Mixed-Scaling-Rotation CORDIC (MSR-CORDIC) Algorithm and Architecture for High-Performance Vector Rotational DSP Applications

Chih-Hsiu Lin and An-Yeu Wu

Abstract—**The coordinate rotational digital computer (CORDIC) algorithm is a well-known iterative arithmetic for performing vector rotations in many digital signal processing (DSP) applications. However, the large number of iteration is a major disadvantage of this algorithm for its speed performance. Many researchers have proposed schemes to reduce the number of iterations. Nevertheless, in performing the existing CORDIC algorithms, the norm of the vector is usually enlarged so that extra scaling operations are required to deliver the normalized output. In this paper, we merge the two operation phases (microrotations and scaling phases) and propose a new vector rotational scheme called *mixed-scaling-rotation coordinate rotational digital computer* (MSR-CORDIC) algorithm. It can eliminate the overhead of the scaling operations that are inevitable in existing CORDIC algorithms; hence, it can significantly reduce the total iteration number so as to improve the speed performance. The proposed MSR-CORDIC can be applied to DSP applications, in which the rotational angles are known in advance [e.g., twiddle factor in fast Fourier transform (FFT) processor designs]. Moreover, most CORDIC algorithms generally suffer from the roundoff noise in the fixed-wordlength implementations. We also propose two schemes to control and reduce the impairment. Our simulation results show that the MSR-CORDIC algorithm can enhance the *signal-to-quantization-noise ratio* (SQNR) performance by controlling the internal dynamic range. We also investigate the first- and second-order statistical properties, including the mean and variance of the SQNR. Simulation results show that the MSR-CORDIC can enhance SQNR performance of both first- and second-order statistical properties. At the VLSI architecture level, we proposed a generalized MSR-CORDIC engine for the tradeoff between hardware complexity and quantization error performance. It can further reduce the hardware complexity when compared with the newly proposed extend elementary angle set CORDIC algorithm [5]. The MSR-CORDIC scheme has been applied to a variable-length FFT processor design [29], and results in significant hardware reduction in implementing the twiddle factor operations.**

Index Terms—**Coordinate rotational digital computer (CORDIC), extend elementary angle set (EEAS)-CORDIC, twiddle factor, fast Fourier transform (FFT).**

I. Introduction

THE coordinate rotational digital computer (CORDIC) algorithm is a well-known hardware-efficient iterative algorithm for the computation of elementary arithmetic functions, such as trigonometric, hyperbolic, exponential, and logarithmic operations [1]. The CORDIC algorithm can also be applied to the rotation-based arithmetic functions, such as fast Fourier transformation (FFT) [7], QRD-RLS filtering [10], [11], eigenvalue decomposition (EVD) [9], and singular value decomposition (SVD) [8].

The basic CORDIC algorithm is carried out only by a sequence of shift-and-add operations. Despite its simplicity, it encounters the disadvantage of large number of iterations, which impedes the speed performance in practical implementations. The major computational time is to reduce the carry-propagate delay in each iteration. A Radix-2 redundant signed-digit adders (SDAs) are employed to alleviate the inherent carry-delay [8]. Another solution to solve such a problem is to *reduce the iteration number*. Some approaches are based on this concept. For example, a table-lookup-based scheme was proposed in [2]. It makes use of the elementary-angle-recoding to accelerate the convergence rate of the rotation angle. The higher radix number representation, e.g., a Radix-4 and very-high radix algorithms, is also dedicated to reduce the iteration numbers [19], [20], [26], [27]. In [5], the authors proposed the design concept of the extend elementary angle set (EEAS)-based CORDIC algorithm. It defines and extends the elementary angle set to reduce the number of iteration significantly. They also proposed the schemes to determine the optimal iteration numbers of rotation and scaling operation under a fixed-number iteration. Compared with existing CORDIC approaches, the EEAS-CORDIC can achieve the same performance but with much smaller iteration number. The fixed-point property of the EEAS-CORDIC is also analyzed in [30].

Aiming at reducing the iteration number, in this paper, we reformulate the iteration function to enhance the CORDIC algorithm. The advantages of our proposed mixed-scaling-rotation CORDIC (MSR-CORDIC) algorithm are as follows.

- *Enhanced Extend Elementary Angle Set (Enhanced EEAS):* We enhance the EEAS of [5] to an even larger angle set, which can be used to compose of the target angle. That is, the targeted rotational angles can be achieved with a very small number of iteration due to the more feasible elementary angle set in our algorithm.
- *Mixed Scaling and Rotation Operations:* In the conventional CORDIC algorithm, the norm is usually larger than 1. However, the norm of the rotated vector in the MSR-CORDIC can be smaller than 1 in each iteration. Therefore, we can merge the Scaling and Rotation Operations to eliminate the scaling operations. Without the overhead of the scaling operation, we can not only reduce the number of iterations but also speed up the computation of the CORDIC operations.

Manuscript received February 24, 2004; revised June 23, 2004. This paper was recommended by Associate Editor T. Arslan.

The authors are with the Graduate Institute of Electronics Engineering, Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan, R.O.C. (e-mail: fil@access.ee.ntu.edu.tw).

Digital Object Identifier 10.1109/TCSI.2005.853908

- *Universal Vector Rotational CORDIC Engine:* The MSR-CORDIC can be seen as a universal vector rotational CORDIC engine. The AR CORDIC [18], fast CORDIC [25], MVR-CORDIC [3], and EEAS-CORDIC [5] can be considered as subsets of the proposed MSR-CORDIC. At the architecture level, we propose a generalized MSR-CORIDC to achieve better signal-to-quantization-noise ratio (SQNR) performance at only small hardware overhead. That is, we can tradeoff between the SQNR performance and hardware complexity.
- *Reduction of Roundoff Noise:* In the conventional CORDIC algorithms, rotation operations are performed to approximate the target rotation angle. Then, scaling operations are executed to minimize the norm error between the initial and final vectors. From the viewpoint of the dynamic range and fixed-point properties, it is disadvantageous to enlarge the norm repeatedly. In the MSR-CORDIC, the norm of the input vector can be either scaled down or enlarged. Due to this interesting feature, the variation of the norm can be adjusted and guaranteed to have a very small dynamic range throughout all the rotation procedure. Two schemes are proposed in this paper to control and alleviate the impairment of dynamic range. Furthermore, minimizing the SQNR values in the two separate phases (rotation and scaling) are usually processed independently in conventional CORDIC algorithms. Hence, the solution will be local, but not global. In the proposed MSR-CORDIC, we aim at searching for a global solution so that merge both the angle and norm error can be minimized together.
- *Smaller Variation of SQNR:* The mean of SQNR is a good index to compare the error performance of the vector rotational operations. The SQNR value indicates the first-order statistical property of the error distribution. On the order hand, the variance of SQNR, the second order statistics, can help to see the variation of SQNR in achieving the target rotational angles. Intuitively, it is better that the variation of quantization error is smaller. Simulation results show that our design can enhance SQNR performance in terms of both first- and second-order statistical properties.

The proposed MSR-CORDIC is very suitable for a variety of DSP applications such as digital lattice filter [12]–[14], and discrete trigonometer transforms [15]–[17]. The major feature of these applications is that all the rotation angles are fixed and known in advance, so that we can calculate the set of control parameters corresponding to each angle in advance and store them in the ROM. By using the proposed MSR-CORDIC algorithm, the vector rotation can be accomplished with only 20 adders/subtractors of 16 bit wordlength, whereas the SQNR performance in an average sense is as high as 90.0 dB. Compared with the AR technique in [18] and EEAS approach in [5], we only need 44.6% and 72.4% of hardware complexities, respectively, to achieve the similar SQNR performance. The proposed MSR-CORDIC is very suitable for performing the twiddle Factor function of the FFF processor, and it has been demonstrated in [29].

The rest of the paper is organized as follows. We have a brief review of the CORDIC algorithm in Section II. We propose the MSR-CORDIC algorithm in Section III and discuss the issue of the error performance and roundoff noise in Section IV. We also propose two schemes to reduce the impairment. The RDP strategy is applied to the tradeoff between hardware consumption and SQNR performance. In SectionV, we make some system performance comparisons and establish the relationship between the signal dynamic range and error performance. Section VI shows the iterative and parallel system architectures. Finally, Section VII concludes our work.

II. Review of Related CORDIC Algorithms

A. Conventional CORDIC Algorithm

Given a rotation angle and input vector $[x, y]^T$, the target vector $[x' \;\; y']^T$ can be computed as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \tag{1}$$

Using the concept of microrotation in [1]–[3], we can decompose the target rotation angle into predefined elementary angles. Hence, (1) can be rewritten as

$$\begin{aligned} \begin{bmatrix} x' \\ y' \end{bmatrix} &= \left(\prod_{n=0}^{N-1} \begin{bmatrix} \cos\theta_n & -\mu_n \sin\theta_n \\ \mu_n \sin\theta_n & \cos\theta_n \end{bmatrix} \right) \begin{bmatrix} x \\ y \end{bmatrix} \\ &= \left(\prod_{n=0}^{N-1} \cos\theta_n \right) \left(\prod_{n=0}^{N-1} \begin{bmatrix} 1 & -\mu_n \tan\theta_n \\ \mu_n \tan\theta_n & 1 \end{bmatrix} \right) \\ &\quad \times \begin{bmatrix} x \\ y \end{bmatrix} \end{aligned} \tag{2}$$

$$\Theta = \sum_{n=0}^{N-1} \mu_n \theta_n + \zeta \tag{3}$$

where Θ is the expected rotation angle; N is the number of rotations; $\theta_n \triangleq \tan^{-1}(2^{-n}), n = 1, 2, \ldots, N-1$ denotes the elementary angle (or microangle); $\mu_n \in \{-1, 1\}$ indicates the direction of rotation for $n = 0, \ldots, N-1$; and ζ is the residual angle.

