On the Fixed-Point Accuracy Analysis and Optimization of FFT Units with CORDIC Multipliers

Omid Sarbishei and Katarzyna Radecka
Department of Electrical and Computer Engineering, McGill University, Montreal, Quebec, Canada H3A 2A7
Email: {omid.sarbishei@mail.mcgill.ca, katarzyna.radecka@mcgill.ca}

Abstract— **Fixed-point Fast Fourier Transform (FFT) units are widely used in digital communication systems. The twiddle multipliers required for realizing large FFTs are typically implemented with the Coordinate Rotation Digital Computer (CORDIC) algorithm to restrict memory requirements. Recent approaches aiming to optimize the bit-widths of FFT units while satisfying a given maximum bound on Mean-Square-Error (MSE) mostly focus on the architectures with integer multipliers. They ignore the quantization error of coefficients, disabling them to analyze the exact error defined as the difference between the fixed-point circuit and the reference floating-point model. This paper presents an efficient analysis of MSE as well as an optimization algorithm for CORDIC-based FFT units, which is applicable to other Linear-Time-Invariant (LTI) circuits as well.**

Keywords – Fast Fourier Transform, Fixed-Point Number Format, CORDIC Complex Multiplier, Mean-Square-Error, Signal-to-Quantization-Noise-Ratio.

I. INTRODUCTION

Fast Fourier Transform (FFT) is widely used in many Digital Signal Processing (DSP) applications, where a fixed-point number representation on ASIC/FPGAs is a common procedure justified by the lower hardware complexity of fixed-point designs compared to floating-point ones [14]. To further minimize hardware costs, extensive efforts have been made to improve the VLSI implementation of FFT units.

Several approaches have been proposed aiming to optimize the FFT units at the architecture level [1,2][4-10]. Some of this methods explore the trade-offs between different radix FFT units and focus on choosing the suitable radix-*n* FFT architecture for specific applications using Cooley-Tuckey's algorithm [11-13], while some other work addresses the advantages of making use of Coordinate Rotation Digital Computer (CORDIC) complex multipliers [22, 24, 25] instead of integer ones [2]. Particularly, employing integer multipliers in realizing FFT units require a Look-Up-Table (LUT) to store the cosine and sinus values of constant angles. The size of such a LUT is proportional to the size of FFT. Hence, for implementing large FFT units like an 8K one [1], a large Read-Only-Memory (ROM) is needed for integer multipliers. However, this is not the case for a CORDIC multiplier, where the size of the required LUT is proportional to the number of CORDIC iterations.

Another reason for the interest in new ways of FFT realization comes from the accuracy analysis, which is used to determine the values of Integer and Fractional Bit-widths (IB and FB) of fixed-point variables and constant coefficients to avoid overflow and satisfy a given bound on Mean-Square-Error (MSE), and Signal-to-Quantization-Noise-Ratio (SQNR), calculated at the outputs of the FFT unit. Further, if the FFT incorporates the CORDIC complex multipliers, the accuracy analysis must include the setting of CORDIC parameters such as the bit-widths of variables and coefficients as well as number of iterations.

Several approaches have been proposed to analyze the MSE in Linear Time Invariant (LTI) systems and CORDIC units [15,16,23,24,29]. Most of them suffer from major inefficiencies. Particularly, they analyze the variance of quantization error for LTI circuits, e.g., FFT units, but ignore the error originating from quantizing constant coefficients.

This paper presents an analysis of MSE/SQNR for FFT units with CORDIC/integer multipliers, which is applicable to other LTI circuits. Our analytical optimization algorithm efficiently sets the bit-widths of fixed-point variables and constant coefficients as well as the number of iterations required for pipeline CORDIC multipliers for a given FFT architecture (radix-*n* *M*-point FFT). Hence, a given maximum/minimum bound on MSE/SQNR is satisfied on the outputs of the FFT unit.

The rest of the paper is organized as follows. Section II presents the related works on FFT architectures and MSE/SQNR analysis. Section III addresses the proposed analysis of MSE/SQNR for a conventional CORDIC multiplier, which takes into account the effect of all the error sources including the quantization error of constant rotation angles as well as the constant coefficient *K* in the CORDIC algorithm. Section IV extends the analysis to a radix-8 8K FFT unit with CORDIC multipliers, which is applicable to an arbitrary radix *M*-point FFT in general. Section V illustrates the proposed analytical optimization algorithm, while Section VI addresses the experimental results. Finally conclusions are drawn in Section VII.

II. RELATED WORK

Analysis of SQNR/MSE for LTI systems has been studied in [15,16]. Particularly, the approach in [23] focuses on the fixed-point accuracy of FFT units realized with integer multipliers. However, the results in the above references do not take into consideration the quantization error of constant coefficients leading to an unrealistic SQNR

1063-6889/11 $26.00 © 2011 IEEE
DOI 10.1109/ARITH.2011.17

and MSE analysis, when error is defined as the difference between the fixed-point implementation and the reference floating-point model. Evaluating such an error is a crucial issue for verification purposes in scientific hardware computations [26]. Furthermore, the work in [19] has formally proved that if for the suitable low resolution variable or coefficient, its FB is increased by 1 bit only, it is possible to widely improve the hardware cost by reducing the bit-width of several other variables, which are reached through the next stages of logic. Hence, not having flexibility in controlling the bit-width of coefficients, as in [15,16,23], puts restrictions on the efficient design optimization.

The analysis of MSE/SQNR for CORDIC units has been studied in [24] and [29]. In order to avoid the complex correlation between the quantization noise of intermediate variables and the value of input vectors, both approaches [24] and [29] ignore the quantization error of constant coefficients including the constant rotation angles as well as the constant coefficient *K* in the CORDIC algorithm. Although the MSE/SQNR analysis becomes straightforward, it is not safe, as it *underestimates* the exact error, defined as the difference between the fixed-point implementation and the reference floating-point model. Furthermore, the schemes in [24], [29] provide no flexibility in controlling bit-widths of coefficients, which may result in inefficient design optimizations. The approach in [27], which makes use of the error model in [28], focuses on the analysis of maximum mismatch for CORDIC multipliers required for FFT units. However, the SQNR analysis, still suffers from the same problems discussed for other conventional methods.

The analysis of MSE/SQNR presented in this paper negligibly overestimates/underestimates the exact value of MSE/SQNR, which is robust and safe in contrast to the methods in [15,16,23,24], which underestimate/overestimate the exact MSE/SQNR. Furthermore, we present an analytical optimization technique, which provides flexibility in setting all the error sources independently.

