Integer Fast Fourier Transform

Soontorn Oraintara, *Member, IEEE*, Ying-Jui Chen, *Student Member, IEEE*, and Truong Q. Nguyen, *Senior Member, IEEE*

***Abstract*—In this paper, a concept of integer fast Fourier transform (`IntFFT`) for approximating the discrete Fourier transform is introduced. Unlike the fixed-point fast Fourier transform (`FxpFFT`), the new transform has the properties that it is an integer-to-integer mapping, is power adaptable and is reversible. The lifting scheme is used to approximate complex multiplications appearing in the FFT lattice structures where the dynamic range of the lifting coefficients can be controlled by proper choices of lifting factorizations. Split-radix FFT is used to illustrate the approach for the case of 2^N-point FFT, in which case, an upper bound of the minimal dynamic range of the internal nodes, which is required by the reversibility of the transform, is presented and confirmed by a simulation. The transform can be implemented by using only bit shifts and additions but no multiplication. A method for minimizing the number of additions required is presented. While preserving the reversibility, the `IntFFT` is shown experimentally to yield the same accuracy as the `FxpFFT` when their coefficients are quantized to a certain number of bits. Complexity of the `IntFFT` is shown to be much lower than that of the `FxpFFT` in terms of the numbers of additions and shifts. Finally, they are applied to noise reduction applications, where the `IntFFT` provides significantly improvement over the `FxpFFT` at low power and maintains similar results at high power.**

***Index Terms*—Discrete Fourier transform, fixed-point FFT, integer transforms, lifting scheme.**

I. Introduction

THE DISCRETE Fourier transform (DFT) is one of the most fundamental operations in digital signal processing. Because of the efficiency of the convolutional property, the DFT is often used in linear filtering found in many applications such as quantum mechanics [1], noise reduction [2], and image reconstruction [3]. However, the computational requirements for completing the DFT of a finite length signal are relatively intensive. In particular, if the input signal has length N, directly calculating its DFT requires N^2 complex multiplications ($4N^2$ real multiplications) and some additional additions. In 1965, Cooley and Tukey introduced the fast Fourier transform (FFT), which efficiently and significantly reduces the computational cost of calculating N-point DFT from $O(N^2)$ to $O(N \log_2 N)$ [4]. Since then, there have been numerous further developments that extended Cooley and Tukey's original contribution. Many efficient structures for computing DFT have been discovered by taking advantage of the symmetry and periodicity properties of the roots of unity $e^{j2\pi k/N}$ such as the radix-2 FFT, radix-4 FFT, and split-radix FFT [6]. In this paper, since we mainly focus on the fast structures of the DFT, the terms DFT and FFT will be used interchangeably.

The order of the multiplicative complexity is commonly used to measure and compare the efficiency of the algorithms since multiplications are intrinsically more complicated among all operations [5]. It is well-known in the field of VLSI that among the digital arithmetic operations (addition, multiplication, shifting and addressing, etc.), multiplication is the operation that consumes most of the time and power required for the entire computation and, therefore, causes the resulting devices to be large and expensive. Therefore, reducing the number of multiplications in digital chip design is usually a desirable task. In this paper, utilizing the existing efficient structures, a novel structure for approximating the DFT is presented. This proposed structure is shown to be a reversible integer-to-integer mapping called Integer FFT (`IntFFT`). All coefficients can be represented by finite-length binary numbers. The complexity of the proposed `IntFFT` will be compared with the conventional fixed-point implementation of the FFT (`FxpFFT`). Moreover, the performances of the new transforms are also tested in noise reduction problem.

The invertibility of the DFT is guaranteed by orthogonality. The inverse (the IDFT) is just the conjugate transpose. In practice, fixed-point arithmetic is often used to implement the DFT in hardware since it is impossible to retain infinite resolution of the coefficients and operations [6], [7]. The complex coefficients of the transform are normally quantized to a certain number of bits depending on the tradeoff between the cost (or power) and the accuracy of the transform. However, direct quantization of the coefficients used in the conventional structures, including both direct and reduced-complexity (e.g., radix-2, radix-4, etc.) methods, destroys the invertibility of the transform. The novel approach presented in this paper guarantees the invertibility property of the transform while keeping the coefficients of the forward and inverse transforms to be finite-length binary numbers.

A. Previous Works

Recently, there has been a reasonable amount of attention in trying to approximate the existing floating-point orthogonal transforms such as the DCT [8]–[10] or DFT [11] with invertibility property preserved. In [8], the eight-point DCT used in the image and video coding standards is approximated by a

Manuscript received October 17, 2000; revised October 29, 2001. The associate editor coordinating the review of this paper and approving it for publication was Dr. Athina Petropulu.

S. Oraintara is with the Department of Electrical Engineering, University of Texas at Arlington, Arlington, TX 76019-0016 USA (e-mail: oraintar@uta.edu).

Y.-J. Chen is with the Civil and Environmental Department, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: yrchen@mit.edu).

T. Q. Nguyen is with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093 USA (e-mail: nguyent@ece.ucsd.edu).

Publisher Item Identifier S 1053-587X(02)01345-4.

brute-force technique, i.e., the transform coefficients are optimized over the set of real integers[1] by having the orthogonality property (a system of nonlinear equations) as a constraint. The same approach is extended to the case of the eight -point DFT whose coefficients are selected from the set of complex integers [11]. Although this approach is simple and straight forward, it is very difficult to extend the approach to the case of large N—imagine solving a big system of nonlinear equations. In addition, once a set of coefficients is obtained, it is not trivial as to how to adjust them for different transform accuracy unless one reoptimizes the coefficients.

In [9] and [10], lifting factorization is proposed to replace the 2×2 orthogonal matrices appearing in the eight-point DCT using, respectively, the fast structure and the Hadamard structure [12]. The resulting transforms are shown to be simple and invertible, even though the lifting coefficients are quantized and are power-adaptable, i.e., different quantization step sizes can be used to quantize the lifting coefficients without destroying the invertibility. In this paper, lifting factorization is proposed to be used in the fast structures of the DFT where complex multiplications are expressed in terms of liftings. The approach can be used in many existing structures such as radix-2, radix-4, and split-radix with arbitrary sizes. However, the split-radix structure will be used to illustrate the proposed method, which can also be extended to other existing FFT structures as well.

B. Outline of the Paper

The next section reviews necessary backgrounds used in the development of the paper including the discrete Fourier transform and split-radix FFT, its fixed-point implementation, and lifting scheme. Section III presents a method for converting a complex multiplier into lifting structure. The dynamic range at internal nodes of the FFT lattice structure is discussed in Section IV. Section V discusses methods for optimizing the complexity and limiting the dynamic range of the coefficients. The accuracy and complexity of the proposed approach are presented in Section VI. Section VII presents the use of the `IntFFT` in noise reduction application and compares its performance with the `FxpFFT`, and Section VIII concludes the paper.

II. Backgrounds

A. Discrete Fourier Transform and Its Fast Structure

The DFT of an N-point discrete-time signal $x(n)$ is defined by

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn},\qquad \text{for } k=0,\,1,\,\ldots,\,N-1 \quad (1)$$

where $W_N=e^{-j2\pi/N}$ or, in vector-matrix form

$$\mathbf{X}=\mathbf{F}\mathbf{x} \quad (2)$$

where $\mathbf{x}=[x(0)\;\;x(1)\;\;\cdots\;\;x(N-1)]^T$, $\mathbf{X}=[X(0)\;\;X(1)\;\;\cdots\;\;X(N-1)]^T$, and $\mathbf{F}$ is an $N\times N$ matrix with elements $[\mathbf{F}]_{i,\ell}=W_N^{i\ell}$. Similarly, the IDFT can be given by

$$x(n)=\frac{1}{N}\sum_{k=0}^{N-1}X(k)W_N^{-kn},\qquad \text{for } n=0,\,1,\,\ldots,\,N-1 \quad (3)$$

which follows from the fact that $\mathbf{F}$ is an orthogonal matrix, i.e., $\mathbf{F}^{\dagger}\mathbf{F}=N\mathbf{I}$, where † denotes complex conjugate transpose.

Discarding the factor $1/N$ in (3), it is clear that in order to calculate one coefficient of the DFT or IDFT, it requires N complex multiplications and $N-1$ complex additions. Therefore, the total number of complex multiplications for computing an N-point DFT is N^2. However, this direct computation is inefficient and can be significantly simplified by taking the advantage of the symmetry and periodicity properties of the twiddle factor W_N. These properties are

- Symmetry property: $W_N^{k+N/2}=-W_N^k$;
- Periodicity property: $W_N^{k+N}=W_N^k$.

