VLSI Computational Architectures for the Arithmetic Cosine Transform

Nilanka Rajapaksha, *Student Member, IEEE,* Arjuna Madanayake, *Member, IEEE,* Renato J. Cintra, *Senior Member, IEEE,* Jithra Adikari, *Member, IEEE,* and Vassil S. Dimitrov

Abstract—The discrete cosine transform (DCT) is a widely-used and important signal processing tool employed in a plethora of applications. Typical fast algorithms for nearly-exact computation of DCT require floating point arithmetic, are multiplier intensive, and accumulate round-off errors. Recently proposed fast algorithm arithmetic cosine transform (ACT) calculates the DCT exactly using only additions and integer constant multiplications, with very low area complexity, for null mean input sequences. The ACT can also be computed non-exactly for any input sequence, with low area complexity and low power consumption, utilizing the novel architecture described. However, as a trade-off, the ACT algorithm requires 10 non-uniformly sampled data points to calculate the 8-point DCT. This requirement can easily be satisfied for applications dealing with spatial signals such as image sensors and biomedical sensor arrays, by placing sensor elements in a non-uniform grid. In this work, a hardware architecture for the computation of the null mean ACT is proposed, followed by a novel architectures that extend the ACT for non-null mean signals. All circuits are physically implemented and tested using the Xilinx XC6VLX240T FPGA device and synthesized for 45 nm TSMC standard-cell library for performance assessment.

Index Terms—Discrete cosine transform, Arithmetic cosine transform, fast algorithms, VLSI

1 Introduction

The discrete cosine transform (DCT) was first proposed by Ahmed *et al.* in 1974 and published in *IEEE Transactions on Computers* [1]. It has since attracted much attention in the computer engineering community [2], [3], [4], [5]. In particular, the 8-point DCT and its variants, in the form of fast algorithms, has been widely adopted in several image and video coding standards [6] such as JPEG, MPEG-1/2, and H.261-5 [7]. Some applications which use image and video compression include automatic surveillance [8], geospatial remote sensing [9], traffic cameras [10], homeland security [11], satellite based imaging [12], unmanned aerial vehicles [13], automotive [14], multimedia wireless sensor networks [15], the solution of partial differential equations [16] etc.

A particular class of fast algorithms is constituted by the arithmetic transforms. An arithmetic transform is an algorithm for low-complexity computation of a given trigonometric transform, based on number-theoretical results. A prominent example is the arithmetic Fourier transform (AFT) proposed by Reed *et al.* [17], [18]. The AFT allows multiplication-free calculation of Fourier coefficients using number-theoretic methods and non-uniformly sampled inputs. A feature of the AFT is its suitability for parallel implementation [17], [18].

Recently, an arithmetic transform method for the computation of the DCT, called the arithmetic cosine transform (ACT) was proposed in [19]. The ACT can provide a multiplication-free framework and leads to the *exact computation* of the DCT—provided that the input signal has null-mean and is non-uniformly sampled [19]. The computational gains of the ACT are only possible when its prescribed non-uniformly sampled data is available.

Classically the required non-uniform samples are derived by means of interpolation over uniformly sampled data [19]. Such interpolation implies a computational overhead. Another aspect of the ACT is that, for arbitrary input signal, it requires the computation of the input signal mean value [19]. Usually, the mean value is computed from uniformly sampled data [19]. In fact, this dependence on uniformly sampled data has been precluding the implementation of the ACT based exclusively on non-uniformly sampled data.

On the other hand, the requirement for non-uniform samples can be satisfied when spatial input signals are considered. In spatial signal processing, non-uniformly sampled signals can be directly obtained, without interpolation using a non-uniform placement of sensors [20], [7]. This motivates the search for architectures which could solely rely on non-uniformly sampled inputs.

In this paper we address two main problems: (i) the proposition of a method to obtain the mean value of a given input signal from its non-uniform samples as prescribed by the ACT and (ii) the introduction of efficient architectures for calculation of the 8-point DCT

- *N. Rajapaksha and A. Madanayake are with the Department of Electrical and Computer Engineering, The University of Akron, Akron, OH, USA. E-mail: {ntr3,arjuna}@uakron.edu*
- *R. J. Cintra is with the Signal Processing Group, Departamento de Estatística, Universidade Federal de Pernambuco, Recife, PE, Brazil. E-mail: rjdsc@ieee.org*
- *J. Adikari is with the Elliptic Technologies Inc., Ottawa, ON, Canada. E-mail: jithra.adikari@gmail.com*
- *A. Madanayake and V. S. Dimitrov are with the Department of Electrical and Computer Engineering, University of Calgary, Calgary, AB, Canada. E-mail: vdvsd103@gmail.com*

based on the ACT, operating on non-uniformly sampled data *only*. This leads to designs with low computational complexity. Having ACT architectures that compute 1-D DCT can be utilized as a building block to implement such 2-D DCT architectures that take inputs from sensors placed on a non uniform grid.

