論 文

DCT, DST와 DHT의 高速 알고리즘 開發

正會員 朴 鍾 演*

The Development of the Fast Algorithms for the DCT, DST and DHT

Chong Yeun PARK* *Regular Member*

要 約 DCT-Ⅲ型의 高速處理方法을 다른 型의 DCT, DST 및 DHT의 高速變換에 확대 適用하여 시스템의 구성이 간단하고 곱셈및 덧셈 回數가 기존의 方法보다 작은 高速 알고리즘을 開發하였다. 또한 各各 四種의 DCT, DST및 DHT의 相互 관계식과 各各에 곱셈回數및 덧셈回數에 관한 公式이 유도되었다.

ABSTRACT By the extension and the modification of the efficient aigorithms for the DCT-Ⅲ, the fast algorithms to compute three versions for DCT, four versions of DST, and a DHT are developed., It is shown that the algorithms developed in this paper have simple structures and the numbers of multipication and addition are reduced as compaired with the existing efficient algorithms. The algorithms presented in this paper indicate the close reia tionship among different versions of the DCT and DST as well as a DHT. The formulas to compute the num bers of multiplication and addition of them are derived.

I. INTRODUCTION

In recent years, there has been an increasing interest with respect to using a class of orthogonal transforms in general area of digital signal processing. Although the Karhunen-Loeve Transform(KLT) is optimal in various orthogonal transforms, there is no general algorithm that enables its fast computation[1]. This correspondence treats of the algorithms that enable the fast computation of the Discrete Cosine Transform (DCT), Discrete Sine Transform(DST), and Discrete Hartley Transform(DHT).

The DCT has been successfully applied to the coding of high resolution imaginary[2]. The use of the DCT in a wide variety of applications has not been as extensive as its properties that

* 江原大學校 電氣工學科
Dept. of Electrical Engineering, Kangweon Nat. Univ., 200 Korea.
論文番號 : 87-25(接受 1987. 3. 20)

would imply due to the lack of an efficient algorithm. The first method of implementing the DCT utilized a double size Fast Fourier Transform(FFT) algorithm employing complex arithmetic throughout the computation[1]. A more efficient algorithm involving only real operations for computing the fast DCT was found[3], [4]. The most efficient algorithm of the DCT was proposed by the Lee[5], the number of real multiplications appears to be $(N/2)\log_2(N)$ for a N-points DCT with $N=2^m$. But his fast algorithm was applied to only two versions of four types of DCT which were introduced by the Wang[6]. Wang has shown that the fast algorithms for all versions of the DCT and DST depend only on a fast algorithms for DCT-IV (One of the four versions of DCT defined by Wang[6]). More recently, even better efficient algorithms[7], [8] were found. In this paper another fast algorithms were developed for the all versions of DCT by the extention of the methods that Lee[2] proposed. This fast algorithms were compaired with the existing efficient algorithms.

Jain[9] has shown that for a first-order Markov sequence with certain boundary conditions, KLT reduces to the DST-I or DST-IV. Subsequently DST-II and DST-III were redefined by Krekre and Solanki[10]. Yip and Rao[11] have proven that for large sequence($n \geq 32$)and low correlation coefficient($\rho \leq 0.6$)DST performs even better than the DCT-I, which is widely accepted as the best substitute for the KLT. They have developed a fast algorithm for DST-I, which is similar to fast DCT and Wang[12] has found the fast algorithms for DCT-II and DCT-III. Recently a systematic method of the spare matrix factorization for all versions of DST was found[16]. In this correspondence another methods for the fast computation of four versions of DST were discussed and compaired with the existing efficient algorithms. One of this object is to find the most efficient algorithm for DST by the extention of the algorithms for DCT.

A sequence of N real numbers possesses a DHT that is a sequence of the same length and is also real valued. One such transform closely related to the DFT is the DHT[13]. Since DHT is the transform that directly maps a real valued sequence to a real valued spectrum, it is more convient than the DFT in application to numerical spectral analysis. For the fast algorithm of DHT, Bracewell[14] has recently presented a radix-2 decimation in time fast DHT algorithm, demonstraing that the DHT gives rise to at least one fast algorithm that exist for the DFT. R. Ansari[15] defined the Discrete Combination Fourier Transform(DCFT) of the sequence x(n) and showed that DHT is the special case of DCFT. Soresen and et.[16] showed that the philosophy of all the common FFT algorithm can also be applied to the computation of DHT. In this paper, the another fast algorithm for computation of DHT is discussed with the most efficient algorithm for DCT and DST. One of the objects of this report is to develop the fast algorithms for computing DHT. We assume that data sequence length is $n=2^m$.

II. DERIVATION OF FAST COMPUTATIONAL ALGORITHMS FOR DCT

Four versions of DCT are introduced and their fast algorithms are derived respectively. Since the versions of DCT-II and DCT-III are the most important of all versions for DCT, at first the fast algorithm for the DCT-II and DCT-III are discussed, and then the version of DCT-I and DCT-IV discussed.

II-1. Fast Algorithm for DCT-III

The version of DCT-III for the data sequence $\hat{X}(n)$(for n=0, 1 ... N-1) is defined[1] by

$$x(k)=\sum_{n=0}^{N-1} e(n)\cdot \hat{X}(n)\cdot\cos[\pi(k+\frac{1}{2})n/N]\ :$$

$$0\leqq k\leqq N-1 \qquad (1)$$

where

$$e(n)=\begin{cases} 1/\sqrt{2} : n=0 \\ 1 : n\neq 0 \end{cases}$$

If we denote C_y^x and S_y^x as

$$\left.\begin{aligned} C_y^x &= \cos(\pi x/y) \\ S_y^x &= \sin(\pi x/y) \end{aligned}\right\} \qquad (2)$$

then equation (1) is

$$x(k)=X(n)\,C_N^{(k+\frac{1}{2})n} \ : \ 0\leqq k\leqq N-1 \qquad (3)$$

where $X(n)=e(n)\cdot\hat{X}(n)$

The efficient algorithm for the fast computation of (3) was proposed by Lee[5] and the following equations (4)-(6) were given by him. Let us take G(n) and H(n)

$$\left.\begin{aligned} G(n) &= X(2n) \\ H(n) &= X(2n+1)+X(2n-1) \end{aligned}\right\} \ 0\leqq k\leqq \frac{N}{2}-1 \qquad (4)$$