There are many existing fast structures to compute the DFT depending on the length of the input. In this paper, the split-radix structure that is suitable for input with length of $N=2^K$ will be used to illustrate the proposed approach of `IntFFT`. The approach can also be applied to other structures such as radix-2 and radix-4 as well. We now briefly describe the split-radix algorithm and its network structure.

1) Split-Radix FFT and Its Lattice Structure: Let us assume that $N=2^K$ with $K\geq 2$. From (1), we get

$$X(2k)=\sum_{n=0}^{N/2-1}\left[x(n)+x\left(n+\frac{N}{2}\right)\right]W_{N/2}^{kn}\qquad k=0,\,1,\,\ldots,\,\frac{N}{2}-1 \quad (4)$$

and

$$X(4k+1)=\sum_{n=0}^{N/4-1}\left[x(n)-jx\left(n+\frac{N}{4}\right)-x\left(n+\frac{N}{2}\right)+jx\left(n+\frac{3N}{4}\right)\right]W_N^nW_{N/4}^{kn} \quad (5)$$

$$X(4k+3)=\sum_{n=0}^{N/4-1}\left[x(n)+jx\left(n+\frac{N}{4}\right)-x\left(n+\frac{N}{2}\right)-jx\left(n+\frac{3N}{4}\right)\right]W_N^{3n}W_{N/4}^{kn} \quad (6)$$

for $k=0,\,1,\,\ldots,\,N/4-1$. It follows from (4) that the even-indexed terms $X(2k)$ can be obtained by the $N/2$-point DFT of the new signal

$$A_1^1(n)=x(n)+x(n+N/2),\quad \text{for } n=0,\,1,\,\ldots,\,N/2.$$

Similarly, the first- and third-indexed terms can be expressed as the $N/4$-point DFT of two signals

$$A_1^2(n)=W_N^n[A_2^1(n)-jA_2^1(n+N/4)]$$
$$A_2^2(n)=W_N^{3n}[A_2^1(n)+jA_2^1(n+N/4)]$$

for $n=0,\,1,\,\ldots,\,N/4$, where $A_2^1(n)=x(n)-x(n+N/2)$. It is clear from (4)–(6) that the N-point DFT coefficients of

[1]In this paper, we refer a real integer to a quantity whose real part has integer value and whose imaginary part is zero. Similarly, a complex integer is defined as a quantity whose real and imaginary parts have integer values.

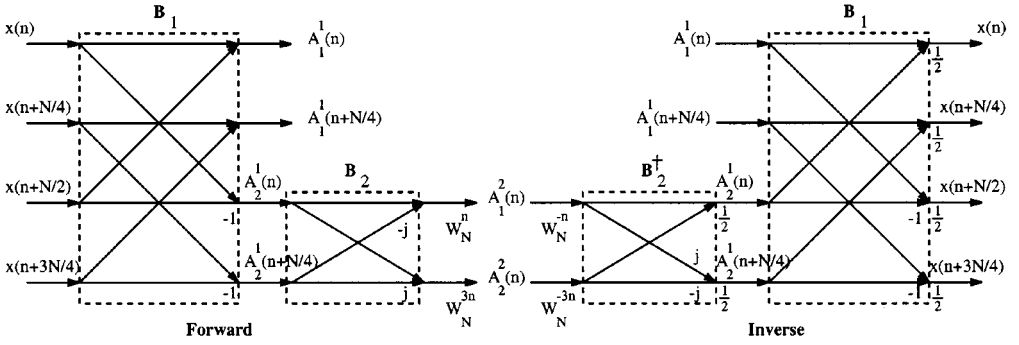

Fig. 1. Basic butterfly structure for N-point FFT using split-radix structure.

$x(n)$ can be achieved by computing one $N/2$-point DFT and two $N/4$-point DFTs for the new signals $A_q^p(n)$. The same procedure can be iteratively applied to these smaller size DFTs until the size is reduced to one. Fig. 1 shows the forward and inverse lattice structures for calculating $A_q^p(n)$ from $x(n)$ and vice versa. It is clear that the signals A_q^1 can be obtained from $x(n)$ passing through the orthogonal butterfly lattices

$$\mathbf{B}_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

as indicated in Fig. 1. Similarly, the signals $A_q^2(n)$ can be obtained from $A_2^1(n)$ passing through another orthogonal butterfly lattices

$$\mathbf{B}_2 = \begin{bmatrix} 1 & -j \\ 1 & j \end{bmatrix}$$

and the twiddle factor W_N^n. It should be noted that the multipliers -1, j, and $-j$ in $\mathbf{B}_2$ will not be counted for the number of complex multiplications. Since the matrices $\mathbf{B}_i$ and their complex conjugate transposes map integers to integers, to obtain the `IntFFT`, we only need to focus on the complex multipliers W_N^n, and their complex conjugates as will be discussed in details in the next section.

B. Fixed-Point Arithmetic Implementation

One of the factors that primarily affects the cost of the DSP implementation is the resolution of the internal nodes (the size of the registers at each stage). In practice, it is impossible to retain infinite resolution of the signal samples and the transform coefficients. Since the floating-point operation is very expensive, these numbers are often quantized to a fixed number of bits. Different formats of fixed-point representation of numbers can be found in [7]. In this paper, the variables N_c and N_i are used to denote the numbers of bits required to store the coefficients in the DFT calculation and the input signal, respectively.

Each addition can increase the number of bits by one, whereas each multiplication can increase up to $N_c - 1$, where N_c is the number of bits of the multiplier. The nodes in latter stages require more bits than those in earlier stages to store the output after each arithmetic operation without overflows. As a result, the number of bits required to store the results grows accumulatively as N or the number of stages increases. In general, the number of bits at each internal node is fixed to a certain number of bits. Call this number N_n. The most significant bits (MSBs) of the result after each operation will be kept up to N_n bits, and the tail will be truncated. However, this conventional fixed-point arithmetic destroys the invertibility of the transform because the DFT coefficients are quantized. This paper presents a novel way to quantize the DFT coefficients that preserves the invertibility property.

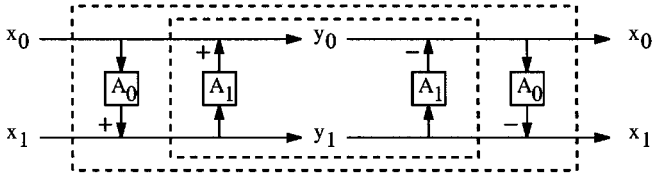

Fig. 2. Lifting scheme that allows perfect reconstruction.

C. Lifting Scheme

The lifting scheme has been proposed to be a tool in constructing wavelets and perfect reconstruction (PR) filterbanks [13], [14]. Biorthogonal filterbanks having integer coefficients can be easily implemented and can be used as integer-to-integer transform. Lossless image coding is an example of applications that uses such a transform.

Consider the following two-channel system in Fig. 2. The two inputs x_0 and x_1 are fed into a two-port network that contains operators A_0 and A_1. The first branch, which is operated by A_0, is called *dual-lifting*, whereas the second one, which is operated by A_1, is called *lifting*, as indicated in the figure. One can immediately see that the system is PR for any choices of A_0 and A_1. Since the operators A_0 and A_1 can be nonlinear, rounding or flooring operations, etc., can be used without destroying the PR property.

III. Converting Complex Numbers to Real Lifting Steps

Recall that all the coefficients appearing in the FFT and IFFT structures are complex numbers with magnitude one, i.e., every coefficient can be expressed as $e^{j\theta}$, where θ is some real number. Since these coefficients are scalar with magnitude one, the inverses are simply their complex conjugates. However, if a coefficient is quantized, the inverse of the new coefficient is no longer guaranteed to be its complex conjugate. Specifically, let a be a complex number with magnitude one, i.e.,

$$a = c + js$$

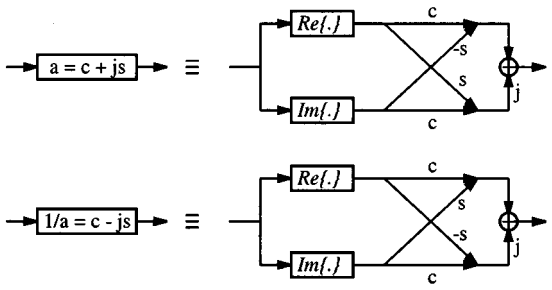

Fig. 3. Butterfly structure for implementing a complex multiplication.

where c and s are real numbers, and $c^2 + s^2 = 1$. Let a^q be the quantized version of a, i.e.,

$$a^q = c^q + js^q$$

where c^q and s^q are finite-word length approximations of c and s, respectively. Hence, the reciprocal of a^q is

$$\frac{1}{a^q} = \frac{c^q}{|a^q|^2} - j\frac{s^q}{|a^q|^2}.$$

In general, $|a^q|$ is not one, although $|a| = 1$. Instead, $1/a^q$ may not even be a finite word-length complex number, even though a^q is. This is the reason why the conventional fixed-point arithmetic does not preserve the PR property.