Two architectures based on the ACT are sought, being referred as Architectures I and II. Architecture I provides the hardware implementation of the ACT algorithm proposed in [19], and calculates the DCT with exact precision for null mean 8-point sequences. The proposed Architecture I is designed to require only additions and multiplications by integers. Thus, no source of intrinsic computation error is present, such as rounding-off and truncation. Therefore, area consuming hardware multipliers are not necessary. We propose Architecture II that implements the novel modified ACT algorithm for DCT calculation of arbitrary, non-null-mean input signals, using 11 hardware multiplications. Both architectures require only non-uniformly sampled inputs.

This paper unfolds as follows. In Section 2, the fundamental mathematical operations of the ACT are briefly described. Section 3 details how to compute the mean value from non-uniformly sampled data and provides a matrix formalism for the 8-point ACT. In Section 4 the proposed architectures are detailed. Section 5 brings the implementation results as well as comparisons with competing structures. Conclusions and final remarks are furnished in Section 6.

2 The Arithmetic Cosine Transform

The usual input sequence to the DCT can be considered as uniform samples of a continuous input signal $v(t)$. This results in an N-point column vector $\mathbf{v} = \{v_n\}_{n=0}^{N-1}$ which has its DCT denoted by the N-point column vector $\mathbf{V} = \{V_k\}_{k=0}^{N-1}$. To calculate $\mathbf{V}$, the ACT algorithm requires non-uniformly sampled points of the continuous input signal $v(t)$ [19]. These points are given by

$$r = \frac{2mN}{k} - \frac{1}{2},$$

where $k = 1, 2, \ldots, N-1$, and $m = 0, 1, \ldots, k-1$ [19]. We can define the set R as:

$$R = \{\text{Set of all values of } r\} \tag{1}$$

It is important to notice that the values of r are not necessarily integer. In fact, they are expected to be fractional.

If the signal of interest has zero mean, then the ACT algorithm can be used to calculate the DCT coefficients as follows. First, let the ACT averages S_k, $k = 1, 2, \ldots, N-1$, of the non-uniform sampled inputs be defined as [19]:

$$S_k \triangleq \frac{1}{k} \sum_{m=0}^{k-1} v_{2m\frac{N}{k}-\frac{1}{2}}, \quad k = 1, 2, \ldots N-1. \tag{2}$$

The ACT averages can be employed to computed DCT coefficients according to [19]:

$$V_k = \sqrt{\frac{N}{2}} \sum_{l=1}^{\lfloor \frac{N-1}{k} \rfloor} \mu(l) \cdot S_{kl}, \quad k = 1, 2, \ldots, N-1, \tag{3}$$

where $\mu(\cdot)$ is the Möbius function [17], [18], [19]. The derivation of the ACT [19] utilizes the Möbius inversion formula. Because the Möbius function values are limited to $\{-1, 0, +1\}$, (3) results in no additional multiplicative complexity.

In practice, input sequences are not always null mean, therefore a correction term is necessary to (3). In [19] an expression suitable for the non-null mean signals is given as:

$$V_k = \sqrt{\frac{N}{2}} \sum_{l=1}^{\lfloor \frac{N-1}{k} \rfloor} \mu(l) \cdot S_{kl} - \sqrt{\frac{N}{2}} \bar{v} \cdot \mathrm{M}\left(\left\lfloor \frac{N-1}{k} \right\rfloor \right), \tag{4}$$

where $\mathrm{M}(n) \triangleq \sum_{m=1}^{n} \mu(m)$ is the Mertens function [19] and $\bar{v}$ is the mean value of the uniformly sampled input sequence.

3 Proposed Algorithm with Only Non-uniformly Sampled Inputs

3.1 Mean Value Calculation

Although (4) leads to the DCT coefficients of non-null mean input signals, it requires the knowledge of quantity $\bar{v}$, which could be calculated straightforwardly from the N uniform samples in $\mathbf{v}$. Since uniform samples are not available, $\bar{v}$ should be directly calculated from non-uniform samples.

