A FAST DIRECTION SEQUENCE GENERATION METHOD FOR CORDIC PROCESSORS

Seunghyeon Nahm *Wonyong Sung*

School of Electrical Eng., Seoul National University
Shinlim-dong, Gwanak-gu, Seoul 151-742, KOREA
namsh@hdtv.snu.ac.kr, wysung@dsp.snu.ac.kr

ABSTRACT

This paper describes a new direction sequence generation method for the circular CORDIC algorithm. A conventional approach employs an angle computation algorithm to control the direction of rotation in the form of a sign sequence, where the sign generation is a bottle-neck for the fast implementations. The proposed method reduces the number of sequential computations by employing a new angle representation model and linearizing the arctangent function in small angles. The direction sequence can be generated by about a third of the iterative computations required in the conventional algorithm, which also reduces the hardware requirements as much. Especially, this algorithm is attractive when pipelining is not allowed for feedback control, such as found in phase tracking applications. A VLSI implementation example for a high-speed quadrature demodulator is also discussed.

1. INTRODUCTION

The CORDIC (COordinate Rotation DIgital Computer) is an iterative arithmetic algorithm for computing many elementary functions, which frequently appear in modern signal processing computing algorithms, such as the Toeplitz system solver, the lattice filter, the QR factorization, and the fast Fourier transform[1]. The basic operation is to rotate a 2×1 vector through an angle using a linear, circular, or hyperbolic coordinate system. This is accomplished by rotating the vector through a sequence of elementary angles whose algebraic sum approximates the desired rotation angle. In this paper, we only consider the circular CORDIC algorithm which can be efficiently used in digital communication systems, such as the DDFS (Direct Digital Frequency Synthesizer)[2].

The basic iterative formulation of the circular CORDIC algorithm can be summarized as follows.

Given a rectangular coordinate, $[x_{in}\, y_{in}]^t$, and the rotation angle, z_{in}, initialize as

$$x(0) = -\mu(0) y_{in} \quad (1)$$
$$y(0) = +\mu(0) x_{in} \quad (2)$$
$$z(0) = z_{in} - \mu(0) a(0) \quad (3)$$

and compute for $i = 1, 2, \ldots, n-1$

$$x(i) = x(i-1) - \mu(i) 2^{1-i} y(i-1) \quad (4)$$
$$y(i) = y(i-1) + \mu(i) 2^{1-i} x(i-1) \quad (5)$$
$$z(i) = z(i-1) - \mu(i) a(i) \quad (6)$$

where $\{\mu(i)\}$ is a sign sequence indicating the direction of rotation and obtained by

$$\mu(i) = \begin{cases} \text{sign}[z_{in}], & i = 0 \\ \text{sign}[z(i-1)], & i \geq 1 \end{cases} \quad (7)$$

and $a(i)$ is the elementary angle and given by

$$a(i) = \begin{cases} \frac{\pi}{2}, & i = 0 \\ \arctan(2^{1-i}), & i \geq 1 \end{cases} \quad (8)$$

A conventional method computes the direction sequence by using Eq. (3) and Eq. (6), which requires the sign detection result in the previous stage. Thus, the sequential nature of the angle computation inhibits the fast implementation of a CORDIC processor. Our previous study on the hardware requirements of the parallel CORDIC processor showed that the angle computation hardware requires about a third of the area occupied by the angle rotation hardware.

In this paper, a new direction sequence generation method which requires much less number of sequential computations is developed. Although the conventional CORDIC algorithm determines the direction of rotation sequentially at each stage, the direction sequence can be determined non-recursively when the remaining angle to rotate is small. In the proposed method, the direction sequences that correspond to large angles are determined in a sequential manner, but those of small angle portions are generated directly. Madisetti's method also utilizes the linear approximation of tangent function[3], but it requires multiplications for the angle rotation. Our developed method retains the simplicity of the CORDIC angle rotation while not only reducing the angle computation hardware but also improving the speed as well.