If we denote the version of DCT-III for G(n) and H(n) by g(k) and h(k) respectively, from (3) we obtain

$$\left.\begin{aligned} g(k) &= \sum_{n=0}^{N/2-1} G(n)\,C_{N/2}^{(k+\frac{1}{2})n} \\ h(k) &= \sum_{n=0}^{N/2-1} H(n)\,C_{N/2}^{(k+\frac{1}{2})n} \end{aligned}\right\} \ 0\leqq k\leqq \frac{N}{2}-1 \qquad (5)$$

Now from (3) and (5) We can obtain the following equations

$$\left.\begin{aligned} x(k) &= g(k)+h(k)/(2C_N^{k+\frac{1}{2}}) \\ x(N-1-k) &= g(k)-h(K)/(2C_N^{k+\frac{1}{2}}) \end{aligned}\right\} \qquad 0\leqq k\leqq \frac{N}{2}-1 \qquad (6)$$

Therefore we have decomposed the N-points DCT-III in (3) into the sum of two $\frac{N}{2}$ -points DCT-III in (5). By repeating this process, we can decompose the DCT-III further. In this correspondence, above procedures are extended for all versions of DCT, DST, and a DHT. For example the versions of DCT-III for N=4-points is represented by fig. 1.

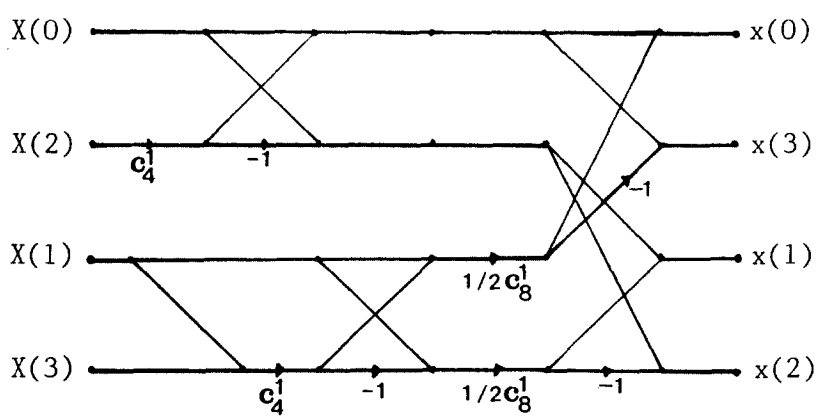


Fig. 1 Signal Flow graph for fast DCT-III (N=4-points).

If $\mu[c_N^{III}]$ and $\alpha[c_N^{III}]$ are defined as the numbers of multiplications and additions required to implement a N-points DCT-III respectively; then it can be shown that

$$\left.\begin{aligned}\mu[C_N^{III}] &= \frac{N}{2}\log_2(N)\\ \alpha[C_N^{III}] &= \frac{3}{2}N\cdot\log_2 N - N + 1\end{aligned}\right\} \quad (7)$$

II-2. Fast Algorithms for DCT-II

The version of DCT-II for data sequence x(k) (for k=0,1 ...N-1) is defined[1]

$$\hat{X}(n) = \frac{2}{N}e(n)\sum_{k=0}^{N-1} x(k)\cdot C_N^{(2k+1)n} : 0 \leqq n \leqq N-1 \quad (8)$$

So we have

$$\hat{X}(n) = \sum_{n=0}^{N-1} x(k)\cdot C_N^{(2k+1)n} : 0 \leqq n \leqq N-1 \quad (9)$$

Now (9) for the DCT-II can be obtained by transposing the (3), i.e. transposing the direction of the arrows in the flow graph of DCT-III since DCT-II is an orthogonal transform. For example the fast algorithm of 4-points DCT-II is obtained by reversing the direction of arrows in Fig. 1.

II-3. Fast Algorithms for DCT-I

The DCT-I was defined for the order N+1 and it was introduced into digital signal processing by Wang[6] We denote the DCT-I of data sequence $\hat{X}(n)$ (for n=0,1 ... N) by $X(n)$ (for n=0,1 ... N). Then we have

$$\hat{x}(k) = \sqrt{\frac{2}{N}}e(k)\sum_{n=0}^{N} e(n)\cdot\hat{X}(n)\,C_N^{kn} \quad (10)$$

Where

$$e(j) = \begin{cases} 1 & : j = 0 \text{ and } j = N \\ 1/\sqrt{2} & : j \neq 0 \text{ and } j \neq N \end{cases}$$

If we assume the notation of

$$\left.\begin{aligned} x(k) &= \frac{\hat{x}(k)}{e(k)}\sqrt{\frac{N}{2}}\\ X(n) &= e(n)\hat{X}(n)\end{aligned}\right\} \quad (11)$$

then (10) is rewritten as

$$x(k) = \sum_{n=0}^{N} X(n)\,C_N^{kn} : 0 \leqq k \leqq N \quad (12)$$

In this correspondence, the fast algorithm of computing the x(k) in (12) is developed. We rewrite the (12)

$$x(k) = \sum_{n=0}^{N/2} X(2n)\,C_{N/2}^{nk} + \sum_{n=0}^{\frac{N}{2}-1} X(2n+1)\,C_N^{(n+\frac{1}{2})k} : 0 \leqq k \leqq \frac{N}{2} \quad (13)$$

If we assume the following notations

$$g(k) = \sum_{n=0}^{N/2} X(2n)\,C_{N/2}^{nk} : 0 \leqq k \leqq \frac{N}{2} \quad (14)$$

$$h(k) = \sum_{n=0}^{\frac{N}{2}-1} X(2n+1)\,C^{(n+\frac{1}{2})k} : 0 \leqq k \leqq \frac{N}{2}-1 \quad (15)$$

then the important relationship is obtained.

$$x(k) = g(k) + h(k) \quad (16)$$

$$x\left(\frac{N}{2}\right) = g\left(\frac{N}{2}\right) = \sum_{n=0}^{N/2} X(2n)\cdot(-1)^n \quad (17)$$

$$x(N-k) = g(k) - h(k) \quad (18)$$

Therefore we have decomposed the N-points DCT-I in (12) into sum of $\frac{N}{2}+1$ points DCT-I and $\frac{N}{2}$-points DCT-II in (14) and (15) respectively. By repeating this process we can decompose the DCT-I further. For example, consider N(=8)+1 points DCT-I. Now from the (14) and (15), the following equations are valid.