We summarize the iterative equations of the conventional CORDIC algorithm in Table I, where $[x(n), y(n)]^T$ is the vector to be rotated; $z(n)$ is the accumulation of rotation angle; the vector $[x_N, y_N]^T$ is the final vector; and P is the scaling factor, which is equal to $\prod_{n=0}^{N-1} \cos\theta_n$. In the rotational phase, the rotated vector is moved toward the target rotation angle iteratively according to (4). As we can see, the norm of the rotated vector will be changed in this procedure; therefore, we have to perform the scaling operations in the scaling phase to obtain the corrected norm value. Given the number of iterations, the scaling factor can be computed in advance. Several CORDIC-like schemes are proposed to perform the scaling operation in an efficient way [21]–[23].

B. MVR-CORDIC and EEAS-CORDIC Algorithms

In the modified vector rotation (MVR)-CORDIC algorithm [3], the authors made modifications on the rotation procedure to reduce the iteration as well as to accelerate the computational speed. Meanwhile, in [5] the authors extended the MVR-CORDIC work, and proposed a new idea to extend the Elementary Angle Set as shown

$$\theta \in \left\{ \arctan\left(\sum_{i=1}^{I} \mu_i 2^{-s_i} \right) \mid \mu_i \in \{-1, 0, 1\} \right. \\ \left. s_i \in \{0, 1, \ldots, S\} \right\} \tag{8}$$

TABLE I
SUMMARY OF THE CONVENTIONAL CORDIC ALGORITHM

I. Rotational Phase

For n = 0,..., N-1

– *Perform micro rotation:*

$$\begin{bmatrix} x(n+1) \\ y(n+1) \end{bmatrix} = \begin{bmatrix} 1 & -\mu_n \times 2^{-n} \\ \mu_n \times 2^{-n} & 1 \end{bmatrix} \begin{bmatrix} x(n) \\ y(n) \end{bmatrix}, \tag{4}$$

– *Calculate elementary angle*

$$\theta_n = \mu_n \tan^{-1}\left(2^{-n}\right), \tag{5}$$

– *Update accumulation angle*

$$z(n+1) = z(n) + \mu_n \theta_n, \tag{6}$$

End

II. Scaling Phase (adjust norm)

$$\begin{bmatrix} x_N \\ y_N \end{bmatrix} = P \begin{bmatrix} x(N) \\ y(N) \end{bmatrix}, \tag{7-a}$$

$$P = \prod_{n=0}^{N-1} \left(\sqrt{1+2^{-2n}}\right)^{-1}. \tag{7-b}$$

where I is defined as the Extending Factor and denotes the number of Singed Power-of-Two (SPT) terms; S denotes the number of maximum shift. In [5], the extending factor is set to 2. Based on (8), the EEAS has larger elementary angle set than the conventional CORDIC algorithm. We illustrate the effectiveness of the EEAS-CORDIC algorithm in Fig. 1. Fig. 1(a) depicts the elementary angles that can be achieved in the conventional CORDIC algorithm. For a fair comparison, we show the elementary angles of the EEAS-CORDIC algorithm, which has the same adder count as the conventional CORDIC algorithm, in Fig. 1(b). Fig. 1(b) shows that there are the more reachable elementary angles in the EEAS-CORDIC algorithm. This special feature can achieve better performance, including smaller quantization errors, hardware cost reduction, and lower computational complexity. As discussed in [5], the scaling function can be written as

$$P = \prod_n p_n, \quad p_n = \left(\sqrt{1 + \left(\sum_{i=1}^{I} 2^{-s_i}\right)^2}\right)^{-1} \tag{9}$$

where p_n is the n-th iteration scaling factor. Equation (9) shows that all scaling factors are different and depend on their own rotation angle. We depict the *constellations* (all reachable points of rotational angles) of the EEAS-CORDIC in Fig. 2(b). Note that the EEAS-CORDIC is suitable for vector rotation of fixed angles (e.g., twiddle factor of FFT). That is, in order to perform the forward rotation, we have to calculate the scaling factors of all rotation angles *off-line*, and store the pre-computed parameters in ROM in practical applications.

III. THE PROPOSED MSR-CORDIC ALGORITHM

Since the scaling factor is always greater than 1 in the existing CORDIC algorithms, it is necessary to scale down the norm of the input vector to its initial value in the scaling phase. Furthermore, the SQNR performance will be degraded due to the growth of the scaling factor. To alleviate the degradation of the SQNR performance, it is better to keep the norm of the input vector as close as to unity during each iteration. We will analyze this dynamic range issue in Section IV-B. Additionally, if the norm of the rotated vector is moved to the same as the original vector in the final microrotation operation, we can reduce the overhead of the scaling operation. Based on the idea, we reformulate the iterative arithmetic as follows:

Mixed Scaling Rotation

For $n = 1, \ldots, N$

$$\begin{bmatrix} x(n) \\ y(n) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{J} \eta_j(n) 2^{-s_j(n)} & -\sum_{i=1}^{I} \mu_i(n) 2^{-t_i(n)} \\ \sum_{i=1}^{I} \mu_i(n) 2^{-t_i(n)} & \sum_{j=1}^{J} \eta_j(n) 2^{-s_j(n)} \end{bmatrix} \times \begin{bmatrix} x(n-1) \\ y(n-1) \end{bmatrix} \tag{10}$$

— *Calculate elementary angle*

$$\theta_n = \tan^{-1}\left(\frac{\sum_{i=1}^{I} \eta_i(n) 2^{-s_i(n)}}{\sum_{j=1}^{J} \mu_j(n) 2^{-t_j(n)}}\right) \tag{11}$$

— *Update accumulation angle*

$$z(n) = z(n-1) + \theta_n \tag{12}$$

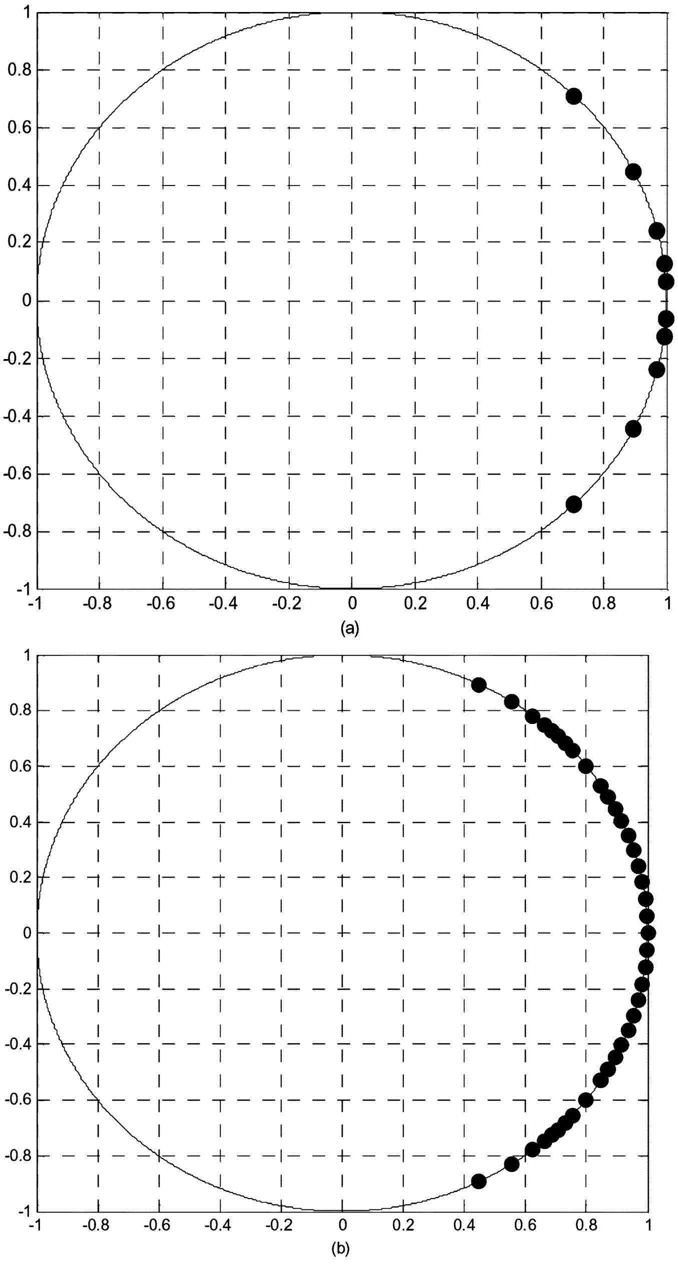


Fig. 1. Constellation of Elementary angle Set. (a) Conventional CORDIC with $N = 4$. (b) EEAS-CORDIC with $S = 4$ after applying the scaling operations.

