_5 \cdot 2^{-22} + S_5 \cdot 2^{-5} \quad (10a)$$

$$\cos(\theta_5) = 1 - S_5 \cdot 2^{-11} + S_5 \cdot 2^{-25} + S_5 \cdot 2^{-27} + S_5 \cdot 2^{-29} \quad (10b)$$

In circuit implementation, exponent terms in Eq. (10) correspond to truncation of X_i or Y_i , so lower bit-width adders/subtractors can be used.

In summary, the proposed CORDIC algorithm does not need to determine the rotation direction at each micro-rotation, which eliminates comparators at each iteration. The expansion of rotation matrix elements is very hardware-friendly, vector $[X_i, Y_i]^T$ only needs to be truncated, and this corresponds to lower bit-width adders/subtractors. And for smaller micro-rotation angles and higher expansion orders, some terms can be discarded because the truncation results are 0.

Algorithm-1 : Searching the undetermined coefficients of Eq. (6a)

Initialization: $error=1; N=131;$

$K[N]=\{0, \pm 2^{-32}, \pm 2^{-31}, \pm 2^{-30}, \dots, \pm 2^{-30}, \pm 2^{-31}, \pm 2^{-32}\}$

```

01: for  $i_{s0}=0:N-1$ 
02:   for  $i_{s1}=0:N-1$ 
03:     for  $i_{s2}=0:N-1$ 
04:       for  $i_{s3}=0:N-1$ 
05:         for  $i_{s4}=0:N-1$ 
06:            $\sin_1 = K[i_{s0}] + K[i_{s1}] \times 0 + K[i_{s2}] \times 0 + K[i_{s3}] \times 0 + K[i_{s4}] \times 0;$ 
07:            $\sin_2 = K[i_{s0}] + K[i_{s1}] \times 2^{-4} + K[i_{s2}] \times 2^{-24} + K[i_{s3}] \times 2^{-34} + K[i_{s4}] \times 2^{-44};$ 
08:            $error\_tmp = [(sin_1(0) - \sin_1)^2 + (sin_1(2^{-4}) - \sin_2)^2] / 2;$ 
09:           if  $error\_tmp < error$ 
10:              $K_{s0} = K[i_{s0}]; K_{s1} = K[i_{s1}]; K_{s2} = K[i_{s2}]; K_{s3} = K[i_{s3}]; K_{s4} = K[i_{s4}];$ 
11:              $error = error\_tmp;$ 
12:           end
13:         end
14:       end
15:     end
16:   end
17: end

```

3. Hardware architectures

In this section, 32-bits and 16-bits pipeline CORDICs are proposed, and their stages are only 13 and 5, respectively. For 32-bits implementation, angle adopts Q3.29 fixed-point number format and X_i, Y_i adopt Q2.30 fixed-point number format. As shown in Fig. 2, Stage0 and Stage12 are pre-processing and post-processing modules. Pre-processing folds the angles in $[-\pi, \pi]$ to $[0, \pi/2]$, and post-processing folds the calculated X_N and Y_N back to original quadrant. Stage1 is used for initial values of CORDIC iteration, which are applied to pre-compensate the micro-rotation error of

latter stages. Stage1 will be explained later in this section. There are 3 types of micro-rotation from Stage2 to Stage11, namely, Type-I, Type-II and Type-III, which are illustrated below:

Stage2-Stage5: Stage2 to Stage5 correspond to micro-rotation Type-I. Taking Stage2 as an example, Fig. 3 (a) is its hardware circuit, which corresponds to Eq. (10). In actual implementation, the $S_5 \cdot 2^{-27}$ and $S_5 \cdot 2^{-29}$ terms in Eq. (10b) are omitted to reduce the delay. From Stage2 to Stage5, the structures are similar and addition/subtraction terms are selected by $S_i, i \in \{5, 6, 7, 8\}$ (although the truncation bit-width of each terms is different).

Fig. 2 32-bits pipeline CORDIC architecture.

