Finite Word Length Effect in Practical Block-Floating-Point FFT

Gil Naveh
Tel-Aviv Research Center
Huawei Technologies Co. Ltd
Tel-Aviv, Israel
Email: Gil.naveh@huawei.com

***Abstract*— Fixed-Point FFT implementation is very sensitive to finite-word-length-effects due to the large quantization noise that is being accumulated throughout the FFT stages. In FFT implementations on fixed register size processors like CPUs and DSPs, Block-Floating-Point (BFP) is a well-known scheme for controlling the tradeoffs between the fixed-point register size and the resultant accuracy. The performance of the ideal BFP FFT, in terms of the output Signal to Quantization Noise Ratio (SQNR), has been investigated in depth. However, ideal BFP-FFT suffers from implementation complexity, and especially non-deterministic latency. This is caused by the inherent mechanism that requires to re-calculate an entire FFT stage if one of the stage's output overflows. Because of this, most of the implementations are of a more practical variant for the BFP-FFT that does guarantee fixed latency. This, however, comes on the expense of reduced accuracy (degraded SQNR). In this paper, we derive the SQNR formulas for the practical BFP-FFT for radix-2 and radix-4 Cooley-Tukey Decimation-In-Time (DIT) FFTs. The derived model is compared to computer simulations and found highly accurate (less than 0.2dB difference). We use the derived model to compare the SQNR performance of the practical algorithm to the ideal one and show a 6-14dB penalty cost for guaranteeing fixed latency implementation.**

Keywords - Block Floating Point; Fixed Point; DIT; SQNR;

I. INTRODUCTION

The Fast Fourier Transform (FFT) serves as an important tool in many signal processing applications. Throughout the Years it has been successfully used in radar application, spectral analysis, filtering, voice enhancement, advanced audio codecs (like MP3 and AAC), and during the last three decades, with the introduction of multitone modulations, it is also being successfully used in wired and wireless modems such as discrete-multi-tone in Digital-Subscriber-Line (DSL) modems [1], Orthogonal-Frequency-Division-Modulation (OFDM) in several wireless modems, e.g., [2] and in advanced fiber optic modems [3].

Finite-word-length effects (denoted hereafter also as quantization noise) have substantial effect on the accuracy performance of FFTs. This is a result of the native characteristic of the FFT in which quantization noise that is added at the output of each stage of the FFT is accumulated toward the FFT output. Since the maximal value at each stage's output grows as we proceed with the stages [4], in many hardware implementations, the performance degradation due to the quantization noise is mitigated by adapting the register size at each stage to accommodate the signal growth [5]-[7]. On the other hand, in software implementations (as in CPUs and Digital Signal Processors - DSPs), or hardware implementations where intermediate values are forced to be written to memory, increasing the bit-width of the stored values is not possible. For those cases, a dynamic-scaling BFP based schemes are commonly used.

The straight-forward dynamic-scale BFP is such that throughout the calculation of each FFT stage, the butterflies' outputs are tested for an overflow. If an overflow is detected, the entire stage is recalculated and scaled down before stored to memory. The advantage of this BFP scheme is that the scale down is done only on a concrete need, which leads to the best accuracy performance among other BFP-FFT schemes. For that reason, we relate to the straight-forward dynamic-scale BFP-FFT as "ideal BFP-FFT" herein. The drawbacks of this scheme are its complexity and the fact that it results in non-deterministic latency. Deterministic latency may have high importance when the FFT is used within a synchronized pipelined system, such as a modulator or demodulator in OFDM modems [8].

Multiple schemes that overcome the non-deterministic latency drawback have been proposed, e.g., [9] [10], but they all involve non-negligible SQNR performance degradation as compared to the ideal BFP. Among the class of the deterministic latency BFP-FFTs, the one proposed by Shively [11] leads to the least SQNR loss as compared to the ideal BFP-FFT. Thanks to this fact, it turns to be among the most common schemes for practical implementations, e.g., [12] [13]. We refer to the Shively's scheme herein as "practical BFP-FFT".

The ideal BFP-FFT was originally analyzed in [14], which provided a lower and upper bound for the output quantization noise variance. In [4] and [9], a more accurate statistical model was used to project the expected value of the ideal BFP-FFT output noise power for an uncorrelated input sequence. Although the practical BFP-FFT is widely used in practical systems for deterministic latency BFP-FFT, to the best knowledge of the author, its accuracy performance has not been analyzed.

In this work, we refine the commonly used statistical model of quantization noise within FFTs, apply this refinement to the SQNR of the ideal BFP-FFT, and derive the analytical model of the SQNR of the practical BFP-FFT. We adapt the noise models to represent modern processor having

 ISBN: 978-1-68558-245-6

embedded complex multipliers and wide accumulators, and we evaluate the accuracy degradation of the practical BFP-FFT as compared to the ideal one.

