On the Realization of Discrete Cosine Transform Using the Distributed Arithmetic

Yuk-Hee Chan, *Student Member, IEEE,* and Wan-Chi Siu, *Senior Member, IEEE*

Abstract—**In this paper, we propose a unified approach for the realization of forward and inverse discrete cosine transforms. By making use of this approach, one can realize an odd prime length DCT/IDCT with two half-length convolutions without extra overheads in terms of the number of multiplications. This formulation is most suitable for the realization using the distributed arithmetic. In such a case, typical convolvers can be used as the core unit for the hardware implementation of the transforms. Hence, an efficient unified DCT/IDCT chip can be designed. A 2-D 11 × 11 unified DCT/IDCT chip is also proposed to demonstrate the superiority of the proposed formulation in this paper. The proposed architecture can easily meet the speed requirement of 14.3-MHz real-time operation with the current 2-μm CMOS technology.**

I. Introduction

THE DISCRETE cosine transform (DCT) [1] is widely used in digital image processing, especially in image transform coding, as it performs much like the optimal Karhunen–Loeve transform (KLT) [2] under a variety of criteria. Many algorithms [3]–[14] for the computation of the DCT have been proposed since the introduction of the DCT by Ahmed, Natarajan, and Rao [1] in 1974. However, though most of them are good software solutions to the realization of DCT, only a few of them are really suitable for VLSI implementation.

Cyclic convolution plays an important role in digital signal processing due to its nature of easy implementation. Specifically, there exists a number of well-developed convolution algorithms [15] and it can be easily realized through modular and structural hardware such as distributed arithmetic [16] and systolic array [17].

The way of data movement forms a significant part in the determination of the efficiency of the realization of a transform using the distributed arithmetic. The realization of a cyclic convolution with the distributed arithmetic requires only simple table look-up technique and some simple rotations of the corresponding data set. Hence, the cyclic convolution structure can be considered as the simplest form that is most suitable to be realized with the distributed arithmetic. It is because of this reason, one may consider that the basic criterion for the realization of a transform using the distributed arithmetic relies on the possibility of having an efficient way to convert the transform into the cyclic convolution form. If we could be able to convert a transform into the cyclic convolution form with the minimum number of operations, it would imply an optimal approach for the realization of the transform using the distributed arithmetic.

Some basic formulations [8]–[11] have been suggested for the realization of the DCT using the distributed arithmetic. In their formulations, they either still required some extra multiplications for their formulations [9], [10], or have to use cyclic convolutions of different lengths [8], [11]. The former case has the major problem that it violates the major advantage of the distributed arithmetic which replaces multiplications by additions. The latter case requires relatively complicated circuitry to allow the realization of cyclic convolutions of variable lengths. Different from the above approaches, one may also convert the DCT into the Discrete Fourier Transform (DFT) [3], [13] and make use of the famous algorithms [18], [19] to convert the corresponding DFT into cyclic convolution form. Indeed, this is a possible approach; however, it turns a real transform into a transform with complex numbers. The realization could still be complicated even if some simplification techniques are to be applied.

In this paper, we propose an algorithm to convert an odd prime length DCT/IDCT into two half-length cyclic convolutions directly. This algorithm involves no multiplication during the conversion and suggests a possible solution to design a unified DCT/IDCT chip. Due to the nature of the structure, this algorithm is most suitable for the VLSI implementation using the distributed arithmetic. A 2-D 11 × 11 unified DCT/IDCT chip design is also provided in this paper to demonstrate the superiority of the proposed algorithm.

II. One-Dimensional DCT

The DCT [1] of data $\{y(i): i = 0, 1 \cdots N-1\}$ is given by the following:

$$Y(k) = \sum_{i=0}^{N-1} y(i) \cos\left(\frac{\pi}{2N}(2i+1)k\right), \quad k = 0, 1 \cdots N-1. \quad (1)$$

Manuscript received July 30, 1991; revised July 15, 1992. This paper was recommended by Associate Editor M. A. Soderstrand.

The authors are with the Department of Electronic Engineering, Hong Kong Polytechnic, Hung Hom, Kowloon, Hong Kong.

IEEE Log Number 9204228.

1057-7122/92$03.00

If N is an odd number, there exists a bijective mapping on the set $\{i{:}i = 0, 1 \cdots N-1\}$:

$$\xi(i) = \frac{\langle N - 2i \rangle_{2N} - 1}{2}, \quad \text{for } i = 0,1 \cdots N-1. \quad (2)$$

For example, if $N = 11$, we have $\{\xi(i) = 5, 4, 3, 2, 1, 0, 10, 9, 8, 7, 6\}$ where $i = 0, 1, 2 \cdots 10$ accordingly. By making use of this bijective mapping, we can split (1) and rewrite it as

$$\begin{cases} Y(0) = \sum_{i=0}^{N-1} f(i) \\ Y(2k) = (-1)^k \{A(k) + f(0)\} \\ Y(N-2k) = (-1)^{k+1} B(k) \end{cases} \quad \text{for } k = 1,2 \cdots (N-1)/2 \quad (3)$$

where

$$\begin{cases} A(k) = \sum_{i=1}^{N-1} f(i) \cos\left(\frac{2\pi ik}{N}\right) \\ B(k) = \sum_{i=1}^{N-1} h(i) \sin\left(\frac{2\pi ik}{N}\right) \end{cases} \quad \text{for } k = 1,2 \cdots (N-1)/2 \quad (4)$$

$$\begin{cases} f(i) = y(\xi(i)) \\ h(i) = (-1)^{\xi(i)} y(\xi(i)) \end{cases} \quad \text{for } i = 0,1 \cdots N-1. \quad (5)$$