— *Amplifying factor in the nth rotation*

$$p_n = \left(\sqrt{ \left(\sum_{i=1}^{I} 2^{-s_i(n)} \right)^2 + \left(\sum_{j=1}^{J} 2^{-t_j(n)} \right)^2 } \right)^{-1} \tag{13}$$

— *Product of the amplifying factor in the nth rotation*

$$\overline{p_n} = \overline{p_{n-1}} \times p_n \tag{14}$$

End

— *Scaling factor*

$$P = \prod_{n=1}^{N} p_n \tag{15}$$

where n denotes the nth iteration and N denotes the total number of iterations; $\eta_i(n), \mu_j(n) \in \{-1, 0, 1\}$. In the MSR-CORDIC algorithm, $\eta_i(n)$ or $\mu_j(n) = 0$ is valid, but it is invalid in the conventional CODDIC algorithm; $s_i(n), t_j(n) \in \{0, 1, \ldots, S\}$, and S denotes the number of maximum shift; I and J denote the number of SPT terms of $x(n)$ and $y(n)$, respectively, and they are referred to as the Extending Factor; θ_n is the nth elementary angle; z_n is the accumulative angle, and z_0 is 0; $\overline{p_n}$ denotes the product of the amplifying factors in nth iteration. The initial value of $\overline{p_0}$ is 1, and P denotes the Scaling Factor. In the MSR-CORDIC algorithm, P is always designed as close as possible to unity. N_{spt} is denoted as the number of SPT terms used in performing (6), which is the sum of I and J.

The proposed MSR-CORDIC algorithm is called MSR-CORDIC. The reason is that we can perform the microrotation operation and scaling operations at the same time. Equations (10)–(15) show that the $x(n)$ and $y(n)$ are rotated and scaled simultaneously in one iteration. In the conventional CORDIC and EEAS-CORDIC algorithms, the norm of both the schemes is enlarged after the microrotation operations, as illustrated in Fig. 2(a) and (b). On the contrary, (13) shows that the factor p_n can be either greater or less than 1 in the proposed MSR-CORIDC algorithm. In Fig. 2(c), we depict all reachable points in the two–dimensional (2-D) plane to emphasize the feature of p_n. Some other interesting features of the proposed scheme are discussed.

1) According to (11), the angles in the MSR-CORDIC is much denser than the conventional CORDIC and the EEAS-CORDIC. When we perform the MSR-CORDIC with more iterations, the combinational points are very dense around the unit circle. We illustrate this in Fig. 2(d) with two iterations. Furthermore, if we design the parameters, $s_i(n), t_j(n), \eta_i(n)$, and $\mu_j(n)$, appropriately such that the angle error $|z(N) - \Theta|$ and norm error $|1 - P|$ are minimized at the same time, where Θ is the target angle. Then, we can avoid the scaling operations. The MSR-CORDIC is faster in computational speed and the total hardware cost can be reduced.
2) In some applications, the rotation angles are larger than $\pi/4$, such as the twiddle factors in FFT. It is difficult for the conventional CORDIC to rotate to such an angle. In the MVR-CORDIC [3], the authors employ the prerotation scheme to overcome the problem and have the improved error performance. However, one extra multiplexer is needed to exchange the two input vectors. Besides the extra hardware cost, the computing speed will be degraded due to the extra multiplexer. On the contrary, in the MSR-CORDIC algorithm, the reachable angles are distributed from 0 to 2π, as illustrated in Fig. 2(d). Hence, it is easier to perform the rotation operation for all angles in the range of $[0, 2\pi]$ without performing the prerotation scheme.

IV. Analysis of Error Performance and Dynamic Range of MSR-CORDIC

In this section, we discuss two issues on the finite-wordlength performance of the proposed MSR-CORDIC, including the error performance and roundoff noise.

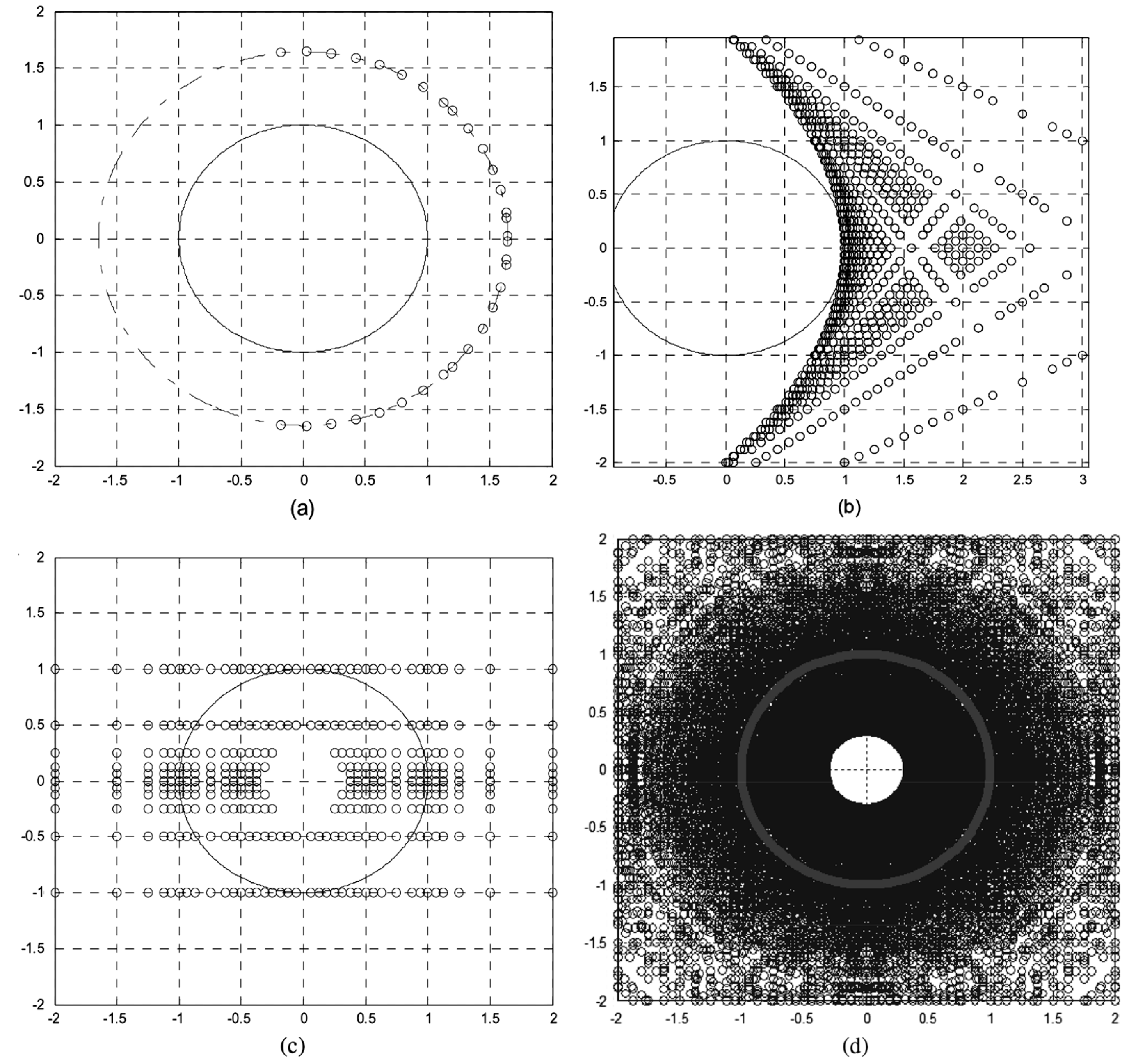


Fig. 2. Constellation of reachable points under the rotation process. (a) Conventional CORDIC with $N = R_m = 4$. (b) EEAS-CORDIC with maximum shift range $S = 4$ and $R_m = 2$. (c) MSR-CORDIC with $I = 2, J = 1$, and $N = 1$. (d) MSR-CORDIC with $I = 2, J = 1$, and $N = 2$ for $1/3 \leq \overline{P_n} \leq 3$ with maximum shift range $S = 4$.

A. Error Performance Analysis

To analyze the error performance, we define the error ε as the error distance between the ideal rotated point and the feasible rotated point. As shown in Fig. 3(a), we intend to rotate the point T to the target position, point A. Sometimes, it is impossible to reach the target point A with the innate constraint of the CORDIC algorithms. In practical VLSI implementations, we will also encounter the fixed-point impairment, which makes point A more unreachable. In such conditions, we must replace point A by another feasible point B. Assume that point B is closest to point A, then the error ε is equal to $\Delta l'$. Without loss of generality, we assume the length $\overline{OT}$ to be unity for the simplicity of analysis. When the angle $\Delta\theta$ is very small, we have

$$\begin{aligned}\Delta l'^2 &= \overline{OA}^2 + \overline{OB}^2 - 2\overline{OA} \times \overline{OB}\cos(\Delta\theta)\\ &= 1 + (1+\Delta l)^2 - 2 \times 1 \times (1+\Delta l) \times \sqrt{1-\sin^2(\Delta\theta)}\\ &\approx \Delta l^2 + \Delta\theta^2 \qquad (16)\end{aligned}$$

where $\Delta l = \overline{BC}$. Equation (16) shows the interesting fact that the error in angle $\Delta\theta$ and the norm error Δl have the same order on the degradation of the error performance. Hence, we must make the same effort to minimize both errors.

