Volume 3, Issue 5, May 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

VHDL Implementation of MDCT/IMDCT for Audio Codec Application

Sonali Baroniya
PG Student (M.tech) (Digital Communication)
B.I.E.T., Jhansi, India.

Dr. Deepak Nagaria
Head of the EE department,
READER ECE, B.I.E.T., Jhansi, India.

Abstract--MDCT/ IMDCT is used to realize analysis of filter bank for reduce the time domain aliasing cancellation and scheme for sub-band coding. This paper presents a fast MDCT/IMDCT algorithm for implementation of low complexity by using DCT type –IV coefficients. In algorithm derivation, the MDCT/IMDCT computation is first converted into a form of matrix multiplication consisting of a half size DCT-IV kernel and a projection matrix. The DCT-IV kernel is then realized by a fast DCT-II computing scheme. Since MDCT and IMDCT algorithms use the same DCT kernel, so this paper uses the same architecture for MDCT and IMDCT. The proposed design requires less hardware utilization and less memory complexity for storing the spectral coefficient and temporary data.

Keyword: Modified Discrete Cosine transform (MDCT), Discrete Cosine Transform (DCT), Time Domain Aliasing Cancellation (TDAC)

I INTRODUCTION:

The modified discrete cosine transform (MDCT)/inverse MDCT (IMDCT) is extensively used to realize the analysis filter bank of time domain aliasing cancellation scheme for sub band coding. This transform has been adopted in several international standards and commercial products such as MPEG-1, MPEG-2, and AC-3 in audio coding. However, the computation of the MDCT and IMDCT in the MPEG audio coder can involve an extensive computation. Therefore, there is a need for a fast method to compute the forward and Inverse MDCT so that it is naturally suitable for hardware or firmware implementations [1, 2].

The Modified Discrete Cosine Transform is a transformation technique to transform time to frequency domain representation. It is based on the Discrete Cosine Transform-IV (DCT- IV). Therefore it has a 50% overlapping property, which is mainly designed for large consecutive block of dataset. This overlapping property can increase the energy compaction to Discrete Cosine Transform [3, 4].

II MDCT/IMDCT ALGORITHM:

MDCT (Modified Discrete Cosine Transform) can be represented by the following formula [1, 2]:

$$Y(k) = \sum_{n=0}^{N-1} y(n) cos\left[\frac{\pi}{2*N}\left(2*n+1+\frac{N}{2}\right)*(2*k+1)\right], \quad 0 \le k \le \frac{N}{2}-1 \qquad (1)$$

And the Inverse Modified Discrete Cosine Transform will be governing by

$$\hat{y}(n) = \sum_{k=0}^{N/2-1} y(n) cos\left[\frac{\pi}{2*N}\left(2*n+1+\frac{N}{2}\right)*(2*k+1)\right], \quad 0 \le n \le N-1 \qquad (2)$$

Here $\hat{y}(n)$ is the recovered signal after IMDCT but it is not perfectly reconstructed. For perfectly reconstruction use the windowing function before MDCT in encoder side and same windowing function after IMDCT in decoder side. According to [5], an MDCT computation can be expressed in a matrix form as

$$Y = \sqrt{\frac{N}{4}} * C_{N/2}^{IV} * S * P^T * y \qquad (3)$$

Where $C_{N/2}^{IV}$ is a size N/2 type IV DCT computing matrix, S is and P are sign and permutation matrices, respectively. The type IV DCT is further converted into a type II kernel plus some pre- and post processing [4, 5]

$$\sqrt{\frac{N}{4}} C_{N/2}^{IV} = L * C_{N/2}^{II} * D \qquad (4)$$

Where D is a diagonal scaling matrix and L is lower triangular matrix corresponding to recursive subtractions. In [5], the fast computation of type II DCT is not specified. The shared computing kernel of MDCT and IMDCT is not derived,

either. In this paper, based on the framework of [6], and also re-derive the fast computing algorithms of MDCT and IMDCT so that both transforms can share a maximum part of computing kernel. In addition, an efficient type II DCT realization is given. Starting with MDCT, first convert it into a matrix product of a type IV DCT and a projection matrix P. The derivation process is left out to save the space and the final result is given (5).

$$Y = C^t . P . y \tag{5}$$

Where C' is a size N/2 type IV DCT with coefficients

$$c^t(k,n) = \cos\left[\frac{(2n+1)(2k+1)\pi}{2N}\right] = C'(k,n) \tag{6}$$

And the projection matrix P is defined as

$$P = \begin{pmatrix} 0 & 0 & -J_{N/4} & -I_{N/4} \\ I_{N/4} & -J_{N/4} & 0 & 0 \end{pmatrix} \tag{7}$$