If N is an odd prime P, there exist two bijective mappings defined as

$$\begin{cases} \varphi(i) = \langle g^i \rangle_P & \text{for } i = 1,2 \cdots P-1 \\ \zeta(k) = \langle g^{-k} \rangle_P & \text{for } k = 1,2 \cdots P-1 \end{cases} \quad (6)$$

where g is a primitive root of P.

To make use of these two mappings, one can redefine sequences $\{A(k)\}$ and $\{B(k)\}$ for $k = 1, 2 \cdots P-1$ as

$$A(k) = \sum_{i=1}^{P-1} f(i) \cos\left(\frac{2\pi ik}{P}\right) \quad (7a)$$

$$B(k) = \sum_{i=1}^{P-1} h(i) \sin\left(\frac{2\pi ik}{P}\right) \quad (7b)$$

Then both $A(k)$ and $B(k)$ defined in (7) can be converted into a $(P-1)$-length cyclic convolutions by mapping i and k to $\varphi(i)$ and $\zeta(k)$, respectively. In formulation, we have

$$A(\zeta(k)) = \sum_{i=1}^{P-1} f(\varphi(i)) \cos\left(\frac{2\pi}{P}\langle g^{-(k-i)} \rangle_P\right) \quad \text{for } k = 1,2 \cdots P-1 \quad (8a)$$

$$B(\zeta(k)) = \sum_{i=1}^{P-1} h(\varphi(i)) \sin\left(\frac{2\pi}{P}\langle g^{-(k-i)} \rangle_P\right) \quad \text{for } k = 1,2 \cdots P-1. \quad (8b)$$

However, to make the algorithm more efficient, we can make a further simplification on (8a) and (8b). In particular, as

$$\begin{cases} \cos\left(\frac{2\pi}{P}\langle g^{((P-1)/2+i-k)} \rangle_P\right) = \cos\left(\frac{2\pi}{P}\langle g^{(i-k)} \rangle_P\right) \\ \sin\left(\frac{2\pi}{P}\langle g^{((P-1)/2+i-k)} \rangle_P\right) = -\sin\left(\frac{2\pi}{P}\langle g^{(i-k)} \rangle_P\right) \end{cases} \quad \text{for } i = 1,2 \cdots (P-1)/2 \quad (9)$$

and

$$\begin{cases} A(\zeta((P-1)/2 + k)) = A(\zeta(k)) \\ B(\zeta((P-1)/2 + k)) = -B(\zeta(k)) \end{cases} \quad \text{for } k = 1,2 \cdots (P-1)/2 \quad (10)$$

then (8a) and (8b) can be rewritten as (11a) and (11b), respectively:

$$A(\zeta(k)) = \sum_{i=1}^{(P-1)/2} \{f(\varphi(i)) + f(\varphi((P-1)/2 + i))\} \cdot \cos\left(\frac{2\pi}{P}\langle g^{-(k-i)} \rangle_P\right) \quad \text{for } k = 1,2 \cdots (P-1)/2 \quad (11a)$$

$$B(\zeta(k)) = \sum_{i=1}^{(P-1)/2} \{h(\varphi(i)) - h(\varphi((P-1)/2 + i))\} \cdot \sin\left(\frac{2\pi}{P}\langle g^{-(k-i)} \rangle_P\right) \quad \text{for } k = 1,2 \cdots (P-1)/2. \quad (11b)$$

Equations (11a) and (11b) are exactly a $(P-1)/2$ length cyclic convolutions and a $(P-1)/2$-length skew-cyclic convolution respectively. Hence, $A(k)$ and $B(k)$ for $k = 1, 2 \cdots (P-1)/2$ defined as (4) can be realized through two $(P-1)/2$-length convolutions (one cyclic convolution and one skew-cyclic convolution) with an additional cost of $P-1$ additions.

Let us use an example with $P = 11$ (primitive root $g = 2$) to clarify our approach.