Algorithm-2 : Searching the undetermined coefficients for rotation Type-II

Initialization: $error=1; N=131;$

$K[N]=\{0, \pm 2^{-32}, \pm 2^{-31}, \pm 2^{-30}, \dots, \pm 2^{-30}, \pm 2^{-31}, \pm 2^{-32}\}$

```

01: for  $i_{s0}=0:N-1$ 
02:   for  $i_{s1}=0:N-1$ 
03:     for  $i_{s2}=0:N-1$ 
04:        $\sin_1 = K[i_{s0}] + K[i_{s1}] \times 0 + K[i_{s2}] \times 0;$ 
05:        $\sin_2 = K[i_{s0}] + K[i_{s1}] \times (2^{-4}) + K[i_{s2}] \times (2^{-4})^2;$ 
06:        $\sin_3 = K[i_{s0}] + K[i_{s1}] \times (2^{-4}) + K[i_{s2}] \times (2^{-4})^2;$ 
07:        $\sin_4 = K[i_{s0}] + K[i_{s1}] \times (2^{-4} + 2^{-4}) + K[i_{s2}] \times (2^{-4} + 2^{-4})^2;$ 
08:        $error\_tmp = [(sin_1(0) - \sin_1)^2 + (sin_1(2^{-4}) - \sin_2)^2 + (sin_1(2^{-4}) - \sin_3)^2 + (sin_1(2^{-4} + 2^{-4}) - \sin_4)^2] / 4;$ 
09:       if  $error\_tmp < error$ 
10:          $K_{s0} = K[i_{s0}]; K_{s1} = K[i_{s1}]; K_{s2} = K[i_{s2}];$ 
11:          $error = error\_tmp;$ 
12:       end
13:     end
14:   end
15: end

```

Stage6-Stage8: Stage6 to Stage8 correspond to micro-rotation Type-II. Different from Type-I, where each micro-rotation corresponds to one rotation angle, Type-II corresponds to two rotation angles per rotation module. Take Stage6 as example, its rotation angle is:

$$\theta_{9-10} = S_9 \cdot 2^{-9} + S_{10} \cdot 2^{-10}, S_9, S_{10} \in \{0, 1\} \quad (11)$$

Type-II merges two angles for two reasons: 1) For smaller elementary angles 2^{-9} and 2^{-10} , if expansion method of Eq. (7) is still adopted, $K_{s3/c3} \cdot (2^{-9})^3, K_{s4/c4} \cdot (2^{-9})^4, K_{s3/c3} \cdot (2^{-10})^4$ and $K_{s4/c4} \cdot (2^{-10})^4$ terms are less than 2^{-31} , the truncation result is 0 even for 32-bits data; 2) Combining more angles in a micro-rotation module can significantly reduce the number of pipeline stages. So for rotation Type-II, cos and sin expand up to square terms. Since there

are 4 possible rotation angles $(0, 2^{-i}, 2^{-i-1}, 2^{-i} + 2^{-i-1})$, coefficient search in Algorithm-1 needs to be adjusted to Algorithm-2. Take *Stage6* as example, coefficient search results are $(K_{s0}, K_{s1}, K_{s2}) = (1, 2^{-30}, -2^{-1})$ and $(K_{c0}, K_{c1}, K_{c2}) = (0, 1, -2^{-11})$, and the terms less than 2^{-31} are eliminated. Expansion expressions of sin and cos are as follows:

$$\sin(\theta_{9-10}) = 1 - S_9 \cdot (1 + S_{10}) \cdot 2^{-19} - S_{10} \cdot 2^{-21} \quad (12a)$$

$$\cos(\theta_{9-10}) = S_9 \cdot 2^{-9} + S_{10} \cdot 2^{-10} - S_9 \cdot (1 + S_{10}) \cdot 2^{-29} \quad (12b)$$

Where $S_9 \cdot (1 + S_{10})$ is realized with 3:1 multiplexer. The hardware structure of Eq. (12) is shown in Fig. 3 (b).

Fig. 3 Different stages implementation examples of 32-bits CORDIC. (a) *Stage2* circuit structure. (b) *Stage6* circuit structure. (c) *Stage9* circuit structure.

