Input and Output Index Mappings for a Prime-Factor-Decomposed Computation of Discrete Cosine Transform

BYEONG GI LEE, MEMBER, IEEE

Abstract—**This paper provides a direct derivation of the prime-factor-decomposed computation algorithm of an N-point discrete cosine transform for the number N decomposable into two relative prime numbers. It also presents input and output index mappings in the form of tables—namely, $\hat{n}$-, $\tilde{n}$-, n_C-, n_R-, and k-tables. The index mapping tables are useful for practical use of the prime-factor-decomposed computation of arbitrarily sized discrete cosine transforms.**

I. Introduction

SINCE its first introduction in 1974 [1], the discrete cosine transform (DCT) has found applications in speech and image signal processing [1]–[8] as well as in telecommunication signal processing [9], [10]. The DCT has been applied for speech and image compression because its performance was nearly optimal, yet not being signal dependant. On the other hand, the DCT has been utilized for realizing filter banks in FDM-TDM transmultiplexers because its real computation was simpler and faster than the complex computation of the discrete Fourier transform (DFT).

Along with the expanded applications of the DCT, a number of fast computation techniques have been also introduced [11]–[19]. Depending on the number of points N, the computation techniques can be divided into two categories: one on the general composite number cases, and the other one on the prime-factor cases. For the former case, N of special interest is of 2^m type; for the latter case, N is factorizable into two mutually relative prime numbers N_1 and N_2. A recent work reports that the number of real multiplications for the power-of-two case can reduce to $(N/2) \log N$, and its structure resembles that of the fast Fourier transform (FFT) [14]. For the prime-factor case, the number of multiplications reduces to $N(N_1 + N_2)$ in its most primitive form, and its structure is similar to that of the prime-factor algorithm of the DFT [19].[1]

The prime-factor-decomposed computation of the DCT was proven to be powerful in reducing the computational work and time, still providing a simple and nice structure. However, this technique has not yet been widely utilized mainly because its input and output index mappings are seemingly too involved. In fact, the mappings are the only barrier to overcome in applying the prime-factor algorithm. This paper is therefore intended to provide a simple and organized method to perform the index mappings.

In this paper, a formal direct derivation of the prime-factor-decomposed computation algorithm will be presented first. The derivation is a direct one in the sense that it is based on the real cosine function without resorting to the DFT expressions or the complex functions. Then, based on the equations obtained during the derivation, input and output index mappings will be introduced in the form of tables. This tabulation will enable us to implement any prime-factor-decomposable DCT in a straightforward manner. Finally, the index mapping tables will be demonstrated through the 12-point DCT.

II. Direct Derivation of Prime Factor Decomposition

Let $x(k)$, $k = 0, 1, \cdots, N-1$, be a time-domain sequence and $X(n)$, $n = 0, 1, \cdots, N-1$, be its transform-domain data sequence. Then, by definition, the DCT and the inverse DCT (IDCT), respectively, have the expressions

$$X(n) = \frac{2}{N} e(n) \sum_{k=0}^{N-1} x(k) \cos\left[\pi(2k+1)n/2N\right], \quad n = 0, 1, \cdots, N-1, \tag{1}$$

$$x(k) = \sum_{n=0}^{N-1} e(n) X(n) \cos\left[\pi(2k+1)n/2N\right], \quad k = 0, 1, \cdots, N-1, \tag{2}$$

where

$$e(n) = \begin{cases} 1/\sqrt{2}, & \text{if } n = 0, \\ 1, & \text{otherwise.} \end{cases} \tag{3}$$

Since (1) can be realized simply by transposing the flowgraph for (2), and since the term $e(n)$ means nothing but a slight modification of the data $X(n)$, it is sufficient

Manuscript received September 12, 1987; revised March 5, 1988.

The author is with the Department of Electronics Engineering, Seoul National University, Seoul, 151-742, Korea.

IEEE Log Number 8825133.

[1] On the prime-factor algorithm of DFT, refer to [20]–[22].

0096-3518/89/0200-0237$01.00 © 1989 IEEE

for our discussion to consider the IDCT-like equation[2]

$$x(k) = \sum_{n=0}^{N-1} X(n) \cos\left[\pi(2k+1)n/2N\right], \quad k = 0, 1, \cdots, N-1. \tag{4}$$

Throughout this paper we will assume that

$$N = N_1 N_2, \tag{5}$$

where N_1 and N_2 are mutually prime integers. Suppose we could decompose the N-point IDCT in (4) into the cascade of N_2 N_1-point IDCT's and N_1 N_2-point IDCT's.[3] Then, the expression for the resulting decomposed transform would be of the form

$$x(k_1, k_2) = \sum_{n_2=0}^{N_2-1} \left\{ \sum_{n_1=0}^{N_1-1} X(n_1, n_2) \cdot \cos\left[\pi(2k_1+1)n_1/2N_1\right] \cdot \cos\left[\pi(2k_2+1)n_2/2N_2\right] \right\}, \tag{6}$$

for $k_1 = 0, 1, \cdots, N_1 - 1$, and $k_2 = 0, 1, \cdots, N_2 - 1$.

The main goal of this section is in deriving (6) from (4) by finding two appropriate mappings: the input mapping connecting $X(n)$, $n = 0, 1, \cdots, N-1$, to $X(n_1, n_2)$, $n_1 = 0, 1, \cdots, N_1 - 1$, $n_2 = 0, 1, \cdots, N_2 - 1$, and the output mapping connecting $x(k)$, $k = 0, 1, \cdots, N - 1$, to $x(k_1, k_2)$, $k_1 = 0, 1, \cdots, N_1 - 1$, $k_2 = 0, 1, \cdots, N_2 - 1$.[4] The input and output mappings are directly tied with the input and output index mappings among the corresponding indexes.

We first consider the input mapping which connects $X(n)$ to $X(n_1, n_2)$.