Lemma 1: To reduce the error, the angle and norm errors should be minimized at the same time.

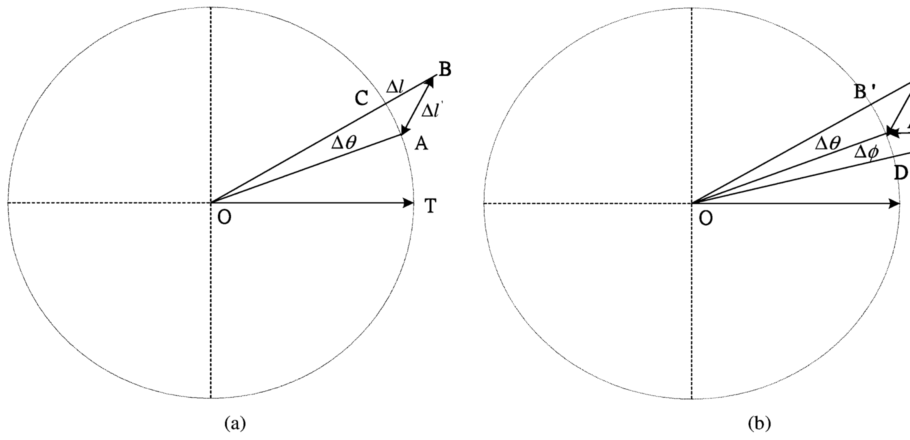


Fig. 3. The error performance analysis and the cause of its degradation. (a) Angle errors and norm errors that cause the error performance. (b) Degradation of error performance caused by the greedy search.

In Fig. 3(b), let both point D and point B be feasible points. In general, both points are very close to point A. So we can have three assumptions.

A1) Both angles $\Delta\phi$ and $\Delta\theta$ are very small.

B1) Both the lengths $\overline{B'B}$ and $\overline{D'D}$ are very short.

A3) Without lose of generality, let $\Delta\phi < \Delta\theta$.

Based on these assumptions, we have

$$\varepsilon = \min\{(\overline{B'B}^2 + \Delta\theta^2), (\overline{D'D}^2 + \Delta\phi^2)\}. \tag{17}$$

However, in the CORDIC algorithms [3], [5], the greedy algorithms are used to minimize the angle error $\Delta\theta$. Therefore, point D will be always chosen as a result of the assumption (A3), even though point B has smaller error than point D. That is

$$(\overline{B'B}^2 + \Delta\theta^2) < (\overline{D'D}^2 + \Delta\phi^2). \tag{18}$$

Q.E.D.

B. Roundoff Noise Analysis

In DSP systems, signals must be quantized and represented in fixed wordlength. A limited wordlength will result in the roundoff noise and degradation of the SQNR performance. There is no doubt that the larger dynamic range of signals will cause more severe roundoff noise. One trivial solution to the problem is to make use of larger wordlength; however, this will reduce the computational speed. On the other hand, if we implement with shorter wordlength, we will suffer from the danger of overflow. Therefore, in the rotation operation, it is better to make the norm variations as small as possible in each iteration.

1) Theoretical Analysis: To analyze the effect of the scaling factor on roundoff noise, we define the amplitude of the input signal ρ_s. It satisfies

$$\begin{aligned} &-2^{W_a} \le \rho_s \le 2^{W_b} - 1, \qquad \text{for } W_a, W_b \ge 0 \\ &W_{\max} = \max\{W_a, W_b\} \end{aligned} \tag{19}$$

where 2^{W_a} and 2^{W_b} denote the lower bound and upper bound of the input signals, respectively. $2^{W_{\max}}$ represents the upper bound of the absolute amplitude of the input signal. If the quantization level is, $\{\ldots, 2^1, -2^0, 0, 2^0, 2^1, \ldots\}$, $(W_{\max} + 1)$ is the minimum wordlength required to express the signal. In the quantization process, both the overflow and quantization noise need to be considered; the quantization noise reduces the error performance, whereas the overflow probably causes an *incorrect* result. Hence, we have to minimize the quantization error under the overflow-free constraint. When the signal is expressed in signed-digital representation with a fixed wordlength W, the quantization levels are $\{-2^{W-1-i}, \ldots, -2^{-i+1}, -2^{-i}, 0, 2^{-i}, 2^{-i+1}, \ldots, 2^{W-1-i}\}$, where i is the number of fractional digits. To avoid overflow, the following constraint must be satisfied:

$$2^{W_{\max}} \le 2^{W-1-i}. \tag{20}$$

Assume that the roundoff noise e_n is uniformly distributed, wide-sense stationary, and uncorrelated with other signals. Based on the quantization levels, the range of e_n is in the range of $(-2^{-i-1}, 2^{-i-1})$. Then, the variance of roundoff noise can be written as

$$\sigma_{e_n}^2 = \frac{V_{\text{LSB}}^2}{12} \tag{21}$$

where $V_{\text{LSB}} = 2^{-i}$ denotes the weight of the least significant bit. From (21), the variance of roundoff noise is proportional to V_{LSB}^2. Hence, to minimize the quantization noise is to minimize V_{LSB}. It can be easily shown that the optimal quantization levels are

$$\begin{aligned} \{-2^{W_{\max}} = -2^{W-1-i}, \ldots, -2^{-i+1}, -2^{-i}, 0, 2^{-i} \\ 2^{-i+1}, \ldots 2^{W-1-i} = 2^{W_{\max}}\}. \end{aligned} \tag{22}$$

Next, we discuss the scaling effect for two different cases ($P_n \ge 1$ and $P_n \le 1$).

A) Case 1 [$p_n \ge 1$ (denoted as p_{upper})]:
Assume that $1 \le p_{\text{upper}} \le 2^s$, for $s > 0$, and the quantization levels are the same as (22). When the rotation or scaling operations are performed, we need extra s bits to avoid the overflow. The practical method is to change the scale of quantization levels by P_{upper} times. Based on (21), the roundoff noise will be amplified by p_{upper}^2; equivalently, the SQNR is reduced to $1/p_{\text{upper}}^2$ times of the old value.

B) Case 2 [$p_n \le 1$ (denoted as p_{lower})]:
In this case, let $2^{-s} \le p_{\text{lower}} \le 1$, for $s > 0$, and the input

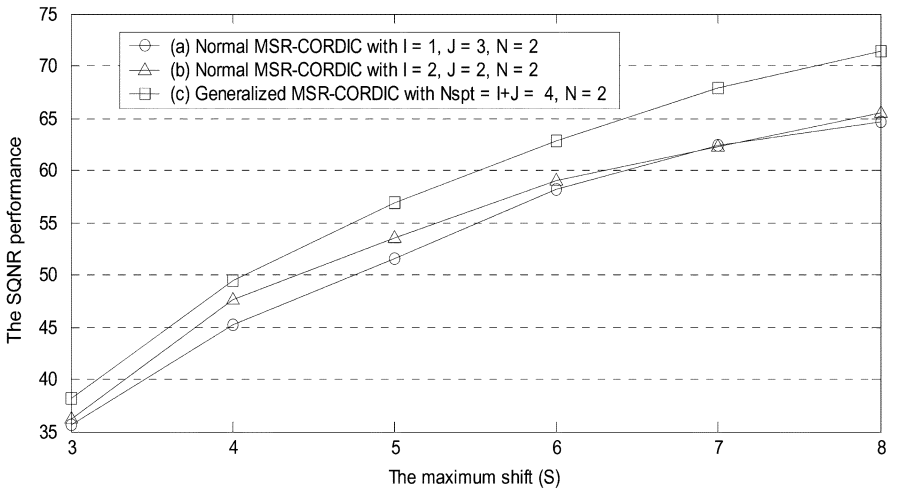


Fig. 4. The SQNR performance comparison among the MSR-CORDIC family. (a) Normal MSR-CORDIC with $I = 1, J = 3$. (b) Normal MSR-CORDIC with $I = 2, J = 2$. (c) Generalized MSR-CORDIC with $I + J = 4$.

signal is quantized as in Case 1. The norm of the input vector must be moved to its original value in the last iteration. We cannot adopt a smaller quantization levels to reduce the roundoff noise nor reduce the wordlength to save the hardware cost. Otherwise, overflow may occur. Hence, the signal level is degraded as p_{lower} times as the old one, when the quantization levels are kept unchanged. In this case, the SQNR is also reduced by a factor of p_{lower}^2.

The above analyses show that the scaling factor will cause more severe roundoff noise in both cases $p_n \geq 1$ and $p_n \leq 1$. That is, p_{upper} and p_{lower} amplify the same roundoff noise. We will use the result to adjust the scaling factors so as to control the roundoff noise.

2) Searching Schemes of Roundoff Noise Reduction: As aforementioned, the scaling factor plays an important role in affecting the roundoff noise. We will propose two schemes, Boundary Constraint and Best Candidate, to reduce the roundoff noise.