III. Analysis of MSE/SQNR for CORDIC Twiddle Multipliers

A CORDIC multiplier performs a series of rotations to estimate the result of:

$$Y = (A + jB) \times (cos\varphi + j \times sin\varphi) \tag{1}$$

where A, B are input variables and φ is a constant angle in the range of $-\frac{\pi}{2} < \varphi < \frac{\pi}{2}$. If φ belongs to the interval $(\frac{\pi}{2} < \varphi < \frac{3\pi}{2})$, a simple block must be implemented to convert φ to $\pi - \varphi$, and perform a final negation of the results of the multiplier.

The operation of the rotation-mode CORDIC twiddle multiplication algorithm can be described as:

$$x_{i+1} = x_i - y_i d_i 2^{-i},\ y_{i+1} = y_i + x_i d_i 2^{-i}$$
$$z_{i+1} = z_i - d_i \tan^{-1}(2^{-i}) \tag{2}$$

where z_i is the remaining angle of rotation and x_i, y_i are rotating vectors at iteration #*i*. The parameter d_i is:

$$d_i = \begin{cases} 1 & if\ z_i \geq 0 \\ -1 & otherwise \end{cases}, (i = 0, \dots, N-1),$$

$x_0 = A \times K(N), y_0 = B \times K(N), z_0 = \varphi,$
$K(N) = \prod_{i=0}^{N-1} \cos(\tan^{-1}(2^{-i}))$

For the above initialization of x_0, y_0 and z_0, if N goes to infinity, the values of x_N and y_N converge to:

$$x_\infty = Acos\varphi - Bsin\varphi$$
$$y_\infty = Asin\varphi + Bcos\varphi \tag{3}$$

with $K(\infty) = 0.60725293500888\dots$

The values of x_∞ and y_∞ in Eqn. (3) are equal to the real and imaginary parts of Y in Eqn. (1) respectively.

Fixed-point implementation of the CORDIC multiplier given by Eqn. (2) must address a few quantization error sources: quantization error of *A, B,* and the initial values of $x_0 = A \times K(N)$ and $y_0 = B \times K(N)$, the finite number of iterations *N,* and finally, the round-to-nearest quantization error generated by performing the multiplications by 2^{-i} for x_i and y_i. Note that in the fixed-point representation, the residue angles $\tan^{-1}(2^{-i})$, $i = 0, \dots, N-1$, are the N-bit pre-computed values stored in an LUT. This is affordable, as $\tan^{-1}(2^{-i}) \approx 2^{-i}$ for high values of i, and as a result, for N iterations, N bits are sufficient to represent the constant *arctangent* values for all the N iterations. The fixed-point CORDIC algorithm is then:

$$\hat{x}_{i+1} = \hat{x}_i - \hat{y}_i d_i 2^{-i} + e_{\hat{x}_i},$$
$$\hat{y}_{i+1} = \hat{y}_i + \hat{x}_i d_i 2^{-i} + e_{\hat{y}_i}$$
$$\hat{z}_{i+1} = \hat{z}_i - d_i(\tan^{-1}(2^{-i}) + e_{\hat{z}_i}) \tag{4}$$

where $e_{\hat{x}_i}$ and $e_{\hat{y}_i}$ are the round-to-nearest errors generated by quantizing the results after shifting operations of 2^{-i}, and $e_{\hat{z}_i}$ is the quantization noise of $\tan^{-1}(2^{-i})$ (it is a constant value). Furthermore we have:

$$d_i = \begin{cases} 1 & if\ \hat{z}_i \geq 0 \\ -1 & otherwise \end{cases}, (i = 0, \dots, N-1), \hat{z}_0 = \hat{\varphi}$$
$$\hat{x}_0 = (A + e_A) \times \hat{K}(N)$$
$$\hat{y}_0 = (B + e_B) \times \hat{K}(N) \tag{5}$$

where $e_{A/B}$ is the quantization noise of A/B. $\hat{K}(N)$ is the quantized version of $K(N)$, which is constant if N is given and the FB of $K(N)$ is known. Furthermore, $\hat{\varphi}$ is the quantized version of the given angle φ.

Note that in Eqn. (4) at $i = 0$ there is no shifting and truncation error for $\hat{x}$ and $\hat{y}$. However, as $\hat{x}_0$ and $\hat{y}_0$ are realized by multipliers, Eqn. (5), then the truncation of the multipliers' outputs generates the quantization errors $e_{\hat{x}_0}$ and $e_{\hat{y}_0}$, present in Eqn. (4). After N iterations we obtain:

$$\hat{x}_N = \hat{x}_0 \times \frac{1}{K(N)} \times \cos(\hat{z}_N) - \hat{y}_0 \times \frac{1}{K(N)} \times \sin(\hat{z}_N) + e_{X_N}$$

$$\hat{y}_N = \hat{x}_0 \times \frac{1}{K(N)} \times \sin(\hat{z}_N) + \hat{y}_0 \times \frac{1}{K(N)} \times \sin(\hat{z}_N) + e_{Y_N} \quad (6)$$

where $\hat{z}_N$ is obtained by Eqn. (4) and e_{X_N}/e_{Y_N} is the error generated by the shifting and truncation operations performed in all the previous iterations ($i = 0, \dots, N-1$). Hence, using Eqn. (4) and setting $\hat{x}_0 = \hat{y}_0 = 0$ we obtain:

$$\hat{x}_N = e_{X_N} = \left(\sum_{j=0}^{N-1} u_j \times e_{\hat{x}_j}\right) + \left(\sum_{j=0}^{N-1} v_j \times e_{\hat{y}_j}\right) \quad (7)$$

$$\hat{y}_N = e_{Y_N} = \left(\sum_{j=0}^{N-1} w_j \times e_{\hat{x}_j}\right) + \left(\sum_{j=0}^{N-1} s_j \times e_{\hat{y}_j}\right) \quad (8)$$

where u_j, v_j, w_j and s_j ($j = 0, \dots, N-1$) are all positive real numbers. Now using Eqn. (5) we can re-write Eqn. (6) as:

$$\hat{x}_N = c_1 \times (A + e_A) + c_2 \times (B + e_B) + e_{X_N} \quad (9)$$

$$\hat{y}_N = f_1 \times (A + e_A) + f_2 \times (B + e_B) + e_{Y_N} \quad (10)$$

where e_{X_N} and e_{Y_N} are given by Eqns. (7-8), c_{1-2} and f_{1-2} are real numbers that can be calculated using φ, N and the FB of the constant coefficient representing $\widehat{K}(N)$.