The non-uniform samples are related to the uniform samples according to the interpolation scheme given by [19]:

$$v_r = \sum_{n=0}^{N-1} w_n(r) \cdot v_n, \quad r \in R, \tag{5}$$

where $w_n(r)$ is the interpolation weight function expressed by

$$w_n(r) = \frac{1}{2N} \Bigg[\mathrm{D}_{N-1}\left(\frac{\pi}{N}(n+r+1)\right) + \mathrm{D}_{N-1}\left(\frac{\pi}{N}(n-r)\right) \Bigg], \quad n = 0, 1, \ldots, N-1,$$

and

$$\mathrm{D}_N(x) = \frac{\sin((N+1/2)x)}{\sin(x/2)}$$

denotes the Dirichlet kernel [21, p. 312]. Here, the set R is defined in (1). More compactly, (5) can be put in matrix form. Indeed, we can write $\mathbf{v_r} = \mathbf{W} \cdot \mathbf{v}$, where $\mathbf{v_r} = [v_r]_{r \in R}$ is a column vector containing the required non-uniform samples, and $\mathbf{W} = \{w_n(r)\}$, $n = 0, 1, \ldots, N-1$, $r \in R$, is the implied interpolation matrix.

For the particular case of the 8-point ACT, the following 10 non-uniform sampling instants are required [19]:

$$r \in R = \left\{ -\frac{1}{2}, \frac{25}{14}, \frac{13}{6}, \frac{27}{10}, \frac{7}{2}, \frac{57}{14}, \frac{29}{6}, \frac{59}{10}, \frac{89}{14}, \frac{15}{2} \right\}. \quad (6)$$

Moreover, matrix $\mathbf{W}$ is found to possess full column rank. Thus, its Moore-Penrose pseudo-inverse $\mathbf{W}^+$ is the left inverse of $\mathbf{W}$ [22, p. 93]. Therefore, we obtain

$$\mathbf{v} = \mathbf{W}^+ \cdot \mathbf{v_r}.$$

Consequently, the mean value of $\mathbf{v}$ can be determined exclusively from the non-uniform samples, according to:

$$\bar{v} = \frac{1}{8}\mathbf{w} \cdot \mathbf{v_r}, \quad (7)$$

where $\mathbf{w}$ is the 8-point vector of the sums of each column of $\mathbf{W}^+$. Scaled vector $\mathbf{w}/8$ has constant elements given by:

$$\frac{1}{8}\mathbf{w}^\top = \begin{bmatrix} 0.131763492716950 \\ 0.498388117552161 \\ -0.313306526814540 \\ 0.018837637958148 \\ 0.389746948996966 \\ -0.178465262210960 \\ 0.166302458810496 \\ 0.269801852271683 \\ -0.131541981375149 \\ 0.148473262094246 \end{bmatrix},$$

were the superscript $^\top$ denotes the transposition operation.

3.2 Matrix Factorization of ACT

In view of (2) and (7), (4) can be interpreted as the sought relation between $\mathbf{V}$ and $\mathbf{v_r}$. Thus, we can consider a transformation matrix $\mathbf{T}$ relating these two vectors. Notice that $\mathbf{T}$ is not a square matrix. Since $k = 1, 2, \ldots, N-1$, the size of $\mathbf{T}$ is $(N-1) \times |R|$, where $|R|$ is the number of elements of R. This transformation matrix returns all the DCT components, except the zeroth one, according to:

$$\begin{bmatrix} V_1 & V_2 & \cdots & V_{N-1} \end{bmatrix}^\top = \mathbf{T} \cdot \mathbf{v_r}.$$

Notice that $V_0 = \sqrt{N} \cdot \bar{v}$.

For $N = 8$, matrix $\mathbf{T}$ has size 7×10 and admits the following matrix factorization:

$$\mathbf{T} = 2 \cdot \mathbf{Mo} \cdot \mathbf{D_1} \cdot \mathbf{S} + \mathbf{Me} \cdot \mathbf{W}^+, \quad (8)$$

where

$$\mathbf{Mo} = \begin{bmatrix} 1 & -1 & -1 & 0 & -1 & 1 & -1 \\ 0 & 1 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{D_1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{6} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{7} \end{bmatrix},$$

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 \\ 1 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 1 \\ 1 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 2 & 0 \end{bmatrix},$$

and $\mathbf{Me}$ is the implied matrix by the Mertens function in (4). This last matrix is furnished by

$$\mathbf{Me} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{4} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{4} \end{bmatrix} \cdot \mathbf{1_7},$$

where $\mathbf{1_7}$ is the 7×7 matrix of ones.

In (8), matrix $\mathbf{D_1}$ and $\mathbf{S}$ are related to (2). Matrix $\mathbf{Mo}$ contains the values of the Möbius functions as required in (3). The second term in the right-hand side of (8) accounts for the Mertens functions and the mean value calculation as required in (4) and (7).