Similarly matrix form of IMDCT can be written as

$$y = P^T . C' . X \tag{8}$$

It is clear that MDCT and IMDCT now share the same type IV DCT kernel C' but differentiate from each other by the order of applying projection. A direct implementation of type IV DCT kernel would require order $O\left((N/2)^2\right)$ computations. [7, 8]

Therefore, fast computing algorithm must be employed. It is well known that a type IV DCT can be converted into a type II DCT. That is

$$C' = R . C'' . D' \tag{9}$$

Where R

$$\mathrm{R}_{\frac{N.N}{2.2}} = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 1 & 0 & 0 & \cdots & 0 \\ 1 & -1 & 1 & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & \vdots & & \ddots & \vdots \\ -1 & 1 & \cdots & \cdots & 0 & 0 \end{pmatrix} \tag{10}$$

And diagonal matrix D with element

$d(i.j) = \cos\left[\frac{\pi}{2*N}(2*i+1)\right]$, For $i = 0,1,2,\ldots\left(N/2 - 1\right)$ (11)

And C'' is the type II DCT coefficient

$C_{N/2}^{II} = 1$, For $i = 0,1,2,\ldots\left(N/2 - 1\right), J = 0$

$C_{N/2}^{II} = 2 * \cos\left[\frac{\pi}{2*N}(2*i+1)*J\right]$, For $i = 0,1,2,\ldots\left(N/2 - 1\right), J = 1,\ldots.\left(N/2 - 1\right)$ (12)

The matrix form of MDCT and IMDCT are

MDCT:

$Y = R . C'' . D' . P . Y$ (13), IMDCT $y = P^t \cdot \mathrm{R} . C'' . D' . Y$ (14)

from the above equation get the computing diagram for MDCT and IMDCT are shown in fig.1. and fig.2. [20, 21]

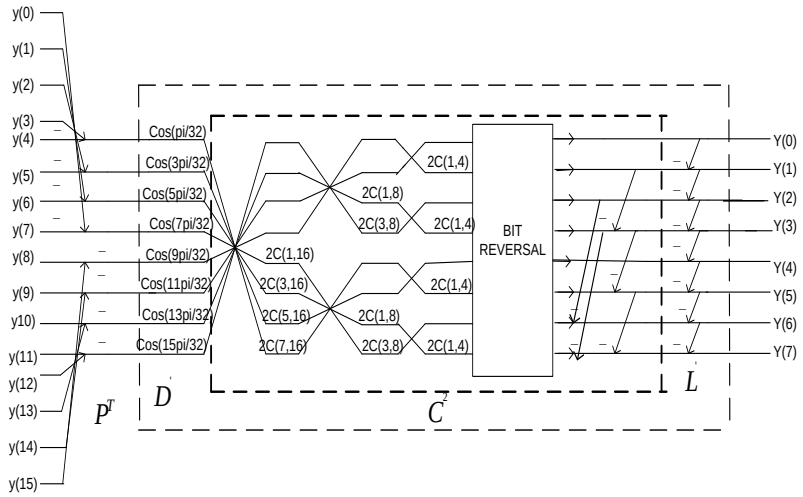

Fig.1. computing diagram for MDCT

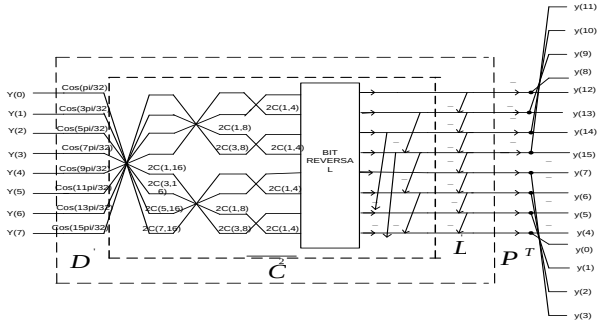

Fig.2. computing diagram for MDCT

III. ALGORITHM MAPPING AND MEMORY ACCESS CONTROL:

Note that the issues of computing complexity reduction and a shared DCT kernel for both MDCT and IMDCT algorithms have already been resolved in algorithm development. And next address the algorithm mapping to derive an efficient hardware design. In algorithm mapping, this paper needs to first determine the hardware allocation and then derive its scheduling. As mentioned, MDCT/IMDCT algorithms are mostly for audio application with only moderate data rate. Therefore, allocation of only one butterfly module suffices to meet the computing demand in real time operation. The structure of a butterfly module is given in fig.3. It requires two adders and one multiplier. In each clock cycle, it accesses two input data and generates two outputs. It also needs to access one spectral coefficients. In addition this butterfly module is also use in the projection matrix and recursive addition (last stage of the computing diagram). The basic computing butterfly module is given below [6, 7]

