Real-Time Parallel and Fully Pipelined Two-Dimensional DCT Lattice Structures with Application to HDTV Systems

Ching-Te Chiu, *Student Member, IEEE,* and K. J. Ray Liu, *Member, IEEE*

Abstract—**The two-dimensional discrete cosine transform (2-D DCT) has been widely recognized as the most effective technique in image data compression, especially for high-speed video processing applications, such as high-definition television (HDTV). In this paper, we propose a new fully pipelined architecture to compute the 2-D DCT from a frame-recursive point of view. Based on this approach, two real-time parallel lattice structures for successive frame and block 2-D DCT are developed. These structures are fully-pipelined with throughput rate N clock cycles for an N × N successive input data frame. These are the fastest pipelined structures known so far. Moreover, the resulting 2-D DCT architectures are modular, regular, and locally connected and require only two 1-D DCT blocks that are extended directly from the 1-D DCT structure without transposition. It is therefore very suitable for VLSI implementation for high-speed HDTV systems. We also propose a parallel 2-D DCT architecture and a new scanning pattern for HDTV systems to achieve higher performance. The VLSI implementation of the 2-D DCT using distributed arithmetic to increase computational efficiency and reduce round-off error is also discussed.**

I. Introduction

In recent years, a considerable amount of research has been focused on image data compression, which plays a significant role in image/signal processing and transmission, especially for the next generation television—high-definition television (HDTV). Image data compression can be classified into three categories: the predictive coding, transform coding, and hybrid coding [8]. In high-speed image processing, transform coding has gained the most attention due to a better compressional capability to achieve bit-rate reduction. The Karhunen-Loeve transform (KLT), which minimizes the mean square error of the system, is the optimal transform among various transform codings but is rarely employed because of its computational complexity. A suboptimal transform, the discrete cosine transform (DCT), which possesses superior energy compaction property and near optimal performance with much simpler computations, is the most popular transform coding in image processing.

It is well known that block coding based on the two dimensional (2-D) DCT produces highly compact 2-D transformed coefficients on the spatial domain [3]. By applying appropriate bit allocations and entropy coding schemes, i.e., variable length coding and run length coding [28], [34], the bit rate of the HDTV systems can be greatly reduced [23], [28]. Because of the hardware limitations in practical applications, only small transform block size (typically 8 × 8 or 16 × 16) is used.

Many 2-D DCT algorithms have been proposed to reduce the computational complexity and to increase the operational speed. These algorithms can be divided into two groups, the row-column method and the direct 2-D method. The row-column method computes the 2-D DCT by applying the one-dimensional (1-D) DCT on the rows (or columns) of the input data frames, storing the transformed results in an intermediate matrix, transposing the matrix, and performing the 1-D DCT again on the columns (or rows) of the transposed matrix. The 2-D DCT, then, is decomposed into two 1-D DCT's. Since there exist many 1-D DCT algorithms, there are also many realizations for the row-column methods. The performance of a specific row-column method depends on the realization of the 1-D DCT algorithm. The classification of the 1-D DCT algorithms and their comparisons are given in [6]. The systolic array approaches, which are assumed to be fully pipelined, also employ the row-column method in the 2-D DCT realization [5], [7]. Therefore, the fully pipelined operation is impaired since the second DCT cannot start until the transposition of the first transform coefficients finishes.

The direct 2-D method, in contrast to the row-column decomposition method, is a complete 2-D approach. Duhamel and Guillemot [15] proposed two direct 2-D methods, the indirect and direct polynomial transforms (PT) for the 2-D DCT. Vetterli [14] used the indirect PT approach for 2-D DCT to map the $N \times N$ DCT into a real $N \times N$ DFT followed by a number of rotations, and the real $N \times N$ DFT can be realized by using PT. The direct PT for the 2-D DCT is shown to be more effective than the indirect approach [9], but the procedure for a general implementation is compli-

Manuscript received September 3, 1991; revised January 10, 1992. This work is partially supported by NSF grant ECD-8803012. This paper was recommended by Associate Editor Takao Nishitani.

The authors are with the Department of Electrical Engineering, University of Maryland, College Park, MD 20742.

IEEE 9106992.

1051-8215/92$03.00 © 1992 IEEE

cated. It has also been shown [14] that these kinds of direct 2-D methods are more efficient than the row-column methods. However, due to their simplicity in hardware realization, row-column decomposition methods are still widely adopted in implementing the 2-D DCT chips [23], [36].

All the approaches mentioned above do have a common requirement: the availability of all the data in the processing 2-D block. This may not be true for the real-time data transmission systems such as HDTV systems. To eliminate the waiting time for the data to arrive, a time-recursive processing concept can be exploited, i.e., the result is adaptively updated when a new datum arrives. Once all the data arrive, the result is completely available. One of the most important issues here is to design a real-time VLSI system that is compatible with the data transmission speed. In this paper, we show that frame-recursive architecture is a feasible solution. Liu and Chiu [6] proposed new unified parallel lattice structures for time-recursive 1-D orthogonal sinusoidal transforms. These transforms are decoupled into N independent lattice modules, hence, there are no global communications in the structure. Moreover, every lattice module is regular and modular and has the same architecture. The number of multipliers of the lattice structure for the best case in [6] is $4N$. Therefore, it is very suitable for VLSI implementation.

In this paper, we propose a new architecture for the 2-D DCT by employing the frame-recursive concept on the successive input frames. It is a direct 2-D method and the transposition in the row-column method is eliminated. Because the system requires only two 1-D lattice DCT module arrays, the hardware complexity of this system equals that of the row-column method for series rows (or columns) input frame. All the components needed in the architecture are independent lattice modules and shift registers. The total number of multipliers required for an $N \times N$ 2-D DCT is $8N$. There is no limitations on the transform size N and the structure can be easily extended to any number N, including the prime numbers. This is a rather promising architectures either from speed or hardware point of view. Since the system is modular, regular, and fully pipelined, it is very suitable for high-speed video signal transmission. We show that by employing distributed arithmetic technique in hardware implementation, the system performance is further improved. For the HDTV applications, very high operating speed is achieved by using parallel processings and appropriate scanning arrangements.

We organize the rest of the paper as follows. In Section II, the algorithm to achieve the 2-D DCT using the frame-recursive manner is proposed. It is shown that we can dually generate a 2-D discrete sine-cosine transform (DSCT) simultaneously. The architectures for calculating moving frame 2-D DCT and block 2-D DCT are discussed in Section III. Comparisons of different 2-D DCT algorithms are given in Section IV. In Section V, the VLSI implementation of the block 2-D DCT by employing the distributed arithmetic is discussed. In Section VI we consider the application of our 2-D DCT block architecture to the high-speed HDTV systems. Finally, the conclusion is given in Section VII.

II. Dual Generation of 2-D DCT and DSCT

In this section, we describe a new architecture for 2-D DCT that requires only two 1-D DCT arrays. Focusing directly on the 2-D transformed signal and applying the frame recursive approach, we can derive not only the frame recursive relation of two successive frames of the 2-D DCT but also the dual generation properties between the 2-D DCT and 2-D discrete sine-cosine transform (DSCT). Here the DSCT serves as an auxiliary transform that supports the time-recursive computations of the 2-D DCT.

A. Frame-Recursive 2-D Discrete Cosine Transform

The $N \times N$ 2-D DCT $\{X_c(k, l, t)\colon k, l = 0, 1, \ldots, N - 1.\}$ of an $N \times N$ 2-D data sequence $\{x(m, n)\colon m = t, t + 1, \ldots, t + N - 1;\ n = 0, 1, \ldots, N - 1.\}$ is defined as

$$X_c(k,l,t) = \frac{4}{N^2} C(k)C(l) \sum_{m=t}^{t+n-1} \sum_{n=0}^{N-1} x(m,n) \cdot \cos\left[\frac{\pi[2(m-t)+1]k}{2N}\right] \cos\left[\frac{\pi(2n+1)l}{2N}\right] \quad (1)$$

where

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

Here the time index t in $X_c(k, l, t)$ denotes that the transform starting from the tth row of the 2-D data sequence $\{x(m, n)\colon m = 0, 1, 2, \ldots;\ n = 0, 1, \ldots, N - 1.\}$ as shown in Fig. 1. In the following, we call $X_c(k, l, t)$ the 2-D DCT of the tth frame of the 2-D data sequence $x(m, n)$. To derive the time-recursive relations between the successive data frames, let us start by considering the 2-D DCT of the tth frame data sequence:

$$X_c(k, l, t) = \frac{4}{N^2} C(k)C(l) \sum_{m=t}^{t+N-1} \sum_{n=0}^{N-1} x(m, n) \cdot \cos\left[\frac{\pi(2(m-t)+1)k}{2N}\right] \cos\left[\frac{\pi(2n+1)l}{2N}\right]. \quad (2)$$