The paper is organized as follows: Section II introduces the models used throughout the paper covering the DIT FFT model, the underline processor model, and the quantization noise models. In Section III the analytical SQNR formulas of a generic scaling policy are derived and in Section IV the associated scaling policies for the ideal and practical BFP FFT are described. Section V applies the SQNR formulas to the associated scaling policies while the results are presented in Section VI., Finally, conclusions are given in Section VII.

II. FFT, PROCESSOR AND QUANTIZATION NOISE MODELS

We relate to fixed-point representation of fractional datatypes. We assume a processor having registers of b bits (including sign) and accumulators of at least $B = 2b + \lceil \log_2 R \rceil + 1$ bits, where R is the FFT radix and $\lceil a \rceil$ is the smallest integer that is larger than a. The numbers represented by the registers are in 2's complement representation and in the range $-1 \le x \le 1 - 2^{-(b-1)}$. The numbers represented by the accumulators are in the rage $-2^{\lceil \log_2 R \rceil + 1} \le x < 2^{\lceil \log_2 R \rceil + 1}$. The width of the data stored to memory is always of b bits.

Our focus is of fixed-radix, Cooley-Tukey, DIT-FFTs of radix-2 and radix-4. A generic model of a finite-word-length radix-2/radix-4 butterfly of the DIT-FFT is given in Figure 1.

In the DIT topology the inputs loaded from the memory are first multiplied by the Twiddle Factors (TFs), w_N^{kn}, then multiplied by the butterfly's coefficients $\gamma_{r,t}; r, t \in \{0, 1, \dots, R-1\}$, and then summed up within the butterfly before being stored back to the memory. The processing model that we will deal here with is a model that is most common to DSPs and dedicated FFT processors. In this model the inputs x_n and the TFs w_N^{kn} are represented by b bits per component (b bits for the real component and b bits for the imaginary component) and are within the range of $\left[-1, 1 - 2^{-(b-1)}\right]$. When multiplied, the multiplication is spanned over $2b + 1$ bits (recalling that the TF multiplication is a complex multiplication). Since in radix-2 and radix-4 FFTs the butterfly's internal coefficients, $\gamma_{r,t}$, belong to the sets $\{1, -1\}$ and $\{1, -1, j, -j\}$; $j = \sqrt{-1}$ respectively, there are no truly multiplications within the butterfly. The bit-width of the butterfly's output can grow to span over up to B bits and then potentially scaled down by a factor of α, where we restrict α to be a power of 2. The scaled down butterfly output is quantized to b bits per component via rounding before being stored to memory.

The quantization model that we use here is the so-called Rounding-Half-Up (RHU) [15], which is also known as hardware-friendly-rounding and is being used in most digital signal processors and hardware implementations of digital signal processing functions. The mathematical function of RHU rounding to b bits is

$$y = Q[s] \triangleq 2^{-b} \cdot \lfloor s \cdot 2^b + 0.5 \rfloor \quad (1)$$

where $\lfloor a \rfloor$ is maximal integer lower than a and $s \in \left[-1, 1 - 2^{-(b-1)}\right]$. The quantization error is $v = s - y$ and in the general case is modeled as an additive noise having uniform distribution [16]

$$v \sim U[-2^{-b}, 2^{-b}) \quad (2)$$

and is independent of s. As we deal here with finite-word-length, in fact v has a discrete distribution. However, for large enough b it is common to treat it as a zero mean continuous uniform distribution. As such its variance is

$$\sigma_v^2 = \frac{2^{-2(b-1)}}{12}. \quad (3)$$

In addition, throughout the FFT there are plenty of cases where all the TFs preceding a given butterfly are among the set

$$\mathcal{T}_1 \triangleq \{1, -1, j, -j\} \; ; \; j = \sqrt{-1}. \quad (4)$$

In such cases, the multiplication of a b-bits value $x \in \left[-1, 1 - 2^{-(b-1)}\right]$ by the TF $w \in \mathcal{T}_1$ would result in a $2b$-bits number, $t = w \cdot x$, that it's lower b bits are equal to zero. If all the TFs preceding a given butterfly are among the set $\mathcal{T}_1$, then the lower b bits of the butterfly's outputs, before down scaling, are also equal to zero. When such a number is scaled down by very few bits, the quantization noise does not obey to the uniform distribution anymore [16]. In this case we get a Random Variable (RV) having discrete distribution and non-zero mean. For example, in the case that such a number is shifted one bit to the right, the quantization noise ε_1 is distributed as

$$\varepsilon_1 = \begin{cases} 0 & w.p.\, 0.5 \\ -\frac{1}{2} 2^{-(b-1)} & w.p.\, 0.5 \end{cases}, \quad (5)$$

where the subscript 1 in ε_1 refers to the case of quantization noise generated by right shift of the b-bits number by one bit. The expected value of this noise equals $-2^{-(b-1)}/4$ and hence when dealing with Signal-to-Quantization-Noise-Ratios of those RVs we will relate to the noise power rather than to its variance. To distinguish the power from the variance we use the symbol ρ^2 for power. The expected value of the power of ε_1 RV then is

$$\rho_{\varepsilon_1}^2 = \frac{1}{2} \cdot 0 + \frac{1}{2} \cdot \left(\frac{1}{2} 2^{-(b-1)}\right)^2 = \frac{2^{-2(b-1)}}{8}. \quad (6)$$

