IJCAES
ISSN: 2231-4946
Volume III, Special Issue, August 2013

International Journal of Computer Applications in Engineering Sciences
Special Issue on National Conference on Information and Communication (NCIC'13)
www.caesjournals.org

DA-Based DCT with Error-Compensated Adder Tree

Chinababu Panduru[1], P. Mahesh Kumar[2]

[1,]M.Tech scholar,[2]Assistant professor, Department of ECE,
Chadalawada Ramanamma Engineering College, Tirupathi, India.
maheshpenubaku@gmail.com

ABSTRACT: Here we are operating the shifting and addition in parallel, an error-compensated adder-tree (ECAT) is proposed to deal with the truncation errors and to achieve low-error and high-throughput discrete cosine transform (DCT) design. Instead of the 12 bits used in previous works, 9-bit distributed arithmetic-precision is chosen for this work for peak-signal-to-noise-ratio (PSNR) requirements. Thus, an area-efficient DCT core is implemented to achieve 1 Gpels/s throughput rate with gate counts of 22.2 K for the PSNR requirements outlined in the previous works.

INDEX TERMS: **Distributed arithmetic (DA)-based, error-compensated adder-tree (ECAT), 2-D Discrete Cosine Transform (DCT).**

I. INTRODUCTION

Discrete cosine transform (DCT) is a widely used tool in image and video compression applications [1]. Recently, the high-throughput DCT designs have been adopted to fit the requirements of real-time applications [2].

The multiplier-based DCTs were presented and implemented in [2] and [3].To reduce area, ROM-based distributed arithmetic (DA) was applied in DCT cores [4]–[6]. Uramoto*etal.* [4] implemented the DA-based multipliers using ROMs to produce partial products together with adders that accumulated these partial products. In this way, instead of multipliers, the DA-based ROM can be applied in a DCT core design to reduce the area required. In addition, the symmetrical properties of the DCT transform and parallel DA architecture can be used in reducing the ROM size in [5] and [6], respectively. The high-throughput shift-adder-tree (SAT) and adder-tree (AT), those unroll the number of shifting and addition words in parallel for DA- based computation, were introduced in [10]. However, a large truncation error occurred. In order to reduce the truncation error effect, several error compensation bias methods have been presented [13] based on statistical analysis of the relationship between partial products and multiplier-multiplicand.

This brief addresses a DA-based DCT core with an error-compensated adder-tree (ECAT). The proposed ECAT operates shifting and addition in parallel by unrolling all the words required to be computed Therefore, the hardware cost is reduced, and the speed is improved using the proposed ECAT.

II. MATHEMATICAL DERIVATION OF DISTRIBUTED ARITHMETIC

The inner product is an important tool in digital signal processing applications. It can be written as follows:

$$Y = A^T X = \sum_{i=1}^{L} AiXi \quad \text{..... (1)}$$

Where A_i, X_i, and L are i^{th} fixed coefficient, i^{th} input data, and number of inputs, respectively. Assume that coefficient A_i is Q-bit two's complement binary fraction number. Equation (1) can be expressed as follows:

The inner product computation in (1) can be implemented by using shifting and adders instead of multipliers. Therefore, low hardware cost can be achieved by using DA-based architecture.

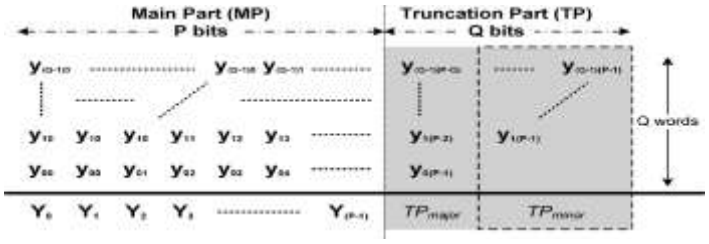


Fig. 1. Q *P*-bit words shifting and addition operations in parallel.

