A Memory-Efficient and High-Speed Sine/Cosine Generator Based on Parallel CORDIC Rotations

Shen-Fu Hsiao, *Member, IEEE*, Yu-Hen Hu, *Fellow, IEEE*, and Tso-Bing Juang, *Member, IEEE*

***Abstract*—The sine/cosine function generator is based on parallelization of the original CORDIC algorithm by predicting all the rotation directions directly from the binary bits of the initial input angle. Unlike previous approaches that require complicated circuits or exponentially increased ROM, our proposed architecture has a relatively simple prediction scheme through an efficient angle recoding. The critical path delay is also reduced by utilizing the predicted rotation directions to design an efficient multioperand carry–save addition structure.**

***Index Terms*—CORDIC, microrotation angle recoding, multioperand carry–save addition, parallel sign prediction, sine/cosine function generator.**

I. Introduction

One of the key components in direct digital frequency synthesizer (DDFS) system [1] is the sine/cosine function generator that computes binary representation of $\sin\theta$ and $\cos\theta$ to a precision of N fractional bits is. In this letter, we propose a novel realization of a sine/cosine generator based on the CORDIC algorithm [2]. CORDIC is an arithmetic algorithm developed to compute various elementary functions through a series of iterations of a unified *microrotations* operation. In particular, in a circular rotation mode, N microoperations as illustrated below will be executed for $i = 1, 2, \cdots, N$

$$\begin{cases} x_{i+1} = x_i + \sigma_i 2^{-i} y_i \\ y_{i+1} = y_i - \sigma_i 2^{-i} x_i \\ z_{i+1} = z_i - \sigma_i \tan^{-1}(2^{-i}) \\ \qquad = z_i - \sigma_i \alpha_i \end{cases}, \qquad \sigma_i = \operatorname{sign}(z_i) \in \{1, -1\}. \tag{1}$$

After N iterations, the accumulated rotation angle is

$$\theta = z_1 - z_{N+1} = \sum_{i=1}^{N} \sigma_i \tan^{-1}(2^{-i}) \equiv \sum_{i=1}^{N} \sigma_i \alpha_i.$$

Using the definition of α_i, one has (note that $\cos \sigma_i\alpha_i = \cos\alpha_i$)

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & \sigma_i 2^{-i} \\ -\sigma_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \frac{1}{\cos\alpha_i} \begin{bmatrix} \cos\sigma_i\alpha_i & \sin\sigma_i\alpha_i \\ -\sin\sigma_i\alpha_i & \cos\sigma_i\alpha_i \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}.$$

Then, it can be easily deduced that

$$\begin{bmatrix} x_{N+1} \\ y_{N+1} \end{bmatrix} = \frac{1}{K} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$$

where $K = \prod_{i=1}^{N} \cos\alpha_i = \prod_{i=1}^{N}(1 + 2^{-2i})^{-1/2}$ is a constant that can be precomputed in advance. Set $x_1 = K$, $y_1 = 0$, $z_1 = \theta$, then $x_{N+1} = \cos\theta$, $y_{N+1} = \sin\theta$ can be easily computed after N iterations. In conventional CORDIC, the direction $\sigma_i = \operatorname{sign}(z_i)$ is determined sequentially since it depends on the sign of z_i calculated at the previous iteration. This dependence relation makes it difficult to execute multiple microrotations in parallel. In this letter, we propose a new method to quickly select the rotation directions $\{\sigma_i\}$ in order to speed up the calculation.

II. Predictions of Rotation Directions

A. Binary to Bipolar Recoding (BBR)

The initial input angle $\theta = (-\theta_0) + \sum_{j=1}^{N} \theta_j 2^{-j}$ with $\theta_j \in \{0, 1\}$ is assumed to be in the range $|\theta| < \pi/4$ as in the application example of DDFS. It has been shown in [3] that $\tan^{-1} 2^{-i} = 2^{-i}$ to N-bit precision if $i \geq m = \lceil (N - \log_2 3)/3 \rceil$. Thus, the last $2N/3$ rotation directions (from σ_m to σ_N) can be obtained in parallel after completing the first $N/3$ iterations. As proposed in [3], we divide the angle into two parts (the higher part and the lower part)

$$\theta = \theta_H + \theta_L = \underbrace{(-\theta_0) + \sum_{j=1}^{m-1} \theta_j 2^{-j}}_{\theta_H} + \underbrace{\sum_{j=m}^{N} \theta_j 2^{-j}}_{\theta_L}. \tag{2}$$