As expected, this is larger than the variance of the zero mean uniformly distributed quantization noise of (3). In a similar way we can calculate the noise power of quantization noises that are generated due to the rounding after right shift of a b-bits number by q bits. In most FFT topologies and radices up to Radix-5, the right shifts are in the range of 0 to 3. Moreover, for right shifts of 4 and above the quantization noise power is very close to the variance of the zero mean uniform quantization noise of (3). Therefore, for our analytical derivations we use

$$\rho_{\varepsilon_q}^2 = \begin{cases} 0 & ; \quad q = 0 \\ \frac{1}{8}2^{-2(b-1)} & ; \quad q = 1 \\ \frac{3}{32}2^{-2(b-1)} & ; \quad q = 2 \\ \frac{11}{128}2^{-2(b-1)} & ; \quad q = 3 \\ \frac{1}{12}2^{-2(b-1)} & ; \quad q \geq 4\,. \end{cases} \tag{7}$$

In the sequel, we designate the set of butterflies that all their inputs were multiplied by TFs belonging to $\mathcal{T}_1$, as the $\mathcal{B}_1$ set or $\mathcal{B}_1$ butterflies.

III. SQNR OF A GENERIC BFP-FFT

By "generic BFP-FFT" we refer to a BFP-FFT that incorporates policy for down-scaling by right shifts at the outputs of the FFT stages, where the decision at which stages to scale down and by what factor are the policy parameters. In the following paragraphs we will relate to specific BFP scaling policies and will analyze their SQNR performance. We assume zero mean i.i.d. input sequence, $x(n)$, and that the quantization is regarded as an i.i.d. noise source. Moreover, multiple quantization noises at the input to a given butterfly that have been generated at earlier stages are mutually uncorrelated [9]. In order to derive the analytical expression of the SQNR, we will adopt the analysis strategy of Weinstein [9]. Let us relate to an input sequence of length N, $x(n)$, and a fixed-radix FFT of radix R. Define $M = \log_R N$, and α_m as the scale value at the output of the m^{th} stage, $m \in \{1, 2, \dots, M\}$, where we restrict α_m to be of the form $\alpha_m = 2^{-q_m}$ and q_m is a positive integer. We denote $x_m(n)$ as the array values at the output of the m^{th} stage, where $x_M(k) \triangleq X(k)$ is the FFT output, and $x_0(n) \triangleq x(n)$ is the FFT input. For a zero mean, i.i.d. sequence $x(n)$, the variance of the signal at the FFT output is given by

$$\sigma_{x_M}^2 = N\sigma_{x_0}^2 \prod_{m=1}^{M} \alpha_m^2 = N\sigma_{x_0}^2 2^{-2\sum_{m=1}^{M} q_m}\,. \tag{8}$$

The noise at the output of a given butterfly is composed of two components: the noise that is generated by that particular butterfly, which we call butterfly self-noise, and the noise that is propagated through the butterfly (noise that was generated at earlier stages), which we call propagated-noise. The propagated-noise power is multiplied by a factor of $R\alpha^2$ as each butterfly output is composed of the sum of R i.i.d. noise values and is multiplied by a scaling factor α. The self-noise, v, is the noise generated by the quantization at the butterfly output after being multiplied by α as depicted in Figure 1. Its variance is denoted as σ_v^2 (or power of ρ_v^2). Looking at the output noise of an M stages FFT, it is observed that the noise from the first stage propagates through the following M-1 stages, which results in accumulation of R^{M-1} such i.i.d. noise sources, each attenuated by a factor of $\prod_{m=2}^{M} \alpha_m^2$. The propagation of the noise from the second stage results in accumulation R^{M-2} such i.i.d. noise sources, each attenuated by a factor of $\prod_{m=3}^{M} \alpha_m^2$, and so on. The total output noise variance, σ_E^2, for an M stages FFT, assuming all the quantization operations are modeled as uniform RVs, $U[-2^{-b}, 2^{-b})$, is given by the following expression

$$\begin{aligned} \sigma_E^2 &= \sigma_v^2 \left(1 + \sum_{m=1}^{M-1} \prod_{i=m+1}^{M} R\alpha_i^2\right) \\ &= \sigma_v^2 \left(1 + \sum_{m=1}^{M-1} R^{M-m} \prod_{i=m+1}^{M} \alpha_i^2\right). \end{aligned} \tag{9}$$

In (9) it was assumed that the self-noise is a continuous RV and have the same PDF at all the butterflies. For b sufficiently large (e.g., $b = 16$) this assumption is commonly accepted. However, this is not the case for butterflies belonging to the $\mathcal{B}_1$ set in which their outputs are discrete RVs with Probability-Mass-Function (PMF) that depend on the number of right shifts took place at the butterfly output. The power of those noise sources is larger than that of the uniform RV, and hence they have negative effect on the quantization noise power at the FFT output. In order to be able to evaluate the effect of those noise sources, we want to incorporate their statistical model in the derivation of ρ_E^2.