III. ECAT ARCHITECTURE

From (2), the shifting and addition computation can be written as follows:

$$Y = \sum_{j=0}^{Q-1} y_j \cdot 2^{-j} \quad \ldots\ldots\ldots\ldots\ldots (2)$$

In general, the shifting and addition computation uses a shift-and-add operator [7] in VLSI implementation in order to reduce hardware cost. However, when the number of the shifting and addition words increases, the computation time will also increase. Therefore, the shift-adder-tree (SAT) presented in [10] operates shifting and addition in parallel by unrolling all the words needed to be computed for high-speed applications. However, a large truncation error occurs in SAT, and ECAT architecture is proposed in this brief to compensate for the truncation error in high-speed applications.

In Fig. 1, the Q P-bit words operate the shifting and addition in parallel by unrolling all computations. Furthermore, the operation in Fig. 1 can be divided into two parts: the main part (MP) that includes *P* most significant bits (MSBs) and the truncation part (TP) that has *Q* least significant bits (LSBs). Then, the shifting and addition output can be expressed as follows:

$$Y = MP + TP.2^{-(P-2)} \quad \ldots\ldots\ldots\ldots\ldots\ldots (3)$$

The output *Y* will obtain the *P*-bit MSBs using a rounding operation called post truncation (Post-T), which is used for high-accuracy applications. However, hardware cost increases in the VLSI design. In general, the TP is usually truncated to reduce hardware costs in parallel shifting and addition operations, known as the direct truncation (Direct-T) method. Thus, a large truncation error occurs due to the neglecting of carry propagation from the TP to MP. In order to alleviate the truncation error effect, several error compensation bias methods have been presented [10]. All previous works were only applied in the design of a fixed-width multiplier Note that the addition elements Y_{qp} in the TP in Fig. 1 (where $1 \le q \le (Q-1)$ and $(P-q-1) \le p \le (P-1)$) are independent from each other. Therefore, the previous compensation method cannot be applied in this work, and the proposed ECAT is explained as follows.

A. Proposed Error-Compensated Scheme

From Fig. 1, (4) can be approximated as

$$Y \approx MP + \sigma \cdot 2^{-(P-2)} \quad \ldots\ldots\ldots\ldots\ldots (4)$$

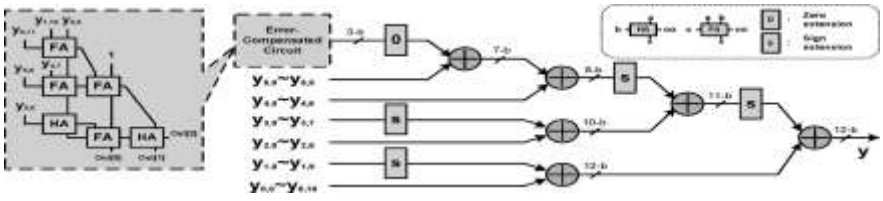


Fig 2. Proposed ECAT architecture of shifting and addition operators for the (P, Q) = (12, 6) example. where σ is the compensated bias from the TP to the MP as listed in (6) where Round( ) is rounded to the nearest integer.

$$\sigma = \text{Round}\,(TP_{major} + TP_{minor}) \ldots\ldots\ldots (5)$$

$$\mathrm{TP}_{\mathrm{major}} = \frac{1}{2}\sum_{j=0}^{Q-1} y_{j(P-1-j)}$$

$$\mathrm{TP}_{\mathrm{minor}} = \frac{1}{4}\left(y_{1(P-1)} + \cdots + y_{(Q-1)(P-Q+1)}\right) + \frac{1}{8}\left(y_{2(P-1)} + \cdots + y_{(Q-1)(P-Q+2)}\right) + \cdots + \left(\frac{1}{2}\right)^{Q} y_{(Q-1)(P-1)} \quad \text{.. (6)}$$