First of all, we realize sequences $\{A(k){:}k = 1, 2 \cdots 5\}$ and $\{B(k){:}k = 1, 2 \cdots 5\}$ via a 5-length cyclic convolution and a 5-length skew-cyclic convolution, respectively. In

particular, by making use of (2), (5), (6), and (11a) we have

$$\begin{bmatrix} A(5) \\ A(3) \\ A(4) \\ A(2) \\ A(1) \end{bmatrix} = \begin{bmatrix} c(1) & c(2) & c(4) & c(3) & c(5) \\ c(5) & c(1) & c(2) & c(4) & c(3) \\ c(3) & c(5) & c(1) & c(2) & c(4) \\ c(4) & c(3) & c(5) & c(1) & c(2) \\ c(2) & c(4) & c(3) & c(5) & c(1) \end{bmatrix} \cdot \begin{bmatrix} y(3) + y(7) \\ y(1) + y(9) \\ y(8) + y(2) \\ y(0) + y(10) \\ y(6) + y(4) \end{bmatrix}$$

where $c(n) = \cos(2n\pi/11)$

and, by making use of (2), (5), (6), and (11b), we have

$$\begin{bmatrix} -B(5) \\ B(3) \\ -B(4) \\ -B(2) \\ -B(1) \end{bmatrix} = \begin{bmatrix} s(1) & s(2) & s(4) & -s(3) & s(5) \\ -s(5) & s(1) & s(2) & s(4) & -s(3) \\ s(3) & -s(5) & s(1) & s(2) & s(4) \\ -s(4) & s(3) & -s(5) & s(1) & s(2) \\ -s(2) & -s(4) & s(3) & -s(5) & s(1) \end{bmatrix} \cdot \begin{bmatrix} -y(3) + y(7) \\ -y(1) + y(9) \\ y(8) - y(2) \\ y(0) - y(10) \\ y(6) - y(4) \end{bmatrix}$$

where $s(n) = \sin(2n\pi/11)$.

Then the coefficients $\{Y(k): k = 0, 1 \cdots 10\}$ can be computed with (3):

$$\begin{bmatrix} Y(2) \\ Y(4) \\ Y(6) \\ Y(8) \\ Y(10) \end{bmatrix} = \begin{bmatrix} -\{A(1) + y(5)\} \\ A(2) + y(5) \\ -\{A(3) + y(5)\} \\ A(4) + y(5) \\ -\{A(5) + y(5)\} \end{bmatrix},$$

$$\begin{bmatrix} Y(1) \\ Y(3) \\ Y(5) \\ Y(7) \\ Y(9) \end{bmatrix} = \begin{bmatrix} B(5) \\ -B(4) \\ B(3) \\ -B(2) \\ B(1) \end{bmatrix}$$

and

$$\begin{aligned} Y(0) = {} & \{y(3) + y(7)\} + \{y(1) + y(9)\} \\ & + \{y(2) + y(8)\} + \{y(0) + y(10)\} \\ & + \{y(4) + y(6)\} + y(5). \end{aligned}$$

As the sequence $\{f(\varphi(i)) + f(\varphi((P-1)/2 + i)): i = 1, 2 \cdots (P-1)/2\}$ is computed during the realization of $A(\zeta(k))$, the computation of $Y(0)$ requires $(P-1)/2$ additions only. In other words, a P-length DCT can be realized with two $(P-1)/2$ length convolutions with a cost of $2(P-1)$ additions totally.

III. One-Dimensional IDCT

The IDCT of data $\{Y(k): k = 0, 1 \cdots N-1\}$ is given by the following:

$$y(i) = \sum_{k=0}^{N-1} Y(k) \cos\left(\frac{\pi}{2N}(2i+1)k\right), \quad i = 0, 1 \cdots N-1. \quad (12)$$

If N is an odd number, (12) can be rewritten as

$$\begin{aligned} y(i) = {} & Y(0) + \sum_{k=1}^{(N-1)/2} Y(2k) \cos\left(\frac{\pi k}{N}(2i+1)\right) \\ & + (-1)^i \sum_{k=1}^{(N-1)/2} Y(N-2k) \sin\left(\frac{\pi k}{N}(2i+1)\right) \\ & i = 0, 1 \cdots N-1. \quad (13) \end{aligned}$$

By making use of the bijective mapping defined in (2), equation (10) can be further rewritten as

$$\begin{cases} y(\xi(0)) = Y(0) + \sum_{k=1}^{(N-1)/2} (-1)^k Y(2k) \\ y(\xi(i)) = Y(0) + G(i) + (-1)^{\xi(i)} H(i) \\ y(\xi(N-i)) = Y(0) + G(i) - (-1)^{\xi(i)} H(i) \end{cases}$$

for $i = 1, 2 \cdots (N-1)/2$ (14)

where

$$G(i) = \sum_{k=1}^{(N-1)/2} \{(-1)^k Y(2k)\} \cos\left(\frac{2\pi ik}{N}\right)$$

for $i = 1, 2 \cdots (N-1)/2$ (15a)

$$H(i) = \sum_{k=1}^{(N-1)/2} \{(-1)^{k+1} Y(N-2k)\} \sin\left(\frac{2\pi ik}{N}\right)$$

for $i = 1, 2 \cdots (N-1)/2$. (15b)