The PR property can be preserved via the lifting scheme. Each complex multiplication is equivalent to four real multiplications. Specifically, let $x = x_r + jx_i$ be a complex number, and hence, $y = ax = (cx_r - sx_i) + j(cx_i + sx_r)$, or in the vector-matrix form

$$y = [1 \quad j]\begin{bmatrix} c & -s \\ s & c \end{bmatrix}\begin{bmatrix} x_r \\ x_i \end{bmatrix} = [1 \quad j]\mathbf{R}\begin{bmatrix} x_r \\ x_i \end{bmatrix} \tag{7}$$

where

$$\mathbf{R} = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}.$$

Fig. 3 shows the butterfly structure of a single complex multiplication. Notice that a has magnitude one if and only if $\mathbf{R}$ is an orthonormal matrix. Assuming that $c^2 + s^2 = 1$ with $s \neq 0$ (if $s = 0$, then $a = 1$ or -1, which does not need to be quantized), it is easy to see that $\mathbf{R}$ can be decomposed into three lifting steps [13].

$$\mathbf{R} = \begin{bmatrix} c & -s \\ s & c \end{bmatrix} = \begin{bmatrix} 1 & \frac{c-1}{s} \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ s & 1 \end{bmatrix}\begin{bmatrix} 1 & \frac{c-1}{s} \\ 0 & 1 \end{bmatrix}. \tag{8}$$

Fig. 4 illustrates the conversion from one complex multiplication to three-step lifting scheme. In order to distinguish the coefficients in the lifting structure from the original coefficients c and s, we now call the new coefficients $(c-1)/s$ and s *lifting coefficients* and refer the original coefficients c and s as *butterfly coefficients.*

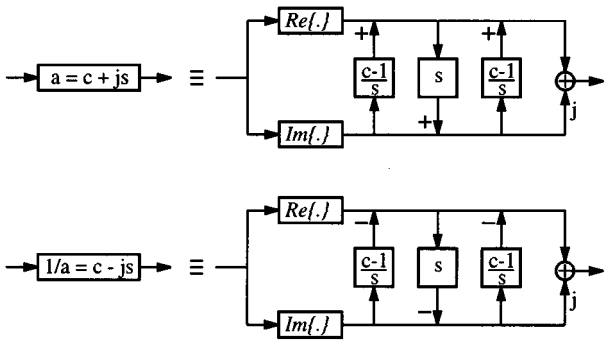

Fig. 4. Lifting structure for implementing a complex multiplication and its inverse.

The advantages of this factorization are twofold. First, the number of real multiplications is reduced from four to three, although the number of real additions is increased from two to three. Second, it allows for quantization of the lifting coefficients and the quantization of the result of each multiplication without destroying the PR property. To be specific, instead of quantizing a directly, the lifting coefficients s and $(c-1)/s$ are quantized, and therefore, its inverse $1/a$ also consists of three lifting steps with the same lifting coefficients but with opposite signs. Further than that, nonlinear operators can also be applied to the product at each lifting step. Fig. 5 shows four equivalent lifting structures for approximating $\mathbf{R}$, where Q indicates a nonlinear operator such as rounding or flooring operation. Each structure has its own advantage for optimizing the computational complexity of the transform, as will be discussed in the next section.

A. *Eight-Point Split-Radix* `IntFFT`

This subsection demonstrates how to construct an eight-point `IntFFT` based on the split-radix structure. Fig. 6 shows the lattice structure of the eight-point `IntFFT`, where the twiddle factors W_8^1 and W_8^3 are implemented using the proposed conversion. Note that instead of using the one in Fig. 5(a), the lower lifting factorization is obtained from the structure in Fig. 5(b), of which the lifting coefficients are between -1 and 1. This detail will be discussed in Section V. The rounded boxes at the lifting coefficients represent quantization of both the lifting coefficients and the results after multiplications.

It is clear from Fig. 6 that there are 24 butterflies with coefficients 1, -1, j, and $-j$ that do not require any multiplication or addition. These 24 butterflies can be implemented using $(24)(2) = 48$ complex adders or 96 real adders. The rest of the computation is based on the binary representation of the lifting coefficients $\pm 1/\sqrt{2}$ and $\pm(\sqrt{2} - 1)$.

IV. Resolution of the Internal Nodes

Recall that in the implementation of FFT and IFFT, all the coefficients are of the form $W_N^k = e^{j\theta}$, where $\theta = 2\pi k/N$ for some integer k. These twiddles can be replaced by lifting scheme as

$$e^{j\theta} := [1 \quad j]\mathbf{R}_\theta \begin{bmatrix} Re\{\cdot\} \\ Im\{\cdot\} \end{bmatrix} \tag{9}$$

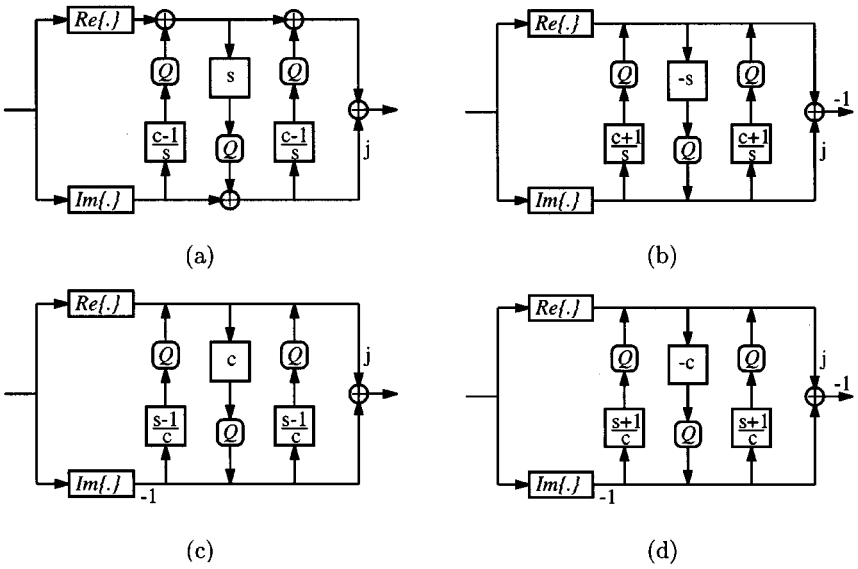

Fig. 5. Four equivalent lifting structures with quantization for implementing a complex multiplication.

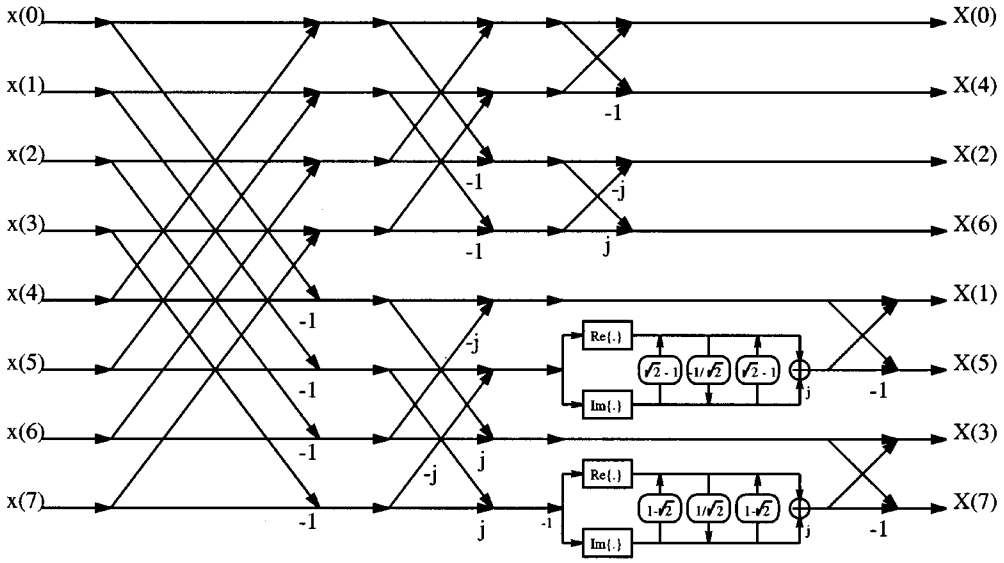

Fig. 6. Lattice structure of eight-point `IntFFT` using split-radix structure.

where

$$\mathbf{R}_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} 1 & \dfrac{c-1}{s} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ s & 1 \end{bmatrix} \begin{bmatrix} 1 & \dfrac{c-1}{s} \\ 0 & 1 \end{bmatrix} \tag{10}$$

