*Supported by the
National Science Foundation
Engineering Research Center
Program (NSFD CD 8803012),
the University of Maryland,
Harvard University,
and Industry*

Unified Parallel Lattice Structures for Time-Recursive Discrete Cosine/Sine/Hartley Transforms

by K.J.R. Liu and C.T. Chiu

TR 91-36

Unified Parallel Lattice Structures for Time-Recursive Discrete Cosine/Sine/Hartley Transforms

K.J. Ray Liu and C.T. Chiu

Electrical Engineering Department
Systems Research Center
University of Maryland
College Park, MD 20742
Ph: (301) 405-6619

ABSTRACT

The problems of unified efficient computations of the discrete cosine transform (DCT), discrete sine transform (DST), discrete Hartley transform (DHT), and their inverse transforms are considered. In particular, a new scheme employing the time-recursive approach to compute these transforms is presented. Using such approach, unified parallel lattice structures that can dually generate the DCT and DST simultaneously as well as the DHT are developed. These structures can obtain the transformed data for sequential input time-recursively and the total number of multipliers required is a linear function of the transform size N. Furthermore, there is no any constraint on N. The resulting architectures are regular, module, and without global communication so that it is very suitable for VLSI implementation for high-speed applications such as ISDN network and HDTV system. It is also shown in this paper that the DCT, DST, DHT and their inverse transforms share an almost identical lattice structure. The lattice structures can also be formulated into pre-lattice and post-lattice realizations. Two methods, the *SISO* and double-lattice approaches, are developed to reduce the number of multipliers in the parallel lattice structure by $2N$ and N respectively. The trade-off between time and area for the block data processing is also considered.

This work is partially supported by the NSF grant ECD-8803012-06.

1 Introduction

Transform coding has found lots of applications in image, speech, and digital signal transmission and processing. Due to the advances in ISDN network and high definition television (HDTV) technology, high speed transmission of digital video signal becomes very desirable. Among many transforms, the discrete cosine transform (DCT), discrete sine transform (DST), and discrete Hartley transform (DHT) are very effective in transform coding applications to digital signals such as speech and image signals. The DCT is most widely used in speech and image processing for data compression. This is all due to its better energy compaction property and its near optimal performance which is mostly close to that of the Karhunen-Loeve Transform (KLT) among many discrete transforms for highly correlated signals, especially for the first order Markov Process [1, 2, 3]. It was shown by Jain that the performance of the DST closes to that of the KLT for a first-order Markov sequence with given boundary conditions, especially for signal with low correlation coefficients [4, 5]. In 1983, Bracewell introduced the DHT [6] which uses a transform kernel similar to that of the discrete Fourier transform (DFT), while it is a real-valued transform. Therefore, it is simpler than the DFT with respect to the computational complexity [7]. Like the DCT and DST, the DHT has found many applications in signal and image processing [6, 8, 24, 28].

Since the DCT was introduced, many algorithms were proposed to improve the computation speed and to reduce the hardware complexity. These algorithms can be classified into the following categories: (1) indirect computation, (2) matrix factorization, (3) recursive computation, and (4) systolic structure implementation. The indirect computation [9, 10, 11, 12, 13] applies the existing fast algorithms in the DFT or the Walsh-Hadamard transform to the DCT. It is not particularly efficient because the inherent properties of the DCT are not exploited. The matrix factorization [14, 15, 25, 26] decomposes the DCT into multiplications of many sparse matrices, therefore the numbers of multiplications and additions can be comprehensively reduced. The recursive computations [16, 7] generate the higher order DCT from the lower order one, but their signal flow architectures need global communication which is not suitable for VLSI implementation. By using the recursive properties effectively, this kind of DCT algorithms has fewer multipliers and adders, while additional multiplexers are required. As for the systolic structure implementation [17, 18, 27], it uses exiting systolic architectures for the DFT or other transforms to implement the DCT in a systolic manner. But some of the methods require that the number of samples of the signal must be decomposed into mutually prime numbers. Like the DCT, many fast algorithms have been proposed to improve the performance of the DST and DHT [8, 19, 20, 4, 5]. Basically, they can be classified into the same ways as those of the DCT and similar advantages and disadvantages can also been found.

In this paper, we propose a unified time-recursive lattice structure which can be used for those discrete orthogonal transforms mentioned above, *i.e.*, the DCT, DST, and DHT. We consider the orthogonal transforms from a time-recursive point of view instead of the whole block of data. It is all due to the fact that in digital signal transmission, the data arrives serially. Also, many operations such as filtering and coding are done in a time-recursive way. Based on this approach, the resulting architectures are almost identical for the DCT, DST, and DHT. This algorithm decouples the transformed data components, hence, there is no global communication needed in this structure. Besides, the lattice module is regular and modular. Therefore, it is very suitable for VLSI implementation. In general, this method requires $6N$ multipliers for N-point data. And later we will show that a denormalization can reduce the number to $4N$. Since the number of multipliers is on the order of N, this method requires fewer multipliers than most of other algorithms when N is large. One of the important characteristics of this structure is that there are no any constraints on N. This means that N can be any integer. As we know, most of the fast algorithms for discrete transforms do have certain constraints on N. Another important result is that based on the time-recursive approach, the fundamental properties among the DCT, DST, and DHT, as well as some relative transforms can be obtained.

The rest of the paper is organized as follows. In Section 2, the dual generation of lattice structures for the DCT and DST with the time-recursive approach is considered. The inverse discrete cosine transform (IDCT) and the inverse discrete sine transform (IDST) based on the lattice structures are discussed in Section 3. In Section 4, the time-recursive lattice structure for the DHT is presented. All the above time-recursive properties are derived by updating the time index by one. With block data processing, the time index needed to be updated is more than one. The detailed effects and results of the block data processing are discussed in Section 5. Reducing the number of multipliers in the lattice structure is considered in Section 6. Then we compare these kinds of lattice structures with other architectures in terms of the number of multipliers and adders in Section 7. Finally, we give the conclusion in Section 8.

2 Dual Generation of DCT and DST

Instead of either using matrix factorizations to find fast algorithms or converting the DCT to the DFT, which can be implemented on various existing architectures, we will show an efficient implementation of the DCT from the time recursive point of view. Focusing on the sequence instead of the block of the input data, we can obtain not only the time recursive relation between the DCT of two successive data sequences, but also the fundamental relation between the DCT and DST. In the following, the time recursive relation for the DCT will be considered first.

2.1 Time-Recursive Discrete Cosine Transform

The one-dimensional (1-D) DCT of a sequential input data with length N beginning from $x(0)$ to $x(N-1)$ is defined as

$$X_c(k,0) = \frac{2C(k)}{N}\sum_{n=0}^{N-1} x(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right], \qquad k = 0,1,...,N-1, \tag{1}$$

where

$$C(k) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } k = 0, \\ 1 & \text{otherwise.} \end{cases}$$

Here the time index t in $X_c(k,t)$ denotes that the transform starts from $x(t)$. Since $C(k)$'s in (1) are constants, it is sufficient to consider first the DCT-like equation and fix the constant later on. Let

$$X_c(k,0) = \sum_{n=0}^{N-1} x(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right]. \tag{2}$$

From the definition, the 1-D DCT of the input data vector $[x(1), x(2), ..., x(N)]$ is given by

$$X_c(k,1) = \sum_{n=1}^{N} x(n)\cos\left\{\frac{\pi[2(n-1)+1]k}{2N}\right\}. \tag{3}$$

This can be rewritten as

$$\begin{aligned} X_c(k,1) &= \sum_{n=1}^{N} x(n)\cos\left[\frac{\pi(2n+1)k}{2N} - \frac{k\pi}{N}\right] \\ &= \cos\left(\frac{k\pi}{N}\right)\sum_{n=1}^{N} x(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right] + \sin\left(\frac{k\pi}{N}\right)\sum_{n=1}^{N} x(n)\sin\left[\frac{\pi(2n+1)k}{2N}\right]. \end{aligned} \tag{4}$$

Define

$$\overline{X}_c(k,1) = \sum_{n=1}^{N} x(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right], \tag{5}$$

and

$$\overline{X}_s(k,1) = \sum_{n=1}^{N} x(n)\sin\left[\frac{\pi(2n+1)k}{2N}\right], \tag{6}$$

we then have

$$X_c(k,1) = \overline{X}_c(k,1)\cos\left(\frac{\pi k}{N}\right) + \overline{X}_s(k,1)\sin\left(\frac{\pi k}{N}\right). \tag{7}$$

As we can see, a DST-like term $\overline{X}_s(k,1)$ exists in the equation. This motivates us to investigate the time-recursive DST.

2.2 Time-Recursive Discrete Sine Transform

There are several definitions for the DST. Here we prefer the definition made by Wang in [20] since it is of the form we are interested in the following derivations. The 1-D DST of a data vector $[x(0), x(1), ..., x(N-1)]$ is defined as

$$X_s(k,0) = \frac{2D(k)}{N}\sum_{n=0}^{N-1} x(n)\sin\left[\frac{\pi(2n+1)k}{2N}\right], \qquad k = 1,...,N, \tag{8}$$

where

$$D(k) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if } k = N, \\ 1 & \text{otherwise.} \end{cases}$$

Note that the range of k is from 1 to N. Again, we consider the DST-like equation first

$$X_s(k,0) = \sum_{n=0}^{N-1} x(n)\sin\left[\frac{\pi(2n+1)k}{2N}\right]. \tag{9}$$

The DST of the time update sequence $[x(1), x(2), ..., x(N)]$ is given by

$$\begin{aligned} X_s(k,1) &= \sum_{n=1}^{N} x(n)\sin\left\{\frac{\pi[2(n-1)+1]k}{2N}\right\} \\ &= \overline{X}_s(k,1)\cos\left(\frac{\pi k}{N}\right) - \overline{X}_c(k,1)\sin\left(\frac{\pi k}{N}\right). \end{aligned} \tag{10}$$

Here the terms $\overline{X}_s(k,1)$ and $\overline{X}_c(k,1)$ which are used in (7) to generate $X_c(k,1)$ appear in the equation of the new DST transform $X_s(k,1)$ again. It suggests that the DCT and DST can be dually generated from each other.

2.3 The Lattice Structures

From (7) and (10), it is noted that the new DCT and DST transforms $X_c(k,1)$ and $X_s(k,1)$, can be obtained from $\overline{X}_c(k,1)$ and $\overline{X}_s(k,1)$ in the lattice form as shown in Fig. 1. The next step is to update $\overline{X}_c(k,1)$ and $\overline{X}_s(k,1)$ from the previous transforms $X_c(k,0)$ and $X_s(k,0)$.

We notice that $X_c(k,0)$ and $\overline{X}_c(k,1)$ have similar terms except the old data $x(0)$ and the incoming new data $x(N)$. Therefore $\overline{X}_c(k,1)$ and $\overline{X}_s(k,1)$ can be obtained by downdating the old data $x(0)$ and updating the new data $x(N)$ as

$$\begin{aligned}\overline{X}_c(k,1) &= X_c(k,0) - x(0)\cos\left(\frac{\pi k}{2N}\right) + x(N)\cos\left[\frac{\pi(2N+1)k}{2N}\right] \\ &= X_c(k,0) + \left[-x(0) + (-1)^k x(N)\right]\cos\left(\frac{\pi k}{2N}\right),\end{aligned} \tag{11}$$

and

$$\begin{aligned}\overline{X}_s(k,1) &= X_s(k,0) - x(0)\sin\left(\frac{\pi k}{2N}\right) + x(N)\sin\left[\frac{\pi(2N+1)k}{2N}\right] \\ &= X_s(k,0) + \left[-x(0) + (-1)^k x(N)\right]\sin\left(\frac{\pi k}{2N}\right).\end{aligned} \tag{12}$$

From (7), (10), (11), and (12), the new transforms $X_c(k,1)$ and $X_s(k,1)$ can be calculated from the previous transforms $X_c(k,0)$ and $X_s(k,0)$ by adding the effect of input signals $x(0)$ and $x(N)$. This manifests that the DCT and DST can be dually generated from each other in a recursive way. The complete recursive lattice structure is shown in Fig. 2. Since the multiplications can be reduced to addition and substraction for $k=0$ in the DCT and $k=N$ in the DST respectively, these two cases can be simplified and combined together as shown in Fig. 3, while Fig. 2 is good for $k=1,2,...,N-1$.

To obtain the transformed signals defined in (1) and (8), we need to take the leading coefficients $C(k)$'s and $D(k)$'s into consideration. The problem is examined under three cases: (1) $k=1,2,...,N-1$, for the DCT and DST, (2) $k=0$ for the DCT, and (3) $k=N$ for the DST. For $k=1,2,...,N-1$, the functions $C(k)$ and $D(k)$ have the same values $2/N$. By employing the same approach, the terms $\overline{X}_c(k,1)$ and $\overline{X}_s(k,1)$ become

$$\overline{X}_c(k,1) = X_c(k,0) + \left[-x(0) + (-1)^k x(N)\right]\left(\frac{2}{N}\right)\cos\left(\frac{\pi k}{2N}\right), \tag{13}$$

and

$$\overline{X}_s(k,1) = X_s(k,0) + \left[-x(0) + (-1)^k x(N)\right]\left(\frac{2}{N}\right)\sin\left(\frac{\pi k}{2N}\right). \tag{14}$$

Therefore the time-recursive relations for the new transforms $X_c(k,1)$ and $X_s(k,1)$ as well as the previous transforms $X_c(k,0)$ and $X_s(k,0)$ are given by

$$\begin{aligned} X_c(k,1) = &\left\{X_c(k,0) + \left[-x(0) + (-1)^k x(N)\right]\left(\frac{2}{N}\right)\cos\left(\frac{\pi k}{2N}\right)\right\}\cos(\frac{\pi k}{N}) \\ &+ \left\{X_s(k,0) + \left[-x(0) + (-1)^k x(N)\right]\left(\frac{2}{N}\right)\sin\left(\frac{\pi k}{2N}\right)\right\}\sin(\frac{\pi k}{N}), \end{aligned} \tag{15}$$

and

$$\begin{aligned} X_s(k,1) = &\left\{X_s(k,0) + \left[-x(0) + (-1)^k x(N)\right]\left(\frac{2}{N}\right)\sin\left(\frac{\pi k}{2N}\right)\right\}\cos(\frac{\pi k}{N}) \\ &- \left\{X_c(k,0) + \left[-x(0) + (-1)^k x(N)\right]\left(\frac{2}{N}\right)\cos\left(\frac{\pi k}{2N}\right)\right\}\sin(\frac{\pi k}{N}). \end{aligned} \tag{16}$$

It is easy to see that the recursive structure is unchange except that the coefficients of the multipliers in the input are $\frac{2}{N}\cos(\frac{\pi k}{2N})$ and $\frac{2}{N}\sin(\frac{\pi k}{2N})$ instead of $\cos(\frac{\pi k}{2N})$ and $\sin(\frac{\pi k}{2N})$ as shown in Fig. 4. Cases (2) and (3) can be overcomed by incorporating one multiplier with a coefficient $\frac{2}{N\sqrt{2}}$ at the outputs as shown in Fig. 5.