Let $\mathfrak{N}$ denote the set of N integers 0 through $N-1$. Similarly, let $\mathfrak{N}_1$ and $\mathfrak{N}_2$, respectively, denote the sets of N_1 integers 0 through $N_1 - 1$ and the set of N_2 integers 0 through $N_2 - 1$. We define $\hat{f}$ and $\tilde{f}$ to be mappings from $\mathfrak{N}_1 \times \mathfrak{N}_2$ to $\mathfrak{N}$ such that

$$\hat{f}(n_1, n_2) = \begin{cases} n_1N_2 + n_2N_1, & \text{if } n_1N_2 + n_2N_1 < N, \\ 2N - (n_1N_2 + n_2N_1), & \text{otherwise,} \end{cases} \tag{7a}$$

$$\tilde{f}(n_1, n_2) = |n_1N_2 - n_2N_1| \tag{7b}$$

for an n_1 in $\mathfrak{N}_1$, and an n_2 in $\mathfrak{N}_2$. Then neither $\hat{f}$ nor $\tilde{f}$ is a one-to-one mapping since

$$\hat{f}(n_1, n_2) = \hat{f}(N_1 - n_1, N_2 - n_2), \tag{8a}$$

$$\tilde{f}(n_1, n_2) = \tilde{f}(N_1 - n_1, N_2 - n_2), \tag{8b}$$

for all n_1 in $\mathfrak{N}_1$ and all n_2 in $\mathfrak{N}_2$. We denote by $\hat{\mathfrak{N}}$ and $\tilde{\mathfrak{N}}$ sets of N integers such that

$$\hat{\mathfrak{N}} = \left\{ n \mid n = \hat{f}(n_1, n_2), \quad n_1 \in \mathfrak{N}_1, \quad n_2 \in \mathfrak{N}_2 \right\}, \tag{9a}$$

$$\tilde{\mathfrak{N}} = \left\{ n \mid n = \tilde{f}(n_1, n_2), \quad n_1 \in \mathfrak{N}_1, \quad n_2 \in \mathfrak{N}_2 \right\}. \tag{9b}$$

Then it can be shown that the $2N$ integers in the collection of $\hat{\mathfrak{N}}$ and $\tilde{\mathfrak{N}}$ are identical to the $2N$ integers in the collection of $\mathfrak{N}$ and $\mathfrak{N}$.[5] This implies that a summation over N indexes in $\mathfrak{N}$ can split into two terms—a summation over the N indexes in $\hat{\mathfrak{N}}$ and a summation over N indexes in $\tilde{\mathfrak{N}}$. Therefore, we can rewrite (4) as follows:

$$x(k) = 1/2 \sum_{n \in \hat{\mathfrak{N}}} X(n) \cos\left[\pi(2k+1)n/2N\right] + 1/2 \sum_{n \in \tilde{\mathfrak{N}}} X(n) \cos\left[\pi(2k+1)n/2N\right]. \tag{10}$$

We denote, for all (n_1, n_2) in $\mathfrak{N}_1 \times \mathfrak{N}_2$,

$$\hat{X}(n_1, n_2) = s(n)\, X(n)\big|_{n=\hat{f}(n_1, n_2)}, \tag{11a}$$

$$\tilde{X}(n_1, n_2) = X(n)\big|_{n=\tilde{f}(n_1, n_2)}, \tag{11b}$$

where

$$s(n) = \begin{cases} 1, & \text{if } n_1N_2 + n_2N_1 < N, \\ -1, & \text{otherwise.} \end{cases} \tag{12}$$

Then (10) can be rewritten as

$$x(k) = 1/2 \sum_{n_2=0}^{N_2-1} \sum_{n_1=0}^{N_1-1} \left\{ \hat{X}(n_1, n_2) \cdot \cos\left[\pi(2k+1)(n_1N_2 + n_2N_1)/2N\right] + \tilde{X}(n_1, n_2) \cos\left[\pi(2k+1) \cdot (n_1N_2 - n_2N_1)/2N\right] \right\}. \tag{13}$$

The term $s(n)$ reflects the negative sign appearing in the relation

$$\cos\left[\pi(2k+1)(2N-n)/2N\right] = -\cos\left[\pi(2k+1)n/2N\right]. \tag{14}$$

We do not need such a term for the second part of (10), since $|n_1 N_2 - n_2 N_1| < N$ for all (n_1, n_2) in $\mathfrak{N}_1 \times \mathfrak{N}_2$.

We now define $X(n_1, n_2)$ such that

$$X(n_1, n_2) = \begin{cases} \hat{X}(n_1, n_2) = \tilde{X}(n_1, n_2), \\ \qquad \text{if } n_1 = 0 \quad \text{or } n_2 = 0, \\ \hat{X}(n_1, n_2) + \tilde{X}(n_1, n_2), \quad \text{otherwise.} \end{cases} \tag{15}$$

[2] It should be noted that the transposed flowgraph of DCT performs the IDCT function (with $e(n)$ related coefficient modification), while the transposed flowgraph of DFT still performs the same DFT function. Refer to the arrow marks in Fig. 1 to appear.

[3] The term IDCT we encounter hereafter will denote the IDCT in the sense of (4).

[4] We name the former input mapping and the latter output mapping, even though the meaning of input and output could be reversed for the forward DCT.

[5] A proof of this is given in Appendix A. The term collection here indicates the set obtained by listing all the elements in two sets. See footnote 11.

Then, since

$$\sum_{n_2=1}^{N_2-1}\sum_{n_1=1}^{N_1-1} \tilde{X}(n_1, n_2) \cos\left[\pi(2k+1) \cdot (n_1N_2 + n_2N_1)/2N\right] = 0, \tag{16a}$$

$$\sum_{n_2=1}^{N_2-1}\sum_{n_1=1}^{N_1-1} \hat{X}(n_1, n_2) \cos\left[\pi(2k+1) \cdot (n_1N_2 - n_2N_1)/2N\right] = 0, \tag{16b}$$

equation (13) can be written in the form

$$x(k) = 1/2 \sum_{n_2=0}^{N_2-1}\sum_{n_1=0}^{N_1-1} X(n_1, n_2) \cdot \left\{\cos\left[\pi(2k+1)(n_1N_2 + n_2N_1)/2N\right] + \cos\left[\pi(2k+1)(n_1N_2 - n_2N_1)/2N\right]\right\}. \tag{17}$$