Consider the output Y, Eqn. (2), which is also equal to $Y = x_\infty + j \times y_\infty$ with x_∞ and y_∞ described by Eqn. (3), and its fixed-point representation $Y_{fixed} = \hat{x}_N + j \times \hat{y}_N$ with $\hat{x}_N$ and $\hat{y}_N$, Eqns. (9) and (10). For the given φ, N and the FB of the constant coefficient representing $\widehat{K}(N)$ the error (mismatch) function between the reference Y and its fixed-point representation Y_{fixed} can be determined as:

$$e_r = real\left(Y - Y_{fixed}\right) = x_\infty - \hat{x}_N \overset{(3),(9)}{\Longrightarrow}$$

$$= (cos\varphi - c_1)(A + e_A) + (sin\varphi - c_2)(B + e_B) + e_{X_N}$$

$$e_i = imaginary\left(Y - Y_{fixed}\right) = y_\infty - \hat{y}_N \overset{(3),(10)}{\Longrightarrow}$$

$$= (sin\varphi - f_1)(A + e_A) + (cos\varphi - f_2)(B + e_B) + e_{Y_N} \quad (11)$$

Equation (11) is a linear function of the variables $A, B, e_A, e_B, e_{\hat{x}_j}$ and $e_{\hat{y}_j}$ ($j = 0, \dots, N-1$). The input values A, B as well as the quantization errors $e_A, e_B, e_{\hat{x}_j}, e_{\hat{y}_j}$ have a uniform probability distribution over their given intervals, which can be computed as follows, if round-to-nearest quantization is used for scaling the variables:

$$-2^{-FB_A-1} \le e_A \le 2^{-FB_A-1}, -2^{-FB_A-1} \le e_B \le 2^{-FB_A-1}$$
$$-2^{-FB_x-1} \le e_{\hat{x}_j} \le 2^{-FB_x-1}$$
$$-2^{-FB_y-1} \le e_{\hat{y}_j} \le 2^{-FB_y-1} \quad (12)$$

where $FB_{A/B}$ is the FB of A/B, and $FB_{x/y}$ is the FB of $\hat{x}_j/\hat{y}_j$ ($j = 0, \dots, N-1$). Note that for simplicity we consider the quantization type of $e_{\hat{x}_j}, e_{\hat{y}_j}$ to be round-to-nearest and not truncation, since we have $E(e_{\hat{x}_j}) = 0$ for round-to-nearest quantization, when $e_{\hat{x}_j}$ has a uniform distribution.

The quantization error generated by scaling the intermediate variables in a fixed-point arithmetic circuit generally does not have a uniform distribution over its interval presented by Eqn. (12) [20]. However, for LTI systems, it has been shown that assuming a uniform probability distribution for round-to-nearest and truncation quantization noise is mostly acceptable, [18], [20], [21].

The SQNR of a twiddle CORDIC multiplier can now be computed as:

$$SQNR = \frac{E(|Y|^2)}{E({e_r}^2)+E({e_i}^2)} = \frac{E(A^2)+E(B^2)}{E({e_r}^2)+E({e_i}^2)} \quad (13)$$

where $E(X)$ is the expected value of X, the reference Y is described by Eqn. (1), and $e_{r/i}$ is given by Eqn. (11). As there is a complicated correlation among A, B, e_{X_N} and e_{Y_N} in Eqn. (11), the computation of $E({e_{r/i}}^2)$ in Eqn. (13) is not a straightforward task. However, by making use of the *overestimation* of the error function, it becomes possible to easily analyze an underestimation (pessimist analysis) of SQNR.

Corollary 1: *The error function: $e_{r/i}$ in Eqn. (11) can be overestimated with $\hat{e}_{r/i}$, which by itself is a function of the statistically independent variables: $e_A, e_B, e_{X_N}, e_{Y_N}$. The overestimation is then:*

$$\hat{e}_r = (cos\varphi - c_1)(\max\{|A|\} + e_A) + (sin\varphi - c_2)(\max\{|B|\} + e_B) + e_{X_N}$$
$$\hat{e}_i = (sin\varphi - f_1)(\max\{|A|\} + e_A) + (cos\varphi - f_2)(\max\{|B|\} + e_B) + e_{Y_N} \quad (14)$$

■

The overestimated error given by Eqn. (14) includes the term: $\max\{|A/B|\}$, which is independent from A/B. Based on Corollary 1 and the fact that $E(e_A) = E(e_B) = E(e_{\hat{x}_j}) = E(e_{\hat{y}_j}) = 0 \quad (j = 0, \dots, N-1)$ for round-to-nearest quantization with a uniform distribution, Eqn. (12), we can determine an overestimated value of $E\left({e_{r/i}}^2\right)$ as:

$$E({e_r}^2) < E\left({\hat{e}_r}^2\right) = (cos\varphi - c_1)^2 \times ((\max\{|A|\})^2 + E(|e_A|^2)) + (sin\varphi - c_2)^2 \times ((\max\{|B|\})^2 + E(|e_B|^2)) + E(|e_{X_N}|^2)$$

$$E({e_i}^2) < E\left({\hat{e}_i}^2\right) = (sin\varphi - f_1)^2 \times ((\max\{|A|\})^2 + E(|e_A|^2)) + (cos\varphi - f_2)^2 \times ((\max\{|B|\})^2 + E(|e_B|^2)) + E(|e_{Y_N}|^2) \quad (15)$$

Functions $E(|e_{A/B}|^2)$, $E\left(\left|e_{X_N}\right|^2\right)$ and $E(|e_{Y_N}|^2)$ can be computed as follows based on Eqns. (12), (7) and (8) respectively and the uniform distribution of $e_{\hat{x}_j}, e_{\hat{y}_j}, e_A$:

$$E(|e_A|^2) = \frac{2^{-2FB_A}}{12}, E(|e_B|^2) = \frac{2^{-2FB_B}}{12},$$

$$E\left(\left|e_{X_N}\right|^2\right) = \frac{2^{-2FB_x}}{12}\left(\sum_{j=0}^{N-1} u_j\right)^2 + \frac{2^{-2FB_y}}{12}\left(\sum_{j=0}^{N-1} v_j\right)^2,$$

$$E\left(\left|e_{Y_N}\right|^2\right) = \frac{2^{-2FB_x}}{12}\left(\sum_{j=0}^{N-1} w_j\right)^2 + \frac{2^{-2FB_y}}{12}\left(\sum_{j=0}^{N-1} s_j\right)^2 \quad (16)$$

Note, that FB_x, FB_y, FB_A and FB_B correspond to the quantization errors $e_{\hat{x}_j}$, $e_{\hat{y}_j}$, e_A and e_B respectively. By replacing $E\left(e_{r/i}{}^2\right)$ in Eqn. (13) with $E\left(\hat{e}_{r/i}{}^2\right)$ given by Eqn. (15) an overestimated/underestimated value of MSE/SQNR can be determined. Experimental results in Section VI show that the amount of overestimation/underestimation for MSE/SQNR based on Corollary 1 is small. It is also notable that Eqn. (16) is valid for discrete variables, if the number of truncation bits is high enough (typically higher than 8) [21]. The results in [19] and our experiments in Section VI, prove that the assumption mostly holds true in DSP applications.