Following is to illustrate how this dually generated DCT and DST lattice structure works to obtain the DCT and DST with length N of a series input data $[x(0), x(1), .., x(N-1), x(N), ...]$ for a specific k. The initial values of the transformed signals $X_c(k)$ and $X_s(k)$ are set to zero, so do the initial values in the shift register in the front of the lattice module. The input sequence $[x(0), x(1), ...]$ shifts sequentially into the shift register as shown in Fig. 4. Then the output signals $X_c(k)$ and $X_s(k)$, $k = 0, 1, ..., N-1$, are updated recursively according to (15) and (16). After the input datum $x(N-1)$ shifts into the shift register, the DCT and DST of the input data vector $[x(0), x(1), ..., x(N-1)]$ are obtained at the output for this index k. It takes N clock cycle to get the $X_c(k)$ and $X_s(k)$ of the input vector $[x(0), x(1), ..., x(N-1)]$. Since there are N different values for k, the total computational time to obtain all the transformed data is N^2 clock cycles, if only one lattice module is used. As shown in Fig. 6, a parallel lattice structure consists of N lattice modules can be used for parallel computations and improves the compuational speed drastically. Here we have seen that the transform domain data $X(k,t)$ have been decomposed into N disjoint components that have the same lattice modules with different multipliers coefficients in them. In this case the total computational time decreases to N clock cycle. It is important to notice that when the next input datum $x(N)$ arrives, the transformed data of the input data vector $[x(1), x(2), ..., x(N)]$ can be obtained immediately. Likewise, the transformed data of subsequent inputs can be generated per clock cycle.

It is obvious that this lattice structure is quite different from the signal flow graph realization obtained from the fast DCT algorithms [14, 15]. Since there is no global communication and the structure is modular and regular, it is suitable for practical VLSI implementation. The most fascinating result is that this architecture can be applied to any number of N. From this point of view, it is more attractive than other existing algorithms. In fact, most algorithms [21, 18] are limited to the sequence length N which either must be power of 2 or must be decomposable into mutually prime number. In addition, this lattice structure reveals some interesting properties between the DCT and DST. The DCT and DST can be dually generated simultaneously. Since the DCT is near the optimal tranform KLT in highly correlated signal, while the DST approaches to the KLT in low correlation coefficient. As we are able to obtain the DCT and DST at the same time, this lattice structure is very useful especially when we do not know the statistics of the incoming signal. Furthermore, we can use a single lattice module with only 6 multipliers and 5 adders to recursively compute any N-point DCT and DST simultaneously. To have the transformed data parallely, we need N lattice modules. As mentioned before, it is suitable for VLSI implementation since all the modules have the same structures except the 0th module can be simplified as shown in Fig. 5. This parallel lattice structure requires $6N - 4$ multipliers and $5N - 1$ adders.

3 Inverse Discrete Cosine Transform (IDCT) And Inverse Discrete Sine Transform (IDST)

3.1 Time-Recursive IDCT

According to the definition of the DCT in (1), the IDCT for the transform domain sequence $[X(0), X(1), ..., X(N-1)]$ is

$$x_c(n,0) = \sum_{k=0}^{N-1} C(k)X(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right], \qquad n = 0,1,...,N-1. \tag{17}$$

The coefficients $C(k)$'s are given in (1). From the time-recursive point of view, the IDCT of the new sequence $[X(1), X(2), ..., X(N)]$ can be expressed as

$$x_c(n,1) = \sum_{k=1}^{N} C(k-1)X(k)\cos\left[\frac{\pi(2n+1)(k-1)}{2N}\right]. \tag{18}$$

Similar to the the previous sections, we can decompose (18) into

$$\begin{aligned} x_c(n,1) \quad = \quad & \cos\left[\frac{\pi(2n+1)}{2N}\right]\sum_{k=1}^{N} C(k-1)X(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right] \\ & + \sin\left[\frac{\pi(2n+1)}{2N}\right]\sum_{k=1}^{N} C(k-1)X(k)\sin\left[\frac{\pi(2n+1)k}{2N}\right]. \end{aligned} \tag{19}$$

Define

$$\overline{x}_c(n,1) = \sum_{k=1}^{N} C(k-1)X(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right], \tag{20}$$

and

$$\overline{x}_s(n,1) = \sum_{k=1}^{N} C(k-1)X(k)\sin\left[\frac{\pi(2n+1)k}{2N}\right], \tag{21}$$

then we have

$$x_c(n,1) = \overline{x}_c(n,1)\cos\left[\frac{\pi(2n+1)}{2N}\right] + \overline{x}_s(n,1)\sin\left[\frac{\pi(2n+1)}{2N}\right]. \tag{22}$$

In order to be a dually generated pair of the IDCT given in (17), we define the auxiliary inverse discrete sine transform (AIDST) as

$$x_s(n,0) = \sum_{k=0}^{N-1} C(k)X(k)\sin\left[\frac{\pi(2n+1)k}{2N}\right], \qquad n = 0,1,...,N-1. \tag{23}$$

Although this definition utilizes the sine functions as the transform kernel, it is not the inverse transform of the DST. To tell it from the IDST, we call this the AIDST. The AIDST for the data sequence $[X(1), X(2), ..., X(N)]$ can be written as

$$x_s(n,1) = \sum_{k=1}^{N} C(k-1)X(k)\sin\left[\frac{\pi(2n+1)(k-1)}{2N}\right]. \tag{24}$$

By using the trigonometric function expansions, $x_s(n,1)$ becomes

$$x_s(n,1) = \overline{x}_s(n,1)\cos\left[\frac{\pi(2n+1)}{2N}\right] - \overline{x}_c(n,1)\sin\left[\frac{\pi(2n+1)}{2N}\right]. \tag{25}$$

3.1.1 Lattice Structure for IDCT

Combining (22) and (25), we can see that the IDCT and the AIDST can be generated in exactly the same way as the dual generation of the DCT and DST. Therefore, the lattice structure in Fig.1 can be applied here except that the coefficients need to be modified. Since the coefficients $C(k)$'s are inside the expression in the inverse transform, the relation between $x_c(n,0)$ and $\overline{x}_c(n,1)$ will be different from what we have in the DCT. Equations (17) and (20) as well as (21) and (23) have the same terms for $k \in \{2, 3..., N-1\}$. After adding the effects of the terms for $k = 0$ and $k = 1$, we obtain

$$\overline{x}_c(n,1) = x_c(n,0) - \frac{1}{\sqrt{2}}X(0) + \left(\frac{1}{\sqrt{2}} - 1\right)\cos\left[\frac{\pi(2n+1)}{2N}\right]X(1), \tag{26}$$

and

$$\overline{x}_s(n,1) = x_s(n,0) + (-1)^n X(N) + \left(\frac{1}{\sqrt{2}} - 1\right)\sin\left[\frac{\pi(2n+1)}{2N}\right]X(1). \tag{27}$$

The complete lattice module for the IDCT and AIDST is shown in Fig.7. This IDCT lattice structure has the same lattice module as that of the DCT except for the input stage where one more adder and one more multiplier are needed. However, the procedure to calculate the transformed data is the same. Therefore, this IDCT lattice structure has all the advantages as that of the DCT. To obtain the inverse transform parallelly, we need N such IDCT lattice modules where $7N$ multipliers and $6N$ adders are required. Again, we can see that the numbers of adders and multipliers are linear functions of N.

3.2 Time-Recursive IDST

From the definition of the DST in (8), the IDST for the transform domain sequence $[X(1), X(2), ..., X(N)]$ is given by

$$x_s(n,0) = \sum_{k=1}^{N} D(k)X(k)\sin\left[\frac{\pi(2n+1)k}{2N}\right], \qquad n = 0, 1, ..., N-1. \tag{28}$$

The coefficients $D(k)$'s are given in (8). Analogous to Section 3.1, we define the auxiliary inverse discrete cosine transform (AIDCT)

$$x_c(n,0) = \sum_{k=1}^{N} D(k)X(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right], \qquad n = 0, 1, ..., N-1, \tag{29}$$

which is the dually generated pair of the IDST. The IDST and AIDCT of the new sequence of transformed data $[X(2), X(3), ..., X(N+1)]$ are given respectively by

$$x_s(n,1) = \sum_{k=2}^{N+1} D(k-1)X(k)\sin\left[\frac{\pi(2n+1)(k-1)}{2N}\right], \tag{30}$$

and

$$x_c(n,1) = \sum_{k=2}^{N+1} D(k-1)X(k)\cos\left[\frac{\pi(2n+1)(k-1)}{2N}\right]. \tag{31}$$

Same as before, we can decompose (30) and (31) to

$$x_s(n,1) = \overline{x}_s(n,1)\cos\left[\frac{\pi(2n+1)}{2N}\right] - \overline{x}_c(n,1)\sin\left[\frac{\pi(2n+1)}{2N}\right], \tag{32}$$

and

$$x_c(n,1) = \overline{x}_c(n,1)\cos\left[\frac{\pi(2n+1)}{2N}\right] + \overline{x}_s(n,1)\sin\left[\frac{\pi(2n+1)}{2N}\right], \tag{33}$$

where

$$\overline{x}_c(n,1) = \sum_{k=2}^{N+1} D(k-1)X(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right], \tag{34}$$

and

$$\overline{x}_s(n,1) = \sum_{k=2}^{N+1} D(k-1)X(k)\sin\left[\frac{\pi(2n+1)k}{2N}\right]. \tag{35}$$

3.2.1 Lattice Structure for IDST and AIDCT

If we employ the technique in the previous section to exploit the relations between $x_s(n,1)$ and $\overline{x}_s(n,1)$ as well as $x_c(n,1)$ and $\overline{x}_c(n,1)$, the results are

$$\begin{aligned}\overline{x}_s(n,1) &= x_s(n,0) - \sin\left[\frac{\pi(2n+1)}{2N}\right]X(1) \\ &- \left(\frac{1}{\sqrt{2}}-1\right)(-1)^n X(N) + \frac{1}{\sqrt{2}}(-1)^n \cos\left[\frac{\pi(2n+1)}{2N}\right]X(N+1), \quad (36)\end{aligned}$$

and

$$\overline{x}_c(n,1) = x_c(n,0) - \cos\left[\frac{\pi(2n+1)}{2N}\right]X(1) - \frac{1}{\sqrt{2}}(-1)^n \sin\left[\frac{\pi(2n+1)}{2N}\right]X(N+1). \quad (37)$$

Equations (32), (33), (36) and (37) reveal that to dually generate the IDST and AIDCT requires 9 multipliers and 7 adders which are more than that of the IDCT and AIDST. The result is shown in Fig. 8. To reduce the number of multipliers and adders, substituting (36) and (37) into (32) and (33) and rearranging (32) and (33), then we have

$$\begin{aligned}x_s(n,1) &= \cos\left[\frac{\pi(2n+1)}{2N}\right]x_s(n,0) - \sin\left[\frac{\pi(2n+1)}{2N}\right]x_c(n,0) \\ &-\left(\frac{1}{\sqrt{2}}-1\right)(-1)^n \cos\left[\frac{\pi(2n+1)}{2N}\right]X(N) + \left(\frac{1}{\sqrt{2}}\right)(-1)^n X(N+1), \quad (38)\end{aligned}$$

and

$$\begin{aligned}x_c(n,1) &= \cos\left[\frac{\pi(2n+1)}{2N}\right]x_c(n,0) + \sin\left[\frac{\pi(2n+1)}{2N}\right]x_s(n,0) \\ &-X(1) - \left(\frac{1}{\sqrt{2}}-1\right)(-1)^n \sin\left[\frac{\pi(2n+1)}{2N}\right]X(N) + \left(\frac{1}{\sqrt{2}}\right)(-1)^n X(N+1) \quad (39)\end{aligned}$$

The lattice module of this rearranged IDST and AIDCT is shown in Fig. 9. This structure differs from all the previous lattice modules in that the input signals are added at the end of the lattice. From now on, we call this lattice structure a post-lattice module and the previous ones as pre-lattice modules. This post-lattice module needs 7 multipliers and 5 adders which are less than the corresponding pre-lattice module. A parallel post-lattice structure, which generates N transformed data simultaneously, requires $7N$ multipliers and $5N$ adders. All the transform pairs mentioned above have pre-lattice and post-lattice structures. Not all post-lattice structures are superier to their pre-lattice counterparts in the hardware complexity. For example, the IDCT and AIDST post-lattice module has 9 multipliers and 7 adders which are more than its pre-lattice realization. As to the DCT and DST, the pre-lattice and post-lattice modules have the same numbers of multipliers and adders.

4 Discrete Hartley Transform (DHT)

According to Bracewell's definition of the DHT in [6], the data sequence $x(n)$ and the DHT transformed data $X(k)$ have the following relation

$$\begin{aligned} X(k,0) &= \frac{1}{N}\sum_{n=0}^{N-1} x(n) cas\left(\frac{2\pi kn}{N}\right) \\ &= \frac{1}{N}\sum_{n=0}^{N-1} x(n)\left[\cos\left(\frac{2\pi kn}{N}\right) + \sin\left(\frac{2\pi kn}{N}\right)\right], \\ & k = 0, 1, ..., N-1. \end{aligned} \tag{40}$$

The DHT uses real variables $\cos(\frac{2\pi kn}{N}) + \sin(\frac{2\pi kn}{N})$ as the transform kernel, while discrete Fourier transform (DFT) uses the complex exponential $\exp(\frac{i2\pi kn}{N})$, as the transform kernel. Since one complex multiplication needs four real multiplications, the compuation of the DHT is simpler than that of the DFT [19]. Because the kernel of the DHT is a summation of cosine and sine terms, we can separate them to make it look like a combination of a DCT-like and a DST-like transforms in the following way

$$X(k,0) = \acute{X}_c(k,0) + \acute{X}_s(k,0), \tag{41}$$

where

$$\acute{X}_c(k,0) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\left[\cos\left(\frac{2\pi kn}{N}\right)\right], \tag{42}$$

and

$$\acute{X}_s(k,0) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\left[\sin\left(\frac{2\pi kn}{N}\right)\right]. \tag{43}$$

The preceding coefficient $\frac{1}{N}$ in (40) is neglected here for simplicity. The $\acute{X}_c(k,0)$ is the so called DCT-I and the $\acute{X}_s(k,0)$ is the DST-I that are defined by Yip and Rao in [22]. Since the DHT can be decomposed into the combination of the DCT-I and DST-I, the dual generation of both for the DHT is thus possible. The DCT-I and the DST-I of the data sequence $[x(1), x(2), ..., x(N)]$ are

$$\acute{X}_c(k,1) = \frac{1}{N}\sum_{n=1}^{N} x(n)\cos\left[\frac{2\pi k(n-1)}{N}\right], \tag{44}$$

and

$$\acute{X}_s(k,1) = \frac{1}{N}\sum_{n=1}^{N} x(n)\sin\left[\frac{2\pi k(n-1)}{N}\right]. \tag{45}$$

The new transforms $\acute{X}_c(k,1)$ and $\acute{X}_s(k,1)$ can be further expressed as

$$\acute{X}_c(k,1) = \overline{\acute{X}}_c(k,1)\cos\left(\frac{2\pi k}{N}\right) + \overline{\acute{X}}_s(k,1)\sin\left(\frac{2\pi k}{N}\right), \tag{46}$$

and

$$\acute{X}_s(k,1) = \overline{\acute{X}}_s(k,1)\cos\left(\frac{2\pi k}{N}\right) - \overline{\acute{X}}_c(k,1)\sin\left(\frac{2\pi k}{N}\right), \tag{47}$$

where

$$\begin{aligned}\overline{\acute{X}}_c(k,1) &= \frac{1}{N}\sum_{n=1}^{N} x(n)\cos\left(\frac{2\pi kn}{N}\right) \\ &= \acute{X}_c(k,0) + \frac{1}{N}\left[-x(0) + x(N)\right],\end{aligned} \tag{48}$$

and

$$\begin{aligned}\overline{\acute{X}}_s(k,1) &= \frac{1}{N}\sum_{n=1}^{N} x(n)\sin\left(\frac{2\pi kn}{N}\right) \\ &= \acute{X}_s(k,0).\end{aligned} \tag{49}$$

The lattice module for the DHT is shown in Fig. 10. For the case of $k = 0$, the lattice structure can be further simplified into Fig. 11. From Fig. 10, we can see that the numbers of multipliers and adders are less than those of the dual generation of the DCT and DST. The total numbers of multipliers and adders in the parallel DHT lattice architecture are $5N-4$ and $5N-3$ respectively.