Instead of focusing on $X_c(k, l, t)$ and utilizing various techniques to reduce the computational complexity, we will consider the 2-D DCT sequence of the $(t + 1)$th frame, which is

$$X_c(k, l, t+1) = \frac{4}{N^2} C(k)C(l) \sum_{m=t+1}^{t+N} \sum_{n=0}^{N-1} x(m, n) \cdot \cos\left[\frac{\pi[2(m-t-1)+1]k}{2N}\right] \cos\left[\frac{\pi(2n+1)l}{2N}\right]. \quad (3)$$

By using trigonometric function expansions on $\cos\{\pi[2(m$

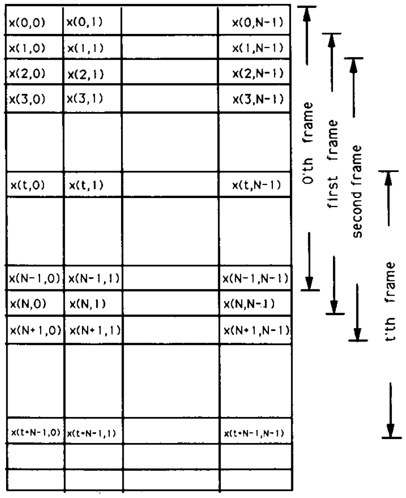

Fig. 1. The 2-D successive data frame.

$- t - 1) + 1]k/2N\}$, (3) can be rewritten as

$$\begin{aligned}X_c(k,l,t+1) &= \frac{4}{N^2}C(k)C(l)\sum_{m=t+1}^{t+N}\sum_{n=0}^{N-1}x(m,n)\\ &\quad\cdot\cos\left[\frac{\pi[2(m-t)+1]k}{2N}\right]\cos\left(\frac{\pi k}{N}\right)\\ &\quad\cdot\cos\left[\frac{\pi(2n+1)l}{2N}\right]\\ &\quad+\frac{4}{N^2}C(k)C(l)\sum_{m=t+1}^{t+N}\sum_{n=0}^{N-1}x(m,n)\\ &\quad\cdot\sin\left[\frac{\pi[2(m-t)+1]k}{2N}\right]\sin\left(\frac{\pi k}{N}\right)\\ &\quad\cdot\cos\left[\frac{\pi(2n+1)l}{2N}\right]\\ &= \overline{X}_c\cos\left(\frac{\pi k}{N}\right)+\overline{X}_{sc}\sin\left(\frac{\pi k}{N}\right)\end{aligned}\tag{4}$$

where

$$\overline{X}_c = \frac{4}{N^2}C(k)C(l)\sum_{m=t+1}^{t+N}\sum_{n=0}^{N-1}x(m,n)\cdot\cos\left[\frac{\pi[2(m-t)+1]k}{2N}\right]\cos\left[\frac{\pi(2n+1)l}{2N}\right]\tag{5}$$

and

$$\overline{X}_{sc} = \frac{4}{N^2}C(k)C(l)\sum_{m=t+1}^{t+N}\sum_{n=0}^{N-1}x(m,n)\cdot\sin\left[\frac{\pi[2(m-t)+1]k}{2N}\right]\cos\left[\frac{\pi(2n+1)l}{2N}\right].\tag{6}$$

We can view the term $\sin\{\pi[2(m - t) + 1]k/2N\}$ $\cos[\pi(2n + 1)l/2N]$ in (6) as a new transform kernel, and define an $N \times N$ 2-D discrete sine-cosine transform (DSCT) sequence $\{X_{sc}(k, l, t)\colon k = 1, 2, \ldots, N;\ l = 0, 1, \ldots, N - 1\}$ of the 2-D data sequence $\{x(m, n)\colon m = t, t + 1, \ldots, N + t - 1;\ n = 0, 1, \ldots, N - 1\}$ as

$$X_{sc}(k,l,t) = \frac{4}{N^2}C(k)C(l)\sum_{m=t}^{N+t-1}\sum_{n=0}^{N-1}x(m,n)\cdot\sin\left[\frac{\pi[2(m-t)+1]k}{2N}\right]\cos\left[\frac{\pi(2n+1)l}{2N}\right].\tag{7}$$

Here we extend the definition of $C(k)$ to $C(N)$ and define $C(N) = 1/\sqrt{2}$. Similarly, we are interested in the 2-D DSCT of the $(t + 1)$'s frame. According to the definition, it is

$$\begin{aligned}&X_{sc}(k,l,t+1)\\ &= \frac{4}{N^2}C(k)C(l)\sum_{m=t+1}^{t+N}\sum_{n=0}^{N-1}x(m,n)\\ &\quad\cdot\sin\left[\frac{\pi[2(m-t-1)+1]k}{2N}\right]\cos\left[\frac{\pi(2n+1)l}{2N}\right]\\ &= \frac{4}{N^2}C(k)C(l)\sum_{m=t+1}^{t+N}\sum_{n=0}^{N-1}x(m,n)\\ &\quad\cdot\sin\left[\frac{\pi(2(m-t)+1)k}{2N}\right]\cos\left[\frac{\pi(2n+1)l}{2N}\right]\cos\left(\frac{\pi k}{N}\right)\\ &\quad-\frac{4}{N^2}C(k)C(l)\sum_{m=t+1}^{t+N}\sum_{n=0}^{N-1}x(m,n)\\ &\quad\cdot\cos\left[\frac{\pi(2(m-t)+1)k}{2N}\right]\cos\left[\frac{\pi(2n+1)l}{2N}\right]\sin\left(\frac{\pi k}{N}\right)\\ &= \overline{X}_{sc}\cos\left(\frac{\pi k}{N}\right)-\overline{X}_c\sin\left(\frac{\pi k}{N}\right)\end{aligned}\tag{8}$$

where the terms $\overline{X}_c$ and $\overline{X}_{sc}$ used in (4) to generate $X_c(k, l, t + 1)$ appear again. This suggests that the 2-D DCT and 2-D DSCT can be dually generated from each other.

B. Lattice Structure Representations for Frame-Recursive 2-D DCT

We show in this section that 2-D DCT can be generated by using two lattice arrays. From (4) and (8), it is noted that the new 2-D DCT and DSCT transforms can be obtained from the intermediate values $\overline{X}_c$ and $\overline{X}_{sc}$ in a lattice form as shown in Fig. 2. A similar relation also exists in the dual generations of the 1-D DCT and 1-D DST [6]. The intermediate data $\overline{X}_c$ and $\overline{X}_{sc}$ differ from the original signal $X_c(k, l, t)$ and $X_{sc}(k, l, t)$ only in the tth row and the $(t + N)$th row of the input data frames. So the intermediate data $\overline{X}_c$ and $\overline{X}_{sc}$ can be obtained from $X_c(k, l, t)$ and $X_{sc}(k, l, t)$ by removing the tth row of the transformed data and updating the $(t + N)$th row of the transformed data.

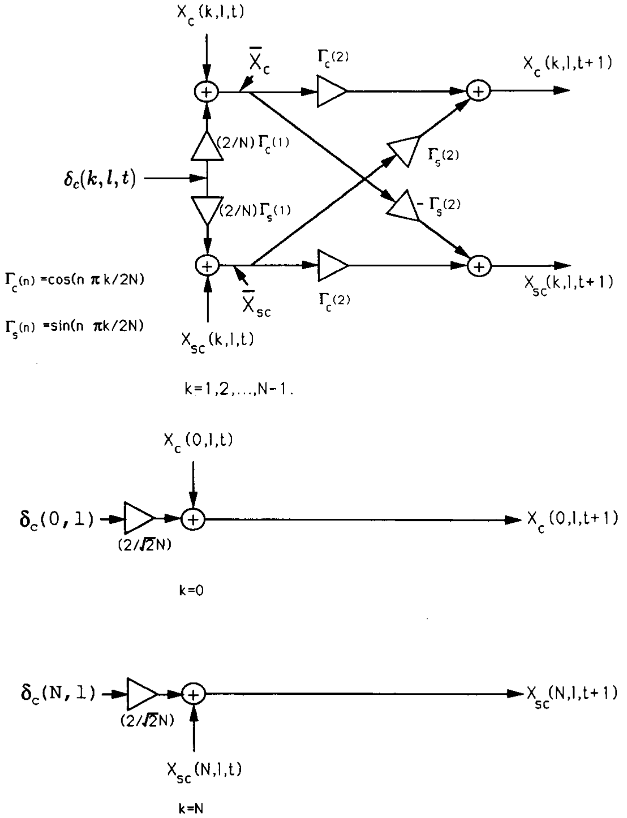

Fig. 2. The lattice module of lattice array II.

