

# EFFICIENT CORDIC DESIGNS FOR MULTI-MODE OFDM FFT

*Cheng-Ying Yu, Sau-Gee Chen and Jen-Chuan Chih*

Department of Electronics Engineering and Institute of Electronics  
National Chiao Tung University  
Hsinchu, Taiwan, ROC

## ABSTRACT

In this paper, we propose a new CORDIC algorithm and architectures which can generate close-to-optimum rotation sequences easily with small lookup table sizes. This new design is particularly suitable for the applications of adjustable-length FFT. In all, the required number of shift-and-add operations for micro-rotations and scale-factor compensations is only  $n/2$ , where  $n$  is the output precision. For design verification, we synthesized both serial and pipelined architectures, by using Synopsys Design Compiler based on UMC 0.18  $\mu m$ , 1P6M CMOS technology. The synthesized 16-bit pipelined FFT PE runs at 222MHz, with a total gate count of 89263 and a low-power consumption of 26.75 mW. It meets the FFT speed requirements of most OFDM-based communication systems, including DAB, DVB, 802.16 and VDSL. Compared with a conventional multiplier-based FFT PE and the existing CORDIC-based FFT PE's, the proposed designs has better performances in terms of area, speed and power consumption.

## 1. INTRODUCTION

DFT is one of most important computations in engineering applications. It is also a key component in OFDM communication systems. Cooley-Tukey FFT algorithms [1] are efficient realization techniques of DFT operations. They are composed of a sequence of unit butterfly operations. A butterfly operation involves addition operations and multiplication operation by twiddle factors  $W_N^k$ ,  $k = 0, 1, 2, \dots, N/2 - 1$ , where  $N$  is the FFT length. In the literature, a butterfly unit is generally realized by complex multipliers and adders, together with the stored required twiddle factors in memory. This approach has the advantages of design simplicity and high-speed operations. However, the required complex multipliers and the storage for twiddle factors incur high area overhead. In this work, in order to reduce these two overheads, we will propose a new CORDIC algorithm and its architectures specifically tailored for FFT computations. The proposed design will consume smaller areas than con-

ventional multiplier-based FFT units and those FFT processing elements based on general CORDIC algorithms.

CORDIC algorithm [2] is a well-known and efficient algorithm for the computations of vector rotations, vector angles and magnitudes. Since CORDIC algorithm only needs a sequence of micro rotations based on simple shift-and-add operations, it is efficient in hardware realization.

Since a twiddle factor multiplication is equivalent to rotation of a 2-D vector by the twiddle factor's phase, CORDIC algorithms are very suitable for twiddle factor multiplications. An additional significant benefit of applying CORDIC to FFT computations is that there is no need to store twiddle factors which are required in conventional multiplier-based FFT designs. In the literature, most of the CORDIC algorithms are directly applied to FFT computations without optimizations. As a result, they are not efficient in terms of operations counts and area complexities.

There are only a few optimized CORDIC-based FFT designs (that reduce redundant iterations as much as possible) [3-5]. However, some of the optimized designs require considerable memory overheads [4,5] to generate optimized rotation sequences for low iteration counts. Besides, those optimized rotation sequences introduce the serious problem of variable scale factors. There involves considerable overheads in generating on-line scale factors for those optimized rotation sequences. In previous works, we proposed an efficient on-line generation and compensation scheme [3] for variable scale factor. In this work, we will alleviate the problem by applying the similar idea to the new designs.

Besides, when apply CORDIC techniques to FFT computations, we can take advantages of special properties of FFT operations, for more efficient design results. For example, for a radix-2 FFT algorithm, only  $N$  twiddle factors are requested for computations (in the order of  $W_N^0, W_N^1, W_N^2, \dots$  for the first stage). In our proposed design, we will utilize this property for efficient generations of rotation sequences as will be discussed later. The result is that the proposed designs can generate close-to-optimum rotation sequences, at the cost of little hardware overhead and very small table size.