5 Block Processing

All the time-recursive discrete transforms derived above are all based on the block-size-one update which means the time index is updated by one. That is, at each iteration only the effect of one old data is removed and the information of one new data is updated. We expect better performance by increasing the block size. This motivates us to discuss the effect on the lattice structure when the block size is increased.

5.1 Block Processing of time-recursive DCT and DST

We begin the discussion of block processing with the block-size-two update. As before, the transformed data $X_c(k,2)$ and $X_s(k,2)$ are defined as the DCT and DST of the input vector $[x(2), x(3), ..., x(N), x(N+1)]$. That is,

$$X_c(k,2) = \sum_{n=2}^{N+1} x(n) \cos\left\{\frac{\pi[2(n-2)+1]k}{2N}\right\}, \tag{50}$$

and

$$X_s(k,2) = \sum_{n=2}^{N+1} x(n) \sin\left\{\frac{\pi[2(n-2)+1]k}{2N}\right\}. \tag{51}$$

To obtain $X_c(k,2)$ from $X_c(k,0)$ and $X_s(k,2)$ from $X_s(k,0)$ directly, we can rewrite $X_c(k,2)$ and $X_s(k,2)$ as

$$X_c(k,2) = \overline{X}_c(k,2)\cos\left(\frac{2\pi k}{N}\right) + \overline{X}_s(k,2)\sin\left(\frac{2\pi k}{N}\right), \tag{52}$$

and

$$X_s(k,2) = \overline{X}_s(k,2)\cos\left(\frac{2\pi k}{N}\right) - \overline{X}_c(k,2)\sin\left(\frac{2\pi k}{N}\right), \tag{53}$$

where

$$\begin{aligned}\overline{X}_c(k,2) &= \sum_{n=2}^{N+1} x(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right] \\ &= X_c(k,0)+[-x(0)+(-1)^k x(N)]\cos\left(\frac{\pi k}{2N}\right) \\ &\quad +[-x(1)+(-1)^k x(N+1)]\cos\left(\frac{3\pi k}{2N}\right), \qquad (54)\end{aligned}$$

and

$$\begin{aligned}\overline{X}_s(k,2) &= \sum_{n=2}^{N+1} x(n)\sin\left[\frac{\pi(2n+1)k}{2N}\right] \\ &= X_s(k,0)+[-x(0)+(-1)^k x(N)]\sin\left(\frac{\pi k}{2N}\right) \\ &\quad +[-x(1)+(-1)^k x(N+1)]\sin\left(\frac{3\pi k}{2N}\right). \qquad (55)\end{aligned}$$

The lattice module for the block-size-two update is shown in Fig. 12. There are two more multipliers in the lattice, *i.e.*, the total number of multipliers is eight. To have the transformed data parallelly, we need N such lattice modules. The latency for this kind of parallel structure is $N/2$ and the total number of multipliers is $8N$. Since there is no complex communication problem in the lattice structure, the area-time complexity (AT) can be approximated by the product of the number of multipliers and the time latency. Denote ATm as the area-time complexity of the block-size-m update, then $AT1 = 6N^2$ and $AT2 = 4N^2$. Next, let us consider the more general case for the block-size-m update, where m ranges from one to N. The 1-D DCT and DST of block-size-m update are to obtain the transform of $[x(m), x(m+1), ..., x(N+m-1)]$ directly from the transform of $[x(0), x(1), ..., x(N-1)]$. We have

$$X_c(k,m) = \sum_{n=m}^{N+m-1} x(n)\cos\left[\frac{\pi[2(n-m)+1]k}{2N}\right], \qquad (56)$$

and

$$X_s(k,m) = \sum_{n=m}^{N+m-1} x(n)\sin\left[\frac{\pi[2(n-m)+1]k}{2N}\right]. \qquad (57)$$

Applying the same procedure in the case of block-size-two update, we can write (56) and (57) as

$$X_c(k,m) = \overline{X}_c(k,m)\cos\left(\frac{m\pi k}{N}\right) + \overline{X}_s(k,m)\sin\left(\frac{m\pi k}{N}\right), \tag{58}$$

and

$$X_s(k,m) = \overline{X}_s(k,m)\cos\left(\frac{m\pi k}{N}\right) - \overline{X}_c(k,m)\sin\left(\frac{m\pi k}{N}\right). \tag{59}$$

where

$$\begin{aligned}
\overline{X}_c(k,m) &= \sum_{n=m}^{N+m-1} x(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right] \\
&= X_c(k,0) - \sum_{n=0}^{m-1} x(n)\cos\left(\frac{\pi(2n+1)k}{2N}\right) \\
&\quad + \sum_{n=N}^{N+m-1} x(n)\cos\left[\frac{\pi(2n+1)k}{2N})\right]\cos\left(\frac{\pi mk}{2N}\right) \\
&= X_c(k,0) + \sum_{n=0}^{m-1}[-x(n)+(-1)^k x(N+n)]\cos\left[\frac{\pi(2n+1)k}{2N}\right],
\end{aligned} \tag{60}$$

and

$$\begin{aligned}
\overline{X}_s(k,m) &= \sum_{n=m}^{N+m-1} x(n)\sin\left[\frac{\pi(2n+1)k}{2N}\right] \\
&= X_s(k,0) - \sum_{n=0}^{m-1} x(n)\sin\left[\frac{\pi(2n+1)k}{2N}\right] \\
&\quad - \sum_{n=N}^{N+m-1} x(n)\sin\left[\frac{\pi(2n+1)k}{2N}\right]\sin\left(\frac{\pi mk}{2N}\right) \\
&= X_c(k,0) + \sum_{n=0}^{m-1}[-x(n)+(-1)^k x(N+n)]\sin\left[\frac{\pi(2n+1)k}{2N}\right].
\end{aligned} \tag{61}$$

Combining those input terms with same cosine and sine multiplier coefficients together, we can obtain the lattice module for block size m as shown in Fig. 13. To obtain the transform data $X(k)$ parallelly, N lattice modules of Fig. 13 are required. The total number of multipliers of the parallel structure is $(4+2m)N$ and the latency time is N/m. Therefore, the area-time complexity is $ATm = \frac{(4+2m)N^2}{m}$. In the limiting case of the block-size-N update, *i.e.*, we move a whole block of the input data sequence, $ATN = (4+2N)N$ which reduces to the smallest area-time complexity. The area-time product is getting smaller when the block size m is increased. This means that when the block size is increased, the area-time complexity becomes more efficient. That is, if the hardware resource is allowed, we can trade the number of multipliers for better time efficiency.

6 Multiplier-Reduction of the lattice structure

In the VLSI implementation, the number of multipliers is an important factor to the cost and complexity of the system. In this section, we develop two methods to reduce the number of multipliers in our parallel lattice structures. The first scheme makes use of a series input series output ($SISO$) approach and $2N$ multipliers can be saved; the trade-off is that the latency is increased. The second approach, which reconstructs the structure into a double-lattice realization, reduces N multipliers and the latency remains intact.

6.1 $SISO$ Approach

Denote the output and input data at time k as $(X_c(k), X_s(k))$ and (x_{ck}, x_{sk}) respectively for a lattice structure as shown in Fig. 14, where the input and output have the following relations

$$\begin{aligned}
X_c(k) &= [X_c(k-1) + \Gamma_1 x_{ck}]\,\Gamma_2 + [X_s(k-1) + \Gamma_3 x_{sk}]\,\Gamma_4, \\
X_s(k) &= [X_s(k-1) + \Gamma_3 x_{sk}]\,\Gamma_2 - [X_c(k-1) + \Gamma_1 x_{ck}]\,\Gamma_4. \qquad (62)
\end{aligned}$$

By dividing both equations by Γ_4, we have

$$\begin{aligned}
X_c(k)/\Gamma_4 &= [X_c(k-1) + \Gamma_1 x_{ck}]\,\Gamma_2/\Gamma_4 + [X_s(k-1) + \Gamma_3 x_{sk}]\,, \\
X_s(k)/\Gamma_4 &= [X_s(k-1) + \Gamma_3 x_{sk}]\,\Gamma_2/\Gamma_4 - [X_c(k-1) + \Gamma_1 x_{ck}]\,. \qquad (63)
\end{aligned}$$

The lattice structure manifesting the above relations is plotted in Fig. 15. It is noted that only two multipliers exist in this structure and the outputs obtained differ from the original one by a factor Γ_4. To examine the effect of this multiplier reduction on the recursive operation from $X_c(1)$ to $X_c(N)$, we start with the derivation from $k = 1$. That is

$$\begin{aligned} X_c(1)/\Gamma_4 &= [X_c(0) + \Gamma_1 x_{c1}]\,\Gamma_2/\Gamma_4 + [X_s(0) + \Gamma_3 x_{s1}]\,, \\ X_s(1)/\Gamma_4 &= [X_s(0) + \Gamma_3 x_{s1}]\,\Gamma_2/\Gamma_4 - [X_c(0) + \Gamma_1 x_{c1}]\,. \end{aligned} \tag{64}$$

For $k = 2$

$$\begin{aligned} X_c(2)/\Gamma_4 &= [X_c(1) + \Gamma_1 x_{c2}]\,\Gamma_2/\Gamma_4 + [X_s(1) + \Gamma_3 x_{s2}]\,, \\ X_s(2)/\Gamma_4 &= [X_s(1) + \Gamma_3 x_{s2}]\,\Gamma_2/\Gamma_4 - [X_c(1) + \Gamma_1 x_{c2}]\,. \end{aligned} \tag{65}$$

Because the outputs at time $k = 1$ are $X_c(1)/\Gamma_4$ and $X_s(1)/\Gamma_4$, $X_c(1)$ and $X_s(1)$ at (65) should be replaced by $X_c(1)/\Gamma_4$ and $X_s(1)/\Gamma_4$. To keep the above equations valid, we can multiply both equations by $1/\Gamma_4$ as shown

$$\begin{aligned} X_c(2)/\Gamma_4^2 &= [X_c(1)/\Gamma_4 + (\Gamma_1/\Gamma_4)x_{c2}]\,\Gamma_2/\Gamma_4 + [X_s(1)/\Gamma_4 + (\Gamma_3/\Gamma_4)x_{s2}]\,, \\ X_s(2)/\Gamma_4^2 &= [X_s(1)/\Gamma_4 + (\Gamma_3/\Gamma_4)x_{s2}]\,\Gamma_2/\Gamma_4 - [X_s(1)/\Gamma_4 + (\Gamma_1/\Gamma_4)x_{c2}]\,. \end{aligned} \tag{66}$$

The coefficients of the input multipliers are Γ_1/Γ_4 and Γ_3/Γ_4, instead of Γ_1 and Γ_3 at time $k = 1$, and the output are $X_c(2)/\Gamma_4^2$ and $X_s(2)/\Gamma_4^2$. For $k = N$, the recursive equations become

$$\begin{aligned} X_c(N)/\Gamma_4^N &= \left[X_c(N-1)/\Gamma_4^{N-1} + (\Gamma_1/\Gamma_4^{N-1})x_{cN}\right]\Gamma_2/\Gamma_4 \\ &\quad + \left[X_s(N-1)/\Gamma_4^{N-1} + (\Gamma_3/\Gamma_4^{N-1})x_{sN}\right], \\ X_s(N)/\Gamma_4^N &= \left[X_s(N-1)/\Gamma_4^{N-1} + (\Gamma_3/\Gamma_4^{N-1})x_{sN}\right]\Gamma_2/\Gamma_4 \\ &\quad - \left[X_c(N-1)/\Gamma_4^{N-1} + (\Gamma_1/\Gamma_4^{N-1})x_{cN}\right]. \end{aligned} \tag{67}$$

From the above derivations, we observe that the two multipliers can be removed by using variable multipliers in the input stage where the multiplicators $(\Gamma_1, \Gamma_1/\Gamma_4, .., \Gamma_1/\Gamma_4^{N-1})$ and $(\Gamma_3, \Gamma_3/\Gamma_4, .., \Gamma_3/\Gamma_4^{N-1})$ are stored in the shift registers. The structure are shown in Fig. 16.

The output can be obtained by multiplying the factor Γ_4^N. This kind of rearrangement does not save the multipliers. However, for N such lattice structure, the number of multipliers can be reduced by using variable multipliers at the output stage and the coefficients for each stage $\Gamma_4^N(i)$, $i = 0, 1, 2, .., N-1$, are stored in the shift registers. Fig. 17 shows the final structure where the total number of multipliers are $4N + 2$. This means that the number of multipliers for N parallel such lattice structures are reduced from $6N$ to $4N + 2$. The tradeoff is that $2N + 2$ shift registers are required and the latency becomes $2N$ instead of N. Also, this resulting structure is a $SISO$ system, while the original parallel structure is a $SIPO$ system.

For example, the variable-multiplier method derived above can be applied to the lattice structure of the DCT and DST. There is no multipliers needed for $k = 0$, therefore the module remain the same. For $k = 1, 2, .., N-1$, the multiplier-reduced lattice structure is shown in Fig. 17, where the coefficients are $\Gamma_1 = \cos(k\pi/2N)$, $\Gamma_2 = \cos(k\pi/N)$, $\Gamma_3 = \sin(k\pi/2N)$, and $\Gamma_4 = \sin(k\pi/N)$. The total multipliers are $4N - 2$ and the latency for this $SISO$ structure is $2N$.

6.2 Double-lattice Approach

Generally, a post-lattice structure has the following forms

$$\begin{aligned} X_c(k) &= \Gamma_2 X_c(k-1) + \Gamma_4 X_s(k-1) + \Gamma_1 x_{ck}, \\ X_s(k) &= \Gamma_2 X_s(k-1) - \Gamma_4 X_c(k-1) + \Gamma_3 x_{sk}. \end{aligned} \tag{68}$$

The $X_c(k)$ and $X_s(k)$ can be rearranged in the following manner

$$\begin{aligned} X_c(k) &= \frac{1}{2}\{(\Gamma_2 + \Gamma_4)[X_c(k-1) + X_s(k-1)] + (\Gamma_2 - \Gamma_4)[X_c(k-1) - X_s(k-1)]\} \\ &\quad +\Gamma_1 x_{ck}, \\ X_s(k) &= \frac{1}{2}\{(\Gamma_2 + \Gamma_4)[X_c(k-1) + X_s(k-1)] - (\Gamma_2 - \Gamma_4)[X_c(k-1) - X_s(k-1)]\} \\ &\quad -2\Gamma_4 X_c(k-1) + \Gamma_3 x_{ck}. \end{aligned} \tag{69}$$

The operational flow graph of (69) is demonstrated in Fig. 18. Instead of calculating the outputs from (68) directly, we add and subtract $X_c(k-1)$ and $X_s(k-1)$ first, then mutiply the results by the sum and difference of the multiplying coefficients Γ_2 and Γ_4 respectively. The results are called $t1$ and $t2$ as shown in the Fig. 18. We add and subtract $t1$ and $t2$ again, then divide the results by 2, which can be achieved by left shifting. Finally, we complete the computations by adding the inputs $\Gamma_1 x_{ck}$ and $\Gamma_3 x_{ck} - 2\Gamma_4 X_c(k-1)$. This reconstruction can save one multiplier. A parallel lattice structure with N lattice modules based on (68) needs $6N$ multipliers and $4N$ adders. As for this reconstructed parallel structure, there are $5N$ multipliers and $7N$ adders in the architecture. This approach can be applied to all the parallel post-lattice structures of different orthogonal transforms. For all the lattice architectures, a total of N multipliers can be saved, but $3N$ more adders are required. The latency is N clock cycles and the system remains $SIPO$.

