Finite Wordlength Design for VLSI FFT Processors

Randall B. Perlow and Tracy C. Denk
Broadcom Corporation
16215 Alton Parkway
Irvine, CA 92618
Email: rperlow@broadcom.com, tdenk@broadcom.com

Abstract

Resource efficient FFT processors have become a common requirement for high-speed xDSL and OFDM transceivers. Hardwired VLSI implementations often result in smaller area and lower power consumption than general-purpose DSP processors. For hardwired designs, intelligent wordlength selection can be employed to further reduce hardware resource requirements. This paper describes a technique for quick and accurate estimation of FFT noise performance by modeling the FFT as a series of amplifier stages. The technique is employed to specify wordlengths that provide good tradeoffs between noise performance and hardware requirements. The technique is also used to show that the decimation-in-time radix-2 FFT algorithm has better finite wordlength properties than the decimation-in-frequency radix-2 FFT algorithm.

1: Introduction

Resource efficient Fast Fourier Transform (FFT) processors have become a common requirement for many high-speed communications applications including xDSL and OFDM transceivers. For some applications, FFT processing can be handled by programmable digital signal processors that use fixed-precision hardware multipliers and fixed-wordlength memory. These programmable digital signal processors are typically suboptimal for dedicated FFT processing because the same maximum hardware resources are applied at every stage of the FFT. For most applications, custom hardwired designs can be tailored by reducing the arithmetic precision at each stage to allow a tradeoff between the amount of required hardware resources and output signal integrity. This requires, however, that the designer choose a method for determining where to trim resources and understand the impact that these choices have on overall performance.

A first order analysis of quantization effects in fixed-point FFT algorithms was presented in [1]. In this analysis, roundoff noise was added at the output of each fixed-point multiplication, but quantization was not considered for the following:

1) Roundoff of the complex twiddle factors
2) Roundoff at the output of the butterfly computation adders
3) Nonuniform quantization across radix-2 stages – the analysis assumes that the intermediate result at all stages is stored in a $(B+1)$-bit register.

Also, the presented analysis does not account for the fact that that some twiddle factors are trivial (multiplication by ± 1 or $\pm j$) and do not contribute to the total noise.

The analysis in this paper considers all of these factors and leads to a systematic technique for quick and accurate estimation of FFT noise performance.

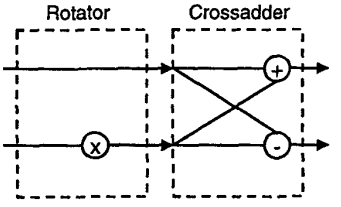

Figure 1. Simple butterfly

2: FFT flowgraph

A complete treatment of FFT flowgraphs and their derivation is given in [1]. In this paper, we briefly review only some key concepts that are required for examining the finite-precision noise effects.

The key structure in any FFT architecture is the "butterfly" computation. The simplified butterfly computation requires a single complex multiplier and two complex adders as shown in Figure 1. Since the datapath for an FFT

0-7803-7147-X/01/$10.00©2001 IEEE

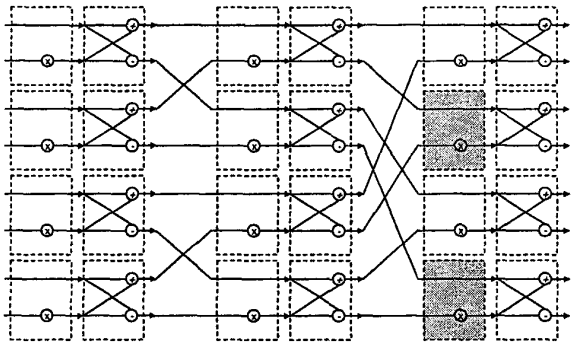

Figure 2. 8-FFT decimation-in-time algorithm

is a simple cascade of these butterflies as shown in Figure 2, the focus of this paper is on the effects of finite precision within the butterfly computation and the propagation of these effects to the output of the FFT.

For the purpose of analyzing finite precision effects in the FFT, it is convenient to divide the butterfly computation into a cascade of two distinct stages that we shall refer to as the "rotator" and the "crossadder" stages as shown in Figure 1. Every datapath through the FFT passes through a cascade of alternating rotator and crossadder stages.