Recalling the relation

$$1/2 \left\{\cos\left[\pi(2k+1)(n_1N_2 + n_2N_1)/2N\right] + \cos\left[\pi(2k+1)(n_1N_2 - n_2N_1)/2N\right]\right\} = \cos\left[\pi(2k+1)\, n_1N_2/2N\right] \cdot \cos\left[\pi(2k+1)\, n_2N_1/2N\right], \tag{18}$$

we can rewrite it again as

$$x(k) = \sum_{n_2=0}^{N_2-1}\sum_{n_1=0}^{N_1-1} X(n_1, n_2) \cos\left[\pi(2k+1)\, n_1/2N_1\right] \cdot \cos\left[\pi(2k+1)\, n_2/2N_2\right]. \tag{19}$$

Now, we consider the output mapping which connects $x(k)$ to $x(k_1, k_2)$. We denote by $\bar{k}_1$ and $\bar{k}_2$, respectively, $\bar{k}_1 = k$ modulo $2N_1$, $\bar{k}_2 = k$ modulo $2N_2$. We define g to be a mapping from $\mathfrak{N}$ to $\mathfrak{N}_1 \times \mathfrak{N}_2$ such that $(k_1, k_2) = g(k)$ with

$$k_1 = \begin{cases} \bar{k}_1, & \text{if } \bar{k}_1 < N_1, \\ 2N_1 - 1 - \bar{k}_1, & \text{otherwise}, \end{cases} \tag{20a}$$

$$k_2 = \begin{cases} \bar{k}_2, & \text{if } \bar{k}_2 < N_2, \\ 2N_2 - 1 - \bar{k}_2, & \text{otherwise}, \end{cases} \tag{20b}$$

for each (k_1, k_2) in $\mathfrak{N}_1 \times \mathfrak{N}_2$, and a k in $\mathfrak{N}$. Then, it can be shown that g is a one-to-one mapping, so there exists an inverse mapping g from $\mathfrak{N}_1 \times \mathfrak{N}_2$ to $\mathfrak{N}$.[6] Thus, we can define

$$x(k_1, k_2) = x(k)\big|_{k=g^{-1}(k_1,k_2)}. \tag{21}$$

If we apply this to (19), we finally obtain the expression in (6), since

$$\cos\left[\pi(2k+1)\, n_1/2N_1\right] = \cos\left[\pi(2k_1+1)\, n_1/2N_1\right], \tag{22a}$$

$$\cos\left[\pi(2k+1)\, n_2/2N_2\right] = \cos\left[\pi(2k_2+1)\, n_2/2N_2\right]. \tag{22b}$$

Therefore, we have shown that (4) and (6) are identical if $X(n)$ and $X(n_1, n_2)$ are connected through (7), (11), and (15), and if $x(k)$ and $x(k_1, k_2)$ are connected through (20) and (21). The former three equations form the input mapping; and the latter two equations form the output mapping. Thus, we can now perform an N-point IDCT by cascading N_2 N_1-point IDCT's and N_1 N_2-point IDCT's, as is demonstrated in Fig. 1 for the numbers $N = 12$, $N_1 = 3$, $N_2 = 4$.[7]

III. Tabulation of Index Mappings

For any given N, the main body performing N_2 N_1-point IDCT's and N_1 N_2-point IDCT's is quite straightforward to implement, as is illustrated in Fig. 1. But more care should be taken on the input mapping which converts $X(n)$, $n = 0, 1, \cdots, N-1$ to $X(n_1, n_2)$, $n_1 = 0, 1, \cdots, N_1 - 1$, $n_2 = 0, 1, \cdots, N_2 - 1$, and the output mapping which converts $x(k_1, k_2)$, $k_1 = 0, 1, \cdots, N_1 - 1$, $k_2 = 0, 1, \cdots, N_2 - 1$, to $x(k)$, $k = 0, 1, \cdots, N - 1$. In this section, we will tabulate the index mappings relating n to (n_1, n_2) and relating (k_1, k_2) to k, and will discuss how to utilize the resulting tables to realize the above input and output mappings.

We first consider the input mapping, which is represented by (7), (11), and (15). We set up a table with N_1 rows and N_2 columns, naming each row 0 through $N_1 - 1$ and each column 0 through $N_2 - 1$. Then we fill location (n_1, n_2), which is at row n_1 and column n_2, with $\hat{f}(n_1, n_2)$ for $n_1 = 0, 1, \cdots, N_1 - 1$, $n_2 = 0, 1, \cdots, N_2 - 1$, putting a negative sign to every $\hat{f}(n_1, n_2)$ meeting $n_1N_2 + n_2N_1 \geq N$. The negative sign is meant to reflect the $s(n)$ term in (12), designating that the input data $X(\hat{f}(n_1, n_2))$ corresponding to the index should accompany a negative sign as (11a) indicates. We name the resulting table *$\hat{n}$-table*. Then the $\hat{n}$-table represents (7a) and (11a). We define *$\tilde{n}$-table* in a similar manner by listing $\tilde{f}(n_1, n_2)$ to location (n_1, n_2), $n_1 = 0, 1, \cdots, N_1 - 1$, $n_2 = 0, 1, \cdots, N_2 - 1$. Then the $\tilde{n}$-table will represent (7b) and (11b). Note that there is no negative sign in the $\tilde{n}$-table.

If we denote by $\hat{n}(n_1, n_2)$ and $\tilde{n}(n_1, n_2)$ the entries at (n_1, n_2) of the $\hat{n}$- and $\tilde{n}$-tables, respectively, then we have $\hat{n}(n_1, n_2) = \tilde{n}(n_1, n_2)$ for $n_1 = 0$ or $n_2 = 0$, and $\hat{n}(n_1, n_2) = \hat{n}(N_1 - n_1, N_2 - n_2)$, $\tilde{n}(n_1, n_2) = \tilde{n}(N_1 - n_1, N_2 - n_2)$ for the other n_1 and n_2. This reflects (8a) and (8b), implying that neither the entries in the $\hat{n}$-table nor the entries in the $\tilde{n}$-table cover the set $\mathfrak{N}$. So we introduce new tables which can cover the set $\mathfrak{N}$. We define *n_C-table* to be the table obtained from the $\hat{n}$- and $\tilde{n}$-tables in the following manner. If $n_1 = 0$ or $n_2 = 0$, we take $n_C(n_1, n_2) = \hat{n}(n_1, n_2) = \tilde{n}(n_1, n_2)$, where $n_C(n_1, n_2)$ denotes the entry at (n_1, n_2) of the n_C-table. Otherwise, we take $n_C(n_1, n_2) = \tilde{n}(n_1, n_2)$ for the first $(N_1 - 1)(N_2 - 1)/2$

[6]A proof of this is given in Appendix B.