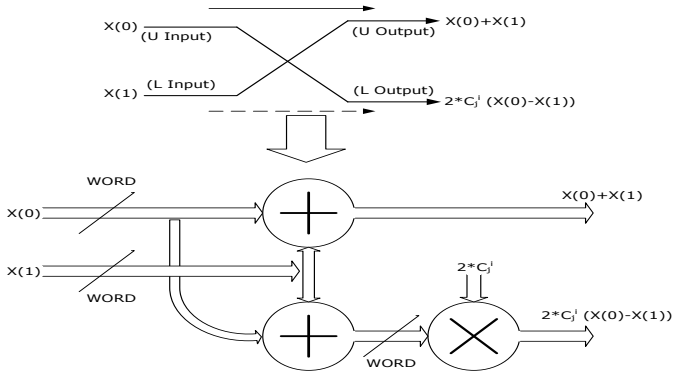

Fig.3. Butterfly modules

It should be noted that the data access sequence (scheduling) in fig.4. And the data locations in the memory module play a crucial role in the design performance. Since two data will be consumed by a computing butterfly to generate two output data, a minimum memory bandwidth of 4 words two for data read and two for data write. Using expensive dual-port memories is often considered as the only viable solution to tackle the bandwidth problem. In this paper, instead, with careful scheduling and data allocation, four size *NI4* single port memory modules suffice. The four memory modules are organized into *two* sets, one for read and one for write Each all set contains two memory modules (banks) for two simultaneous memory accesses.

Y(k)/stage	1	2	3
Y(0)	$0_{(1)}$	$0_{(1)}$	$0_{(1}$
Y(1)	$1_{(1)}$	$1_{(1)}$	$\underline{0}_{(1)}$
Y(2)	$2_{(1)}$	$\underline{1}_{(1)}$	$1_{(2)}$
Y(3)	$3_{(1)}$	$\underline{0}_{(1)}$	$\underline{1}_{(2)}$
Y(4)	$\underline{3}_{(1)}$	$2_{(2)}$	$2_{(3)}$
Y(5)	$\underline{2}_{(1)}$	$3_{(2)}$	$\underline{2}_{(3)}$
Y(6)	$\underline{1}_{(1)}$	$\underline{3}_{(2)}$	$3_{(4)}$
Y(7)	$\underline{0}_{(1)}$	$\underline{2}_{(2)}$	$\underline{3}_{(4)}$

Fig. 4 Data access sequence

The memory mapping of first two stages in 8 point DCT is given fig.5 and fig.6. This memory mapping used two counters for read and one counter for write one of the read counters is up counter and another is down counter but in writing operation use only up counter for both memory banks. [8, 9]

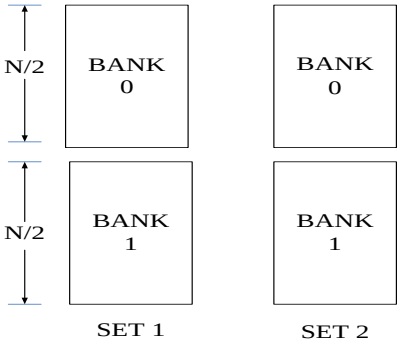

Fig. 5 Memory organization

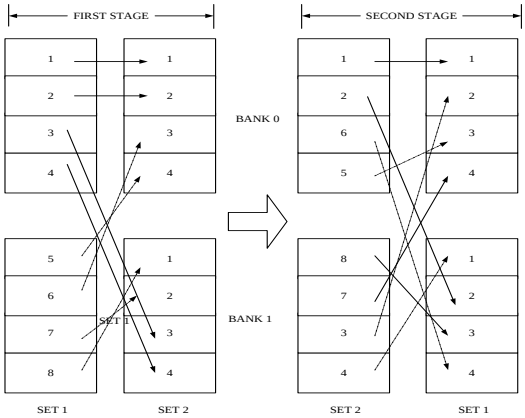

Fig.6.Memory mapping for first two stages in DCT 8 point

Fig. **7** illustrates a possible data access graph between tags. The number in a table entry can be regarded as the array index. Its parenthesized subscript represents the butterfly group it belongs. Each pair of solid and dashed arrowed lines indicates the memory location change after the butterfly computation (assume the memory address pace is along the vertical direction). The two sets of memory modules work alternately as the input and the output data buffers across the stages. [12, 14]