In Section 3, the rotator and crossadder are both represented using the same simple noise propagation model at the top of Figure 3. This model, which treats every processing stage as a signal amplifier combined with an additive noise source, allows a very fast and accurate method for predicting noise performance in the FFT.

The network of butterfly stages in Figure 2 is replaced by the simple string of amplifier stages shown at the bottom of Figure 3. Since an N-point FFT normally requires $\log_2 N$ stages of butterfly operations, the equivalent noise propagation model has $2\log_2 N$ amplifier stages, one representing each rotator and crossadder stage. This model is used to represent the path from any FFT input to any FFT output.

If the power gain at stage k is G_k, and the additive noise power is σ_k^2, then the signal power, S_k, and noise power, N_k, at the output of stage k are given by the simple recursion

$$S_k = G_k S_{k-1} \tag{1}$$

$$N_k = G_k N_{k-1} + \sigma_k^2 \tag{2}$$

and the output signal-to-noise ratio at stage k is simply $SNR_k = S_k/N_k$. These recursions allow the effect of finite wordlength decisions to be quickly estimated in a powerful and systematic manner.

The analysis in this paper applies to any FFT architecture that computes all butterflies in the same column of the

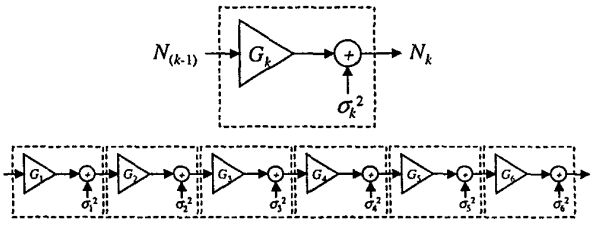

Figure 3. Simplified noise propagation model

FFT flowgraph using the same wordlengths. Two examples are fully-parallel designs and the serial designs suggested in [2].

3: Noise propagation models

In the analysis that follows we focus on the *average* signal-to-noise power that exists at the output of each stage in an FFT architecture. To do this we model the datapath through each rotator and crossadder stage as the single-input, single-output noise amplifier in Figure 3.

Examination of Figure 2 shows that not all datapaths are alike. For example, the datapath that runs directly from the top input to the top output does not pass through any complex multipliers. In contrast, the datapath that runs from the bottom input to the top output passes through three complex multipliers. Because all datapaths are different, the finite precision effects along each datapath are also different. We note however that the interconnectivity present in the FFT combines the finite-precision effects from different datapaths. This combination averages the noise along different paths, and justifies the use of an "average" noise propagation model. Our model employs noise averaging along columns so that a single noise propagation model replaces each column of processing.

In the following, we analyze the finite precision effects and develop average noise propagation model parameters for both the crossadder and the rotator.

3.1: Crossadder

The datapath for each output of the complex crossadder is a simple adder shown in Figure 4. The datapath is the same for both the real and imaginary outputs, and the inversion that is required for the subtractor leg has no impact on the noise performance.

Each B_x-bit input is the sum of the desired input signal $\{x_1, x_2\}$ and an accumulated rounding noise from the previous stage $\{n_{x1}, n_{x2}\}$. To avoid overflow or saturation at the output of the adder, one bit of precision must be added at the output of the adder such that $B_y = B_x+1$. If desired, wordlength growth can be avoided in the butterfly stage if

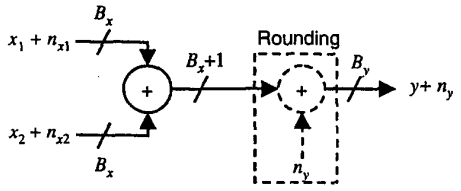

Figure 4. Crossadder datapath

the LSB is rounded. In this case, the output wordlength for the next stage is maintained at B_x-bits at the cost of additional roundoff noise.