## 2. THE NEW CORDIC ALGORITHM FOR FFT

First, consider the basic CORDIC algorithm. Given a vector  $(x, y) = (x_0, y_0)$  to be rotated by an  $(n+1)$ -bit angle  $\theta \in [0, \pi/2]$ , and  $\theta = \alpha_0 \cdot \alpha_1 \alpha_2 \cdots \alpha_n$ , then the rotated output vector  $(x', y')$

$$x' = x \cos \theta + y \sin \theta; \quad y' = y \cos \theta - x \sin \theta \quad (1)$$

can be executed by the following CORDIC algorithm:

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & \delta_i 2^{-i} \\ -\delta_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \left( \prod_{m=1}^i \begin{bmatrix} 1 & \delta_m 2^{-m} \\ -\delta_m 2^{-m} & 1 \end{bmatrix} \right) \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad i=0,1,\dots,n \quad (2)$$

$$x' = K \cdot x_{n+1}, \quad y' = K \cdot y_{n+1} \quad (3)$$

where

$$K = \prod_{i=0}^n \cos \theta_i = \prod_{i=0}^n 1/\sqrt{1+2^{-2i}}, \quad \theta_i = \tan^{-1} 2^{-i} \quad (4)$$

$$\delta_i = \text{sign}(z_i), \quad z_{i+1} \equiv z_i - \delta_i \theta_i \equiv \theta - \sum_{m=0}^i \delta_m \theta_m \quad (5)$$

Notice that the whole computations are mapped into a sequence of shift-and-add operations. To reduce the iteration counts and speed up the whole operations, in our previous works, we proposed high-radix CORDIC algorithms such as the radix-4 [6], radix-16 [7] algorithms. Although those designs are efficient, as usual they require hardware overhead for obtaining residue angles. Next, we will introduce a new CORDIC design for FFT computations, by taking advantage of FFT properties. The required rotation sequences can be easily obtained by looking up a small table and performing some few simple operations in one shot, and there is no need to update the residue angles in each iteration (as detailed below).

### 2.1 The Basic Design Idea

Our design is based on an idea of efficient angle decomposition. Since we focus on FFT computation, there are only  $N$  twiddle factors (that correspond to only  $N$  rotation angles) involved in the CORDIC operations, we can decompose the rotation angle of a twiddle factor into a coarse angle component and a fine angle component. The fine angle component is small enough to satisfy the well-known linear mapping property  $\tan^{-1}(2^{-i}) \approx 2^{-i}$ , if  $i \geq n/3$ , while the coarse component is relatively large and does not satisfy the condition. For example, for any input angle  $\theta \geq 2^{-(n/3+2)}$ , then it is said to have a coarse angle component. As a result, for the fine angle component, its rotation sequence can be readily obtained by inspection, while its corresponding scale factor can be easily obtained (as will be detailed later). Further, for the coarse angle component, we can store its optimized rotation sequences and scale factor sequences in a lookup table. Then, whenever we want to do a twiddle factor multiplication (i.e., a CORDIC rotation), we directly decompose the input angle into these two components, and at the same time

obtain all the required optimized rotation sequences and scale factors. Doing so, we will have a very low shift-and-add operation count, and we don't need to compute residue angles iteratively. Hence, the normally required  $\tan^{-1}(2^{-i}) \approx 2^{-i}$  table can be saved.

### 2.2 Generations of Twiddle Factor Angles

The twiddle factors (and the corresponding rotation angles) of a radix-2<sup>n</sup> FFT algorithm come in a particular order in accordance with the order of butterfly operations. For example, for radix-2 FFT algorithm, the twiddle factors can be requested in the order of  $W_{N/2^{k-1}}^l$ ,  $l=0,1,\dots,N/2^k-1$ ,  $k=1,2,\dots,\log_2 N$ , where  $l$  is the index for the twiddle factors (and also the butterflies) and  $k$  is FFT stage number. As such, the corresponding twiddle factor angles  $(-2\pi/(N/2^{k-1}))$ ,  $l=0,1,\dots,N/2^k-1$  come in successively and incrementally, which are  $l$  multiples of the "base rotation angle"  $-2\pi/(N/2^{k-1})$ . Generations of those ordered twiddle factor angles therefore can be done by successively accumulating the base rotation angles. The base angles can be either a "base fine angle" or a "base coarse angle". Hence, we need to store all the optimized rotation sequences and their corresponding scale factors in a lookup table, for those base rotations angles. This requires a small memory size of  $\log_2 N$  entries.