Let us denote by β_m the fraction of the butterflies belonging to the $\mathcal{B}_1$ set at stage m, and by $\rho_{q_m}^2$ the self-noise power at the output of those butterflies. Using those notations, and relating to power-of-two FFTs, we can now re-write (9) as

$$\begin{aligned} \rho_E^2 = \sigma_v^2 &\sum_{m=1}^{M} R^{M-m+1} \prod_{i=m+1}^{M+1} \alpha_i^2 \\ &+ \sum_{m=1}^{M} \beta_m \left(\rho_{q_m}^2 - \sigma_v^2\right) R^{M-m+1} \prod_{i=m+1}^{M+1} \alpha_i^2\,, \end{aligned} \tag{10}$$

where we defined a virtual α_{M+1} set to $\alpha_{M+1} = 1/\sqrt{R}$.
The second term in (10) is a positive quantity that represents the increased output noise power caused by butterflies of the set $\mathcal{B}_1$. As we are dealing with power-of-two DIT FFTs, we can write the precise expression of β_m as a function of the radix R. This is easily extracted from the flow graphs of those FFTs and is equal to

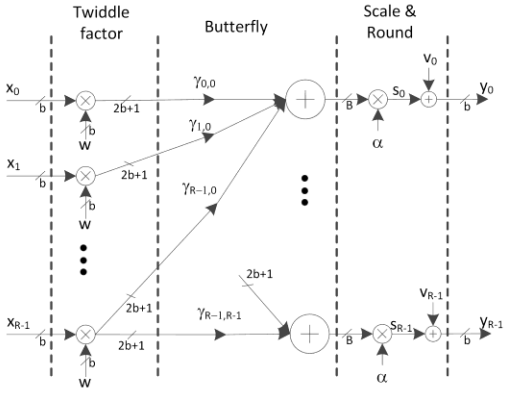

Figure 1. Generic model of DIT FFT Butterfly

$$\beta_m(R) = \begin{cases} R^{-(m-1)} & ;\ R > 2 \\ 1 & ;\ R = 2, m = 1 \\ R^{-(m-2)} & ;\ R = 2, m > 1\,. \end{cases} \quad (11)$$

Now we can plug β_m into (10) and get for $R = 2$

$$\begin{aligned} \rho_E^2 = \sigma_v^2 \sum_{m=1}^{M} R^{M-m+1} \prod_{i=m+1}^{M+1} \alpha_i^2 \\ + \left(\rho_{q_1}^2 - \sigma_v^2\right) R^M \prod_{i=2}^{M+1} \alpha_i^2 \\ + \sum_{m=2}^{M} \left(\rho_{q_m}^2 - \sigma_v^2\right) R^{M-2m+3} \prod_{i=m+1}^{M+1} \alpha_i^2 \end{aligned} \quad (12)$$

and for $R > 2$

$$\begin{aligned} \rho_E^2 = \sigma_v^2 \sum_{m=1}^{M} R^{M-m+1} \prod_{i=m+1}^{M+1} \alpha_i^2 \\ + \sum_{m=1}^{M} \left(\rho_{q_m}^2 - \sigma_v^2\right) R^{M-2m+2} \prod_{i=m+1}^{M+1} \alpha_i^2\,. \end{aligned} \quad (13)$$

Using (8), (12) and (13), the SQNR for a given scale pattern, $\boldsymbol{q} = [q_1, q_2, \dots, q_M]$, can be calculated by $\sigma_{x_M}^2/\rho_E^2$ where assigning $\alpha_i = 2^{-q_i}$.