Stage9-Stage11: *Stage9* to *Stage11* adopt micro-rotation Type-III where five angles are merged at a rotation module. Specifically, for *Stage9*, the rotation angle is:

$$\begin{aligned} \theta_{15-19} = & S_{15} \cdot 2^{-15} + S_{16} \cdot 2^{-16} + S_{17} \cdot 2^{-17} + S_{18} \cdot 2^{-18} \\ & + S_{19} \cdot 2^{-19}, S_{15}, S_{16}, S_{17}, S_{18}, S_{19} \in \{0, 1\} \end{aligned} \quad (13)$$

In addition, sin and cos are expanded to first power because $K_{s2/c2} \cdot (2^{-15})^2$ is less than 2^{-31} . The undetermined coefficient search is similar to Algorithm-2, but possible angle

combinations are extended to 2^5 . Fig. 3 (c) corresponds to *Stage9*, the expansions of sin and cos of *Stage9* is directly given:

$$\begin{aligned} \sin(\theta_{15-19}) = & S_{15} \cdot 2^{-15} + S_{16} \cdot 2^{-16} + S_{17} \cdot 2^{-17} \\ & + S_{18} \cdot 2^{-18} + S_{19} \cdot 2^{-19} \end{aligned} \quad (14a)$$

$$\cos(\theta_{15-19}) = 1 \quad (14b)$$

Stage1: *Stage1* are two look-up-tables (LUTs) with size of 26, which are used as pre-compensation for subsequent micro-rotation stages errors. Like other scaling-free CORDICs, the proposed CORDIC has three error sources: quantization error, truncation error and approximation error. *Stage1* corrects the error by pre-biasing the initial rotation vectors. The output vector $[X_5, Y_5]^T$ of *Stage1* is not obtained by micro-rotation, but by LUTs. The corresponding angle of *Stage1* is as follows:

$$\begin{aligned} \theta_{0-4} = & S_0 + S_1 \cdot 2^{-1} + S_2 \cdot 2^{-2} + S_3 \cdot 2^{-3} + S_4 \cdot 2^{-4}, \\ & S_0, S_1, S_2, S_3, S_4 \in \{0, 1\} \end{aligned} \quad (15)$$

Theoretically, $[X_5, Y_5]^T$ should be $[\cos(\theta_{0-4}), \sin(\theta_{0-4})]^T$, but since bias terms need to be added to correct the error, the vector is adjusted to:

$$[X_5, Y_5]^T = [\cos(\theta_{0-4}) + \text{offset}_X, \sin(\theta_{0-4}) + \text{offset}_Y]^T \quad (16)$$

The optimal value of term $\text{offset}_{X/Y}$ is determined by performing exhaustive search. Set the range of $\text{offset}_{X/Y}$ as $[-1, 1]$, search for $[\text{offset}_X, \text{offset}_Y]^T$ in this range to get the optimal $[X_5, Y_5]^T$, so that the MSE of output $[X_{out}, Y_{out}]^T$ is minimized. Here Table II gives the optimal $[X_5, Y_5]^T$ corresponding to different angle combinations under 32-bits implementation:

Table II The optimal $[X_5, Y_5]^T$ corresponding to different angle combinations under 32-bits implementation

$[S_0 S_1 S_2 S_3 S_4]$	$[X_5, Y_5]^T$	$[S_0 S_1 S_2 S_3 S_4]$	$[X_5, Y_5]^T$	$[S_0 S_1 S_2 S_3 S_4]$	$[X_5, Y_5]^T$
[00000]	[1.000, 0.000] ^T	[01001]	[0.846, 0.533] ^T	[10010]	[0.431, 0.902] ^T
[00001]	[0.998, 0.063] ^T	[01010]	[0.811, 0.585] ^T	[10011]	[0.374, 0.927] ^T
[00010]	[0.992, 0.125] ^T	[01011]	[0.773, 0.635] ^T	[10100]	[0.315, 0.949] ^T
[00011]	[0.983, 0.186] ^T	[01100]	[0.732, 0.682] ^T	[10101]	[0.255, 0.967] ^T
[00100]	[0.969, 0.247] ^T	[01101]	[0.688, 0.726] ^T	[10110]	[0.195, 0.981] ^T
[00101]	[0.952, 0.307] ^T	[01110]	[0.641, 0.768] ^T	[10111]	[0.133, 0.991] ^T
[00110]	[0.931, 0.366] ^T	[01111]	[0.592, 0.806] ^T	[11000]	[0.071, 0.998] ^T
[00111]	[0.906, 0.424] ^T	[10000]	[0.540, 0.842] ^T	[11001]	[0.008, 1.000] ^T
[01000]	[0.878, 0.479] ^T	[10001]	[0.487, 0.874] ^T		

