A RADIX-4 REDUNDANT CORDIC ALGORITHM WITH FAST ON-LINE VARIABLE SCALE FACTOR COMPENSATION

Chieh-Chih Li

Opto-Electronics & Systems Laboratories
Industrial Technology Research Institute
Hsinchu, Taiwan, ROC

Sau-Gee Chen

Department of Electronics Engineering
National Chiao Tung University
Hsinchu, Taiwan, ROC
E-mail: sgchen@cc.nctu.edu.tw

ABSTRACT

In this work, a fast radix-4 redundant CORDIC algorithm with variable scale factor is proposed. The algorithm includes an on-line scale factor decomposition algorithm that transforms the complicated variable scale factor into a sequence of simple shift-and-add operations and does the variable scale factor compensation in the same fashion. On the other hand, the on-line decomposition algorithm itself can be realized with a simple and fast hardware. The new CORDIC algorithm has the smallest number of *0.8n* iterations among all the CORDIC algorithms, which requires only about two-third rotation number that of the existing best (hybrid radix-2 and radix-4) redundant algorithms. Therefore, the new algorithm achieves fast rotation iterations, high-speed and low-overhead scale factor compensations, which are hard to attain simultaneously for the existing algorithms. The on-line scale factor compensation can be also applied to the existing on-line CORDIC algorithms.

1. INTRODUCTION

CORDIC [1,2] algorithm is an efficient scheme for computing elementary functions especially for the trigonometric functions. Since the algorithm can be realized as a sequence of shift-and-add operations followed by a scale factor compensation operation, it is very suited for VLSI implementation and widely applied to DSP applications.

Most of the CORDIC algorithms assume a constant scale factor for the ease of scale factor compensation. However, they have to either do an accurate but slow decision operation for rotation direction or do rough direction decision at the expense of extra compensation operations [4], [6]. In addition, they have to rotate even when the rotation angle has been converged. To speedup CORDIC operations, the following techniques are widely used: (1) applying carry-free redundant addition scheme [3-8]; (2) fast decision of rotation direction with only a few most significant digits (MSDs) of the related parameters [3-8]; (3) skipping unnecessary rotations; (4) recoding rotation angle for saving rotation iterations; and (5) applying radix-4 rotation scheme [5,10,13]. The the 2nd to 4th techniques result in variable scale factors. Variable scale factors have the trouble of complicated scale factor computation followed by penalty compensation [7,8]. Due to the considerable overhead generated by variable scale factor, the existing radix-4 CORDIC algorithms resort to constant scale factor approach [5,10,13]. However, these constant scale-factor CORDICs are not pure radix-4 algorithms. In fact, they are all hybrid radix-2 and radix-4 algorithms. As a result, all these approaches have minorly reduced iteration numbers, at the cost of control overheads. Ideally, a pure radix-4 algorithm would achieve the best performance.

To alleviate the mentioned disadvantages related to prior arts, a pure fast radix-4 redundant CORDIC algorithm with variable scale factor is proposed. The algorithm includes an on-line variable scale factor decomposition algorithm that transforms the complicated variable scale factor $\prod_{i=0}^{n/2} /1\sqrt{(1+\delta_i^2 2^{-4i-2})}$ into a sequence of simple shift-and-add operations of $\prod_{i=0}^{n/2}(1+s_i 2^{-2i-1})$ in an on-line fashion, where δ_i, $s_i \in \{-2,-1,0,1,2\}$. Here s_i only depends on δ_i. Both δ_i and s_i can be easily determined by estimating their corresponding intermediate variables with very short wordlength. In all, the new algorithm has the smallest number 0.8*n* of shift-and-add steps among all the CORDIC algorithms. Therefore, the new CORDIC algorithm achieves fast rotation iterations, high-speed and low-overhead scale factor compensations, which are hard to attain simultaneously for the existing algorithms. The on-line scale factor compensation can be also applied to the existing on-line CORDIC algorithms.

2. THE NEW RADIX-4 CORDIC ALGORITHM FOR ROTATION MODE

Here, the new redundant CORDIC algorithm to be proposed is based on the fast signed-digit addition (SDA) [12]. The proposed radix-4 rotation mode algorithm for initial vector of $[X_0,Y_0]$ to be rotated by an angle of Z_0 is given as follows:

for $i=0$ to $n/2+1$

$$X_{i+1}=X_i+\delta_i 2^{-2i-1}Y_i, \qquad Y_{i+1}=Y_i-\delta_i 2^{-2i-1}X_i,$$

$$R_{i+1}=4(R_i-2^{2i}\tan^{-1}\delta_i 2^{-2i-1})=2^{2(i+1)}Z_{i+1},$$

The final scale factor is

$$K^{-1}=\prod_{i=0}^{n/2} 1/\sqrt{(1+\delta_i^2 2^{-4i-2})}=$$

$$\begin{cases} \prod_{i=1}^{n/2}(1+s_i 2^{-2i-1}) & \text{if } \delta_0=0 \\ (1-2^{-2})\prod_{i=1}^{n/2}(1+s_i 2^{-2i-1}) & \text{if } \delta_0\neq 0 \end{cases}.$$