1) *Boundary Constraint:* In the MSR-CORDIC algorithm, one special feature is that the amplifying scaling factor $\overline{P_n}$ can be larger or less than 1. By utilizing the feature, the first proposed scheme is to limit the value of $\overline{p_n}$. Then, the maximum value (dynamic range) of signal can be predicted and controlled, and we can determine the minimum wordlength, and avoid the overflow. Furthermore, according to the analysis of Section IV–B-1, the best choice is to let lower bound p_{lower} and upper bound p_{upper} of the scaling factor, $\overline{p_n}$, equal to each derivatives; equivalently, $p_{\text{lower}} = 1/p_{\text{upper}}$.
2) *Best Candidate:* When multiplying all the parameters $s_i(n)$ and $t_j(n)$ by one constant k, the norm of the rotational vector is scaled by k times. However, the rotation angle remains unchanged. This is due to some different sets of CORDIC parameters can generate the same angle but with different norms. To alleviate the impairment of roundoff noise, we can select one particular parameter set, which genetates the amplifying scaling factor $\overline{p_n}$ under the boundary constraint and with the least roundoff noise.

V. Simulation Comparisons and Results

In this section, we will conduct computer simulations to show the effectiveness of the proposed schemes. The measurement of error performance is the averaged SNQR, which is obtained based on the ensemble average of 512 angles from $(\pi/1024)$ to $(\pi/4)$ with equal space. The optimal design parameters are obtained by employing exhaustive searching method for all design cases.

A. SQNR Performance Analysis Among the MSR-CORDIC Family With $N_{\text{spt}} = 4$

In this simulation, we compare different types of MSR-CORDIC with $N_{\text{spt}} = 4$. The parameters (I, J) can be $(1, 3) = (3, 1)$ and (2, 2). We remove sets (0, 4) and (4, 0), because they can only be used for the scaling operation. The MSR-CORDIC with the parameters $(I, J) = (1, 3), (3, 1)$, or $(2, 2)$ denotes the normal MSR-CORDIC. The generalized MSR-CORDIC means that (I, J) just satisfies $I + J = 4$ as shown in Fig. 4. As expected, the generalized MSR-CORDIC has the best error performance at the cost of 4 extra switches (to be discussed in Section VI-A).

It is interesting to note that the SQNR in Case (b) is better than Case (a), even though they have the same hardware complexity. To explain this, we use a fixed wordlength $N(>3)$ as an example. In case (b), the number $x(n+1)$ is the sum of $x(n)$ and $y(n)$ multiplied by a constant. Both two multipliers are expressed by two SPT terms. On the other hand, in Case (a) the multipliers of $x(n)$ and $y(n)$ are expressed by one and three SPT terms $(I = 1, J = 3)$. It may cause more errors when we try to express one number in wordlength W by using insufficient SPT terms. Especially, the error is more severe in the presentation of less SPT terms. Hence, Case (b) enhances the SQNR performance than Case (a). From this viewpoint, we should implement the MSR-CORDIC of Case (b) rather than Case (a). Below is a guideline to determine I and J in MSR-CORDIC implementations.

a) Both I and J are equal to $N_{\text{spt}}/2$, when N_{spt} is even.
b) $I = (N_{\text{spt}} + 1)/2$, and $J = I - 1$, when N_{spt} is odd.

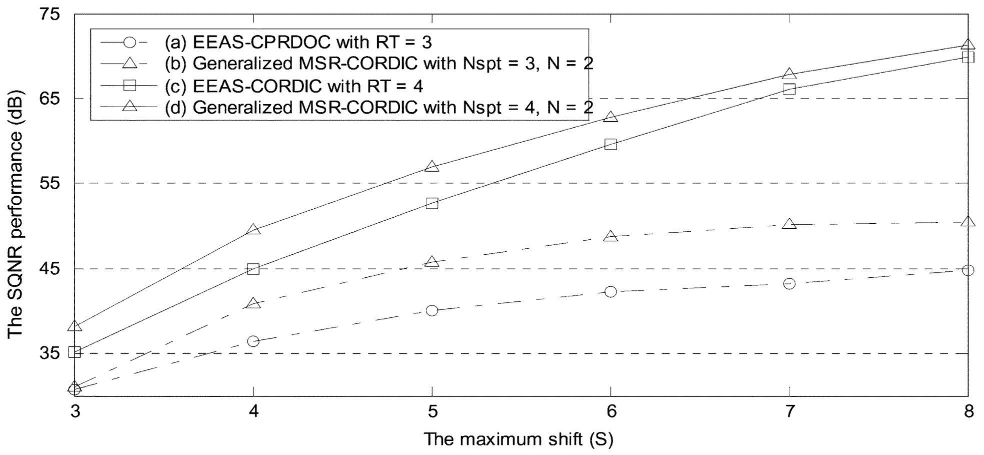


Fig. 5. SQNR comparison between MSR-CORDIC and the EEAS-CORDIC.

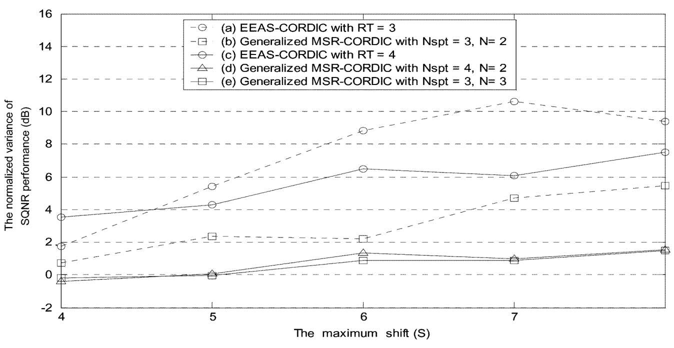


Fig. 6. Analysis of SQNR variance among EEAS-CORDIC and the MSR-CORDIC.

B. Comparison of SQNR Variance

The mean of SQNR is a good index to compare the error performance, and it indicates the first-order statistical property of the error distribution. Furthermore, we also investigate the second-order statistical property, the variance of SQNR, in our analysis. Fig. 5 shows their SQNR performance, and the variance is normalized as

$$\sigma^2_{\text{SQNR}} = \frac{\text{Var(SQNR)}}{\text{mean(SQNR)}}. \tag{23}$$

We perform two sets of experiments for EEAS-CORDIC and MSR-CORDIC, respectively. For the fair comparison, both sets have the similar SQNR mean values. The simulation result shows that the error distribution becomes smaller as the mean of SQNR becomes higher.

Fig. 6 shows that the MSR-CORDIC has better second-order statistical property than the EEAS-CORDIC in cases (a-b) and (c-e). Another interesting fast is that a different kind of MSR-CORDIC has nearly the same variance, as shown in Cases (d) and (e). That implies that the MSR-CORDIC algorithm is less sensitive to the kinds of MSR-CORDIC architecture with similar SQNR performance.

C. Roundoff Noise Impairment of the Scaling Factor

In the VLSI design flow, one important step is the fixed-point simulation, which assists the designer to determine the required wordlength. If the wordlength is overdetermined, we will suffer from higher cost and slower computational speed. However, we will degrade the SQNR performance or encounter the overflow if the wordlength is too short. To avoid both cases, we will check the scaling factor, and see it relationship between SQNR performance and the dynamic range.

In the conventional CORDIC algorithm with eight iterations of microrotation, the norm is amplified by 1.6468. In both MVR-CORDIC and EEAS-CORDIC algorithms, the scaling factors are not fixed, and the norm is changed larger in each iteration. With the amplification, both the aforementioned CORDIC algorithms will cause roundoff noise more seriously. However, the scaling factor $\overline{p_n}$ in the proposed MSR-CORIDC algorithm can be designed to be either greater or less than 1. According to the theoretical analysis in Section IV-B-1, p_{lower} is set to $1/p_{\text{upper}}$. In Fig. 7, the parameters in Case (a) and (b) are $N_{\text{spt}} = 3, N = 3$, and $N_{\text{spt}} = 4, N = 2$, respectively. As shown in Fig. 7(a) and (b), the SQNR performance saturates as p_{upper} is close to 1.5. Hence, our proposed algorithm can design to suffer from the least impairment of roundoff noise.

VI. VLSI Architecture of MSR-CORDIC

In this section, we illustrate the generalized VLSI structure of the MSR-CORDIC algorithm, and employ the configurable-data-path scheme to enhance the SQNR performance.