2. NEW ANGLE REPRESENTATION MODEL

The actual rotation angle obtainable by the circular CORDIC algorithm is given by $A = \sum_{i=0}^{n-1} \mu(i) a(i)$, where a sign sequence $\mu(i)$ represents the angle A. Recoding $\mu(i)$ by the relation, $\mu(i) = 2b_i - 1$, results in

$$A = \sum_{i=0}^{n-1} (2b_i - 1) a(i) = \sum_{i=0}^{n-1} b_i \cdot 2a(i) - \sum_{i=0}^{n-1} a(i) \quad (9)$$

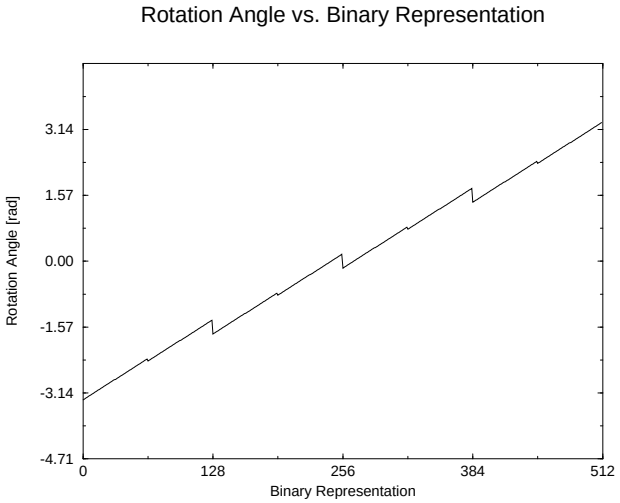

Figure 1. Actual rotation angle vs. binary representation

Table 1. Indices for linear approximation

m	1	2	3	4	5	6
n	4	7	9	12	15	18

where $\{b_i\}$ is a binary sequence which directs the angle rotation in such a way that $b_i = 1$ means rotation by $+a(i)$ and $b_i = 0$, by $-a(i)$. The rotation angle A can be represented by a binary number $B = \sum_{i=0}^{n-1} b_i 2^{n-i-1}$. We named this representation a BSR (Binary Sequence Representation). Figure 1 shows the actual values of A represented by B when $n = 9$. Note that the curve looks piece-wise linear, because the arctangent function is approximately linear for small angles. Using this property, small elementary angles can be approximated as $2a(i) = 2\arctan(2^{1-i}) \approx s \cdot 2^{n-i-1}$ for $i \geq m$, where m is an integer which makes the approximation tolerable and s is the slope of the linearity. Then the following approximation can be derived.

$$\sum_{i=m}^{n-1} b_i \cdot 2a(i) \approx s \cdot \sum_{i=m}^{n-1} b_i \cdot 2^{n-i-1} \qquad (10)$$

For keeping the monotonically increasing tendency of the actual rotation angle according to the increase of the corresponding BSR, the condition of $a(m) > \sum_{i=m+1}^{n-1} a(i)$ should be satisfied. Table 1 shows the computed m and n satisfying the condition, where we can find that m is only about a third of n. Thus, the number of stages which require the sequential determination of the direction control is only a third of the total stages. Dividing Eq. (10) by s gives the following approximation.

$$\begin{aligned}\frac{1}{s}\sum_{i=0}^{n-1} b_i \cdot 2a(i) &\approx \sum_{i=0}^{m-1} b_i \cdot \frac{2a(i)}{s} + \sum_{i=m}^{n-1} b_i \cdot 2^{n-i-1} \\ &= \sum_{i=0}^{n-1} b_i \cdot 2^{n-i-1} - R(b_0 b_1 b_2 \ldots b_{m-1}) \qquad (11)\end{aligned}$$

where the first term is a BSR of the angle and $R(\cdot)$ is the angle adjustment value which relates the BSR and the actual rotation angle. $R(\cdot)$ is only dependent on $b_0 b_1 b_2 \ldots b_{m-1}$ and given by

$$R(b_0 b_1 b_2 \ldots b_{m-1}) = \sum_{i=0}^{m-1} b_i \cdot 2^{n-i-1} - \sum_{i=0}^{m-1} b_i \cdot \frac{2a(i)}{s} \qquad (12)$$

3. DIRECTION SEQUENCE GENERATION

Using the new angle representation model, the direction sequence generation algorithms and its implementation methods are developed. Firstly, the absolute angle rotation case is studied, where the CORDIC algorithm is used for rotating a rectangular coordinate by an absolute angle. Secondly, the accumulated angle rotation case is studied, which can be used for continuous phase rotation applications, such as the phase tracking system.