The 16-bits pipeline architecture has 5 stages. Like 32-bits implementation, *Stage0* and *Stage4* are pre-processing and post-processing modules, and *Stage1* are initial values LUTs for rotation vector. Therefore, 16-bits architecture has only two micro-rotation stages (*Stage2*, *Stage3*), and the corresponding rotation angles are:

$$\theta_{5-7} = S_5 \cdot 2^{-5} + S_6 \cdot 2^{-6} + S_7 \cdot 2^{-7}, S_5, S_6, S_7 \in \{0, 1\} \quad (17)$$

$$\begin{aligned} \theta_{8-13} = & S_8 \cdot 2^{-8} + S_9 \cdot 2^{-9} + S_{10} \cdot 2^{-10} + S_{11} \cdot 2^{-11} \\ & + S_{12} \cdot 2^{-12} + S_{13} \cdot 2^{-13}, \\ & S_8, S_9, S_{10}, S_{11}, S_{12}, S_{13} \in \{0, 1\} \end{aligned} \quad (18)$$

Like 32-bits architecture, the elements sin and cos of rotation matrix are obtained by solving the undetermined coefficients. Rotation module is also realized by truncation, adders/subtractors and multiplexers.

4. Error analysis

Fig. 4 illustrates the 32/16-bits cos and sin functions and their absolute error curves in first quadrant. Fig. 4 (a), (c), (e) and (g) are functions curves of our proposed 32/16-bits CORDICs. Absolute error curves are the errors between our proposed CORDICs and Matlab floating-point library functions. It can be seen from Fig. 4 that due to pre-biasing scheme of *Stage1* and high-precision expansion of subsequent stages, 32-bits CORDIC error is on the level of 10^{-8} and 16-bits CORDIC is 10^{-4} .

Fig. 4 32/16-bits cos and sin functions and their absolute error curves. (a) cos function curve (32-bits). (b) cos function absolute error (32-bits). (c) sin function curve (32-bits). (d) sin function absolute error (32-bits). (e) cos function curve (16-bits). (f) cos function absolute error (16-bits). (g) sin function curve (16-bits). (h) sin function absolute error (16-bits).

5. Experiment results

In this section the proposed CORDICs are synthesized in multiple FPGA devices and 65/130nm CMOS process. In addition, the state-of-the-art CORDICs with pipeline architectures, FPGA or ASIC implementation results and arbitrary input angles are also selected. From Table III to Table IV, the proposed CORDICs adopt FPGA devices consistent with the references, and the data bit-width and error level are also as close as possible to the references. ASIC power results are obtained by Synopsys Prime Time using synthesized netlists and Switching Activity Interchange Format files. For ASIC implementation, since the power consumption is related to frequency, the energy index (unit: pJ/cycle) in [22] is adopted. For Tables III and IV, the proposed CORDIC's error is 10^{-8} level, which is consistent

with [12] (10^{-8}) and better than [4] (10^{-7}). Also, FPGA resource occupation, SDP, ADP, gate counts and Energy of the proposed CORDICs are better than [12] and [4]. For Tables V and VI, the error of [3], [29] and our CORDICs are all 10^{-4} level, which are better than [10] (10^{-3}). The MSE of [22] and our 16-bits CORDICs are all 10^{-8} level. As shown in Tables V and VI, the proposed CORDICs also achieve the lowest resource consumption and the best energy efficiency.