A. Normal MSR-CORDIC Structure

First, we reformulate (10) as

$$x(n) = \sum_{i=1}^{I} \eta_i(n) 2^{-s_i(n)} x(n-1) - \sum_{j=1}^{J} \mu_j(n) 2^{-t_j(n)} y(n-1) \tag{24}$$

$$y(n) = \sum_{j=1}^{J} \mu_j(n) 2^{-t_j(n)} x(n-1) + \sum_{i=1}^{I} \eta_i(n) 2^{-s_i(n)} y(n-1). \tag{25}$$

In (24) and (25), both $x(n)$ and $y(n)$ are linear combinations of their prior $x(n-1)$ and $y(n-1)$ values. All the coefficients of $x(n)$ and $y(n)$ are the sum of power of two and can be implemented by shift operations. Therefore, two Barrel Shifter Arrays (BSAs) can be used to perform these shift operations. The

TABLE II
FOUR TYPES IN MSR-CORDIC WITH $N_{\mathrm{spt}} = I + J = 3$

Type	I	J	Equation
I (*Scaling Type*)	3	0	$x(n)=\eta_1(n)2^{-s_1(n)}x(n-1)+\eta_2(n)2^{-s_2(n)}x(n-1)+\eta_3(n)2^{-s_3(n)}x(n-1)$ $y(n)=\eta_1(n)2^{-s_1(n)}y(n-1)+\eta_2(n)2^{-s_2(n)}y(n-1)+\eta_3(n)2^{-s_3(n)}y(n-1)$
II (*Normal type*)	2	1	$x(n)=\eta_1(n)2^{-s_1(n)}x(n-1)+\eta_2(n)2^{-s_2(n)}x(n-1)-\mu_1(n)2^{-t_1(n)}y(n-1)$ $y(n)=\mu_1(n)2^{-t_1(n)}x(n-1)+\eta_1(n)2^{-s_1(n)}y(n-1)+\eta_2(n)2^{-s_2(n)}y(n-1)$
III (*Normal type*)	1	2	$x(n)=\eta_1(n)2^{-s_1(n)}x(n-1)-\mu_1(n)2^{-t_1(n)}y(n-1)-\mu_2(n)2^{-t_2(n)}y(n-1)$ $y(n)=\mu_1(n)2^{-t_1(n)}x(n-1)+\mu_2(n)2^{-t_2(n)}x(n-1)+\eta_1(n)2^{-s_1(n)}y(n-1)$
IV (Exchange-Scaling Type)	0	3	$x(n)=-\mu_1(n)2^{-t_1(n)}y(n-1)-\mu_2(n)2^{-t_2(n)}y(n-1)-\mu_3(n)2^{-t_3(n)}y(n-1)$ $y(n)=\mu_1(n)2^{-t_1(n)}x(n-1)+\mu_2(n)2^{-t_2(n)}x(n-1)+\mu_3(n)2^{-t_3(n)}x(n-1)$

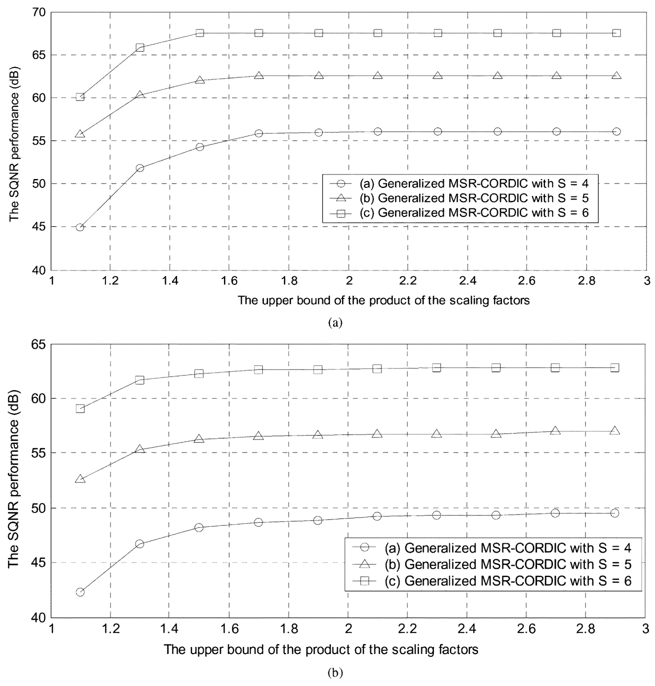


Fig. 7. Relationship between SQNR performance and scaling factors of the upper bound, p_{upper}, in the generalized MSR-CORDIC. (a) $N_{\mathrm{spt}} = 3$ and $N = 3$. (b) $N_{\mathrm{spt}} = 4$ and $N = 2$.

number of output signals from each BSA is N_{spt}. To sum the output signals, $2(N_{\mathrm{spt}} - 1)$ add/subtract operations must be performed. That is, $2(N_{\mathrm{spt}} - 1)$ adders/subtractors are necessary and sufficient to finish the process in one clock cycle.

To show the normal structure of the MSR-CORDIC, we take $N_{\mathrm{spt}} = 3$. In this example, the sum of $x(n)$ and $y(n)$ consists of four different types, as listed in Table II.

A) *Scaling Type* (Type I): $x(n)$ and $y(n)$ are the sum of the shifted version of their each preceding value, $x(n-1)$ and $y(n-1)$. The operation is the same as the scaling operation in the EEAS-CORDIC.

B) *Exchange-Scaling Type* (Type IV): In this type, $x(n)$ and $y(n)$ are the sum of the shifted version of their each preceding value, $y(n-1)$ and $x(n-1)$. That is, this type operation induces a rotation of angle $\pi/2$. The operation in this type is similar to the prerotation scheme in the EEAS-CORDIC.

C) *Normal Type* (Other Types): Note that it is impossible to use only the Scaling Type and/or Exchange-Scaling Type to implement the rotation circuits. The reason is that they perform only the scaling operation at fixed rotation angles $0, \pm\pi/2$ and π. Hence, we can use only Types II and III to construct the rotation circuits.

All normal types of the MSR-CORDIC can be used to implement the rotation circuits, and the hardware cost and computational speed are the same. However, they have different SQNR performances. We have shown how to determine I and J with a given number N_{spt} by using the simulations in Section V-A. Fig. 8 illustrates the structure of Type II for the normal MSR-CORDIC. The Control Unit is in charge of controlling BSAs and adders/subtractors.

B. Universal Architecture of the MSR-CORDIC

The MSR-CORDIC can be seen as a universal vector rotational CORDIC Engine. Take the MVR-CORDIC and EEAS-CORDIC for example. When $s_0 = 0$ and $\mu_0 = 1$ in Type II of Table II, the MSR-CORDIC is equivalent to the EEAS-CORDIC. Furthermore, when $s_1 = \mu_0 = 1$ and $\mu_1 = 1$, the MSR-CORDIC is reduced to MVR-CORDIC. Therefore, the proposed MSR-CORDIC can be considered as a more generalized CORDIC engine.

C. Generalized MSR-CORDIC Structure

In all normal types of MSR-CORDIC structure, the number of the output signals from two BSAs is the same. Therefore, the architecture is the same among them, except the data-path of BSA output signals. If we employ the switches to control the output signals, $x(n)$ can be the sum of I-shifted versions of $x(n-1)$ and J-shifted versions of $y(n-1)$; $y(n)$ can be the sum of J-shifted versions of $x(n-1)$ and I-shifted versions of $y(n-1)$; and (I, J) satisfies $I + J = N_{\mathrm{spt}}$. We call it the generalized MSR-CORDIC structures [illustrated in Fig. 9(a)]. The Control Unit is in charge of controlling BSAs, switches, and adders/subtractors. Three extra switches are used to control the output signal of two BSAs based on the signal from the controller. Fig. 9(b) shows the complete data-paths, which operates in Type II and Fig. 9(c) shows the operation of switches.

TABLE III
ROM SIZE FOR EACH ANGLE IN THE MSR-CORDIC STRUCTURE WITH $N_{\rm spt} = 3$ AND $S = 6$. (UNIT = BITS)

Type	Generalized MSR-CORDIC	Normal MSR-CORDIC
Bits of Coefficients S_i	$log_2(S)$	$log_2(S)$
Bits of Coefficients μ_i	2	2
Control Signal for Each Switch	1	0
Total	N_{spt}*($log_2(S)$ + 3)	N_{spt}*($log_2(S)$ + 2)

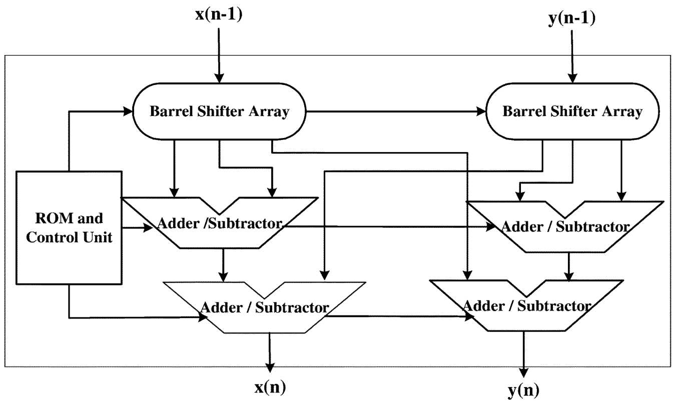


Fig. 8. Normal type (Type II, III) MSR-CORDIC iterative structure with $I = 2, J = 1$.

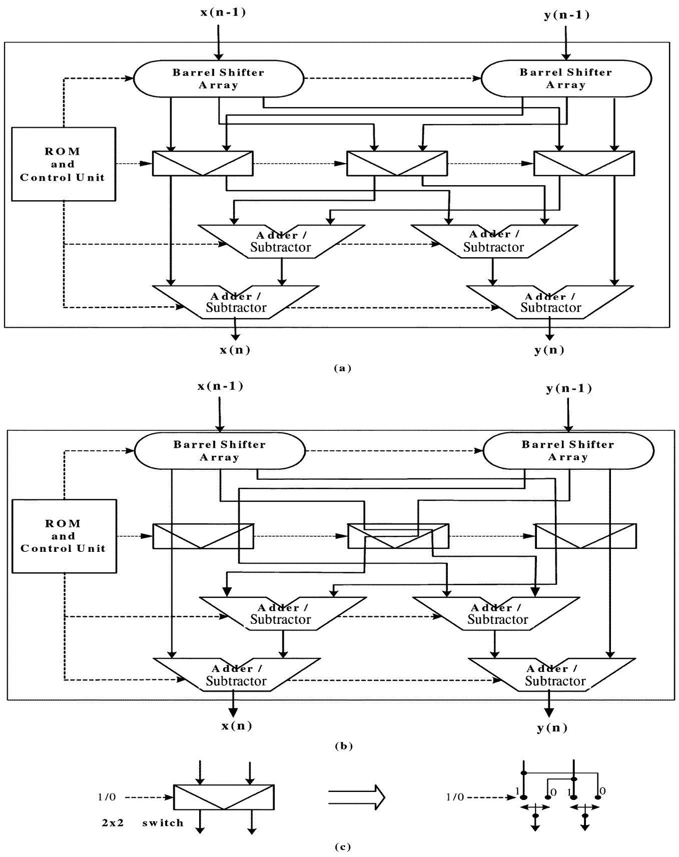


Fig. 9. (a) Generalized MSR-CORDIC iterative structure with $N_{\rm spt} = 3$. (b) Data-Path of BSA output signals in case II. (c) Operation of the 2×2 switching box.