The binary bits $\theta_j \in \{0, 1\}$ in the higher part θ_H can be recoded into bipolar digits as follows:

$$\begin{aligned} \theta_H &= (-\theta_0) + \sum_{j=1}^{m-1} \theta_j 2^{-j} \\ &= (-\theta_0) + \sum_{j=1}^{m-1} \left[2^{-j-1} + (2\theta_j - 1)2^{-j-1}\right] \\ &= (-\theta_0) + 2^{-1} + \sum_{k=2}^{m} r_k 2^{-k} - 2^{-m} \\ &\qquad \text{where } r_k = (2\theta_k - 1) \in \{1, -1\}. \end{aligned} \tag{3}$$

Equation (3) is called BBR for $\theta_j, j = 0, 1, \ldots, m-1$.

B. Microrotation Angle Recoding (MAR)

Since $\tan^{-1}(2^{-i}) \neq 2^{-i}$ for $i = 1, \cdots, m-1$, we decompose each positional binary weighting 2^{-i}, $i = 1, \cdots, m-1$ into the combination of significant $\tan^{-1}(2^{-j})$ terms plus

Manuscript received October 21, 2002; revised April 8, 2003. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Richard C. Kavanagh.

S.-F. Hsiao and T.-B. Juang is with the Department of Computer Science and Engineering, National Sun Yat-sen University, 807 Kaohsiung, Taiwan, R.O.C. (e-mail: sfhsiao@cse.nsysu.edu.tw; tsobing@garfield.cse.nsysu.edu.tw).

Y.-H. Hu is with the Department of Electrical and Computer Engineering, University of Wisconsin-Madison, Madison, WI 53706-1691 USA (e-mail: hu@engr.wisc.edu).

Digital Object Identifier 10.1109/LSP.2003.821705

an error term e_i collecting all the other insignificant values of $\tan^{-1}(2^{-j})$, $j > m$. For simplicity, we take $N = 24$ as an example where $m = \lceil (N - \log_2 3)/3 \rceil = 8$. The microrotation angle recoding from 2^{-i} to $\tan^{-1}(2^{-i})$, $i = 1, \cdots, 7$ is

$$\begin{aligned}
2^{-1} &= \tan^{-1}(2^{-1}) + \tan^{-1}(2^{-5}) + \tan^{-1}(2^{-8}) \\
&\quad + \underbrace{0.000\,000\,000\,100\,111\,100\,001\,110_2}_{e_1} \\
2^{-2} &= \tan^{-1}(2^{-2}) + \tan^{-1}(2^{-8}) \\
&\quad + \underbrace{0.000\,000\,000\,100\,100\,100\,010\,100_2}_{e_2} \\
2^{-3} &= \tan^{-1}(2^{-3}) + \underbrace{0.000\,000\,000\,010\,101\,001\,000\,101_2}_{e_3} \\
2^{-4} &= \tan^{-1}(2^{-4}) + \underbrace{0.000\,000\,000\,000\,010\,101\,010\,010_2}_{e_4} \\
2^{-5} &= \tan^{-1}(2^{-5}) + \underbrace{0.000\,000\,000\,000\,000\,010\,101\,010_2}_{e_5} \\
2^{-6} &= \tan^{-1}(2^{-6}) + \underbrace{0.000\,000\,000\,000\,000\,000\,010\,101_2}_{e_6} \\
2^{-7} &= \tan^{-1}(2^{-7}) + \underbrace{0.000\,000\,000\,000\,000\,000\,000\,010_2}_{e_7}. \qquad (4)
\end{aligned}$$

The BBR for θ_H with $N = 24$ is

$$\theta_H = (1 - 2\theta_0)2^{-1} + \sum_{k=2}^{8} r_k 2^{-k} - 2^{-8}. \qquad (5)$$

The first eight rotation directions are selected concurrently as

$$\begin{aligned}
\sigma_1 &= (1 - 2\theta_0) \\
\sigma_k &= r_k = (2\theta_{k-1} - 1), \quad k = 2, \cdots, 8. \qquad (6)
\end{aligned}$$