IV. Analysis of MSE/SQNR for FFT Units with CORDIC Multipliers

In this section the proposed analysis of MSE/SQNR for FFT units with CORDIC complex multipliers is presented. Although, the discussion focuses on the radix-8 8K FFT specification, the study is applicable to any arbitrary-radix *M*-point FFT units. Furthermore, the analysis can be simplified to handle FFT units with conventional integer multipliers as well. Other LTI circuits like Finite-Impulse-Response (FIR) and Infinite-Impulse-Response (IIR) filters can similarly be handled by our analysis as well.

Using Cooley-Tukey's FFT algorithm the radix-8 8192-point FFT can be expressed as [1]:

$$X[k] = \sum_{n=0}^{8192} x[n] \times W_{8192}^{nk} =$$
$$= \sum_{n_0=0}^{1} W_2^{n_0 k_0} \left(W_{8192}^{n_0 k'} \sum_{n_4=0}^{7} W_8^{n_4 k_1} \left(W_{64}^{n_4 k_2} W_{512}^{n_4 k_3} W_{4096}^{n_4 k_4}\right.\right.$$
$$\left.\left.\sum_{n_3=0}^{7} W_8^{n_3 k_2} \left(W_{64}^{n_3 k_3} W_{512}^{n_3 k_4} \sum_{n_2=0}^{7} W_8^{n_2 k_3} \left(W_{64}^{n_2 k_4} \cdots\right.\right.\right.$$
$$\left.\left.\left.\cdots \sum_{n_1=0}^{7} W_8^{n_1 k_4} \left(x[n']\right)\right)\right)\right)\right) \quad (17)$$

where $x[n]$ is the time-domain input of the 8K FFT (n=0,…,8191), $X[k]$ is the frequency domain output pattern (k=0,…,8191), and the twiddle factor (constant coefficient) W_N^a is defined as: $W_N^a = e^{-\frac{j2\pi a}{N}}$. For time and frequency domain indexes of n' and k we have:

$$k = 4096k_0 + k' =$$
$$4096k_0 + (512k_1 + 64k_2 + 8k_3 + k_4)$$
$$n' = n_0 + 1024n_1 + 128n_2 + 16n_3 + 2n_4$$

Eqn. (17) is the reference representation of the FFT unit, which can be in floating-point format. Each of the summation loops, Eqn. (17), is implemented using pipelining. Note, that all twiddle factors $W_{64}^{n_2 k_4}$, $W_{64}^{n_3 k_3} W_{512}^{n_3 k_4}$, $W_{64}^{n_4 k_2} W_{512}^{n_4 k_3} W_{4096}^{n_4 k_4}$ and $W_{8192}^{n_0 k'}$ are the complex exponential functions of the type $W_N^a = e^{-\frac{j2\pi a}{N}}$. As multiplications by these factors involve many angles, the hardware realization in terms of conventional integer multiplications requires big ROMs. Hence, CORDIC twiddle multipliers are used to realize them. However, for other twiddle factors $W_8^{n_i k_j}$ $(i, j = 1, \ldots, 4)$ and $W_2^{n_0 k_0}$, the number of possible angles is much smaller, and, as a result, those multiplications are assumed to be implemented with conventional integer multipliers and LUTs.

A. Fixed-Point Representation

Based on the discussion in Section III, we can express the fixed-point representation of the 8K FFT in Eqn. (1):

$$X_{fixed}[k] = \sum_{n_0=0}^{1} W_2^{n_0 k_0} \left(e_{4,N} + \{(c_{4,1} + j \times c_{4,2}), (f_{4,1} + j \times f_{4,2})\} \otimes \sum_{n_4=0}^{7} \{e_{q_4} + \widehat{W}_8^{n_4 k_1} \left(e_{3,N} + \{(c_{3,1} + \right.\right.$$
$$j \times c_{3,2}), (f_{3,1} + j \times f_{3,2})\} \otimes \sum_{n_3=0}^{7} \{e_{q_3} + \widehat{W}_8^{n_3 k_2} \left(e_{2,N} + \{(c_{2,1} + j \times c_{2,2}), (f_{2,1} + j \times f_{2,2})\} \otimes \right.$$
$$\left.\left.\sum_{n_2=0}^{7} \{e_{q_2} + \widehat{W}_8^{n_2 k_3} \left(e_{1,N} + \{(c_{1,1} + j \times c_{1,2}), (f_{1,1} + j \times f_{1,2})\} \otimes \sum_{n_1=0}^{7} \{e_{q_1} + \widehat{W}_8^{n_1 k_4} \left(x[n'] + e_{in}\right)\}\right)\}\right)\}\right)\}\right)\right) \quad (18)$$

where the operator $\otimes$ is defined below:

$$\{A, B\} \otimes b = A_r b_r + A_i b_i + j(B_r b_r + B_i b_i)$$

Parameters $A_{r/i}$, $B_{r/i}$ and $b_{r/i}$ are the real and imaginary parts of A, B and b respectively. The values of $\widehat{W}_8^{n_i k_j}$ $(i, j = 1, \ldots, 4)$ are the quantized versions of $W_8^{n_i k_j}$ in Eqn. (17) w.r.t. the FB that is used to realize the coefficient $\pm\frac{\sqrt{2}}{2}$. The term e_{in} is the input quantization error. The quantization errors $e_{q_{1-4}}$ are generated after scaling the bit-width of variables that realize the results of multiplication by $\widehat{W}_8^{n_i k_j}$. Both e_{in} and $e_{q_{1-4}}$ also have uniform probability distributions over the following intervals:

$$-2^{-FB_{in}-1} \le e_{in} \le 2^{-FB_{in}-1}$$
$$-2^{-FB_{1-4}-1} \le e_{r/i\_q_{1-4}} \le 2^{-FB_{1-4}-1}$$

where FB_{in} is the FB of the input x, FB_j $(j = 1, \ldots, 4)$ is the FB of the real/imaginary part of the variables realizing the result of multiplication by $\widehat{W}_8^{n_j k_{5-j}}$. The element $e_{r/i\_q_{1-4}}$ is the real/imaginary part of the quantization error $e_{q_{1-4}}$. The values of $E(|e_{in}|^2)$ and $E\left(|e_{q_{1-4}}|^2\right)$ can be computed as:

$$E(|e_{in}|^2) = 2 \times 2^{-2FB_{in}}/12$$
$$E\left(|e_{q_{1-4}}|^2\right) = 2 \times 2^{-2FB_{1-4}}/12 \quad (19)$$

Note that the term "2 ×" in Eqn. (19) is due to the existence of real and imaginary parts for the quantization errors e_{in} and $e_{q_{1-4}}$.