D. ROM Size of the MSR-CORDIC

When hardware cost is concerned, we may employ the iterative VLSI structure to perform the rotation operation in our proposed algorithm. From (10), the MSR-CORDIC requires $2(N_{\rm spt} - 1)$ adders/subtrators and 2 BSAs in each iteration (or in one MSR-CORDIC module). The control signals, including $\eta_i(n), \mu_j(n), s_i(n), t_j(n)$, and the switch control signals, need to be calculated in advance and stored in the ROM. In the normal MSR-CORDIC structure, the data-path is fixed, so the switch control signal is not required, which leads to a smaller ROM size. Taking the aforementioned MSR-CORDIC with $N_{\rm spt} = 3$ and $S = 6$ for example; the bits to store each parameter, $\{\eta_i(n), \mu_j(n)\} \in \{-1, 0, 1\}, \{s_i(n), t_j(n)\} \in \{0, 1, \ldots, 6\}$ are 2 and 3, respectively. One bit is needed for the switch control signal in the generalized type, but it is not necessary in any normal type. For each iteration, the ROM size for the generalized and normal type are $(\log_2(S) + 3)N_{\rm spt}$ and $(\log_2(S) + 2)N_{\rm spt}$ bits, respectively. We summarize the ROM size in Table III.

E. Unfolded Architecture of the MSR-CORDIC for High-Speed Applications

For high-speed rotational operations, we can unfold the iterative implementation of Fig. 9 to obtain the unfolded parallel structure as depicted in Fig. 10. It is composed of an N-cascaded basic MSR-CORDIC rotator. Each rotator performs one microrotation-scaling as defined in (10). The MSR-CORDIC architecture is very regular and modular. It is very VLSI-friendly, and can be easily implemented in pipeline and parallel. For the requirement of a very high computational speed, each rotation operation can employ one copy of the MSR-CORDIC rotation circuit to accelerate the computing speed. For example, in a CORDIC-based orthogonal IIR digital lattice filters design [24], the rotational angles can be designed by the method of filter design to satisfy the required specification. Then, we can appropriately apply each MSR-CORDIC rotator to perform an angle in the designed lattice filter. Moreover, due to the fixed rotation angles, the BASs, switches, Control Unit, and ROM can be eliminated. Therefore, we only need few adders/subtractors to implement this filter.

F. Application of the MSR-CORDIC to 16-Point FFT Twiddle Factor Design

In [29], we have demonstrated the use of MSR-CORDIC in performing twiddle factor operations of FFT processors. To see the details of the MSR-CORDIC, we list the parameters of twiddle factor angles of a 16-point FFT in Table IV. The parameters of the generalized MSR-CORDIC is $I + J = 3, N = 3$, and $S = 16$.

TABLE IV
PARAMETERS OF TWIDDLE FACTOR ANGLES OF THE GENERALIZED MSR-CORDIC WITH $I + J = 3, N = 3$, AND $S = 16$ FOR THE TWIDDLE FACTOR OPERATIONS OF A 16 POINT-FFT

	$x(n=3)$			SQNR
Θ	Iteration 1 $(n=0)$	Iteration 2 $(n=1)$	Iteration 3 $(n=2)$	(dB)
0	$x(n)$	$x(n)$	$x(n)$	---
$\pi/16$	$(1-2^{-7})x(n)-(2^{-4})y(n)$	$x(n)-(2^{-7}+2^{-10})y(n)$	$(1-2^{-9})x(n)-(2^{-3})y(n)$	85.1
$2\pi/16$	$(2^{-1})x(n)-(-1+2^{-3})y(n)$	$(2^{-3})x(n)-(1-2^{-6})y(n)$	$x(n)-(-2^{-13}-2^{-16})y(n)$	112.1
$3\pi/16$	$(1-2^{-9})x(n)-(2^{-3})y(n)$	$(1-2^{-9})x(n)-(2^{-1})y(n)$	$(1-2^{-3}+2^{-6})x(n)$	92.8
$4\pi/16$	$x(n)-(1-2^{-15})y(n)$	$(1-2^{-2}-2^{-9})x(n)$	$(1-2^{-4}+2^{-7})x(n)$	89.7
$5\pi/16$	$(1-2^{-9})x(n)-(-2^{-3})y(n)$	$(2^{-1})x(n)-(1-2^{-9})y(n)$	$(1-2^{-3}+2^{-6})x(n)$	92.8
$6\pi/16$	$(2^{-1})x(n)-(1-2^{-3})y(n)$	$(1-2^{-6})x(n)-(2^{-3})y(n)$	$x(n)-(2^{-13}+2^{-16})y(n)$	112.1
$7\pi/16$	$(1-2^{-7})x(n)-(-2^{-4})y(n)$	$(2^{-7}+2^{-10})x(n)-y(n)$	$(1-2^{-9})x(n)-(-2^{-3})y(n)$	85.1
$\pi/2$	$x(n)$	$x(n)$	$-y(n)$	---

TABLE V
COMPARISON OF EXISTING ALGORITHMS PERFORMING VECTOR ROTATION IN 2-D PLANE, WHERE WORDLENGTH (W) IS 16

	Hardware Required	FA Count	Latency	Latency $1T_{CSA}=9T_{FA}$	T x A	SQNR Performance in dB
Direct Implementation	4 multipliers + 3 Adders	1,072	$17T_{FA}$ + $1T_{CSA}$	26 T_{FA}	27,872	98.7
Booth-Encoded Multiplier	4 multipliers + 3 Adders	592	$9T_{FA}$ + $1T_{CSA}$	18 T_{FA}	10,656	96.3
Conventional CORDIC Algorithm	83 Adders (Average)	1,328	$40T_{FA}$ + $1T_{CSA}$	49 T_{FA}	65,072	97.4
Angle Recoding Technique ($R_m = 6$, $R_s = 6$) [18]	47 Adders	752	$22T_{FA}$ + $1T_{CSA}$	31 T_{FA}	23,312	93.3
CORDIC Algorithm [25]	63 Adders	1008	$30T_{FA}$ + $1T_{CSA}$	39 T_{FA}	39,312	96.3
Radix-4 CORDIC Algorithm [26-27]	51 Adders ($N = 4W/5$)	816	$24T_{FA}$ + $1T_{CSA}$	33 T_{FA}	26,928	96.7
EEAS-CORDIC Algorithm ($R_m = 2$, $R_s = 2$) [5]	29 Adders	464	$10T_{FA}$ + $1T_{CSA}$	19 T_{FA}	8,816	95.1
Proposed Generalized MSR-CORDIC Algorithm ($N_{spt} = 3$, $N = 3$)	21 Adders	336	$7T_{FA}$ + $1T_{CSA}$	16 T_{FA}	5,376	90.0

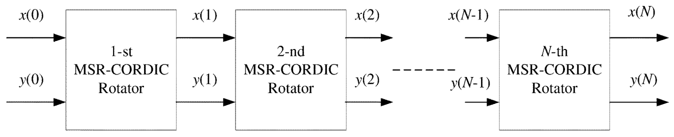


Fig. 10. Unfolded parallel VLSI structure of the MSR-CORDIC algorithm.

Note that Table IV only lists the part of $x(n)$. We can easily obtain $y(n)$ from (10). Table IV shows that the iteration number of the MSR-CORDIC algorithm can be reduced to 3. As a result, we can significantly reduce the long latency as compared with the conventional CORDIC algorithms.

G. Hardware Comparison With Existing CORDIC Algorithms

In this section, we compare the MSR-CORDIC with existing rotational algorithms [5], [18], [26], [27], which perform the vector rotations in 2-D Plan. In Table V, we provide a commonly used comparison index $(T * A)$, which is a product of estimated area (A) and operation time (T). In this table, FA denotes the Full Adder, T_{FA} denotes the delay introduced by a single full adder, and T_{CSA} denotes the delay of Carry Select Adder (CSA). To give a fair comparison, we employ the Carry-Save Adder (CSA) scheme in all CORDIC architectures, and perform vector merge with carry select adder at the last stage. As can be seen in Table V, the proposed MSR-CORDIC algorithm requires the smallest $T * A$ among all the rotational algorithms.