$Q = 4k + 2, 4k+3$ ($k \geq 1$)

$\sigma = k + \mathrm{Round}(\ \mathrm{TP}_{\mathrm{major}})$ ..... (7)

B. Performance Simulation for an Error-Compensated Circuit

In this subsection, comparisons of the absolute average error $\mathcal{E}$, the maximum error $\mathcal{E}_{\max}$, and the mean square error $\mathcal{E}_{\mathrm{mse}}$ for the proposed error-compensated circuit with Direct-T and Post-T are listed in Table I.
The $\mathcal{E}$, $\mathcal{E}_{\max}$ and $\mathcal{E}_{\mathrm{mse}}$ are defined as follows:

$\mathcal{E} = \mathrm{Avg}\ \{|\mathrm{TP} - \sigma|\}$

$\mathcal{E}_{\max} = \max\ \{|\ \mathrm{TP}\ \_\ \sigma|\}$

$\mathcal{E}_{\mathrm{mse}} = \mathrm{Avg}\ \{\ (\mathrm{TP} - \sigma)^2\ \}$ ..... (8)

where Avg{ } is the average operator.

The internal word-length usually uses 12 bits in a DCT design. Consequently, word length P=12 is chosen together with different Q values of 3, 6, 9, and 12, which are listed in Table I. The Post-T method provides the most accurate values for fixed-width computation nowadays. In addition, the Direct-T method has the largest inaccuracies of the errors shown in Table I for low-cost hardware design. The proposed ECAT is more accurate than Direct-T and is close to the performance of the Post-T method using a compensated circuit. Because the truncation part TP_{minor} is estimated using statistical analysis, the magnitude of errors also increases as the number of shift-and-add words increases.

C. Proposed ECAT Architecture

The proposed ECAT architecture is illustrated in Fig. 2 for (P, Q) = (12, 6) (*case 3*), where block FA indicates a full-adder cell with three inputs (a, b, and c) and two outputs, a sum (s) and a carry-out (co). Also, block HA indicates half-adder cell with two inputs (a and b) and two outputs, a sum (s) and a carry-out (co). The comparisons of area, delay, area-delay product, and accuracy for the proposed ECAT with other architectures are listed in Table II. The area and delay are synthesized using a Synopsys Design Compiler with the Artisan TSMC 0.18μm Standard cell library.

TABLE II
Comparisons of the Proposed ecat with other Architectures for a Six 8-bit words example

	Shift-add-add	SAT	Proposed ECAT
Area (gares)	236	406	463
Delay (ns)	10.8	3.72	3.89
Area × delay	100%	59.3%	70.7%
$\mathcal{E}_{\mathrm{mse}}$	0.326	6.761	0.218

The proposed ECAT has the highest accuracy with a moderate area delay product. The shift-and-add [7] method has the smallest area, but the overall computation time is equal to 10.8(=1.8× 6) ns that is the longest. Similarly, the SAT [10], which truncates the TP and computes in parallel, takes 3.72 ns to complete the computation and uses 406 gates, which is the best area-delay product performance. However, for system accuracy, the SAT is the worst option shown in Table II. Therefore, the ECAT is suitable for high-speed and low-error applications.

IV. PROPOSED 8 × 8 2-D DCT CORE DESIGN

The 1-D DCT employs the DA-based architecture and the proposed ECAT to achieve a high-speed, small area, and low-error design. The 1-D 8-point DCT can be expressed as follows:

$$Z_n = \frac{1}{2} k_n \sum_{m=0}^{7} x_m \times \cos\left(\frac{(2m+1)n\pi}{16}\right) \quad \ldots(9)$$

where x_m denotes the input data; z_n denotes the transform output; $0 \leq n \leq 7$; $k_n = 1/\sqrt{2}$ for n=0; and $k_n = 1$ for other n values. By neglecting the scaling factor 1/2, the 1-D 8-point DCT in (18) can be divided into even and odd parts: Z_e and Z_o as listed in (19) and (20), respectively

