

Low-Complexity Rotators for the FFT Using Base-3 Signed Stages

Petter Källström, Mario Garrido, and Oscar Gustafsson
Department of Electrical Engineering, Linköping University
SE-581 83 Linköping, Sweden
Email: {petterk, mariog, oscarg}@isy.liu.se

Abstract—Rotations by angles that are fractions of the unit circle find applications in e.g. fast Fourier transform (FFT) architectures. In this work we propose a new rotator that consists of a series of stages. Each stage calculates a micro-rotation by an angle corresponding to a power-of-three fractional parts. Using a continuous powers-of-three range, it is possible to carry out all rotations required. In addition, the proposed rotators are compared to previous approaches, based of shift-and-add algorithms, showing improvements in accuracy and number of adders.

I. INTRODUCTION

In fast Fourier transform (FFT) calculations [1], there is a need to compute rotations of complex numbers, that are carried out by complex rotators. Each complex rotation can be defined as a multiplication by a twiddle factor

$$W_N^\phi = e^{-j\frac{2\pi\phi}{N}}, \quad (1)$$

where N is the number of rotation angles in which the circumference is divided, and $\phi \in \{0, 1, \dots, N-1\}$ is the index of the rotation angle.

One approach to implement the rotator is to store the twiddle factors in a ROM, and use a complex multiplier to carry out the rotation [2]. The complex multiplier can be implemented with three or four real multipliers [3].

In order to avoid those multipliers, many multiplierless shift-and-add methods have been developed. An early such algorithm was the CORDIC [4]–[6]. It is based on breaking down the rotation angle into a set of predefined angles. Accordingly, the rotation is calculated by a series of micro-rotation by those angles. The angles are chosen so that the micro-rotations can be easily implemented in hardware. Each micro-rotation stage only consists of an adder, a subtractor and some multiplexers. All micro-rotator stages introduce a scaling, which means that the magnitude is increased. In CORDIC, the total gain for all stages is $R = \prod_i |1 + 2^{-i}j| \approx 1.647$ [5].

Other CORDIC-based methods that consist of a cascade of micro-rotations have also been proposed [7], [8]. For instance, the redundant CORDIC algorithm [7] introduces a redundant phase representation, which removes long carry chains, leading to lower latency. This approach has the drawback that it gives a non-uniform scaling, or gain, depending on the required angle to rotate. This was solved in [7] by using another set of adders. However, this requires to increase the number of adders with respect to the conventional CORDIC.

Recently, a method based on scaling the coefficients in order to reduce the rotation error was presented in [2]. This approach uses complex multipliers, so it is not comparable with shift-and-add algorithms, but the described method is used in the proposed method to find efficient coefficients.

When the rotator must only compute rotations by a reduced set of angles, the design can be optimized for those angles. This is the case for the proposed method. Thus, by optimizing the rotations only for the needed angles, the total number of stages can be kept much lower than in the CORDIC. The proposed approach also uses a “base 3”-inspired method, which lowers the number of stages even more. The base-3 structure uses, like the redundant CORDIC, the possibilities of a rotation by 0° and, hence, it rotates either $+\alpha$, 0 , or $-\alpha$. However, contrary to the redundant CORDIC, by using the coefficient scaling method [2], the scaling error can be considered in the selection of the coefficients. This assures a uniform scaling, i.e., the same gain for all angles.

Finally, in the proposed method, the number of stages depends only on number of points to rotate, N , and not on the precision. The precision is related to the complexity of the rotator, which can be arbitrarily chosen.

The proposed method has been compared with recent shift-and-add based methods. In [9], [10] complex rotators using trigonometric identities were proposed. They implemented only a few constants, and then derive the rest of them to those, by using, e.g., $\sin(\pi/8) = 2\cos(\pi/16)\sin(\pi/16)$. In [11], a reduced Booth-like multipliers is presented. It is based on the observation that when the set of coefficients is known, it is enough to design the accumulation tree for the maximum number of actual partial products.