In summary, Direct Implementation, Booth-Encoded Multiplier, EEAS-CORDIC, Angle Recoding [18], CORDIC algorithm [25], and MSR-CORDIC are employed to applications such as FFT, DCT, and lattice filters, where the rotational angles are known in advance. All these rotational circuits except Angle Recoding [18] use the same hardware module to process all rotational angles. In Angle Recoding [18], the required iteration number of each angle is different. This makes it is necessary to implement in the worse case for parallel implementations. On the contrary, the conventional CORDIC and radix-4 CORDIC algorithm [26], [27] can be applied to applications, which require vector rotational mode or angle accumulation mode. In addition, radix-4 requires extra hardware to process in the angle accumulation mode. Another advantage of the MSR-CORDIC is the very low hardware complexity. The vector rotation can be accomplished with only 20 adders/subtractors of 16-bit wordlength. Meanwhile, the SQNR performance in an average sense is 90.0 dB. Compared with the AR technique [18] and EEAS approach [5], only 44.6% and 72.4%, respectively, of the hardware complexity are required. The saving in hardware complexity is significant.

VII. Conclusion

In this paper, we proposed the novel MSR-CORDIC algorithm, which enhance SQNR performance of both first- and second-order statistical properties. In practical implementations, the proposed MSR-CORDIC can be appropriately applied to various DSP systems, which require high computational speed and the angles are known in advance. We also propose a generalized MSR-CORDIC structure to tradeoff hardware cost and SQNR performance.

References

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

[2] J. C. Chih and S. G. Chen, "A fast CORDIC algorithm based on a novel angle recoding scheme," in *Proc. 2000 IEEE Int. Symp. Circuits Syst.*, vol. 4, pp. 621–624.

[3] C. S. Wu and A. Y. Wu, "Modified vector rotational CORDIC (MVR-CORDIC) algorithm and architecture," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 48, no. 6, pp. 548–561, Jun. 2001.

[4] A. Y. Wu and C. S. Wu, "A unified view for vector rotational CORDIC algorithms and architectures based on angle quantization approach," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 49, no. 10, pp. 1442–1456, Oct. 2002.

[5] 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, Analog Digit. Signal Process.*, vol. 50, no. 9, pp. 589–601, Sep. 2003.

[6] Y. H. Hu, "The quantization effects of the CORDIC algorithm," *IEEE Trans. Signal Process.*, vol. 40, no. 4, pp. 834–844, Apr. 1992.

[7] A. M. Despain, "Fourier transform computers using CORDIC iterations," *IEEE Trans. Comput.*, vol. 23, no. 10, pp. 993–1001, Oct. 1974.

[8] M. D. Ercegovac and T. Lang, "Redundant and on-line CORDIC: Application to matrix triangularization and SVD," *IEEE Trans. Comput.*, vol. 39, no. 6, pp. 725–740, Jun. 1990.

[9] G. J. Hekstra and E. F. A. Deprettere, "Fast rotations: Low-cost arithmetic methods for orthonormal rotation," in *Proc. 13th IEEE Symp. Comput. Arithmetic, 1997*, pp. 116–125.

[10] M. Jun, K. K. Parhi, and E. F. Deprettere, "Annihilation-reordering look-ahead pipelined CORDIC-based RLS adaptive filters and their application to adaptive beamforming," *IEEE Trans. Signal Process.*, vol. 48, no. 8, pp. 2414–2431, Aug. 2000.

[11] G. Lijun and K. K. Parhi, "Hierarchical pipelining and folding of QRD-RLS adaptive filters and its application to digital beamforming," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 47, no. 12, pp. 1503–1519, Dec. 2000.

[12] P. P. Vaidyanathan, "A unified approach to orthogonal digital filters and wave digital filters based on the LBR two-pair extraction," *IEEE Trans. Circuits Syst.*, vol. CAS-32, no. 7, pp. 673–686, Jul. 1985.

[13] A. Y. Wu, K. J. R. Liu, and A. Raghupathy, "System architecture of an adaptive reconfigurable DSP computing engine," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 2, pp. 54–73, Feb. 1998.

[14] E. F. Deprettere, P. Dewilde, and R. Udo, "Pipelined CORDIC architectures for fast VLSI filtering," in *Proc. IEEE Int. Conf. ASSP*, 1984, pp. 1–4.

[15] L. W. Chang and S. W. Lee, "Systolic arrays for the discrete Hartley transform," *IEEE Trans. Signal Process.*, vol. 29, no. 11, pp. 2411–2418, Nov. 1991.

[16] W. H. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-25, no. 9, pp. 1004–1009, Sep. 1977.

[17] Y. H. Hu and S. Naganathan, "Efficient implementation of the Chirp Z-transform using a CORDIC processor," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, no. 2, pp. 352–354, Feb. 1990.

[18] ——, "An angle recoding method for CORDIC algorithm implementation," *IEEE Trans. Comput.*, vol. 18, no. 1, pp. 99–102, Jan. 1993.

[19] J. M. Muller, *Elementary Functions: Algorithms and Implementations*. Cambridge, MA: Birkhauser, 1997.

[20] E. Antelo, T. Lang, and J. D. Bruguera, "Very-high radix circular CORDIC: Vectoring and unified rotation/vectoring," *IEEE Trans. Comput.*, vol. 49, no. 7, pp. 727–739, Jul. 2000.

[21] G. L. Haviland and A. A. Tuszynski, "A CORDIC arithmetic processor chip," *IEEE Trans. Comput.*, vol. 29, pp. 68–79, 1980.

[22] H. M. Ahmed, J. M. Delosme, and M. Morf, "Highly concurrent computing structures for matrix arithmetic and signal processing," *IEEE Trans. Comput.*, vol. 15, no. 1, pp. 65–82, Jan. 1982.

[23] J. M. Delosme, "A processor for two-dimensional symmetric eigenvalue and singular value arrays," in *Proc. 21st Asilomar Conf. Circuits, Syst., Comput.*, 1987, pp. 217–221.

[24] J. Ma, K. K. Parhi, and E. F. Deprettere, "Pipelined CORDIC-based cascade orthogonal IIR digital filters," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 47, no. 11, pp. 1238–1253, Nov. 2000.

[25] A. A. J. de Lange and Ed. F. Deprettere, "Design and implementation of a floating point quasi-systolic general purpose CORDIC rotator for high rate parallel data and signal processing," in *Proc. IEEE Symp. Comput. Arithmetic*, 1991, pp. 272–281.

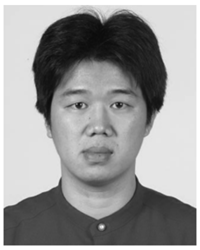
[26] E. Antelo, J. Villalba, D. Bruguera, and E. Zapata, "High performance rotation architectures based on the radix CORDIC algorithm," *IEEE Trans. Comput.*, vol. 46, no. 8, pp. 855–870, Aug. 1997.

[27] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low power CMOS digital design," *IEEE J. Solid-State Circuits*, vol. 27, no. 4, pp. 473–484, Apr. 1992.

[28] J. C. Chih and S. G. Chen, "Fast CORDIC algorithm based on a new recoding scheme for rotation angles and variable scale factors," *J. VLSI Signal Process.*, vol. 33, pp. 19–29, Jan. 2003.

[29] J. C. Kuo *et al.*, "VLSI design of a variable-length FFT/IFFT processor for OFDM-based communication systems," *EURASIP J. Appl. Signal Process.*, no. 13, pp. 1306–1316, Dec. 2003.

[30] S. Y. Park and N. I. Cho, "Fixed-point error analysis of CORDIC processor based on the variance propagation formula," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 51, no. 3, pp. 573–584, Mar. 2004.



Chih-Hsiu Lin received the B.S. degree in mechanical engineering from National Taiwan University (NTU), Taiwan, R.O.C., in 1998.

He is currently working toward the Ph.D. degree with the Graduate Institute of Electronics Engineering, (NTU). His research interests include the VLSI implementation of communication systems and multiuser detections.

An-Yeu (Andy) Wu (S'91–M'96) received the B.S. degree from National Taiwan University (NTU), Taiwan, R.O.C., in 1987, and the M.S. and Ph.D. degrees from the University of Maryland, College Park, in 1992 and 1995, respectively, all in electrical engineering.

From August 1995 to July 1996, he was a Member of Technical Staff with AT&T Bell Laboratories, Murray Hill, NJ, working on high-speed transmission IC designs. From 1996 to July 2000, he was with the Electrical Engineering Department, National Central University, Taiwan. He is currently an Associate Professor with the Graduate Institute of Electronics Engineering, Department of Electrical Engineering, NTU. His research interests include low-power/high-performance VLSI architectures for DSP and communication applications, adaptive/multirate signal processing, and reconfigurable broad-band access systems and architectures.

Dr. Wu served as an Associate Editor for EURASIP *Journal of Applied Signal Processing* (JASP) from 2001 to 2004. He became the Associate Editor of the IEEE Transactions on Very Large Scale Integration (VLSI) Systems since July 2003. He has served on the Technical Program Committees of major IEEE International Conferences such as ISCAS, SiPS, AP-ASIC, and A-SSCC. He received the A-class Research Award from National Science Council, Taiwan, four times during 1997 and 2000. He also received the Macronix International Corporation (MXIC) Young Chair Professor Award in 2003, and the Distinguished Young EE Engineer Award from The Chinese Institute of Electrical Engineering in 2004.