That is,

$$\overline{X}_c = X_c(k, l, t) - \frac{4}{N^2}C(k)C(l)\sum_{n=0}^{N-1} x(t, n) \cdot \cos\left[\frac{\pi(2n+1)l}{2N}\right]\cos\left(\frac{\pi k}{2N}\right) + \frac{4}{N^2}C(k)C(l)\sum_{n=0}^{N-1} x(t+N, n) \cdot \cos\left[\frac{\pi(2n+1)l}{2N}\right]\cos\left[\frac{\pi(2N+1)k}{2N}\right] \tag{9}$$

and

$$\overline{X}_{sc} = X_{sc}(k, l, t) - \frac{4}{N^2}C(k)C(l)\sum_{n=0}^{N-1} x(t, n) \cdot \cos\left[\frac{\pi(2n+1)l}{2N}\right]\sin\left(\frac{\pi k}{2N}\right) + \frac{4}{N^2}C(k)C(l)\sum_{n=0}^{N-1} x(t+N, n) \cdot \cos\left[\frac{\pi(2n+1)l}{2N}\right]\sin\left[\frac{\pi(2N+1)k}{2N}\right]. \tag{10}$$

These two equations can be further simplified as

$$\overline{X}_c = X_c(k, l, t) + \delta_c(k, l, t)\frac{2}{N}C(k)\cos\left(\frac{\pi k}{2N}\right) \tag{11}$$

and

$$\overline{X}_{sc} = X_{sc}(k, l, t) + \delta_c(k, l, t)\frac{2}{N}C(k)\sin\left(\frac{\pi k}{2N}\right) \tag{12}$$

where

$$\delta_c(k, l, t) = \frac{2}{N}C(l)\sum_{n=0}^{N-1}\left[(-1)^k \cdot x(t+N, n) - x(t, n)\right]\cos\left[\frac{\pi(2n+1)l}{2N}\right]. \tag{13}$$

By substituting $\overline{X}_c$ and $\overline{X}_{sc}$ in (11) and (12) into the updated transformed signal $X_c(k, l, t+1)$ and $X_{sc}(k, l, t+1)$ in (4) and (8), the relation between the updated transform signal and previous transformed signal for $k = 1, 2, \ldots, N-1$, are represented in a lattice form as shown in the upper part of Fig. 2. Since the multiplications can be reduced to addition and subtraction for $k = 0$ in the 2-D DCT and $k = N$ in the DSCT, respectively, these two cases can be simplified as shown in the lower part of Fig. 2. What is worth noticing is that $\delta_c(k, l, t)$ in (13) is the 1-D DCT of the data vector, which is the difference between the parity of the $(t+N)$th row and tth row of the 2-D input sequence. As described in [6], $\delta_c(k, l, t)$ can be generated in a lattice form as shown in Fig. 3. We call this as lattice array I (LAI) and that in Fig. 2 as lattice array II (LAII). Comparing these two structures, we observe that these two lattice modules have the same architecture except that LAI feedbacks the outputs directly through a delay stage to add with the inputs.

The 2-D DCT and DSCT are produced by applying input data frames to LAI's, which generate the $\delta_c(k, l, t)$. After obtaining the $\delta_c(k, l, t)$, the updated transformed signal can be obtained recursively by feeding $\delta_c(k, l, t)$ into the lattice array II. We observe that the 2-D DCT can be obtained by using two 1-D DCT lattice arrays. It will be shown in the next section that the 2-D DCT obtained by this time-recursive approach is fully pipelined and no transposition is required. This is because that by using the frame-recursive approach, we start from the transformed 2-D DCT directly and avoid calculating the 2-D DCT indirectly from the 1-D DCT. Our architectures are efficient since it is a direct 2-D approach. This method can also obtain the 2-D DCT and 2-D DSCT simultaneously. In contrast to processing the input 2-D data sequence by rows, the input data can be updated by columns. In this case, 2-D DCT and 2-D discrete cosine-sine transform (DCST) are dually generated, and all other results are similar and can be easily derived.

III. Architectures of Frame-Recursive Lattice 2D-DCT and 2-D DSCT

The fully pipelined parallel lattice structures for successive frame and block 2-D DCT and DSCT are described in this section. As we know, most of the 2-D DCT architectures are

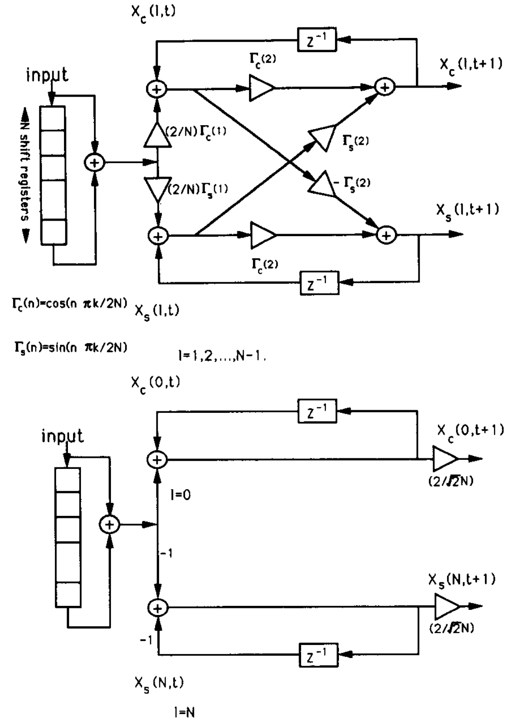

Fig. 3. The lattice module of lattice array I.

implemented by the row-column decomposition method [18]-[20]. This is due to the fact that the architectures of most fast direct 2-D algorithms are usually irregular and globally connected, therefore it is not practical for VLSI implementation. Another reason is that it is beneficial to generate a 2-D DCT system from existing 1-D DCT circuit rather than to build a new multiplier-saved 2-D DCT architecture that may not be compatible with the 1-D DCT system. By using the frame-recursive method, the 2-D DCT can be implemented by 1-D DCT lattice arrays that are regular, modular, and suitable for VLSI implementation. Thus the difficulties mentioned above are avoided. We will discuss two architectures, the moving frame 2-D DCT architecture and the block 2-D DCT architecture. The moving frame 2-D DCT architecture is used to calculate the 2-D DCT of sequential frames. For example, the 2-D DCT's of the 0th frame, first frame, second frame, and so on. The block 2-D DCT architecture computes the 2-D DCT of an $N \times N$ input data matrix for every N frames, i.e., the 0th frame, the Nth frame, the $2N$th frame and so on.

A. The Moving Frame 2-D DCT Architecture

The moving frame 2-D DCT architectures generate the 2-D DCT of successive input frames. From the frame-recursive concept derived in Section II, the 2-D DCT recursive lattice structures can be constructed according to (4), (8), and (13). Although the intermediate values $\delta_c(k, l, t)$ in (13) are functions of both k and l, it is noted that the effect due to the index k is equivalent to sign changes in the input data sequences. Using this property, we will show two approaches, the prematrix method and the postmatrix method, to implement the moving frame 2-D DCT architectures. The prematrix method will be discussed first.

The Prematrix Method: In this method, the intermediate values of $\delta_c(k, l, t)$ are realized directly from (13). As we can see, the index k in (13) affects only the sign of the new input data sequence. Thus, there are only two possible input sequence combinations: $\{x(t + N, n) - x(t, n)\}$ and $\{-x(t + N, n) - x(t, n)\}$. The resulting prematrix moving-frame 2-D DCT architecture is shown in Fig. 4, which includes a circular shift matrix I, two adders, two LAI's, one LAII, and two circular shift arrays and shift register arrays. Except for the LAI, LAII, and adders, all other components are shift registers. We will describe the functions of every blocks first, then demonstrate how the system works.

The circular matrix I (CSMI is an $(N + 1) \times N$ shift registers connected as shown in the upper part of Fig. 5. When a new input datum $x(m, n)$ arrives every clock cycle, all the data are shifted in the way as indicated in Fig. 5. Both of the first elements in the tth row and $(t + N)$th row are sent to the adders for summation and subtraction as shown in Fig. 4. The prematrix architecture contains two LAI's, which includes N lattice modules as shown in Fig. 3. The upper and lower LAI's execute the 1-D DCT on the rows of the 2-D input data for the even and odd transformed components k, respectively. Because the length of the input vector is N and only the discrete cosine transformed data are needed, the 1-D DCT transformed data $\delta_c(k, l, t)$ generated by the LAI's are sent out every N clock cycles [6]. Due to the time-recursive approach used, the initial values $X_c(l, t)$ and $X_s(l, t)$ in the LAI's (see Fig. 3) are reset to zeros every N clock cycles.