7 Comparisons of the Architecture

From the previous discussions, we see that the proposed unified parallel lattice structures have many attractive features. There are no constraints on the transform size N. It dually generates two discrete transforms the DCT and DST simultaneously and evidences the time evolutions of sequential input vectors. Since it produces the transformed data of subsequent input vector per clock cycle, it is especially efficient for systems with series input data such as communication systems. Further, the structure is regular, module, and without global communication. As a consequence, it is suitable for VLSI implementation.

Here, we would like to compare our lattice structures of the DCT and DST with those proposed in [14, 15, 7]. The architecture in [14] uses the matrix factorization method which is a repesentive of fast algorithms. In [15], an improved fast structrue with fewer multipliers is proposed. Hou's architecture in [7] uses recusive computations to generate the higher order DCT from the lower order one. The characteristics of these structures are disscused in the introduction. A comparison regarding their inherent properties is listed in Table 1. To be clear, the quantitative comparisons in terms of the parameters, which are the numbers of multipliers, adders, and the latency, are given in Table 2, Table 3, and Table 4.

The lattice architecture with six multipliers in the module as shown in Fig. 4 is called Liu-Chiu1 structure, the one in Fig. 17 is called Liu-Chiu2, and the parallel structure with the double-lattice modules as shown in Fig. 18 is called Liu-Chiu3. The structure in Liu-Chiu1 has $6N - 4$ multipliers, $5N - 1$ adders, and the latency is N. There are $4N$ multipliers, $5N - 1$ adders, and the latency is $2N$ in the structure of Liu-Chiu2. The number of multipliers is reduced by the order $2N$ in the expense of that the latency is double and the data flow becomes $SISO$. The Liu-Chiu3 architecture has $5N$ multipliers and $7N$ adders and the the latency is N clock cycles. From these Tables, it is noted that the number of multipliers in our architectures is higher than that of others when N is small. This is due to the dual gerenation of two transforms (the DCT and DST) at the same time. If we consider one transform only, the number of multipliers needed is $2N$ in Liu-Chiu2's structure which is compatible with Lee's. Since the numbers of multipliers and adders of our structures are on the order N, our algorithms have fewer multipliers and adders than those proposed in [14, 15]. Although Hou's algorithm has the fewest multipliers, his architecture needs global communications and the design complexity is much higher than ours. In addition, it is apparently that our structures are faster than others for series input data systems. The operations of other structures can not start until all of the data in the block arrive.

A comparison for our DHT structure based on the lattice module in Fig. 10 and different DHT algorithms [23, 18] is listed in Table 5. The architecture in [23], a representive fast algorithm, is developed based on the existing FFT method. Chaitali-JaJa's algorithm in [18] decomposes the transform size N into mutually prime numbers and implements them in a systolic manner. Their structure needs extra registers and the latency is higher than others. It is easy to see that our structure is better than others in terms of hardware complexity and speed.

8 Conclusion

In this paper, unified time-recursive algorithms and lattice structures that can be applied to the DCT, DST, DHT and their inverse transforms, are considered. In fact, there are various forms of sin and cosine transform pairs, (the DCTI/DSTI, DCTII/DSTII, DCTIII/DSTIII, and DCTIV/DSTIV) as mentioned by Yip and Rao in [22]. They also have their time-recursive lattice realizations. The procedures to attain the lattice structures of different transforms are similar and the resulting $SIPO$ lattice stuctures differ only in the multiplying coefficients and the input stage. All the transform pairs have their pre- and post-lattice realizations that differ in that the input signals are added in the front and the end of the lattice respectively. The hardware complexity of the pre-lattice realizations and their post-lattice counterparts depends on the definitions of the transforms and it cannot be readily determined which one is better. The number of multiplers in these parallel lattice structures is a linear function of the transform size N and the latency is N clock cycles. Two methods, the $SISO$ and double-lattice approaches, are developed to reduce the number of multipliers for the parallel lattice structures. The $SISO$ approach can reduce $2N$ multipliers and the latency becomes $2N$. The double-lattice approach can reduce N mutipliers and the the latency remains intact. From the discussion of the block processing, it is noted that the area-time complexity becomes more efficient when the block size m is increased. That is, if the hardware resource is allowed, we can trade the hardware for better time efficiency. All the resulting parallel structures are module, regular, and only locally connected. Further, there is no any constraint on the transform size N. It is obvious that the design complexity of these structures is relatively low compared with other algorithms. The charteristics of these algorithms are suitable for processing series input data since the transformed data for sequential input can be obtained per clock cycle. Therefore, it is very attractive to VLSI implementations and high speed applications such as HDTV signal coding and transmission.

References

[1] A. Rosenfeld, and A. C. Kak, *Digital Picture Processing*, 2nd edition, Academic Press, 1982.

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

[3] R. J. Clark, "Relation between the Karhunen-Loeve and cosine transform," IEE Proc., vol. 128, pt. F, no. 6, pp. 359-360, Nov. 1981.

[4] A. K. Jain, "A fast Karhunen-Loeve transform for a class of stochastic process," IEEE Trans., Commun., vol. COM-24, pp.1023-1029, 1976.

[5] R. Yip and K. R. Rao, "On the computation and effectiveness of discrete sine transform," Comput. Elec. Eng., vol. 7, pp. 45-55, 1980.

[6] R. N. Bracewell, "Discrete Hartley transform," J. opt. Soc. Amer., vol. 73, pp. 1832-1835, Dec. 1983.

[7] 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.

[8] R. N. Bracewell, "The fast Hartley transform," Proc. IEEE, vol. 72, pp1010-1018, Aug. 1984.

[9] B. D. Tseng, and W. C. Miller, "On computing the discrete cosine transform," IEEE Trans. on Computer, vol. C-27, pp. 966-968, July 1976.

[10] M. Vetterli and H. Nussbaumer, "Simple FFT and DCT algorithm with reduced number of operations," Signal Processing, vol. 6, no. 4, pp. 267-278, Aug. 1984.

[11] M. Vetterli, and A. Ligtenberg, "A discrete Fourier-Cosine transform chip," IEEE Journal on Selected Areas in Communications, vol. SAC-4, No. 1, pp. 49-61. Jan. 1986.

[12] H. W. Jones, D. N. Hein, and S. C. Knauer, "The Karhunen-Loeve, discrete cosine and related transform via the Hadmard transform," in Proc. Inc. Telemeter, Conf., Los Angeles, CA, pp. 87-98, Nov. 1978.

[13] M. J. Narashimha and A. M. Peterson, "On the computation of the discrete cosine transform," IEEE Trans. Communication, vol. COM-26, pp. 934-936, June 1978.

[14] W. H. Chen, C. H. Smith, and S. C. Fralick, ' 'A fast computational algorithm for the discrete cosine transform," IEEE Trans. Communication, vol. COM-25, pp. 1004-1009, Sept. 1977.

[15] 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.

[16] B. G. Kashef and A. Habibi, "Direct computation of higher-order DCT coefficients from lower-order DCT coefficients," SPIE 28th Annu. Int. Tech. Symp., San Diego, CA, Aug. 19-24, 1984.

[17] M. H. Lee, "On computing 2-D systolic algorithm for discrete cosine transform," IEEE Trans. on Circuit and Systems, vol-37, No. 10, pp. 1321-1323, Oct. 1990.

[18] C. Chakrabarti, and J. J'aJ'a, "Systolic architectures for the computation of the discrete Hartley and the discrete cosine transforms based on prime factor decomposition," IEEE Trans. on Computer, vol. 39, No. 11, pp. 1359-1368, Nov. 1990.

[19] H. S. Hou, "The fast Hartley transform algorithm," IEEE Trans. on Computer, vol. C-36, No. 2, pp. 147-156, Feb. 1987.

[20] Z. Wang, "Fast algorithms for the discrete W transform and for the discrete Fourier transform," IEEE Trans. Acoust., Speech, Signal processing, vol. ASSP-32, Aug. 1984.

[21] R. W. Owens and J. J'aJ'a, "A VLSI chip for the Winograd/Prime factor algorithm to compute the discrete Fourier transform," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-34, pp. 979-989, Aug. 1986.

[22] R. Yip and K. R. Rao, "On the shift property of DCT's and DST's," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-35, No. 3, pp. 404-406, March. 1987.

[23] H. V. Sorenson, *et al.*, "On computing the discrete Hartley transform," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-33, No. 4, pp. 1231-1238, Oct. 1985.

[24] S. B. Narayanan and K. M. M. Prabhu, "Fast Hartley transform pruning," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-39, No. 1, pp. 230-233, Jan. 1991.

[25] W. Kou and J.W. Mark, "A new look at DCT-type transforms," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-37, No. 12, pp. 1899-1908, Dec. 1989.

[26] N. I. Cho and S. U. Lee, "DCT algorithms for VLSI parallel implementations," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-38, No. 1, . 1899-1908, Dec. 1989.

[27] L. W. Chang and M. C. Wu, "A unified systolic array for discrete cosine and sine transforms," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-39, No. 1, pp. 192-194, Jan. 1991.

[28] E. A. Jonckheere and C. Ma, "Split-radix fast Hartley transform in one and two dimensions," IEEE Trans. Acous., Speech, Signal Processing, vol. ASSP-39, No. 2, pp. 499-503, Feb. 1991.

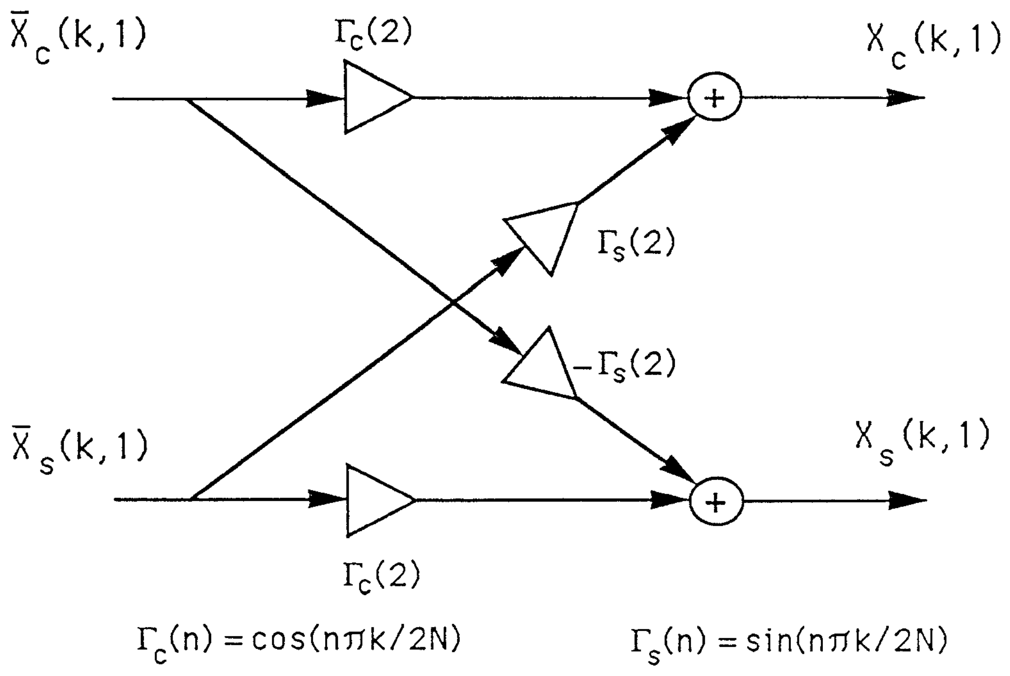


Figure 1: The lattice module.

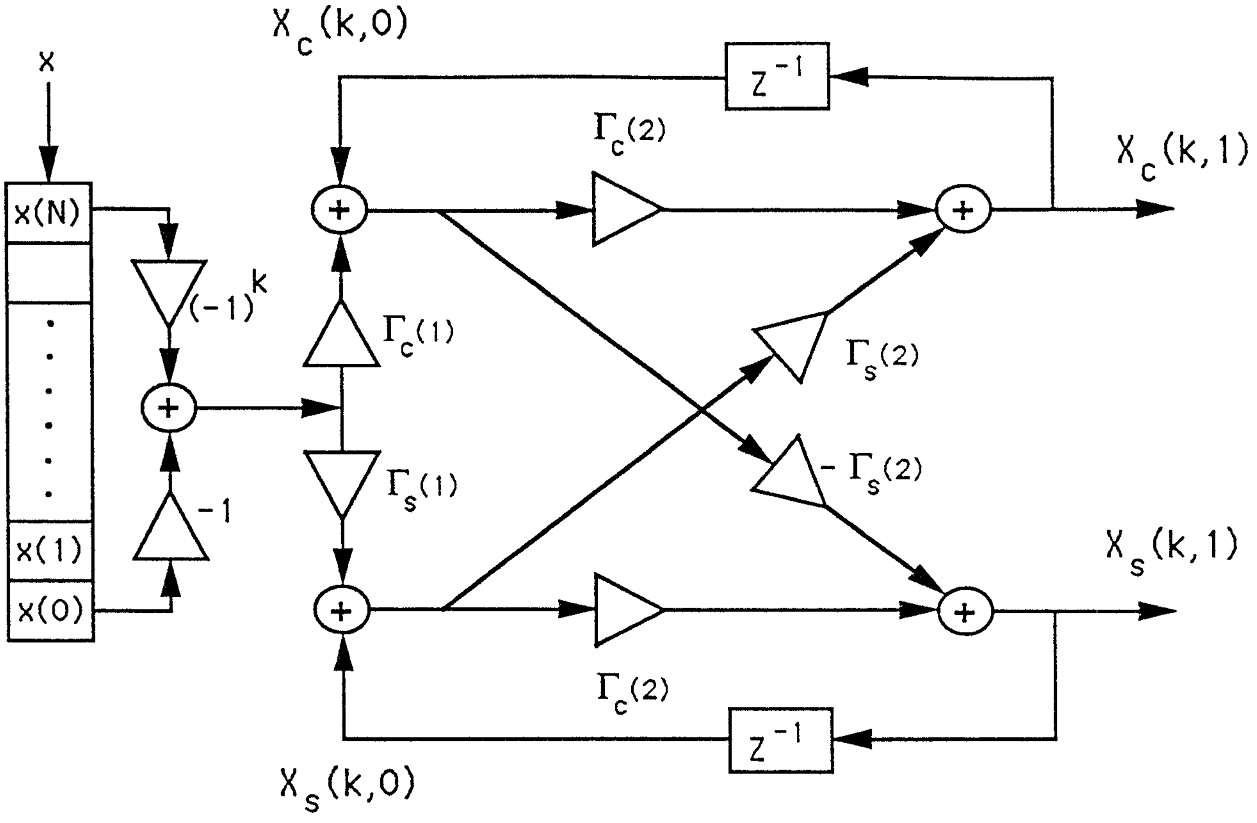


Figure 2: The lattice structure for the DCT and DST, $k = 1, 2, ..., N-1$.