A simple selection rules (derived in Appendix B) for δ_i is as follows,

$$\delta_i=\begin{cases} 2 & \text{if } \hat{Ri}>5/8 \\ 1 & \text{if } 1/4\le \hat{Ri}\le 5/8 \\ 0 & \text{if } -1/4<\hat{Ri}<1/4 \\ -1 & \text{if } -5/8\le \hat{Ri}\le -1/4 \\ -2 & \text{if } \hat{Ri}<-5/8 \end{cases}$$

where $\hat{Ri}$ consists of the three most-significant fractional digits of R_i. On the other hand, a simple selection rule (derived in Appendix A) for s_i can be obtained by defining the following iterative operations:

$$W_{i+1}=4[W_i-2^{2i+1}\ln(1+\delta_{i+1}^2 2^{-4(i+1)})-2^{2i}\ln(1+s_i 2^{-2i-1})],$$

$$A_{i+1}=A_i(1+s_i 2^{-i})$$

where $W_0=-2^{-1}\ln(1+\delta_0^2 2^{-2})$, $\delta_0\in\{0,\pm1,\pm2\}$, $A_0=1$, $K^{-1}=A\lceil n/2+1\rceil$ for n-bit precision.

$$s_i=\begin{cases} 2 & \text{if } 3/4<\hat{Wi} \\ 1 & \text{if } 1/4\le \hat{Wi}\le 3/4 \\ 0 & \text{if } -1/4<\hat{Wi}<1/4 \\ -1 & \text{if } -13/16\le \hat{Wi}\le -1/4 \\ -2 & \text{if } \hat{Wi}<-13/16 \end{cases},$$

where $\hat{w_i}$ is the five most-significant fractional digits of W_i. The scale factor compensation can then be combined with rotation iteration or executed after all the rotation iterations are finished.

3. THE NEW RADIX-4 CORDIC ALGORITHM FOR VECTORING MODE

Since the iterated vectors are scaled in magnitude in each iteration and can only be tested after rotation, the decision operations are slower and more complicated than that of the rotation mode. For this reason, the proposed new vectoring mode algorithm is still a hybrid radix-2 and radix-4 one. However, the new algorithm reduces radix-2 iterations to four which is much smaller than the existing $n/2$. Derivation of the new algorithm is more involved than and similar to the rotation mode algortihm.

The new algorithm starts with four radix-2 iterations based on the Ercegovac and Lang's algorithm [7], followed by $(n-4)/2+1$ radix-4 iterations based on a fixed selection rule as follows, for $i=0,1,\ldots,(n-4)/2$

$$\delta_i^*=\begin{cases} 2 & \text{if } \hat{Wi}^*>3\hat{X}4/2 \\ 1 & \text{if } 1/2\hat{X}4\le \hat{Wi}^*\le 3\hat{X}4/2 \\ 0 & \text{otherwise} \\ -1 & \text{if } -3\hat{X}4/2\le \hat{Wi}^*\le -\hat{X}4/2 \\ -2 & \text{if } \hat{Wi}^*<-3\hat{X}4/2 \end{cases},$$

$$X_{i+1}^*=X_i^*+\delta_i^* 2^{-4i-2r}W_i^*,$$

$$W_{i+1}^*=4(W_i^*-\delta_i^* X_i^*)=4^{i+1}Y_{i+1}^*,$$

$$Z_{i+1}^*=Z_i^*+\tan^{-1}\delta_i^* 2^{-2i-r}$$

where $\hat{w}_i^*$ and $\hat{X}4$ are the *6* and *5* most significant fractional digits of w_i^* and X_4 respectivley, and

$X_0^* = X_4$, $Y_0^* = Y_4$, $Z_0^* = Z_4$; X_4, Y_4, Z_4 are the results from the first 4 radix-2 iterations. The final result $X_{(n-4)/2+1}^* = K, R = K\sqrt{X_0^2 + Y_0^2}$, $Z_{(n-4)/2+1}^* = \tan^{-1} X_0/Y_0$. The resulted variable scale factor decomposition can be performed similarly to the rotation mode on-line decomposition algorithm.