$$g(k)=\sum_{n=0}^{4} X(2n)\,C_4^{nk}$$

$$=X(0)+X(4)C_2^k+(-1)^k\cdot x(8)$$

$$+C_4^k[X(2)+(-1)^k\,x(6)]\ :\ 0\leqq k\leqq 4 \quad (19)$$

$$h(k)=\sum_{n=0}^{3} X(2n+1)\,C_4^{k(n+\frac{1}{2})}\ :\ 0\leqq k\leqq 3 \quad (20)$$

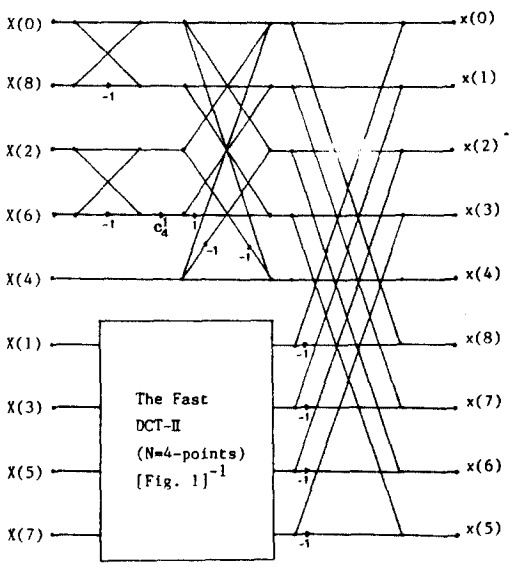


Fig. 2 Signal Flow graph for fast DCT-I (N=8-points).

In order to compute the value of g(k) in (19) only one number of multiplication is required, since C_2^k has the one of 0, 1, and -1, and the numbers of additions are 11. Because the h(k) in (20) is the 4-points DCT-II, Fig. 1 can be used in the Fig. 2. which is the flow graph of the 8-points fast DCT-I.

Since $\mu[C_{N+1}^I]$ and $\alpha[C_{N+1}^I]$ are very important factors for the fast algorithm of DCT-I, we have to compute them of (12). If we assume the data length $N=2^m$, then the following formulas are valid from (7)

$$\mu[C_{N+1}^I]=\frac{N}{2}\log_2^N-N+1 \quad (21)$$

$$\alpha[C_{N+1}^I]=(\frac{3}{2}N+1)\log_2^N-3N+13 \quad (22)$$

Now the present algorithm is compaired with the existing efficient algorithm by the Table 1.

II-4. Fast Algorithm for DCT-IV

The version of DCT-IV data sequence $\hat{X}(n)$ (for n=0,1 ... N-1) is defined [4] as

Table 1 Comparison of DCT-I Algorithms.

N	Multiplications			Additions		
	Present Method	References		Present Method	References	
		(6)	(8)		(6)	(8)
5	1	5	1	15	10	15
9	5	9	5	28	27	28
17	17	21	17	65	72	65
33	49	55	49	162	187	162
65	129	145	129	403	470	403
129	321	371	321	980	1145	980
257	769	917	769	2325	2716	2325
513	1793	2199	1793	5398	6303	5398
1025	4097	5145	4097	12311	14370	12311
2049	9217	11803	9217	27672	32293	27672
4097	20481	26653	20481	61465	71720	61465

$$x(k)=\sum_{n=0}^{N-1} e(k)\,\hat{X}(n)C_N^{(k+\frac{1}{2})(n+\frac{1}{2})} \;:\; 0\leqq k\leqq N-1 \tag{23}$$

We rewrite x(k) in (23)

$$x(k)=\sum_{n=0}^{N-1} X(n)\,C_N^{(k+\frac{1}{2})(n+\frac{1}{2})} \;:\; 0\leqq k\leqq N-1 \tag{24}$$

Where $X(n)=e(n)\cdot X(n)$

In this correspondence, the fast algorithm computing x(k) in (24) is discussed. From (24), we obtain that

$$x(k)\cdot 2\,C_N^{(k+\frac{1}{2})\frac{1}{2}}=\sum_{n=0}^{N-1}[X(n-1)+X(n)]C_N^{(k+\frac{1}{2})n} \tag{25}$$

Where X(-1)=0

If we assume G(n)=X(n-1)+X(n), then

$$x(k)=g(k)/\,2\,C_N^{(k+\frac{1}{2})\frac{1}{2}} \;:\; 0\leqq k\leqq N-1 \tag{26}$$

$$\text{Where}\quad g(k)=\sum_{n=0}^{N-1} G(n)\cdot C_N^{(k+\frac{1}{2})n} \tag{27}$$

Since the values of g(k) in (27) are computed by the fast algorithm for DCT-III, x(k) in (26) is obtained easily. For example, the fast computation of a 8-points DCT-IX is represented by Fig. 3.

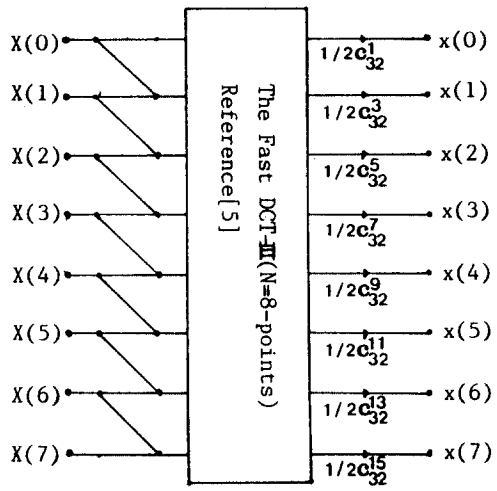


Fig. 3 Signal Flow graph for fast DCT-IV (N=8-points).

We can find the values of $\mu[C_N^N]$ and $\alpha[C_N^N]$ easily with

$$\mu[C_N^N]=\mu[C_N^{III}]+N=\frac{N}{2}\log_2(N)+N \tag{28}$$

Table 2 Comparison of DCT-IV Algorithms.

N	Multiplications			Additions		
	Present Method	References		Present Method	References	
		(6)	(8)		(6)	(8)
4	8	8	8	12	12	16
8	20	22	20	36	38	44
16	48	56	48	96	104	112
32	112	136	112	240	264	272
64	256	320	256	576	640	640
128	576	736	576	1344	1504	1472
256	1280	1664	1280	3072	3456	3328
512	2816	3712	2816	6912	7808	7424
1024	6144	8192	6144	15360	17408	16384
2048	13312	17920	13312	33792	38400	35840
4096	28672	38912	28672	73728	83968	77824

$$\alpha[C_N^N] = \alpha[C_N^{III}] + N - 1 = \frac{3}{2}N \cdot \log_2(N) \quad (29)$$

$\mu[C_N^N]$ and $\alpha[C_N^N]$ are identical with the results of the most efficient algorithm given by Suehiro and Hatori[8] for DCT-IV. Therefore $\mu[C_N^N]$ and $\alpha[C_N^N]$ in (28) and (29) respectively are compared with existing efficient algorithms in Table II.