II. ERROR IN HARDWARE ROTATIONS

A hardware rotator takes an input $x + jy$ and provides an output $X + jY$ being

$$X + jY = (C_\alpha + jS_\alpha) \cdot (x + jy), \quad (2)$$

where $C_\alpha + jS_\alpha$ is the coefficient for the rotation by the angle α . It approximates $R\cos\alpha + jR\sin\alpha$, which is the exact value of the rotation, scaled by the scaling factor R [2]. According to this, the normalized rotation error is

$$\epsilon_\alpha = \left| \frac{C_\alpha + jS_\alpha}{R} - (\cos\alpha + j\sin\alpha) \right|. \quad (3)$$

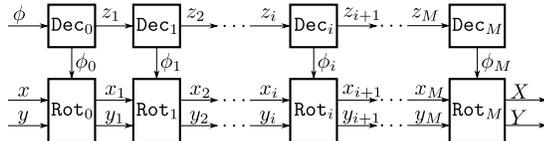


Fig. 1. Overview of a cascaded design using micro-rotators.

According to (1), the angles of the FFT are given by

$$\alpha = -\frac{2\pi}{N}\phi, \quad (4)$$

where α is the angle in radians and ϕ is the angle in steps. A step is here defined as the angle corresponding to $1/N$ of the circumference, or $\frac{2\pi}{N}$ radians.

By substituting equations (1), (2) and (4) in (3), the rotation error can be obtained as

$$\epsilon_\phi = \left| \frac{X + jY}{(x + jy) \cdot R} - W_N^\phi \right| \quad (5)$$

The error for a rotator is the maximum error for any angle the rotator can carry out, i.e.,

$$\epsilon = \max_{\phi} \epsilon_\phi. \quad (6)$$

For the CORDIC, the maximum error has an upper limit given by $\epsilon \leq \tan^{-1}(2^{-M}) \approx 2^{-M}$ after M micro-rotators [4].

The error can also be expressed in terms of effective word length, WL_E , using the relation

$$WL_E = -\log_2 \frac{\epsilon}{2\sqrt{2}} = -\log_2 \epsilon + \frac{3}{2}, \quad (7)$$

III. PROPOSED BASE-3 ROTATOR

The proposed approach carries out the rotation in a series of stages, as depicted in Fig. 1. Thus, the twiddle factors are calculated as

$$W_N^\phi = W_N^{\phi_0} W_N^{\phi_1} \dots W_N^{\phi_M}. \quad (8)$$

This approach is similar to the CORDIC algorithm, but uses a different set of rotations. Specifically, the rotation angle ϕ is broken down into a set of angles, ϕ_i , where

$$\phi = \sum_{i=0}^M \phi_i. \quad (9)$$

The first angle, $\phi_0 \in \{0, \frac{N}{4}, \frac{N}{2}, \frac{3N}{4}\}$, is used for trivial rotations by the angles $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$. This sets the remaining angle, z_1 , in the range $[-\frac{N}{8}, \frac{N}{8}]$, or $[-45^\circ, 45^\circ]$. The rest M angles are used for micro-rotations by

$$\phi_i = 3^{M-i} \delta_i \text{ steps}, \quad i = 1, 2, \dots, M \quad (10)$$

where $\delta_i \in \{-1, 0, 1\}$. This set of angles is based on a base-3 number system, where δ_i are considered as the digits of the numbers. Together they span all steps in the range $[-\frac{3^M-1}{2}, \frac{3^M-1}{2}]$. They correspond to the required twiddle factors, which makes this approach very suitable for the FFT.

TABLE I
EXAMPLE WHEN DECODING $\phi = 10$ WHEN $N = 32$, $M = 2$.

i	$z_i \in [\dots]$	$\phi_i \in \{\dots\}$	$x_{i+1} + jy_{i+1}$
0	$10 \in [0, 31]$	$8 \in \{0, 8, 16, 24\}$	$(x + jy)W_N^8$
1	$2 \in [-4, 4]$	$3 \in \{-3, 0, 3\}$	$(x + jy)W_N^{8+3}$
2	$-1 \in [-1, 1]$	$-1 \in \{-1, 0, 1\}$	$(x + jy)W_N^{11-1}$
out	$0 \in [0, 0]$	$\sum = 10$	$(x + jy)W_N^{10}$