To analyze noise propagation in the crossadder, all input signal components $\{x_1, x_2, n_{x1}, n_{x2}\}$ are modeled as zero-mean uncorrelated random variables. The desired inputs $\{x_1, x_2\}$ are modeled with variance $S_{(k-1)}$, and $\{n_{x1}, n_{x2}\}$ with variance $N_{(k-1)}$. Two cases are considered:

3.1.1 Without output rounding

In the absence of output rounding, the desired output signal y has variance $S_k = 2S_{k-1}$, and the undesired output noise n_y has variance $N_k = 2N_{k-1}$. Since, the *SNR* is unchanged, the crossadder acts like an ideal amplifier with power gain and additive noise

$$G_k = 2 \tag{3}$$

$$\sigma_k^2 = 0 \tag{4}$$

3.1.2 With output rounding

If the LSB is rounded off to reduce the required output precision, the desired output signal y is scaled in amplitude by a factor of 2. The output variance is then $S_k = S_{k-1}/2$. The noise output is also scaled, and the rounding introduces a zero-mean roundoff noise with variance σ_{nO}^2. The output noise power is $N_k = N_{k-1}/2 + \sigma_{nO}^2$.

For unbiased rounding of the LSB, the variance of the error is $\sigma_{nO}^2 = 1/8$. Therefore, the rounded crossadder is modeled by the cascade of an ideal amplifier with power gain and additive noise

$$G_k = 1/2 \tag{5}$$

$$\sigma_k^2 = 1/8 \tag{6}$$

3.2: Rotator

The processing performed by the rotator can be characterized by two classes: *trivial* and *nontrivial* rotation. Trivial rotation – multiplication by ± 1 or $\pm j$ – introduces no additive noise to the output. Trivial rotation occurs in the upper leg of the complex rotator (see Figure 1) and also occurs in the lower leg when the rotation factor is ± 1 or $\pm j$.

In contrast, nontrivial rotation – multiplication by anything other than ± 1 or $\pm j$ – will introduce finite precision effects due to rounding at the output of the multiplier. This section starts with finite precision analysis of the nontrivial multiplier and concludes with an average noise propagation model for the rotator that takes into account both trivial and nontrivial rotations.

3.2.1 Nontrivial phase rotation

A model for the complex phase rotator is shown in Figure 5. Only the datapath for the real (inphase) output is shown in the figure; a similar datapath also exists for the imaginary (quadrature) output.

The B_x-bit input to the phase rotator is the sum of a desired complex input $(x_i + jx_q)$ and a complex input noise $(n_{xi} + jn_{xq.})$. The noise source includes the accumulated rounding noise from all previous stages. As in the case of the crossadder, all inputs are modeled as zero-mean, uncorrelated random variables. The inputs $\{x_i, x_q\}$ are modeled with variance S_{k-1}, and $\{n_{xi}, n_{xq}\}$ with variance N_{k-1}.

The input is phase rotated by complex multiplication with a desired phasor, $w = e^{j\phi}$, whose real and imaginary components are rounded to B_w-bits. Rounding of the phasor introduces a second complex noise source $(n_{wi} + jn_{wq})$ with the components $\{n_{wi}, n_{wq}\}$ having variance σ_{nw}^2. Without loss of generality, we assign this phasor unit magnitude and represent each component of w with one sign bit and (B_w-1) fractional bits of precision.

This model assumes the multiplier and adder operate with full internal precision, so the precision is only limited at the output of the complex rotator. Because the inputs are signed, the precision required at the output of each multiplier is $(B_x + B_w - 1)$-bits.

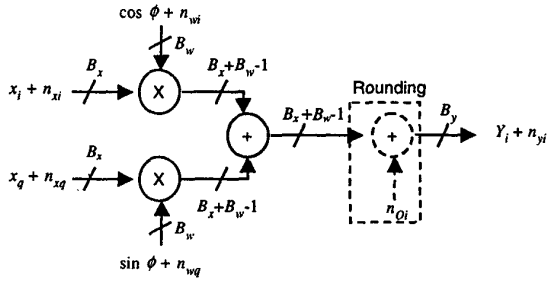

Figure 5. Nontrivial rotator datapath

In Figure 5, the output of the adder is also $(B_x + B_w - 1)$-bits. Overflow protection is not required on this adder if the magnitudes of both the rotator input and the rounded phasor satisfy the following constraints:

$$|(x_i + n_{xi}) + j(x_q + n_{xq})| < 2^{Bx-1} \quad (7)$$

$$|(w_i + n_{wi}) + j(w_q + n_{wq})| \leq 1 \quad (8)$$

These constraints ensure that rotation of the input signal cannot overflow the real or imaginary output of the complex multiplier. If the inputs are unconstrained, adding an extra bit to the output will protect against overflow. Since protecting against overflow will produce an output that meets constraint (7), overflow protection can be limited to a single rotator in the FFT processing chain.