Y(k)/stage	1	2	3
Y(0)	$0_{(1)}$	$0_{(1)}$	$0_{(1)}$
Y(1)	$1_{(1)}$	$1_{(1)}$	$1_{(1)}$
Y(2)	$2_{(1)}$	$2_{(1)}$	$3_{(2)}$
Y(3)	$3_{(1)}$	$3_{(1)}$	$2_{(2)}$
Y(4)	$4_{(1)}$	$7_{(2)}$	$7_{(3)}$
Y(5)	$5_{(1)}$	$6_{(2)}$	$6_{(3)}$
Y(6)	$6_{(1)}$	$5_{(2)}$	$4_{(4)}$
Y(7)	$7_{(1)}$	$4_{(2)}$	$5_{(4)}$
	Mem. set 1	Mem. set 2	Mem. set 1

Fig. 7 Data Access graph for 8 point DCT

MDCT and IMDCT use same type of computing diagram for implementation and implemented them into a same architecture Figure 8 shows the MDCT/IMDCT architecture. In the top level of this architecture two paths one is data path for data processing and other control path for controlling the data into memory and data bus. [11]

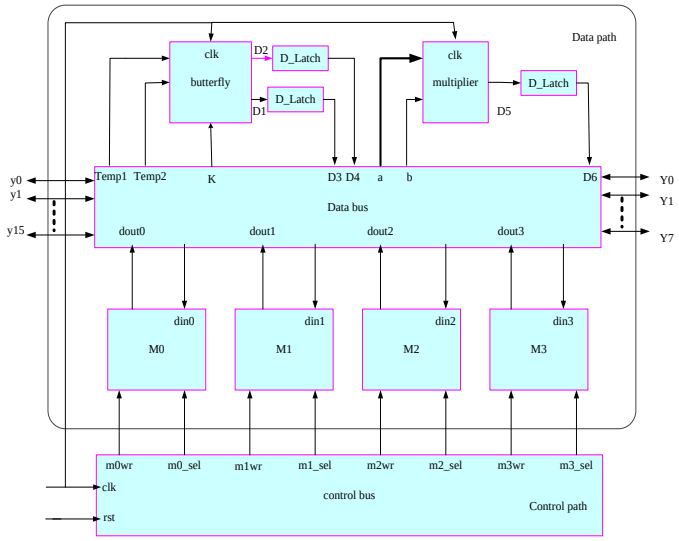

Fig. 8 MDCT/IMDCT architecture

Table 1 shows the synthesis paper of MDCT for 16 bits and 20 bits fixed point implementation used the (1, 4, 11) format for 16 bits Where MSB shows the sign bits, next 4 bits represent the integer part And last 11 bits represent the fractional part, similarly in the 20 bits implementation use (1, 4, 15) format. [11, 17]

TABLE.1. FPGA IMPLEMENTATION RESULT FOR MDCT/IMDCT MODULE

Resource	16 bit fixed point method	20 bit fixed point method
Device	3s500epq208-5	3s500epq208-5
Spectral coefficients	16 bit	20 bit
Data and temporal variable	16 bit	20 bit
Number of slices	828/4656	1037/4656
Slice flip flop	892/9312	1065/9312
LUT	1109/9312	1427/9312
Block RAM	4 number of 4x16-bit single-port RAM	4 number of 4x20-bit single-port RAM
18*18 Multiplier	2/20	6/20
Area constraint ratio	19%	24%
Maximum clock frequency	88.53 MHz	87.633 MHz

IV. PSEUDO CODE FOR VERIFICATION:

For verification take a signal of 32 samples and framing is done in such a way that 50% overlapping between the frames will occur then and apply the sine window.