[7]It should be noted that the block diagram in Fig. 1 performs IDCT if signals flow from left to right, and performs DCT if signals flow from right to left.

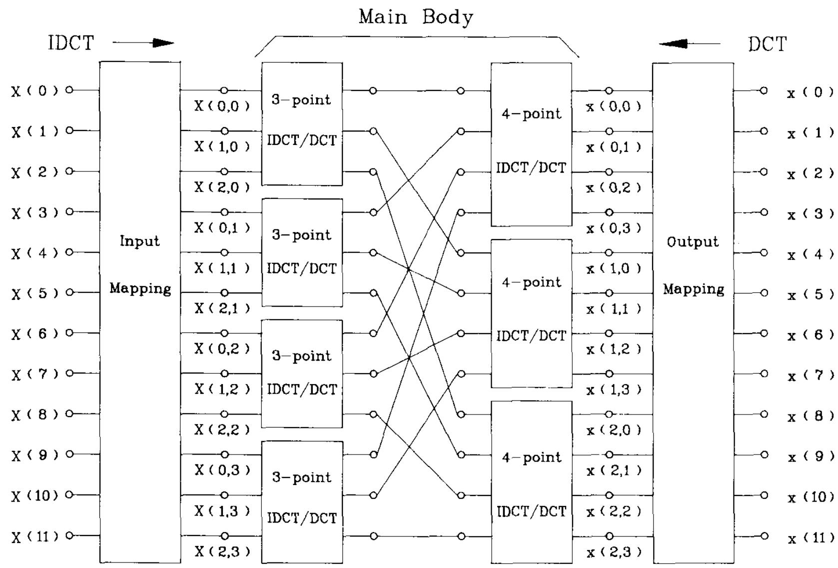


Fig. 1. Decomposition of N-point IDCT into the cascade of N_1-point IDCT's and N_2-point IDCT's, where $N = 12$, $N_1 = 3$, and $N_2 = 4$.

entries taken columnwise,[8] and $n_C(n_1, n_2) = \hat{n}(n_1, n_2)$ for the others but without negative signs. Also, we define n_R-*table* in a similar manner, except that we take entries rowwise.[8] Then, clearly, the entries in the n_C-table, as well as the entries in the n_R-table, cover the set $\mathfrak{N}$. Therefore, we can align data $X(n)$ and $X(n_1, n_2)$ for the input mapping by taking either n_C- or n_R-table.

We consider the case when N_2 N_1-point IDCT's appear first in the main body, as in Fig. 1. For this case, the output data of the input mapping box should be aligned in the order of $X(0, 0), X(1, 0), \cdots, X(N_1 - 1, 0); X(0, 1), X(1, 1), \cdots, X(N_1 - 1, 1); \cdots; X(0, N_2 - 1), X(1, N_2 - 1), \cdots, X(N_1 - 1, N_2 - 1)$. This implies that we have to read out data columnwise. So we take the n_C-table and read out the input data $X(n)$ columnwise, juxtaposing them to the left-hand side of the corresponding $X(n_1, n_2)$. Based on these alignments, we now perform the operation in (15). We can use $\hat{n}$- and $\tilde{n}$-tables, respectively, for the terms $\hat{X}(n_1, n_2)$ and $\tilde{X}(n_1, n_2)$. Due to the symmetric properties of the $\hat{n}$- and $\tilde{n}$-tables described by (8), equation (15) can be realized as a flowgraph of the following shape. The output data $X(n_1, n_2)$ with $n_1 = 0$ or $n_2 = 0$ is joined with the corresponding input data $X(n)$ by a unity-gain line. For other input data, there exists a butterfly operation joining every $X(n_1, n_2)$ and $X(N_1 - n_1, N_2 - n_2)$ with their corresponding counterparts in $X(n)$, $n = 0, 1, \cdots, N - 1$. The signs for the butterfly operation follow those in the $\hat{n}$- and $\tilde{n}$-tables.[9]

[8]The term "columnwise" implies that we are taking the entries at the first column first, followed by the second through the last columns sequentially. More specifically, we are taking entries in the order $(0, 0), (1, 0), \cdots, (N_1 - 1, 0); (0, 1), (1, 1), \cdots, (N_1 - 1, 1); \cdots; (0, N_2 - 1), (1, N_2 - 1), \cdots, (N_1 - 1, N_2 - 1)$. A similar interpretation is also valid for the term "rowwise."

[9]Note that the signs of all but those having negative signs are positive.

In the case when N_1 N_2-point IDCT's appear first in the main body, the output data of the input mapping box are aligned in the order of $X(0, 0), X(0, 1), \cdots, X(0, N_2 - 1); X(1, 0), X(1, 1), \cdots, X(1, N_2 - 1); \cdots; X(N_1 - 1, 0), X(N_1 - 1, 1), \cdots, X(N_1 - 1, N_2 - 1)$. Since this implies taking data rowwise, we now take the n_R-table and read out data $X(n)$ rowwise. The other processes are the same as the previous case.

We need not consider both n_C- and n_R-tables for a given N. If we want to perform the N_1-point IDCT operation first, followed by the N_2-point IDCT operation, then the n_C-table together with the $\hat{n}$- and $\tilde{n}$-tables will suffice. But if we want to do N_2-point IDCT first, then we need the n_R-table instead of the n_C-table. Depending on the table we take for the input mapping, the output mapping also changes, as will be discussed below.

We now consider the output mapping, which is represented by (20) and (21). This case is rather simple compared to the previous case. Notice that this tabulation is possible due to the one-to-oneness of the mapping g.

We make a table with N_1 rows and N_2 columns, naming each row 0 through $N_1 - 1$ and each column 0 through $N_2 - 1$ as before. We write k, $k = 0, 1, \cdots, N - 1$, to location (k_1, k_2), where k_1 and k_2 follow (20a) and (20b), respectively. We name the resulting table k-*table*.