In order to reduce the precision required by the next stage, the L least significant bits are rounded at the output of the adder. Rotation and rounding causes scaling of the input magnitude by

$$A_r = 2^{Bw-L-1} \quad (9)$$

and adds a zero mean roundoff noise n_{Oi}.

The output of the phase rotator is therefore the sum of two terms: the desired full-precision output

$$y_i = A_r(x_i \cos\phi + x_q \sin\phi) \quad (10)$$

and a propagated output noise

$$n_{yi} = A_r[(n_{xi}\cos\phi + n_{xq}\sin\phi) + (x_i + n_{xi})n_{wi} + (x_q + n_{xq})n_{wq}] + n_{Oi} \quad (11)$$

Computing the output variance for y_i yields $S_{k,\ nontriv} = A_r^2 S_{k-1}$. The output noise variance for the nontrivial rotation is given by

$$N_{k,\ nontriv} = A_r^2[N_{k-1} + 2(S_{k-1} + N_{k-1})\sigma_{nw}^2] + \sigma_{nO}^2 \quad (12)$$

Using a uniform random noise model for both the phasor and output roundoff noise, we can conclude that

$$N_{k,nontriv} = A_r^2\left[N_{k-1} + \frac{1}{6}(S_{k-1} + N_{k-1})2^{-2(Bw-1)} + \frac{1}{12}\right]. \quad (13)$$

3.2.2 Average noise propagation parameters

Our discussion in the previous section considered only *nontrivial* multiplication. In the following we also consider the presence of trivial rotations in order to characterize the average noise added by the rotator. If $A_r \geq 1$, trivial rotation does not introduce additive noise and can therefore be modeled by an ideal amplifier with signal and noise outputs $S_{k,\ trivial} = A_r^2 S_{k-1}$ and $N_{k,\ trivial} = A_r^2 N_{k-1}$.

The average noise propagation model is derived by weighting the models for the nontrivial and trivial rotators. If M is the number of nontrivial rotations that appear in one column of processing for an N-point FFT, then $\rho = M/N < 1/2$ is the *nontrivial multiplier ratio* used to derive the average noise propagation model.

The weighted output variances are given by

$$S_k = \rho_k S_{k,\ nontriv} + (1-\rho_k)S_{k,\ trivial} = A_r^2 S_{k-1} \quad (14)$$

$$N_k = \rho_k N_{k,\ nontriv} + (1-\rho_k)N_{k,\ trivial} = A_r^2 N_{k-1} + \rho_k A_r^2\left[\frac{1}{6}(S_{k-1} + N_{k-1})2^{-2(Bw-1)} + \frac{1}{12}\right] \quad (15)$$

and the gain and additive noise parameters for the average rotator stage are therefore

$$G_k = A_r^2 \quad (16)$$

$$\sigma_k^2 = \rho_k A_r^2\left[\frac{1}{6}(S_{k-1} + N_{k-1})2^{-2(Bw-1)} + \frac{1}{12}\right] \quad (17)$$

4: Results

In this section we use the noise propagation model to compare predicted results for different FFT architectures to finite-wordlength FFT simulations. All architectures are designed with 10-bit inputs and twiddle factors with precision $B_w = 12$ bits. To satisfy the condition in (7) at the input to the first rotator, the real and imaginary inputs to the FFT are selected from a uniform random distribution over the interval $[-2^{8.5}, 2^{8.5}]$.

4.1: Full crossadder precision

We first consider N-point FFT decimation-in-time (DIT) and decimation-in-frequency (DIF) architectures where the precision grows by one bit at each crossadder stage, and precision is maintained at each rotator stage. The output precision therefore grows to $(N+10)$ bits.

The noise propagation parameters for the crossadders are given by (3) and (4). The parameters for the rotator stages are given by (16) and (17) with $A_r = 1$. For the given design choices, each crossadder is modeled by an ideal amplifier with gain $G_k = 2$, and each rotator by a simple additive noise source. The DIF flowgraph is essentially a mirror image of the DIT flowgraph, with the ordering of the crossadder and rotator stages reversed. This leads to equivalent noise propagation models of alternating amplifier and noise stages shown in Figure 6.