$c = \cos\theta$, and $s = \sin\theta$. Define $\mathbf{R}_\theta^Q$ to be a nonlinear approximation of $\mathbf{R}_\theta$, where the product at each lifting in $\mathbf{R}_\theta$ is quantized by the nonlinear operator Q, as depicted in Fig. 5(a). Let N_ℓ denote the number of bits used by each quantizer Q. According to Fig. 1, it is obvious that the FFT and IFFT can be approximated by lattice structures with finite word-length (integer) coefficients. The butterfly matrices $\mathbf{B}_1$ and $\mathbf{B}_2$ do not need to be changed since their coefficients are integers, and they map integer inputs to integer outputs (with possibly an increase in the number of bits). What remains to be done is the conversion of each twiddle multiplier W_N^k into lifting steps, as discussed in the previous section. Since the lifting structure is reversible regardless of the presence of quantization, the overall transform will also be reversible. Let us call the new transform and its inverse transform `IntFFT` and `IntIFFT`, respectively. Because the `IntFFT` and `IntIFFT` have integer coefficients, multiplierless `IntFFT` and `IntIFFT` can be accomplished by replacing each real (integer) multiplication by a number of additions. The number of additions for each multiplication depends on the number of 1s appearing in the binary representation of the multiplier.

Similar to the case of `FxpFFT` and `FxpIFFT`, let us define N_c and N_n, respectively, to be the number of bits used to quantize each lifting coefficient and the size of the registers used to store the value at each node, i.e., input and output nodes of each of the butterflies $\mathbf{B}_1$ and $\mathbf{B}_2$ and the result after each lifting step.

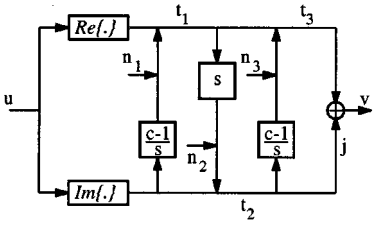

Fig. 7. Statistical model of lifting scheme used in a complex multiplier.

Although the IntFFT produces a reversible integer-to-integer mapping, the number of bits used to represent the outputs must be greater than that used for the inputs. Indeed, according to Fig. 1, this number grows internally from the left to the right of the forward transform. The possible maximum value of this number defines the minimum value of N_n, $N_n^{\min}$, which allows the transform to be reversible. Each butterfly matrix $\mathbf{B}_1$ or $\mathbf{B}_2$ can increase the resolution of its input by up to one bit. Likewise, each lifting step used in the approximation of the twiddle multipliers can also increase the resolution of the signals. In this section, we will estimate an upper bound of $N_n^{\min}$ for the case of $N = 2^K$ using the split-radix structure. In our construction of IntFFT, each twiddle factor is approximated by two liftings and one dual lifting (or two dual liftings and one lifting). In order to simplify the analysis, let us model each twiddle factor as in Fig. 7, where n_i are random noise resulting from the quantization of the lifting coefficient and the quantization of the product after multiplication by the lifting coefficient. Let us assume that

1) all lifting coefficients are between -1 and 1;
2) $|n_i| \le [(2-\sqrt{2})/3]\max(|Re\{u\}|, |Im\{u\}|)$.

The first assumption can be reached by selecting a proper alternative lifting factorization shown in Fig. 5. In the next section, we will show that at least two of the four equivalent structures have lifting coefficients between -1 and 1. The second assumption is equivalent to that N_c and N_ℓ are sufficiently large.

Theorem IV.1: The lifting implementation of each twiddle factor increases the resolution of its input by, at most, one bit.

Proof: See the Appendix.

It should also be noted that the original butterfly structure guarantees that the dynamic range of the output will be less than twice that of $\max(|Re\{u\}|, |Im\{u\}|)$ since

$$\begin{aligned}|cRe\{u\} + sIm\{u\}| &\le |Re\{u\}| + |Im\{u\}| \\ &\le 2\max(|Re\{u\}|, |Im\{u\}|) \\ |cIm\{u\} - sRe\{u\}| &\le |Re\{u\}| + |Im\{u\}| \\ &\le 2\max(|Re\{u\}|, |Im\{u\}|).\end{aligned}$$

Now, we are ready to estimate an upper bound of $N_n^{\min}$. For convenience, let $f(K)$ be $N_n^{\min}$ for the case of $N = 2^K$. It is clear that $f(0) = N_i$, where N_i is the resolution of the input. Since a two-point FFT can be constructed by a butterfly matrix $\mathbf{B}_1$, $f(1) = N_i + 1$. When $K = 2$, the two twiddle multipliers are 1 and j, which do not increase the number of bits. Thus, $f(2) = N_i + 2$. In general, the twiddle factors can be complex irrational numbers that also increase the resolution of the internal nodes. When $K > 2$, according to Fig. 1, it is easy to see that

$$f(K) \le \max(f(K-1)+1,\ f(K-2)+3) \qquad (11)$$

which implies the following theorem.

Theorem IV.2: $f(2r) \le N_i + 3r - 1$ and $f(2r+1) \le N_i + 3r + 1$, where $r \ge 1$.

Proof: The proof can be easily done by induction.

V. Optimizing the Low-Power Multiplierless IntFFT

In the previous section, we have discussed how to convert a complex multiplication with the multiplier having magnitude one into lifting scheme and how to estimate an upper bound of $N_n^{\min}$. The resulting conversion allows each complex multiplication to be approximated by finite word-length lifting coefficients. Each of these quantized lifting multipliers can be constructed by a number of additions and shifts depending on the number of 1s in its binary representation. This section discusses several methods for optimizing the lifting coefficients with dynamic range constrained. The goal is to reduce the number of additions by optimally using an alternative lifting factorization.

A. Controlling the Range of Lifting Coefficients

Consider the three lifting coefficients of $\mathbf{R}_\theta$ in (10). Since the value of $\sin\theta$ can be arbitrarily small, the dynamic range of $|(\cos\theta - 1)/\sin\theta|$ can be arbitrarily large. If $\theta \in (-\pi/2, \pi/2)$, then $\cos\theta > 0$, and

$$\begin{aligned}\left|\frac{\cos\theta - 1}{\sin\theta}\right| &= \left|\frac{\sqrt{1-\sin^2\theta} - 1}{\sin\theta}\right| \\ &= |\sin\theta| \cdot \left|\frac{1}{\sqrt{1-\sin^2\theta}+1}\right| < |\sin\theta| < 1.\end{aligned}$$

If $\theta \in (-\pi, -\pi/2) \cup (\pi/2, \pi)$, $|(\cos\theta - 1)/\sin\theta| > 1$, which is not desirable. Hence, the absolute values of the lifting coefficients can be controlled to be less than or equal to one by replacing $\mathbf{R}_\theta$ by $-\mathbf{R}_{\theta+\pi}$ as

$$\begin{aligned}\mathbf{R}_\theta = -\mathbf{R}_{\theta+\pi} &= -\begin{bmatrix}-\cos\theta & \sin\theta \\ -\sin\theta & -\cos\theta\end{bmatrix} \\ &= -\begin{bmatrix}1 & \frac{c+1}{s} \\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 \\ -s & 1\end{bmatrix}\begin{bmatrix}1 & \frac{c+1}{s} \\ 0 & 1\end{bmatrix}. \qquad (12)\end{aligned}$$

When $\theta \in (-\pi, -\pi/2) \cup (\pi/2, \pi)$

$$\begin{aligned}\left|\frac{\cos\theta + 1}{\sin\theta}\right| &= \left|\frac{-\sqrt{1-\sin^2\theta}+1}{\sin\theta}\right| \\ &= |\sin\theta| \cdot \left|\frac{1}{\sqrt{1-\sin^2\theta}+1}\right| < |\sin\theta| < 1\end{aligned}$$

and hence, the new lifting coefficients fall between -1 and $+1$, as desired. Fig. 5(b) shows the equivalent block diagram for (12).