Using this table, we now consider realizing (21). We first consider the case when the N_1-point IDCT's were performed first, followed by the N_2-point IDCT's in the main body. For this case, the input data $x(k_1, k_2)$ (to the output mapping box), which correspond to the output data of the N_2-point IDCT's of the main body, are aligned in the order $x(0, 0), x(0, 1), \cdots, x(0, N_2 - 1); x(1, 0), x(1, 1), \cdots, x(1, N_2 - 1); \cdots; x(N_1 - 1, 0), x(N_1 - 1, 1), \cdots, x(N_1 - 1, N_2 - 1)$. This implies that we have to take data rowwise. Thus, we read out the output data $x(k)$ rowwise from the k-table, juxtaposing them to the right-hand side of the corresponding $x(k_1, k_2)$. On these alignments, we now perform the operation corresponding to (21), which is nothing but a unity-gain line drawing between each $x(k_1, k_2)$ and the corresponding counterpart $x(k)$. For the other case, where we perform the N_2-point IDCT's first and the N_1-point IDCT's last, we can repeat a similar procedure. Since we now have the alignment $x(0, 0), x(1, 0), \cdots, x(N_1 - 1, 0); x(0, 1), x(1, 1), \cdots, x(N_1 - 1, 1); \cdots; x(0, N_2 - 1), x(1, N_2 - 1), \cdots, x(N_1 - 1, N_2 - 1)$, we have to take the k-table columnwise in this case.

Therefore, we can summarize the usage of the tables as follows. If we want to arrange the main body of the decomposed system such that the N_2 N_1-point IDCT's come first, followed by the N_1 N_2-point IDCT's, then we should take the *n_C-table columnwise* in the input mapping, taking the *k-table rowwise* in the output mapping. But if we want to have the reversed order, we have to take the n_R-table rowwise and the k-table columnwise. For either case, the $\hat{n}$ and $\tilde{n}$-tables are used in common to provide butterfly connections.

IV. Example

We consider the 12-point IDCT/DCT to illustrate the decomposed computation and the corresponding n- and k-tables. We let $N_1 = 3$, $N_2 = 4$, and perform four 3-point IDCT's first followed by three 4-point IDCT's. Then we obtain the structure shown in Fig. 1. One can also show a similar structure for the case when three 4-point IDCT's are taken first. We want to figure out the black boxes named input and output mappings in the figure.

We first draw up the $\hat{n}$-, $\tilde{n}$-, n_C-, and n_R-tables for the input mapping box. By evaluating (7a) with $N_1 = 3$, $N_2 = 4$, $n_1 = 0, 1, 2$, and $n_2 = 0, 1, 2, 3$, we obtain the $\hat{n}$-table shown in Table I(a). Similarly, using (7b), we obtain the $\tilde{n}$-table in Table I(b). We now derive the n_C- and n_R-tables from the above two tables. For the locations with $n_1 = 0$, or $n_2 = 0$ of both tables, we copy the corresponding entries in the $\hat{n}$- or $\tilde{n}$-table. But for the other locations, we copy down the first three entries from the $\tilde{n}$-table and the other three from the $\hat{n}$-table, since $(N_1 - 1)(N_2 - 1)/2 = 3$. If we take the entries columnwise, then we obtain the n_C-table; and if we take the entries rowwise, we obtain the n_R-table. These two tables are shown in Table I(c) and I(d).

Now, we consider aligning the input data $X(n)$, $n = 0, 1, \cdots, 11$. Since we are taking four 3-point IDCT's first in the main body, the output data of the input mapping box are aligned in the order of $X(0, 0)$, $X(1, 0)$, $X(2, 0)$; $X(0, 1)$, $X(1, 1)$, $X(2, 1)$; $X(0, 2)$, $X(1, 2)$, $X(2, 2)$; $X(0, 3)$, $X(1, 3)$, $X(2, 3)$. So, we have to take the n_C-table in aligning the input data $X(n)$ and read out the data columnwise. As a result, we have the order $X(0)$, $X(4)$, $X(8)$; $X(3)$, $X(1)$, $X(5)$; $x(6)$, $X(2)$, $X(10)$; $X(9)$, $X(11)$, $X(7)$.

The next step is to draw flowgraphs connecting those input and output data. For each $X(n_1, n_2)$ having $n_1 = 0$ or $n_2 = 0$, we draw a horizontal unity-gain line joining it with the corresponding $X(n)$. This achieves the first part of (15). The second part of it turns out to be butterfly operations which are realized by joining $X(n_1, n_2)$ and $X(3 - n_1, 4 - n_2)$ with the corresponding two $X(n)$'s identified through the $\hat{n}$- and $\tilde{n}$-tables. For example, when $n_1 = 1$, $n_2 = 1$, we have 1 and 7 at location $(1, 1)$ of the $\hat{n}$- and $\tilde{n}$-table, respectively; and we have 1 and -7 at (3-1, 4-1) of the tables. Thus, we draw a butterfly joining $X(1, 1)$ and $X(2, 3)$ with $X(1)$ and $X(7)$. Each line in this butterfly has unity-gain except the line joining $X(7)$ with $X(2, 3)$ which has the gain -1, as is designated by the negative sign in -7. In a similar way, we can draw the second and the third butterfly operations. Due to the symmetric properties of the $\hat{n}$- and $\tilde{n}$-tables, the butterflies are dwindling with the crossing point of the first butterfly as their axes. All these are depicted in Fig. 2(a).