The other parameter in Eqn. (18) is $e_{p,N}$ $(p = 1 - 4)$, and is equivalent to: $e_{X_N} + j e_{Y_N}$ with e_{X_N} and e_{Y_N} defined by Eqns. (7) and (8) for the p^{th} CORDIC twiddle multiplier. Finally, $c_{p,1-2}$ and $f_{p,1-2}$ are equivalent to the constants

c_{1-2} and f_{1-2} in Eqns. (9) and (10) for the p^{th} CORDIC twiddle multiplier.

Using Eqns. (17) and (18) the error function between the reference and fixed-point FFTs, *i.e.,* $X_e[k] = X[k] - X_{fixed}[k]$, can be computed as:

$$X_e[k] = \left(\sum_{i=0}^{8191} c_{k_i} \times x[i]\right) + \left(\sum_{i=1}^{4} g_{k_i} \times e_{q_i}\right) + \left(\sum_{i=1}^{4} h_{k_i} \times e_{i,N}\right) + t_k \times e_{in} \quad (k = 0, \dots, 8191) \quad (20)$$

where c_{k_i} $(i = 0, \dots, 8191)$, $g_{k_{1-4}}$, $h_{k_{1-4}}$ and t_k are real values that can be computed according to Eqns. (17) and (18) and the known values of the following data: the FB of $\widehat{W}_8^{n_i k_j}$ $(i, j = 1, \dots, 4)$, the maximum number of iterations N for CORDIC multipliers, the FB of the number representing the quantized constant coefficient $\widehat{K}(N)$ as well as the loop indexes n_{1-4} and k_{1-4} in Eqns. (17) and (18).

Using Eqns. (17) and (20) the SQNR of the FFT unit can be determined as:

$$SQNR = \frac{\frac{1}{M}\sum_{i=0}^{M-1} E(|X[i]|^2)}{MSE} = \frac{\frac{1}{M}\sum_{i=0}^{M-1} M \times E(|x[i]|^2)}{MSE} = \frac{M \times E(|x|^2)}{MSE}$$

where $M = 8192$ and the MSE is:

$$MSE = \frac{1}{M}\sum_{k=0}^{M-1} \mathrm{E}(|X_e[k]|^2).$$

As shown in Section III, computing the value of $\mathrm{E}(|X_e[k]|^2)$ is not a straightforward procedure due to the complicated correlation between the input samples $x[i]$ $(i = 0, \dots, 8191)$ and the intermediate quantization errors: $e_{j,N}$ and e_{q_j} $(j = 1 - 4)$ in Eqn. (20). However, similarly to Corollary 1, the error function $X_e[k]$ in Eqn. (20) can be overestimated as follows:

$$\hat{X}_e[k] = \left(\sum_{i=0}^{8191} c_{k_i} \times \max\{|x|\}\right) + \left(\sum_{i=1}^{4} g_{k_i} \times e_{q_i}\right) + \left(\sum_{i=1}^{4} h_{k_i} \times e_{i,N}\right) + t_k \times e_{in} \quad (k = 0, \dots, M-1) \quad (21)$$

Hence, we have:

$$\mathrm{E}(|X_e[k]|^2) < E\left(\left|\widehat{X_e}[k]\right|^2\right) = \left|\sum_{i=0}^{M-1} c_{k_i} \times \max\{|x|\}\right|^2 + \left(\sum_{i=1}^{4} \left|g_{k_i}\right|^2 \times E\left(|e_{q_i}|^2\right)\right) + \left(\sum_{i=1}^{4} \left|h_{k_i}\right|^2 \times E\left(|e_{i,N}|^2\right)\right) + |t_k|^2 \times E(|e_{in}|^2) \quad (22)$$

If the FB values of the fixed-point variables that contribute to $e_{i,N}$ $(i = 1 - 4)$ in Eqn. (22) are represented as FB_{5-8}, we can re-write Eqn. (22) based on Eqn. (19) as:

$$E\left(\left|\widehat{X_e}[k]\right|^2\right) = \left|\sum_{i=0}^{M-1} c_{k_i} \times \max\{|x|\}\right|^2 + a_1 \times 2^{-2FB_1} + \cdots + a_8 \times 2^{-2FB_8} + a_9 \times 2^{-2FB_{in}} \quad (23)$$

where a_{1-9} are constant positive real values.

Now, SQNR can be underestimated as follows:

$$SQNR > \frac{M^2 \times E(|x|^2)}{\sum_{k=0}^{M-1} \mathrm{E}(|\hat{X}_e[k]|^2)} \quad (24)$$

The reason SQNR is underestimated is that the denominator in Eqn. (24) is an overestimation of MSE, i.e., $\sum_{k=0}^{M-1} \mathrm{E}(|X_e[k]|^2)$, Eqn. (22). Note that as the nominator in Eqn. (24) is independent from the fixed-point realization, we can re-define the input constraints on SQNR (minimum allowed SQNR) as an upper bound on MSE, which is the denominator in Eqn. (24).

V. Optimization Algorithm

Based on the analysis of MSE/SQNR in Section IV, we present an optimization algorithm designed to set fractional bit-widths of fixed-point variables and constants as well as CORDIC parameters such that a given minimum SQNR is satisfied. The optimization algorithm can easily be modified to satisfy a given maximum bound on MSE as well. The following lemma expresses the key point of the proposed straightforward optimization algorithm.

Lemma 1: *Consider the terms in Eqn.* (*23*)*, which originate from the input and intermediate quantization errors:*

$$E\left(\left|X_{eq}[k]\right|^2\right) = a_1 \times 2^{-2FB_1} + \cdots + a_8 \times 2^{-2FB_8} + a_9 \times 2^{-2FB_{in}} = \sum_{j=1}^{9} B_j \quad (25)$$

where $B_j = a_j \times 2^{-2FB_j}$ *and* $FB_9 = FB_{in}$*. Furthermore, with the upper bound on MSE equal to* MSE_{max}*, assume that the FB values:* FB_1*, …,* FB_9 *are all set in a way to satisfy the following condition:*

$$\frac{MSE_{max}}{36} < B_j \leq \frac{MSE_{max}}{9} \quad (j = 1, \dots, 9) \quad (26)$$

where $B_j = a_j \times 2^{-2FB_j}$*. Using Eqn.* (*26*) *to select all the FB values, the MSE at the output satisfies the condition* $MSE \leq MSE_{max}$*. At the same time all the other possible solutions of* $\{\widehat{FB_1}, \dots, \widehat{FB_9}\}$ *that result in a lower bound on MSE satisfy the condition* $\sum_{j=1}^{9} FB_j < \sum_{j=1}^{9} \widehat{FB_j}$*, which means that setting the FB values according to Eqn.* (*26*) *minimizes the total sum of FB values (hardware cost).*

Proof: There exists only one specific value of FB_j that can satisfy the condition in Eqn. (26). Furthermore, with all the FBs, i.e., FB_{1-9} satisfying Eqn. (26), and based on Eqn. (25) we have:

$$MSE = E\left(\left|X_{eq}[k]\right|^2\right) \leq MSE_{max}.$$

In order to prove the last part of Lemma 1, i.e., minimization of $\sum_{j=1}^{9} FB_j$, assume that we have initially set the FB values according to Eqn. (26). Now we prove that if we reduce one of the fractional bit-widths, for example FB_1 by 1, then we have to increase at least two other FB values FB_{2-3}, by 1, to keep the new bound on MSE equal to or lower than its initial value, $\sum_{j=1}^{9} B_j$. Hence, if we prove the above statement, it is deduced that setting the FB values according to Eqn. (26) minimizes the total sum of FBs, i.e., $\sum_{j=1}^{9} FB_j$, which implies minimization of hardware costs.