Obviously, by making use of the zero-padding technique, we can redefine sequences $\{G(i)\}$ and $\{H(i)\}$ as

follows:

$$G(i) = \sum_{k=1}^{N-1} \{Y^O(k)\} \cos\left(\frac{2\pi ik}{N}\right) \quad \text{for } i = 1, 2 \cdots N-1 \quad (16)$$

where

$$Y^O(k) = \begin{cases} (-1)^k Y(2k) & \text{for } k = 1, 2 \cdots (N-1)/2 \\ 0 & \text{else} \end{cases} \quad (17)$$

and,

$$H(i) = \sum_{k=1}^{N-1} \{Y'(k)\} \sin\left(\frac{2\pi ik}{N}\right) \quad \text{for } i = 1, 2 \cdots N-1 \quad (18)$$

where

$$Y'(k) = \begin{cases} (-1)^{k+1} Y(N-2k) & \text{for } k = 1, 2 \cdots (N-1)/2 \\ 0 & \text{else.} \end{cases} \quad (19)$$

Then $G(i)$ and $H(i)$ are exactly in the form of (7a) and (7b), respectively. In the previous section, we have proved that equations in the form of (7a) and (7b) can be converted into cyclic convolution form easily by using the mappings defined in (6) if N is an odd prime P. By using a similar approach, we can rewrite (16) and (18) as the following:

$$\begin{cases} G(\varphi((P-1)/2 + i)) = G(\varphi(i)) \\ G(\varphi(i)) = \sum_{k=1}^{(P-1)/2} \{Y^O(\zeta(k)) \\ \quad + Y^O(\zeta((P-1)/2 + k))\} \cos\left(\frac{2\pi}{P}\langle g^{i-k}\rangle_P\right) \end{cases} \quad \text{for } i = 1, 2 \cdots (P-1)/2 \quad (20)$$

$$\begin{cases} H(\varphi((P-1)/2 + i)) = -H(\varphi(i)) \\ H(\varphi(i)) = \sum_{k=1}^{(P-1)/2} \{Y'(\xi(k)) \\ \quad - Y'(\xi((P-1)/2 + k))\} \sin\left(\frac{2\pi}{P}\langle g^{i-k}\rangle_P\right) \end{cases} \quad \text{for } i = 1, 2 \cdots (P-1)/2. \quad (21)$$

Equations (20) and (21) are $(N-1)/2$-length cyclic convolution and skew-cyclic convolution, respectively. In such case, an odd prime length IDCT can also be realized via two half-length convolutions similar to the case for the DCT.

Note that no multiplication is involved as overheads for the conversion of an odd prime P-length IDCT into convolutions. As either $Y'(\zeta(k))$ or $Y'(\zeta((P-1)/2 + k))$ is zero for $k = 1, 2 \cdots (P-1)/2$, no addition is required to compute the sequence $\{Y'(\zeta(k)) - Y'(\zeta((P-1)/2 + k)): k = 1, 2 \cdots (P-1)/2\}$. A similar case occurs during the computation of the sequence $\{Y^o(\zeta(k)) + Y^o(\zeta((P-1)/2 + k)): k = 1, 2 \cdots (P-1)/2\}$. Actually, only $2(P-1)$ additions are required during the conversion. In other words, a P-length IDCT can be realized through two $(P-1)/2$-length convolutions with a cost of $2(P-1)$ additions. This is exactly the same cost that a P-length DCT is required to be realized with convolutions.

Again, we use the example with $N = 11$ to clarify our approach.

To compute the sequence $\{G(i): i = 1, 2 \cdots 5\}$, we can make use of (20), (17), and (6),

$$\begin{bmatrix} G(2) \\ G(4) \\ G(3) \\ G(5) \\ G(1) \end{bmatrix} = \begin{bmatrix} c(1) & c(5) & c(3) & c(4) & c(2) \\ c(2) & c(1) & c(5) & c(3) & c(4) \\ c(4) & c(2) & c(1) & c(5) & c(3) \\ c(3) & c(4) & c(2) & c(1) & c(5) \\ c(5) & c(3) & c(4) & c(2) & c(1) \end{bmatrix} \begin{bmatrix} -Y(10) \\ -Y(6) \\ Y(8) \\ Y(4) \\ -Y(2) \end{bmatrix}$$

where $c(n) = \cos(2n\pi/11)$.

On the other hand, we can obtain sequence $\{H(i): i = 1, 2 \cdots 5\}$ by making use of (21), (19), and (6):

$$\begin{bmatrix} H(2) \\ H(4) \\ -H(3) \\ H(5) \\ -H(1) \end{bmatrix} = \begin{bmatrix} s(1) & -s(5) & s(3) & -s(4) & -s(2) \\ s(2) & s(1) & -s(5) & s(3) & -s(4) \\ s(4) & s(2) & s(1) & -s(5) & s(3) \\ -s(3) & s(4) & s(2) & s(1) & -s(5) \\ s(5) & -s(3) & 5(4) & 5(2) & s(1) \end{bmatrix} \cdot \begin{bmatrix} -Y(1) \\ Y(5) \\ Y(3) \\ Y(7) \\ -Y(9) \end{bmatrix}$$

where $s(n) = \sin(2n\pi/11)$.