III. FAST ALGORITHMS FOR DST

All versions of DST are introduced and their fast algorithms are derived. Since the version of DST-IV becomes the basis for all versions of DST, at first the fast algorithms for DST-IV are discussed and then the others are derived.

III-1. Fast Algorithms for DST-IV

The DST-IV was introduced by Jain[9] and was studied by Wang[6]. If we define $S_y^x = \sin(\pi x/y)$, the version of DST-IV for data sequence $\hat{X}(n)$ (for n=0,1...N-1) is defined as

$$x(k) = \sqrt{\frac{2}{N}} \sum_{n=0}^{N-1} X(n) \cdot S_N^{(n+\frac{1}{2})(k+\frac{1}{2})} \quad : \ 0 \leqq k \leqq N-1 \quad (30)$$

So we defined $X(n) = \sqrt{\frac{2}{N}} \cdot \hat{X}(n)$ then x(k) in (30) is

$$x(k) = \sum_{n=0}^{N-1} X(n) \ S_N^{(n+\frac{1}{2})(k+\frac{1}{2})} \quad : \ 0 \leqq k \leqq N-1 \quad (31)$$

The fast algorithm computing x(k) in (31) is discussed in this correspondence. By appendix, we can rewrite x(k) in (31) and derive the x(N-1-k).

$$x(k) = [-a(k)+b(k)] \,/\, [2\,S_{2N}^{(k+\frac{1}{2})}] \quad : \ 0 \leqq k \leqq \frac{N}{2} - 1 \quad (32)$$

$$x(N-1-k) = [a(k)+b(k)] \,/\, [2\,C_{2N}^{k+\frac{1}{2}}] \quad : \ 0 \leqq k \leqq \frac{N}{2} - 1 \quad (33)$$

Where

$$a(k) = \sum_{n=0}^{N/2-1} [X(2n) - X(2n+1)]\, C_{N/2^2}^{(k+\frac{1}{2})(n+\frac{1}{2})} \quad (34)$$

$$b(k) = \sum_{n=0}^{N/2-1} [X(2n) - X(2n-1)]\, C_{N/2}^{n(k+\frac{1}{2})} \quad (35)$$

Therefore we can compute the N-points DST-IV x(k) in (31) by the use of the fast algorithms for DST-II, because a(k) in (34), b(k) in (35) are obtained by the fast algorithm of DCT-IV and DCT-II respectively. For example, N=4-points DST-IV

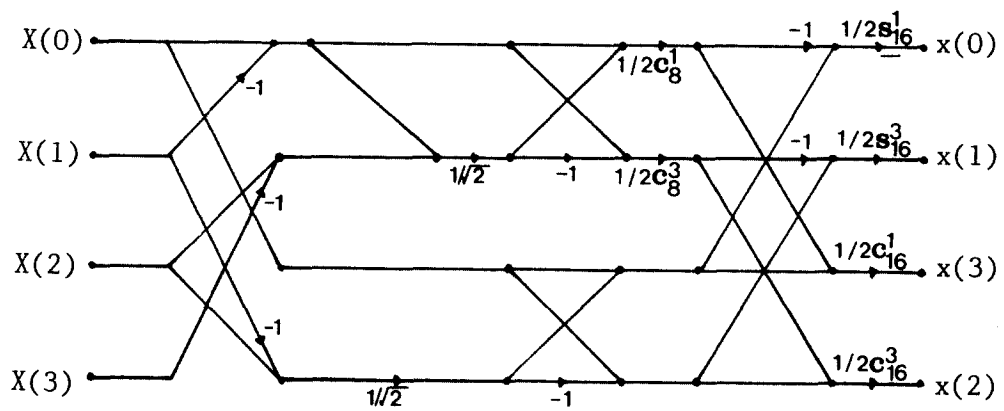


Fig. 4 Signal Flow graph for fast DST-IV (N=4-points).

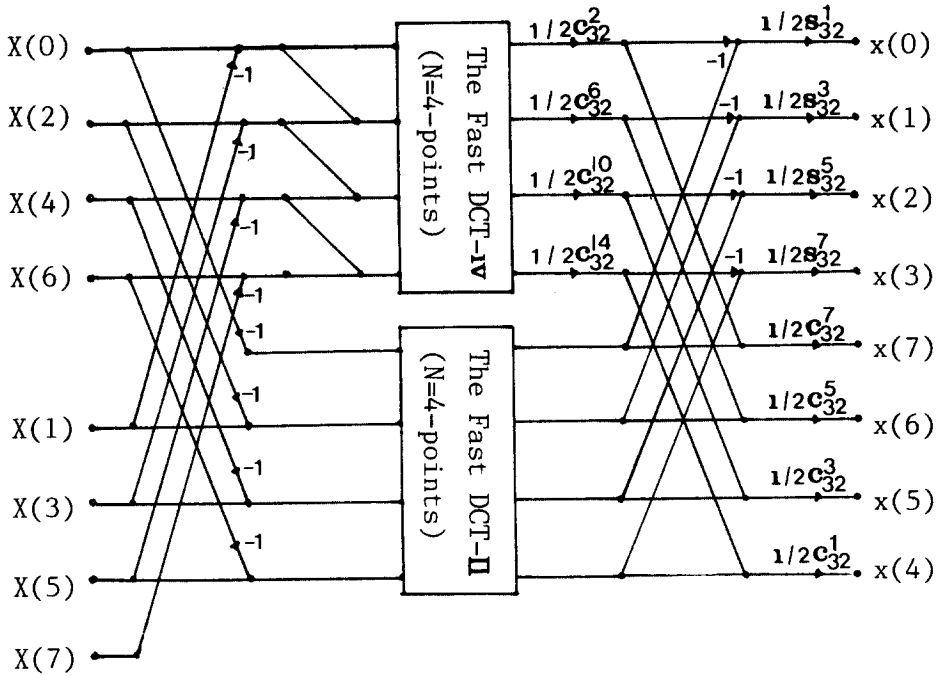


Fig. 5 Signal Flow graph for fast DST-IV (N=8-points).

and N=8-points DST-IV are represented by Fig. 4 and Fig. 5 respectively.