The circular shift array in the middle of the system is an $N \times 1$ shift register array as shown in Fig. 6. This special shift register array loads an $N \times 1$ data vector from the LAI every N clock cycles, then it will shift the data circularly and send the data to the LAII every clock cycle. There are three inputs in LAII, $\delta_c(k, l, t)$, $X_c(k, l, t)$, and $X_s(k, l, t)$, where the $\delta_c(k, l, t)$ comes from the circular shift array and $X_c(k, l, t)$ and $X_s(k, l, t)$ from the shift register arrays located behind the LAII. We divide the LAII into two groups: the $\text{LAII}_{\text{even}}$ and LAII_{odd}. Each includes $N/2$ lattice modules as shown in Fig. 2. $\text{LAII}_{\text{even}}$ contains only those lattice modules for even transformed components k, whereas LAII_{odd} contains only the odd lattice modules. It should be noticed that this system contains two LAI and only one LAII. The shift register array contains $2N \times N$ registers. Their operations are shown in Fig. 7.

The following illustrates how this parallel lattice structure works to obtain the 2-D DCT and DSCT of 2-D input successive frames. All the initial values of the circular shift matrix I (CSMI), circular shift array, and shift register array are set to zeros. The input data sequence $x(m, n)$ sequentially shifts row by row into the $(N + 1) \times N$ CSMI. First we calculate the difference between the tth row and the $(t + N)$th row data vector of the CSMI. The two resulting

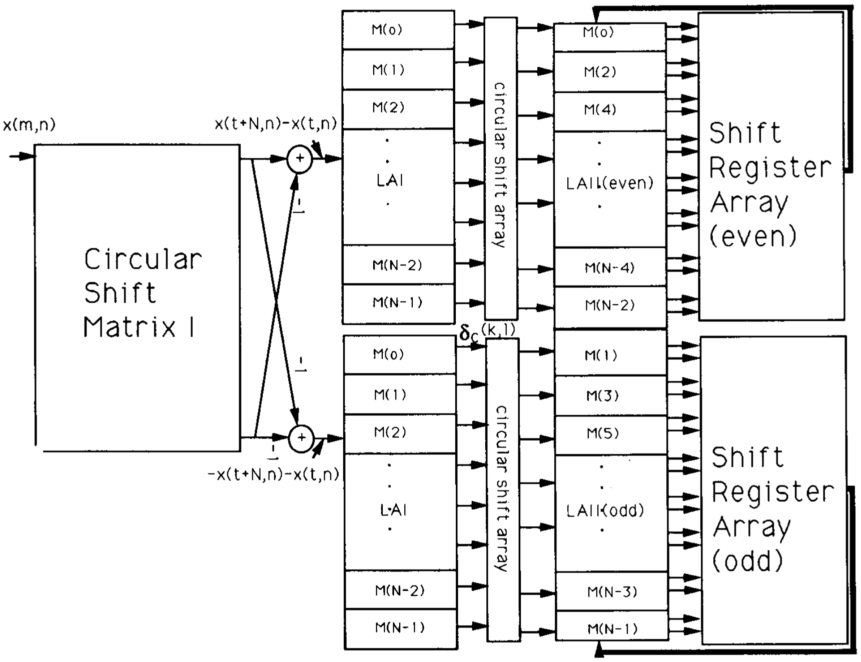

Fig. 4. The pre-matrix moving frame 2-D DCT architecture.

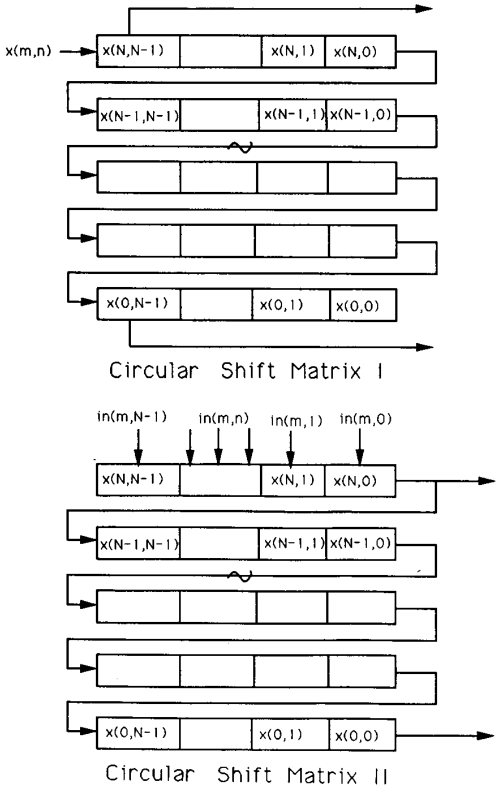

Fig. 5. he circular shift matrix (CSM) I and II.

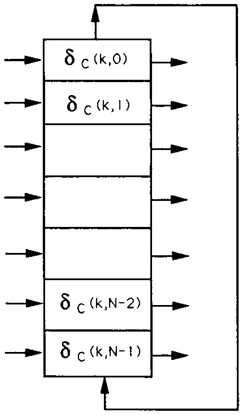

Fig. 6. The circular shift array (CSA).

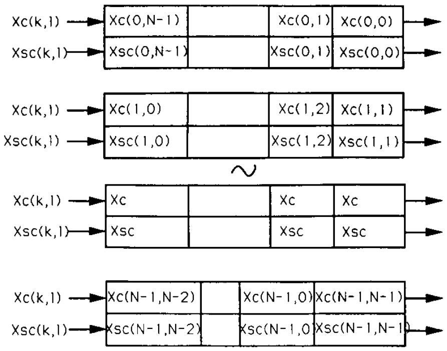

Fig. 7. The shift register array (SRA).

combinations of the input sequences, $x(t+N, n) - x(t, n)$ and $-x(t+N, n) - x(t, n)$ for $n = 0, 1, 2, \ldots, N-1$, are used as the input sequences for the lattice array I's, which consist of $2N$ lattice modules to calculate the $1 - D$ DCT for $\{x(t+N, n) - x(t, n)\}$ and $\{-x(t+N, n) - x(t, n)\}$. The upper LAI is for the even transformed components k and the lower one for odd k. Suppose the data of the input vectors arrive serially per clock cycle, it takes N clock cycles to obtain the $\delta_c(k, l, t)$ for both of the input sequences. At the Nth cycle, the N transformed data $\delta_c(k, l, t)$ are loaded into the circular shift arrays (CSA), which will shift circularly and send the data out of the register into the LAII for different k components every clock cycle. In LAII, $X_c(k, l, t+1)$ and $X_{sc}(k, l, t+1)$ are evaluated according to (4) and (8). Because $\text{LAII}_{\text{even}}$ and LAII_{odd} have only $N/2$ modules, every $\delta_c(k, l, t)$ is floating for $N/2$ clock cycles. It is noted that a specific 2-D transform data $X_c(k, l, t+1)$ and $X_{sc}(k, l, t+1)$ are updated recursively every N clock cycles from $X_c(k, l, t)$ and $X_{sc}(k, l, t)$. Therefore the outputs of the LAII are sent into the shift register array (SRA) where data are delayed by N clock cycles. Each SRA contains $N/2$ shift registers each with length N. The data in the rightest registers are sent back as the $X_c(k, l, t)$ and $X_{sc}(k, l, t)$ of LAII. At the $N^2 + 2N$ clock cycle, the 2-D DCT and DSCT of the 0th frame are available. After this, the 2-D transformed data of successive frames can be obtained every N clock cycles.

We observe two interesting results in the prematrix method. First, both LAI and LAII can be viewed as filter banks. This is because every lattice module itself is an independent digital filter with different frequency components $k, l = 0, 1, \ldots, N-1$. Moreover, all the lattice modules in this architecture have the same structure. Second, the system requires 3 1-D DCT array and is fully pipelined with throughput rate N clock cycles. From the above discussion, transposition for the row-column decomposition method is unnecessary in this realization. According to the 1-D DCT architecture proposed in [6] (Liu-Chiu2 architecture), the total multiplier required in the 2-D DCT is $12N$ and the total number of adders is $15N$. Due to the goal of piping out the results every N clock cycles, it requires three 1-D DCT structures in the system. We will show how to use only two 1-D DCT lattice arrays to attain the results at the same throughput rate in the postmatrix method.

