A Memory-Efficient Realization of Cyclic Convolution and Its Application to Discrete Cosine Transform

Hun-Chen Chen, Jiun-In Guo, Tian-Sheuan Chang, and Chein-Wei Jen

***Abstract*—This paper presents a memory-efficient approach to realize the cyclic convolution and its application to the discrete cosine transform (DCT). We adopt the way of distributed arithmetic (DA) computation, exploit the symmetry property of DCT coefficients to merge the elements in the matrix of DCT kernel, separate the kernel to be two perfect cyclic forms, and partition the content of ROM into groups to facilitate an efficient realization of a one-dimensional (1-D) N-point DCT kernel using $(N$-1)/2 adders or substractors, one small ROM module, a barrel shifter, and $((N-1)/2)+1$ accumulators. The proposed memory-efficient design technique is characterized by rearranging the content of the ROM using the conventional DA approach into several groups such that all the elements in a group can be accessed simultaneously in accumulating all the DCT outputs for increasing the ROM utilization. Considering an example using 16-bit coefficients, the proposed design can save more than 57% of the delay-area product, as compare with the existing DA-based designs in the case of the 1-D seven-point DCT. Finally, a 1-D DCT chip was implemented to illustrate the efficiency associated with the proposed approach.**

***Index Terms*—Cyclic convolution, discrete cosine transform (DCT), distributed arithmetic.**

I. Introduction

DUE to the high computational complexity and real-time processing requirements, the efficient hardware implementation of discrete cosine transform (DCT) is still a challenging problem, which plays a key function in image and signal processing, especially for the demanding multimedia and portable applications. To achieve efficient hardware realization, many researches have been done on realizing the multiplications needed in the DCT through ROM. One is the memory-based systolic array design [1] in which the proposed cyclic convolution-based architecture possesses the features of simple I/O behavior and removes the data redundancy in the DCT coefficients. The others are called distributed arithmetic (DA) based designs [2]–[11]. The DA technique is an efficient method for computing inner products by using ROM tables or adders, and accumulators [2]–[14]. It has been widely adopted in many DSP applications such as discrete Fourier transform (DFT), DCT, convolution, and digital filters. Therefore, there has been great interest in reducing the ROM size required in the implementation of the DA-based architectures [2], [7]–[9], [11]. Most of the DA-based designs exploit some memory reduction techniques such as the partial sum techniques and the offset binary coding (OBC) techniques [2], [8]. The other kind of DA-based design is realized by using adders and accumulators. Chang [12] took advantage of the shared partial sum-of-products and sparse nonzero bits in the fixed input to reduce the number of computations. Guo [13], [14] exploited the feature of cyclic convolution to simplify the computation so that the multiplications and additions can be realized by using a small number of adders. However, their designs are still not efficient enough since they only exploit the constant property of the transform coefficients without considering further optimization in reducing hardware.In this paper, we appropriately combine the features of cyclic convolution and DA technique to propose a memory-efficient architecture in realizing the cyclic convolution and apply it to the one-dimensional (1-D) DCT. By exploiting the cyclic convolution, we find that different DCT outputs can be computed using the same coefficients and the same input data in a rotated order. If we directly realize the DCT using conventional DA technique [2], we find that N ROM modules are used and only one word in a ROM module is accessed at a time in computing the DCT outputs. This reveals a message that the ROM utilization is not good enough. To increase the ROM utilization, we rearrange the contents of ROM in a different way. That is, we first group the candidates of DA inputs with rotated order as the same candidate and then arrange the ROM contents in such a manner that the partial products for accumulating different DCT outputs according to the candidate are grouped together and accessed simultaneously. The partial products arranged in a group should be rotated before accumulating. In this way, the ROM module will contain only few groups of contents and only one ROM module, instead of N identical ROM modules, is needed to compute the 1-D N-point DCT. Unfortunately, due to the rotated input data in the input-data matrix of DCT possess different signs, it is not easy to apply the proposed approach directly to DCT realization. Exploiting the symmetry property of DCT coefficients, we merge the elements in the matrix of DCT kernel and separate the matrix to two perfect cyclic forms. Then, these two smaller perfect cyclic convolution forms can be realized with the proposed design approach efficiently. This realization facilitates reducing the ROM size significantly. As compared with the existing DA-based designs, for an example of 1-D seven-point DCT with 16-bit coefficients, the proposed design can save more than 57% of the delay-area product. Moreover, a 1-D DCT chip was implemented to illustrate the efficiency associated with the proposed approach. This design approach can also be applied to other transforms like the DFT,

Manuscript received December 20, 2002; revised August 17, 2004. This paper was recommended by Associate Editor R. Chandramouli.

H.-C. Chen, T.-S. Chang, and C.-W. Jen are with the Department of Electronics Engineering and the Institute of Electronics, National Chiao Tung University, Hsin-Chu 300, Taiwan, R.O.C. (e-mail: hcchen@swallow.ee.nctu.edu.tw; tschang@swallow.ee.nctu.edu.tw; cwjen@swallow.ee.nctu.edu.tw)

J.-I. Guo is with the Department of Computer Science and Information Engineering, National Chung Cheng University, Chia Yi 621, Taiwan, R.O.C. (e-mail: jiguo@cs.ccu.edu.tw).