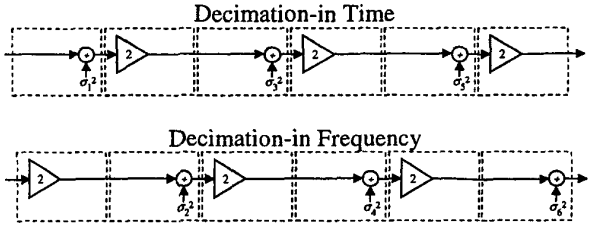

Figure 6. FFT equivalent noise models

For DIT, it can be shown that the nontrivial multiplier ratio for each rotator stage is given by

$$\rho_{2n-1} = \begin{cases} 0 & n < 3 \\ 0.5 - 0.5^{n-1} & 3 \le n \le \log_2 N \end{cases} \quad (18)$$

and for DIF is given by

$$\rho_{2n} = \begin{cases} 0.5 - 0.5^{\log_2 N - n} & 1 \le n \le \log_2 N - 2 \\ 0 & \log_2 N - 2 < n \end{cases} \quad (19)$$

Examination of these equations shows that ρ_k is an increasing function for DIT and decreasing for DIF.

From Figure 6, it is evident that noise sources at the beginning of the chain are amplified more than noise sources at the end of the chain. DIT has a performance advantage over DIF because of the distribution of nontrivial rotations. Another advantage of DIT is that the rotators are one bit smaller than DIF because of the growth through the first crossadder stage in DIF.

Predictions obtained with the noise propagation model are compared to simulated results in Figure 7.

4.2: Reduced crossadder precision

In this section we examine the use of the noise propagation model for making design choices. We begin with a 1024-point FFT and consider the effect of reducing the crossadder precision in five of the ten stages.

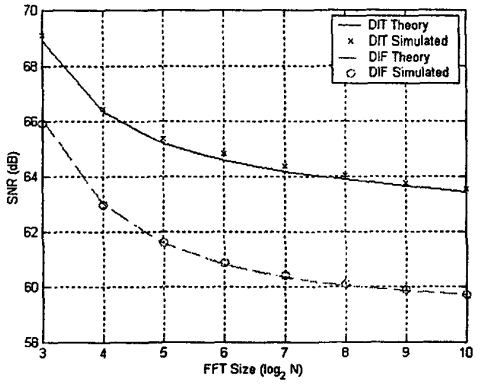

Figure 7. DIT vs. DIF FFT SNR performance

In the first case, we allow only the first five crossadders to grow, and limit precision in the last five by rounding the LSB in each stage. In the second case, we allow only the last five crossadders to grow, and limit precision in the first five. The results are summarized below, where the predicted results compare favorably with simulations:

		SNR (dB) Theory	SNR (dB) Simulated
DIT	Grow first five	51.64	51.62
	Grow last five	35.93	35.91
DIF	Grow first five	51.64	51.64
	Grow last five	35.93	35.96

In summary, a model has been described for quickly and accurately estimating finite wordlength effects in FFT architectures. From this model, we can make the observation that noise sources in the early stages of FFT processing get amplified and therefore have greater negative impact on performance than noise sources in the later stages. This leads to two conclusions: First, DIT results in better SNR than the DIF because the DIT algorithm has fewer non-trivial twiddle factors in the early stages than DIF. Second, rounding the crossadder output LSB in the later stages is better than rounding it in the early stages because the resulting roundoff noise gets amplified less when it occurs in the later stages. These conclusions lead to the choice of a DIT architecture with the crossadder output LSB rounded in the later stages to achieve good noise performance. From the table above we can observe that DIT and DIF perform nearly identically when the crossadder noise dominates over differences in the nontrivial rotator distributions. However, as described earlier, DIT still offers the advantage of reduced rotator wordlengths.

5: References

[1] A.V. Oppenheim and R.W. Schafer, Discrete-Time Signal Processing, Prentice Hall, 1989.

[2] E.H. Wold and A.M. Despain, "Pipeline and Parallel-Pipeline FFT Processors for VLSI Implementations," *IEEE Trans Computers*, C-33, May 1984.