3.1. Absolute Angle Rotation

When the rotation angle A is required, we have to find a binary sequence $\{b_i\}$ which satisfies $\sum_{i=0}^{n-1} b_i \cdot 2a(i) = (A + A_0) \bmod 2\pi$, where $A_0 = \sum_{i=0}^{n-1} a(i)$ is a constant. This value should be normalized to the level of the BSR by multiplying $1/s$. The first m bits of the BSR cannot be directly obtained due to the nonlinearity of the arctangent function, thus it needs to be obtained by the comparison with the elementary angles. The rest of the bits can be directly determined because the linear approximation is valid. The following algorithm summarizes this procedure.

Algorithm 1 *BSR computation*

$\hat{A} = (A + A_0) \bmod 2\pi$;
$\hat{B} = \frac{1}{s}\hat{A}$;
for $i = 0$ to $m - 1$ do begin /* recursive
    if $\hat{B} > \frac{2a(i)}{s}$ then computing */
        $\hat{B} = \hat{B} - \frac{2a(1)}{s}$;
        $b_i = 1$
    else
        $b_i = 0$;
end for;
$B = \hat{B}$;
for $i = 0$ to $m - 1$ do begin /* non-recursive
    $B = B + b_i \cdot 2^{n-i-1}$; computing */
end for;

Note that only the initial fixed angle addition and first m iterations are needed for generating all the direction sequence while the conventional angle computation algorithm needs $n - 3$ iterations.

When the angle is given as a binary number which is normalized to a power of two, the first two iterations need no computation and the multiplication by $2^n/2\pi s$ is needed for proper scaling to the level of the BSR. Note that the multiplication of $2^n/2\pi s$ is very simple, which is composed of only two stages of addition when $n = 9$. The bottleneck that limits the speed is the sign generation of the middle $m - 2$ iterations, which corresponds to only one iteration when $n = 9$.

3.2. Accumulated Angle Rotation

There are applications such as the phase tracking system which does not require the absolute angle, but only the accumulated angle which satisfies $A[k+1] = A[k] + \Omega[k]$ where $A[k]$ is the phase and $\Omega[k]$ is the frequency [2][3][4]. In this case, all the angles need not be converted from the real angle values to the BSRs. A method which accumulates angles in the form of the BSR is devised.

For simplifying hardware complexity, we want to compute the next accumulated angle from the present one in the form of the BSR as $B[k+1] = B[k] + C[k] + M[k]$ where B and C are the BSR of A and Ω, respectively, and $M[k]$ compensates for the error due to the direct addition of two BSR angles. The direct addition brings about problem when the carry is propagated to the upper position, because $a(i) = 2a(i+1)$ is not satisfied for all i. Carry propagation occurs by the addition of $2^{n-i-2} + 2^{n-i-2} = 2^{n-i-1}$. The left side of the equation actually means $\frac{2a(i+1)}{s} + \frac{2a(i+1)}{s} = \frac{2a(i)}{s} + \left[\frac{4a(i+1)}{s} - \frac{2a(i)}{s}\right]$ and the right side means $\frac{2a(i)}{s}$. Thus, when the carry is propagated to the i-th elementary angle position, the value in the parenthesis should be added for correction. When $i = 0$, the correction value is zero because $a(0) = 2a(1)$ and the correction values for $i \geq m$ is approximately zero because of the initial assumption of linearity. Hence, the total correction value is given by

$$M(u_1 u_2 \ldots u_{m-1}) = \sum_{i=1}^{m-1} u_i \cdot \left[\frac{4a(i+1)}{s} - \frac{2a(i)}{s}\right] \quad (13)$$

where u_i is the carry propagated to the i-th elementary angle position. The following algorithm enables the addition of the BSR angles.

Algorithm 2 *Addition of the BSR angles*

$Z = B + C$; $U = \mathrm{Car}_1^{m-1}[B + C]$;
while $U \neq 0$ do begin
  $Z = Z + M(U)$; $U = \mathrm{Car}_1^{m-1}[Z + M(U)]$;
end while;

where Car_1^{m-1} means the concatenation of the carries propagated to the i-th position for $1 \leq i \leq m-1$ and $M(U)$ is the correction value given in Eq. (13). Note that the correction can propagate the carries again, thus the correction should be performed repeatedly until there is no carry propagation. Actually, the number of repeated carry propagation is restricted to $m-1$.