Then, all the signed error terms $\sigma_i e_i$, $i = 1, \ldots, 7$ and the last term -2^{-8} in (5) are added to θ_L, generating the corrected lower part $\hat{\theta}_L$ represented in twos complement format, i.e.,

$$\begin{aligned}
\hat{\theta}_L &= \theta_L + \sum_{i=1}^{7} \sigma_i e_i - 2^{-8} \\
&= (-\hat{\theta}_7)2^{-7} + \sum_{k=8}^{24} \hat{\theta}_k 2^{-k}, \quad \hat{\theta}_k \in \{0, 1\},\ k = 8, \ldots, 24. \qquad (7)
\end{aligned}$$

It can be shown that $|\hat{\theta}_L| < 2^{-7}$. Since $\tan^{-1} 2^{-i} = 2^{-i}$, $i \geq 8$ within precision of 24 fractional bits, the algorithm converges after the above selection of directions for the first several rotations. The directions for the remaining microrotations can be derived immediately from (7) using again the BBR

$$\begin{aligned}
\hat{\theta}_L &= (-\hat{\theta}_7)2^{-7} + \sum_{k=8}^{24} \hat{\theta}_k 2^{-k} \\
&= (-\hat{\theta}_7)2^{-7} + \sum_{k=9}^{25} (2\hat{\theta}_{k-1} - 1)2^{-k} + 2^{-8} - 2^{-25} \\
&= (1 - 2\hat{\theta}_7)2^{-8} + \sum_{k=9}^{25} \hat{r}_k 2^{-k} - 2^{-25} \\
\hat{r}_k &= (2\hat{\theta}_k - 1) \in \{1, -1\} \qquad (8)
\end{aligned}$$

leading to the parallel prediction

$$\begin{aligned}
\hat{\sigma}_8 &= (1 - 2\hat{\theta}_7) \\
\hat{\sigma}_k &= \hat{r}_k = (2\hat{\theta}_k - 1), \qquad k = 9, \cdots, 25
\end{aligned}$$

for the last $2N/3$ microrotations.

III. Implementation and Comparison

Fig. 1(a) is a 24-bit sine/cosine generator based on unfolded CORDIC architecture where each numbered block (a stage) denotes a microrotation performing the recurrence of x and y in (1). Note that the microrotation of angle $\tan^{-1}(2^{-5})$ is repeated once and the microrotation of angle $\tan^{-1}(2^{-8})$ is repeated three times. The scaling factor is still kept constant by taking into account these fixed repetitions

$$K' = \prod_{i=1}^{N} (1 + 2^{-2i})^{-\frac{1}{2}} \times (1 + 2^{-10})^{-\frac{1}{2}} \times (1 + 2^{-16})^{-\frac{3}{2}}.$$

This constant can be precomputed and serves as one of the initial inputs $x_1 = K'$, $y_1 = 0$, $z_1 = \theta$.