4. PERFORMANCE COMPARISONS

To compare different redundant CORDICs for rotation mode, we assume that a basic iteration step consists of a shift operation and a 4-2 SDA. Combined with CORDIC rotation iterations, the new scale factor decomposition algórtithm can compensate the final results in two different schemes:

- ***Scheme-I***: The *n/2* additional shift-and-add compensation operations are performed right after all the *n/2* redundant CORDIC iterations have been done, namely
$$X_{i+1}^* = (1 + s_i 2^{-2i-1}) X_i^*, \quad Y_{i+1}^* = (1 + s_i 2^{-2i-1}) Y_i^*$$
where $X_0^* = X_{n/2}$ and $Y_0^* = Y_{n/2}$, $X_{n/2}$ and $Y_{n/2}$ are the CORDIC rotation results before scale factor compensation. The final compensated results are $X_{n/2+1}^*$ and $Y_{n/2+1}^*$. Consequently, both rotation and compensation operations need *n/2* shift-and-add operations. However, the probability of nonzero δ_i, $s_i \in \{-2,-1,0,1,2\}$ is 4/5. As a result, in average there are *(n/2)*×2×*4/5=0.8n* shift-and-add operations.
- ***Scheme-II***: Each compensation iteration is performed and combined with the rotation iteration immedialtely after its corresponding s_i has been determined, that is
$$X_{i+1} = (1 + s_i 2^{-2i-1})(X_i - \delta_i 2^{-2i-1} Y_i)),$$
$$Y_{i+1} = (1 + s_i 2^{-2i-1})(Y_i + \delta_i 2^{-2i-1} X_i.$$
Similarly, there are *0.8n* shift-and-add operations for this CORDIC operations.

On the other hand, in average, the hybrid radix-2 and radix-4 algortihms in [5] and [10] needs *n/2* radix-2 iterations, *(n/4)* ×*4/5* radix-4 iterations and *n/4* iterations for scale factor compensation. These amount to *0.95n* shift-and-add operations for those two algorithms. Note that the algorithm in [10] is based on slower non-redundant additions.

The double rotation method [4] needs 2.25*n* basic steps where 2*n* steps for rotations and 0.25*n* for scale factor compensation. The branch CORDIC algorithm [6] needs 1.25*n* basic steps where *n* steps for rotations and 0.25*n* for scale factor compensation. However, this algorithm needs two copies of CORDIC operated in parallel. Hence, in fact this algorithm needs *2.5n* basic steps.

Table 1 summaries the comparison results. In the table, all the algorithms are assumed realized with unfolded (sequential) hardwares, which mainly consist of required barrel shifters, adders and ROM tables excluding other minor components. As shown, the new radix-4 redundant CORDIC have the best performance. The comparison statistic for all existing vectoring-mode CORDICs have the similar performance results as the rotation mode.

5. CONCLUSION

The new CORDIC algorithm achieves the best performance among all the existing algorithms in terms of iteration number and hardware complexity. The algorithm can be applied to the computation of hyperbolic functions as well. Moreover, the new algorithm includes a ROM table of $\ln(1 + s_i 2^{-2i-1})$ which can be utilized to compute logarithm and exponential functions, and in turns the hyperbolic functions by using the well-known CCM algorithm. Doing this way, no scale factor compensation is required. As a result, a unified algorithm for the computation of a broad set of elementary functions can be obtained, which is under further investigation.

REFERENCES

[1] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electronic Comput.*, vol. EC-8, no. 3, pp. 330-334, 1959.

[2] J. S. Wather, "A unified algorithm for elementary functions," *AFIPS Spring Joint Comput. Conf.*, pp. 379-385, 1971.

[3] J. R. Cavallaro and N. D. Hemkumar, "Redundant and On-line CORDIC for Unitary Transformations," *IEEE Trans. Comput.*, vol. 43, no. 8, pp. 941-954, August 1994.

[4] N. Takagi, et al., "Redundant CORDIC methods with a constant scale factor for sine and cosine computation," *IEEE Trans. Comput.*, vol. 40, no. 9, pp. 989-995, Sept. 1991.

[5] J. A. Lee and T. Lang, "Constant-Factor Redundant CORDIC for Angle Calculation and Rotation," *IEEE Trans. Comput.*, vol. 41, no. 8, pp. 1016-1015.

[6] J. Duprat and J. M. Muller, "The cordic algorithm: New results for fast VLSI implementation," *IEEE Trans. on Comput.*, vol. 42, no. 2, pp. 168-178, Feb. 1993.

[7] M. D. Ercegovac and T. Lang, "Redundant and online CORDIC: Application to matrix triangularization and SVD," *IEEE Trans. Comput.*, vol. 39, n0. 6, pp. 725-740, June 1990.

[8] H. X Lin and H. J. Sips, "ON-Line CORDIC Algorithms," *IEEE Trans. Comput.*, vol. 39, no. 8, pp.

1038-1052, Aug. 1990.

[9] T. C. Chen, "Automatic computation of logarithms, exponential, ratios and square roots," *IBM J. Res. and Dev.*, vol. 16, pp. 380-388, 1972.