$$\mathbf{Z}_e = \begin{bmatrix} Z_0 \\ Z_2 \\ Z_4 \\ Z_6 \end{bmatrix} = \begin{bmatrix} c_4 & c_4 & c_4 & c_4 \\ c_2 & c_6 & -c_6 & -c_2 \\ c_4 & -c_4 & -c_4 & c_4 \\ c_6 & -c_2 & c_2 & -c_6 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \mathbf{C}_e \cdot \mathbf{a}$$

$$\mathbf{Z}_o = \begin{bmatrix} Z_1 \\ Z_3 \\ Z_5 \\ Z_7 \end{bmatrix} = \begin{bmatrix} c_1 & c_3 & c_5 & c_7 \\ c_3 & -c_7 & -c_1 & -c_5 \\ c_5 & -c_1 & c_7 & c_3 \\ c_7 & -c_5 & c_3 & -c_1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \mathbf{C}_o \cdot \mathbf{b} \quad \ldots(10)$$

Where $C_i = \text{COS}(i\pi/16)$. Moreover, the even part Z_e can be further decomposed into even and odd parts: Z_{ee} and Z_{eo}.

$$\mathbf{Z}_{ee} = \begin{bmatrix} Z_0 \\ Z_4 \end{bmatrix} = \begin{bmatrix} c_4 & c_4 \\ c_4 & -c_4 \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \end{bmatrix} = \mathbf{C}_{ee} \cdot \mathbf{A} \quad \ldots(11)$$

$$\mathbf{Z}_{eo} = \begin{bmatrix} Z_2 \\ Z_6 \end{bmatrix} = \begin{bmatrix} c_2 & c_6 \\ c_6 & -c_2 \end{bmatrix} \begin{bmatrix} B_0 \\ B_1 \end{bmatrix} = \mathbf{C}_{eo} \cdot \mathbf{B} \quad \ldots(12)$$

For the DA-based computation, the coefficient matrix C_o, C_{ee}, and C_{eo}, are expressed as 9-bit binary fraction numbers. In Table III, using given input data A_o and A_1, the transform output Z_{ee} needs only one adder to compute (A_o+A_1) and two separated ECATs to obtain the results of Z_o and Z_4. Similarly, the other transform outputs Z_{eo} and Z_o can be implemented in DA-based forms using 10(=1+9) adders and corresponding ECATs. Consequently, from the (11)–(13), the proposed 1-D 8-point DCT architecture can be constructed as illustrated in Fig. 3 using a DA-Butterfly-Matrix, that includes two DA even processing elements (DAEs), a DA odd processing element (DAO) and 12 adders/subtractors, and 8 ECATs (one ECAT for each transform output Z_n). The eight separated ECATs work simultaneously, enabling high-speed applications to be achieved. After the data output from the DA-Butterfly-Matrix is completed, the transform output Z will be completed during one clock cycle by the proposed ECATs. In contrast, the traditional shift-and-add architecture requires Q clock cycles to complete the transform output Z if the DA-precision is Q bits.

TABLE III

9-Bit DA-Based Coefficient Matrix C_{ee}

Z_0		Z_4	
Weight	Value	Weight	value
-2^0	0	-2^0	A_1

-2^1	A_0+A_1	-2^1	A_0
-2^2	0	-2^2	A_1
-2^3	A_0+A_1	-2^3	A_0
-2^4	A_0+A_1	-2^4	A_0
-2^5	0	-2^5	A_1
-2^6	A_0+A_1	-2^6	A_0
-2^7	0	-2^7	A_1
-2^8	A_0+A_1	-2^8	A_0+A_1

With high-speed considerations in mind, the proposed 2-D DCT is designed using two 1-D DCT cores and one transpose buffer. For accuracy, the DA-precision and transpose buffer word lengths are chosen to be 9 bits and 12 bits, respectively, meaning that the system can meet the PSNR requirements outlined in previous works. Moreover, the 2-D DCT core accepts 9-bit image input and 12-bit output precision.