Digital Object Identifier 10.1109/TCSVT.2004.842608

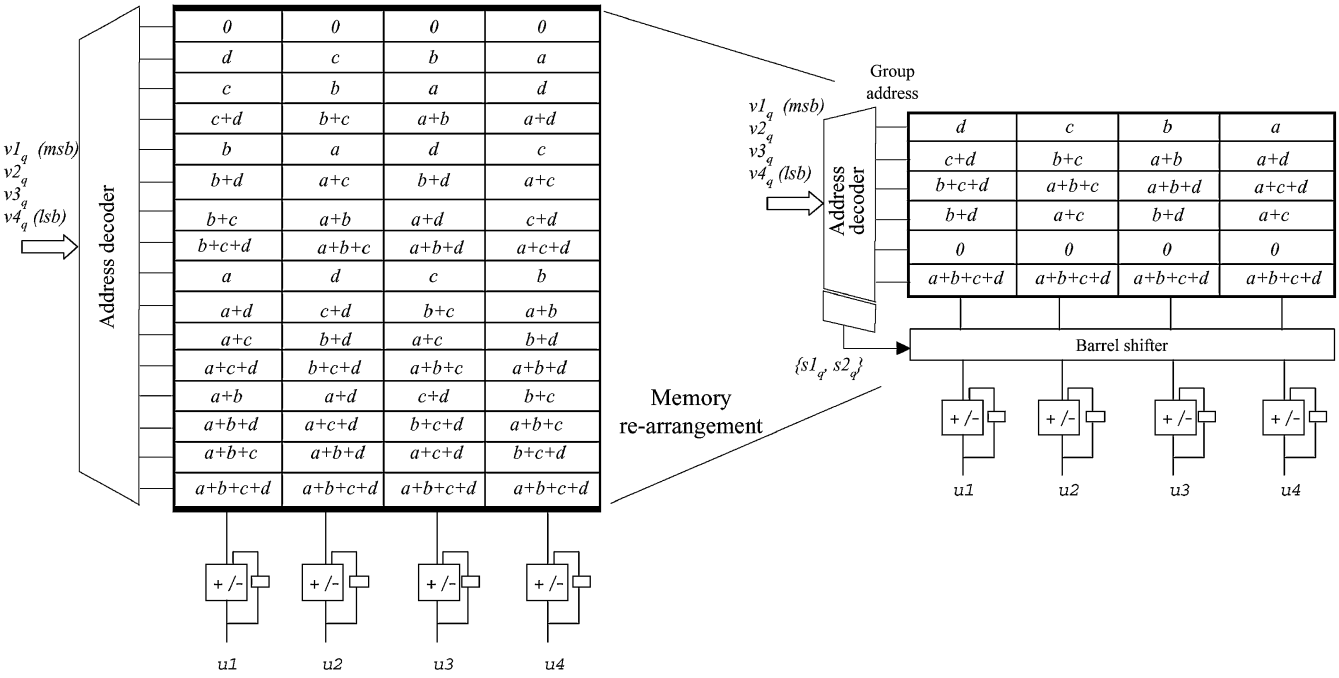


Fig. 1. Proposed memory-efficient architecture and the associated ROM content arrangement in realizing the cyclic convolution example shown in (2).

the discrete Hartley transform (DHT) and the discrete sine transform (DST) used in many DSP applications.

The rest of this paper is organized as follows. In Section II, we illustrate the proposed memory-efficient realization of cyclic convolution and its advantages over the conventional DA-based approach. In Section III, we illustrate the proposed design for 1-D DCT including the algorithm derivation, architecture design, and chip design. In Section IV, we show the performance evaluation of the proposed design with the existing DCT designs. Finally, we conclude this paper in Section V.

II. Memory-Efficient Realization of Cyclic Convolution

A. Proposed Approach

Let us first consider a cyclic convolution example

$$U = \begin{bmatrix} u1 \\ u2 \\ u3 \\ u4 \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ d & a & b & c \\ c & d & a & b \\ b & c & d & a \end{bmatrix} \cdot \begin{bmatrix} v1 \\ v2 \\ v3 \\ v4 \end{bmatrix}. \tag{1}$$

Using the commutative property of convolution, we can rewrite (1) as follows:

$$U = \begin{bmatrix} u1 \\ u2 \\ u3 \\ u4 \end{bmatrix} = \begin{bmatrix} v1 & v2 & v3 & v4 \\ v2 & v3 & v4 & v1 \\ v3 & v4 & v1 & v2 \\ v4 & v1 & v2 & v3 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \tag{2}$$

where $\{v1, v2, v3, v4\}$ are input data, $\{a, b, c, d\}$ are coefficients, and $\{u1, u2, u3, u4\}$ are output data. Observing (2), we find that different outputs in vector U can be computed using the same input data with rotated order and the same set of coefficients $\{a, b, c, d\}$. According to the DA technique, using the same set of coefficients implies that identical ROM modules are used to compute all the different outputs, and using the same input data samples with rotated order implies that we can arrange the partial products generated from the rotated combinations of the input data samples as a group and these partial products can be accessed simultaneously in accumulating all the outputs. Fig. 1 shows the proposed memory-efficient architecture for computing the vector U. We use a group ROM to store all the shared ROM content that contains only 24 words instead of 64 words needed in the conventional DA-based architecture. Meanwhile, we need an additional barrier shifter and address decoder. However, this part of the overhead is minor compared with the savings we get in the ROM size, as we analyzed in the previous works [11].

B. Advantages of the Proposed Approach

In the following, we focus on evaluating the delay time and hardware cost of the inner-product circuits realized by the proposed design approach and the conventional DA approach, respectively. For a fair comparison, we adopt the Avanti 0.35-μm CMOS data-path cell-library [15] in the performance evaluation in terms of the delay-area product shown in Fig. 2. We find that the delay-area product of the inner-product design realized by the proposed approach is much lower than that of the conventional DA design as N increases, which illustrates that the proposed approach possesses better performance than the conventional DA-based approach does.

III. Proposed Memory Efficient Design for 1-D DCT

A. Algorithm Derivation

If transform length N is prime, we can write the 1-D N-point DCT of an input sequence $\{y(n), n = 0, 1, \ldots, N-1\}$ in cyclic

convolution form by exploiting the property of I/O data permutation as

$$\begin{aligned} Y(0) &= \sum_{n=0}^{N-1} y(n) \\ Y((g^k)_N) &= [2 \cdot T((g^k)_N) + x(0)] \\ &\quad \cdot \cos\left(\frac{\pi}{2N} \cdot ((g^k)_N)\right); \quad k = 1, \ldots, N-1 \\ T((g^k)_N) &= \sum_{n=1}^{N-1} x((g^{n-k+1})_N) \\ &\quad \cdot (-1)^m \cdot \cos\left(\frac{\pi}{N} \cdot (g^{n+1})_N\right) \end{aligned} \tag{3}$$

where $(g^k)_N$ denotes the result of "g^k modulo N" for short, g is a primitive element, and the sequence $\{x(n)\}$ is defined as

$$\left\{ \begin{array}{c} x(N-1) = y(N-1) \\ x(n) = y(n) - x(n+1); \quad n = 0, \ldots, N-2 \end{array} \right\}. \tag{4}$$