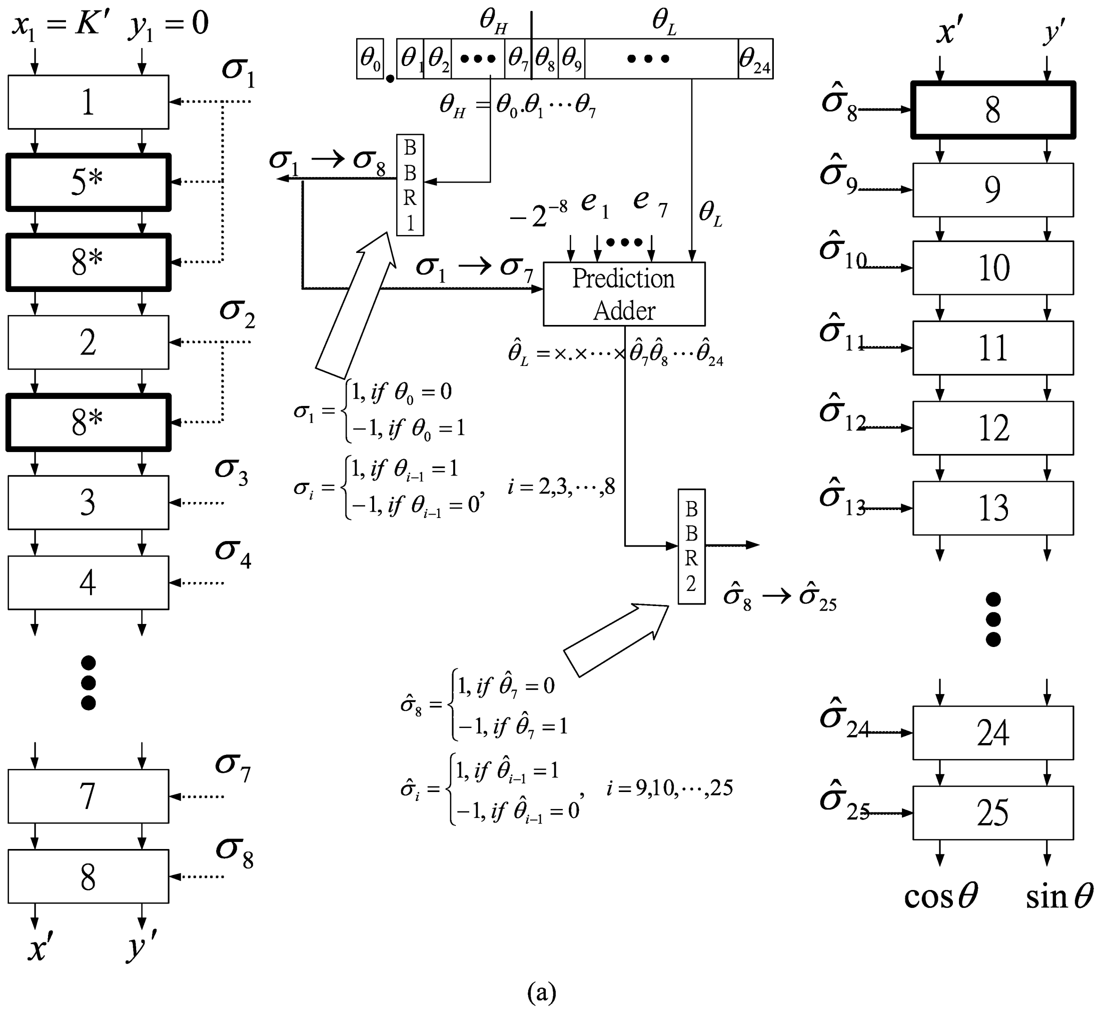
Since the first m bits of the input angle are directly used as the directions in the first m rotations, the execution of stages 1–12 (including repetition stages) in the left of Fig. 1(a) can be performed in parallel with the prediction adder that generates the directions for the remaining microrotations. We adopt carry–save addition (CSA) in each stage where a 4 : 2 compressor is used to produce the carry–save form (a sum term plus a carry term) for each output, as shown in Fig. 1(b). Assuming the delay of a 4 : 2 compressor to be $2T_{\mathrm{FA}}$ where T_{FA} is the delay of a full adder, the delay for the stages in the left of Fig. 1(a) is $20T_{\mathrm{FA}}$. Note that the first stage does not need CSA. The prediction adder is to calculate the sum of the nine operands in (7), using a CSA tree and a fast carry-propagate adder (CPA), leading to a delay of $4T_{\mathrm{FA}} + T_{\mathrm{CPA}}$ where T_{CPA} denotes the delay of a CPA. In general, the prediction adder is not in the critical path as long as a fast CPA is used.

The derivation in Section II can be easily extended to different bit precision N where the total number of repetitions $l(N)$ can be found to be $l(16) = 2$, $l(24) = 4$, $l(32) = 7$. In some application (such as the DDFS) where the input angle is further limited in $0 \leq \theta < \pi/4$, the first rotation direction is always $\sigma_1 = 1$, and thus the first several stages controlled by σ_1 can be merged and precomputed along with the constant factor K' as the initial input to the X/Y datapath. In this situation, the numbers of repetitions in the sine/cosine generation are reduced to $l(16) = 1$, $l(24) = 2$, $l(32) = 4$.