Therefore we can also compute $\mu[S_N^N]$ and $\alpha[S_N^N]$ by the use of (7)

$$\mu[S_N] = 2\cdot\mu[C_{N/2}^{III}] + \frac{3}{2}N = \frac{N}{2}\log_2(N) + N$$

$$= \mu[C_N^{IV}] \tag{36}$$

$$\alpha[S_N] = 2\cdot\alpha[C_{N/2}^{III}] + \frac{5}{2}N - 2 = \frac{3}{2}N\cdot\log_2(N)$$

$$= \mu[C_N^{IV}] \tag{37}$$

Since the facts that $\mu[S_N^N]=\mu[C_N^N]$ and $\alpha[S_N^N]=\alpha[C_N^N]$ were proved by Wang[6] in another method, the values of Table II are also valid with DST-IV.

III-2. Fast Algorithms for DST-III

The DST-III was introduced by Kekre and Solanki[10] and was defined by Wang[6]. The version of DST-III for data sequence $\hat{X}(n)$ (for n=1,2...N)

$$x(k) = \sum_{n=1}^{N} e(n)\cdot\hat{X}(n)\,S_N^{(k-\frac{1}{2})n}: \ 1 \leqq k \leqq N \tag{38}$$

Where

$$e(n) = \begin{cases} \sqrt{\frac{2}{N}} : n \neq 0 \\ \sqrt{\frac{1}{N}} : n = 0 \end{cases}$$

So we defined $X(n)=e(n)\cdot\hat{X}(n)$, then x(k) in (38) ix

$$x(k) = \sum_{n=1}^{N} X(n)\cdot S_N^{(k+\frac{1}{2})n} \ : \ 0 \leqq k \leqq N-1 \tag{39}$$

In this correspondence, the fast algorithm computing X(k) in (39) is developed. We rewrite the (39) as

$$x(k)=g(k)+h(k)\ :\ 0\leqq k\leqq N-1 \tag{40}$$

Where

$$g(k)=\sum_{n=0}^{\frac{N}{2}-1} X(2n+1)\,S_{N/2}^{(k+\frac{1}{2})(n+\frac{1}{2})} \tag{41}$$

$$h(k)=\sum_{n=0}^{\frac{N}{2}-1} X(2n)\cdot S_{N/2}^{(k+\frac{1}{2})n} \tag{42}$$

And x(N-1-k) is obtained with the g(k) and h(k) in (41) and (42) respectively.

$$x(N-1-k)=-g(k)+h(k) \tag{43}$$

In order to compute x(k) in (39) fast, we have to compute g(k) in (41) and h(k) in (42) by means of the fast algorithm for $\frac{N}{2}$ -points DST-IV and DST-III respectively. By repeating this process, we can obtain the value of x(k) in (39) fast. For example, the fast algorithms for N=4-points and N=8-points DST-III are represented by Fig. 6 and Fig. 7 respectively.

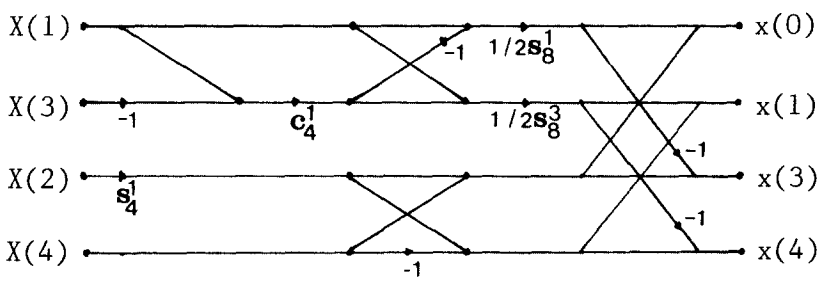


Fig. 6 Signal Flow graph for fast DST-III (N=4-points).

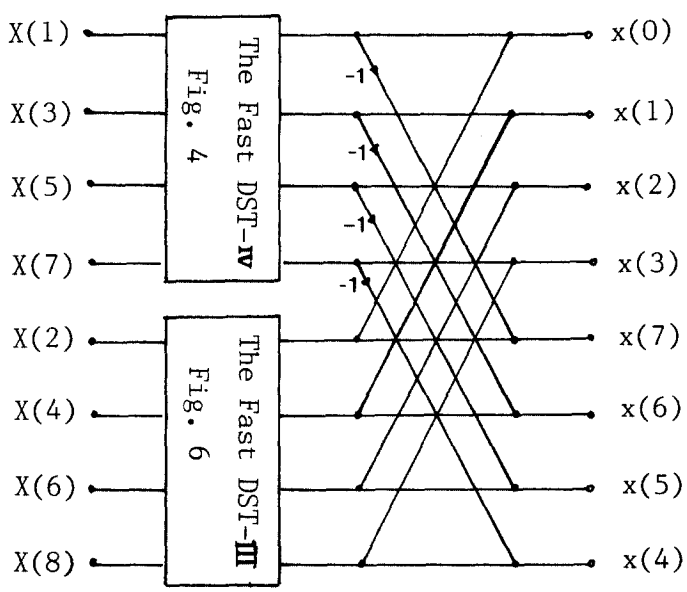


Fig. 7 Signal Flow graph for fast DST-III (N=8-points).

Therefore we can also compute $\mu[S_N^{III}]$ and $\alpha[S_N^{III}]$ by use of $\mu[S_N^{IV}]$ in (36) and $\alpha[S_N^{IV}]$ in (37).

$$\mu[S_N^{IV}] = \mu[S_{N/2}^{IV}] + \mu[S_{N/4}^{IV}] + \cdots + \mu[S_4^{IV}] + \mu[S_4^{III}] = \frac{N}{2} \cdot \log_2(N) = \mu[C_N^{III}] \quad : n \geq 4 \tag{44}$$

$$\alpha[S] = \alpha[S_{N/2}^{IV}] + \alpha[S_{N/4}^{IV}] + \cdots + \alpha[S_4^{IV}] + \alpha[S_4^{III}] + N = \frac{3}{2} N \cdot \log_2(N) - 2N + 9 \quad : n \geq 8 \tag{45}$$

It is pointed out that the values of $\alpha[S_N^{III}]$ is less than the values of $\alpha[C_N^{III}]$ although Wang[6] and Suehiro and Matori [8] showed that $\alpha[S_N^{III}]$ equals to $\alpha[C_N^{III}]$ In Table III, the present algorithm is compaired with the existing efficient algorithms.