In order to support all values $z_1 \in [-\frac{N}{8}, \frac{N}{8}]$, we get the requirement $\frac{3^M-1}{2} \geq \frac{N}{8}$. Therefore, the minimum number of micro-rotators is

$$M = \lceil \log_3(\frac{N}{4} + 1) \rceil. \quad (11)$$

A. Phase Decoding

The first stage of the phase decoding (Dec_0) determines the trivial rotations, whereas the rest of the phase decoders (Dec_i , $i \geq 1$), decode their incoming phases (z_i), that must be in the range

$$-\frac{3^{M-i+1} - 1}{2} \leq z_i \leq \frac{3^{M-i+1} - 1}{2}, \quad (12)$$

and selects which δ_i to pick. This selection is done so the requirements (12) also holds for z_{i+1} , according to

$$\delta_i = \begin{cases} +1, & z_i > (3^{M-i} - 1)/2 \\ -1, & z_i < -(3^{M-i} - 1)/2 \\ 0, & \text{otherwise.} \end{cases} \quad (13)$$

The remaining angle, z_{i+1} , is calculated by subtracting the rotation angle of the i -th stage from its input angle, i.e.,

$$z_{i+1} = z_i - \phi_i = z_i - 3^{M-i} \delta_i. \quad (14)$$

To implement this in hardware, first it is checked if $z_i \geq 0$, by using the most significant bit (MSB), and then a temporary value, $z'_i = (1 - 3^{M-i})/2 + |z_i|$, is calculated.

If $z'_i < 0$ (again using the MSB), $\delta_i = 0$ is selected. Otherwise $\delta_i = \pm 1$, depending on the sign of z_i . For this operation, one adder is required.

The operation $z_{i+1} = z_i - \phi_i$ requires another adder. Note that both adders operate on z_i , which is typically a small number, so they only need a few bits. The last stage will only need an OR gate, and no adder at all.

In general, the decoder path uses $2M - 2$ adders, with only a few bits per adder. Therefore, the complexity of the adders in the phase decoder is negligible compared to the decoders in conventional CORDIC processors. For instance, for W_{32} , Dec_0 needs a 2-bit adder and Dec_1 needs a 2-bit and a 3-bit adder, and Dec_2 does not need any adders.

For example, consider the case $N = 32$, giving $M = \lceil \log_3(\frac{N}{4} + 1) \rceil = 2$, where $x + jy$ should be rotated by ϕ steps. Figure 2 depicts the possible rotations, and highlights the rotations used when $\phi = 10$. Here, the trivial rotator is depicted as the innermost arrows (one arrow per possible value for $\phi_0 \in \{0, 8, 16, 24\}$). The next "layer" of arrows depicts the choices for $\phi_1 \in \{-3, 0, 3\}$, and the outermost "layer"

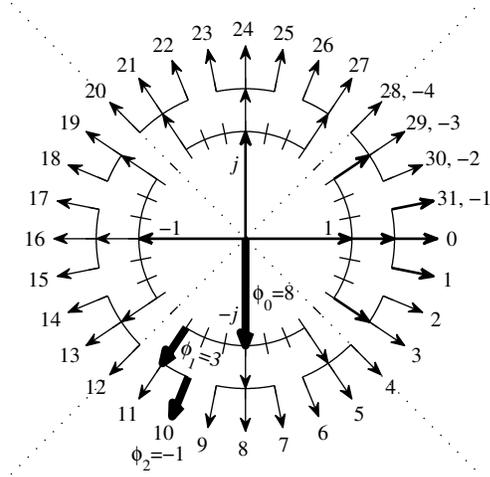


Fig. 2. Illustration of the example from Table I.

depicts the choices of $\phi_2 \in \{-1, 0, 1\}$. The operations used in the different stages when $\phi = 10$ are also listed in Table I, where $z_0 = \phi$, $x_{M+1} = X$ and $y_{M+1} = Y$.

As can be seen, in the case when $N = 32$ the proposed method is well utilized, since most “branches” in Fig. 2 have all three sub branches (this because $3^M \approx N$). In other cases, the utilization is not so good. For instance, if $N = 128$, then $M = 4$ and $\frac{3^M - 1}{2} = 40 \gg \frac{N}{8} = 16$, causing a significant overlap between the different quadrants.