By using the symmetry property of cosine kernel as

$$\begin{aligned} \cos\left(\frac{\pi}{N} \cdot (g^n)_N\right) &= -\cos\left(\frac{\pi}{N} \cdot (N - (g^n)_N)\right) \\ &= -\cos\left(\frac{\pi}{N} \cdot \left(g^{n+\frac{N-1}{2}}\right)_N\right) \end{aligned} \tag{5}$$

we can rewrite the $\mathrm{T}((g^k)_N)$ in (3) as

$$\begin{aligned} T((g^k)_N) = \sum_{n=1}^{(N-1)/2} &\left[x((g^{n-k+1})_N)\right. \\ &\cdot (-1)^m + x\left(\left(g^{(n-k+1+\frac{N-1}{2})}\right)_N\right) \\ &\left. \cdot (-1)^{m+\frac{N-1}{2}}\right] \times \cos\left(\frac{\pi}{N} \cdot (g^{n+1})_N\right); \\ &k = 1, \ldots, N-1. \end{aligned} \tag{6}$$

To describe the proposed algorithm in more detail, we can write the kernel $T((3^k)_7)$ in a design example of 1-D seven-point DCT in matrix form as

$$\begin{bmatrix} T(3) \\ T(2) \\ T(6) \\ T(4) \\ T(5) \\ T(1) \end{bmatrix} = \begin{bmatrix} -x(3) & x(2) & x(6) & -x(4) & x(5) & x(1) \\ x(1) & x(3) & x(2) & -x(6) & -x(4) & -x(5) \\ x(5) & x(1) & x(3) & -x(2) & -x(6) & -x(4) \\ x(4) & x(5) & x(1) & -x(3) & -x(2) & -x(6) \\ x(6) & x(4) & -x(5) & x(1) & x(3) & -x(2) \\ x(2) & x(6) & x(4) & x(5) & x(1) & x(3) \end{bmatrix} \times \begin{bmatrix} \cos(2a) \\ \cos(6a) \\ \cos(4a) \\ \cos(5a) \\ \cos(1a) \\ \cos(3a) \end{bmatrix} \tag{7}$$

where a denotes $(\pi/7)$. However, the input data elements of the kernel possess different signs so that it is not easy to apply the proposed memory-efficient approach directly to DCT realization. According to the symmetry property of DCT coefficients as shown in (5), we can write (7) as

$$\begin{bmatrix} T(3) \\ T(2) \\ T(6) \\ T(4) \\ T(5) \\ T(1) \end{bmatrix} = \begin{bmatrix} -x(3) & x(2) & x(6) & -x(4) & x(5) & x(1) \\ x(1) & x(3) & x(2) & -x(6) & -x(4) & -x(5) \\ x(5) & x(1) & x(3) & -x(2) & -x(6) & -x(4) \\ x(4) & x(5) & x(1) & -x(3) & -x(2) & -x(6) \\ x(6) & x(4) & -x(5) & x(1) & x(3) & -x(2) \\ x(2) & x(6) & x(4) & x(5) & x(1) & x(3) \end{bmatrix} \times \begin{bmatrix} \cos(2a) \\ \cos(6a) \\ \cos(4a) \\ -\cos(2a) \\ -\cos(6a) \\ -\cos(4a) \end{bmatrix} \tag{8}$$

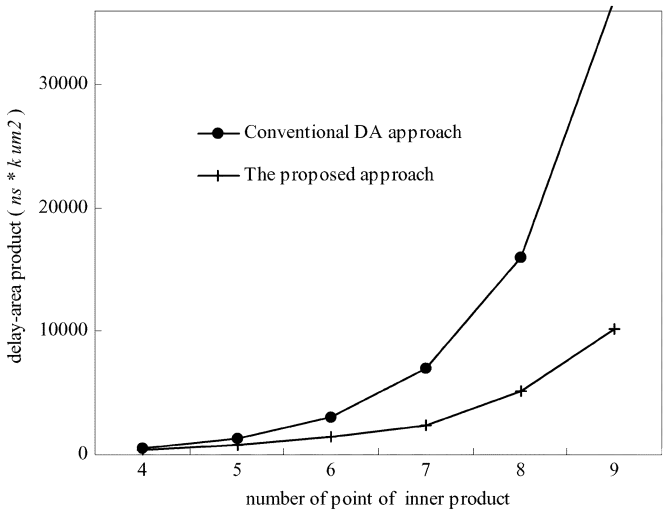


Fig. 2. Delay-area product of the designs using the proposed approach and the conventional DA approach with 16-bit data word length.

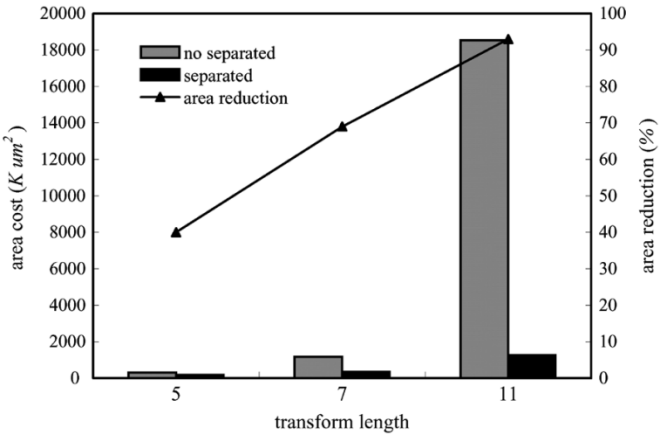


Fig. 3. Area reduction of the ROM cost whether or not applying the symmetry property of DCT coefficients.