The Postmatrix Method: The intermediate value $\delta_c(k, l, t)$ in (13) can be rewritten as

$$\begin{aligned}\delta_c(k, l, t) &= (-1)^k \frac{2}{N} C(l) \sum_{n=0}^{N-1} x(t+N, n) \cos\left[\frac{\pi(2n+1)l}{2N}\right] \\ &\quad - \frac{2}{N} C(l) \sum_{n=0}^{N-1} x(t, n) \cos\left[\frac{\pi(2n+1)l}{2N}\right] \\ &= (-1)^k X_c'(t+N, l) - X_c'(t, l). \qquad (14)\end{aligned}$$

That is, we can calculate the 1-D DCT of the tth row and the $(t+N)$th row of the input frame individually, then perform the summation later on. Our approach is to send the input sequence $x(m, n)$ row by row directly into the LAI. It takes N clock cycles for the LAI to complete the 1-D DCT of one row input vector, then the array sends the 1-D DCT data in parallel to the CSMII as shown in Fig. 8. The operations of CSMII are shown in the lower part of Fig. 5. At the output of the CSMII, the 1-D transformed data of the $(t+N)$th row and tth row are added together according to (14) depending on the sign of the k components (see Fig. 8). Then the results are sent to CSA's, LAII, and SRA's, whose operations remain intact as in the prematrix method. The whole structure is demonstrated in Fig. 8. Therefore, by transforming the input data first, we can implement the 2-D DCT by using only two 1-D DCT lattice arrays and retain the same pipeline rate. The total numbers of multipliers and adders needed for the postmatrix method are $8N$ and $10N$, respectively.

B. The Block 2-D DCT Architecture

In most image processing applications, the 2-D DCT are executed block by block instead of in successive frames [27], [29]. We will show how to apply the frame-recursive concept to obtain the block 2-D DCT. It will be easier to understand if we use an example to show how to obtain the 0th frame 2-D DCT in the postmatrix moving frame 2-D DCT architecture. Recall that the CSMII in Fig. 8 is used to store the transformed data $X_c'(t, l)$ of the previous input row vectors. Since the 0th frame is the first input data frame, all the values in the CSMII are set to zeros. Corresponding to (14), this means that the second terms $X_c'(t, l)$ are zeros. When the $(N-1)$th row data vector (the last row of the 0th frame) arrive, the 2-D DCT of the 0th input data frame is obtained. During this initial stage, the 2-D DCT of the 0th frame obtained by the moving frame approach is equal to the block 2-D DCT of the 0th frame. Therefore, if we want to compute the 2-D DCT of the Nth frame, then all the values in the CSMII are resetting to zeros when the first datum in the Nth data frame (i.e., $x(N, 0)$) arrives. That is, we can obtain the block 2-D DCT by reset the values of the CSMII every N^2 cycles. This means that the CSMII in Fig. 8 is redundant and the second terms $X_c'(t, l)$ in (14) are zeros. Thus, the intermediate value of $\delta_c(k, l, t)$ can be rewritten as

$$\delta_c(k, l, t) = (-1)^k \frac{2}{N} C(l) \sum_{n=0}^{N-1} x(t+N, n) \cdot \cos\left[\frac{\pi(2n+1)l}{2N}\right]. \qquad (15)$$

The block 2-D DCT architecture is shown in Fig. 9. Corresponding to our frame-recursive algorithm, we obtain another block of input data every N^2 clock cycles. Note that this is also the total time required for all the N^2 data to arrive in a transmission system.

The following is an example to calculate block 2-D DCT for the 0th frame. When row data vectors arrive, LAI performs the 1-D DCT on them. Every N clock cycles, after the last datum of each row $x(m, N-1)$ is available, LAI

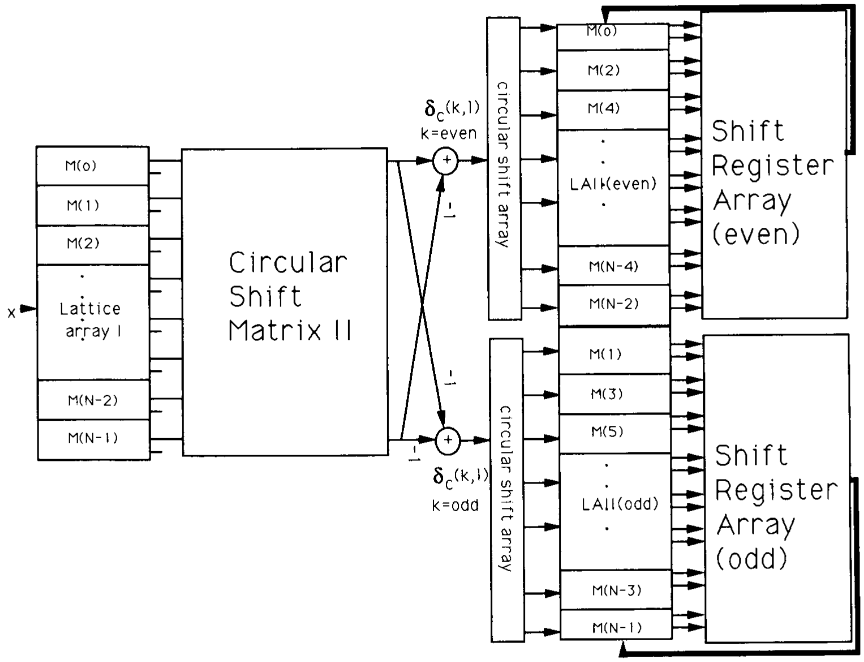

Fig. 8. The postmatrix moving frame 2-D DCT architecture.

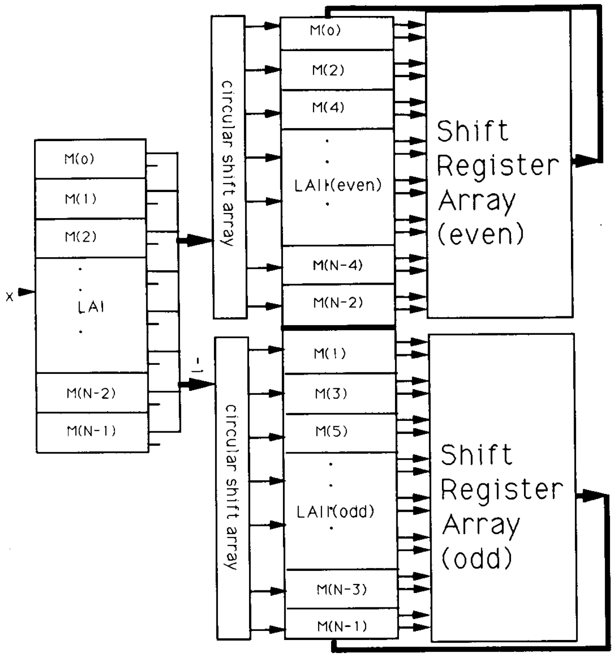

Fig. 9. The block 2-D DCT architecture.

completes the 1-D DCT for every row and sends the N 1-D DCT transformed data to the two length-N CSA's. The upper CSA translates the intermediate value $\delta_c(k, l, t)$ to the lattice array $\mathrm{II}_{\mathrm{even}}$, as do the lower CSA except that the signs of the output of the CSA are changed before being sent to the lattice array $\mathrm{II}_{\mathrm{odd}}$. The operations of the lattice array II and SRA are the same as those in the previous methods. As we can see, the hardware complexity of our block 2-D DCT architecture is as simple as the row–column methods. Moreover, our system can be operated in a fully pipelined manner.

IV. Comparisons

Since most of the 2-D DCT algorithms proposed are based on manipulating $N \times N$ block signals, we compare only the 2-D block architecture as described in Section III-B with other algorithms. The block 2-D DCT architecture is a fully-pipelined serial input parallel output (SIPO) system with throughput rate every N clock cycles and in terms of hardware complexity, it requires only two 1-D DCT architectures without transpositions. It is attractive, therefore, not only for its efficiency in term of system throughput but also for its hardware simplicity and regularity.