Finally, we use (14) to compute the final result, $\{y(i):i = 0, 1 \cdots 10\}$:

$$\begin{bmatrix} y(4) \\ y(3) \\ y(2) \\ y(1) \\ y(0) \end{bmatrix} = \begin{bmatrix} Y(0) + G(1) \\ Y(0) + G(2) \\ Y(0) + G(3) \\ Y(0) + G(4) \\ Y(0) + G(5) \end{bmatrix} + \begin{bmatrix} H(1) \\ -H(2) \\ H(3) \\ -H(4) \\ H(5) \end{bmatrix},$$

$$\begin{bmatrix} y(6) \\ y(7) \\ y(8) \\ y(9) \\ y(10) \end{bmatrix} = \begin{bmatrix} Y(0) + G(1) \\ Y(0) + G(2) \\ Y(0) + G(3) \\ Y(0) + G(4) \\ Y(0) + G(5) \end{bmatrix} - \begin{bmatrix} H(1) \\ -H(2) \\ H(3) \\ -H(4) \\ H(5) \end{bmatrix}$$

and

$$y(5) = Y(0) - Y(2) + Y(4) - Y(6) + Y(8) - Y(10).$$

Both the DCT and the IDCT can be realized via convolutions with the same cost. Specifically, if both of them possess the same length, one can make use of the same convolution module to realize both the forward and the inverse DCT. As cyclic convolution is the core module of this algorithm, this algorithm is most suitable for the realization using the distributed arithmetic and it also suggests an efficient and effective way to design a unified DCT/IDCT chip.

IV. VLSI Implementation of Unified DCT / IDCT Chip

In the preceding sections, we have proposed an algorithm to convert a P-length DCT/IDCT into a half-length cyclic convolution and a skew cyclic convolution. This provides a straightforward but ideal solution for the VLSI implementation of a unified DCT/IDCT chip by making use of the distributed arithmetic.

Consider a cyclic convolution defined as $F(k) = \sum_{\eta=0}^{N-1} g(\eta - k)C(\eta)$. Since $g(\eta - k)$ can be expressed as $g(\eta - k) = -g(\eta - k)_0 + \sum_{j=1}^{M-1} g(\eta - k)_j 2^{-j}$, where M, $g(\eta - k)_j$ and $g(\eta - k)_0$ are the word length, the jth most significant bit, and the sign bit, respectively. After scaling to 2's-complement fractional number, $F(k)$ can be rewritten as $F(k) = \sum_{j=1}^{M-1}\{\sum_{\eta=0}^{N-1} g(\eta - k)_j C(\eta)\}2^{-j} - \sum_{\eta=0}^{N-1} g(\eta - k)_j C(\eta)$. Values of $\sum_{\eta=0}^{N-1} g(\eta - k)_j C(\eta)$ can be precalculated and stored in a ROM with ROM size $= 2^N$ words. Then $F(k)$ can be obtained by M ROM accesses and $M - 1$ shift-additions after $g(n)$'s are available. Note that the same table can be used for the computation of $F(k)$ for any value of k, which is impossible in the case of computing inner products other than a cyclic convolution. Hence, to a certain extent, one can consider that the distributed arithmetic is most suitable for VLSI implementation of cyclic convolutions.

Several high-performance chips have been designed by making use of the distributed arithmetic [20]–[26]. However, in most designs, the distributed arithmetic is used to realize a typical inner product directly without first converting the transform into cyclic convolutions. In such a case, optimal performance of the distributed arithmetic can not be achieved and the consequence of which is the requirement of a large memory size for the construction of the data tables.

A $P \times P$ unified DCT/IDCT can be implemented by the row-column decomposition technique as shown in Fig. 1. In fact, the row-column approach is commonly applied in most 2-D DCT chips due to its flexible and regular nature. We first compute the $PP \times 1$ DCT/IDCT's along each row and store the results in an intermediate array. We then compute the $P\,P \times 1$ DCT/IDCT's along each column to yield the final results. Note that the intermediate memory is realized by a RAM of $P \times P$ words and the transposition operation can be easily achieved by a suitable control of the addresses of the intermediate array.

Fig. 2 shows the block diagram on the one-dimensional unified DCT/IDCT module. The module mainly consists of three operating units, namely, an accumulator, a pre/post-processing unit, and a kernel-processing unit. Note that the whole process is a three-state pipeline. The accumulator is responsible for the computation of the dc term in the DCT mode and the $y((N - 1)/2)$ term in the IDCT mode, which involves additions or subtractions only. A typical accumulator can satisfy this requirement. The pre/post-processing unit is actually a typical adder which is responsible for the preparation of the input data for convolutions in the DCT mode and the computation of the final results from the convolution outputs in the IDCT mode. The arrangement of the pre/post-processing stage and the kernel-processing stage determines the configuration of the unified chip, which can be easily handled with multiplexers. The table provided in Fig. 2 specifies the relationship between the MUX's configuration and the mode configuration of the module.

Both preshuffling and postshuffling of data can be easily done through the table lookup technique. In a typical pipeline design, input data and output data are normally buffered. Hence, if the sequence of the addresses can be generated in such a way that the input or the output data are fetched in a desirable order, then both the preshuffle and the postshuffle can be achieved. As the transform size is typically fixed and small, the desirable address sequence can be precomputed and stored in a small table. In such case, appropriate data can be fetched with indirect addressing method.