III-3. Fast Algorithms for DST-II

The DCT-III was introduced by Kekre and Solanki[10] and was defined by Wang[6],[11]. The definition of DST-II for data sequence x(k) (for k=1,2....N)

$$\hat{X}(n) = \frac{2}{N} e(n) \sum_{k=1}^{N} x(k) \cdot S_N^{(k-\frac{1}{2})n} \quad : 1 \leq n \leq N \tag{46}$$

From the (46), We can get the following equation which is regarded with the version of DST-III and its the fast computational method is discussed.

$$X(n) = \sum_{k=0}^{N-1} x(k) \; S_N^{(k+\frac{1}{2})n} \quad : 0 \leq n \leq N-1 \tag{47}$$

Where $X(n) = N \cdot \hat{X}(n) / (2e(n))$

The DST-II can be obtained by transposing the DCT-III. For example the fast algorithms for N=4-points and N=8-points DST-II are represented by reversing the direction of the arrows in Fig. 6 and Fig. 7 respectively. Also $\mu[S_N^{II}]$ and $\alpha[S_N^{II}]$ are equal to $\mu[S_N^{III}]$ and $\alpha[S_N^{III}]$ respectively. Therefore the numbers in Table III are also valid for DCT-II.

III-4. Fast Algorithms for DST-I

The DST-I was defined by Jain[9] and was introduced by Wang[6] and Yip and rao[12]

Table 3 Comparison of DST-III Algorithms.

N	Multiplications			Additions		
	Present Method	References		Present Method	References	
		(6)	(8)		(6)	(8)
4	4	5	5	13	9	9
8	12	13	13	29	29	29
16	32	35	33	73	83	81
32	80	91	81	185	219	209
64	192	227	193	457	547	513
128	448	547	449	1097	1315	1217
256	1024	1283	1025	2569	3075	2817
512	2304	2947	2305	5897	7043	6401
1024	5120	6659	5121	13321	15875	14337
2048	11264	14851	11265	29705	35331	31745
4096	24576	32771	24577	65545	77287	69633

The version of DST-I for data sequence X(n) (for n=1,2...N-1)

$$x(k)=\sqrt{\frac{2}{N}}\sum_{n=1}^{N-1}\hat{X}(n)\,S_N^{kn}:\ 1\leqq k\leqq N-1 \quad (48)$$

If we take $X(n)=X(n)\sqrt{\frac{2}{N}}$, he version of DST-I is

$$x(k)=\sum_{n=1}^{N-1}X(n)\,S_N^{kn}\ :\ 1\leqq k\leqq N-1 \quad (49)$$

The fast algorithm computing x(k) in (49) is developed in this correspondence and the following equations are obtained by manipulating (49).

$$x(N/2)=g(N/2) \quad (50)$$

$$x(k)=g(k)+h(k):\ 0\leqq k\leqq\frac{N}{2}-1 \quad (51)$$

$$x(N-k)=-g(k)+h(k):\ 0\leqq k\leqq\frac{N}{2}-1 \quad (52)$$

Where

$$g(k)=\sum_{n=0}^{\frac{N}{2}-1}X(2n+1)\,S_{N/2}^{(n+\frac{1}{2})k}\ :\ 0\leqq k\leqq\frac{N}{2} \quad (53)$$

$$h(k)=\sum_{n=0}^{\frac{N}{2}-1}X(2n)\,S_{N/2}^{nk}\ :\ 0\leqq k\leqq\frac{N}{2}-1 \quad (54)$$

To find the fast algorithm of DST-I, we have to decompose h(k) in (54) by repeating process and g(k) in (53) by the fast algorithm for DST-II. For example, the fast algorithm computing the 7-points DST-I is represented by Fig. 8. Therefore $\mu[S_{N-1}^I]$ and $\alpha[S_{N-1}^I]$ are also given with the following equations.

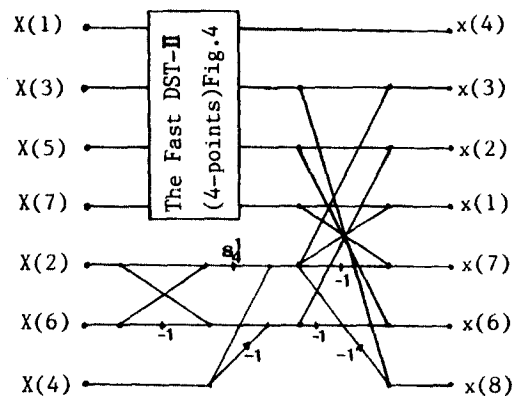


Fig. 8 Signal Flow graph for fast DST-I (N=8-points)

Table 4 Comparison of DST-II Algorithms.

N	Multiplications			Additions		
	Present Method	References		Present Method	References	
		(12)	(6)		(12)	(6)
3	1	1	2	6	4	4
7	5	5	8	23	19	22
15	17	17	30	60	62	62
31	49	51	54	149	175	166
63	129	141	130	366	456	422
127	321	367	310	887	1129	1030
255	769	913	730	2112	2698	2438
512	1793	2195	?	4937	6283	?
1023	4097	5141	?	11346	14348	?
2047	9127	11799	?	25691	32269	?
4097	20481	26649	?	57444	71694	?

$$\mu[S^{I}_{N-1}]=\mu[S^{II}_{N/2}]+\mu[S^{III}_{N/2-1}]=\mu[S^{II}_{N/2}]$$
$$+\mu[S^{II}_{N/4}]+\cdots+\mu[S^{II}_{4}]+\mu[S^{III}_{3}]$$
$$=\frac{N}{2}\log_2(N)-N+1 \quad (55)$$

$$\alpha[S^{I}_{N-1}]=\alpha[S^{II}_{\frac{N}{2}-1}]+N-2+\alpha[S^{II}_{\frac{N}{2}}]$$
$$=\alpha[S^{II}_{\frac{N}{2}}]+\alpha[S^{II}_{\frac{N}{4}}]+\cdots+\alpha[S^{II}_{4}]+4+N-2$$
$$=(\frac{N}{2}+3)\cdot 3\cdot\log_2(N)-4N-8 \quad (56)$$

And the present method is compaired with the existing efficient algorithms in Table IV.