In the following section, the comparisons between our 2-D DCT block structure and those of others are based on the number of multipliers, adders, and speed. For the sake of clarity, we divide the algorithms into two groups: parallel input parallel output (PIPO) and serial input parallel output (SIPO). The fast algorithms presented by Vetterli and Nussbaumer [13], [14], Duhamel and Guillemot [15], and Cho and Lee [10] belong to the former class. Vetterli's algorithm [14] mapped the 2-D DCT into a 2-D cosine DFT and sine DFT through a number of rotations, and the 2-D DFT are computed by polynomial transform (PT) methods [14]. Vetterli's method can reduce the number of multipliers more than 50% in comparison to the row–column method based on Chen *et al.*'s algorithms [2] and has a comparable amount of additions. Duhamel and Guillemot [15] present a PT-based algorithm that uses the direct DCT approach. This direct PT

TABLE I
Comparisons of Different 2-D DCT Algorithms

	Row-Column Method Based on Chen in [2]	Duhamel [15] *et al.*	Cho-Lee [10]	Ma [7]	Liu-Chiu2D
No. of multipliers	$2N^2 \ln(N) - 6N^2/2 + 8N$	$N^2 + 2N + 2$	$N^2 + 2N + 2$	$4N(N+1)$	$8N$
Throughput	$N +$ transposition	$2N$	N	$2N + 1$	N
Limitation on transform size N	Power of 2	Power of 2	Power of 2	No	No
Communication	Global	Global	Global	Local	Local
I/O operation	PIPO	PIPO	PIPO	SIPO	SIPO
Approach of algorithm	Indirect	Direct	Direct	Indirect	Direct

TABLE II
Comparisons of the Number of Multipliers

No.	Row-Column Method Based on Chen in [2]	Duhamel [15] *et al.*	Cho-Lee [10]	Ma [7]	Liu-Chiu2D
8	256	96	96	288	64
16	1 408	512	512	1 088	128
32	7 424	2 560	2 560	4 224	256
64	37 376	12 288	12 288	16 640	512

TABLE III
Comparisons of the Number of Adders

No.	Row-Column Method Based on Chen in [2]	Duhamel [15] *et al.*	Cho-Lee [10]	Ma [7]	Liu-Chiu2D
8	416	484	466	432	78
16	2 368	2 531	2 530	1 632	158
32	12 416	12 578	12 738	6 336	318
64	61 696	60 578	42 461	24 960	638

2-D DCT method provides a 10% improvement in both the numbers of additions and multiplications compared to Vetterli's result [14], but it requires complex computations. Cho and Lee's algorithm is a direct 2-D method that employs the properties of trigonometric functions. The number of multipliers are the same as that of Duhamel and Guillemot's, but the structure is more regular, and only real arithmetic is involved. Up to now, the best results for the first PIPO systems in terms of the number of multipliers are $(N^2 + 2N + 2)$, which were obtained by Duhamel and Guillemot, as well as by Cho and Lee. But assuming that all the N^2 input data arrive at the same time is not practical in communication systems. The data waiting time is N^2, which is always neglected in these approaches.

The systolic array approaches proposed by Lee and Yasuda [5], Ma [7], and Liu and Chiu belong to the SIPO method. Lee and Yasuda presented a 2-D systolic DCT/DST array algorithm based on an IDFT version of the Goertzel algorithm via Horner's rule in [5]. The latest systolic array algorithm for 2-D DCT was proposed by Ma [7], where he showed two systolic architectures of 1-D DCT arrays based on the indirect approach proposed by Vetterli and Nussbaumer [13], then he exploited the 2-D DCT systolic array by using the features of the two 1-D DCT systolic arrays. This method requires $(N + 1)$ 1-D DCT structures and the total number of time steps is $(N^2 + 2N + 2)$ [7]. We call the block 2-D DCT structure shown in Fig. 9, based on the Liu-Chiu2 module [6], Liu-Chiu2. This needs only two 1-D DCT, and the total time steps are N^2. The comparisons regarding their inherent characteristics are given in Table I. In addition, the quantities comparisons in terms of the number of multipliers and adders are given in Tables II and III. In general, the SIPO method is more workable in hardware implementations. Our structure requires fewer multipliers than Ma's structure and is highly regular, systematic, and uses only local communications. In addition, this lattice 2-D DCT architecture can be generated from the 1-D DCT lattice array without modifications.

V. VLSI Implementation Using High-Speed Distributed Arithmetic

In this section, we discuss the VLSI architecture of our block 2-D DCT structure shown in Fig. 9. Since this structure contains only shift registers and $2N$ lattice modules, which are exactly the same except for the multipliers' coefficients, we can foresee that the VLSI implementation of this system is labor saving. Every lattice module is a modified normal form digital filter [21], which has the least roundoff noise and is more insensitive to the coefficient inaccuracy. Due to the fact that the block operation will reset all the outputs of the LAI and LAII every N and N^2 clock cycles, respectively, the round-off errors will be further minimized. In the following we focus our discussion on the 8 × 8 block DCT with 12 b 2's complement implementation. Suppose the lattice module is based on the Liu-Chiu2 module with four multipliers [6], then the total number of multipliers needed for the 2-D DCT is 64, which requires an enormous area under 2 or 1.2 μm CMOS technology. Moreover, the system throughput is also limited by the operational speed of multipliers.

Sun *et al.* [19] proposed the first working 16 × 16 DCT chip that incorporates distributed arithmetics methods. Based on this memory-oriented structure, high-speed, high-accuracy, and efficient hardware implementation of the 2-D DCT can be achieved. Here we adopt the distributed arithmetic in our implementations. By employing this scheme, the system will have higher accuracy because given the same word length, the result will undergo less round-off operations than direct implementation using multipliers. The lattice module in Figs. 2 and 3 can be redrawn in Fig. 10. The dashed box in Fig. 10 can be implemented by using a single ROM with three inputs and four outputs. Under this realization, the round-off errors due to the multiplication are minimized since

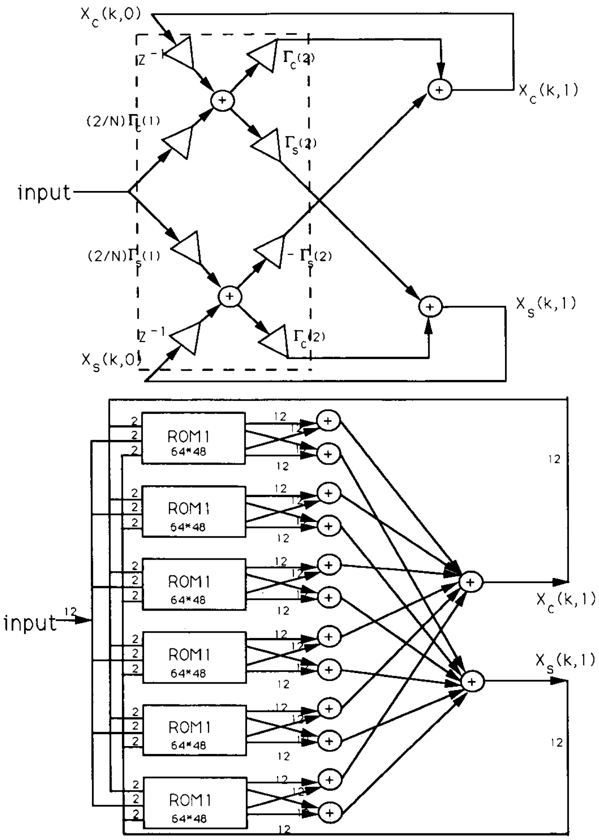

Fig. 10. The realization of the lattice module under distributed arithmetic using one ROM.

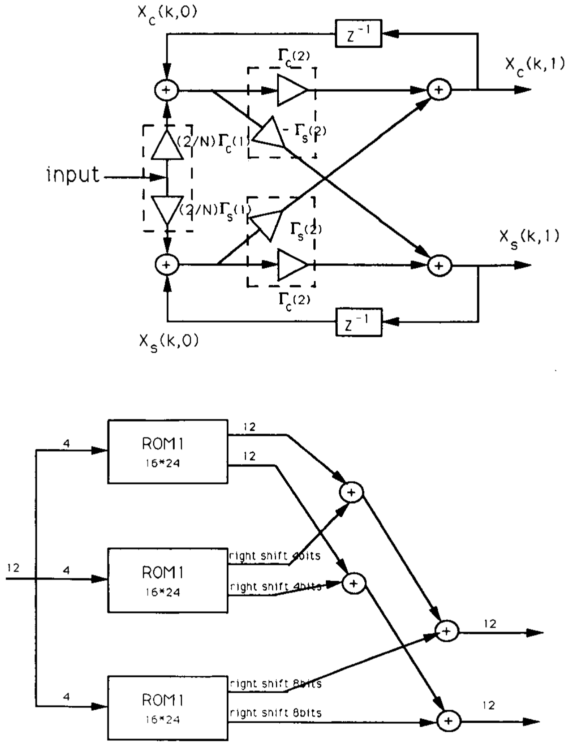

Fig. 11. The realization of the lattice module under distributed arithmetic using three ROM's.

the distributed arithmetic convert the explicit multiplication into the implicit multiplication. Therefore, the errors of the systems are all due to the quantization errors under finite precision implementations and adder operations. Under the 12 b 2's complement realization, the rms error values are approximately 40 dB [19], which is satisfactory for most applications. Assuming that every input of the ROM is 2-b long, then the lattice module can be implemented by six ROM's and 22 adders as shown in Fig. 10. The ROM size for each lattice array is 18 432 b. By reducing the number of bits of every input of the ROM to one, the ROM size becomes 4608 b, which is one-fourth of the previous case, but the number of adders is doubled.