The kernel-processing unit basically consists of two convolvers. Both convolvers are realized with the distributed arithmetic. Fig. 3 shows the implementation of a 5-point convolver, which can be used in the VLSI realization of an 11-point unified DCT/IDCT chip. The two convolvers differ from each other in both of their address generators and their lookup tables stored in ROM's. In this example, the internal word length, the word length of data $\{x(i)\}$ and $\{X(k)\}$ are, respectively, 12, 8, and 12 bits. Note that these parameters can always achieve a signal-to-noise ratio of greater than 44 dB under the simulation test.

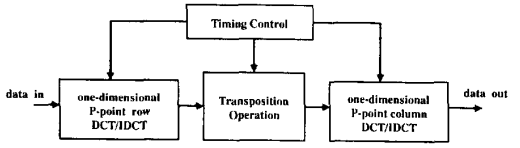


Fig. 1. Block diagram of the row-column approach for 2-D DCT/IDCT.

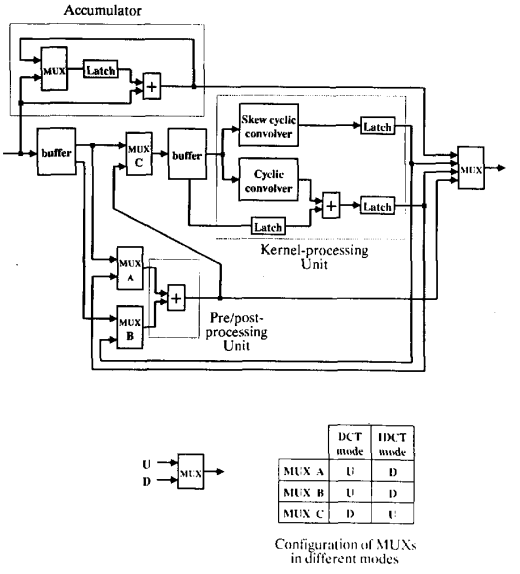


Fig. 2. Block diagram of the one-dimensional unified DCT/IDCT module of the VLSI chip.

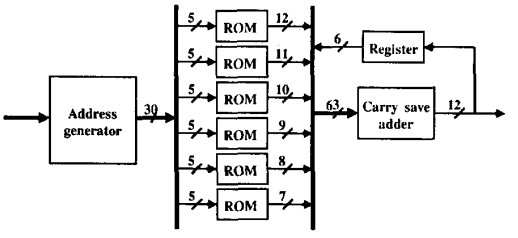


Fig. 3. The construction of the convolvers.

In such a case, the address generator of the cyclic convolver can be implemented with a 60-bit bitwise circular buffer with shift operations. At the beginning of a specific convolution cycle, the input data for the convolutions (five 12-bit words in this example) are loaded into the circular buffer in parallel. In order to make the chip achieve a throughput rate of 1 output per clock cycle, any one of the convolvers has to produce an output every 2 clock cycles. In the first cycle, the six least significant bits of the five data form six 5-bit addresses to access six ROM tables, respectively. All fetched data are summed up with a carry-save adder to form a partial result. In the second cycle, the 6 most significant bits of the five data form another 6 addresses to fetch other six data. These data are then summed up with the shifted partial result to obtain the final result. The circular buffer advances 6 bits and repeats the foregoing procedures until all results are obtained. This completes a full convolution cycle and starts another one by loading another input sequence one clock cycle later. In such a case, the circular buffer rotates 6 bits every clock cycle. Hence, the address generator can actually be implemented with six independent 10-bit bitwise shift registers with parallel load function to release the burden of the clock synchronization of the circular buffer. Note also that a complete convolution cycle spans $P-1$ clock cycles only while one gets P clock cycles to complete a transform. The inputs of the convolvers can be split and loaded into the address generators in two cycles to reduce the input bandwidth of the convolvers.

Each ROM table consists of 32 words. Note that the contents of all ROM tables of a specific convolver are identical. In other words, one can use multiport ROM to save a number of ROM tables. Besides, as shown in Fig. 3, the word lengths of different ROM tables are not necessary identical since the fetched data are not equally significant. These features are obviously superior to other chip designs which use the distributed arithmetic to implement inner products without first converting them into cyclic convolutions.

For the implementation of the address generator of the skew-cyclic convolver, a small additional circuit is required to perform a 2's complement negation to the datum passing through the head of the circular buffer. The contents of the ROM tables are also different from those used in the cyclic convolver.