B. Micro Rotators

The micro-rotators, Rot_i , $i \geq 1$, multiply $x_i + jy_i$ by a complex value, P_i , using shift-and-add methods according to

$$x_{i+1} + jy_{i+1} = (x_i + jy_i) \cdot P_i. \quad (15)$$

The complex value P_i depends on the given rotation angle, and includes a scaling, $R_i \approx |P_i|$. It is defined as

$$P_i = \begin{cases} C_i - j\delta_i S_i, & \delta_i = \pm 1 \\ K_i, & \delta_i = 0, \end{cases} \quad (16)$$

where C_i , S_i and K_i are positive integer constants. Therefore, the micro-rotations are calculated as

$$x_{i+1} = \begin{cases} C_i x_i + \delta_i S_i y_i, & \delta_i = \pm 1 \\ K_i x_i, & \delta_i = 0 \end{cases} \quad (17)$$

$$y_{i+1} = \begin{cases} C_i y_i - \delta_i S_i x_i, & \delta_i = \pm 1 \\ K_i y_i, & \delta_i = 0 \end{cases} \quad (18)$$

The values C_i , S_i and K_i are obtained by the coefficient scaling method [2], optimized for the sets of angles $\{0, \frac{2\pi}{N} 3^{M-i}\}$ radians, and modified to also calculate the number of adders for the kernels (where each kernel is a suggested set $\{K_i, C_i + jS_i\}$). The calculated number of adders for a kernel is the greatest number used to implement any of the two angles of the kernel, and the other angle will then reuse the same adders by using multiplexers.

The cases when $\delta_i = \pm 1$ are implemented using multiple constant multiplication (MCM) [12], where the same MCM

TABLE II
ALTERNATIVES FOR THE MICRO-ROTATION STAGES, WHEN $N = 32$.

Rot ₁	Add	K_1	$C_1 + jS_1$	R	ϵ^1	WL_E
$W_N^{\{0, \pm 3\}}$	4	231	$192 + 128j$	230.76	1.0×10^{-3}	11.49
	6	1847	$1536 + 1026j$	1847.2	1.5×10^{-4}	14.25
	8	7423	$6172 + 4124j$	7423.0	1.9×10^{-7}	23.85
Rot ₂	Add	K_2	$C_2 + jS_2$	R	ϵ^1	WL_E
$W_N^{\{0, \pm 1\}}$	2	4	$4 + 1j$	4.1231	4.9×10^{-2}	5.86
	4	325	$320 + 64j$	325.71	2.2×10^{-3}	10.33
	6	1313	$1288 + 256j$	1313.2	1.5×10^{-4}	14.21
	8	42155	$41345 + 8224j$	42155	7.4×10^{-7}	21.86

¹ Error of the micro-rotations.

TABLE III
ALTERNATIVES FOR ENTIRE ROTATOR, WHEN $N = 32$.

Adders			Error ¹	
Rot ₁	Rot ₂	Total ²	ϵ	WL_E
4	2	$\sqrt{6}$	5.2×10^{-02}	5.77
6	2	8	5.1×10^{-02}	5.79
4	4	$\sqrt{8}$	2.8×10^{-03}	9.99
8	2	10	5.1×10^{-02}	5.80
6	4	10	2.4×10^{-03}	10.20
4	6	$\sqrt{10}$	1.3×10^{-03}	11.11
8	4	12	2.3×10^{-03}	10.26
4	8	12	1.2×10^{-03}	11.24
6	6	$\sqrt{12}$	3.2×10^{-04}	13.13
8	6	14	1.7×10^{-04}	14.05
6	8	$\sqrt{14}$	1.5×10^{-04}	14.19
8	8	$\sqrt{16}$	9.0×10^{-07}	21.58

¹ Error of the entire rotator.