B. Another Choice of Lifting

In the previous subsection, we have presented an alternative solution if the original lifting coefficients have magnitude greater than one. This is because of the fact that multiplying by -1 does not increase the complexity of the algorithms (except one negation is used). It should be noted that to multiply by j or $-j$, one only needs to switch between the real and imaginary

TABLE I
POSSIBLE LIFTING STRUCTURES FOR EACH VALUE OF θ WITH ALL LIFTING COEFFICIENTS BETWEEN -1 AND 1

Range of θ	Possible lifting structures
$(0,\ \pi/2)$	Figures 5(a) and (c)
$(\pi/2,\ \pi)$	Figures 5(b) and (c)
$(-\pi,\ -\pi/2)$	Figures 5(b) and (d)
$(-\pi/2,\ 0)$	Figures 5(a) and (d)

parts of the input with one sign changing, and thus, there is no addition or multiplication required. Utilizing this fact, another choice of lifting factorization can be achieved as

$$\begin{aligned}\mathbf{R}_\theta &= \begin{bmatrix}\sin\phi & -\cos\phi\\ \cos\phi & \sin\phi\end{bmatrix} = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\mathbf{R}_\phi\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\\ &= \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}1 & \dfrac{s-1}{c}\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ c & 1\end{bmatrix}\begin{bmatrix}1 & \dfrac{s-1}{c}\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\end{aligned} \tag{13}$$

where $\phi = (\pi/2) - \theta$. Fig. 5(c) shows an equivalent block diagram described by (13). Similar to the previous case, if $\theta \in (0, \pi)$, $\sin\theta > 0$, and hence, $|(\sin\theta - 1)/\cos\theta| < 1$. However, if $\theta \in (-\pi, 0)$, $\sin\theta < 0$, and the magnitude of the lifting coefficient $(\sin\theta - 1)/\cos\theta$ will be larger than one. Therefore, $\mathbf{R}_\theta$ should be replaced by $-\mathbf{R}_{\theta+\pi}$ as

$$\begin{aligned}\mathbf{R}_\theta &= -\begin{bmatrix}-\sin\phi & \cos\phi\\ -\cos\phi & -\sin\phi\end{bmatrix}\\ &= -\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\mathbf{R}_{\phi+\pi}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}\\ &= -\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}\begin{bmatrix}1 & \dfrac{s+1}{c}\\ 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0\\ -c & 1\end{bmatrix}\begin{bmatrix}1 & \dfrac{s+1}{c}\\ 0 & 1\end{bmatrix}\\ &\quad\cdot\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}.\end{aligned} \tag{14}$$

Fig. 5(d) shows an alternative block diagram described by (14).

To summarize, for each value of θ, there are two out of four possible lifting factorizations (see Fig. 5) where all the lifting coefficients fall within -1 and 1, as summarized in Table I.

C. Fewest Addition Optimization

We have seen different choices in parameterizing the lifting coefficients for the FFT twiddle coefficients. Without any quantization of the lifting coefficients and the incoming signals, each twiddle coefficient, as presented in Fig. 5, requires three real multiplications and three real additions. However, for `IntFFT` and `IntIFFT`, the lifting coefficients are quantized to N_c bits, and the number of additions required for each multiplication can be counted from the number of 1s presented in the binary representation of the quantized multiplier. The structure with the smallest number of 1s appearing in the quantized coefficients should be chosen so that the total number of required additions is minimized.

Taking into account that the complexity of one subtraction is equivalent to that of one addition and should be counted as an addition, the optimal representation is not necessarily represented as a single binary number, but it can be a difference between two binary numbers, as long as the total number of 1s is less. For example, the 4-bit representation of 7 can be given by

$$7 = 0111_2 \quad \text{or} \quad 7 = 8 - 1 = 1000_2 - 0001_2.$$

One can see that in the left binary representation of 7, there are three 1s, and hence, two additions are required to complete the multiplication. On the other hand, there are only two 1s appearing in the right representation, and hence, only one addition (subtraction) is needed. Table II summarizes the conventional and optimal binary representations of the 16-bit lifting coefficients used in the split-radix 16-point FFT, where the twiddle factors are constructed using the lifting structures in Fig. 5(a) or (b). Similarly, Table III summarizes the conventional and optimal binary representations of the 16-bit lifting coefficients used in the split-radix 16-point FFT, where the twiddle factors are constructed using the lifting structures in Fig. 5(c) or (d). In order to minimize the number of additions used to represent each twiddle factor W_N^k, the binary representation with less total number of 1s (and thus the corresponding structure) will be selected between Tables II and III. For example, the numbers of 1s used to represent W_{16}^1 from Tables II and III are $7 \times 2 + 4 = 18$ and $6 \times 2 + 5 = 17$, respectively. Hence, the structure in Fig. 5(c) should be used to implement W_{16}^1 in order to achieve the minimum number of additions.

VI. PERFORMANCES AND COMPLEXITIES

A. Minimal N_n for PR Systems

In order for the `IntFFT` to be invertible, the number of the internal node bits and, thus, the number of bits of the output must be large enough since, after each addition, the dynamic range of the output will be increased by at most one bit. This number directly depends on the size of the FFT and the resolution of the input, as discussed in Section IV. In this section, the following experiment is carried out. Random signals with the resolution (N_i) of 16 bits are used as the input of the N-point `IntFFT` for $N = 4, 8, \ldots, 1024$. Each lifting step contains a 16-bit real multiplier, i.e., $N_c = 16$ and a 16-bit uniform quantizer. For a fixed value of N, the `IntFFT` followed by `IntIFFT` of each random input is computed and compared with the original input. $N_n^{\min}$ is determined by the smallest N_n, which causes the output to be the same as the input, i.e., for each input, $N_n^{\min}$ is estimated. The maximum value of $N_n^{\min}$ for different inputs is used to represent $N_n^{\min}$ of this particular case (N, N_i, N_c and resolution of the quantization Q). It should be noted that the upper bound of $N_n^{\min}$ presented in Theorem IV.2 only depends on N and N_i but does not depend on N_c or the resolution of the quantization Q. Fig. 8 shows the estimated $N_n^{\min}$ obtained from the experiment for different values of N and the upper bound calculated using Theorem IV.2. It is evident that $N_n^{\min}$ is a nondecreasing function of N experimentally and theoretically. The estimated values of $N_n^{\min}$ fall below the upper bound, as expected.

B. Accuracy of the `IntFFT`

In this section, the performances of the `IntFFT` are experimentally evaluated and compared with the conven-

TABLE II
EXAMPLE OF OPTIMAL 16-BIT BINARY REPRESENTATION FOR THE QUANTIZED LIFTING COEFFICIENTS OF THE STRUCTURES IN FIG. 5(a) OR (b) USED IN THE SPLIT-RADIX 16-POINT FFT

Multiplier	Lifting coefficient $\times 2^{15}$	Binary representation	Optimal binary representation
W_{16}^1	-3227	-110010011011_2	-110010011011_2
	6392	1100011111000_2	$1100100000000_2 - 1000_2$
W_{16}^2	-6517	-1100101110101_2	$-1100110000101_2 + 10000_2$
	12539	11000011111011_2	$11000100000000_2 - 101_2$
W_{16}^3	-9940	-10011011010100_2	$-10100000010100_2 + 101000000_2$
	18204	100011100011100_2	$100100000100000_2 - 100000100_2$
W_{16}^4	-13572	-11010100000100_2	-11010100000100_2
	23170	101101010000010_2	101101010000010_2
W_{16}^6	-21894	-101010110000110_2	-101010110000110_2
	30273	111011001000001_2	$1000000001000001_2 - 101000000000_2$

TABLE III
EXAMPLE OF OPTIMAL 16-BIT BINARY REPRESENTATION FOR THE QUANTIZED LIFTING COEFFICIENTS OF THE STRUCTURES IN FIGS. 5(c) OR (d) USED IN THE SPLIT-RADIX 16-POINT FFT

Multiplier	Lifting coefficient $\times 2^{15}$	Binary representation	Optimal binary representation
W_{16}^1	-26892	-110100100001100_2	-110100100001100_2
	32138	111110110001010_2	$1000000000001010_2 - 1010000000_2$
W_{16}^2	-21894	-101010110000110_2	-101010110000110_2
	30273	111011001000001_2	$1000000001000001_2 - 101000000000_2$
W_{16}^3	-17514	-100010001101010_2	-100010001101010_2
	27245	110101001101101_2	$11010101000000_2 - 10100_2$
W_{16}^4	-13572	-11010100000100_2	-11010100000100_2
	23170	101101010000010_2	101101010000010_2
W_{16}^6	-6517	-1100101110101_2	$-1100110000101_2 + 10000_2$
	12539	11000011111011_2	$11000100000000_2 - 101_2$

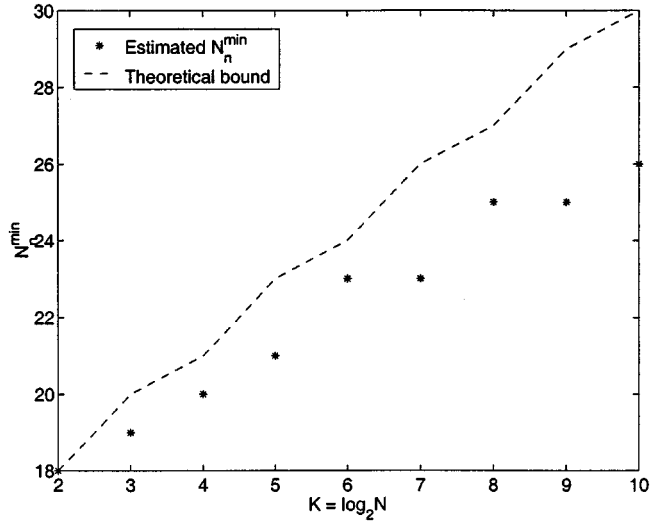

Fig. 8. Number of internal-node bits required for an N-point Split-Radix IntFFT to be invertible when the signals have $N_i = 16$ bits.