For the proposed 2-D DCT, the Synopsys Design Compiler was applied to synthesize the RTL design of the proposed core, and the Cadence SoC Encounter was adopted for placement and routing (P&R). Implemented in a 1.8-V TSMC 0.18-×m 1P6M CMOS process, the proposed 8×8 2-D DCT core has a latency of 10 clock cycles and is operated at 125 MHz. As a result of the 8 parallel outputs, the proposed 2-D DCT core can achieve a throughput rate of 1 Gpixels per second (= 8×125 MHz), meeting the 1080 p (1920×1080 × 60 pixels/s) high definition television (HDTV) specifications for 200 MHz based on low power operations. The core layout and simulated characteristics are shown in Fig. 4.

Table IV compares the proposed 8 × 8 2-D DCT core with previous 2-D DCT cores. In [3], a multiplier-based DCT core based on pipeline radix-4^2 single delay feedback path($R4^2$ SDF) architecture to achieve high-speed design. The ROM-based DCT core is presented in [4] to reduce hardware cost. In [7], a NEDA architecture is presented by using adders to reduce the chip area of DCT core. Nevertheless, a speed limitation for shift-and-add is in NEDA design. In [8] and [9], the SAT and AT architectures for DA-based DCTs improve the throughput rate of the NEDA method. However, DA-precision must be chosen as 13 bits to meet the system accuracy with more area overhead. The proposed DCT core uses low-error ECAT to achieve a high-speed design, and the DA-precision can be chosen as 9 bits to meet the PSNR requirements for reducing hardware costs. The proposed DCT core has the highest hardware efficiency, defined as follows

Hardware Efficiency (10^3 pels/s) = (Throughput rate) / (Gate counts)

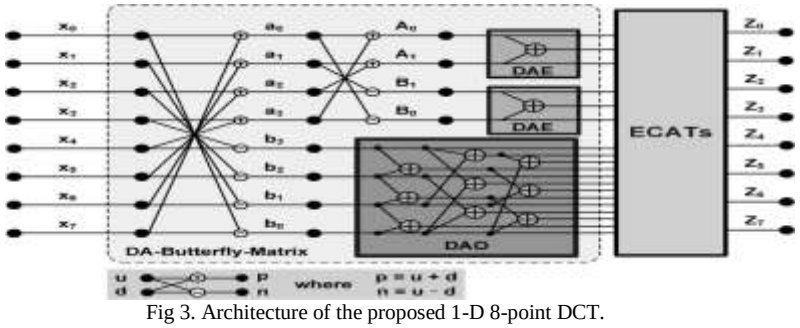
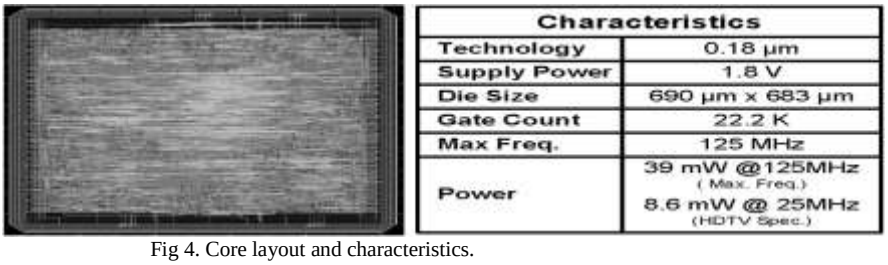


Fig 3. Architecture of the proposed 1-D 8-point DCT.



Characteristics	
Technology	0.18 µm
Supply Power	1.8 V
Die Size	690 µm x 683 µm
Gate Count	22.2 K
Max Freq.	125 MHz
Power	39 mW @125MHz (Max. Freq.) 8.6 mW @ 25MHz (HDTV Spec.)