According to Eqn. (26), if we reduce FB_1 by 1, Eqn. (25) is changed to $E\left(|X_{eq}[k]|^2\right) = 3 \times B_1 + \sum_{j=1}^{9} B_j$. In order to reduce the new bound on MSE, we increase another FB, say FB_2 by 1. This task updates the MSE to $E\left(|X_{eq}[k]|^2\right) = 3 \times B_1 + \left(\sum_{j=1}^{9} B_j\right) - \frac{3}{4} \times B_2$. Hence, The term $\sum_{j=1}^{9} B_j$ in the initial MSE now becomes $3 \times B_1 - \frac{3}{4} \times B_2$. The best-case scenario is realized when B_1 is at its minimum and B_2 at its maximum level according to Eqn. (26), i.e., $B_1 = \frac{MSE_{max}}{36} + \varepsilon$, $B_2 = \frac{MSE_{max}}{9}$. The parameter ε is a small positive number as $B_1 > \frac{MSE_{max}}{36}$, Eqn. (26). Under such conditions the additional error can be obtained as:

$$3 \times B_1 - \frac{3}{4} \times B_2 = \frac{MSE_{max}}{12} + 3\varepsilon - \frac{MSE_{max}}{12} = 3\varepsilon > 0.$$

As can be seen the new bound on MSE even with the best case scenario for B_1 and B_2 is still higher than the initial MSE of $\sum_{j=1}^{9} B_j$. Therefore, it is imperative to increase another FB, e.g., FB_3 by 1 to keep the new bound on MSE lower than or equal to $\sum_{j=1}^{9} B_j$. Hence, the initial solution of $\{FB_1, \ldots, FB_9\}$ obtained by Eqn. (26), minimizes the value of $\sum_{j=1}^{9} FB_j$. ■

Lemma 1 indicates that the most suitable way of choosing values of FBs for input and intermediate variables in terms of hardware cost is to set equal upper bounds of MSE errors at the output originating from different sources of quantization, Eqn. (25). Otherwise, a higher total sum of FB values has to be assigned to the fixed-point variables, which impacts the hardware cost.

Based on Lemma 1 we present the proposed optimization algorithm for FFT units. The pseudo-code of the algorithm is given in Fig. 1. The inputs of the algorithm are the FFT reference representation, Eqn. (17), the minimum allowed value of SQNR is $SQNR_{min}$, and the maximum absolute value of the input x, *is* x_{abs}. The outputs are the number of CORDIC iterations N, the FB values of variables and coefficients as well as the SQNR of the corresponding fixed-point FFT, *i.e.,* $SQNR_{fixed}$. The parameter FB_c is the FB of all constant coefficients including real/imaginary parts of $\widehat{W}_8^{n_i k_j}$ in Eqn. (18), the value of $\widehat{K}(N)$ in Eqn. (5), and the constant rotation angles within CORDIC iterations. The minimum starting point for this variable is set to $FB_c = 1$ in Step1. At Step2 a *while loop* is executed to find the minimum possible value of FB_c, while the other imprecision parameters N and FB_{1-9} are all kept at "infinity", such that the given minimum allowed SQNR is satisfied. Note that computation of SQNR is based on the discussion in Section IV. In Step4 an analogous search is performed to set the minimum possible value of N with the pre-computed FB_c and the rest of FBs, i.e., FB_{1-9}, all being set to infinity. Since N (the number of CORDIC iterations) contributes to a higher hardware cost compared to these performed to check whether the computed value of N is smaller than FB_c. If this is the case then the *while loop* in Step4 is terminated; otherwise, the *while loop* continues after increasing FB_c by 1 such that a lower value of N can be obtained. After establishing FB_c and

FFT_ Opt ($SQNR_{min}$, *spec*, x_{abs})
{ /* *Inputs: Spec: Equation (17), Minimum allowed SQNR:* $SQNR_{min}$
Maximum magnitude of Input x: $x_{abs} = (\max\{|x|\})^2$
Outputs: FBs, MSE of the design: MSE_{fixed} */
1. $FB_c = 1$; // *initial minimum point for the FB of all constant coefficients*
// *Setting the minimum possible value of* FB_c
2. while ($SQNR(spec, FB_c, N = \infty, FB_{1-9} = \infty) < SQNR_{min}$)
3. { FB_c++; }
4. while(1) // *Setting the two values of* FB_c *and N simultaneously*
5. { $N = 6$; // *initial minimum point for N: CORDIC iterations*
6. while ($SQNR(spec, FB_c, N, FB_{1-9} = \infty) < SQNR_{min}$)
7. { N++; }
8. if ($N < FB_c$) break;
9. else FB_c++;
}
10. while (1) // *Setting the values of* FB_c, N *and* FB_{1-9} *simultaneously*
11. { Set FB_{1-9} using Lemma 1 so that:
$\{SQNR_{fixed} = SQNR(spec, FB_c, N, FB_{1-9})\} < SQNR_{min}$
12. if ($FB_c > \max\{FB_{1-9}\}$) break;
13. else FB_c++;
}
14. Return N, $FB_{in} = FB_9$, FB_{1-8}, FB_c, $SQNR_{fixed}$;
}

Figure 1. Proposed algorithm for optimizing a radix-8 8K FFT unit with CORDIC multipliers while satisfying a given minimum SQNR

N, it becomes possible to compute all the constant positive real values a_{1-9} in Eqn. (25). Therefore, in Step11 the values of FB_{1-9} are all analytically set based on Eqn. (26) and Lemma 1 such that a minimum total sum of FB values is obtained. Since, FB_{1-9} refers to fixed-point variables rather than constant coefficients, it contributes to a higher hardware cost compared to FB_c. Hence, in Step12 the maximum value of FB among FB_{1-9}, which have been already set analytically using Eqn. (26), is compared with FB_c. If FB_c is higher, then the *while loop* in Step10 is terminated. Otherwise, FB_c is increased by 1 and the loop is re-executed to obtain lower values of FBs for intermediate variables using of Lemma 1 and Eqn. (26). Finally in Step14, the final FB values, N and $SQNR_{fixed}$ are all returned.