The other way to implement the lattice module by ROM is shown in Fig. 11. Every dashed box is realized by a ROM with one input and two outputs. Fig. 11 illustrates the realization of each ROM when the number of bits of the input signal is 4 b. Using this method, the ROM size of each lattice array is 3456 b and the number of adders needed is 16. When the number of bits of the input signal is reduced, the ROM size is reduced, but the number of adders is increased. We will implement the system based on the schematic diagram for each lattice module as shown in Fig. 11. The adder is a 12-b carry-look-ahead full adder/subtracter that is combined from three 4-b carry-look-ahead adders. Since the area of ROM is much less than that of the multipliers and the speed is higher, circuit implementations under this approach can be fabricated for very-high-speed video signal processing. The design of the VLSI circuit of the 2-D DCT system is in progress and will be reported in the future.

VI. Applications to HDTV Systems

In recent years, the focus of video signal processing research has been concentrated on high-definition television (HDTV), which will become future standard for the next generation television [32]. According to CCIR Recommendation 601, the bit rate for transmitting an uncompressed digital HDTV is about 1 Gbps. This bit rate is too high even for broadband ISDN (BISDN) [32]. Furthermore, video signals contain a great deal of redundancy when psychological and visual effects are considered. To make HDTV systems practical, bit rate reduction and data compression are indispensable. In the past decades, many studies have been conducted on differential pulse code modulation (DPCM), subband coding, and transform coding (especially DCT) to achieve bit rate reduction [30], [35], [36]. DCT has obtained most attention due to its diverse attractive features. DCT approaches the statistical optimal transform, the Karhunen-Loeve transform (KLT), which minimizes the mean square errors, for highly correlated signals [1]. Additionally, DCT has superior energy compaction properties for transform coding. Many HDTV systems based on DCT coding schemes show satisfactory speed and promising performance [23], [26]–[28], [33]–[35]. A commonly used encoder configuration of the DCT based source coding is shown in Fig. 12 [26], [34]. The DCT is performed on the 2-D video signals with block size 8 × 8, which is widely used due to its acceptable SNR and implementation complexity [34].

Although there is no uniform standard for HDTV, the interlaced mode with 1080 active lines per frame, 30 Hz frame frequency, and 2:1 interlaced ratio is presently under wide investigation due to its reasonable data rate [24]. With an assumption coding each pel (luminance and chrominance) with 2 b, the bit rate required for transmission of the video signal under interlaced modes is 119.232 Mb/s which satisfies the requirement of the 140 Mb/s H4 hierarchy level [23] and allows sufficient margin for error protection and auxiliary data. The 4:2:2 YUV signals are obtained from RGB signal by A/D converters and coordinate rearrangements. The intrafield 2-D DCT is used for data compression. The transformed signal is processed by an entropy encoder, which is usually combined with the run-length coder and variable length encoder. The run-length coder can reduce the bit rate by coding every sequence of zeros with a single codeword. The variable length coder encodes the DCT coefficients with a variable length code adapted to their probability density function (pdf) distribution.

Most of the 2-D DCT implementations are based on the row–column decomposition methods. Although fast algorithms exist for the 1-D DCT, the second 1-D DCT cannot start until all the first 1-D DCTs are completed. To speed up the operations, one method is to execute the first 1-D DCT in parallel. For the 8 × 8 case, there are 8 1-D DCT blocks to perform the first transform simultaneously. Assuming that each signal is 10-b long, in order to satisfy the precision, then the total number of bits required in the input is 640 b, which is not practical in the circuit realizations. From this point of view, our serial input 2-D DCT system is more practical in hardware implementations. Moreover, if the speed of the circuit components, such as the ROM and adder, is high enough, our 2-D DCT system can be executed as fast as the sample clocking rate.

Although our 2-D DCT implementations are effective, transforming a video frame of 1080 × 1920 still requires intensive computations. Therefore, we designed a 2-D DCT architecture suitable for the HDTV system to achieve higher performance. The block diagram of the 2-D DCT encoder is shown in Fig. 13, where five 2-D DCT chips are included. Five chips were used because the ratio of pixel numbers per line for luminance signal Y and color difference signals U and V is 4:2:2. As the sampling frequency of HDTV is very high, the pixels of Y are divided into four groups, in order to carry out DCT in parallel. Additionally, the color difference signals U and V are switched alternatively to another DCT coder. The scanning processor shown in Fig. 13 is used to divide the signal into four luminance components and one color difference component. The outputs of the 2-D DCT transformed data are sent to the entropy encoder in parallel or through multiplexers. Since the transform block size is 8 × 8, we divided the frame into 135 × 240 blocks and 240 channels as shown in Fig. 14. The 2-D DCT are executed on each channel whose scanning pattern is shown in Fig. 14. This scanning pattern reflects the fact that our system is based on row by row scanning order and is fully pipelined. Thus, such a scanning method would maximize the system throughput.

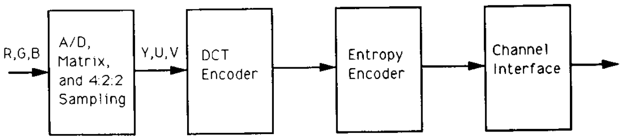

Fig. 12. System diagram of the DCT-based HDTV coder.

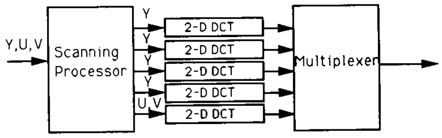

Fig. 13. The block diagram of the DCT encoder.

VII. Conclusions

In this paper, we propose a new 2-D DCT algorithm based on a frame-recursive approach. The resulting 2-D DCT architectures can be obtained by using only two 1-D DCT arrays, at the same time, the transposition procedure is eliminated. It, therefore, does not have the drawback of the row-column decomposition method in which a transposition is needed between the first and the second 1-D DCT. In addition, this algorithm generates the 2-D DCT and the 2-D DSCT or the 2-D DCST simultaneously. There are two methods, the prematrix method and the postmatrix method, to realize the moving frame 2-D DCT architecture. From the postmatrix method, the block 2-D DCT architecture is developed. These architectures are fully pipelined with throughput rate N clock cycles for an $N \times N$ input frame. As to the hardware complexity, the structures contain two 1-D DCT arrays, each with N lattice modules that are modified normal form digital filters with different multiplying coefficients. The total number of multipliers required in the system is $8N$. Because of the regularity and efficiency of the systems, they are very suitable for VLSI implementation for the high-speed HDTV systems.

Many HDTV systems based on the DCT coding shows satisfactory speed and promising performance since the DCT has superior energy compaction property. However, most of the 2-D DCT portion of the HDTV systems are still implemented by row–column decomposition methods. To speed up the throughput, the first DCT operation in the row–column method has to be performed in parallel, although it is not practical in circuit realizations because of the hardware complexity. In view of these facts, our serial input 2-D DCT system is more feasible in HDTV applications. The parallel 2-D DCT architecture and the scanning pattern proposed in Section VI can process the video data in real time and eliminate the waiting time in the DCT codings so that the system performance can be maximized. Consequently, our real-time parallel and fully pipelined 2D-DCT structure is

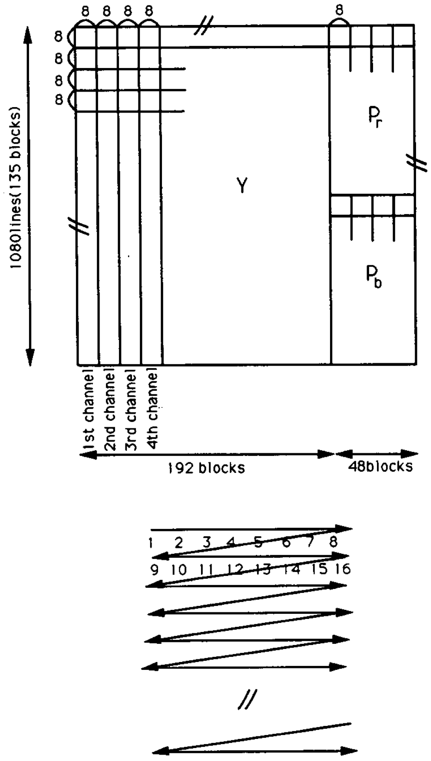

Fig. 14. Block construction of a video frame and proposed scanning pattern.

very attractive in high-speed transmission systems where every arrived data can be processed immediately.