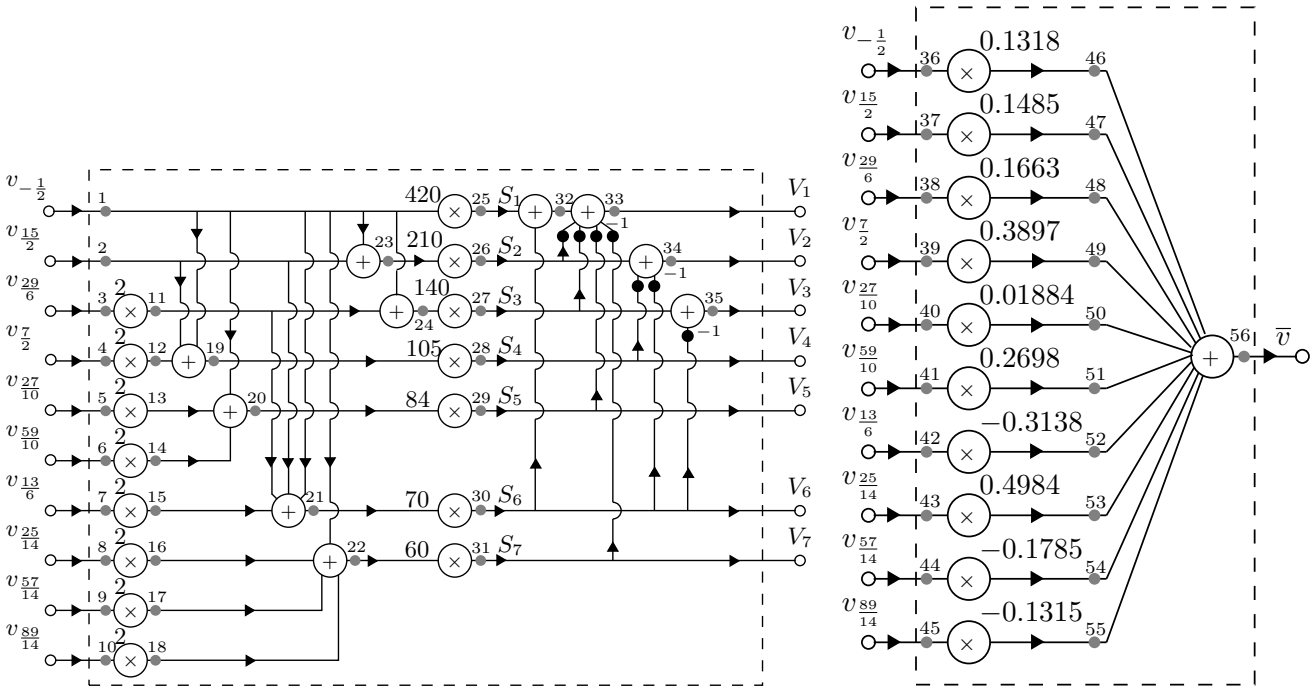
4 VLSI Architectures

In this section, above discussed methods are employed to furnish two novel low complexity architectures, which take only non-uniformly sampled inputs. Integer multiplications, which are exact in nature, are realized using simple shift-add structures. The designs are fully pipelined by judicious insertion of registers at internal nodes, leading to low critical path delay at the cost of latency.