#### 2.2.1 Generation of fine rotation sequences

As mentioned above, a radix-2 FFT performs twiddle factor multiplications by  $W_{N/2^{k-1}}^l$  successively, in the order of,  $l=0,1,\dots,N/2^k-1$ ,  $k=1,2,\dots,\log_2 N$ . As such, the base angle is  $\theta = -2\pi/(N/2^{k-1})$ , which will be accumulated to generate all the other twiddle factor angles. If  $\theta = -2\pi/N$  is small enough so that it and its initially accumulated angles are all fine angles. For those fine rotation angles, the corresponding fine rotation sequences are exactly the same as their binary angle representations from the accumulator output. In the proposed design, by using the contents of FFT stage counter and the butterfly counter as the address lines, we can lookup the base fine rotation sequence corresponding to the base fine rotation angle from a table. Then the base fine rotation sequence is sent to an accumulator for the generation of other rotation sequences. Next, those sequences are converted to CSD (canonical signed digit) formats which guarantee minimum numbers of micro-rotations.

#### 2.2.2 Generation of coarse rotation sequences

Coarse angles can be generated owing to two different conditions. The first one is that accumulations of a base fine rotation sequence may end up with output angles with coarse angle components. The second condition is that a base rotation angle is already contains a coarse angle com-

ponent. As a result, all its subsequent accumulated values also contain coarse angle components. In those cases, we have to decompose those twiddle factor angles into coarse and fine angle components. Then the corresponding fine rotation sequences (in binary formats) can be easily obtained. On the other hand, the optimized coarse rotation sequences (in CSD formats for their MSB parts and in binary formats for their LSB parts) and their corresponding scale-factor sequences can be obtained from a lookup table as discussed before.

In fact, differentiation and decomposition of a twiddle factor angle into a coarse angle component and a fine angle component can be easily done from the contents of the stage and butterfly counters.

### 2.2.3 Combined rotation sequences

Since there are overlaps in the micro-rotation angles between the coarse and fine rotation sequences, we can combine them altogether for further reduction of the numbers of shift-and-add operations. Specifically, one can combine the LSB portions of coarse rotation sequences with fine rotation sequences by simply adding them up, because both satisfy the property of  $\tan^{-1}(2^{-i}) \approx 2^{-i}$ . Then the combined rotation sequences are converted to CSD formats which correspond to the minimum numbers of shift-and-add operations. Finally, the CSD signals are sent to the rotator unit of a CORDIC-based FFT PE.

### 2.2.4 Generations and compensations of scale factors

In the proposed design, we skip many redundant micro-rotations. Therefore, the scale factors will not be constant. Here, by taking into account of the pre-stored scale factor sequences (for coarse rotation angles), we modify our previous work [3] and propose a low-complexity generation and compensation scheme for variable scale factors as follows. Based on the following approximation of a basic scale factor:

$$\begin{aligned} \cos \theta_i &= \frac{1}{\sqrt{1+2^{-2i}}} = (1-2^{-(2i+1)}) + O(2^{-(4i+1)}) \\ &\approx (1-2^{-(2i+1)}) \prod_j (1+2^{-C_j}) \end{aligned} \quad (6)$$

where  $O(2^{-(4i+1)})$  stands for the error term.

we propose the following modified on-line scale factor compensation scheme:

1. When  $i \leq n/3 + 2$ , the scale factor sequences can be retrieved from a lookup table (defined before) for later compensation operations.
2. When  $n/2 - 1 \geq i > n/3 + 2$ , then  $\cos \theta_i \approx (1-2^{-(2i+1)})$ .

As such:

$$\begin{aligned} x_{i+1} &= [x_i + \delta_i 2^{-i} y_i] (1-2^{-(2i+1)}) \\ y_{i+1} &= [y_i - \delta_i 2^{-i} x_i] (1-2^{-(2i+1)}) \end{aligned} \quad (7)$$

3. When  $i > n/2 - 1$ , then  $\cos \theta_i \approx 1$  and no compensation is required.

### 2.2.5 Overall description of the new CORDIC algorithm

The new CORDIC algorithm can be summarized in steps as follows.