[10] M. R. D. Rodrigues et. al, "Hardware evaluation of mathematical functions," *IEE Proc.*, vol. 128, Pt. E, no. 4, July 1981.

[11] M. D. Ercegovac, "On-line arithmetic: An overview," in *Proc. SPIE Real-Time Signal Processing*, 495 (VII), Aug. 1984, pp. 86-93.

[12] K. Hwang, *Computer Arithmetic, Principles, Architecture, and Design.* New York: Wiley, 1979.

[13] D. Timmermann et. al, "Low Latency Time CORDIC Algorithms," *IEEE Trans. Comput.*, vol. 41, no. 8, pp. 110-115, Aug. 1992.

APPENDIX A

Derivation of the radix-4 on-line decomposition algorithm for variable scale factor

The decisoin of $s_i=k$ should make W_{i+1} remain bounded in $[L_{-2},U_2]$ whenever W_i is in the interval $[L_k,U_k]$. Then the following equations have to be satisfied:

$$U_2 = 4(U_k - 2^{2i}\ln(1+k2^{-2i-1}))$$

$$L_{-2} + 2^{2i+1}\ln(1+2^{-4(i+1)}) = 4(L_k - 2^{2i}\ln(1+k2^{-2i-1})).$$

By substituting k=-2, -1, 0, 1, 2 successively into the above equations, the exact bounds for $i\geq 2$ are found to be:

$$U_2 = \frac{2^{2i+2}}{3}\ln(1+2^{-2i}) \geq 1.2933 \text{ ,} U_0 = \frac{2^{2i}}{3}\ln(1+2^{-2i}) \geq 0.3233$$

$$U_1 = \frac{2^{2i}}{3}\ln(1+2^{-2i}) + 2^{2i}\ln(1+2^{-2i-1}) \geq 0.8157$$

$$U_{-1} = \frac{2^{2i}}{3}\ln(1+2^{-2i}) + 2^{2i}\ln(1-2^{-2i-1}) \geq -0.1846$$

$$U_{-2} = \frac{2^{2i}}{3}\ln(1+2^{-2i}) + 2^{2i}\ln(1-2^{-2i}) \geq -0.7093 \cdot$$

Similarly,

$$L_{-2} = \leq -4/3, \quad L_{-1} = \leq -5/6, L_0 = \leq -1/3$$

$$L_1 = \leq 1/6, \quad L_2 = \leq 2/3$$

It can be shown that the overlap intervals $[L_2,U_1]$, $[L_1,U_0]$, $[L_0,U_{-1}]$ exist for all i, while $[L_{-1},U_{-2}]$ exists for $i\geq 2$. With the overlap regions, a simple selection rule for s_i can be obtained as shown in the second section. The term $\ln(1+s_i 2^{-2i-1})$ doesn't exist when i=0 and s_0= -2. To solve this problem and ensure convergence, we introduce the initial steps as follows,

$$1 + s_0 2^{-2i-1} = \begin{cases} 1 & \text{if } \delta_0 = 0 \\ (1-2^{-2}) & \text{if } \delta_0 \neq 0 \end{cases}.$$

APPENDIX B

Derivation of selection rule for rotation direction of the radix-4 CORDIC in rotation mode

Similar to the on-line scale factor decompostion algorithm, the decision rules for rotation direction is to make R_{i+1} still bounded if R_i is bounded in the interval $[L_k,U_k]$. Therefore, L_k and U_k can be found from U_2=4(U_k-2^{2i}tan^{-1}$k2^{-2i-1}$) and L_{-2}=4(L_k-2^{2i}tan^{-1}$k2^{-2i-1}$). And the smallest (largest) values of U_k (L_k) can be found by letting i=0. Specifically,

$$U_2 = -L_{-2} \geq \frac{\pi}{3}, \quad U_1 = -L_{-1} \geq 0.7254,$$

$$U_0 = -L_0 = \geq \frac{\pi}{12}, U_{-1} = -L_1 \geq -0.2019,$$

$U_{-2} = -L_{-2} \geq -0.5235$. From the overlap intervals of $[L_2,U_1]$, $[L_1,U_0]$, $[L_0,U_{-1}]$ and $[L_{-1},U_{-2}]$, a simple decision rule for δ_i can be deduced as shown in the second section.

Table 1. Comparisons of the rotation-mode redundant CORDIC algorithms.

Algorithm	**New CORDIC**	Lee & Tang's CORDIC [5]	Rodrigues' CORDIC [10]	Branch CORDIC [6]	Double rotation CORDIC [4]
Area	***~A***	*~A*	*~A*	*~2A*	*~A*
No. of Shift-&-add steps	***0.8n***	*0.95n*	*0.95n*	*1.25n*	*2.25n*