4.1 Architecture I

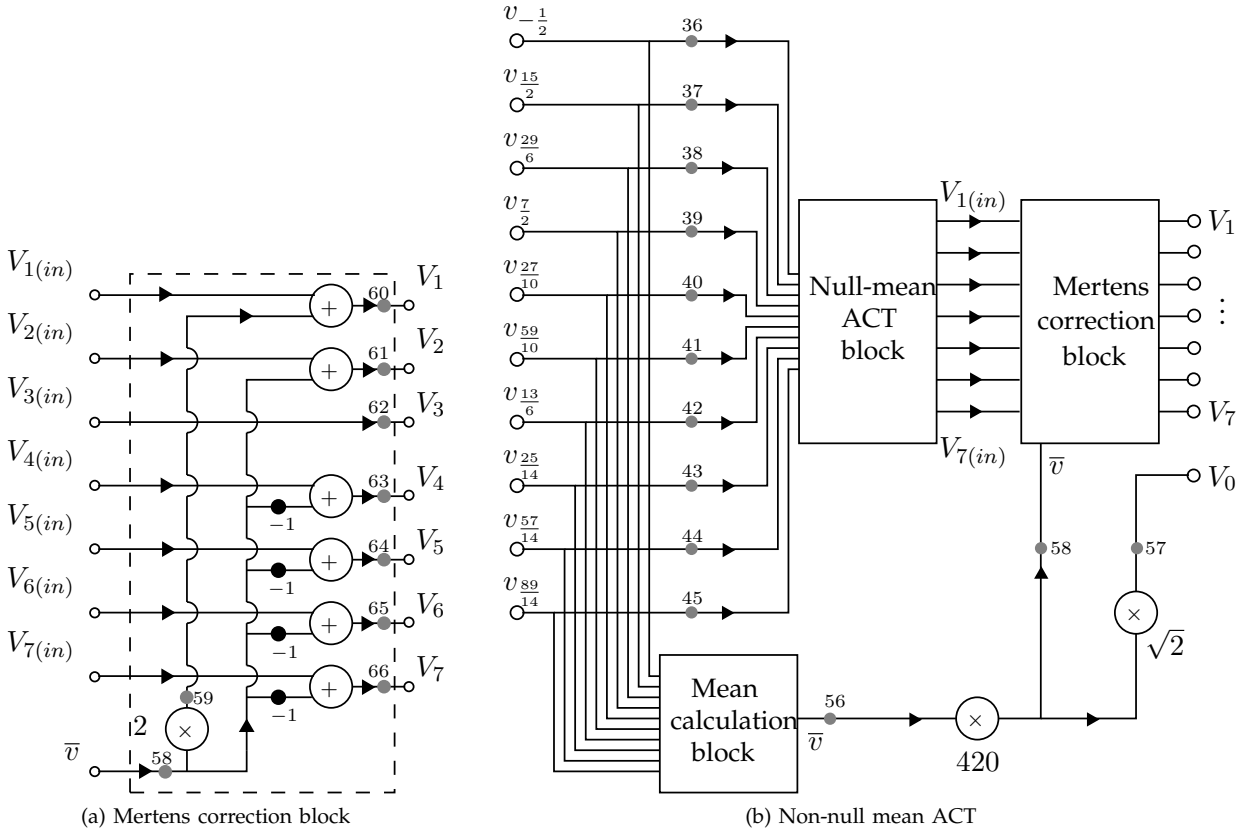
The ACT expressions for null mean signals in (2) and (3) can be implemented for $N = 8$ as shown in Fig. 1(a). We refer to this design as Architecture I. The 8-point null mean ACT block admits the 10 non-uniformly sampled inputs corresponding to (6). Constant multiplications by two are implemented as left-shift operations; and the fractional constant multipliers $1/2, 1/3, 1/4, \ldots$ are converted to integers by scaling them by the least common multiple of their denominators: 420. The integer constant multipliers can be implemented as Booth encoded shift-and-add structures making the architecture multiplier free, and the outputs of the block are scaled by $420 \cdot \sqrt{2/N} = 210$. This architecture is useful in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2014.2366732, IEEE Transactions on Computers



(a) Architecture I (b) Mean calculation block

Fig. 1. (a) Null mean ACT and (b) mean calculation block.



(a) Mertens correction block (b) Non-null mean ACT

Fig. 2. Architecture II: Non-null mean DCT calculation using the Mertens correction block.

TABLE 1
Computational complexity of proposed Architecture I and Architecture II.

	Architecture I	Architecture II
Constant multipliers	0	11
Two-input Adders	36	54

applications that have null mean input sequences, and can be implemented with very low computational and area complexity.

4.2 Architecture II

The proposed method in Section 3 for the computation of $\bar{v}$ from the non-uniformly sampled 10-point signal can be implemented as shown in Fig. 1(b). We will refer to it as the mean calculation block, which computes (7). The correction term associated to the Mertens function required in (4) is shown in Fig. 2(a). A combination of (i) this particular block, (ii) the Architecture I block, and (iii) the mean calculation block yields the proposed Architecture II as shown in Fig. 2(b).

Note that calculation of the DCT coefficients using the null mean ACT block can also be achieved by subtracting the mean $\bar{v}$ from its inputs. However, Architecture II has a lower computational complexity when compared to such alternative. Computation complexity of both Architecture I and Architecture II are listed in Table 1 in terms of constant multipliers and two-input adders. Integer constant multiplications are implemented as shift-and-add structures, therefore are not counted as multipliers. Note that the adder count also include the adders required for the Booth encoded structures.

5 Implementation and Results

5.1 FPGA Implementation

We implemented both architectures described in the previous section. These architectures were tested on Xilinx Virtex-6 XC6VLX240T FPGA using the stepped hardware co-simulation feature in ML605 evaluation platform. They were also fully pipelined to achieve the maximum throughput. Word-length is L at the inputs, which are assumed to be in the range $[-1, 1]$. Throughout the fixed point implementation the word-length increases to avoid overflow. Depending on the particular quantization point, the actual allocated word-length is given by $L + \Delta L$, where the values of ΔL are listed in Table 2 for both proposed architectures. The referred quantization points are shown in Fig. 1 and Fig. 2. The number of fractional bits are maintained constant throughout the design and is equal to $L - 1$.