Table III FPGA synthesis results and comparison of 32-bits version

Reference	[12]	[4]	This work	
Scheme	Modified/Original	2D	NEAS+SF	
FPGA device	XC3S500E	XC5VLX30	XC3S500E	XC5VLX30
Bits	32	30	32	32
Stages	32	24	13	13
Registers	2938/2969	—	772	769
LUTs	2986/3018	2543	2205	2145
Slices	1520/1535	—	1156	591
Frequency[MHz]	40	24	73.568	153.337
SDP*	38.0/38.4	—	15.7	3.85
ADP**	73.5/74.2	—	10.5	5.02

* Slice-Delay-Product (SDP)=Slices/Frequency

** Area-Delay-Product (ADP)=Registers/Frequency

Table IV ASIC synthesis results and comparison of 32-bits version

Reference	[4]	This work	
Scheme	2D	NEAS+SF	
Tech. [nm]	130	65	130
Bits	30	32	32
Stages	24	13	13
Area[μm^2]	—	23364	114448
Gate Counts	45000	10563	10131
Power[mW]	360 (Peak Power)	0.87	1.57
Frequency[MHz]	67	250	250
Energy[pJ/cycle]	5373	3.48	6.28

* Energy=Power/Frequency [pJ/cycle]

Table V FPGA synthesis results and comparison of 16-bits version

Reference	[10]	This work
Scheme	RRMC	NEAS+SF
FPGA device	XC5VLX50	XC5VLX50
Stages	7	5
Bits	16	16
Registers	205	125
LUTs	1301	266
Slices	—	96
Frequency[MHz]	202	187.303
SDP	—	0.51
ADP	1.01	0.67

Table VI ASIC synthesis results and comparison of 16-bits version

Reference	[22]	[9]	[3]	[29]	This work	
Scheme	ARD+SCFE	Conventional	CORDIC II	SFB	NEAS+SF	
Tech. [nm]	180	45	65	65	65	130
Bits	18	16	16	16	16	16
Stages	10	16	6	11	5	5
Area[μm^2]	—	—	7816	14676	2853	11748
Gate Counts	10720	—	—	—	1155	1099
Power[mW]	12.96	29.4	—	1.26	0.37	0.55
Frequency[MHz]	100	1000	261	100	250	250
Energy[pJ/cycle]	129.6	29.4	—	12.6	1.48	2.2

6. Conclusion

The proposed CORDICs adopt a new redefined Elementary-

Angle-Set which can be applied to determine the elementary angle composition of input angle. Also, micro-rotation of the vector in first quadrant is deterministic anti-clockwise. Rotation matrix elements are expanded with base-2 exponent, which corresponds to truncation processing of data and lower bit-width adders/subtractors. Implemented in multiple FPGA devices and CMOS processes, the CORDICs feature low resource consumption and power.