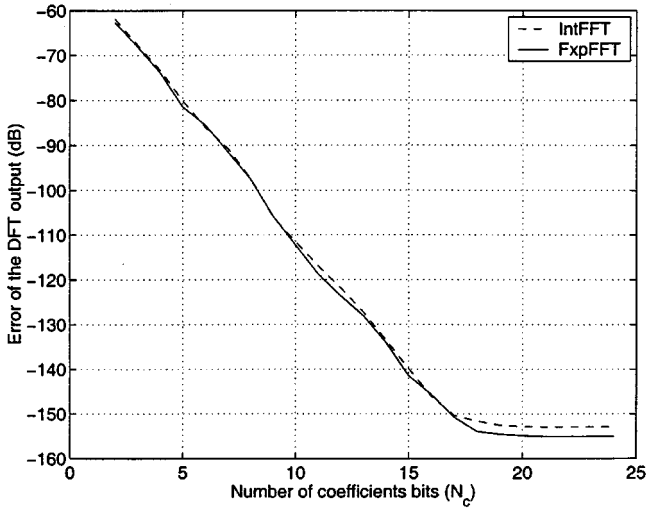

Fig. 9. Comparison of the accuracy of the conventional fixed-point arithmetic FFT and the new IntFFT measured in frequency domain for 256-point transform.

tional FxpFFT. The experiment is performed for the case of $N = 256$. The input signal is quantized to 16 bits, whereas the internal nodes of both structures are set to 27 bits. Obtained from Theorem IV.2, 27 bits will ensure that the IntFFT is reversible and, thus, zero reconstruction error. Fig. 9 compares the errors of the Fourier transform using the FxpFFT and the IntFFT for different values of N_c. In the structure of the FxpFFT, N_c is the number of bits used to quantize the twiddle factors, whereas in case of the IntFFT, N_c is the number of bits used to quantize the lifting coefficients obtained from the structures in Fig. 5. It is evident that the proposed IntFFT yields approximately the same accuracy as that of the conventional FxpFFT when $N_c \leq N_i$. The errors start to converge when $N_c > N_i$, where the error of the IntFFT approximately 3 dB higher. Fig. 10 shows the reconstruction error for the case of FxpFFT and FxpIFFT, whereas the reconstruction error of the proposed IntFFT and IntIFFT remains zero.

Although the Fourier transform is a linear operator because of the nonlinearity in the quantization at each lifting, the resulting IntFFT becomes nonlinear. Therefore, the convolutional rule

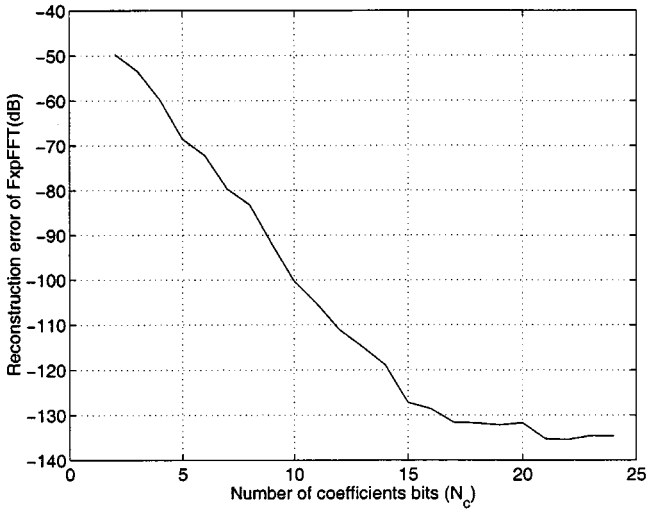

Fig. 10. Reconstruction error of the conventional fixed-point arithmetic FFT for 256-point transform.

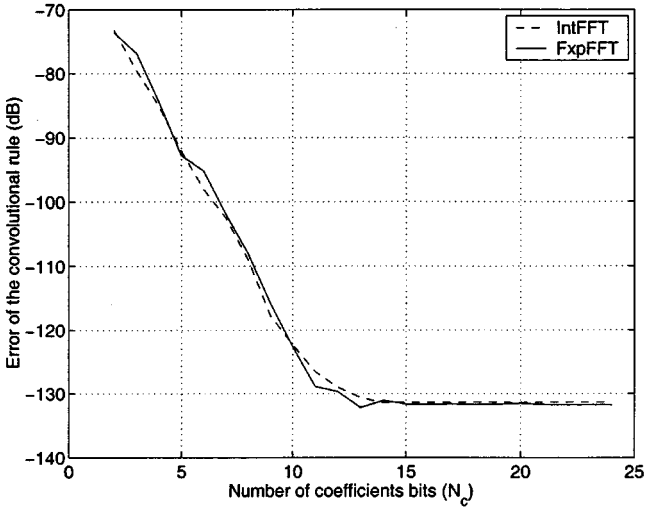

Fig. 11. Comparison of the accuracy of the conventional fixed-point arithmetic FFT and the new IntFFT based on the convolutional rule for 256-point transform.

does not apply. Likewise, in the case of FxpFFT, the butterfly coefficients also have to be quantized, which makes the resulting transform nonlinear as well. In order to simulate the distortion after using the convolutional rule, another experiment is conducted. Using the same parameters as in the previous experiment, two input signals are randomly generated. The Fourier transforms of the two signals are then calculated using the ideal FFT: IntFFT and FxpFFT. For each case, the inverse Fourier transform of the product of the two Fourier transforms is then calculated. The difference between the resulting signal and the one obtained by using the ideal FFT and IFFT defines the error of the convolutional rule. Fig. 11 compares the error of the convolutional rule of the FxpFFT to the IntFFT. Unlike the case of the forward transform, it is evident from Fig. 11 that although two transforms (and their inverses) are nonlinear, the error of the convolutional rule can be decreased by increasing the resolution of the coefficients (N_c). At $N_c = 7$, one can see that the error is less than -100 dB, which is acceptable in general

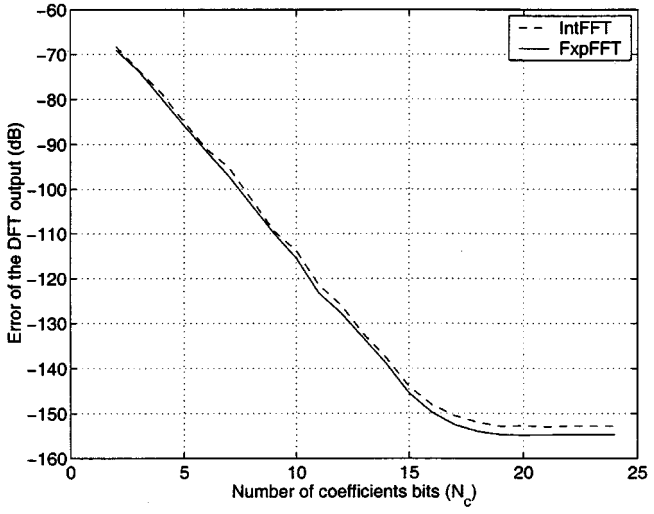

Fig. 12. Comparison of the accuracy of the conventional fixed-point arithmetic FFT and the IntFFT measured in frequency domain for 256-point transform when the signal is complex exponential.

TABLE IV
COMPUTATIONAL COMPLEXITIES OF THE SPLIT-RADIX FFT AND THE INTEGER VERSIONS (FxpFFT AND IntFFT) WHEN THE COEFFICIENTS ARE QUANTIZED TO $N_c = 10$ BITS

	FFT		FxpFFT		IntFFT	
N	Multiplications	Additions	Additions	Shifts	Additions	Shifts
16	20	148	262	144	202	84
32	68	388	746	448	559	261
64	196	964	1910	1184	1420	694
128	516	2308	4674	2968	3448	1742
256	1284	5380	10990	7064	8086	4160
512	3076	12292	25346	16472	18594	9720
1024	7172	27652	57398	37600	41997	22199

since it is below the quantization error (-97 dB for the case of $N_i = 16$ bits). It should also be noted that both structures (butterfly and lifting in each twiddle coefficient) create approximately the same amount of error to the convolutional rule.