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] 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.

[3] N. S. Jayant and P. Noll, *Digital Coding of Waveform*. Englewood Cliffs, NJ: Prentice Hall, 1984.

[4] D. E. Dudgeon and R. M. Mersereau, *Multidimensional Digital Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1984.

[5] N. H. Lee and Y. Yasuda, "New 2-D systolic array algorithm for DCT/DST," *Electron. Lett.*, vol. 25, pp. 1702-1704, 1989.

[6] K. J. R. Liu and C.-T. Chiu, "Unified Parallel Lattice Structures for Time-Recursive Discrete Cosine/Sine/Hartley Transforms," submitted to *IEEE Trans. Signal Processing*.

[7] W. Ma, "2-D DCT systolic array implementation," *Electron. Lett.*, vol. 27, no. 3, pp. 201-202, Jan. 1991.

[8] H. G. Musmann, P. Pirsch, and H. G. Grallert, "Advances in picture coding," *Proc. IEEE*, vol. 73, pp. 523-548, April 1985.

[9] H. J. Nussbaumer and P. Quandale, "Fast polynomial transform computation of 2-D DCT," in *Proc. Int. Conf. Digital Signal Processing*,Florence, Italy, 1981, pp. 276-283.

[10] N. I. Cho and S. U. Lee, "Fast algorithm and implementation of 2-D Discrete Cosine Transform," *IEEE Trans. Circuits Syst.*, vol. 38, no. 3, pp. 297-305, March 1991.

[11] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, 2nd edition. New York: Academic Press, 1982.

[12] 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.

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

[14] M. Vetterli, "Fast 2-D discrete cosine transform," *IEEE ICASSP Proc.*, pp. 1538-1541, March 1985.

[15] P. Duhamel and C. Guillemot, "Polynomial transform computation of the 2-D DCT," *IEEE ICASSP Proc.*, pp. 1515-1518, March 1990.

[16] G. Peceli, "A common structure for recursive discrete transforms," *IEEE Trans. Circuits Syst.*, vol. 33, no. 10, pp. 1035-10386, Oct. 1986.

[17] K. Rose, A. Heiman, and I. Dinstein, "DCT/DST alternate-transform image coding," *IEEE Trans. Commun.*, vol. 38, no. 1, pp. 94-101, Jan. 1990.

[18] B. Silkstrom *et al.*, "A high speed 2-D discrete cosine-transform," *Integration, VLSI J.*, vol. 5, pp. 159-163, 1987.

[19] M. T. Sun, T. C. Chen, and A. M. Gottlieb, "VLSI implementation of a 16 × 16 discrete cosine transform," *IEEE Trans. Circuits Syst.*, vol. 36, no. 4, pp. 610-617, Apr. 1989.

[20] U. Totzek, F. Matthiesen, S. Wohlleben, and T. G. Noll, "CMOS VLSI implementation of the 2D-DCT with linear processor array," *IEEE ICASSP Proc.*, pp. 937-940, May 1990.

[21] S. A. White, "High-speed distributed-arithmetic realization of a second-order normal-form digital filter," *IEEE Trans. Circuits Syst.*, vol. 33, no. 10, pp. 1036-1038, Oct. 1986.

[22] S. A. White, "Applications of distributed-arithmetic to digital signal processing: A tutorial review," *IEEE ASSP Mag.*, pp. 4-19, July 1989.

[23] M. Barbero, S. Cucchi, and H. Bailon, "A flexible architecture for a HDTV codec based on DCT," *Signal Processing of HDTV, II*, L. Chiariglione, ed. City: Elsevier, 1990, pp. 587-594.

[24] M. Barbero, S. Cucchi, and M. Stroppiana, "Coding strategies based on DCT for the transmission of HDTV," *Signal Processing of HDTV*, L. Chiariglione, ed. City: Elsevier, 1988, pp. 503-508.

[25] J. A. Bellisio and S. Chu, "Television coding for broadband ISDN," *IEEE Globecom'86*, Houston, TX, vol. 2, Dec. 1-4, 1986, pp. 894-890.

[26] W. H. Chen and W. K. Pratt, "Scene adaptive coder," *IEEE Trans. Commun.*, vol. Com-32, no. 3, pp. 225-232, March 1984.

[27] S. Cucchi and F. Molo, "DCT-based television codec for DS3 digital transmission," *SMPTE J.*, pp. 640-646, Sept. 1989.

[28] S. S. Dixit and J. B. Nardone, "A variable bit rate layered DCT video coder for packet switched (ATM) networks," *IEEE ICASSP Proc.*, pp. 2253-2256, May 1990.

[29] D. LeGall, H. Gaggioni, and C. T. Chen, "Transmission of HDTV signals under 140 Mbits/s using a subband decomposition and discrete cosine transform coding," in *Proc. 2nd Int. Workshop on Signal Processing on HDTV*, Feb. 29-Mar. 2, 1988, L'Aquilla Italy.

[30] R. K. Jurgen, "The challenges of digital HDTV," *IEEE Spectrum*, pp. 28-30, 71-73, April 1991.

[31] K. Kinoshita, T. Nakahashi, and Eto, "130 M bit/s (H4 rate) HDTV Codec based on the DCT algorithms," *Electron. Lett.*, vol. 26, no. 16, pp. 1245-1246, Aug. 1990.

[32] R. L. Nickelson, "The evolution of HDTV in the work of CCIR," *IEEE Trans. Broadcast.*, vol. 35, no. 3, pp. 250-258, Sept. 1989.

[33] W. Paik, "Digicipher-all digital, channel, compatible, HDTV broadcast system," *IEEE Trans. Broadcast.*, vol. 36, no. 4, pp. 245-254, Dec. 1990.

[34] J. Suzuki, M. Nomura, and S. Ono, "Comparative study of transform coding for super high definition images," *IEEE ICASSP Proc.*, pp. 2257-2260, May 1990.

[35] K. H. Tzou, T. C. Chen, P. E. Fleischer, and M. L. Liou, "Compatible HDTV coding for broadband ISDN," in *Proc. Globecom '88*, Nov. 1988, pp. 743-749.

[36] Y. Yashima, and K. Sawada, "100 Mbit/s HDTV transmission using a high efficiency codec," *Signal Processing of HDTV, II*, L. Chiariglione, ed. City: Elsevier, 1990, pp. 587-594.

Ching-Te Chiu (S'90) received the B.S. and M.S. degree in electrical engineering from National Taiwan University, Taiwan, in 1986 and 1988, respectively. She is currently pursuing the Ph.D. degree in electrical engineering at the University of Maryland, College Park.

Her research experience includes as a summer research student at Electronics Research Service Organization (ERSO), National Taiwan Institute of Technology in 1987. Since 1989, she has been a research assistant in electrical engineering at the University of Maryland, College Park. Her current research interests include signal processing, VLSI architectures and algorithms, image processing, and HDTV systems.

K. J. Ray Liu (S'86-M'90) received the B.S. degree in electrical engineering from National Taiwan University in 1983, the M.S.E. degree in electrical engineering and computer science from the University of Michigan, Ann Arbor in 1987, and the Ph.D. degree in electrical engineering from the University of California, Los Angeles, in 1990.

During 1983-1985, he served in the Signal Corps, Taiwan, as a Communications Officer. He then became a Teaching and Research Assistant at the University of Michigan and the University of California, Los Angeles. He is currently an Assistant Professor of the Electrical Engineering Department and Systems Research Center, University of Maryland, College Park. His research interest include parallel processing algorithms and architectures for signal/image processing and communications, adaptive signal processing, spectral estimation, video signal processing, fault-tolerant computing in VLSI systems, design automation for DSP VLSI systems, and fast algorithms.

Dr. Liu was awarded the President Research Partnership from the University of Michigan in 1987, and the University Fellowship and the Hortense Fishbaugh Memorial Scholarship from UCLA in 1987-88 and 1989 respectively. He was also awarded the Outstanding Graduate Student Award in Science and Engineering from the Taiwanese-American Foundation. He is a member of the Association of Computing Machinery and Society for Industrial and Applied Mathematics.