Fig 4. Core layout and characteristics.

VI. CONCLUSION

In this brief, a high-speed and low-error 8 × 8 2-D DCT design with ECAT is proposed to improve the throughput rate significantly up to about 13 folds at high compression rates by operating the shifting and addition in parallel. Furthermore, the proposed error-compensated circuit alleviates the truncation error in ECAT. In this way, the DA-precision can be chosen as 9 bits instead of 12 bits so as to meet the PSNR requirements. Thus, the proposed DCT core has the highest hardware efficiency than those in previous works for the same PSNR requirements. Finally, an area-efficient 2-D DCT core is implemented using a TSMC 0.18-_m process, and the maximum throughput rate is 1 Gpels/s. In summary, the proposed architecture is suitable for high compression rate applications in VLSI designs.

ACKNOWLEDGEMENT

P. CHINABABU would like to thank to my guide **MR. P. MAHESH KUMAR, Assistant professor of ECE Department, CREC, TIRUPATHI.** Who had been guiding throughout the project and supporting me in giving technical ideas about the paper and motivating me to complete the work effectively and successfully.

REFERENCES

[1] Y.Wang, J. Ostermann, and Y. Zhang, *Video Processing and Communications*, 1st ed. Englewood Cliffs, NJ: Prentice-Hall, 2002.
[2] C. T. Lin, Y. C. Yu, and L. D. Van, "Cost-effective triple-mode reconfigurable pipeline FFT/IFFT/2-D DCT processor," *IEEE Trans. Very Large Scale Integer. Syst.*, vol. 16, no. 8, pp. 1058–1071, Aug. 2008.
[3] S. Yu and E. E. S. , Jr., "DCT implementation with distributed arithmetic," *IEEE Trans. Compute.*, vol. 50, no. 9, pp. 985–991, Sep. 2001.
[4] P. K. Meher, "Unified systolic-like architecture for DCT and DST using distributed arithmetic," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 12, pp. 2656–2663, Dec. 2006.
[5] A. M. Shams, A. Chidanandan, W. Pan, and M. A. Bayoumi, "NEDA: A low-power high-performance DCT architecture," *IEEE Trans. Signal Process.*, vol. 54, no. 3, pp. 955–964, Mar. 2006.
[6] M. R. M. Rizk and M. Ammar, "Low power small area high performance 2D-DCT architecture," in *Proc. Int. Design Test Workshop*, 2007, pp. 120–125.
[7] Y. Chen, X. Cao, Q. Xie, and C. Peng, "An area efficient high performance DCT distributed architecture for video compression," in *Proc. Int. Conf. Adv. Comm. Technol.*, 2007, pp. 238–241.
[8] C. Peng, X. Cao, D. Yu, and X. Zhang, "A 250 MHz optimized distributed architecture of 2D 8□8 DCT," in *Proc. Int. Conf. ASIC*, 2007, pp. 189–192.
[9] K. J. Cho, K. C. Lee, J. G. Chung, and K. K. Parhi, "Design of low-error fixed-width modified booth multiplier," *IEEE Trans. Very Large Scale Integer. (VLSI) Syst.*, vol. 12, no. 5, pp. 522–531, May 2004.
[10] L. D. Van and C. C. Yang, "Generalized low-error area-efficient fixed width multipliers," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 8, pp. 1608–1619, Aug. 2005.
[11] C. C. Sun, P. Donner, and J. Goetz, "Low-complexity multi-purpose IP core for quantized discrete cosine and integer transform," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2009, pp. 3014–3017.
[12] A. Tumeo, M. Monchiero, G. Palermo, F. Ferrandi, and D. Sciuto, "A pipelined fast 2D-DCT accelerator for FPGA-based SoCs," in *Proc. IEEE Compute. Soc. Annu. Symp. VLSI*, 2007, pp. 331–336.