Accuracy of the results from Architectures I and II were tested with varying values of L by using average percentage error and peak signal to noise ratio (PSNR) as figures of merit. Adopted figures of merit employed the DCT coefficients calculated from the floating point implementation of the DCT available in Matlab as reference. Results given in Table 3 are taken from the simulation of Architectures I and II using 10^4 random input signals. The reduction of the input word-length L degrades the results furnished by the considered figures of merit. However, for small word-lengths, the errors incurred are tolerable for most applications.

TABLE 2
Fixed point word-length increase ΔL at each quantization point of the ACT signal flow graph. Fixed point word-length is $L + \Delta L$

Architecture I		Architecture II	
Points	ΔL	Points	ΔL
1–10	0	36–55	0
11–18	2	56	1
19–22, 24	3	57, 59, 61, 62	13
23	1	58	11
25, 26, 31	10	60	14
27, 32, 34	12	63–66	12
28–30	11		
33, 35	13		

Table 4 shows the resource utilization, power consumption and operational frequency on the Xilinx Virtex-6 XC6VLX240T FPGA device for input fixed point word-lengths (L) 8 and 12. Information about the Xilinx FPGA resources that are listed in Table 4 including slices, slice FFs and 4-input look-up tables (LUTs) can be found in the device datasheet. Architecture I is multiplier-free and possesses the lower complexity, but it is only suitable for null mean signals. To remove the dependence of power consumption to operating frequency the normalized power metric (dynamic power normalized to operating frequency) is given in Table 4. The total power consumption in the FPGA is dominated by the static power since both architectures only occupied roughly 1% of the available area.

5.2 ASIC Synthesis Results

The proposed architecture Architecture I and II are synthesized for application specific integrated circuits (ASIC) using the Cadence RTL Compiler for 45 nm technology. The freePDK45 standard-cell library is used in synthesis with optimization goal set to maximize the speed. Our synthesis was performed at operating voltage of 1.1 V. The area, power, operational frequency, and normalized power metric (dynamic power normalized to operating frequency and square of the supply voltage) for the ASIC synthesis are presented in Table 5.

Table 6 shows the comparison of results between proposed ACT Architectures I and II and other published 8-point DCT implementations. Ideally, a fair comparison requires all implementations to be of the same process, operating frequency, and supply voltage. However, the published literature contains varying technology and operational conditions. Hence in Table 6 a normalized power consumption value is given, where the power consumption is normalized to the corresponding operational frequency and square of supply voltage. From the normalized power consumption given in Table 6 it's apparent that the proposed architectures consume

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2014.2366732, IEEE Transactions on Computers

TABLE 3
Average percentage error and average peak signal to noise ratio (PSNR) of ACT implementations with fixed point input word-length L, when tested with 10,000 input vectors

L	Architecture I		Architecture II	
	% error	PSNR (dB)	% error	PSNR (dB)
8	4.594×10^{-1}	50.3	2.262×10^{-1}	38.8
12	1.977×10^{-2}	74.3	2.149×10^{-1}	63.0
16	-1.840×10^{-3}	98.4	-1.550×10^{-2}	87.1
20	2.943×10^{-4}	122.4	2.565×10^{-3}	110.8
24	-1.001×10^{-5}	145.6	9.462×10^{-6}	135.4
28	1.167×10^{-6}	170.6	3.137×10^{-6}	159.4
32	-2.274×10^{-8}	194.7	3.207×10^{-7}	183.4

TABLE 4
Speed of operation resource utilization and power consumption of the XC6VLX240T FPGA device used for input fixed point word-lengths L and for Architectures I and II

Architecture tested	Fixed point word-length (L)	Slices	Slice FF	Slice LUTs	Dyn. power (W)	Op. freq. (MHz)	Norm. power (W/MHz)
Architecture I	8	263 (1%)	930 (1%)	756 (1%)	1.37	500	2.74×10^{-3}
	12	329 (1%)	1205 (1%)	1019 (1%)	1.16	333.33	3.49×10^{-3}
Architecture II	8	443 (1%)	1276 (1%)	1386 (1%)	0.54	166.66	3.22×10^{-3}
	12	495 (1%)	1678 (1%)	1639 (3%)	0.53	133.33	3.97×10^{-3}

TABLE 5
Speed of operation, critical path delay, power consumption and area utilization in ASIC synthesis results for fixed point word-lengths L for Architecture I (45 nm technology)

Architecture Synthesized	Fixed point word-length (L)	Area (μm^2)	Static power (mW)	Dynamic power (mW)	Total power (mW)	Op. Freq. (GHz)	Norm. Freq. (mW /GHz·V^2)
Architecture I	8	39007.27	0.27	67.31	67.59	1.11	50.12
	12	53961.52	0.37	90.32	90.70	1.11	67.25
Architecture II	8	65314.36	0.46	60.34	60.80	0.625	79.78
	12	96087.77	0.63	79.29	79.92	0.588	111.45

lower power than architectures in [25],[27] and [28]. We emphasize that the proposed Architecture I has the distinct advantage of having *exact* computation. Thus approximate DCT methods as suggested in [23], [24] were not taken into consideration for comparison purposes.