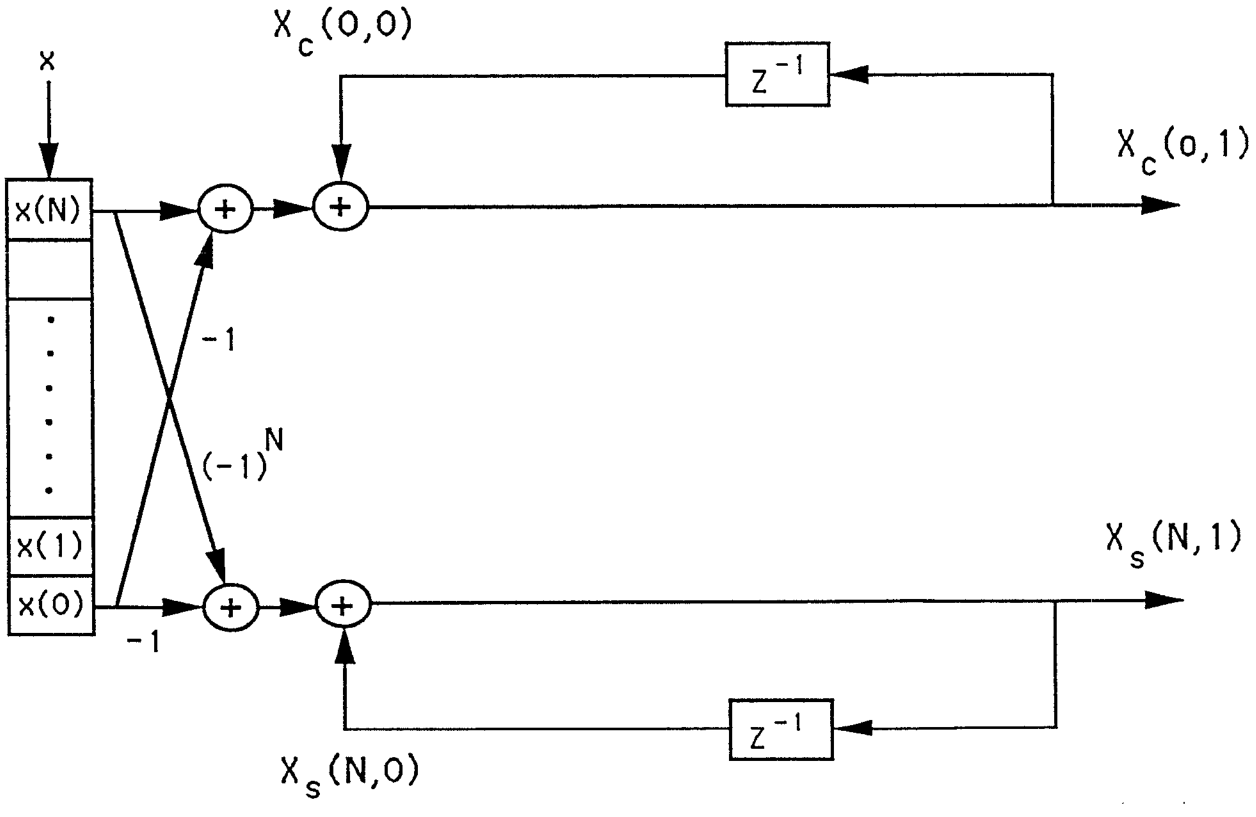


Figure 3: The lattice structure of $k = 0$ for the DCT and $k = N$ for the DST.

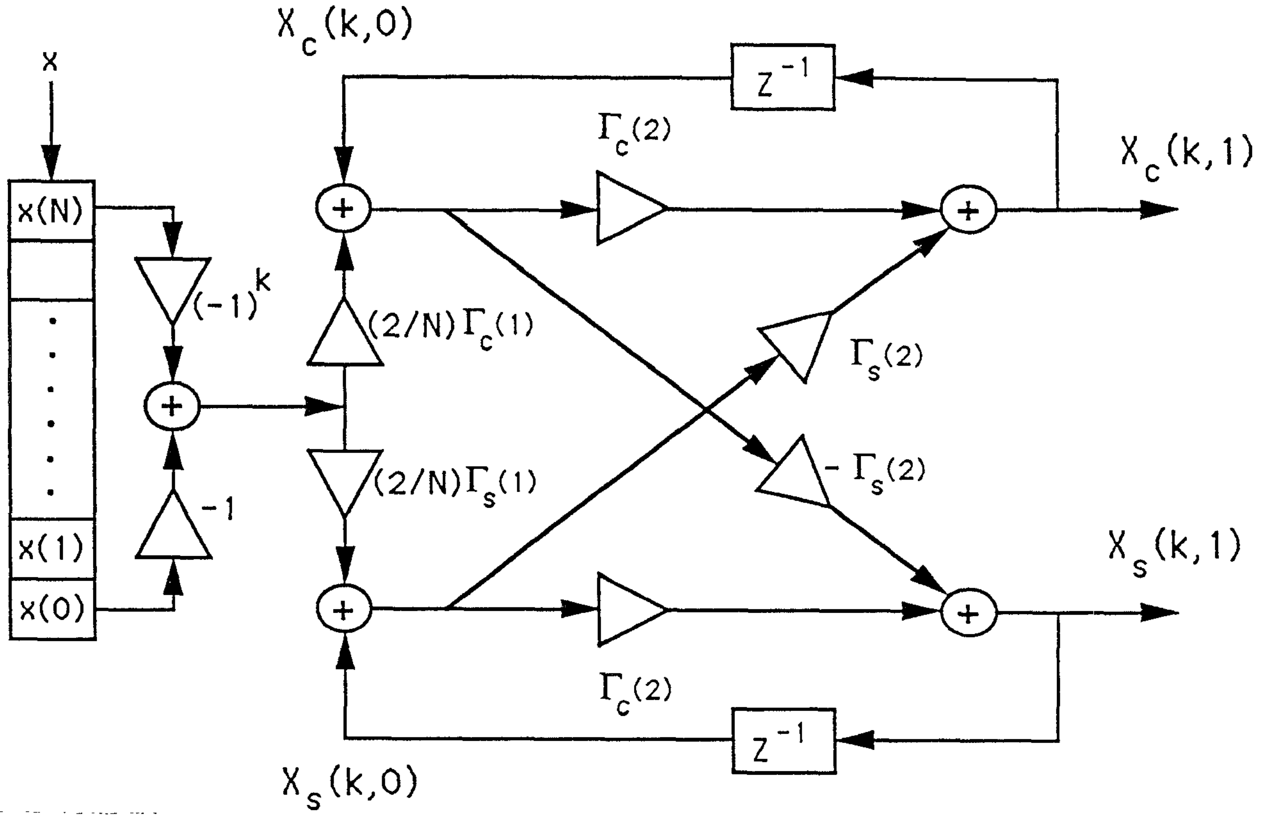


Figure 4: The lattice structure for the DCT and DST with coefficients $C(k)$'s and $D(k)$'s, $k = 1, 2, ..., N-1$.

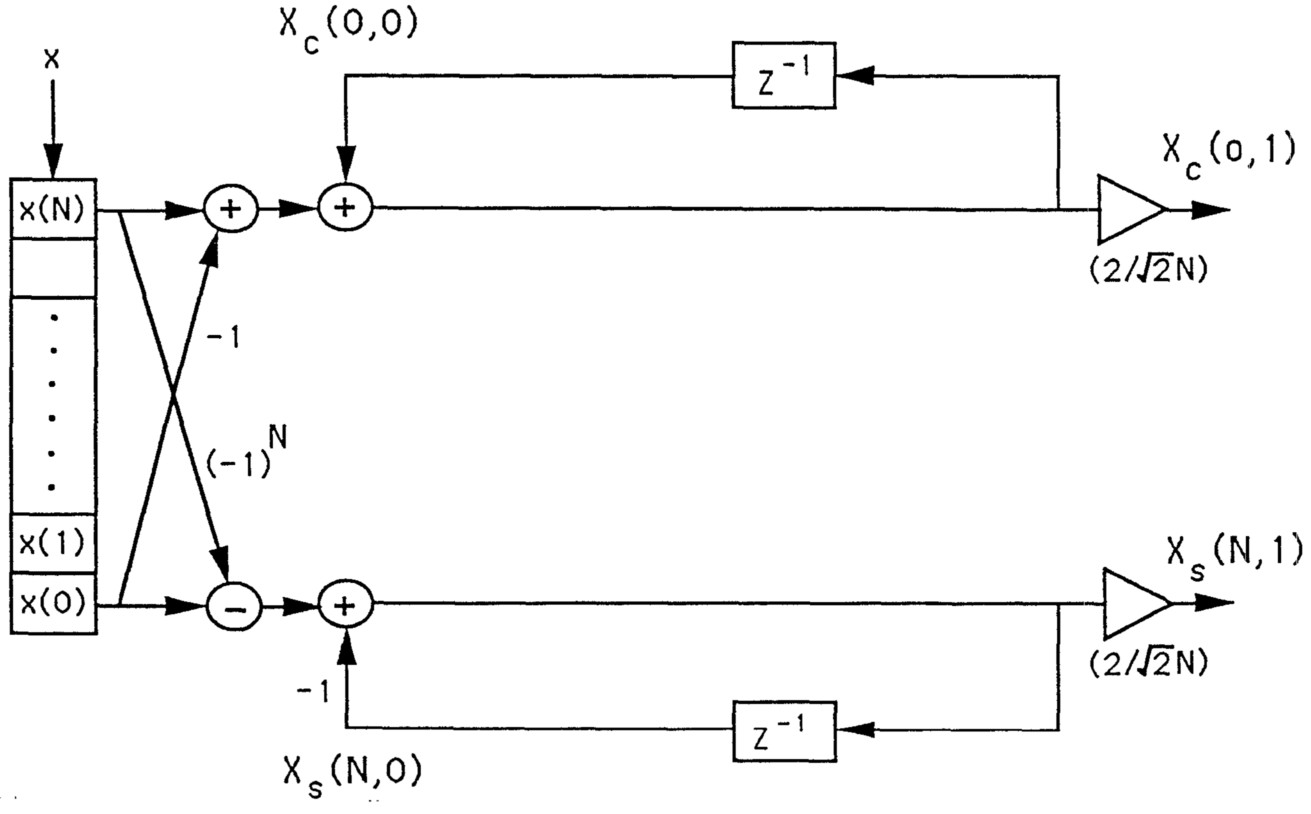


Figure 5: The lattice structure of $k = 0$ for the DCT and $k = N$ for the DST with coefficient $C(0)$ and $D(N)$.

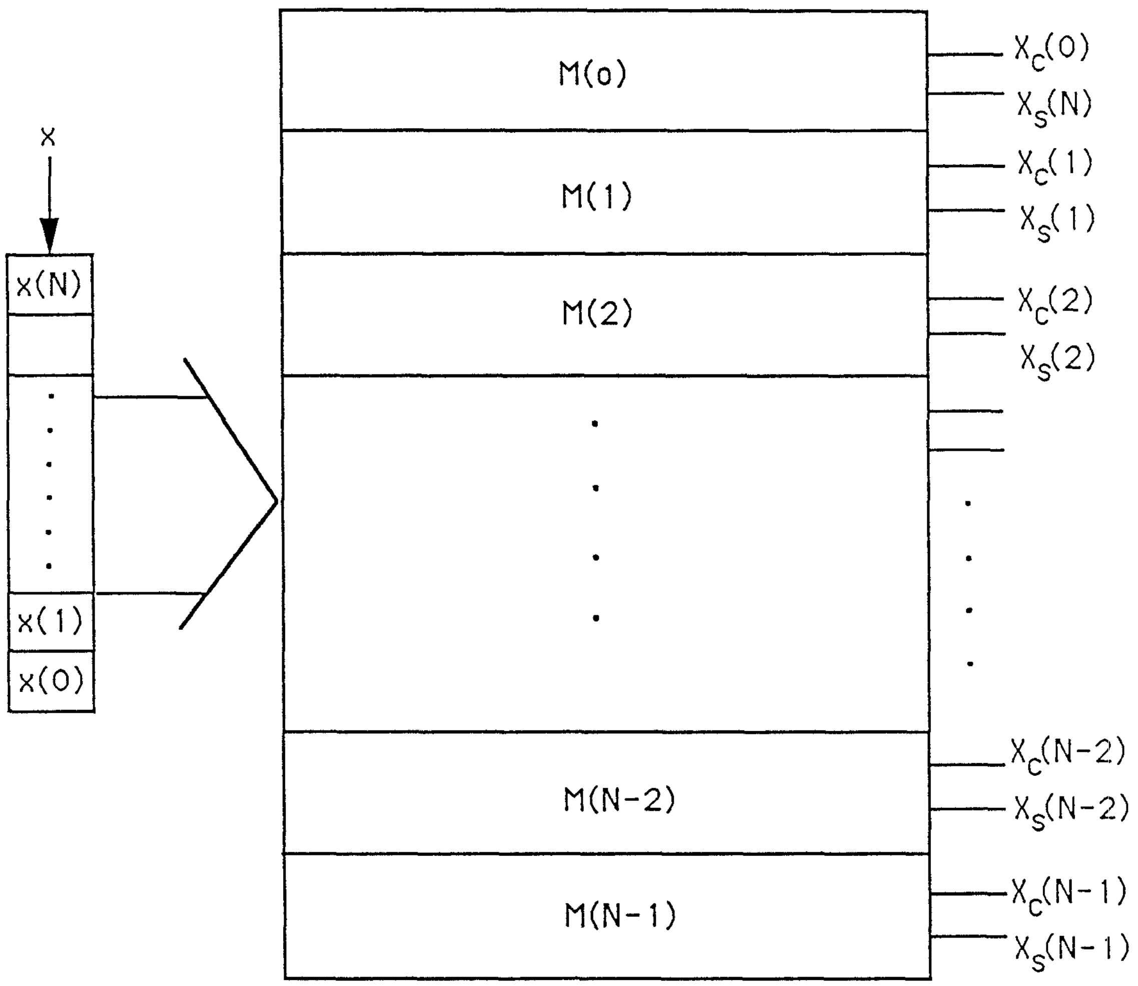


Figure 6: The parallel lattice structure for the DCT and DST.