VI. Experimental Results

In this section the robustness of the proposed analysis of SQNR as well as the efficiency of the optimization algorithm described in Fig. 1, are evaluated compared to the previous works. The algorithm is implemented in MATLAB and run on the Intel 2.8GHz Pentium 4 with 2 GBs of memory.

The proposed analysis of SQNR provides an underestimation, as formulated in Corollary 1 compared to the exact results. The first experiment in this section demonstrates that typically the amount of underestimation is not very high. As computing the exact value of SQNR is not possible for large designs, we target the CORDIC multiplier $S = A \times (cos\varphi + jsin\varphi)$, where the input A is a real number and has a uniform probability distribution over the interval $(-1,1)$. In the reference model the bit-width of A is set to 24, while in the fixed-point implementation it is 10 bits, i.e., $FB_{in} = 9$. The constant coefficient $\widehat{K}(N)$ and all the other fixed-point variables in the fixed-point realization of the CORDIC algorithm are represented by $FB_{1-8} = 9$.

Furthermore, N is set to 9. The reference model of $\widehat{K}(N)$, i.e., $K(N)$, is given by the 32-bit floating-point number format in MATLAB. Four different constant angles, i.e., $\varphi = 10^o, 35^o, 65^o, 70^o$, have been considered here.

Computing the exact values of SQNR requires considering all 2^{24} possible cases of A in the reference model. Table I shows the comparison among the exact SQNR determined by simulations, the underestimated one obtained by our analysis and the approximated SQNR analysis for CORDIC units presented in [24]. Note, that the analysis presented in [24] provides a dangerous overestimation/underestimation of SQNR/MSE (more than 6 dB) by ignoring the quantization noise of constant coefficients as well as the quantization noise generated by scaling the intermediate variables within CORDIC iterations. Our analysis negligibly underestimates/overestimates the SQNR/MSE (less than 1 dB), which is safe.

In the second experiment we utilize the algorithm in Fig. 1, to optimize a radix-8 8K FFT unit, which is the most critical block in Digital-Video-Broadcasting Terrestrial (DVB-T) receivers [1]. The market for digital TVs and DVB-T receivers has gained an interest in Asia and Europe during the last few years [3]. The 8K FFT architecture for DVB-T receivers presented in [1] makes use of CORDIC multipliers. It has been designed with a uniform FB equals to 10 for all the fixed-point variables and constants as well as $N = 10$ for the CORDIC multipliers. The integer bit-widths of fixed-point variables have been obtained using a simple dynamic range analysis [17]. The SQNR of the FFT architecture in [1] is computed to be 48.4884 dB using the analysis presented in Section IV. Then, the algorithm in Fig. 1 is executed to re-design and optimize the FFT unit such that the minimum SQNR of 48.4884 dB is satisfied.

As in many applications designers do not have freedom to change the FFT's input bit-width [3], we have chosen the (constant) value of $FB_{in} = 10$ for the proposed FFT unit. However, the bit-widths of coefficients (FB_c) and intermediate variables as well as the number of CORDIC iterations (N) can be set without any constraints. Table II provides a comparison between the two FFT units.

The word-lengths of different pipeline stages of the FFT units, *i.e., IB+FB*, are addressed in the last row of Table I. Virtex 5 with the speed optimization grade of "-3" has been chosen as the target device in Xilinx ISE 11 synthesis tool. The proposed scaled FFT unit saves about 24% of total sum of logic gates in the datapath block (including small intermediate FFT blocks and CORDIC multipliers) and 25% of critical path delay. At the same time, it achieves a 1.47 dB higher SQNR by optimizing the bit-widths of intermediate variables and constant coefficients along with setting suitable number of CORDIC iterations. The row "#of SRAM bits" refers to the total number of bits stored in the intermediate RAMs corresponding to the intermediate pipeline stages. Note that for both cores, the address spaces of the intermediate RAMs are the same, however, the word length of the RAMs differs based on the output bit-width of the pipeline stages (addressed in the final row). A substantial number of SRAM cells can be reduced in the proposed architecture. Note that the synthesis report in Table II does not include the control unit of the two cores, as it is similar for both of cores, and its complexity is negligible compared to the datapath block.

TABLE I. COMPARISON OF OUR SQNR ANALYSIS WITH EXHAUSTIVE SIMULATIONS AND THE ANALYSIS IN [24] FOR $S = A \times (cos\varphi + jsin\varphi)$

Angle	Exact SQNR using simulation (dB)	Approximate SQNR using the analysis in [24] (dB)	Underestimated SQNR using our analysis (dB)	Run-Time (s)	
				Simulation	Analysis (ours and [24])
$\varphi = 10^o$	34.937	~41.5544	33.5439	~500	<1
$\varphi = 35^o$	34.396		33.685		
$\varphi = 65^o$	35.4769		34.5378		
$\varphi = 70^o$	35.4253		34.8308		

TABLE II. COMPARISON OF THE PROPOSED SCALED 8K FFT UNIT AND THE ARCHITECTURE IN [1] WITH UNIFORM FB VALUES

Parameter	FFT with uniform FBs	Proposed FFT
SQNR (dB)	48.4884	**49.9657**
Datapath Logic Gates	57742	**43964**
#of SRAM bits	476544	**371328**
Critical Path (ns)	9.86	**7.394**
$FB_{in}/FB_c/N$	10/10/10	10/11/10
Pipeline Stage 1-5 word-length (IB+FB)	14/17/20/23/24	10/12/15/18/19

VII. CONCLUSION AND FUTURE WORK

In this paper a robust analysis of MSE and SQNR is presented for FFT units with CORDIC multipliers. The analysis can also be utilized for the units with conventional integer multipliers. The approach provides a negligible overestimation of error to make it straightforward to analyze MSE/SQNR. Based on such consideration an efficient optimization algorithm is introduced to set the fractional bit-width of fixed-point variables and constant coefficients along with selecting the CORDIC parameters in the design. The integer bit-widths of fixed-point variables can also be determined using simple dynamic range analysis methods [17]. Experimental results illustrate the robustness of our MSE/SQNR analysis compared to the previous work.

Furthermore, the proposed optimization algorithm has been evaluated for the FFT architecture with CORDIC multipliers in [1] and the hardware cost improvements are explored. A promising future work avenue is to extend the proposed MSE/SQNR analysis and optimization algorithm to handle nonlinear polynomial-based arithmetic circuits as well.

REFERENCES

[1] R. M. Jiang, "An Area Efficient FFT Architecture for OFDM Digital Video Broadcasting" *IEEE Trans. on Consumer Electronics,* Vol. 53, No. 4, pp. 1322–1326, November 2007.

[2] C-C. Wang, J-M. Huang, H-C. Cheng, "A 2K/8K Mode Small-Area FFT Processor for OFDM Demodulation of DVB-T Receivers", *IEEE Trans. on Consumer Electronics,* Vol. 51, No.1, February 2005.