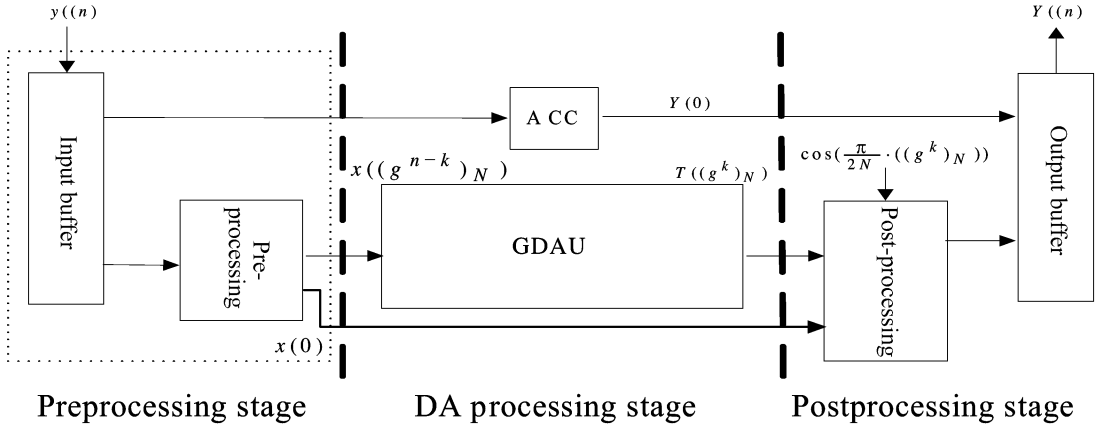
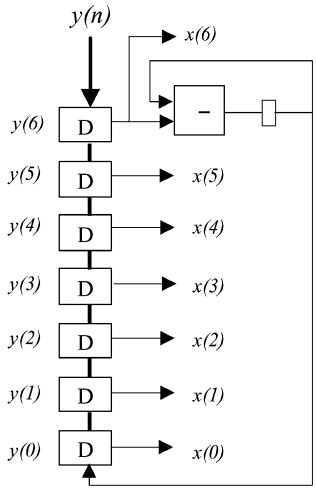


Fig. 4. Block diagram of the proposed pipeline architecture for computing the 1-D N-point DCT.



Input buffer	Cycle-based timing information for input data sampling and preprocessing													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
D	*y(0)*	*y(1)*	*y(2)*	*y(3)*	*y(4)*	*y(5)*	*y(6)*	*y(5)*	*y(4)*	*y(3)*	*y(2)*	*y(1)*	*y(0)*	*x(6)*
D		*y(0)*	*y(1)*	*y(2)*	*y(3)*	*y(4)*	*y(5)*	*y(4)*	*y(3)*	*y(2)*	*y(1)*	*y(0)*	*x(6)*	*x(5)*
D			*y(0)*	*y(1)*	*y(2)*	*y(3)*	*y(4)*	*y(3)*	*y(2)*	*y(1)*	*y(0)*	*x(6)*	*x(5)*	*x(4)*
D				*y(0)*	*y(1)*	*y(2)*	*y(3)*	*y(2)*	*y(1)*	*y(0)*	*x(6)*	*x(5)*	*x(4)*	*x(3)*
D					*y(0)*	*y(1)*	*y(2)*	*y(1)*	*y(0)*	*x(6)*	*x(5)*	*x(4)*	*x(3)*	*x(2)*
D						*y(0)*	*y(1)*	*y(0)*	*x(6)*	*x(5)*	*x(4)*	*x(3)*	*x(2)*	*x(1)*
D							*y(0)*	*x(6)*	*x(5)*	*x(4)*	*x(3)*	*x(2)*	*x(1)*	*x(0)*

Fig. 5. Design of the preprocessing stage in the 1-D seven-point DCT.

and the data elements in the matrix of (8) can be merged as

$$\begin{bmatrix} T(3) \\ T(2) \\ T(6) \\ T(4) \\ T(5) \\ T(1) \end{bmatrix} = \begin{bmatrix} x(4)-x(3) & x(2)-x(5) & x(6)-x(1) \\ x(6)+x(1) & x(4)+x(3) & x(2)+x(5) \\ x(2)+x(5) & x(6)+x(1) & x(4)+x(3) \\ x(4)+x(3) & x(2)+x(5) & x(6)+x(1) \\ x(6)-x(1) & x(4)-x(3) & x(2)-x(5) \\ x(2)-x(5) & x(6)-x(1) & x(4)-x(3) \end{bmatrix} \times \begin{bmatrix} \cos(2a) \\ \cos(6a) \\ \cos(4a) \end{bmatrix}. \quad (9)$$

To separate the even and odd outputs in (9), we can obtain two smaller, perfect cyclic convolution forms as follows:

$$\begin{bmatrix} T(2) \\ T(6) \\ T(4) \end{bmatrix} = \begin{bmatrix} x(6)+x(1) & x(4)+x(3) & x(2)+x(5) \\ x(2)+x(5) & x(6)+x(1) & x(4)+x(3) \\ x(4)+x(3) & x(2)+x(5) & x(6)+x(1) \end{bmatrix} \times \begin{bmatrix} \cos(2a) \\ \cos(6a) \\ \cos(4a) \end{bmatrix}$$

$$\begin{bmatrix} \mathrm{T}(5) \\ \mathrm{T}(1) \\ \mathrm{T}(3) \end{bmatrix} \quad (10)$$

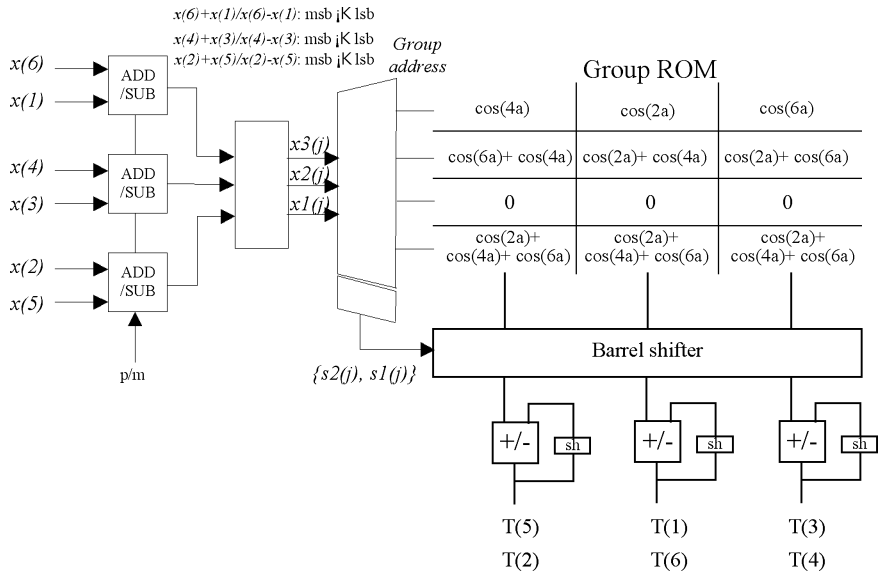


Fig. 6. Design of the DA processing stage that is used to compute the kernel of $T((3^k)_7)$ in the 1-D seven-point DCT.