4. IMPLEMENTATION EXAMPLE

We have developed a VLSI for the quadrature demodulation using the proposed CORDIC architecture. In phase tracking applications, the frequency of the quadrature oscillator is represented as $\Omega[k] = \Omega_0 + \Delta\Omega[k]$, where Ω_0 is the nominal frequency and $\Delta\Omega[k]$ is a relatively small frequency error [5]. Conventionally $\Omega_0 = \pi/2$ so that one cycle can be obtained at every four samples. When $\Delta\Omega[k] \geq 0$, the BSR of $\Omega[k]$, $C[k]$ can be written as

$$C[k] = C_0 + \Delta C[k] = 2^{n-2} + \Delta C[k] \quad (14)$$

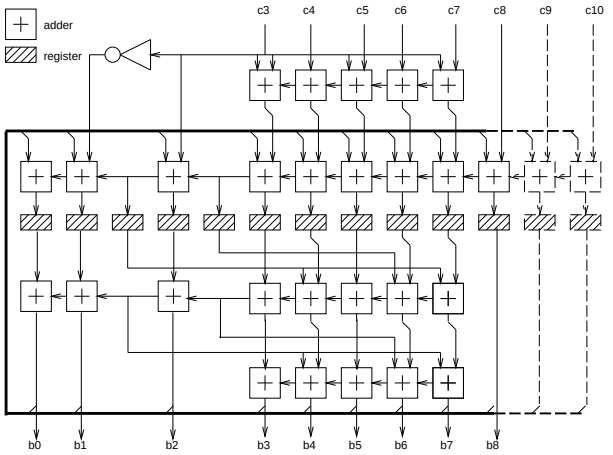

Figure 2. Circuit diagram for angle accumulator

where $\Delta C[k] < 2^{n-m}$ is assumed. When $\Delta\Omega[k] < 0$, the BSR of $\Omega[k]$ can be written as

$$C[k] = \left(\sum_{i=2}^{m-1} 2^{n-i-1} + \frac{2a(1)}{s} - \sum_{i=2}^{m-1} \frac{2a(i)}{s}\right) + \Delta C[k] \quad (15)$$

where the representation of C_0 is modified for the negative $\Delta C[k]$. The frequency control input $\Omega_0 + \Delta\Omega[k]$ can be implemented by a simple addition if $\Delta C[k]$ satisfies the following condition.

$$-\frac{2a(1)}{s} + \sum_{i=2}^{m-1} \frac{2a(i)}{s} \leq \Delta C[k] < 2^{n-m} \quad (16)$$

which is an allowable restriction for the phase tracking application.

Logic design is considered for the hardware implementation. The case of $n = 9$ is considered, which requires $m = 3$. The correction value in Eq. (13) for each u_i is as follows:

$$M(u_1) = 18.0 \approx 000010010$$
$$M(u_2) = 3.78 \approx 000000100$$

The nominal frequency C_0 for $\pi/2$ is

$$C_0 = \begin{cases} 010000000, & \Delta C[k] \geq 0 \\ 001101010, & \Delta C[k] < 0 \end{cases} \quad (17)$$

The allowable range of $\Delta C[k]$ is $-42 \leq \Delta C[k] < 64$, which is acceptable for phase tracking applications. Figure 2 shows the accumulator circuit for this example, where the dotted part can be appended for precise frequency control.

The block diagram of the total demodulator is shown in Fig. 3, where the developed angle accumulator circuit is employed for eliminating the angle computation block. The dotted part can be inserted for the phase tracking. The layout of the total demodulator is also shown in Fig. 4.