The second half of microrotation stages can be merged into a multioperand carry–save addition architecture by observing that

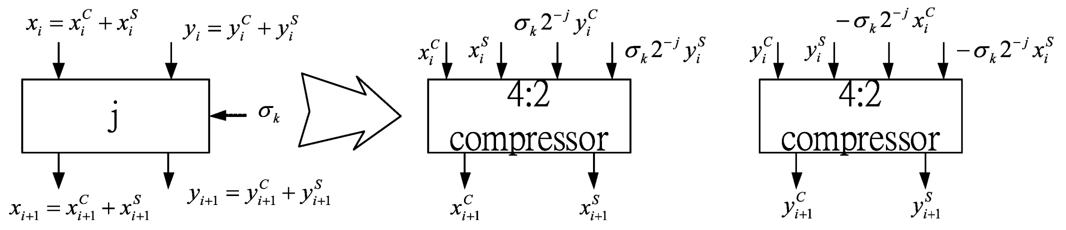
$$\begin{cases} x_{k+l} = x_k + y_k \sum\limits_{i=k}^{k+l-1} r_i 2^{-i} \\ y_{k+l} = y_k - x_k \sum\limits_{i=k}^{k+l-1} r_i 2^{-i} \end{cases}, \quad k > \frac{N}{2}, \quad l = 1, 2, \ldots. \qquad (9)$$

Thus, the microrotation stages numbered from 13–25 in Fig. 1(a) can be merged into a CSA tree performing the parallel addition of 28 operands (14 numbers in carry–save forms) with critical path delay $7T_{\mathrm{FA}} + T_{\mathrm{CPA}}$. Summing up the delay of all the stages, the total delay of our proposed sine/cosine



$x_1 = K'$ $y_1 = 0$
σ_1
1
5*
8*
σ_2
2
8*
σ_3
3
σ_4
4
σ_7
7
σ_8
8
x' y'
θ_H
θ_L
θ_0 θ_1 θ_2 θ_7 θ_8 θ_9 θ_{24}
$\theta_H = \theta_0.\theta_1 \cdots \theta_7$
$\sigma_1 \to \sigma_8$
BBR1
-2^{-8} e_1 e_7
θ_L
$\sigma_1 \to \sigma_7$
Prediction Adder
$\hat{\theta}_L = \times.\times \cdots \times \hat{\theta}_7 \hat{\theta}_8 \cdots \hat{\theta}_{24}$
$\sigma_1 = \begin{cases} 1, if\ \theta_0 = 0 \\ -1, if\ \theta_0 = 1 \end{cases}$
$\sigma_i = \begin{cases} 1, if\ \theta_{i-1} = 1 \\ -1, if\ \theta_{i-1} = 0 \end{cases}, \quad i = 2,3,\cdots,8$
BBR2
$\hat{\sigma}_8 \to \hat{\sigma}_{25}$
$\hat{\sigma}_8 = \begin{cases} 1, if\ \hat{\theta}_7 = 0 \\ -1, if\ \hat{\theta}_7 = 1 \end{cases}$
$\hat{\sigma}_i = \begin{cases} 1, if\ \hat{\theta}_{i-1} = 1 \\ -1, if\ \hat{\theta}_{i-1} = 0 \end{cases}, \quad i = 9,10,\cdots,25$
x' y'
$\hat{\sigma}_8$ 8
$\hat{\sigma}_9$ 9
$\hat{\sigma}_{10}$ 10
$\hat{\sigma}_{11}$ 11
$\hat{\sigma}_{12}$ 12
$\hat{\sigma}_{13}$ 13
$\hat{\sigma}_{24}$ 24
$\hat{\sigma}_{25}$ 25
$\cos\theta$ $\sin\theta$

(a)



$x_i = x_i^C + x_i^S$
$y_i = y_i^C + y_i^S$
j
σ_k
$x_{i+1} = x_{i+1}^C + x_{i+1}^S$
$y_{i+1} = y_{i+1}^C + y_{i+1}^S$
x_i^C x_i^S $\sigma_k 2^{-j} y_i^C$ $\sigma_k 2^{-j} y_i^S$
4:2 compressor
x_{i+1}^C x_{i+1}^S
y_i^C y_i^S $-\sigma_k 2^{-j} x_i^C$ $-\sigma_k 2^{-j} x_i^S$
4:2 compressor
y_{i+1}^C y_{i+1}^S

(b)

Fig. 1. (a) Architecture of a 24-bit sine/cosine generator based on parallel CORDIC rotations and (b) the carry–save addition implementation in a stage.

generator is $37T_{\mathrm{FA}} + T_{\mathrm{CPA}}$ for 24-bit accuracy, a significant speed improvement compared with other previous approaches as will be discussed in the following.