6 Conclusions

The ACT algorithm is suitable for calculating the 8-point DCT coefficients exactly using only adders and integer constant multiplications, also with low computational complexity. ACT architectures for null mean inputs as well as for non-null mean inputs are proposed, implemented and tested on Xilinx Virtex-6 XC6VLX240T FPGA. The average percentage error and PSNR were adopted as figures of merit to assess the measured results. Results show that even for lower fixed point word-lengths, the implementations lead to acceptable margins of error. The resource utilization for various fixed point implementations indicate a trade-off between accuracy and device resources (chip area, speed, and power). It is the first step towards new research on low power and low complexity computation of the DCT by means of the recently proposed ACT.

Acknowledgments

This work was supported by The University of Akron, Ohio, USA; Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and FACEPE, Brazil; and NSERC, Canada.

References

[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *Computers, IEEE Transactions on*, vol. 100, no. 1, pp. 90–93, 1974.

[2] C. Chakrabarti and J. JáJá, "Systolic architectures for the computation of the discrete Hartley and the discrete cosine transforms based on prime factor decomposition," *Computers, IEEE Transactions on*, vol. 39, no. 11, pp. 1359–1368, 1990.

TABLE 6
Comparison of the proposed implementation with published DCT implementations. Some implementations are 2-D but since they are implemented with 1-D DCT module with row column decomposition, results can be taken that can be compared with the proposed architectures.

	Gong *et al.* [25]	Shams *et al.* [26]	Gosh *et al.* [27]	Livramento *et al.* [28]	Proposed architectures			
					Arch. I	**Arch. I**	**Arch. II**	**Arch. II**
1D/2D DCT	2D but 1D results available	1D	2D but 1D results available	2D but 1D results available	**1D**	**1D**	**1D**	**1D**
Replicated and measured results by authors	No	No	No	No	**Yes**	**Yes**	**Yes**	**Yes**
Precision	Non-exact	Non-exact	Non-exact	Non-exact	**Exact**	**Exact**	**Non-exact**	**Non-exact**
Method	Vector matrix DCT core	New distributed arithmetic based DCT	Coefficient arithmetic based DCT	LLM algorithm	**ACT, null mean**	**ACT, null mean**	**ACT, Mertens function**	**ACT, Mertens function**
Multipliers	8	0	0	11	**0**	**0**	**11**	**11**
Input word-length	12	9	9	8	**8**	**12**	**8**	**12**
Operating frequency (GHz)	0.125 (2D)	1.5	0.05 (2D)	0.00489 (2D)	**1.11**	**1.11**	**0.625**	**0.588**
Pixel rate ($\times 10^9 s^{-1}$)	0.125	12	0.4	1.792	**8.88**	**8.88**	**5.00**	**4.70**
Power consumption (mW)	N/A	210	12.45 (1D)	6.08 (2D)	**67.31** ‡	**90.32** ‡	**60.34** ‡	**79.29** ‡
Normalized power consumption ($mW/GHz \cdot V^2$)	N/A	12.86	110.67	114.17	**50.11**	**67.25**	**79.79**	**111.44**
Gate count	30290	N/A	N/A	N/A	**11491**	**16478**	**17673**	**25197**
Implementation technology	0.25 μm CMOS	0.35 μm CMOS	0.12 μm CMOS	0.35 μm CMOS	**45 nm CMOS**	**45 nm CMOS**	**45 nm CMOS**	**45 nm CMOS**
Supply voltage (V)	2.5	3.3	1.5	3.3	**1.1**	**1.1**	**1.1**	**1.1**

[‡] Dynamic power.

[3] F. A. Kamangar and K. R. Rao, "Fast algorithms for the 2-D discrete cosine transform," *Computers, IEEE Transactions on*, vol. 100, no. 9, pp. 899–906, 1982.

[4] H. Kitajima, "A symmetric cosine transform," *Computers, IEEE Transactions on*, vol. 100, no. 4, pp. 317–323, 1980.

[5] S. Yu and E. Swartziander Jr, "DCT implementation with distributed arithmetic," *Computers, IEEE Transactions on*, vol. 50, no. 9, pp. 985–991, 2001.

[6] V. Britanak, P. Yip, and K. R. Rao, *Discrete cosine and sine transforms*. Amsterdam: Academic Press, 2007.