IV. FAST ALGORITHM FOR DHT

The DHT was defined by Bracewell [13], and its fast computational algorithms were developed by Bracewell [14], Ansari [15], and H.V. Soresen and et [16]. The DHT pair is defined for a real valued length-N sequence X(n) (for n=0,1... N-1) by the following equations.

$$x(k)=\sum_{n=0}^{N-1}X(n)[C_N^{2kn}+S_N^{2kn}]:\ 0\leqq k\leqq N-1 \quad (57)$$

$$X(n)=\frac{1}{N}\sum_{k=0}^{N-1}x(n)[C_N^{2kn}+S_N^{2kn}]:\ 0\leqq k\leqq N-1 \quad (58)$$

In this correspondence, we derive the fast algorithm computing x(k) in (57). We rewrite x(k) in (57)

$$x(k)=g(k)+h(k):\ 0\leqq k\leqq\frac{N}{2}-1 \quad (59)$$

Where

$$g(k)=\sum_{n=0}^{N/2}G(n)C_{N/2}^{kn}:\ 0\leqq k\leqq\frac{N}{2} \quad (60)$$

Where $G(n)=X(n)-X(N-n)$

$G(\frac{N}{2})=X(\frac{N}{2})$

$X(N)=0$

$$h(k)=\sum_{n=1}^{\frac{N}{2}-1}H(n)\cdot S_{N/2}^{kn}:\ 1\leqq k\leqq\frac{N}{2}-1 \quad (61)$$

Where $H(n)=X(n)-X(N-n)$

$h(0)=0$

And we can obtain the following equations by applying the properties of trigonometric functions.

$$x(N-k)=g(k)-h(k):\ 1\leqq k\leqq\frac{N}{2} \quad (62)$$

Where $h(\frac{N}{2})=0$

Therefore, we have decomposed the N-points DHT in (57) into the sum of $(\frac{N}{2}+1)$-points DCT-I g(K) and $(\frac{N}{2}-1)$-points DST-I h(k). By the use of the fast algorithm for DST-I and DCT-I, we can

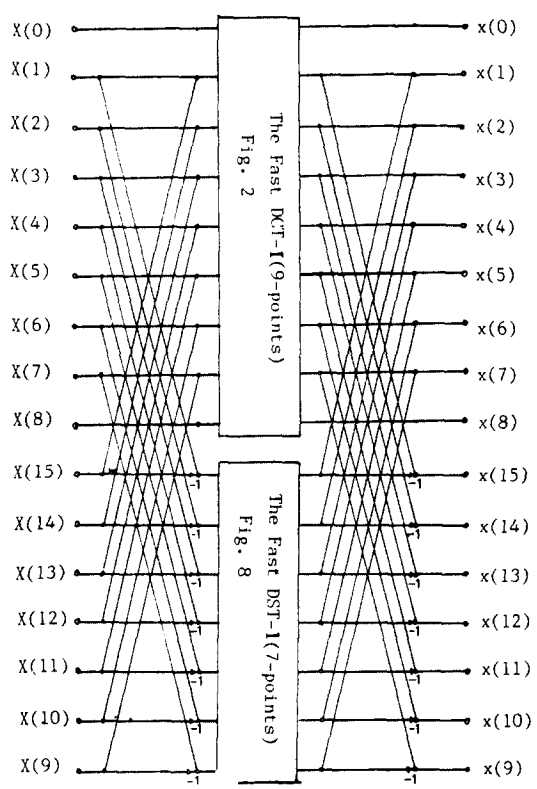


Fig. 9 Signal Flow graph for fast DHT (N=16-points).

obtain the efficient computational algorithm for DHT. For example, the fast computational algorithm for 16-points DHT is represented by Fig. 9.

Therefore the number of multiplication and addition of N-points DHT can be obtained by the following equations respectively.

$$\mu[\mathrm{N}:\mathrm{DHT}]=\mu[\mathrm{C}^{\mathrm{I}}_{\frac{\mathrm{N}}{2}+1}]+\mu[\mathrm{S}^{\mathrm{I}}_{\frac{\mathrm{N}}{2}-1}]$$

$$=\frac{\mathrm{N}}{2}\log_2(\mathrm{N})-\frac{3}{2}\mathrm{N}+2 \tag{63}$$

$$\alpha[\mathrm{N}:\mathrm{DHT}]=\alpha[\mathrm{C}^{\mathrm{I}}_{\frac{\mathrm{N}}{2}+1}]+\alpha[\mathrm{S}^{\mathrm{I}}_{\frac{\mathrm{N}}{2}-1}]+2(\mathrm{N}-2)$$

$$=(\frac{3}{2}\mathrm{N}+10)\cdot\log_2(\mathrm{N})-3\mathrm{N}-9 \tag{64}$$

And the present method is compaired with the existing efficient algorithm in Table V.

V. CONCLUSION

This paper has presented the fast algorithms for all versions of the DCT and DST as well as a DHT. By the extention and the modification of the efficient algorithm that developed for the DCT-III, we have obtained one of the most efficient algorithms to compute the other versions of DCT and four versions of DST as well as a DHT.

The algorithms developed in this paper indicate the close relationship among different versions of the DCT and DST as well as DHT. It has been shown by the computer that the flow graphs for the various examples in this paper are valid. And the numbers of multiplications and additions are comparative with the existing efficient algorithms.

REFERENCES

(1) H. Ahmed, T. Natarajan, and K.R.Rao, "Discrete cosine transform" IEEE Trans. Comput. vol. C-23, pp. 90-93, 1974.

(2) A. J. Viterbi, "Convolutional codes and their performance in communication systems", IEEE Trans. Comm. vol. COM-19, pp. 751-772, Oct. 1971.

(3) W.H.Chen, C. H. Smith, and S.C.Fralick, "A fast computational algorithm for the discrete cosine transf form", IEEE Trans. Commun. vol. COM-25, pp. 1004-1009, 1977.

Table 5 Comparison of DHT Algorithms .