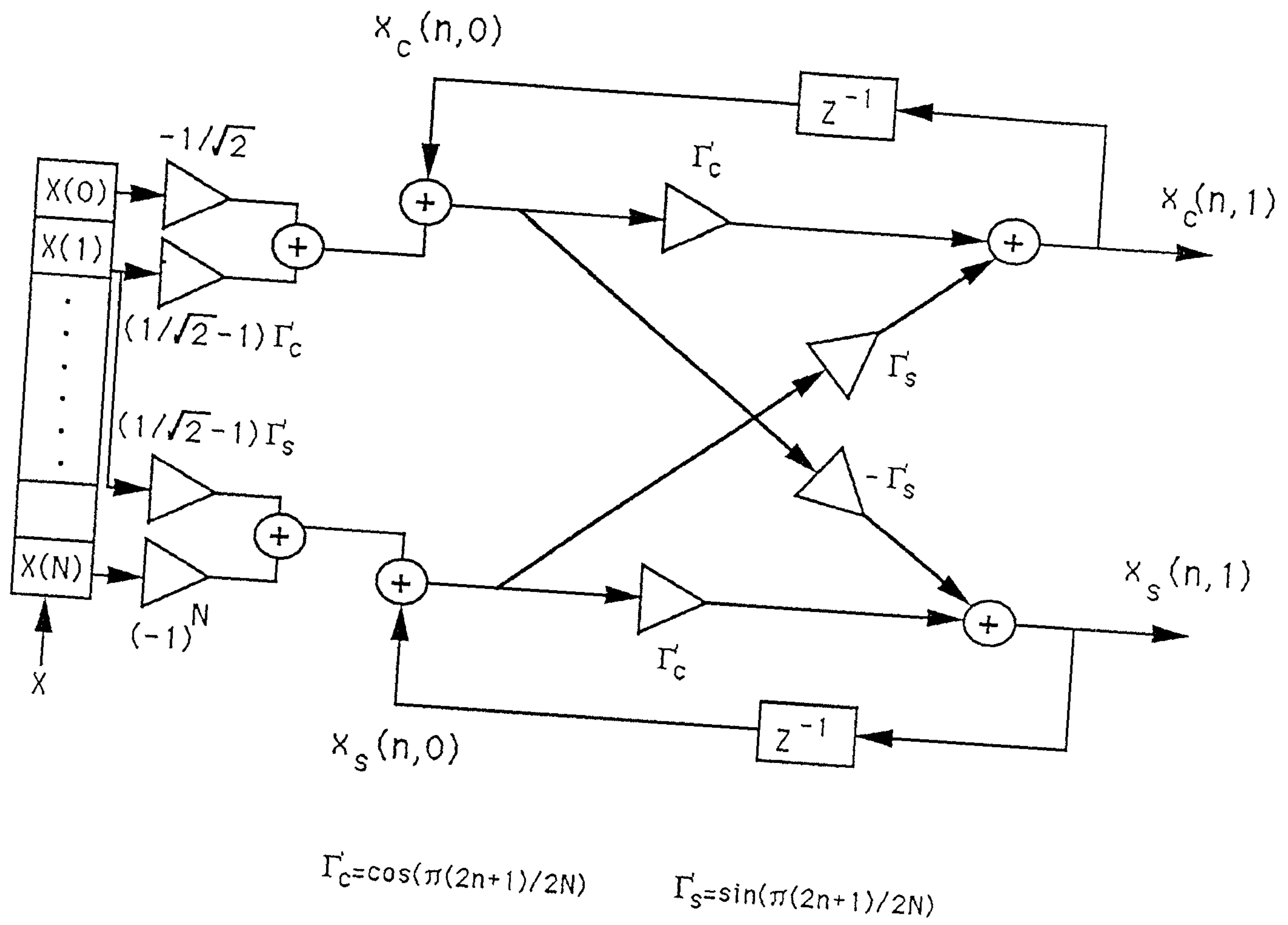


Figure 7: The lattice structure for the IDCT and AIDST.

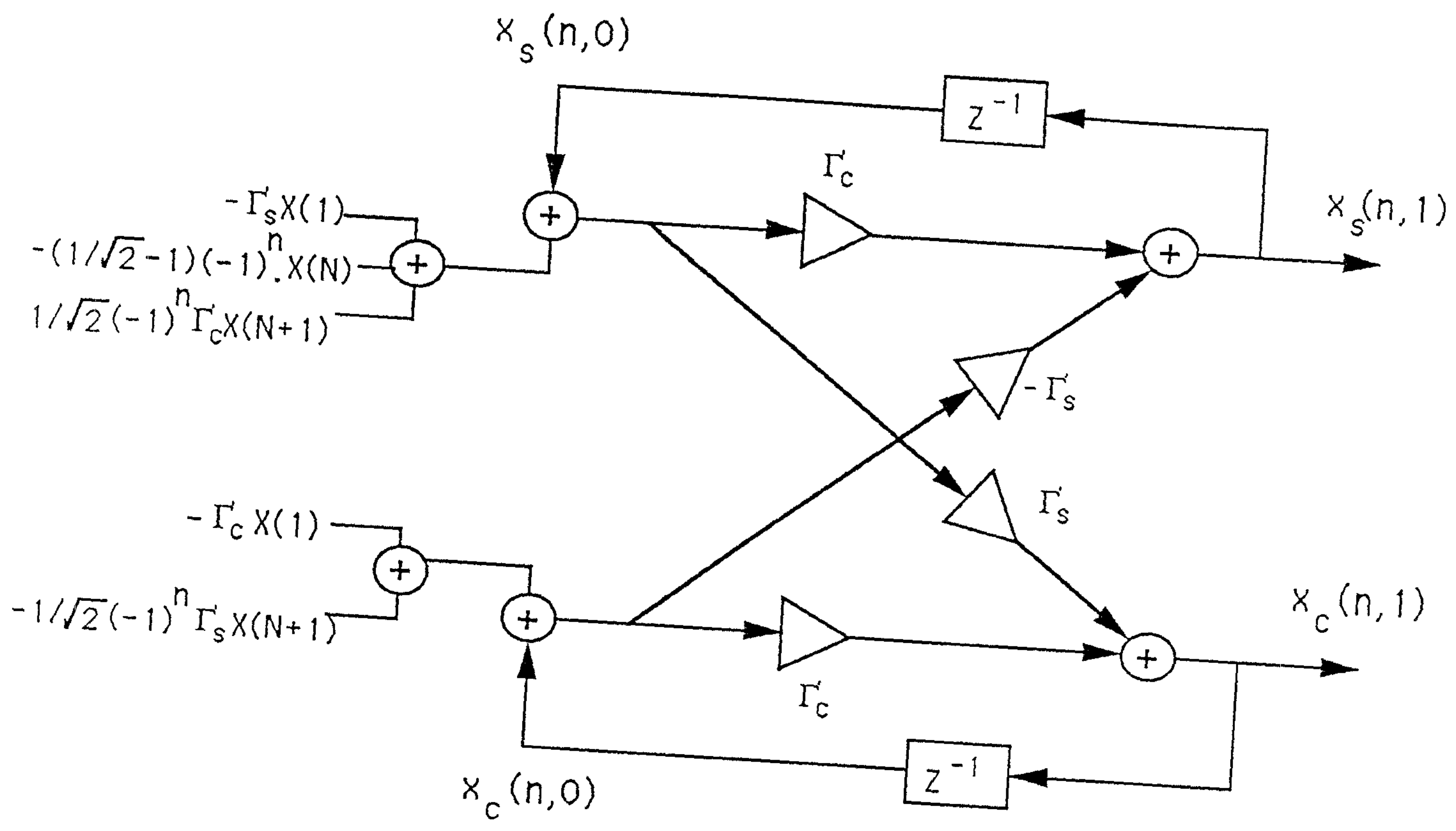
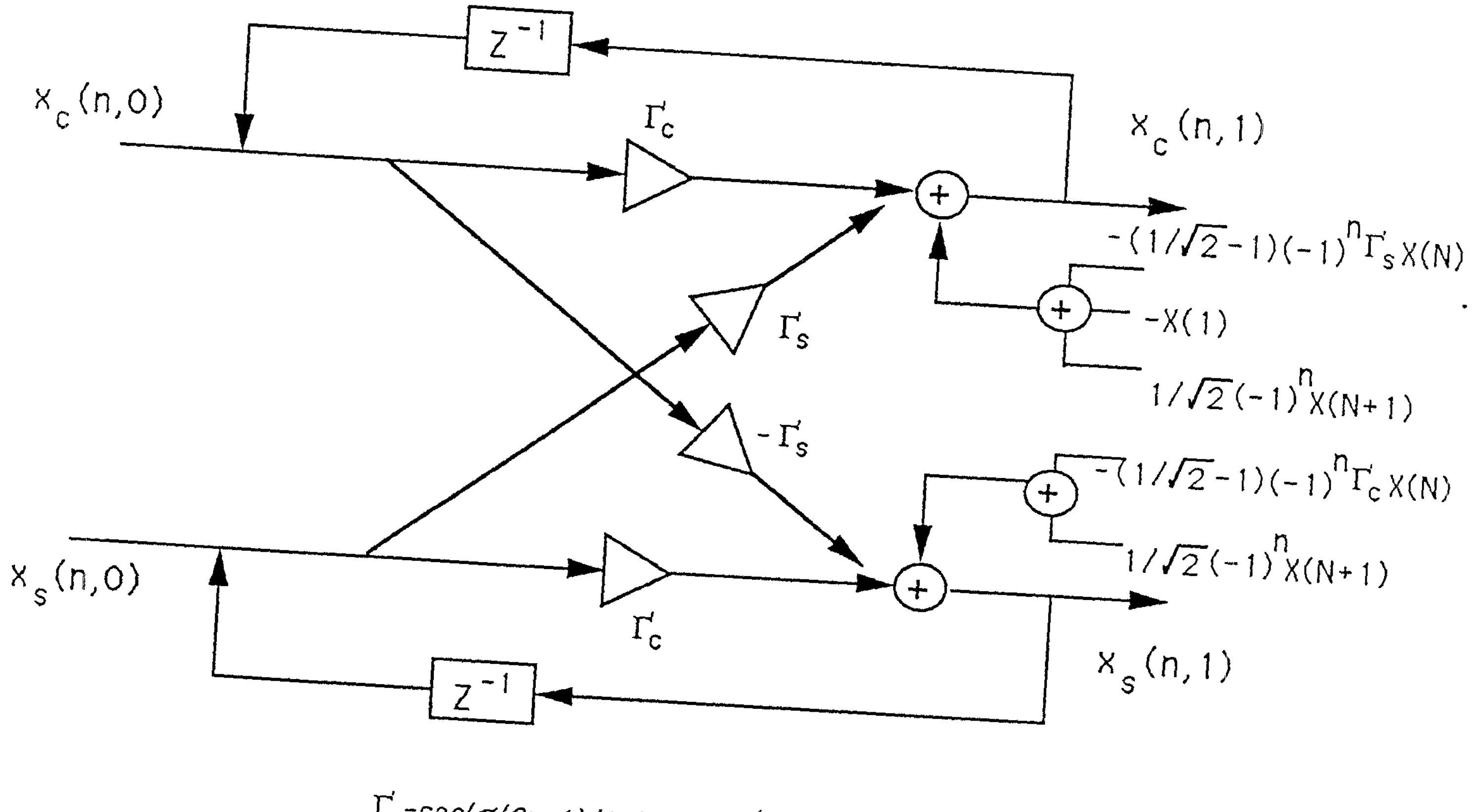


Figure 8: The pre-lattice structure for the IDST and AIDCT.


z^{-1}
$x_c(n,0)$
Γ_c
$x_c(n,1)$
$-(1/\sqrt{2}-1)(-1)^n \Gamma_s X(N)$
$-X(1)$
$1/\sqrt{2}(-1)^n X(N+1)$
Γ_s
$-\Gamma_s$
$-(1/\sqrt{2}-1)(-1)^n \Gamma_c X(N)$
$1/\sqrt{2}(-1)^n X(N+1)$
$x_s(n,0)$
Γ_c
$x_s(n,1)$
z^{-1}

$\Gamma_c = \cos(\pi(2n+1)/2N)$ $\Gamma_s = \sin(\pi(2n+1)/2N)$

Figure 9: The post-lattice structure for the IDST and AIDCT.

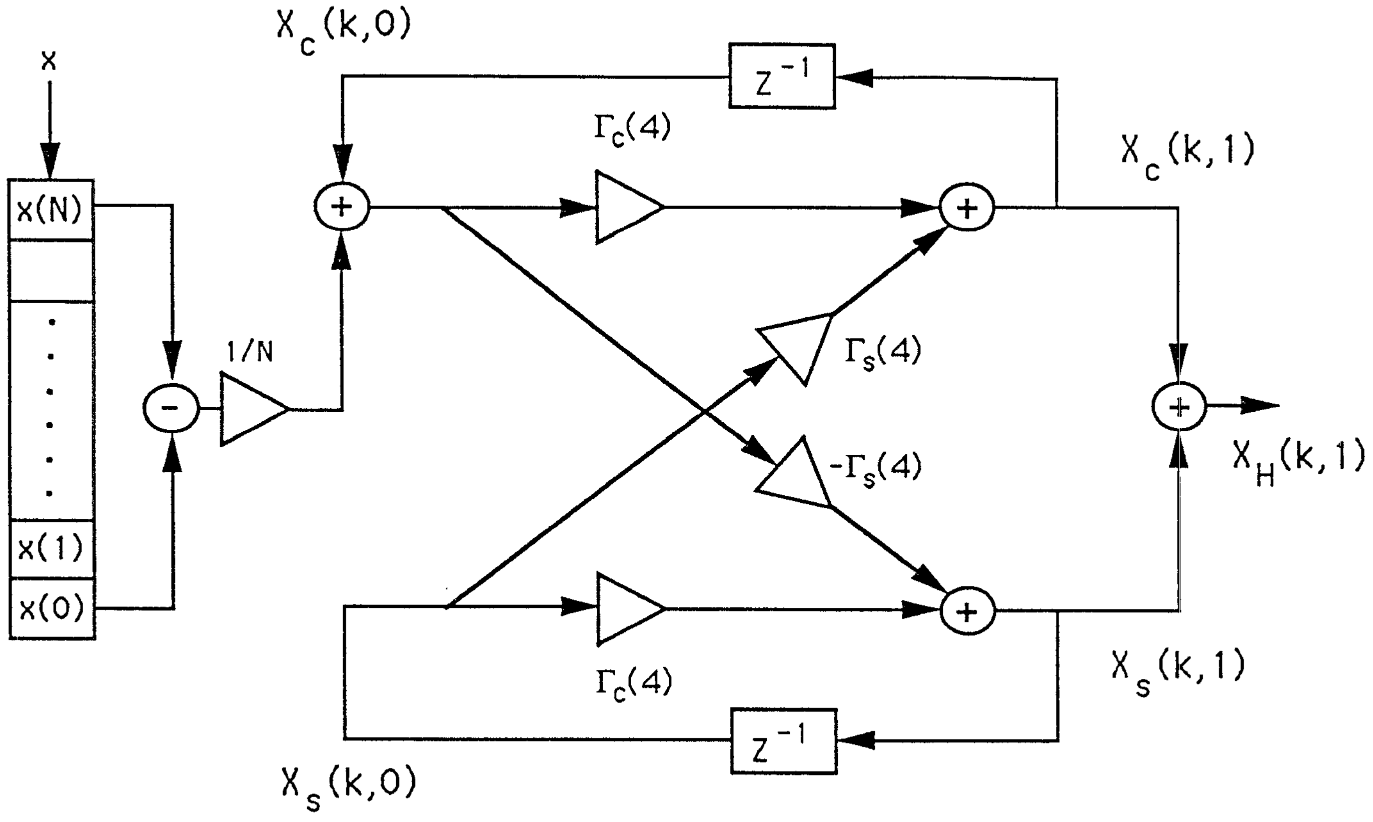


Figure 10: The lattice structure for the DHT for $k = 1, 2..., N-1$.