$$= \begin{bmatrix} \mathrm{x}(6)-\mathrm{x}(1) & \mathrm{x}(4)-\mathrm{x}(3) & \mathrm{x}(2)-\mathrm{x}(5) \\ \mathrm{x}(2)-\mathrm{x}(5) & \mathrm{x}(6)-\mathrm{x}(1) & \mathrm{x}(4)-\mathrm{x}(3) \\ \mathrm{x}(4)-\mathrm{x}(3) & \mathrm{x}(2)-\mathrm{x}(5) & \mathrm{x}(6)-\mathrm{x}(1) \end{bmatrix} \times \begin{bmatrix} \cos(2a) \\ \cos(6a) \\ \cos(4a) \end{bmatrix}. \quad (11)$$

From (8) to (11), we find that exploiting the symmetry property of the DCT coefficient can help merging the input data elements in the DCT kernel and separating the kernel into two perfect cyclic forms, which facilitates the efficient realization of the DCT through the proposed design approach. Fig. 3 shows the area reduction of the ROM cost when applying the symmetry property of the DCT coefficients [shown in (10) and (11)] or not [shown in (7)]. We find that it is helpful in reducing the ROM size greatly when using the symmetry property of the DCT coefficients.

For facilitating the proposed memory-efficient design approach, we further formulate the $T((g^k)_N)$ specified in (6), as shown in (12) at the bottom of the page, where L denotes the data word length of the variable x, N denotes the transform length, the variable $G(x_j((g^{n-k})_N))$ denotes the jth-bit group address of the ROM access operations, and the preprocessed input sequence $\{x(n)\}$ is defined as (13) and (14), shown at the bottom of the page. The value of m is determined by

$$(g^{n+1})_N + m \cdot N = (g^{n-k+1})_N \cdot (g^k)_N \quad n, k = 1, \ldots, N-1. \quad (15)$$

TABLE I
SEED VALUE, GROUP ADDRESS, AND ROTATING FACTOR USED IN THE DESIGN OF GROUP ADDRESS DECODER OF 1-D SEVEN-POINT DCT

Grouped DA input value (Xj) {x3(j), x2(j), x1(j)}	*[1]Rotating factor (Sj) {s2(j), s1(j)}*	*Seed-value (Xj') {x3'(j), x2'(j), x1'(j)}*	*Group address (Gj) {g2(j), g1(j)}*
001	0	001	0
010	1		
100	2		
011	0	011	1
110	1		
101	2		
000	0	000	2
111	0	111	3

Note: (1) Rotating factor denotes the number of position of the output data, corresponding to the candidate of DA input value in a group, should be rotated.

$$T((g^{k+1})_N) = \left\{ \begin{array}{c} \sum_{j=1}^{L} \left(\sum_{n=1}^{(N-1)/2} G_1(x_j((g^{n-k})_N)) \cdot \cos\left(\frac{\pi}{N} \cdot (g^{n+1})_N\right) \right) \cdot 2^{-j}, \quad k = 1, \ldots, \frac{N-1}{2} \\ \sum_{j=1}^{L} \left(\sum_{n=1}^{(N-1)/2} G_2(x_j((g^{n-k})_N)) \cdot \cos\left(\frac{\pi}{N} \cdot (g^{n+1})_N\right) \right) \cdot 2^{-j}, \quad k = \frac{N-1}{2} + 1, \ldots, N-1 \end{array} \right\} \quad (12)$$

$$x((g^{n-k+1})_N) = \left\{ \begin{array}{c} x((g^{n-k+1})_N), \quad \text{if } n-k+1 \geq 0 \\ x\left(\left(g^{(N-1)+(n-k+1)}\right)_N\right), \quad \text{if } n-k+1 < 0 \end{array} \right\} \quad (13)$$

$$x\left(\left(g^{n-k+1+\frac{N-1}{2}}\right)_N\right) = \left\{ \begin{array}{c} x\left(\left(g^{n-k+1+\frac{N-1}{2}}\right)_N\right), \quad \text{if } n-k+1+\frac{N-1}{2} \geq 0 \\ x\left(\left(g^{(N-1)+(n-k+1+\frac{N-1}{2})}\right)_N\right), \quad \text{if } n-k+1+\frac{N-1}{2} < 0 \end{array} \right\}. \quad (14)$$

TABLE II
PARTIAL PRODUCTS DISTRIBUTION FOR DIFFERENT DCT OUTPUTS UNDER THE SAME INPUT VALUE

Input $(X_p)_j/(X_m)_j$	*T(2)/ T(5)*	*T(6) / T(1)*	*T(4) / T(3)*	*Group address*
000	*0*	*0*	*0*	2
001	*cos(4a)*	*cos(2a)*	*cos(6a)*	0
010	*cos(6a)*	*cos(4a)*	*cos(2a)*	0
011	*cos(6a)+ cos(4a)*	*cos(2a)+ cos(4a)*	*cos(2a)+ cos(6a)*	1
100	*cos(2a)*	*cos(6a)*	*cos(4a)*	0
101	*cos(2a)+ cos(4a)*	*cos(2a)+ cos(6a)*	*cos(6a)+ cos(4a)*	1
110	*cos(2a)+ cos(6a)*	*cos(6a)+ cos(4a)*	*cos(2a)+ cos(4a)*	1
111	*cos(2a)+ cos(4a)+ cos(6a)*	*cos(2a)+ cos(4a)+ cos(6a)*	*cos(2a)+ cos(4a)+ cos(6a)*	3