² $\sqrt{\quad}$ indicates a final candidate.

structure is applied to both x_i and y_i . In addition to the adders needed by the MCM, the extra adder used by the additions in (17) is used. This gives $2 + 2 \cdot \text{MCM}(C_i, S_i)$ adders, where $\text{MCM}(C_i, S_i)$ is the number of adders needed for a multiplication by C_i and S_i simultaneously. In the case when $\delta_i = 0$, only a single constant multiplication (SCM) [13] is used, and applied on x_i and y_i , giving $2 \cdot \text{SCM}(K_i)$ adders, where $\text{SCM}(K_i)$ is the number of adders needed for the multiplication by K_i .

For each kernel, the modified coefficient scaling method will output a list of the possible kernels, with their errors and required number of adders respectively.

C. Adders vs Errors

When the list of precision vs number of adders for each micro-rotator in the design is available, simply test all combinations, test the resulting precision for each combination, and finally keep the best result for each total number of adders. This brute force method may seem computationally demanding, but if the optimizer stores four different kernels for each micro-rotator (using 2, 4, 6 and 8 adders respectively), and there is $M = 4$ micro-rotators, then there are $4^M = 256$ combinations to test, which is done quickly by a computer.

As an example, consider the generation of the $N = 32$ case. The optimization procedure gives results shown in Table II for each of the $M = 2$ micro-rotators. The procedure did not find any 2-adder alternative for implementing Rot_1 ($W_{32}^{\{0, \pm 3\}}$).

The entire rotator is obtained by combining the alternatives for Rot_1 and Rot_2 . This leads to the $3 \cdot 4 = 12$ cases for the entire rotator listed in Table III. The best combination for

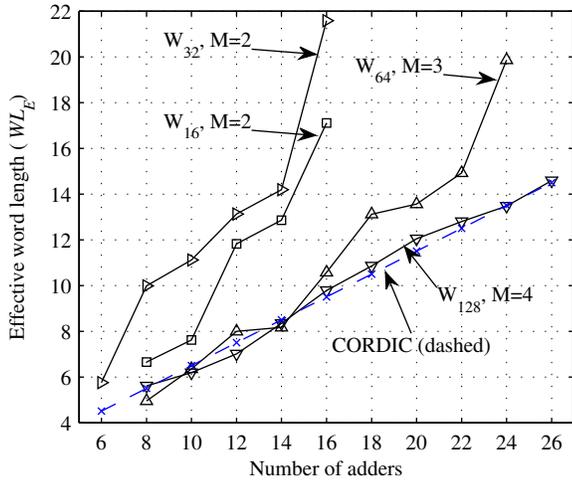


Fig. 3. Effective word length vs number of adders as a function of N .

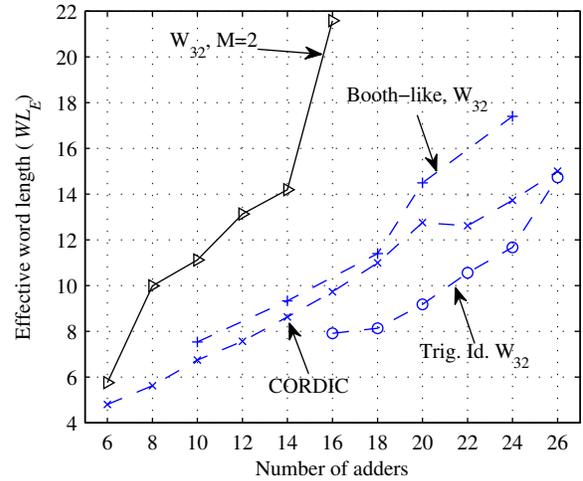


Fig. 4. Comparison of different approaches for W_{32} rotators.

each total number of adders is marked with (\checkmark). This leads to a trade off between accuracy and number of adders.

IV. COMPARISON

The WL_E of the all rotators has been calculated according to (5)–(7), for the best sets of δ . Since the adders in the phase decoder in the proposed method only need a few bits, and the CORDIC phase decoder can be implemented in different ways, only the adders in the data path are considered.

Figure 3 compares some of the proposed designs with CORDIC, showing that the $N = 16$ and $N = 32$ cases are significantly better than the CORDIC. This is due to the use of ad-hoc rotations. The W_{64} case is better than the CORDIC for high precision ($WL_E > 10$). For $N = 128$ the results are similar to the CORDIC, with the advantage of the use of a shorter number of stages, and a much simpler phase decoder. For larger rotators the proposed method is not so suitable, due to the larger number of angles that must be considered.