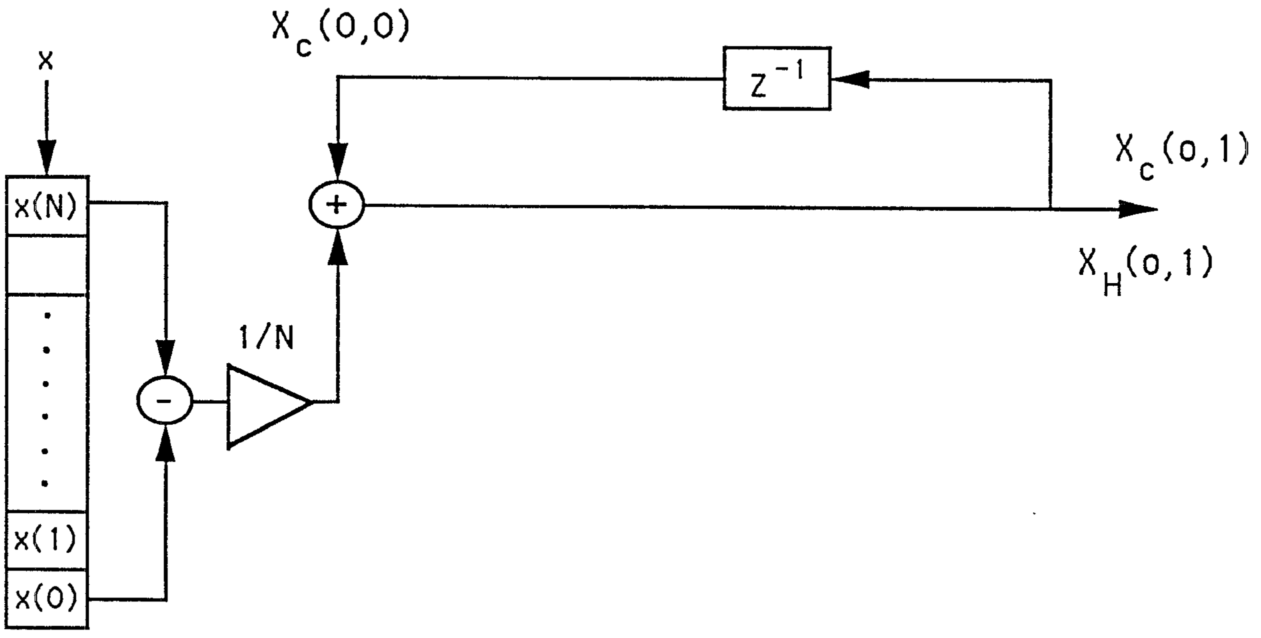


Figure 11: The lattice structure for the DHT for $k = 0$.

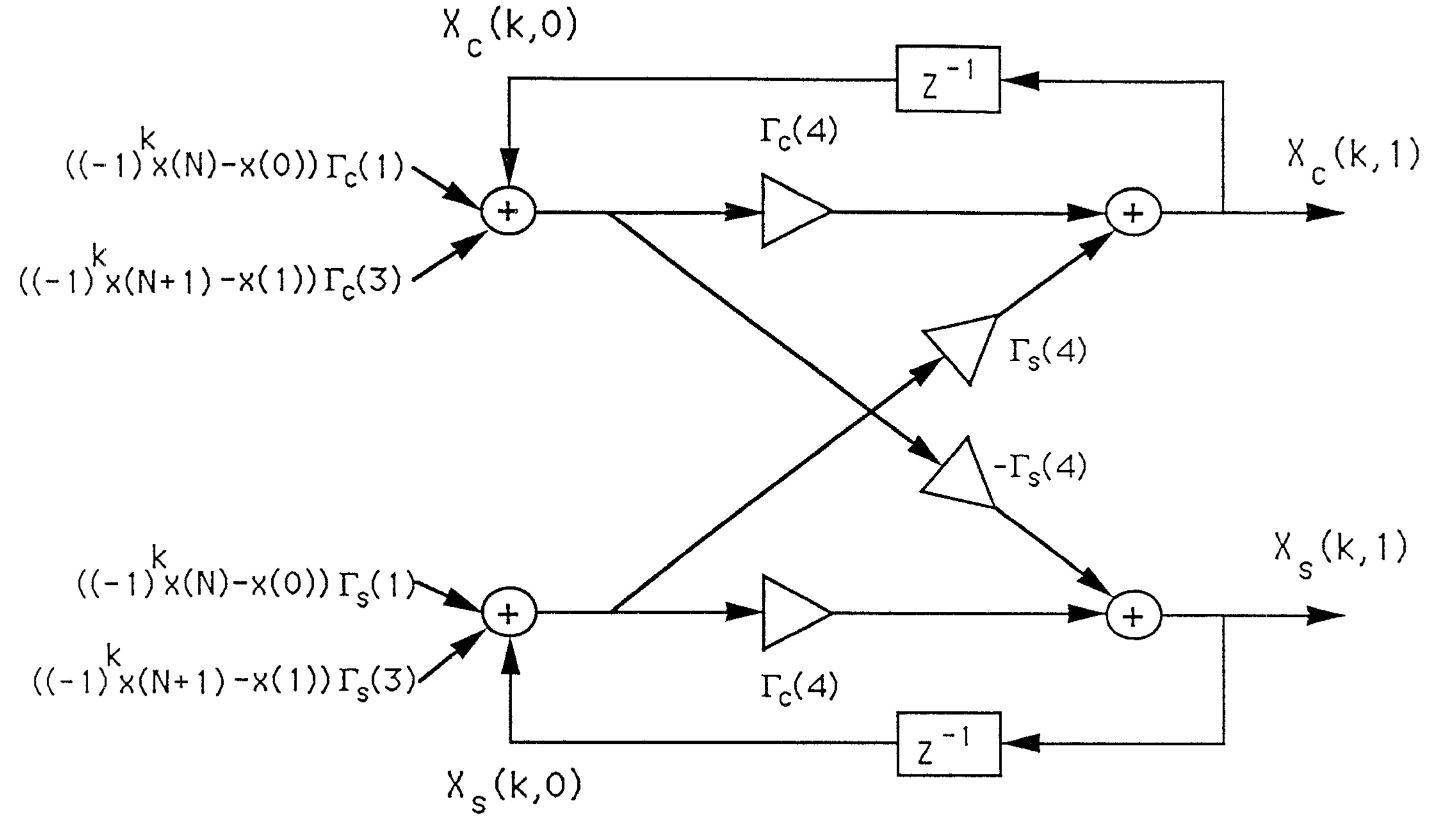

$X_c(k,0)$
$((-1)^k x(N)-x(0))\Gamma_c(1)$
$((-1)^k x(N+1)-x(1))\Gamma_c(3)$
z^{-1}
$\Gamma_c(4)$
$X_c(k,1)$
$\Gamma_s(4)$
$-\Gamma_s(4)$
$X_s(k,1)$
$((-1)^k x(N)-x(0))\Gamma_s(1)$
$((-1)^k x(N+1)-x(1))\Gamma_s(3)$
$\Gamma_c(4)$
z^{-1}
$X_s(k,0)$

Figure 12: The lattice structure for block-size-two operation on the DCT and DST.

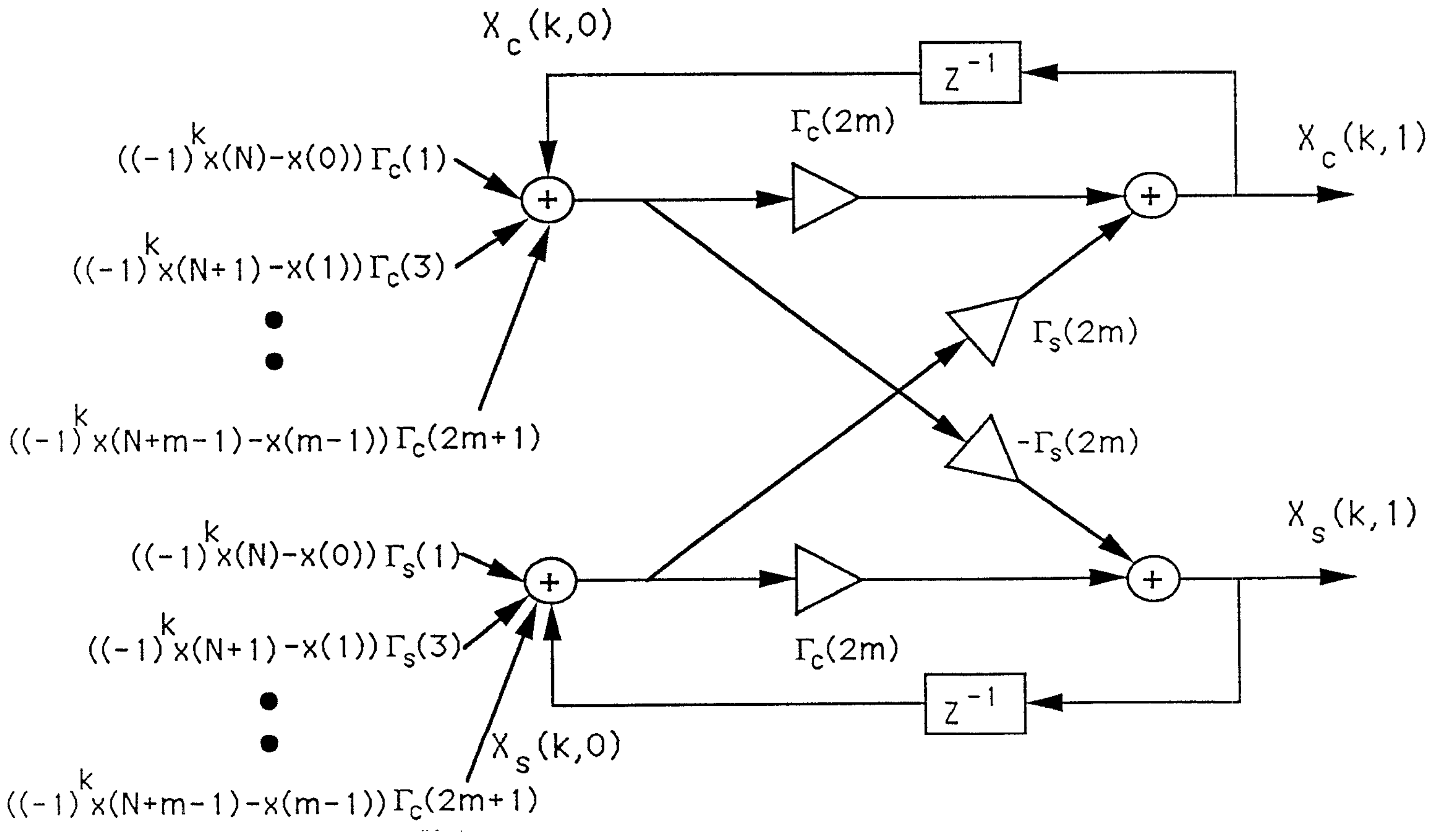

$X_c(k,0)$
z^{-1}
$\Gamma_c(2m)$
$X_c(k,1)$
$((-1)^k x(N)-x(0))\Gamma_c(1)$
$((-1)^k x(N+1)-x(1))\Gamma_c(3)$
$\Gamma_s(2m)$
$((-1)^k x(N+m-1)-x(m-1))\Gamma_c(2m+1)$
$-\Gamma_s(2m)$
$X_s(k,1)$
$((-1)^k x(N)-x(0))\Gamma_s(1)$
$((-1)^k x(N+1)-x(1))\Gamma_s(3)$
$\Gamma_c(2m)$
z^{-1}
$X_s(k,0)$
$((-1)^k x(N+m-1)-x(m-1))\Gamma_c(2m+1)$

Figure 13: The lattice structure for block-size-m operation on the DCT and DST.

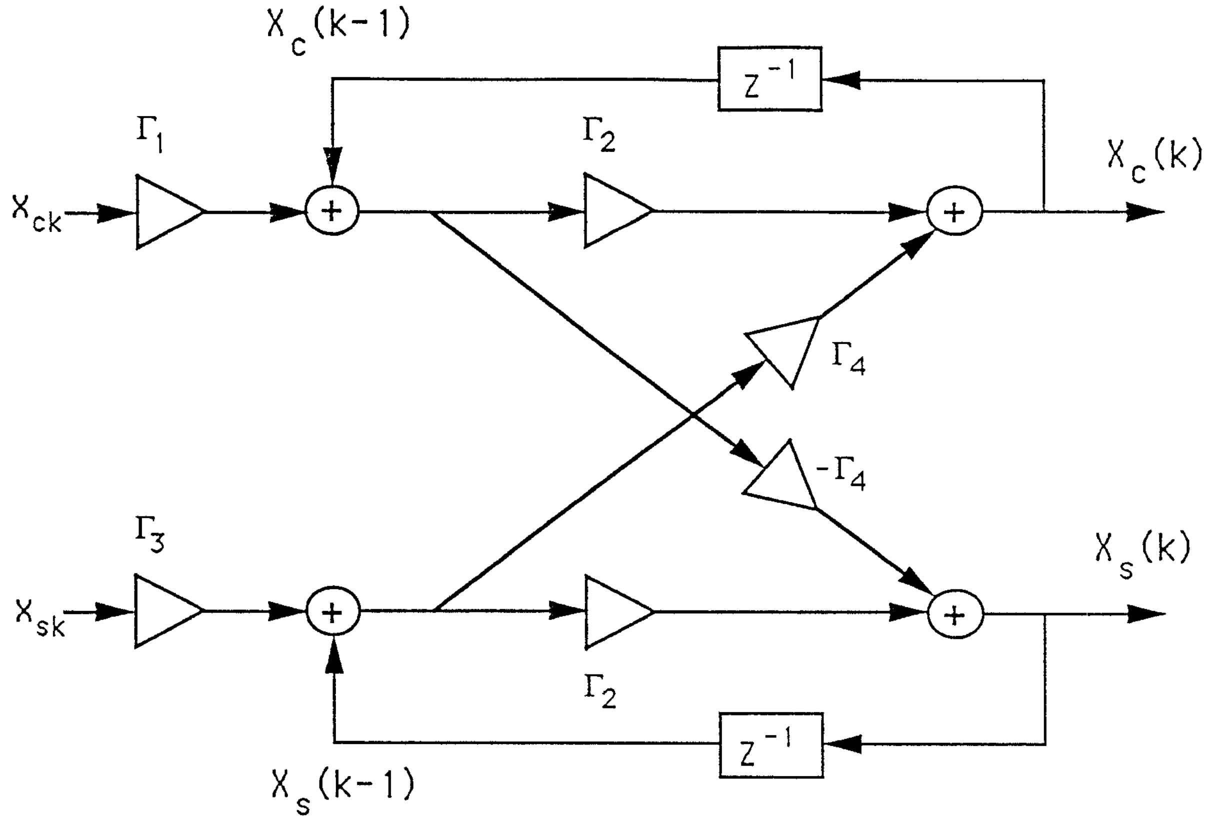


Figure 14: The general lattice module.