We now consider the output mapping. By evaluating (20a) and (20b) with $k = 0, 1, \cdots, 11$, we obtain the table in Table 1(e). Drawing the flowgraph for the output mapping is quite straightforward. Since we took the n_C-table for the output mapping, we have to take the k-table rowwise. Thus, we have the order $x(0, 0)$, $x(0, 1)$, $x(0, 2)$, $x(0, 3)$; $x(1, 0)$, $x(1, 1)$, $x(1, 2)$, $x(1, 3)$; $x(2, 0)$,

TABLE I

Index Mapping Tables for $N = 12$ with $N_1 = 3$, $N_2 = 4$. (a) $\hat{n}$-Table, (b) $\tilde{n}$-Table, (c) n_C-Table, (d) n_R-Table, (e) k-Table

$n_1 \backslash n_2$	0	1	2	3
0	0	3	6	9
1	4	7	10	-11
2	8	11	-10	-7

(a)

$n_1 \backslash n_2$	0	1	2	3
0	0	3	6	9
1	4	1	2	5
2	8	5	2	1

(b)

$n_1 \backslash n_2$	0	1	2	3
0	0	3	6	9
1	4	1	2	11
2	8	5	10	7

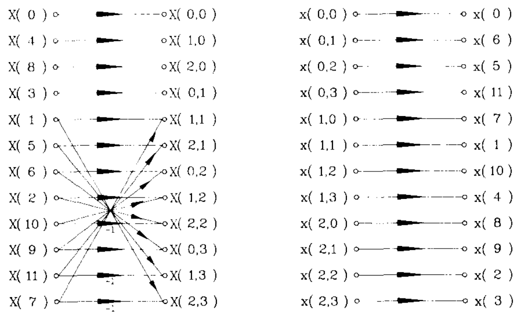
(c)

$n_1 \backslash n_1$	0	1	2	3
0	0	3	6	9
1	4	1	2	5
2	8	11	10	7

(d)

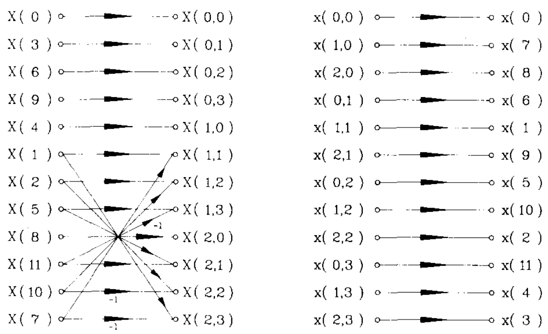
$k_1 \backslash k_2$	0	1	2	3
0	0	6	5	11
1	7	1	10	4
2	8	9	2	3

(e)



(a) (b)

Fig. 2. (a) Input mapping and (b) output mapping when 3-point IDCT's come first, followed by 4-point IDCT's.



(a) (b)

Fig. 3. (a) Input mapping and (b) output mapping when 4-point IDCT's come first, followed by 3-point IDCT's.

$x(2, 1)$, $x(2, 2)$, $x(2, 3)$ for the input data of the output mapping. And by reading out the k-table rowwise, we obtain the sequence $x(0)$, $x(6)$, $x(5)$, $x(11)$; $x(7)$, $x(1)$, $x(10)$, $x(4)$; $x(8)$, $x(9)$, $x(2)$, $x(3)$ for the output data. The remaining process is to realize (21) on these alignments, which is nothing but to join each $x(k_1, k_2)$ and the corresponding $x(k)$ with a unity-gain line. This is depicted in Fig. 2(b).

For the case when three 4-point IDCT's come first, followed by four 3-point IDCT's, one can show, by taking a similar procedure, that the resulting input and output mapping boxes are as shown in Fig. 3(a) and (b). However, care should be taken for this case such that the n_R-table be used in the input mapping and the k-table be used columnwise in the output mapping.

V. Concluding Remarks

In this paper, we have discussed how to decompose an N_1N_2-point DCT into the cascade of N_1-point DCT's and N_2-point DCT's for the case when N_1 and N_2 are relative prime to each other. We have derived and proved appropriate mappings for this decomposition. As a consequence, we can now achieve faster computation of N_1N_2-point DCT in an organized manner. Compared to the case employing brute-force techniques for the computation, the number of real multiplications reduces from $N_1^2N_2^2$ to $N_2N_1^2 + N_1N_2^2$, which amounts to an $N_1N_2/(N_1 + N_2)$-to-1 reduction. The ratio can be further improved by repeatedly applying the prime factor algorithm or by combining it with other available fast algorithms. Since with the present-day computing power the computational complexity is no longer dominated by the number of multiplications, we should examine relevant modularity, location switching, and topology issues also. In this respect, the systematic structure provided by the prime-factor algorithm of DCT can be viewed as another important advantage.

The index mappings employed for the prime-factor algorithm of DCT (PFA-DCT) differ from those for the prime-factor algorithm of DFT (PFA-DFT) in various points. For the PFA-DFT, index mappings are based on the Chinese Remainder Theorem (CRT) and the Second Integer Representation (SIR) in such a manner that if one is used for the input mapping, the other one is used for the output mapping.[10] For the PFA-DCT, however, a variation of SIR along with a variation of CRT are both employed for the input index mapping, while the output index mapping is independent of them. The variations stem from the definitions in (7a) and (7b) which are similar to, but not quite the same as, the conventional SIR and CRT. The input and the output mappings for the PFA-DFT correspond to a reordering of the related data, while the input mapping of the PFA-DCT includes data combining operations in it. This is a phenomenon already known to us for the power-of-two case (see, for example, [14]).

We have discussed tabulation techniques for the input and output index mappings, based on the equations used for the derivation of the prime-factor algorithm. Among the resulting five tables, n_C- and n_R-tables are used for input signal alignment, $\hat{n}$- and $\tilde{n}$-tables for input signal connection (butterfly operation), and the k-table for the output signal alignment and mapping. By employing these tables, we can find the necessary mappings in a straightforward manner. Furthermore, once the tables are generated, we can freely choose the DCT's to put first—either N_1-point or N_2-point. The index mapping tables are therefore expected to play a valuable role in practical applications of the prime-factor-decomposed computation of DCT.

Appendix A

Let the operation $\textcircled{U}$ denote the collection of two sets, preserving the total number of elements, such that[11]

$$\hat{\mathfrak{N}} \textcircled{U} \tilde{\mathfrak{N}} = \left\{ \hat{f}(n_1, n_2), \tilde{f}(n_1, n_2) \,\middle|\, n_1 \in \mathfrak{N}_1,\ n_1 \in \mathfrak{N}_2 \right\}. \tag{A1}$$

We want to prove in this appendix that

$$\hat{\mathfrak{N}} \textcircled{U} \tilde{\mathfrak{N}} = \mathfrak{N} \textcircled{U} \mathfrak{N}. \tag{A2}$$

[10] Refer to [23] for terminology.