It can be observed that the W_{32} rotators are more efficient than the W_{16} ones. Although W_{32} and W_{16} need two micro-rotator stages, so they should have roughly the same efficiency, the angles used in the W_{16} case, $\pi/8$ and $3\pi/8$ radians, are much harder to implement than the angles used in the W_{32} case, $\pi/16$ and $3\pi/16$.

Note that the W_{16} rotator can be implemented using the coefficients from the W_{32} rotator, if non-integer ϕ_1 and ϕ_2 can be accepted ($\{0, \pm 0.5\}$ and $\{0, \pm 1.5\}$ respectively). Ongoing work focuses on relaxing the requirements on ϕ_i , to include, e.g., fractional values.

Figure 4 compares different rotators for W_{32} . The proposed architecture is shown to be more efficient than the Booth-like design [11], the CORDIC and the trigonometric identity solution [10]. Only the errors in the W_{32} angles are considered in this case.

V. CONCLUSION

In this paper, we have presented a method to perform complex rotation, based on cascaded micro-rotators. The main

idea is the inspiration from a base-3 number system, with the digits -1 , 0 and 1 , that allows us to perform rotations in both directions. The focus in the rotations is to use few and accurate stages, that rotate integer number of steps of size $2\pi/N$ radians.

REFERENCES

- [1] M. Garrido, J. Grajal, M. A. Sánchez, and O. Gustafsson, "Pipelined radix- 2^k feedforward FFT architectures," *IEEE Trans. VLSI Syst.*, Accepted for publication.
- [2] M. Garrido, O. Gustafsson, and J. Grajal, "Accurate rotations based on coefficient scaling," *IEEE Trans. Circuits Syst. II*, vol. 58, no. 10, pp. 662–666, Oct. 2011.
- [3] A. Wenzler and E. Luder, "New structures for complex multipliers and their noise analysis," in *ISCAS*, vol. 2, 1995, pp. 1432–1435.
- [4] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electronic Computing*, vol. EC-8, pp. 330–334, Sep. 1959.
- [5] R. Andraka, "A survey of CORDIC algorithms for FPGA based computers," in *Proc. ACM/SIGDA Int. Symp. FPGAs*, Feb. 1998, pp. 191–200.
- [6] M. Garrido and J. Grajal, "Efficient memoryless CORDIC for FFT computation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, vol. 2, Apr. 2007, pp. 113–116.
- [7] N. Takagi, T. Asada, and S. Yajima, "Redundant CORDIC methods with a constant scale factor for sine and cosine computation," *IEEE Trans. Comput.*, vol. 10, no. 9, pp. 989–995, Sep. 1991.
- [8] S.-F. Hsiao, C.-H. Lee, Y.-C. Cheng, and A. Lee, "Designs of angle-rotation in digital frequency synthesizer/mixer using multi-stage architectures," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Nov. 2011, pp. 2181–2185.
- [9] J.-Y. Oh and M.-S. Lim, "New radix-2 to the 4th power pipeline FFT processor," *IEICE Trans. Electron.*, vol. E88-C, no. 8, pp. 1740–1746, Aug. 2005.
- [10] F. Qureshi and O. Gustafsson, "Low-complexity constant multiplication based on trigonometric identities with applications to FFTs," *IEICE Trans. Fundamentals*, vol. E94-A, no. 11, pp. 324–326, Nov. 2011.
- [11] Y.-E. Kim, K.-J. Cho, and J.-G. Chung, "Low power small area modified Booth multiplier design for predetermined coefficients," *IEICE Trans. Fundamentals*, vol. E90-A, no. 3, pp. 694–697, Mar. 2007.
- [12] O. Gustafsson, "A difference based adder graph heuristic for multiple constant multiplication problems," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2007, pp. 1097–1100.
- [13] O. Gustafsson, A. G. Dempster, K. Johansson, M. D. Macleod, and L. Wanhammar, "Simplified design of constant coefficient multipliers," *Circuits Syst. Signal Process.*, vol. 25, no. 4, pp. 225–251, Apr. 2006.