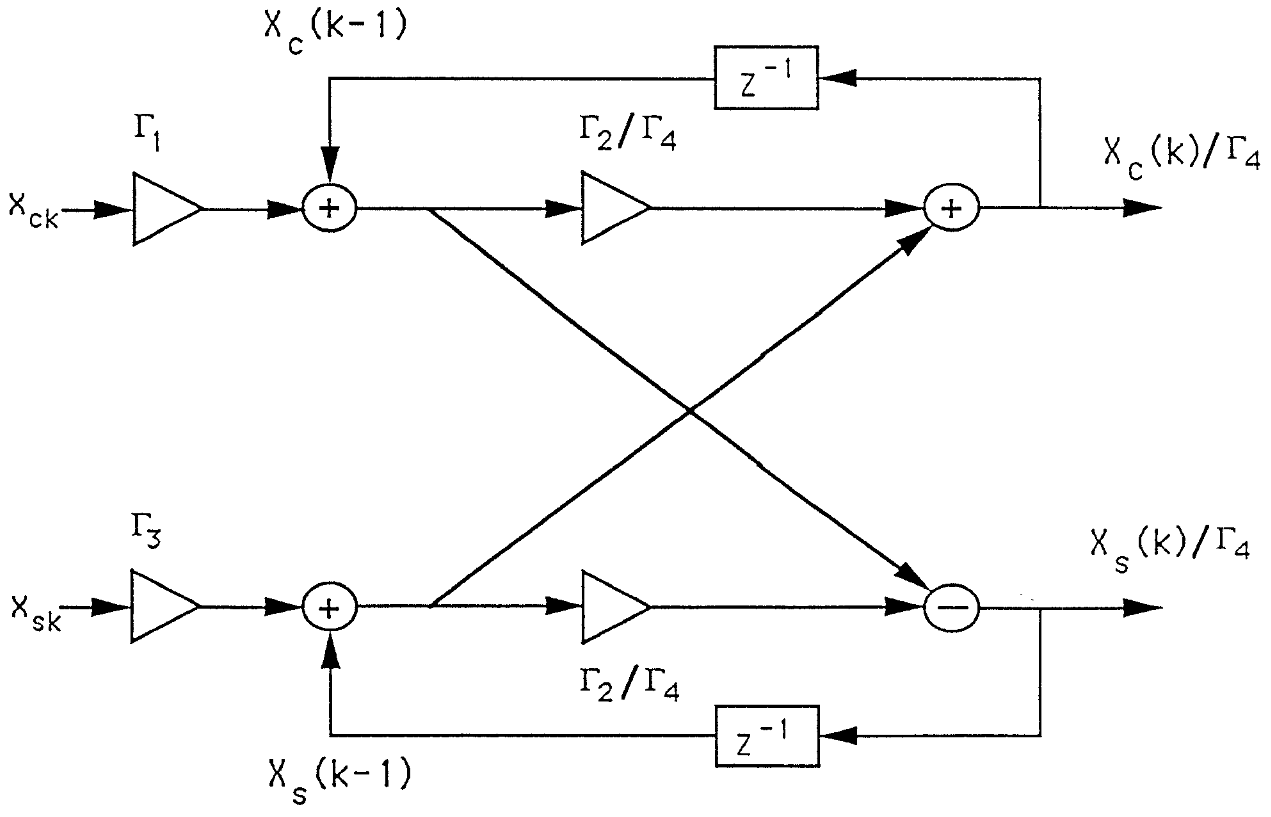


Figure 15: The model of multiplier-reduction.

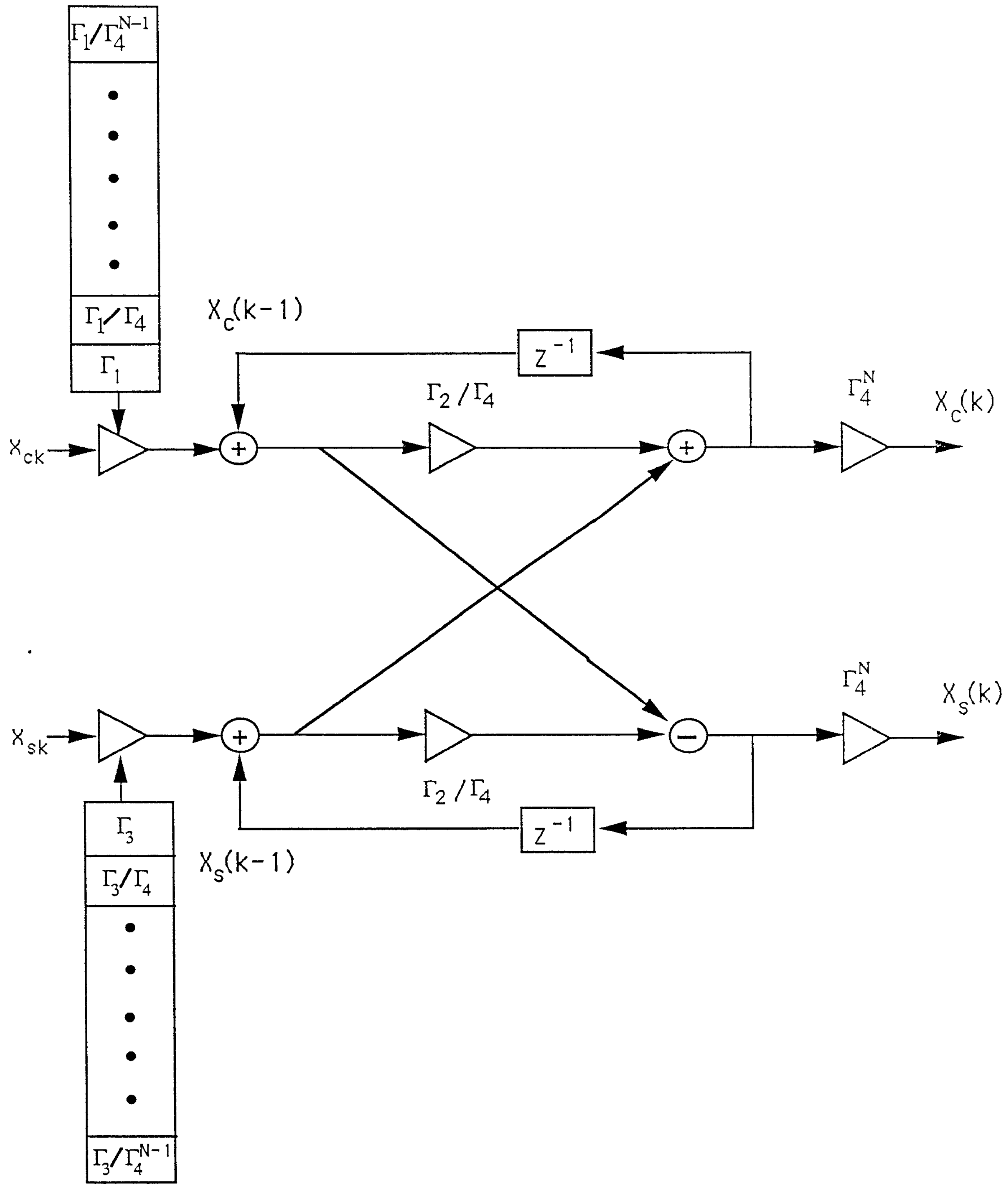


Figure 16: The multiplier-reduced lattice module.

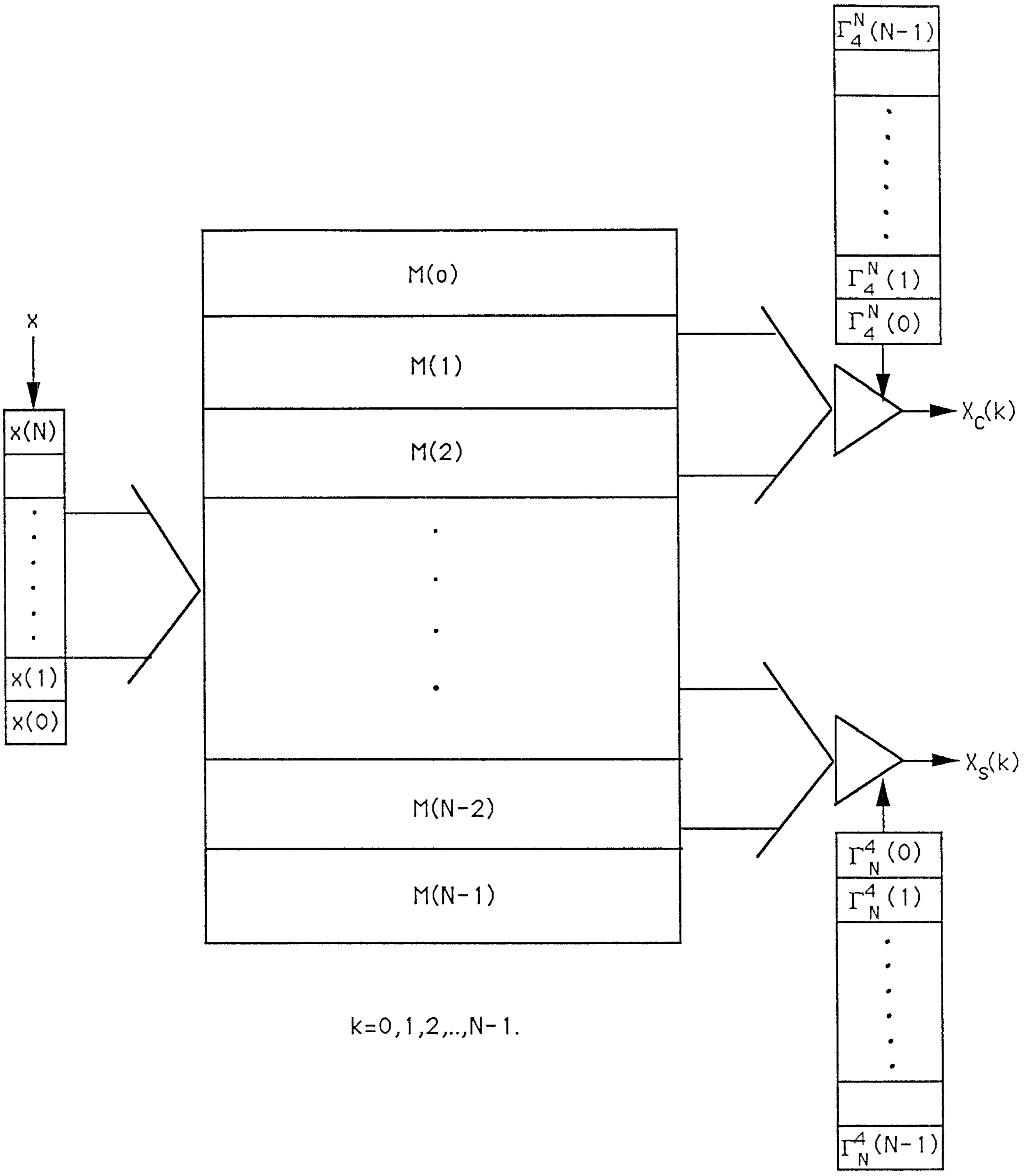
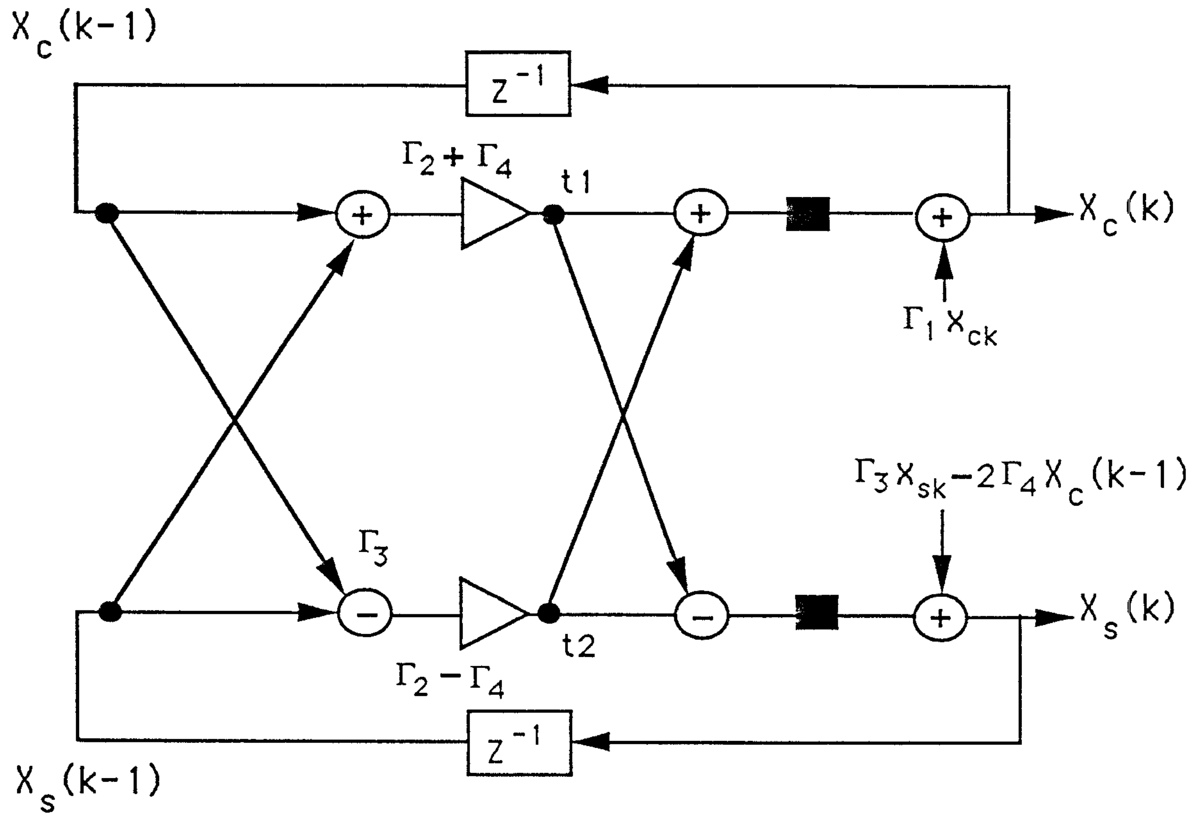


Figure 17: The complete parallel multiplier-reduced lattice structure.



■: denotes left shift one bit

Figure 18: The double-lattice form of the post-lattice realization.

	Liu-Chiu1	Liu-Chiu2	Liu-Chiu3	Chen [14] *et. al.*	Lee[15]	Hou[7]
No. of multipliers	$6N-4$	$4N$	$5N-3$	$N\ln(N)$ $-3N/2+4$	$N/2$ $*\ln(N)$	$N-1$
Latency	N	$2N$	N	$N/2$	$(\ln(N)-1)$ $*\ln(N)/2$	$3N/2$ (order)
Limitation on transform size N	no	no	no	power of 2	power of 2	power of 2
Communication	local	local	local	global	global	global
I/O operation	*SIPO*	*SISO*	*SIPO*	*PIPO*	*PIPO*	*SIPO*

Table 1: Comparisions of different DCT algorithms.

NO	Liu-Chiu1	Liu-Chiu2	Liu-Chiu3	Chen *et. al.*	Lee	Hou
8	44/2	32/2	37/2	16	12	7
16	92/2	64/2	77/2	44	32	15
32	188/2	128/2	157/2	116	80	31
64	380/2	256/2	317/2	292	192	63

Table 2: Comparisions of the number of multipliers.

NO	Liu-Chiu1	Liu-Chiu2	Liu-Chiu3	Chen *et. al.*	Lee	Hou
8	39/2	39/2	52/2	26	29	18
16	79/2	79/2	108/2	74	81	41
32	159/2	159/2	220/2	194	209	88
64	319/2	319/2	444/2	482	513	183

Table 3: Comparisions of the number of adders.

NO	Liu-Chiu1	Liu-Chiu2	Liu-Chiu3	Chen *et. al.*	Lee	Hou
8	8	16	8	4	6	13
16	16	32	16	6	10	21
32	32	64	32	8	15	44
64	64	128	64	10	21	73

Table 4: Comparisions of the latency.

	Liu-Chiu	Sorenson *et. al.* [23]	Chakrabari-JaJa[18]
No. of multipliers	$5N-4$	$N\ln(N)-3N+4$	$2N$
No. of Adders	$5N-3$	$3N\ln(N)-3N+4$	$4N+\sqrt{N}$
Latency	N	$N\ln(N)$	$2N$
Limitation on transform size	no	power of 2	$N = N1 * N2$, $N1$ and $N2$ are mutually prime
Communication	local	global	local
I/O operation	$SIPO$	$PIPO$	$SISO$

Table 5: Comparisions of different DHT algorithms.