References

- [1] J.E. Volder, *et al.*: “The CORDIC trigonometric computing technique,” *IRE Trans. Electronic Computers* **EC-8** (1959) 330 (DOI: [10.1109/TEC.1959.5222693](https://doi.org/10.1109/TEC.1959.5222693)).
- [2] L. Chen, *et al.*: “Algorithm and design of a fully parallel approximate coordinate rotation digital computer (CORDIC),” *IEEE Trans. Multi-Scale Comput. Syst.* **3** (2017) 139 (DOI: [10.1109/TMSSC.2017.2696003](https://doi.org/10.1109/TMSSC.2017.2696003)).
- [3] M. Garrido, *et al.*: “CORDIC II: A new improved CORDIC algorithm,” *IEEE Trans. Circuits Syst. II, Exp. Briefs* **63** (2016) 186 (DOI: [10.1109/TCSII.2015.2483422](https://doi.org/10.1109/TCSII.2015.2483422)).
- [4] E. Antelo, *et al.*: “A low-latency pipelined 2D and 3D CORDIC processors,” *IEEE Trans. Comput.* **57** (2008) 404 (DOI: [10.1109/TC.2007.70796](https://doi.org/10.1109/TC.2007.70796)).
- [5] M.G.B. Sumanasena, *et al.*: “A scale factor correction scheme for the CORDIC algorithm,” *IEEE Trans. Comput.* **57** (2008) 1148 (DOI: [10.1109/TC.2008.41](https://doi.org/10.1109/TC.2008.41)).
- [6] R. Shukla, *et al.*: “Low latency hybrid CORDIC algorithm,” *IEEE Trans. Comput.* **63** (2014) 3066 (DOI: [10.1109/TC.2013.173](https://doi.org/10.1109/TC.2013.173)).
- [7] T.Y. Sung, *et al.*: “Design and simulation of reusable IP CORDIC core for special-purpose processors,” *IET Computers & Digital Techniques* **1** (2007) 581 (DOI: [10.1049/iet-cdt:20060075](https://doi.org/10.1049/iet-cdt:20060075)).
- [8] L. Vachhani, *et al.*: “Efficient FPGA realization of CORDIC with application to robotic exploration,” *IEEE Trans. Ind. Electron.* **56** (2009) 4915 (DOI: [10.1109/TIE.2009.2026225](https://doi.org/10.1109/TIE.2009.2026225)).
- [9] A.N. Sapper, *et al.*: “Exploring the combination of number of bits and number of iterations for a power-efficient fixed-point CORDIC implementation,” *IEEE International Conference on Electronics, Circuits and Systems (ICECS)* (2017) 302 (DOI: [10.1109/ICECS.2017.8292079](https://doi.org/10.1109/ICECS.2017.8292079)).
- [10] S. Aggarwal, *et al.*: “Concept, design, and implementation of reconfigurable CORDIC,” *IEEE Trans. Very Large Scale Integr. Syst.* **24** (2016) 1588 (DOI: [10.1109/TVLSI.2015.2445855](https://doi.org/10.1109/TVLSI.2015.2445855)).
- [11] K.C. Ray, *et al.*: “CORDIC-based unified VLSI architecture for implementing window functions for real time spectral analysis,” *IEEE Proceedings - Circuits, Devices and Systems* **153** (2006) 539 (DOI: [10.1049/ip-cds:20050280](https://doi.org/10.1049/ip-cds:20050280)).
- [12] Y. Liu, *et al.*: “A modified CORDIC FPGA implementation for wave generation,” *Circuits, Systems, and Signal Processing* **33** (2014) 321 (DOI: [10.1007/s00034-013-9638-8](https://doi.org/10.1007/s00034-013-9638-8)).
- [13] H. Mahdavi, *et al.*: “Area-time-power efficient FFT architectures based on binary-signed-digit CORDIC,” *IEEE Trans. Circuits Syst. I, Reg. Papers* **66** (2019) 3874 (DOI: [10.1109/TCSI.2019.2922988](https://doi.org/10.1109/TCSI.2019.2922988)).
- [14] J. Chen, *et al.*: “Configurable floating-point FFT accelerator on FPGA based multiple-rotation CORDIC,” *Chinese Journal of Electronics* **25** (2016) 1063 (DOI: [10.1049/cje.2016.08.002](https://doi.org/10.1049/cje.2016.08.002)).
- [15] H. Huang, *et al.*: “CORDIC based fast Radix-2 DCT algorithm,” *IEEE Signal Process. Lett.* **20** (2013) 483 (DOI: [10.1109/LSP.2013.2252616](https://doi.org/10.1109/LSP.2013.2252616)).