[11] Notice that the operation $\textcircled{U}$ indicates the collection of two sets, which differs from the union operation U in that with $\textcircled{U}$ we list all the elements in $\hat{\mathfrak{N}}$ and $\tilde{\mathfrak{N}}$ even if some of them are identical. As a consequence, we have the relation $|\hat{\mathfrak{N}} \textcircled{U} \tilde{\mathfrak{N}}| = |\hat{\mathfrak{N}}| + |\tilde{\mathfrak{N}}|$, where $|\mathfrak{N}|$ denotes the number of elements in the set $\mathfrak{N}$.

We define set $\hat{\mathfrak{N}}'$ to be

$$\hat{\mathfrak{N}}' = \{n \mid n = n_1 N_2 + n_2 N_1, \quad n_1 \in \mathfrak{N}_1, \quad n_2 \in \mathfrak{N}_2\}, \tag{A3}$$

and we first show that

$$\mathfrak{N} \subset \hat{\mathfrak{N}}' \cup \tilde{\mathfrak{N}}. \tag{A4}$$

Since, by the SIR [23],

$$\{n \mid n = n_1 N_2 + n_2 N_1 \text{ modulo } N, \quad n_1 \in \mathfrak{N}_1, \quad n_2 \in \mathfrak{N}_2\} = \mathfrak{N}, \tag{A5}$$

it is sufficient to show that for each n in $\hat{\mathfrak{N}}'$ with $n \geq N$, we can find $n - N$ in $\tilde{\mathfrak{N}} \cap \mathfrak{N}$. Let $n' = n_1' N_2 + n_2' N_1$ be an element in $\hat{\mathfrak{N}}'$ with $n' \geq N$. We want to show that we can identify the element $n' - N$ in $\tilde{\mathfrak{N}} \cap \mathfrak{N}$. Let n'' denote the element of $\tilde{\mathfrak{N}}$ which is at $(n_1, n_2) = (n_1', N_2 - n_2')$. Then

$$\begin{aligned} n'' &= |n_1' N_2 - (N_2 - n_2') N_1| \\ &= |n_1' N_2 + n_2' N_1 - N_1 N_2| \\ &= |n' - N| \\ &= n' - N. \end{aligned} \tag{A6}$$

Thus, (A4) is verified.

Now, since $\hat{\mathfrak{N}}' \cap \mathfrak{N} \subset \hat{\mathfrak{N}}$ and $\mathfrak{N} \subset \hat{\mathfrak{N}}' \cup \tilde{\mathfrak{N}}$, we have $\mathfrak{N} \subset \hat{\mathfrak{N}} \cup \tilde{\mathfrak{N}}$. Therefore, to prove the relation in (A2), it suffices to show that for each element in $\hat{\mathfrak{N}} \textcircled{U} \tilde{\mathfrak{N}}$, we can identify another identical element in the set. But if $n_1 = 0$ or $n_2 = 0$, $\hat{f}(n_1, n_2) = \tilde{f}(n_1, n_2)$; otherwise, $\hat{f}(n_1, n_2) = \hat{f}(N_1 - n_1, N_2 - n_2)$, and $\tilde{f}(n_1, n_2) = \tilde{f}(N_1 - n_1, N_2 - n_2)$, by (8a) and (8b). Thus, the proof is complete.

Appendix B

In this appendix, we prove that g is a one-to-one mapping from $\mathfrak{N}$ to $\mathfrak{N}_1 \times \mathfrak{N}_2$.

Let $\mathfrak{N}_A$, $\mathfrak{N}_B$, $\mathfrak{N}_C$, and $\mathfrak{N}_D$ denote subsets of $\mathfrak{N}$ such that

$$\mathfrak{N}_A = \{k \mid k \in \mathfrak{N}, \bar{k}_1 < N, \quad \text{and } \bar{k}_2 < N\}, \tag{A7a}$$

$$\mathfrak{N}_B = \{k \mid k \in \mathfrak{N}, \bar{k}_1 < N, \quad \text{and } \bar{k}_2 \geq N\}, \tag{A7b}$$

$$\mathfrak{N}_C = \{k \mid k \in \mathfrak{N}, \bar{k}_1 \geq N, \quad \text{and } \bar{k}_2 < N\}, \tag{A7c}$$

$$\mathfrak{N}_D = \{k \mid k \in \mathfrak{N}, \bar{k}_1 \geq N, \quad \text{and } \bar{k}_2 \geq N\}. \tag{A7d}$$

Then (k_1, k_2) of (20) is obtained by taking $(\bar{k}_1, \bar{k}_2)$ for all k in $\mathfrak{N}_A$; $(\bar{k}_1, 2N_2 - 1 - \bar{k}_2)$ for all k in $\mathfrak{N}_B$; $(2N_1 - 1 - \bar{k}_1, \bar{k}_2)$ for all k in $\mathfrak{N}_C$; and $(2N_1 - 1 - \bar{k}_1, 2N_2 - 1 - \bar{k}_2)$ for all k in $\mathfrak{N}_D$. Also, we have the relation

$$\mathfrak{N}_A \textcircled{U} \mathfrak{N}_B \textcircled{U} \mathfrak{N}_C \textcircled{U} \mathfrak{N}_D = \mathfrak{N} \tag{A8}$$

for the operation $\textcircled{U}$ defined in Appendix A. Therefore, to prove the one-to-oneness of g, it suffices to show that no two (k_1, k_2) taken from the above four sets can be identical. We prove this by contradiction.

To begin with, we suppose that a (k_1, k_2) obtained from $\hat{k}$ in $\mathfrak{N}_A$ is identically obtained from $\tilde{k}$ in $\mathfrak{N}_B$. Then, by (20) we have

$$\hat{k} = 2a_1 N_1 + k_1, \tag{A9a}$$

$$\tilde{k} = 2a_2 N_2 + k_2, \tag{A9b}$$

and

$$\hat{k} = 2b_1 N_1 + k_1, \tag{A10a}$$

$$\tilde{k} = 2b_2 N_2 + (2N_2 - 1 - k_2), \tag{A10b}$$

where a_1, a_2, b_1, b_2 are integers. By (A9a) and (A10a), we have

$$\hat{k} - \tilde{k} = 2(a_1 - a_2) N_1, \tag{A11a}$$

and, by (A9b) and (A9b), we obtain

$$\hat{k} + \tilde{k} = 2(a_2 + b_2 + 1) N_2 - 1. \tag{A11b}$$

But it is a contradiction because by (A10a) $\hat{k}$ and $\tilde{k}$ are both even or both odd, while by (A10b) only one is even and the other one is odd.