TABLE III
EIGHT-WORD ROM CONTENTS ARRANGED INTO GROUPS

Original address	Group address			
1, 2, 4	*0*	*cos(4a)*	*cos(2a)*	*cos(6a)*
3, 5, 6	*1*	*cos(6a)+ cos(4a)*	*cos(2a)+ cos(4a)*	*cos(2a)+ cos(6a)*
0	*2*	*0*	*0*	*0*
7	*3*	*cos(2a)+ cos(4a)+ cos(6a)*	*cos(2a)+ cos(4a)+ cos(6a)*	*cos(2a)+ cos(4a)+ cos(6a)*

B. Architecture Design

Fig. 4 shows the proposed pipeline architecture that realizes the 1-D N-point DCT. It consists of the preprocessing stage, the DA processing stage, and the post-processing stage. For the 1-D seven-point DCT design example, the input buffer and preprocessing in the preprocessing stage shown as Fig. 5 are designed by using the bidirectional shift registers and an accumulator, which is used to generate the data sequence $x(n)$ from input sequence $y(n)$. The detail cycle information shows that the latency consumed by input data sampling and $x(n)$ computation is 14 cycles. The DA processing stage shown as Fig. 6, named group distributed arithmetic unit (GDAU), is designed with the proposed memory-efficient approach to carry out the computation of $T((3^k)_7)$ in the design example of 1-D seven-point DCT. Due to the same content of group ROM, only one group ROM in the GDAU is required to compute the outputs of the separated cyclic operations. In Fig. 6, the combined input vector $Xj = [x3(j), x2(j), x1(j)]$ is first fed into an address decoder to determine which group it should belong to. The address decoder will compute the seed value $Xj' = [x3'(j), x2'(j), x1'(j)]$, group address $Gj = [g2(j), g1(j)]$, and the rotating factor $Sj = [s2(j), s1(j)]$ by decoding the input vector according to Table I. Table II shows the original partial product distributions for computing the outputs of DCT kernel under the same input value. From Table II, the rotation relationship between these partial products is also visible, and then the ROM content arrangement in the proposed design is shown in Table III. It is noted that we only need one small group ROM module of size $(N-1)/2 \times G_{\text{num}}$ words for computing $T((3^k)_7)$. As above, G_{num} denotes the number of groups in the group ROM modules, which is dependent on the transform length N. The post-processing stage shown in Fig. 7(a) is used to perform the post computation for GDAU outputs, including the operations of multiplying by two, added with $x(0)$ and multiplied serially by the cosine coefficients in formulation. Since the operation of multiplying by two is performed by the manner of hardwiring, it has no hardware cost required. The output buffer shown in Fig. 7(b) in the post-processing stage is used to perform the operations of preloadable shifting for serially generating the results of DCT in order.

C. Chip Design

We have verified the proposed design for 1-D seven-point DCT in VERILOG modeling. According to the synthesis result with the Avanti 0.35-μm cell library, this design consumes 7485 gates and possesses the maximum path delay of 12.1 ns. The working frequency of the chip is above 82.6 MHz. In other words, the chip can maintain the throughput rate of 18.1 M samples/s, i.e., $(82.6\ \text{MHz}/32\ \text{cycles}) * 7$ samples. Fig. 8 shows the layout view of the 1-D seven-point DCT chip fabricated using TSMC 0.35-μm CMOS 1P4M process. The core size of proposed DCT design is equal to $1734 * 1732 \mu\text{m}^2$.

IV. PERFORMANCE EVALUATION

In this section, we will illustrate the performance evaluation of the design using the proposed design approach and some existing DCT designs. The existing DCT designs used in this evaluation include memory-based systolic array design [1], direct DA design [2], OBC DA design [8], and adder-based DA design [13]. For a fair comparison, we adopt the Avanti 0.35-μm, 3.3-V CMOS cell library [15] in the performance evaluation in terms of the delay time and area cost. According to these two measures, we can evaluate these designs in terms of delay-area product with respect to different values of N. Table IV shows the comparisons of these designs. The design in [1] is a ROM-based systolic array design. It needs about N adders, N-1 16-bit flip flop, and $(N-1) \cdot 2^{(\text{L}/2)}$ words of ROM if the ROM tables in the design are partitioned once. The silicon area of this design is equal to 1237 N–1217 k$\cdot\mu\text{m}^2$. The design in [2] is the conventional ROM-based DA design; it requires about N 16-bit adders and N 16-bit flip flops used for PISO and $2^N \cdot N$ words of ROM. The silicon area of this design is equal to $13.7N + 4.75 \cdot 2^N \cdot N$ k$\cdot\mu\text{m}^2$. The design in [8] is the modified ROM-based DA design using the reduction technique of OBC, it requires about $2N$ 16-bit adders, N 16-bit flip flop, and $2^{(N-2)} \cdot N$ words of ROM. The silicon area of this design is equal to $19.6N + 4.75 \cdot 2^{(N-2)} \cdot N$ k$\cdot\mu\text{m}^2$. The design in