**Step 0:** Obtain the table index based on the contents of FFT butterfly counter and stage counter, then decide the numbers ( $L_C$ ,  $L_S$  and  $L_F$ ) of iterations for coarse rotation, scale factor compensation, and fine rotation, respectively.

**Step 1:** Obtain the coarse rotation sequence and scale factor compensation sequence from the sequence table using the table index from Step 0, and generate the fine rotation sequences by accumulating the base fine rotation sequence.

**Step 2:** Add the LSB portion of coarse rotation sequence to the fine rotation sequence, and convert the combined sequence to CSD format.

**Step 3:** For data rotation, we first perform the coarse rotation according to the MSB portion of coarse rotation sequence for  $L_C$  iterations. Next, we perform the scale factor compensation according to the scale factor compensation sequence for  $L_S$  iterations. Finally, we perform the fine rotation according to the combined rotation sequence for  $L_F$  iterations.

From 16-bit simulation, the new CORDIC algorithm needs 5.03 shift-and-add operations in average for data rotation which is very close to the optimum 4.14 iterations (due to computer full-search).

## 3. REALIZATION OF THE NEW CORDIC-BASED FFT PROCESSING ELEMENT

We apply the new CORDIC algorithm to the design of multi-standard, multi-mode FFT computations for several mainstream OFDM communication systems, including DAB, DVB, 802.16, ADSL and VDSL systems. Specifically, we design an adjustable-length FFT processing element (PE) which can process FFT lengths up to 8192. Fig. 1 shows the block diagram of the design 16-bit pipelined CORDIC-based FFT PE. The design meets the speed specification requirements of all those OFDM systems.

We realize the pipeline architecture with two different unit rotator cells, i.e., the Rotator\_4\_2 cell and the Rotator\_2\_1 cell. Rotator\_4\_2 cell can handle 4 rotation digits and process up to two micro-rotations at a time, while Rotator\_2\_1 cell can only process 2 rotation digits and process up to one micro-rotation at a time. One can design a unit rotator cell adjusted for desired speed and area specifications. We also realize the new CORDIC algorithm based on a serial single-rotator architecture, for applications with lower data rates such as DAB and DVB.

## 4. SIMULATIONS RESULTS AND COMPARISONS

Table 1 shows the synthesized areas and power performances of the proposed designs and a general multiplier-based processing element. They are synthesized with

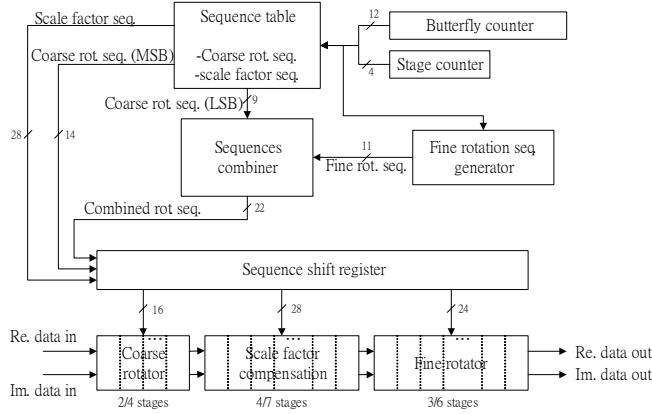


Fig. 1. Block diagram of the proposed pipelined CORDIC architecture

TABLE 1. Comparison of area and power performances.

| @125MHz    | Proposed pipelined PE (Rotator_4_2) | Proposed pipelined PE (Rotator_2_1) | Proposed single-rotator PE | Multiplier-based PE |
|------------|-------------------------------------|-------------------------------------|----------------------------|---------------------|
| Gate count | 89263                               | 104011                              | 28129                      | 92890               |
| Power      | 26.75mW                             | 34.24mW                             | 4.45mW                     | 30.60mW             |