IV. SCALING POLICIES

In most FFT realizations, we wish to select a scaling policy that maximizes the SQNR under the constraint of zero-overflows. At the ideal BFP-FFT, the scaling policy is such that throughout the butterflies' computation, every butterfly's output is tested for an overflow before it is quantized down to b bits. If the real or the imaginary components of the butterfly output overflows, the entire stage is re-calculated where the butterflies' outputs are scaled down by q bits before being rounded to b bits and stored to memory. The value q is selected to guarantee that the scaled result does not overflow anymore. For example, if one of the absolute values of the real or imaginary butterfly's outputs is within the range $\left[1, 2 - 2^{-(b-1)}\right]$, the entire stage will be re-calculated while the butterflies' outputs will be scaled by one bit to the right ($q = 1$). If one of the absolute values of the real or imaginary butterfly's outputs is within the range $\left[2, 4 - 2^{-(b-1)}\right]$, the entire stage will be re-calculated while the butterflies' outputs will be scaled by two bits to the right and so on. The more common, fixed latency policy proposed by Shively [11] guarantees deterministic latency at the expense of decreased SQNR. In this policy, the decision by what factor to down-scale the outputs of stage m is taken based on the values of the outputs of stage $m-1$, which are guaranteed to fit in the range $\left[-1\,, 1 - 2^{-(b-1)}\right]$. While writing the outputs of stage $m-1$ to the memory, the processor finds the maximal absolute value among the real and imaginary components of the whole stage, which serves for the down-scaling decision for the next stage. The down-scaling criteria is similar to that being used at the ideal BFP-FFT, i.e., to guarantee zero overflow at the output of the next stage. Here, there is a need to consider the fact that the maximal absolute value at the next stage (stage m) butterflies' output would grow by a factor that is between 1 and $\sqrt{2}R$ relative the outputs of the current stage (stage $m-1$). In order to formalize this, let us define $x_m^c(n)$ for $n \in \{0, 1, \dots, N-1\}$ as

$$\begin{aligned} x_m^c(2n) &= real\left(x_m(n)\right) \\ x_m^c(2n+1) &= imag\left(x_m(n)\right) \end{aligned} \quad (14)$$

and

$$\tilde{x}_m = \max_n \{|x_m^c(n)|\}\,. \quad (15)$$

Using those, the scaling policy of the practical BFP-FFT can be written as

$$q_m = \begin{cases} 0 & ;\ \tilde{x}_{m-1} < \frac{1}{\sqrt{2}R} \\ 1 & ;\ \frac{1}{\sqrt{2}R} \le \tilde{x}_{m-1} < \frac{2}{\sqrt{2}R} \\ 2 & ;\ \frac{2}{\sqrt{2}R} \le \tilde{x}_{m-1} < \frac{4}{\sqrt{2}R} \\ & \vdots \\ & \vdots \\ \lceil log_2(R) \rceil + 1 & ;\ \frac{1}{\sqrt{2}} \le \tilde{x}_{m-1} \end{cases}. \quad (16)$$

V. SQNR CALCULATION

It is now clear that the SQNR at the FFT output of a particular realization of the FFT depends on the scale pattern that has been used throughout this realization. Each scale pattern, $\boldsymbol{q}$, is associated with a resultant SQNR. We adopt Weinstein's definition for "theoretical" SQNR as the weighted sum of the SQNR per scale pattern over all possible patterns [9]. The probability of a scale pattern depends solely on the PDF of the input sequence and the scaling policy. In the sequel we will derive the scale patterns probabilities as well as the SQNR of the practical BFP-FFT and of the ideal BFP-FFT algorithms for Gaussian input sequences.

A. *Scale patterns probabilities of practical BFP-FFT*

We start with the derivation of the probabilities of scale patterns. Given the practical BFP-FFT's scaling policy, the probability that there will be exactly $q > 0$ right shifts at stage m is equal to

$$\begin{aligned} Pr(q_m = q) &= Pr\left(\frac{2^{q-1}}{\sqrt{2}R} \le \tilde{x}_{m-1} \le \frac{2^q}{\sqrt{2}R}\right) \\ &= Pr\left(-\frac{2^q}{\sqrt{2}R} \le \underset{n}{all}\{x_{m-1}^c(n)\} \le \frac{2^q}{\sqrt{2}R}\right) \\ &\quad - Pr\left(-\frac{2^{q-1}}{\sqrt{2}R} \le \underset{n}{all}\{x_{m-1}^c(n)\} \le \frac{2^{q-1}}{\sqrt{2}R}\right) \end{aligned} \quad (17)$$

whereas for $q = 0$

$$Pr(q_m = 0) = Pr\left(\tilde{x}_{m-1} \le \frac{1}{\sqrt{2}R}\right). \quad (18)$$

By the assumption that the input sequence, $x_{m-1}^c(n); n \in$

$\{0, 1, \dots, 2N-1\}$ is an i.i.d. sequence, (17) and (18), can be written as

$$Pr(q_m = q) = \left[Pr\left(-\frac{2^q}{\sqrt{2}R} \le x_{m-1}^c(n) \le \frac{2^q}{\sqrt{2}R}\right)\right]^{2N} - \left[Pr\left(-\frac{2^{q-1}}{\sqrt{2}R} \le x_{m-1}^c(n) \le \frac{2^{q-1}}{\sqrt{2}R}\right)\right]^{2N} \quad (19)$$

whereas for $q = 0$

$$r(q_m = 0) = \left[Pr\left(-\frac{1}{\sqrt{2}R} \le x_{m-1}^c(n) \le \frac{1}{\sqrt{2}R}\right)\right]^{2N}. \quad (20)$$