Second, we suppose a (k_1, k_2) obtained from $\hat{k}$ in $\mathfrak{N}_A$ is identically obtained from $\tilde{k}$ in $\mathfrak{N}_D$. Then, by (20), we obtain

$$\tilde{k} = 2c_1 N + (2N_1 - 1 - k_1), \tag{A12a}$$

$$\hat{k} = 2c_2 N + (2N_2 - 1 - k_2), \tag{A12b}$$

where c_1, c_2 are integers. By (A9a) and (A12a), we have

$$\hat{k} + \tilde{k} = 2(a_1 + c_1 + 1) N_1 - 1, \tag{A13a}$$

and, by (A9b) and (A12b), we obtain

$$\hat{k} + \tilde{k} = 2(a_2 + c_2 + 1) N_2 - 1. \tag{A13b}$$

Since the smallest integer $\hat{k} + \tilde{k}$ that satisfies both (A13a) and (A13b) is $2N_1N_2 - 1$, the fact that $\hat{k}$ is in $\mathfrak{N}$ implies that $\tilde{k}$ is not in $\mathfrak{N}$, and vice versa. Therefore, the assumption leads to a contradiction.

We can show the other four cases in a similar manner, by applying the above reasonings. This proves the one-to-oneness of g.

References

[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, pp. 90–93, Jan. 1974.

[2] N. Hamidi and J. Pearl, "Comparison of the cosine and Fourier transforms of Markov-1 signals," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 428–429, Oct. 1976.

[3] R. Zelinski and P. Noll, "Adaptive transform coding of speech signals," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 299–309, Aug. 1977.

[4] J. M. Tribolet and R. E. Crochiere, "Frequency domain coding of speech," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 512–530, Oct. 1979.

[5] W. H. Chen and C. H. Smith, "Adaptive coding of monochrome and color images," *IEEE Trans. Commun.*, vol. COM-25, pp. 1285–1292, Nov. 1977.

[6] J. A. Rose and G. S. Robinson, "Interframe cosine transform image coding," *IEEE Trans. Commun.*, vol. COM-25, pp. 1329–1339, Nov. 1977.

[7] R. C. Reininger and J. D. Gibson, "Distributions of the two-dimensional DCT coefficients for images," *IEEE Trans. Commun.*, vol. COM-31, pp. 835–839, June 1983.

[8] N. B. Nill, "A visual model weighted cosine transform for image compression and quality assessment," *IEEE Trans. Commun.*, vol. COM-33, pp. 551-557, June 1985.
[9] M. J. Narasimha and A. M. Peterson, "Design of a 24-channel transmultiplexer," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 752-762, Dec. 1979.
[10] M. J. Narasimha, P. P. N. Yang, B. G. Lee, and M. L. Abell, "The TM-7800-M1: A 60-channel CCITT transmultiplexer," in *Proc. IEEE Int. Conf. Commun.*, May 1984.
[11] W. H. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-25, pp. 1004-1009, Sept. 1977.
[12] M. J. Narasimha and A. M. Peterson, "On the computation of discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-26, pp. 934-936, June 1978.
[13] J. Makhoul, "A fast cosine transform in one and two dimensions," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 27-34, Feb. 1980.
[14] B. G. Lee, "A new algorithm to compute the discrete cosine transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 1243-1245, Dec. 1984.
[15] Z. Wang, "Fast algorithms for the discrete W transform and for the discrete Fourier transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 803-816, Aug. 1984.
[16] H. V. Sorensen, D. L. Jones, M. T. Heideman, and C. S. Burrus, "Real-valued fast Fourier transform algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 849-863, June 1987.
[17] H. S. Malvar, "Fast computation of the discrete cosine transform and the discrete Hartley transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 1484-1485, Oct. 1987.
[18] H. S. Hou, "A fast recursive algorithm for computing the discrete cosine transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 1455-1461, Oct. 1987.
[19] P. P. N. Yang, M. J. Narasimha, and B. G. Lee, "A prime factor decomposition algorithm for the computation of discrete cosine transform," in *Proc. IEEE Int. Conf. Comput., Syst., Signal Processing*, Dec. 1984.
[20] C. S. Burrus, "Index mappings for multidimensional formulation of the DFT and convolutions," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 239-242, June 1977.
[21] C. S. Burrus and P. W. Eschenbacher, "An in-place in-order prime factor FFT algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 806-817, Aug. 1981.
[22] S. Chu and C. S. Burrus, "A prime-factor algorithm using distributed arithmetics," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, pp. 217-227, Apr. 1982.
[23] D. F. Elliott and K. R. Rao, *Fast Transforms: Algorithms, Analyses, Applications*. New York: Academic, 1982.

Byeong Gi Lee (S'80-M'82) was born in Daechon, Korea, on May 12, 1951. He received the B.S. and M.E. degrees in 1974 and 1978, respectively, from Seoul National University, Seoul, and Kyungpook National University, Taegu, Korea, both in electronics engineering; and the Ph.D. degree in 1982 from the University of California, Los Angeles, in electrical engineering.

From 1974 to 1979 he was with the Department of Electronics Engineering of ROK Naval Academy, Chinhae, Korea, as an Instructor and Naval Officer in active service. From 1982 to 1984 he worked for Granger Associates, Santa Clara, CA, as a Senior Engineer doing research and development on applications of digital signal processing to digital transmission. During the period 1984 to 1986 he worked for AT&T Bell Laboratories, North Andover, MA, as a member of the Technical Staff participating in lightwave transmission system development along with related standard works. Since September 1986 he has been with the Department of Electronics Engineering, Seoul National University, Seoul, Korea. His current fields of interest include theory and applications of digital signal processing, digital transmission and lightwave transmission systems, and circuit theory. He is the author of *Electronics Engineering Experiment Series* (5 volumes, all in Korean) and holds four U.S. patents.

Dr. Lee received the 1984 Myril B. Reed Best Paper Award from the Midwest Symposium on Circuits and Systems, and exceptional contribution awards from AT&T Bell Laboratories. He is a member of Sigma Xi.