TABLE 2. Comparison of speed performances.

|           | Proposed pipelined architecture (Rotator_4_2) | Proposed pipelined architecture (Rotator_2_1) | Proposed single-rotator architecture | Multiplier-based PE |
|-----------|-----------------------------------------------|-----------------------------------------------|--------------------------------------|---------------------|
| Op. freq. | 200MHz                                        | 222MHz                                        | 182MHz                               | 125MHz              |

TABLE 3. Table sizes comparison for twiddle factors.

|                         | Proposed  | Multiplier-based FFT PE |              |
|-------------------------|-----------|-------------------------|--------------|
|                         |           | $[0, \pi]$              | $[0, \pi/2]$ |
| Coeff. width            | 16 digits | 12 bits                 |              |
| Size (No. of bit cells) | 672       | 49152                   | 24576        |

Synopsys Design Compiler using UMC 0.18  $\mu\text{m}$  1P6M CMOS technology. We only extract the processing element here for comparison.

Table 2 shows the maximum synthesized clock rates of the proposed architectures and multiplier-based design. Table 3 shows the comparison of required table size for twiddle factor storage in conventional multiplier-base FFT architecture. The sequence table of our design is only about 1% of the size of twiddle factor ROM table. As shown, the proposed FFT PE's based on the pipelined Rotator\_4\_2 structure has better performance than the conventional multiplier-based PE. Table 4 compares the new design with some CORDIC-based FFT designs. Although the simulation is done assuming 16-bit case, we also conduct simulations with other word lengths. Simulations show that in average  $n/2$  iterations are achieved with the new CORDIC algorithms.

TABLE 4. Comparison of CORDIC-based FFT designs

|                |                      | Table size (no. of bit cells) | No. of shift-& add operation for data rotations | Total no. of shift-& add operations |
|----------------|----------------------|-------------------------------|-------------------------------------------------|-------------------------------------|
| Proposed       |                      | 672                           | 5.03                                            | 8.75                                |
| Kuo et al. [5] | $\pi/2$ pre-rotation | 28675                         | 4                                               | 8                                   |
|                | $\pi/4$ pre-rotation | 14336                         | 4                                               | 8                                   |
| Hu [4]         |                      | 32768                         | 4.96                                            | 9.63                                |
| Chen [3]       |                      | 768                           | N/A                                             | 10.6                                |

## 5. CONCLUSION

The CORDIC algorithms and architectures proposed in this work are specifically designed for FFT operations. They combine some FFT properties effectively and achieve close-to-optimum iteration numbers of shift-and-add operations, with small lookup table and hardware complexity. The new designs are advantageous over the existing designs in terms of both speed and area. The designed pipelined multi-mode FFT PE can meet the speed specifications of most OFDM communication systems, including VDSL, 802.16, DAB and DVB.

## 6. REFERENCES

- [1] J. W. Cooley and J. W. Tukey, "An algorithm for machine computation of complex fourier series," *Math. Computation*, Vol. 19, pp. 297-301, Apr. 1965.
- [2] J.E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electronic Comput.*, Vol. EC-8, pp. 330-334, 1959.
- [3] K.L. Chen, "FFT Processor Design for OFDM Systems," NCTU, Master Thesis, Jun. 2004.
- [4] Y.H. Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal Processing Magazine*, Vol. 9, Issue 3, July 1992.
- [5] Jen-Chih Kuo, Ching-Hua Wen, and An-Yeu Wu, "Implementation of a programmable 64~2048-point FFT/IFFT processor for OFDM-based communication systems," *Proc. IEEE International Symposium Circuits and Systems ISCAS*, Vol. 2, May 2003.
- [6] C.C. Li and S.G. Chen, "A radix-4 redundant CORDIC algorithm with fast on-line variable scale factor compensation," *Proc. IEEE International Conference on Acoustic, Speech and Signal Processing*, pp. 639-642, 1997.
- [7] J.C. Chih and S.G. Chen, "Fast CORDIC algorithm based on a new recoding scheme for rotation angles and variable scale factors," *Journal of VLSI signal processing*, Vol. 8, pp.56-61, 2003.