We now define the following auxiliary variables

$$Q_m = \sum_{i=1}^{m} q_i \ ; \ m \in \{1, 2, \dots, M\} \ , \ \ Q_0 = 1 \quad (21)$$

and

$$T_m = 2^{-2Q_m}. \quad (22)$$

Using those, the variance of the sequence at the output of the m^{th} stage is

$$\sigma_{x_m}^2 = \sigma_{x_0}^2 R^m T_m \quad (23)$$

and the variance of the real and imaginary individual components at the output of the m^{th} stage is $\sigma_{x_0}^2 R^m T_m/2$. For an i.i.d complex Gaussian input sequence, $x_0^c(n) \sim N\left(0, \sigma_{x_0}^2/2\right)$; $n \in \{0, 1, \dots, 2N-1\}$, it can be shown that all the intermediate sequences $x_m^c(n)$, $m \in \{1, 2, \dots, M\}$ are also Gaussian i.i.d [9]. Therefore, the probability that the outputs of the m^{th} stage would be shifted by exactly $q_m > 0$ right shifts, given that there were accumulated Q_{m-1} right shifts at the stages preceding stage m is

$$Pr\left(q_m \mid Q_{m-1}; \sigma_{x_0}^2\right) = \left[erf\left(\frac{2^{q_m}}{\sigma_{x_0}\sqrt{2R^{m+1}T_{m-1}}}\right)\right]^{2N} - \left[erf\left(\frac{2^{q_m-1}}{\sigma_{x_0}\sqrt{2R^{m+1}T_{m-1}}}\right)\right]^{2N} \quad (24)$$

and the probability that there would be no right shifts ($q_m = 0$) is given by

$$r\left(q_m = 0 \mid Q_{m-1}; \sigma_{x_0}^2\right) = \left[erf\left(\frac{1}{\sigma_{x_0}\sqrt{2R^{m+1}T_{m-1}}}\right)\right]^{2N} \quad (25)$$

where $erf(x)$ is defined by

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt. \quad (26)$$

B. *Scale patterns probabilities of ideal BFP-FFT*

At the scaling policy of the ideal BFP-FFT there are no per-stage scaling pre-decisions. An FFT stage is calculated without scaling and throughout the calculations, if any of the stage's outputs overflows, the whole stage is re-calculated while the outputs are down-scaled before being written to memory. Note that in the ideal policy there may be multiple re-calculation of the same stage if the strategy is to initiate the re-calculation upon the first detected overflowed value. Different strategies may eliminate the multi re-calculations of the same stage, for example set the scale value to the maximal scale upon the detection of the first overflow, or always calculate the stage to its end and if overflows have been detected, set the scale value according the largest magnitude among the overflowed values. Some strategies suffer degradations in the SQNR performance due to potential mismatch between the scale value and the actual overflowed value. Nevertheless, here, for the sake of SQNR comparison, we assume a strategy that determine scale value according to the largest magnitude output sample, and hence no performance loss is involved.

As opposed to the practical case, at which the scale decision for stage m depends on the outputs of stage $m-1$ after being scaled down, the scale decision of the ideal BFP-FFT depend on the outputs of stage m before being scaled down. Let us denote those values as $s_m(n)$, i.e.

$$x_m(n) = \alpha_m s_m(n) \quad (27)$$

and define $s_m^c(n)$ and $\tilde{s}_m$ in analogous to (14) and (15) as

$$s_m^c(2n) = real\left(s_m(n)\right) \\ s_m^c(2n+1) = imag\left(s_m(n)\right) \quad (28)$$

and

$$\tilde{s}_m = \max_n\{|s_m^c(n)|\}. \quad (29)$$

Now, the SQNR analysis using the ideal BFP-FFT policy follows the steps of the analysis of the practical BFP-FFT scheme. The output signal variance and the output noise power follow (8) and (10) respectively. The probability that there will be exactly $q > 0$ right shifts at stage m is equal to

$$Pr(q_m = q) = Pr(2^{q-1} \le \tilde{s}_m \le 2^q) = Pr\left(-2^q \le \underset{n}{all}\{s_m^c(n)\} \le 2^q\right) - Pr\left(-2^{q-1} \le \underset{n}{all}\{s_m^c(n)\} \le 2^{q-1}\right), \quad (30)$$

and the probability that there will be no right shifts at stage m, i.e. $q = 0$, is

$$Pr(q_m = 0) = Pr(\tilde{s}_m \le 1) = Pr\left(-1 \le \underset{n}{all}\{s_m^c(n)\} \le 1\right). \quad (31)$$

Under i.i.d. Gaussian input assumption, we get for $q_m > 0$

$$Pr\left(q_m \mid Q_{m-1}\,;\,\sigma_{x_0}^2\right) = \left[erf\left(\frac{2^{q_m}}{\sigma_{x_0}\sqrt{R^m T_{m-1}}}\right)\right]^{2N} - \left[erf\left(\frac{2^{q_m-1}}{\sigma_{x_0}\sqrt{R^m T_{m-1}}}\right)\right]^{2N}, \quad (32)$$

and for $q_m = 0$

$$Pr\left(q_m = 0 \mid Q_{m-1}\,;\,\sigma_{x_0}^2\right) = \left[erf\left(\frac{1}{\sigma_{x_0}\sqrt{R^m T_{m-1}}}\right)\right]^{2N}. \quad (33)$$