N	Multiplications			Additions		
	Present Method	References		Present Method	References	
		(8)	(16)*		(8)	(16)*
8	2	2	2	33	26	22
16	10	10	12	79	74	64
32	34	34	42	185	194	166
64	98	98	124	435	482	416
128	258	258	330	1021	1154	998
256	642	642	828	2375	2690	2336
512	1538	1538	1994	5457	6146	5350
1024	3586	3586	4668	12379	13826	12064
2048	8194	8194	10698	27749	30722	26854
4096	18434	18434	24124	61551	67586	59168

* results required by applying a split radix FFT

(4) Z. Wang, "Reconsideration of 'A fast computational algorithm of the discrete cosine transform'", IEEE Trans. Commun., vol. COM-31, pp. 121-123, 1983.
(5) 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.
(6) 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.
(7) Z. Wang, "On computing the Discrete Fourier and cosine transforms," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-33, pp. 1341-1344, Oct. 1985.
(8) N. Seuhiro and M. Hatori, "Fast Algorithms for the DFT and Sinusoidal", IEEE Trans. Acous., Speech, and Signal Processing, vol. ASSP-34, June 1986.
(9) A.K. Jain, "A fast karhunen-loeve transform for a class of stochastic processes", IEEE Trsns. Commun., vol. COM-24 pp. 1023-1029, 1976.
(10) H. B. Kekre and J.K. Solanki, "Comparative Performance of various trigonometric Unitary transforms for transform image coding," Int. J. Electron., vol. 44, pp. 305-315, 1978.
(11) Z. Wang, "A fast algorithm for the discrete sine transform implemented by the fast cosine transform," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-30, pp. 814-815, Oct., 1982.
(12) P. Yip and K.R. Rao, "A fast computational algorithm for the discrete sine transform," IEEE Trans. Comm mun., vol. COM-28, pp. 304-307, 1980
(13) R.N. Bracewell, "Discrete Hartly Transform", J. Opt. Soc. Amer., vol. 73, no. 12, pp. 1832-1835, Dec. 1983.
(14) R.N. Bracewell, "The Fast Hartly Transform," Proc. IEEE, vol. 72, no. 8, pp. 1010-1018, Aug. 1984.
(15) R. Ansari, "An extention of the discrete Fourier transform," IEEE Trans. Circuits Syst., vol. 32, pp. 618-619, june 1985.
(16) H.V. SORENSEN, D. L. JONES, C. S. BURRUS, M.T. Heideman, "On computing the Discrete Hartley Transform", IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-33, no. 4, Oct. 1985.

Appendix

We write equation (31)

$$x(k)=\sum_{n=0}^{N-1} X(n)\, S_N^{(k+\frac{1}{2})(n+\frac{1}{2})} \;:\; 0\leqq k\leqq N-1 \quad (31)$$

now, we can decompose the x(k) in (31).

$$x(k)=g_1(k)+h_1(k) \quad (A1)$$

Where

$$g_1(k)=\sum_{n=0}^{\frac{N}{2}-1} X(2n)\, S_{N/2}^{(k+\frac{1}{2})(n+\frac{1}{4})} \;:\; 0\leqq k\leqq \frac{N}{2}-1 \quad (A2)$$

$$h_1(k)=\sum_{n=0}^{\frac{N}{2}-1} X(2n+1) S^{(k+\frac{1}{2})(n+\frac{3}{4})} \;:\; 0\leqq k\leqq \frac{N}{2}-1 \quad (A3)$$

In a similar manner x(N−1−k) is derived as the following equations.

$$x(N-1-k)=g_2(k)-h_2(k) \quad (A4)$$

$$g_2(k)=\sum_{n=0}^{\frac{N}{2}-1} X(2n+1)\, C_{N/2}^{(k+\frac{1}{2})(n+\frac{1}{4})} \;:\; 0\leqq k\leqq \frac{N}{2}-1 \quad (A5)$$

$$h_2(k)=\sum_{n=0}^{\frac{N}{2}-1} X(2n+1)\, C_{N/2}^{(k+\frac{1}{2})(n+\frac{3}{4})} \;:\; 0\leqq k\leqq \frac{N}{2}-1 \quad (A6)$$

Now we obtain the following equations from equations (A1), (A2), (A5) and (A6).

$$g(k)\cdot 2S_{2N}^{k+\frac{1}{2}}=\sum_{n=0}^{\frac{N}{2}-1} X(2n)\left[C_{N/2}^{n(k+\frac{1}{2})}-C_{N/2}^{(n+\frac{1}{2})(k+\frac{1}{2})}\right] \quad (A7)$$

$$h(k)\cdot 2S_{2N}^{k+\frac{1}{2}}=\sum_{n=0}^{\frac{N}{2}-1} X(2n+1)\left[C_{N/2^2}^{(k+\frac{1}{2})(n+\frac{1}{2})}\quad C_{N/2^2}^{(n+\frac{1}{2})(n+1)}\right] \quad (A8)$$

$$g(k)\cdot 2C_{2N}^{k+\frac{1}{2}}=\sum_{n=0}^{\frac{N}{2}-1} X(2n)\left[C_{N/2^2}^{(k+\frac{1}{2})(n+\frac{1}{2})}+C_{N/2}^{n(k+\frac{1}{2})}\right] \quad (A9)$$

$$h(k)\cdot 2C_{2N}^{k+\frac{1}{2}}=\sum_{n=0}^{\frac{N}{2}-1} X(2n+1)\left[C_{N/2^2}^{(k+\frac{1}{2})(n+\frac{1}{2})}+C_{N/2}^{(n+1)(k+\frac{1}{2})}\right. \quad (A10)$$

If we assume $X(-1)=0$, we have obtained the following equations from the (A1), (A4), (A7), (A8), (A9), and (A10).

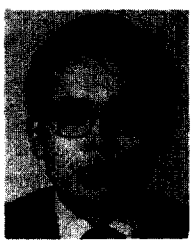
$$x(k)\ 2S_{2N}^{k+\frac{1}{2}} = \sum_{n=0}^{N/2-1} |X(2n)-X(2n-1)|\ C_{N/2}^{n(k+\frac{1}{2})} + \sum_{n=0}^{N/2-1} |X(2n+1)-X(2n)|\ C_{N/2}^{(n+\frac{1}{2})(k+\frac{1}{2})} \quad \text{(A11)}$$

$$x(N-1-k)\ 2C_{2N}^{k+\frac{1}{2}} = \sum_{n=0}^{N/2-1} |X(2n)-X(2$$

$$C_{N/2}^{(k+\frac{1}{2})(n+\frac{1}{2})} + \sum_{n=0}^{N/2-1} |X(2n)-X(2n$$

$$C_{N/2}^{n(k+\frac{1}{2})}$$

Now the equations (32) and (33) are from (A11) and (A12) respectively.



朴 鍾 演(Chong Yeun PARK) 正會員
1951年 2月23日生
1969年 3月～1973年 2月：高麗大學校電子工學科卒業 (工學士)
1978年 3月～1980年 2月：慶北大學校大學院電子工學科卒業 (工學碩士)
1980年 3月～1984年 2月：慶北大學校 大學院 電子工學科卒業 (工學博士)
1974年 4月～1977年 2月：韓國科學技術研究所 電子工學部 研究員
1980年 3月～1984年 8月：蔚山工科大學 電氣및 電子工學科
1984年 9月～1987年 5月 現在：江原大學校 電氣工學科 (副教授)