[3] European Broadcasting Union, "Digital Video Broadcasting (DVB): Framing Structure, Channel Coding and Modulation for Digital Terrestrial Television", Datasheet: ETSI EN 300 744, January 2001.

[4] P. Combelles, C. Del Toso, D. Hepper, D. Le Goff, J. J. Ma, P.Rovertson, F. Scalise, L. Soyer, and M. Zamboni, "A receiver architecture conforming to the OFDM based digital video broadcasting standard for terrestrial transmission (DVB-T)", *IEEE ICC98,* Vol. 2, p. 7, June 1998.

[5] Zhi-Xing Yang, Yu-Peng Hu, Chang-Yong Pan, and Lin Yang, "Design of a 3780-Point IFFT Processor for TDS-OFDM", *IEEE Trans. on Broadcasting,* Vol. 48, No. 1, pp. 57, March 2002.

[6] S. Anikhindi, G. Cradock, R. Makowitz, and C. Petzelt, "A Commercial DVB-T demodulator chipset," *1997 International Broadcasting Convention,* pp. 528-533, September 1997.

[7] E. Bidet. J. C. Castekain, and P. Senn, "A fast single-chip implementation of 8192 complex point FFT," *IEEE J. of Solid-State Circuits*, vol. 20, no. 3, pp. 300-305, March 1995.

[8] Y. Jung, H. Yoon, and J. Kim, "New efficient FFT algorithm and pipeline implementation results for OFDM/DMT applications," *IEEE Trans. on Consumer Electronics*, Vol. 49, No. 1, Feb. 2003.

[9] S. A. Fechtel, and A. Blaickner, "Efficient FFT and equalizer implementation for OFDM receivers,'" *IEEE Trans. on Consumer Electronics*, Vol. 45, No. 4, pp. 1104-1107, Nov. 1999.

[10] J.-H. Suk, D.-W. Kim, T.-W. Kwon, S.-K. Hyung, and J.-R. Choi, "A8192 complex point FFT/IFFT for COFDM modulation scheme in DVB-T system," *Inter. SOC (System-on-Chip) Conference*, Vol. 5, pp. 131-134, Dec. 2003.

[11] K. Maharatna, E. Grass, and U. Jagdhold, "A 64-point Fourier transform chip for high-speed wireless LAN application using OFDM", *IEEE Journal of Solid-State Circuits*, Vol. 39, Issue 3, p.484, March 2004.

[12] A. Perez-Pascual, T. Sansaloni, J. Valls, "FPGA-based radix-4 butterflies for HIPERLAN/2", *IEEE International Symposium on Circuits and Systems*, Vol.3, p.277, May 2002.

[13] M. Jiang, et al, "A Multiplierless Fast Fourier Transform Architecture", Electronics Letters, Vol. 43, No.3, 2007.

[14] C. Shi and R. Brodersen, "Automated fixed-point data-type optimization tool for signal processing and communication systems," Proc. Design Automation Conf. 2004, pp. 478–483.

[15] D. Menard, R. Rocher, O. Sentieys, "Analytical Fixed-Point Accuracy Evaluation in Linear Time-Invariant Systems", *IEEE TCAS I,* Vol 5, No 10, Nov. 2008, pp: 3197-3208.

[16] G. A. Constantinides, P.Y. K. Cheung, and W. Luk, "The Multiple Wordlength Paradigm", *IEEE Symposium for Custom Computing Machines,* 2001.

[17] O. Sarbishei, Y. Pang, K. Radecka, "Analysis of Range and Precision for Fixed-Point Linear Arithmetic Circuits with Feedbacks", *IEEE HLDVT10,* June 2010.

[18] G. A. Constantinides, P.Y. K. Cheung, and W. Luk, "Truncation Noise in Fixed-Point SFGs", *Electronics Letters,* Vol 35, No 23, Nov. 1999.

[19] O. Sarbishei and K. Radecka, "Analysis of Precision for Scaling the Intermediate Variables in Fixed-Point Arithmetic Circuits", *Accepted in IEEE ICCAD,* Nov. 2010.

[20] G. D. Kim, D. M. Chibisov, "Distribution of Rounding Error in Multiplication of two Numbers on a Fixed-Point Computer", Mathematical Notes 1 (2), 1967, pp: 150-155.

[21] A. B. Sripad, D. L. Snyder, "A Necessary and Sufficient Condition for Quantization Errors to be Uniform and White", *IEEE Trans. On Acoustics, Speech and Signal Processing,* Vol 25, No 5, 1977, pp: 442-448.

[22] C-S. Wu and A-Y. Wu, "Modified Vector Rotational CORDIC (MVR CORDIC) Algorithm and Architecture", *IEEE Tran. On Circuits and Systems-II (TCAS-II): Analog and Digital Signal Processing,* Vol. 48, No. 6, June 2001, pp: 548-561.

[23] W. H. Chang, T. Q. Nguyen, "On the Fixed-Point Accuracy Analysis of FFT Algorithms", *IEEE TSP*, Vol 56, No. 10, Oct. 2008.

[24] C-H. Lin and A-Y. Wu, "Mixed-Scaling-Rotation CORDIC (MSR-CORDIC) Algorithm and Architecture for High-Performance Vector Rotational DSP Applications", *IEEE Trans. On Circuits and Systems-I (TCAS-I),* Vol. 52, No. 11, Nov. 2005, pp: 2385-2396.

[25] P. K. Meher, J. Valls, T-B. Juang, K. Sridharan and K. Maharatna, "50 Years of CORDIC: Algorithms, Architectures and Applications", *IEEE TCAS I,* Vol. 56, No. 9, Sept. 2009, pp: 1893-1907.

[26] A. B. Kinsman and N. Nicolici, "Bit-Width Allocation for Hardware Accelerators for Scientific Computing Using SAT-Modulo Theory", *IEEE Trans. on CAD,* Vol 29, No 3, March 2010, Page(s):405-413.

[27] M. Bekooij, J. Huisken, K. Nowak, "Numerical Accuracy of Fast Fourier Transforms with CORDIC Arithmetic", *Journal of VLSI Signal Processing,* Vol 25, No 2, pp: 187-193, 2000.

[28] K. Kota, J. R. Cavallaro, "Numerical Accuracy and Hardware Tradeoffs for CORDIC Arithmetic for Special-Purpose Processors", *IEEE Trans. On Computers,* Vol 42, Issue 7, pp: 769-779, July 1993.

[29] S. Y. Park, and N. Ik. Cho, "Fixed-Point Error Analysis of CORDIC Processor Based on the Variance Propagation Formula", *IEEE Trans. On Circuits and Systems (TCAS I),* Vol 51, No. 3, March 2004.