C. SQNR calculation

We use the per-stage probabilities to calculate the probability of a specific scale pattern, $\boldsymbol{q} = [q_1, q_2, \dots, q_M]$,

$$Pr\left(\boldsymbol{q}; \sigma_{x_0}^2\right) = Pr\left(q_1; \sigma_{x_0}^2\right) \prod_{m=2}^{M} Pr\left(q_m | Q_{m-1}; \sigma_{x_0}^2\right) \quad (34)$$

and the output SQNR is calculated by the weighted sum of the SQNRs per scale pattern as

$$SQNR = \sum_{\boldsymbol{q}} Pr\left(\boldsymbol{q}; \sigma_{x_0}^2\right) \cdot SQNR\left(\boldsymbol{q}, \sigma_{x_0}^2\right) = \sum_{\boldsymbol{q}} Pr\left(\boldsymbol{q}; \sigma_{x_0}^2\right) \cdot \frac{\sigma_{x_M}^2\left(\boldsymbol{q}, \sigma_{x_0}^2\right)}{\rho_E^2(\boldsymbol{q})}. \quad (35)$$

In (35) the expression $Pr\left(\boldsymbol{q}; \sigma_{x_0}^2\right)$ is calculated by (34), $\sigma_{x_M}^2\left(\boldsymbol{q}, \sigma_{x_0}^2\right)$ is calculated by (8) and $\rho_E^2(\boldsymbol{q})$, with $\alpha_i = 2^{-q_i}$, is calculated by (12) or (13) for Radix-2 and Radix-4 respectively.

VI. RESULTS

The derived models of the SQNR for the practical and the ideal BFP-FFT have been validated against simulation. The model and the simulation results for 16-bit datatype ($b = 16$) and Gaussian i.i.d input with standard deviation of $\sigma_{x_0} = 0.15$ are shown in Figure 2 and Figure 3 for radix-2 and radix-4 respectively. For the simulation results we have averaged the SQNR of 1000 FFT runs per FFT length. As can be seen, there is a very good match between the simulation results and the derived model. The gap between the refined statistical model (that incorporate the refinement for $\mathcal{B}_1$ butterflies) and the simulation result for the practical BFP-FFT is in the order of 0.2dB. The results for the ideal BFP-FFT are not shown in the figures since the model has almost perfect match to the simulation result with gaps that are in the order of 0.05dB.

In Figure 2 and Figure 3 we can also see the effect of the refined statistical model for the $\mathcal{B}_1$ butterflies. The model neglecting the effects of the $\mathcal{B}_1$ butterflies, for radix-2 BFP-FFT, is optimistic by about 0.5dB for the practical BFP-FFT and by about 1dB for radix-4.

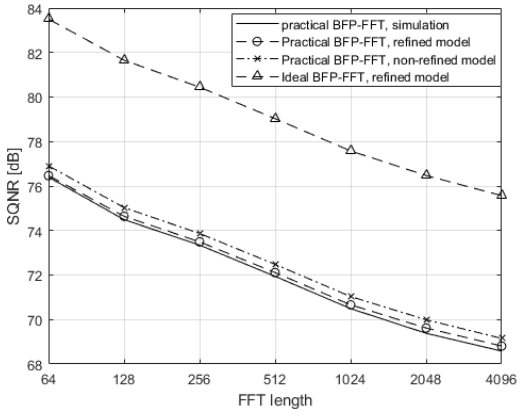

Figure 2. Radix-2 Practical BFP-FFT

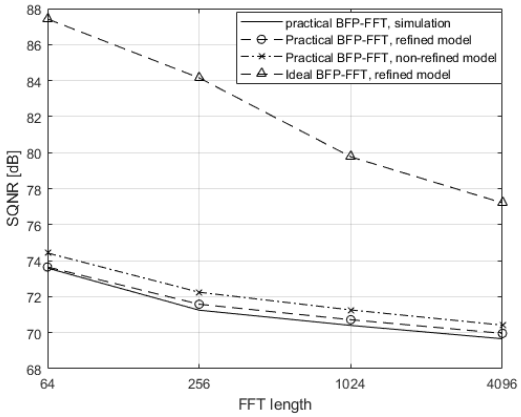

Figure 3. Radix-4 Practical BFP-FFT

One of the main goals of the paper is to provide an analytical tool that enables the prediction of the SQNR penalty one needs to pay for getting fixed latency BFP-FFT. This penalty is clearly seen for radix-2 and radix-4 in Figure 2 and Figure 3 respectively. We see that such a penalty is in the order of 6dB when the number of stages is above five, and grows up to 13.5dB for lower number of stages as seen at the case of 64 points radix-4 FFT.