The silicon efficiency of the unified chip is extremely high. The configuration of the chip, which is controlled by the MUX's, involves the arrangement of the pre/post-processing unit and the kernel unit only. For other typical design [20]–[26], the convolvers have to swap ROM tables whenever the mode of the unified chip is swapped. However, no such step is necessary in the proposed design. By considering that both (11a) and (20) involve the same sequence $\left\{\cos\left(\frac{2\pi}{P}g^n\right): n = 1, 2\cdots(P-1)/2\right\}$, one can find the computation of $\{A(k)\}$ and $\{G(i)\}$ can use the same set of ROM tables. Similar case occurs during the computation of $\{B(k)\}$ and $\{H(i)\}$. For instance, when $P = 11$, for the IDCT realization, we have

$$\begin{bmatrix} G(1) \\ G(5) \\ G(3) \\ G(4) \\ G(2) \end{bmatrix} = \begin{bmatrix} c(1) & c(2) & c(4) & c(3) & c(5) \\ c(5) & c(1) & c(2) & c(4) & c(3) \\ c(3) & c(5) & c(1) & c(2) & c(4) \\ c(4) & c(3) & c(5) & c(1) & c(2) \\ c(2) & c(4) & c(3) & c(5) & c(1) \end{bmatrix} \begin{bmatrix} -Y(2) \\ Y(4) \\ Y(8) \\ -Y(6) \\ -Y(10) \end{bmatrix}$$

and

$$\begin{bmatrix} -H(1) \\ H(5) \\ -H(3) \\ H(4) \\ H(2) \end{bmatrix} = \begin{bmatrix} s(1) & s(2) & s(4) & -s(3) & s(5) \\ -s(5) & s(1) & s(2) & s(4) & -s(3) \\ s(3) & -s(5) & s(1) & s(2) & s(4) \\ -s(4) & s(3) & -s(5) & s(1) & s(2) \\ -s(2) & -s(4) & s(3) & -s(5) & s(1) \end{bmatrix} \cdot \begin{bmatrix} -Y(9) \\ Y(7) \\ Y(3) \\ Y(5) \\ -Y(1) \end{bmatrix}.$$

The two kernel matrices are then respectively identical to the two kernel matrices used for the realization of $[A(5), A(3), A(4), A(2), A(1)]^T$ and $[-B(5), B(3), -B(4), -B(2), -B(1)]^T$ in the DCT realization. Hence, whether the chip is configured to perform a DCT or an IDCT, no modification of the convolvers is necessary. Consequently, nearly no silicon area of the chip is idle in a particular transform. A highly efficient unified chip can be implemented.

Furthermore, as shown in Figs. 1 and 2, the convolvers are the core units of the unified chip and the whole chip involves no multiplier. Since the convolutions are reformulated at the bit level by using the distributed arithmetic, the following advantages can be achieved: 1) no actual multiplication involved as multipliers are replaced by memory look-up tables, 2) high accuracy as it suffers fewer rounding/truncation error than the other structures, 3) possible for modular circuit design as the structure is extremely regular, and 4) simple structure which leads to a saving of gate count and makes routing easy. These features allow a high-speed circuit design composed of memories, adders, and registers only.

The proposed design aims to achieve a throughput rate of 1 output per clock cycle. Obviously, the two convolution modules play a significant role in the unified chip and dominate the timing performance of the whole chip. By making use of the current 2-μm CMOS technology, the proposed architecture can easily meet the speed requirement of 14.3-MHz real-time operation.

V. Conclusions

In this paper, we propose a new algorithm to realize an odd prime P-length DCT with two half-length convolutions (one cyclic convolution and one skew-cyclic convolution). This algorithm can be easily modified to realize an IDCT with odd prime length. In such a case, one can realize both DCT and IDCT with the same convolution module if they possess the same length. As the operations other than the convolutions required for realizing either DCT or IDCT are just $2(P-1)$ additions and some simple permutations, only a small percentage of the unified chip is idle in a particular transform. Hence, one can design a very efficient unified chip. Furthermore, by making use of the distributed arithmetic, the VLSI implementation of the convolution module can result in a very simple and modular structure without multiplier. In other words, an efficient unified DCT/IDCT chip which involves only adders, latches, and memory tables can be implemented in a very straightforward way. These algorithms can also be easily extended to realize a multidimensional DCT/IDCT by using the row-column decomposition technique. A 2-D 11×11 unified DCT/IDCT chip design is also proposed in this paper. The proposed architecture can easily meet the speed requirement of 14.3-MHz real-time operation with the current 2-μm CMOS technology.

References

[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Computers*, vol. C-23, pp. 90–94, 1974.

[2] P. A. Wintz, "Transform picture coding," *Proc. IEEE*, vol. 60, pp. 809–820, July 1972.

[3] M. J. Narasimha and A. M. Peterson, "On the computation of the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-26, pp. 934–936, June 1978.

[4] Z. Wang, "On computing the discrete Fourier and cosine transforms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 1341–1344, Oct. 1985.

[5] M. Vetterli and H. Nussbaumer, "A simple FFT and DCT algorithms with reduced number of operation," *Signal Processing*, vol. 6, pp. 267–278, Aug. 1984.

[6] 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.

[7] 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.

[8] P. Duhamel and H. H'mida, "New 2^n DCT algorithms suitable for VLSI implementation," in *Proc. ICASSP-85*, pp. 780–783, Mar. 1985.

[9] Y. H. Chan and W. C. Siu, "Algorithm for prime length discrete cosine transform," *Electron. Lett.*, vol. 26, pp. 206–208, Feb. 1990.

[10] ——, "A new convolution structure for the realization of discrete cosine transform," in *Proc. ISCAS'90*, pp. 2373–2376, May 1990.