In the above experiments, random vectors are used to test the performances of the IntFFT. It is also interesting to compare the results when the signal is a pure sinusoidal. In this experiment, a complex exponential vector with relative frequency 25.3 is used to test the accuracies of the two approximation methods. Fig. 12 shows the resulting errors of the forward FFT. In this case, it is found that the accuracy of the IntFFT is approximately the same as that of the FxpFFT when $N_c < N_i$ and is approximately 3 dB lower when $N_c \geq N_i$. This result is consistent to the case of random noise input.

C. Complexity of the IntFFT

In this section, the method for minimizing the number of adders discussed in Section V is used to estimate the computational complexities of FxpFFT and IntFFT of different sizes. In particular, the numbers of 1s used in the 10-bit binary representations of the butterfly coefficients in FxpFFT and of the lifting coefficients in IntFFT are counted. Table IV summarizes the numbers of real multiplications and real additions needed to perform N-point split-radix FFT [16] and the numbers of real additions and shifts required in N-point FxpFFT and IntFFT. From Table IV, the numbers

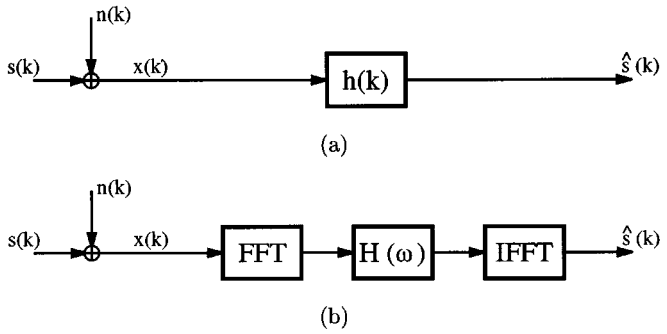

Fig. 13. Noise reduction system. (a) Time-domain processing. (b) Frequency-domain processing.

of additions of FxpFFT and IntFFT are approximately 100% and 50% more than that of the exact FFT; however, no real multiplication is needed. Comparing between FxpFFT and IntFFT, the number of additions of IntFFT is 23–27% less than FxpFFT, whereas the number of shifts is 41–42% less.

VII. Application in Noise Reduction

The performances of the IntFFT and the FxpFFT are tested and compared in noise reduction problem. In Fig. 13(a), the signal of interest $s(k)$ with a known power spectral density (PSD) $\Phi_s(\omega)$ is corrupted by an additive independent Gaussian noise $n(k)$ with a PSD $\Phi_n(\omega)$, resulting in a noisy signal

$$x(k) = s(k) + n(k).$$

The received signal $x(k)$ is then optimally filtered by a filter $h(k)$, resulting in an estimate

$$\hat{s}(k) = x(k) \otimes h(k)$$

where $\otimes$ denotes convolutional operator. Since the system is linear time invariant (LTI), the cost in calculating the convolution between $x(k)$ and $h(k)$ can be reduced by taking the advantage of the convolutional rule. Fig. 13(b) shows an equivalent block diagram of the noise reduction system processed in frequency domain. The Fourier transform of the received signal $x(k)$ is calculated and multiplied by the filter frequency response $H(\omega)$. The signal estimate is then obtained by calculating the inverse Fourier transform of the product

$$\hat{S}(\omega) = H(\omega)X(\omega).$$

It is easy to show that the optimal filter $H(\omega)$ that minimizes the mean square error $E|\hat{s}(k) - s(k)|^2$ can be given by [2]

$$H(\omega) = \frac{\Phi_s(\omega)}{\Phi_x(\omega)} = \frac{\Phi_s(\omega)}{\Phi_s(\omega) + \Phi_n(\omega)} = 1 - \frac{\Phi_n(\omega)}{\Phi_x(\omega)} \qquad (15)$$

where $\Phi_s(\omega)$, $\Phi_n(\omega)$, and $\Phi_x(\omega)$ are the power spectrum densities of $s(k)$, $n(k)$, and $x(k)$, respectively. Note that in the second equation of (15), we have used the fact that $s(k)$ and $n(k)$ are independent.

In this experiment, the signals $s(k)$ and $n(k)$ are generated by two second-order random processes described by

$$s(k) = 1.2s(k-1) - 0.9s(k-2) + w_s(k),$$
$$n(k) = 0.1n(k-1) - 0.85n(k-2) + w_n(k)$$

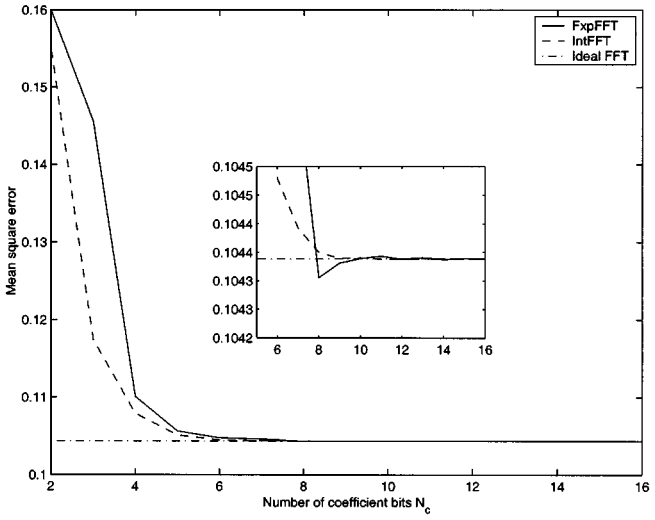

Fig. 14. Mean square error of the signal estimate for different resolutions of the coefficients.

where $w_s(k)$ and $w_n(k)$ are independent white Gaussian noises. The variances of $w_s(k)$ and $w_n(k)$ are chosen so that the signals $s(k)$ and $n(k)$ have unit variance and, hence, the SNR at the received signal $x(k)$ is 0 dB. This mixtured signal $x(k)$ is divided into blocks, each of which has $N = 256$ samples, i.e., the size of the FFT and IFFT is 256. The comparison begins by processing the noise reduction algorithm as depicted in Fig. 13(b), where the FFT and IFFT blocks are performed by the FxpFFT and the IntFFT (and their inverses). The butterfly coefficients of FxpFFT and the lifting coefficients of IntFFT are quantized to N_c bits for different values of N_c. The internal node N_n is set to 24 bits.

Fig. 14 shows the mean square error between the signal estimate $\hat{s}(k)$ and the original signal $s(k)$ over the range of N_c. The solid line is obtained by using the FxpFFT and its inverse, whereas the dash line is obtained by using the IntFFT and its inverse. As N_c increases, the mean square errors of both cases decrease, approaching the dash-dot line, which is when the exact FFT and IFFT are used. It is evident that when the coefficients are severely quantized ($2 \leq N_c \leq 7$), the mean square error in the case of IntFFT is lower than that in the case of FxpFFT by up to 18%. This is because when N_c is small, the reconstruction error of the FxpFFT and its inverse tends to dominate the reconstruction signal. On the other hand, at high resolution ($N_c \geq 8$), both IntFFT and FxpFFT yield similar results, which are very close to the case of the exact FFT as indicated in the enlarged portion of the curves in Fig. 14.

VIII. Conclusion

In this paper, we have presented a concept of IntFFT, which can be used to construct FFT with integer coefficients. It provides a new method for approximating the DFT without using any multiplication and can simply be applied to the case of large-size DFT. Unlike the FxpFFT, which is the fixed-point arithmetic version of FFT, the IntFFT is reversible when the coefficients are quantized. Its inverse IntIFFT can be computed with the same computational cost as that of the forward transform. The new transform is suitable for mobile

computing and any handheld devices that run on batteries since it is adaptable to available power resources. Specifically, the coefficients appearing in the proposed structures can be quantized directly for different resolutions, i.e., different computational costs, while preserving the reversibility property.

Although a large class of FFT structures such as radix-2, radix-4, and split-radix can be approximated by this approach, the split-radix structure is used to illustrate the technique. An analysis of the dynamic range of the internal nodes is presented. Using an appropriate choice of lifting factorizations, it is proven that lifting approximation of a complex multiplier can increase the resolution of the input by at most one bit. An upper bound of the dynamic range of the internal nodes is estimated and confirmed by a simulation for the case of the split-radix FFT. The computational complexity of the resulting transform is calculated by the numbers of real additions and shifts. A method for minimizing the number of additions is presented and used to compare the computational costs of `IntFFT` and `FxpFFT`. According to the simulation, the complexity of `IntFFT` is lower than that of `FxpFFT` by a significant margin.