Table I compares our proposed sine/cosine generator with other CORDIC rotation algorithms. To make a fair comparison, we assume that all methods use CSA in X/Y datapath except for the last stage where a fast CPA is required to obtain the nonredundant representation of the output (T_{CPA} is not counted in Table I for reason of clarity). In [3], the rotation directions after m iterations are derived from the z remainder. However, the first m iterations still adopt the conventional sequential approach where a delay of $\lceil \log_2 N \rceil \times T_{\mathrm{FA}}$ is assumed for an N-bit CPA. In [4], two rotations are executed in a single step at the price of more complicated Z datapath where several most significant digits are examined. In [5], the first m rotation directions are predicted based on approximation of binary angle input to $\tan^{-1} 2^{-i}$, similar to our method. But the direction prediction requires several carry-look-ahead adders plus complicated logic circuits instead of directly from the binary bits of the input angle as in our method. In [6], the first several directions are derived using a ROM of size exponentially increased with N. Unlike these above methods that require some delay to predict the first m rotation directions, our proposed method generates the first m rotation directions immediately from the first m binary bits of

TABLE I
COMPARISON OF N-BIT CORDIC-BASED SINE/COSINE GENERATORS WITH DIFFERENT PREDICTION SCHEMES

method	Delay (in units of T_{FA})	Delay (N=16)	Delay (N=24)	Delay (N=32)	Z-path	ROM	Prediction hardware
ours	$(N+2l(N)+\left\lceil \log_{3/2}(\frac{N}{2}) \right\rceil-2)$	$24T_{FA}$	$37T_{FA}$	$51T_{FA}$	no	no	simple
[3]	$(\lceil \log_2 N \rceil \times N/3+4N/3)$	$43T_{FA}$	$72T_{FA}$	$96T_{FA}$	yes	no	simple
[4]	$5N/2$	$40T_{FA}$	$60T_{FA}$	$80T_{FA}$	yes	no	complex
[5]	$(N+\lceil (\log_3 N)-2 \rceil)\times 2$	$34T_{FA}$	$50T_{FA}$	$68T_{FA}$	no	no	complex
[6]	$(1.625N+N/12+\log_2 N+1)$	$33T_{FA}$	$47T_{FA}$	$61T_{FA}$	no	$2^{N/3}$	simple

the input angle. As of area comparison, our proposed ROM-free CSA architecture also requires less hardware complexity compared with the other methods mentioned above.

A 16-bit CORDIC-based sin/cosine generator similar to the architecture in Fig. 1 was synthesized and mapped on the Virtex-300 type FPGA chip. The critical path delay across the entire architecture is less than 75 ns. Compared with the approach in [3], our design achieves more than 25% improvement in speed performance (due to the parallelization of the sign bit selection), and 30% saving in hardware cost (due to the elimination of the Z datapath). Another comparison is made for the ROM-based approach in [6] that calls for 47 configurable logic blocks (CLBs) for the polarity prediction, our proposed sin/cosine generator, with only 28 CLBs for the sign-bit determination, saves more than 40% hardware cost in the prediction of rotation directions.

IV. CONCLUSION

We presented a novel recoding method to predict all the directions of CORDIC microrotations and apply it to the generation of sine and cosine functions. The proposed architecture does not need exponentially increased ROM or complicated prediction hardware. The speed is also improved by implementing the microrotation stages using carry–save addition with reduced number of operands.

REFERENCES

[1] A. Madisetti, Y. Kwentus, and A. N. Willson, Jr., "A 100-MHz, 16-b, direct digital frequency synthesizer with a 100-dBc spurious-free dynamic range," *IEEE J. Solid-State Circuits*, vol. 34, pp. 1034–1043, Aug. 1999.

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

[3] S. Wang, V. Piuri, and E. E. Swartzlander, "Hybrid CORDIC algorithms," *IEEE Trans. Comput.*, vol. 46, pp. 1202–1207, Nov. 1997.

[4] D. S. Phatak, "Double step branching CORDIC: A new algorithm for fast sine and cosine generation," *IEEE Trans. Comput.*, vol. 47, pp. 587–602, May 1998.

[5] J. H. Kwak and E. E. Swartzlander, Jr., "A new scheme for prediction of rotation directions in CORDIC processing," in *Proc. 42nd Midwest Symp. on Circuits and Systems*, 1999, pp. 870–873.

[6] M. Kuhlmann and K. K. Parhi, "P-CORDIC: A precomputation based rotation CORDIC algorithm," *EURASIP J. Appl. Signal Process.*, vol. 2002, pp. 936–943, Sept. 2002.