- [16] M. Lee, *et al.*: “Reconfigurable CORDIC-based low-power DCT architecture based on data priority,” *IEEE Trans. Very Large Scale Integr. Syst.* **22** (2014) 1060 (DOI: [10.1109/TVLSI.2013.2263232](https://doi.org/10.1109/TVLSI.2013.2263232)).
- [17] D. De Caro, *et al.*: “A 380 MHz direct digital synthesizer/mixer with hybrid CORDIC architecture in 0.25 μm CMOS,” *IEEE J. Solid-State Circuits* **42** (2007) 151 (DOI: [10.1109/JSSC.2006.886527](https://doi.org/10.1109/JSSC.2006.886527)).
- [18] H. Lee, *et al.*: “Efficient low-latency implementation of CORDIC-based sorted QR decomposition for multi-Gbps MIMO systems,” *IEEE Trans. Circuits Syst. II, Exp. Briefs* **65** (2018) 1375 (DOI: [10.1109/TCSII.2018.2853099](https://doi.org/10.1109/TCSII.2018.2853099)).
- [19] P.K. Meher, *et al.*: “50 years of CORDIC: algorithms, architectures, and applications,” *IEEE Trans. Circuits Syst. I, Reg. Papers* **56** (2009) 1893 (DOI: [10.1109/tcsi.2009.2025803](https://doi.org/10.1109/tcsi.2009.2025803)).
- [20] J. Duprat, *et al.*: “The CORDIC algorithm: new results for fast VLSI implementation,” *IEEE Trans. Comput.* **42** (1993) 168 (DOI: [10.1109/12.204786](https://doi.org/10.1109/12.204786)).
- [21] Y.H. Hu, *et al.*: “CORDIC-based VLSI architectures for digital signal processing,” *IEEE Signal Process. Mag.* **9** (1992) 16 (DOI: [10.1109/79.143467](https://doi.org/10.1109/79.143467)).
- [22] H.T. Nguyen, *et al.*: “A high-throughput low-energy arithmetic processor,” *IEICE Trans. Electron.* **E101-C** (2018) 281 (DOI: [10.1587/transele.E101.C.281](https://doi.org/10.1587/transele.E101.C.281)).
- [23] P.K. Meher, *et al.*: “CORDIC designs for fixed angle of rotation,” *IEEE Trans. Very Large Scale Integr. Syst.* **21** (2013) 217 (DOI: [10.1109/TVLSI.2012.2187080](https://doi.org/10.1109/TVLSI.2012.2187080)).
- [24] R. Ramadoss, *et al.*: “Reliable hardware architectures of the CORDIC algorithm with a fixed angle of rotations,” *IEEE Trans. Circuits Syst. II, Exp. Briefs* **64** (2017) 972 (DOI: [10.1109/TCSII.2016.2624508](https://doi.org/10.1109/TCSII.2016.2624508)).
- [25] C.S. Wu, *et al.*: “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, Analog Digit. Signal Process.* **50** (2003) 589 (DOI: [10.1109/TCSII.2003.816923](https://doi.org/10.1109/TCSII.2003.816923)).
- [26] Y.H. Hu, *et al.*: “An angle recoding method for CORDIC algorithm implementation,” *IEEE Trans. Comput.* **42** (1993) 99 (DOI: [10.1109/12.192217](https://doi.org/10.1109/12.192217)).
- [27] B. Lakshmi, *et al.*: “VLSI architecture for parallel radix-4 CORDIC,” *Microprocessors & Microsystems* **37** (2013) 79 (DOI: [10.1016/j.micpro.2012.12.001](https://doi.org/10.1016/j.micpro.2012.12.001)).
- [28] A. Changela, *et al.*: “FPGA implementation of high-performance, resource-efficient Radix-16 CORDIC rotator based FFT algorithm,” *Integration the VLSI Journal* **73** (2020) 89 (DOI: [10.1016/j.vlsi.2020.03.008](https://doi.org/10.1016/j.vlsi.2020.03.008)).
- [29] F.J. Jaime, *et al.*: “Enhanced scaling-free CORDIC,” *IEEE Trans. Circuits Syst. I, Reg. Papers* **57** (2010) 1654 (DOI: [10.1109/TCSI.2009.2037391](https://doi.org/10.1109/TCSI.2009.2037391)).
- [30] S. Aggarwal, *et al.*: “Area-time efficient scaling-free CORDIC using generalized micro-rotation selection,” *IEEE Trans. Very Large Scale Integr. Syst.* **20** (2012) 1542 (DOI: [10.1109/TVLSI.2011.2158459](https://doi.org/10.1109/TVLSI.2011.2158459)).
- [31] K. Maharatna, *et al.*: “Modified virtually scaling-free adaptive CORDIC rotator algorithm and architecture,” *IEEE Trans. Circuits Syst. Video Technol.* **15** (2005) 1463 (DOI: [10.1109/TCSVT.2005.856908](https://doi.org/10.1109/TCSVT.2005.856908)).