The accuracy of the transforms is compared experimentally. It is evident from the simulations that both `IntFFT` and `FxpFFT` have approximately the same distortion from ideal FFT when the resolution of the input N_i is greater than N_c. When $N_c \geq N_i$, the `FxpFFT` yields 3 dB higher accuracy. The results are different when testing with the convolutional rule, where the `IntFFT` and the `FxpFFT` provide essentially the same accuracy for any value of N_c. When the inverse transform is performed after the forward transform, fixed-point arithmetic approach results in reconstruction error, whereas the proposed approach can reconstruct the input perfectly for any fixed resolution of the coefficients. Finally, these two implementations are tested in noise reduction application. At low power, i.e., the coefficients are quantized to low resolution, the `IntFFT` yields significantly better results than the `FxpFFT`, and they yield similar results at high power.

APPENDIX
PROOF OF THEOREM VI.1

According to Fig. 7, we get

$$t_1 = Re\{u\} + \frac{c-1}{s} Im\{u\} + n_1$$
$$t_2 = sRe\{u\} + cIm\{u\} + sn_1 + n_2$$
$$t_3 = cRe\{u\} - sIm\{u\} + (c-1)n_1 + \frac{c-1}{s} n_2 + n_3.$$

In order to prove the statement, it is sufficient to show that $|t_i| \leq 2\max(|Re\{u\}|, |Im\{u\}|)$. Since $|(c-1)/s| \leq 1$

$$\left|\left(\frac{c-1}{s}\right) Im\{u\}\right| \leq |Im\{u\}|.$$

If $Im\{u\}$ has finite word length, the quantization of $((c-1)/s)Im\{u\}$ must be less than or equal to $|Im\{u\}|$, i.e.,

$$\left|\left(\frac{c-1}{s}\right) Im\{u\} + n_1\right| \leq |Im\{u\}|.$$

Hence

$$\begin{aligned}|t_1| &\leq |Re\{u\}| + \left|\left(\frac{c-1}{s}\right) Im\{u\} + n_1\right| \\ &\leq |Re\{u\}| + |Im\{u\}| \leq 2\max(|Re\{u\}|, |Im\{u\}|).\end{aligned}$$

From the expression of t_2, we have

$$\begin{aligned}|t_2| &\leq |sRe\{u\} + cIm\{u\}| + |sn_1| + |n_2| \\ &\leq \sqrt{Re\{u\}^2 + Im\{u\}^2} + |n_1| + |n_2| \\ &\leq \left(\sqrt{2} + \frac{2\left(2-\sqrt{2}\right)}{3}\right) \max(|Re\{u\}|, |Im\{u\}|) \\ &< 2\max(|Re\{u\}|, |Im\{u\}|).\end{aligned}$$

It is easy to see that $|(c-1)/s| \leq 1$ if and only if $c \geq 0$, and thus, $|c-1| \leq 1$. Hence

$$\begin{aligned}|t_3| &\leq |cRe\{u\} - sIm\{u\}| + |(c-1)n_1| \\ &\quad + \left|\left(\frac{c-1}{s}\right) n_2\right| + |n_3| \\ &\leq \left(\sqrt{2} + \frac{3\left(2-\sqrt{2}\right)}{3}\right) \max(|Re\{u\}|, |Im\{u\}|) \\ &= 2\max(|Re\{u\}|, |Im\{u\}|).\end{aligned}$$

REFERENCES

[1] J. S. Walker, *Fast Fourier Transforms*, 2nd ed. Boca Raton, FL: CRC, 1996.

[2] L. Krasny and S. Orantara, "Noise reduction for speech signals using voice activity detector," Ericsson, Inc., Research Triangle Park, NC, EUS/TR/X-98:1662, Oct. 1998.

[3] A. K. Jain, *Fundamental of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[4] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comput.*, vol. 19, pp. 297–301, April 1965.

[5] M. T. Heideman, *Multiplicative Complexity Convolutional, and the DFT*. New York: Springer-Verlag, 1988.

[6] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[7] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: Principles, Algorithms and Applications*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[8] T-C. Pang, C-S. O. Choi, C-F. Chan, and W-K. Cham, "A self-timed ICT chip for image coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 856–860, Sept. 1999.

[9] T. Tran, "Fast multiplierless approximation of the DCT," in *Proc. CISS*, 1999.

[10] Y-J. Chen, S. Orantara, and T. Nguyen, "Video compression using integer DCT," in *Proc. ICIP*, 2000.

[11] S-C. Pei and J-J. Ding, "Integer discrete Fourier transform and its extension to integer trigomatric transforms," in *Proc. ISCAS*, 2000.

[12] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*. New York: Academic, 1990.

[13] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," Bell Labs., Lucent Technol., Red Bank, NJ, 1996.

[14] G. Strang and T. Nguyen, *Wavelets and Filter Banks*. Wellesley, MA: Wellesley-Cambridge, 1996.

[15] R. Storn, "Some results in fixed point error analysis of the Bruun-FFT algorithm," *IEEE Trans. Singal Processing*, vol. 41, pp. 2371–2375, July 1993.

[16] M. T. Heideman and C. S. Burrus, "On the number of multiplications necessary to compute a length 2^n DFT," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 91–95, Feb. 1986.

Soontorn Oraintara (S'97–M'00) received the B.E. degree (with first-class honors) from the King Monkut's Institute of Technology Ladkrabang, Bangkok, Thailand, in 1995 and the M.S. and Ph.D. degrees, both in electrical engineering, from the University of Wisconsin, Madison, in 1996 and Boston University, Boston, MA, in 2000, respectively.

He joined the Department of Electrical Engineering, University of Texas at Arlington, as Assistant Professor in July 2000. From May 1998 to April 2000, he was an intern and a consultant at the Advanced Research and Development Group, Ericsson, Inc., Research Triangle Park, NC. His current research interests are in the field of digital signal processing: wavelets, filterbanks, and multirate systems and their applications in data compression; signal detection and estimation; communications; image reconstruction; and regularization and noise reduction.

Dr. Oraintara received the Technology Award from Boston University for his invention on Integer DCT (with Y. J. Chen and Prof. T. Q. Nguyen) in 1999. He represented Thailand in the International Mathematical Olympiad competitions and, respectively, received the Honorable Mention Award, in Beijing, China, in 1989 and the bronze medal in Sigtuna, Sweden, in 1990.

Ying-Jui Chen (S'00) was born in 1972. He received the B.S. and M.S. degrees in electrical engineering from National Taiwan University, Taipei, Taiwan, R.O.C., in 1994 and 1996, respectively. Currently, he is pursuing the Ph.D. degree with the Civil and Environmental Engineering Department, Massachusetts Institute of Technology, Cambridge, under the supervision of Prof. K. Amaratunga.

His research interests include pattern recognition, multirate filterbanks, wavelet analysis, statistical signal processing, and integer implementation of transforms. Some of his conference papers have been published on the integer discrete cosine transform (see http://wavelets.mit.edu/~yrchen/Research).

Truong Q. Nguyen (SM'95) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the California Institute of Technology (Caltech), Pasadena, in 1985, 1986, and 1989, respectively.

He was with Lincoln Laboratory, Massachusetts Institute of Technology (MIT), Lexington, from June 1989 to July 1994, as a Member of Technical Staff. From 1993 to 1994, he was a Visiting Lecturer at MIT and an Adjunct Professor at Northeastern University, Boston, MA. From August 1994 to July 1998, he was with the Electrical and Computer Engineering Department, University of Wisconsin, Madison. He was with Boston University, Boston, MA, from 1996 to 2001. He is now a Full Professor at University of California at San Diego, La Jolla. His research interests are in the theory of wavelets and filterbanks and applications in image and video compression, telecommunications, bioinformatics, medical imaging and enhancement, and analog/digital conversion. He is the coauthor (with Prof. G. Strang) of a popular textbook, *Wavelets and Filter Banks* (Welleley, MA: Wellesley-Cambridge, 1997), and the author of several MATLAB-based toolboxes on image compression, electrocardiogram compression, and filterbank design. He also holds a patent on an efficient design method for wavelets and filterbanks and several patents on wavelet applications, including compression and signal analysis.

Prof. Nguyen received the IEEE TRANSACTIONS ON SIGNAL PROCESSING Paper Award (in the Image and Multidimensional Processing area) for the paper he co-wrote with Prof. P. P. Vaidyanathan on linear-phase perfect-reconstruction filter banks in 1992. He received the NSF Career Award in 1995 and is currently the Series Editor (Digital Signal Processing) for Academic Press. He served as Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 1994 to 1996 and for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS from 1996 to 1997. He received the Technology Award from Boston University for his invention on Integer DCT (with Y. J. Chen and S. Oraintara). He co-organizes (with Prof. G. Strang at MIT) several workshops on wavelets and served on the DSP Technical Committee for the IEEE CAS society.