[11] W. Li, "A new algorithm to compute the DCT and its inverse," *IEEE Trans. Signal Processing*, vol. 39, pp. 1305–1313, June 1991.

[12] N. I. Cho and S. U. Lee, "Fast algorithm and implementation of 2-D discrete cosine transform," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 297–305, Mar. 1991.

[13] S. C. Chan, "Efficient index mapping for computing discrete cosine transform," *Electron. Lett.*, vol. 25, pp. 1499–1500, Oct. 1989.

[14] B. G. Lee, "Input and output index mappings for a prime-factor-decomposed computation of discrete cosine transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-37, pp. 237–244, Feb. 1989.

[15] H. J. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms*. New York: Springer-Verlag, 1982.

[16] S. A. White, "Applications of distributed arithmetic to digital signal processing: A tutorial review," *IEEE ASSP Mag.*, vol. 6, pp. 4–19, July 1989.

[17] O. Ersoy, "Semisystolic array implementation of circular, skew circular, and linear convolutions," *IEEE Trans. Computers*, pp. 190–196, 1985.

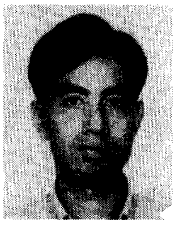
[18] ——, A two-stage representation of DFT and its applications," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 825–831, June 1987.

[19] C. M Rader, "Discrete Fourier transforms when the number of data samples is prime," *Proc. IEEE*, vol. 56, pp. 1107–1108, June 1968.

[20] M. T. Sun, L. Wu, and M. L. Liou, "A concurrent architecture for VLSI implementation of discrete cosine transform," *IEEE Trans. Circuits Syst.*, vol. CAS-34, pp. 992–994, Aug. 1987.

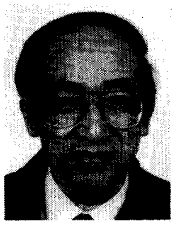
[21] M. Maruyama, H. Uwabu, I. Iwasaki, H. Pujiwara, T. Sakaguchi,

M. T. Sun, and M. L. Liou, "VLSI architecture and implementation of a multi-function forward/ Inverse discrete cosine transform processor," in *Proc. SPIE*, pt. 1, pp. 410–417, Oct. 1990.
[22] N. Demassieux, G. Concordel, J. P. Durandeau, and F. Jutand, "Optimized VLSI architecture for a multiformat discrete cosine transform," in *Proc. ICASSP'87*, pp. 547–550, Apr. 1987.
[23] A. M. Gottlieb, M. T. Sun, and T. C. Chen, "Video rate 16 multiplied by 16 discrete cosine transform IC," in *Proc. IEEE 1988 Custom Integrated Circuits Conf.*, pp. 8.2/1–4, May 1988.
[24] A. Artieri, S. Kritter, F. Jutand, and N. Demassieux, "A one chip VLSI for real time two-dimensional discrete cosine transform," in *Proc. ISCAS'88*, pp. 701–704, June 1988.
[25] J. C. Carlach, P. Penard, and J. L. Sicre, "TCAD: A 27 MHz 8 × 8 discrete cosine transform chip," in *Proc. ICASSP'89*, pp. 2429–2432, May 1989.
[26] T. C. Chen, A. Gottlieb, and M. T. Sun, "VLSI implementation of a 16 × 16 DCT," in *Proc. ICASSP'88*, pp. 1973–1976, Apr. 1988.



Yuk-Hee Chan (S'89) received the B.Sc. (Hons) degree in electronics from the Chinese University of Hong Kong in 1987. He is now working towards the Ph.D. degree in the Department of Electronic Engineering, Hong Kong Polytechnic, Kowloon, Hong Kong.

His research interests include fast computational algorithms, signal processing, image compression, and VLSI techniques.



Wan-Chi Siu (S'77–M'77–SM'90) received the associateship in electronic engineering from Hong Kong Polytechnic, the M.Phil. degree in electronics from the Chinese University of Hong Kong, and the Ph.D. degree in digital signal processing from the Imperial College of Science, Technology and Medicine, London.

Between 1975 and 1980 he was with the Chinese University of Hong Kong, where he was an electronic engineer before he left the Department of Electronics. He joined Hong Kong Polytechnic in 1980, initially as a lecturer, then as senior lecturer, and then as a principal lecturer. He is presently a Reader and the Leader of the Computer Engineering Section of the Department of Electronic Engineering, and is also the Chairman of the Departmental Research Committee. He has published more than 80 research papers. His research interests include digital signal processing, transforms, fast computational algorithms, high-performance computer architecture, parallel processing, fast techniques on image processing and pattern recognition.

Dr. Siu was the Chairman of the Technical Program Committee of the 1987 IEEE Asian Electronic Conference and was also the Chairman of the Technical Program Committee of the 1989 International Symposium on Computer Architecture & Digital Signal Processing organized by the IEE Hong Kong Center. He was a co-chairman of the Technical Program Committee of the IEEE Region 10 Conference on Computer and Communication Systems that was held in Hong Kong in September, 1990, and is now the Chairman of the IEEE Hong Kong Chapter of Signal Processing. He is also chartered engineer.