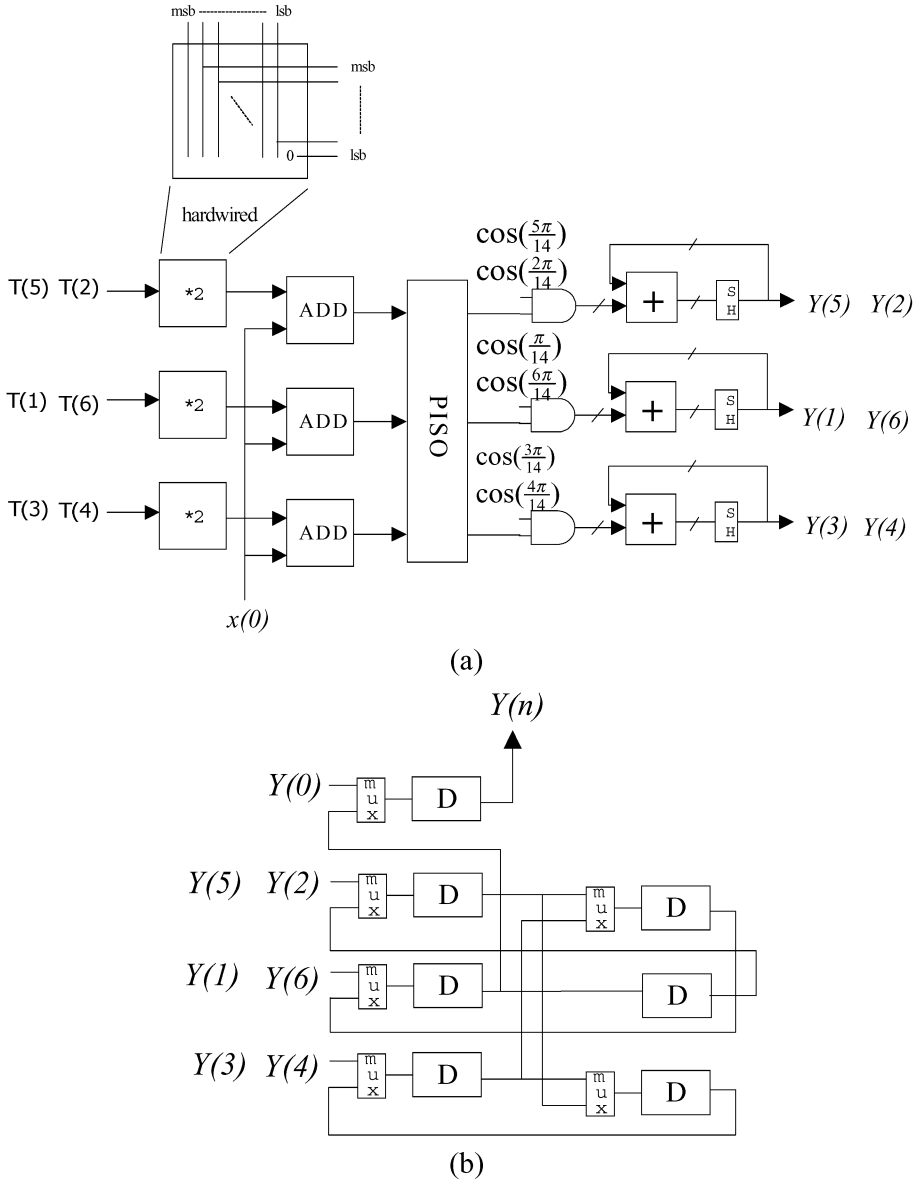


Fig. 7. Design of the post-processing stage in the 1-D seven-point DCT, including (a) the post processing and (b) the output buffer.

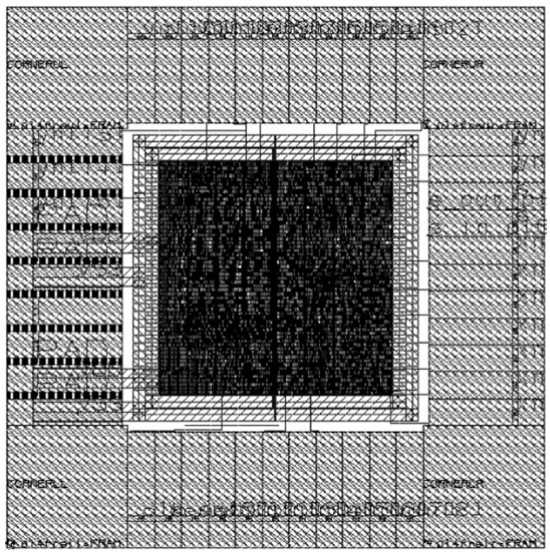


Fig. 8. Layout view of the 1-D seven-point DCT chip.

[13] is the adder-based DA design; it requires about four 16-bit multipliers, $2N + 2\mathrm{L} + 5$, 16-bit adders, $4N + 3$, 16-bit flip flops, and $4N$ words of RAM. The silicon area of this design is equal to $73N + 380$ Kμm ^{2}. Fig. 9 shows the delay-area product of the proposed design and the existing designs [1], [2], [8], [13] in realizing the 1-D DCT with various values of N. As shown in Fig. 9, in the case of 16-bit data word length, the delay-area product of the proposed design is much smaller than the ROM-based systolic array DCT design [1] and the other DA-based designs [2], [8], [13].

As for realizing the long-length DCT, with the issue of cyclic convolution partitioning, we suggest to partition the logn-length cyclic convolution to the short ones with Agarwal–Cooley algorithm [16] such that the logn-length DCT still consists of the short cyclic-convolution blocks. Thus, the proposed GDA design can be extended to realize each of the shortened cyclic convolution blocks for logn-length DCT to achieve low hardware cost and better delay-area product.

TABLE IV
COMPARISON OF THE PROPOSED DESIGN AND THE EXISTING DCT DESIGNS [1], [2], [8], [13] IN REALIZING THE 1-D N-POINT DCT IN TERMS OF DELAY AND SILICON AREA

	Cycle time *(T)*	Mul *(16-bit)*	Adder *(16-bit)*	FF *(16-bit)*	ROM *(16-bit)*	Barrel shifter *(16-bit)*	RAM *(16-bit)*	Delay*Area *(ns * Kum^2)*
Guo [1] (Memory-based systolic array)	$T=tmux+trom+tadd+tadd$		$5.9N$	$7.8(N-1)$	$1216\cdot(N-1)$		$7.4\cdot(N-1)$	$[(N-1)T/N]\ (1237N-1217)$
White [2] (directly DA)	$T = trom+ tadd$		$5.9N$	$23.4N$	$4.75\cdot2^N\cdot N$			$[(16T)/N]*(29.3N+4.75\cdot2^N\cdot N)$
Choi [8] (OBC-based DA)	$T = trom+2\ tadd$		$11.8N$	$23.4N$	$4.75\cdot2^{(N-2)}\cdot N$			$[(16T)/N]*(35.2N+4.75\cdot2^{(N-2)}\cdot N)$
Guo [13] (Adder-based DA)	$T=max\{tmul, tadd+tff\}$	$34.6*4$	$5.9(2N+37)$	$7.8(4N+3)$			$7.4(4N)$	$[(NT)/N]*(73N+380)$
The proposed design	$T= trom+tbr+tadd$		$11.8N$	$23.4(N-1)$	$4.75\cdot2^{\frac{6}{7}\cdot\frac{(N-1)}{2}}\cdot\frac{(N-1}{2})$	$12*[-0.072+0.435*(N-1)+0.053*(N-1)^2]$		$[(32T)/N]*[-28.8+39.1N+0.64N^2+[4.75\cdot2^{\frac{6}{7}\cdot\frac{(N-1)}{2}}\cdot\frac{(N-1}{2})]$