[7] N. Roma and L. Sousa, "Efficient hybrid DCT-domain algorithm for video spatial downscaling," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, no. 2, pp. 30–30, 2007.

[8] H. Lin and W. Chang, "High dynamic range imaging for stereoscopic scene representation," in *Proceedings of the 16th IEEE International Conference on Image Processing (ICIP)*, Nov. 2009, pp. 4305–4308.

[9] E. Magli and D. Taubman, "Image compression practices and standards for geospatial information systems," in *Proceedings of the 2003 IEEE International Geoscience and Remote Sensing Symposium*, vol. 1, Jul. 2003, pp. 654–656.

[10] M. Bramberger, J. Brunner, B. Rinner, and H. Schwabach, "Real-time video analysis on an embedded smart camera for traffic surveillance," in *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium*, May 2004, pp. 174–181.

[11] C. F. Chiasserini and E. Magli, "Energy consumption and image quality in wireless video-surveillance networks," in *Proceedings of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 5, Sep. 2002, pp. 2357–2361.

[12] T. Tada, K. Cho, H. Shimoda, T. Sakata, and S. Sobue, "An evaluation of JPEG compression for on-line satellite images transmission," in *Proceedings of the International Geoscience and Remote Sensing Symposium (IGARSS)*, Aug. 1993, pp. 1515–1518.

[13] B. Bennett, C. Dee, and C. Meyer, "Emerging methodologies in encoding airborne sensor video and metadata," in *Proceedings of the 2009 IEEE Military Communications Conference*, Oct. 2009, pp. 1–6.

[14] S. Marsi, G. Impoco, A. Ukovich, S. Carrato, and G. Ramponi, "Video enhancement and dynamic range control of HDR sequences for automotive applications," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, p. MISSING PAGES, 2007.

[15] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer Networks*, vol. 51, no. 4, pp. 921–960, 2007.

[16] J. G. Proakis and D. G. Manolakis, *Digital signal processing*. Upper Saddle River, NJ: Pearson Prentice-Hall, 2007.

[17] I. S. Reed, D. W. Tufts, X. Yu, T. K. Truong, M. T. Shih, and X. Yin, "Fourier analysis and signal processing by use of the Möbius inversion formula," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP-38, no. 3, pp. 458–470, Mar. 1990.

[18] I. S. Reed, M. T. Shih, T. K. Truong, E. Hendon, and D. W. Tufts, "A VLSI architecture for simplified arithmetic Fourier transform algorithm," *IEEE Transactions on Signal Processing*, vol. 40, no. 5, pp. 1122–1133, May 1992.

[19] R. J. Cintra and V. S. Dimitrov, "The arithmetic cosine transform: Exact and approximate algorithms," *IEEE Transactions on Signal Processing*, vol. 58, no. 6, pp. 3076–3085, Jun. 2010.

[20] E. J. Tan, Z. Ignjatovic, and M. F. Bocko, "A CMOS image sensor with focal plane discrete cosine transform computation," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, May 2007, pp. 2395–2398.

[21] S. G. Krantz, *Real Analysis and Foundations*. Boca Raton, FL: Chapman & Hall/CRC, 2005.

[22] I. C. F. Ipsen, *Numerical Matrix Analysis: Linear Systems and Least Squares*. Philadelphia, PA: SIAM, 2009.

[23] J. Liang and T. D. Tran, "Fast multiplierless approximations of the DCT with the lifting scheme," *IEEE Transactions on Signal Processing*, vol. 49, no. 12, pp. 3032–3044, Dec. 2001.

[24] T. D. Tran, "The binDCT: fast multiplierless approximation of the DCT," *IEEE Signal Processing Letters*, vol. 7, no. 6, pp. 141–144, Jun. 2000.

[25] D. Gong, Y. He, and Z. Cao, "New cost-effective VLSI implementation of a 2-D discrete cosine transform and its inverse," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 4, pp. 405–415, Apr. 2004.

[26] A. Shams and M. Bayoumi, "A 108 Gbps, 1.5 GHz 1D-DCT architecture," in *Proceedings. IEEE International Conference on Application-Specific Systems, Architectures, and Processors*, 2000, pp. 163–172.

[27] S. Ghosh, S. Venigalla, and M. Bayoumi, "Design and implementaion of a 2D-DCT architecture using coefficient distributed arithmetic [implementaion read implementation]," in *Proceedings. IEEE Computer Society Annual Symposium on VLSI*, May 2005, pp. 162–166.

[28] V. S. Livramento, B. G. Moraes, B. A. Machado, and J. L. Guntzel, "An energy-efficient 8×8 2-D DCT VLSI architecture for battery-powered portable devices," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2011, pp. 587–590.