Another interesting observation that the model reveals relates to the comparison of the SQNR between radix-2 and radix-4 BFP-FFT implementations. It is well known that from complexity perspective, the radix-4 has advantages over radix-2 (at least in the number of multiplications). From the results in Figure 2 and Figure 3, we can also see that radix-4 have better SQNR in the ideal BFP-FFT implementation. We get 4dB advantage for 64-points FFT down to about 2dB advantage for 4096-points FFT. However, for the practical BFP-FFT we see an opposite behavior. The radix-2 practical BFP-FFT results in 2.8dB better SQNR for 64-point FFT, down to 1.2dB better SQNR for 4096-points FFT.

VII. CONCLUSIONS

In this paper, we refined the analytical model of the finite-word-length-effects of Cooley Tukey DIT BFP-FFT to incorporate butterflies belonging to the $\mathcal{B}_1$ set, as well as extended the model for the commonly used practical BFP-FFT. The refined analytical model was validated against simulation and found highly accurate for ideal and practical BFP-FFTs. The model enables to accurately predict the SQNR for the practical BFP-FFT and the performance degradation compared to the ideal BFP-FFT scheme.

The analysis covers DIT-FFT for radix-2 and radix-4, but can be easily adapted to DIF FFT topologies and be extended for non-power-of-2 BFP-FFTs as well as for mixed radices, such as the ones used in LTE wireless modems.

REFERENCES

[1] J. M. Cioffi, at al., "Very-high-speed digital subscriber lines," *IEEE Communications Magazine*, vol. 37, no. 4, pp. 72-79, 1999.

[2] B. F. Frederiksen and R. Prasad, "An overview of OFDM and Related Techniques Towards Development of Future Wireless Multimedia Communications," in *IEEE Proc. Radio and Wireless Conference*, Boston, pp. 19-22, 2002.

[3] N. Cvijetic, "OFDM for Next-Generation Optical Access Networks," *IEEE Journal of Lightwave Technology*, vol. 30, no. 4, pp. 384-398, 2012.

[4] A. V. Oppenheim and C. J. Weinstein, "Effects of Finite Register Length in Digital Filtering and the Fast Fourier Transform," *Proceedings of the IEEE*, vol. 60, no. 8, pp. 957-976, 1972.

[5] W.-H. Chang and N. Q. Truong, "On the Fixed-Point Accuracy Analysis of FFT Algorithms," *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp. 4973-4682, 2008.

[6] P. Gupta, "Accurate Performance Analysis of a Fixed Point FFT," in *Twenty Second National Conference on Communication (NCC)*, Guwahati, 2016.

[7] A. Monther and K. Zsolk, "Analysis of Quantization Noise in FFT Algorithms for Real-Valued Input Signals," in *International Conference on Radioelektronika*, Kosice, 2022.

[8] *LTE-A; Evolved Universal Terrestrial Radio Access (E-UTRA), Physical Channels and Modulation*, 3GPP TS 36.211, 2011.

[9] C. J. Weinstein, "Quantization Effects in Digital Filters," M.I.T. Lincoln Lab. Tech. Rep. 468, ASTIA doc. DDC AD-706862, 1969.

[10] Tran-Thong and B. Liu, "Fixed-Point Fast Fourier Transform Error Analysis," *IEEE Transactions on Acoustic, Specch, and Signal Processing*, vol. 24, no. 6, pp. 563-573, 1976.

[11] R. R. Shively, "A Digital Processor to Generate Spectra in Real Time," *IEEE Transactions on Computers*, Vols. C-17, no. 5, pp. 485-491, 1968.

[12] H. G. Kim, K. T. Yoon, J. S. Youn, and J. R. Choi, "8K-point Pipelined FFT/IFFT with Compact Memory for DVB-T using Block Floating-point Scaling Technique," in *International Symposium on Wireless Pervasive Computing (ISWPC)*, Melbourne, pp. 41-47, 2009.

[13] S. J. Huang and S. G. Chen, "A High-Parallelism Memory-Based FFT Processor with high SQNR and novel addressing scheme," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, Montreal, pp. 2671-2674, 2016.

[14] P. D. Welch, "A Fixed-Point Fast Fourier Transform Error Analysis," *IEEE Transactions on audio and Electroacoustics*, vol. 17, no. 2, pp. 151-157, 1969.

[15] L. Xia, M. Athonissen, M. Hochstenbach, and B. Koren, "Improved Stochastic Rounding," *arXiv , 2020, Available: https://arxiv.org/abs/2006.00489.*

[16] B. Widrow, I. Kollar, and M.-C. Liu, "Statistical theory of Quantization," *IEEE Transactions on Instrumentation and Measurement*, vol. 45, no. 2, pp. 353-361, 1996.