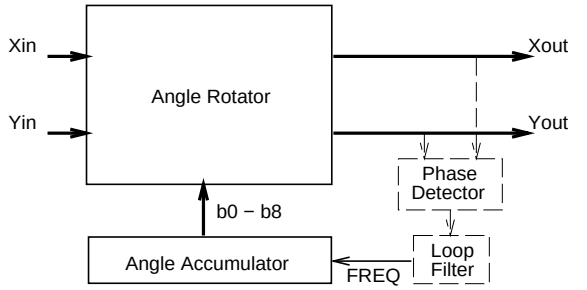

Figure 3. Block diagram of the quadrature demodulator chip

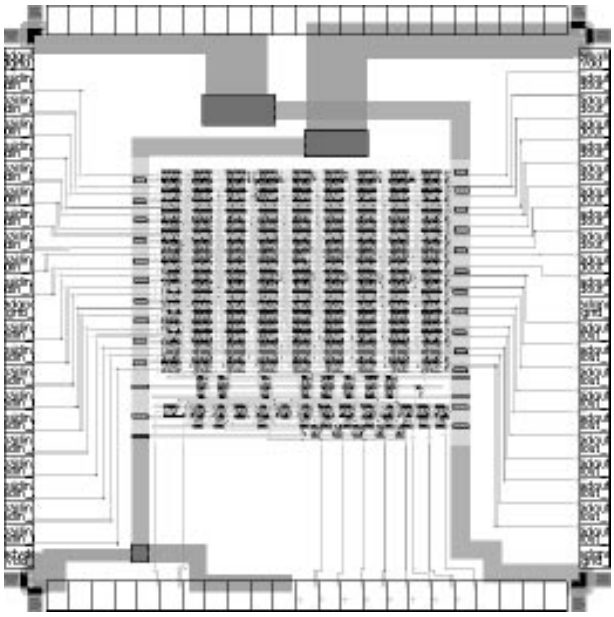

Figure 4. Layout of the quadrature demodulator chip

5. COMPARISON WITH OTHER METHODS

When the absolute angle rotation is necessary, the conventional algorithm requires $n-3$ iterative computations, considering that the first two and the last computations are actually unnecessary for the properly scaled angle representation [2]. On the other hand, the proposed BSR method requires approximately $n/3$ computations for the angle representation scaled by s. When an accumulated angle is used, the proposed phase accumulator circuit eliminates the angle computation circuit.

Madisetti's angle rotation algorithm [3] also utilizes the approximation of $\tan a(i) \approx a(i) = 2^{-k}$ for small $a(i)$, and exploits the 8th wave symmetry of sinusoidal waves. The direction sequence generation can be very fast, which results in a fast DDFS implementation. However, a third of the total stages of the angle rotation require multiplications with the tangent values. These multiplications can be implemented with multiplexers for the DDFS, but real multiplications are required for a general circular rotator. Thus, Madisetti's angle rotation algorithm is suited only for the generation of sine and cosine waves. On the other hand, the proposed BSR method only changes the angle computation for generating the direction sequence and the angle rotation still consists of only shifts and additions. Thus, the BSR method can be adopted for a rotator, which can be used for the digital quadrature demodulation.

6. CONCLUSION

A new direction sequence generation method for the circular CORDIC is developed by employing a new angle representation model. This method requires a third of the sequential computations when compared with the conventional angle computation algorithm. Thus, the hardware cost reduction and the speed-up can be expected as much. This method is more attractive when the applications require a higher precision and do not allow the pipelining technique because of the feedback control. A VLSI for the digital quadrature demodulation employing the proposed method is developed and the functionality is verified.

ACKNOWLEDGMENTS

This work was supported by the Ministry of Education in KOREA, Project ISRC 95-2-2018.

REFERENCES

[1] Yu Hen Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal Processing Magazine*, pp. 16–35, July. 1992.

[2] Gerard Gielis, Rudy van de Plassche, and Johan van Valburg, "A 540-MHz 10-b polar-to-cartesian converter," *IEEE J. Solid-State Circuits*, vol. 26, no. 11, pp. 1645–1650, Nov. 1991.

[3] Avanindra Madisetti, Alan Kwentus, and Jr. Alan N. Willson, "A sine/cosine direct digital frequency synthesizer using an angle rotation algorithm," in *ISSCC 1995 Digest of Technical Papers*, Feb. 1995, pp. 262–263.

[4] Loke Kun Tan and Henry Samueli, "A 200 MHz quadrature digital synthesizer/mixer in 0.8-μm CMOS," *IEEE J. Solid-State Circuits*, vol. 30, no. 3, pp. 193–200, Mar. 1995.

[5] Rajeev Jain, Henry Samueli, Paul T. Yang, Charles Chien, Gloria G. Chen, Linda K. Lau, Bong-Young Chung, and Etan G. Cohen, "Computer-aided design of a BPSK spread-spectrum chip set," *IEEE J. Solid-State Circuits*, vol. 27, no. 1, pp. 44–58, Jan. 1992.

[6] STANFORD TELECOM, *Digital, Fast Acquisition, Spread Spectrum Burst Processor STEL200A*, ASIC Custom Products Division, 1993.