Note: (1) *tmul denotes the delay time of multiplier.*
(2) *tmux denotes the delay time of multiplexer.*
(3) *tadd denotes the delay time of adder.*
(4) *trom denotes the access time of ROM associated with N for DA-based design or associated with wordlength for memory-based systolic array design.*
(5) *tbr denotes the delay time of barrel shifter associated with N.*

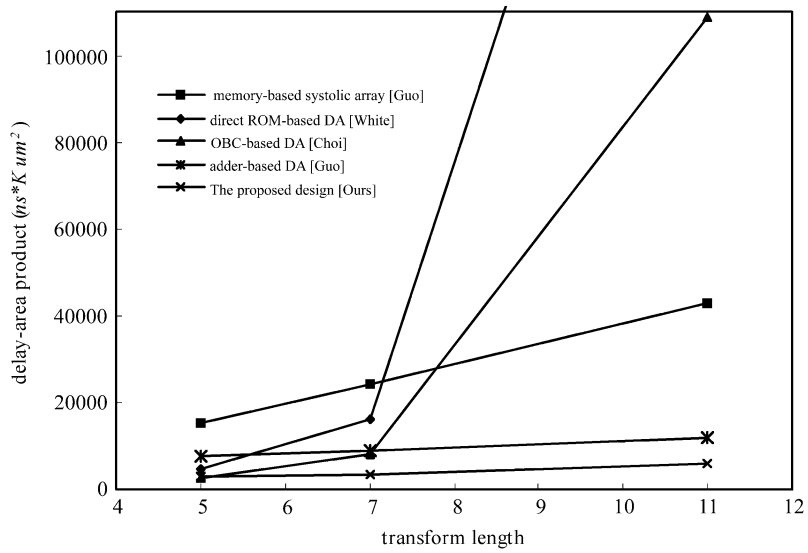


Fig. 9. Delay-area product of the proposed design and the existing DCT designs [1], [2], [8], [13] in realizing the 1-D DCT.

V. CONCLUSION

This paper presents a memory-efficient design for realizing the cyclic convolution and its application to the DCT. We combine the advantages of DA computation as well as the cyclic convolution, and exploit the symmetry property of DCT coefficients to facilitate a memory-efficient realization of 1-D N-point DCT kernel using $(N-1)/2$ adders or substractors, one small ROM module, a barrel shifter, and $((N-1/2))+1$ accumulators. To increase the ROM utilization, we rearrange the content of ROM into several groups in which all the elements in a group will be accessed simultaneously in computing all the DCT outputs. A 1-D seven-point DCT chip was designed and implemented to illustrate the efficiency associated with the proposed approach. Compared with the existing designs, the proposed design can reduce more than 57% of the delay-area product. In summary, the proposed design approach can be easily applied in the transform problems formulated into cyclic convolution.

REFERENCES

[1] J. I. Guo, C. M. Liu, and C. W. Jen, "The efficient memory-based VLSI array designs for DFT and DCT," *IEEE Trans. Circuits Syst. II., Exp. Briefs*, vol. 39, no. 10, pp. 723–733, Oct. 1992.
[2] S. A. White, "Applications of distributed arithmetic to digital sequence processing: A tutorial review," *IEEE Trans. Acoust. Speech Signal. Process.*, vol. 6, no. 1, pp. 4–19, Jan. 1989.
[3] W. P. Burleson and L. L. Scharf, "A VLSI design methodology for distributed arithmetic," *J. VLSI Signal Process.*, vol. 2, pp. 235–252, 1991.
[4] Y. H. Chan and W. C. Siu, "On the realization of discrete cosine transform using the distributed arithmetic," *IEEE Trans. Circuits Syst. I., Reg. Papers*, vol. 39, no. 9, pp. 705–712, Sep. 1992.

[5] S. Wolter, A. Scubert, H. Matz, and R. Laur, "On the comparison between architectures for the implementation of distributed arithmetic," in *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 3, May 1993, pp. 1829–1832.

[6] A. Madisetti and A. N. Wilson, "A 100 MHz 2-D $8 * 8$ DCT/IDCT processor for HDTV applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, no. 2, pp. 158–164, Apr. 1995.

[7] K. Nourji and N. Demassieux, "Optimal VLSI architecture for distributed arithmetic-based algorithm," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 2, Apr. 1994, pp. 509–512.

[8] J. P. Choi, S. C. Shin, and J. G. Chung, "Efficient ROM size reduction for distributed arithmetic," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 2000, pp. II61–II64.

[9] M. Sheu, J. Lee, J. Wang, A. Suen, and L. Liu, "A high throughput-rate architecture for $8 * 8$ 2-D DCT," in *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 3, May 1993, pp. 1587–1590.

[10] W. K. Dae *et al.*, "A compatible DCT/IDCT architecture using hardwired distributed arithmetic," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 2001, pp. II457–II460.

[11] H. C. Chen, J. I. Guo, and C. W. Jen, "A new group distributed arithmetic design for the one dimensional discrete Fourier transform," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 2002, pp. I-421–I-424.

[12] T. S. Chang, C. Chen, and C. W. Jen, "New distributed arithmetic algorithm and its application to IDCT," *Proc. Inst. Elect. Eng.*, vol. 146, no. 4, pp. 159–163, Aug. 1999.

[13] J. I. Guo, "Efficient parallel adder based design for one dimensional discrete cosine transform," *Proc. Inst. Elect. Eng.*, vol. 147, no. 5, pp. 276–282, Oct. 2000.

[14] ——, "An efficient design for one dimensional discrete Hartley transform using parallel additions," *IEEE Trans. Signal Process.*, vol. 48, no. 10, pp. 2806–2813, Oct. 2000.

[15] AVANT!: 0.35 Micron 3.3-Volt High Performance Standard Cell Library, Avant! Corporation, 1996.

[16] R. C. Agarwal and J. W. Cooley, "New algorithms for digital convolution," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-25, no. 5, pp. 392–410, Oct. 1977.