The output of sine window shown in fig. 9(b). MDCT/IMDCT flow diagram shown in fig.10. as describe in flow framing and windowing is done in MATLAB then output of the framing signal is converted to binary number(as format (1,4,11) 16 bits fixed point representation) now MDCT and IMDCT are apply using FILE I/O operation. MDCT/IMDCT is done in VHDL. After getting result of IMDCT convert it to decimal number by using the num2bin command in MATLAB which are shown in figure 9(d). After converting to further use the sine window and after that unframed the signal in MATLAB get the output of reconstructed signal which are shown in figure 9(e). From figure 9 (a), 9 (e) and 9 (f) see that our MDCT and IMDCT gives nearly same result in VHDL

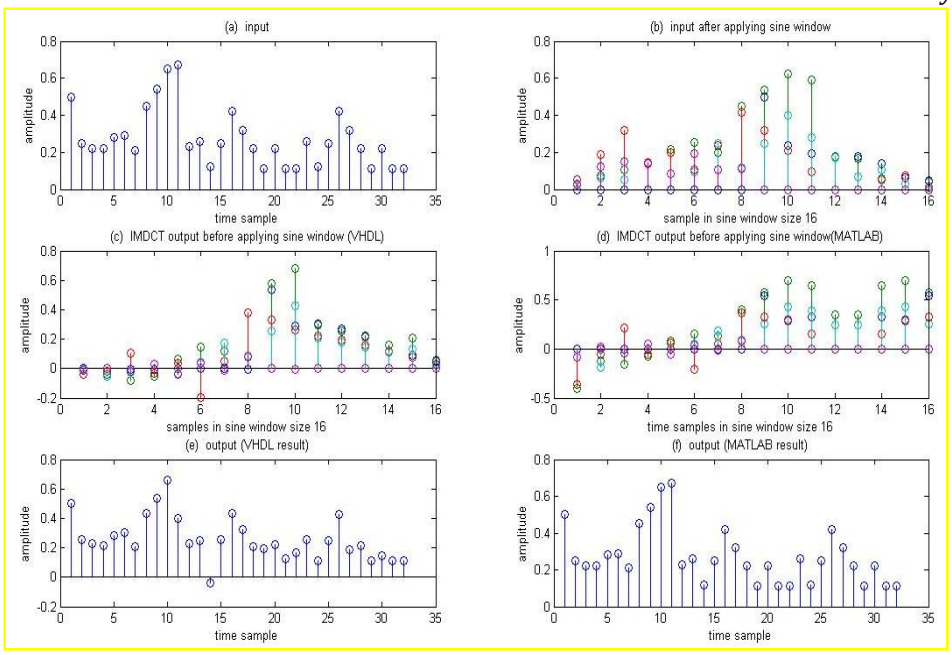

Fig.9.Verification of MDCT/IMDCT

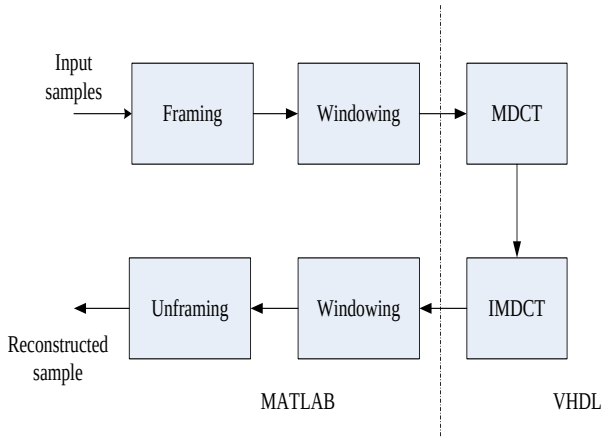

Fig. 10 MDCT/IMDCT flow

V. PERFORMANCE EVALUATIONS AND SUMMARY:

Table 2 this paper summarizes computing analysis of MDCT/IMDCT. 2 adder and one multiplier is used in our algorithm this table shows comparative analysis of hardware in term of number of adder and multiplier. In this analysis our algorithm requires less number of adder and multiplier. Table 3 summarizes the memory uses for the spectral coefficient and temporary data which are responsible for output.

TABLE 2 Computing Analysis of MDCT/IMDCT

method	MDCT		IMDCT		Coefficient sharing	MUX logic
	add	mul	add	mul		
[7]	3	3	3	2	NO	YES
[8]	3	3	3	2	NO	YES
Ours	2	1	2	1	YES	NO

TABLE 3 MDCT/IMDCT Memory Storage Complexities.

method	MDCT		IMDCT		Word
	Spect Coffi.	Temp. data	Spect Coffi.	Temp. data	
[7]	24	8	24	16	40
[8]	24	8	24	16	40
ours	16	8	16	8	24

Table 4 summarizes the computing latency analysis for the MDCT/IMDCT. Where $v=\log_2 N$ from this table see that our how many clock require giving the output. The computing latency is equal to the number of butterflies plus extra ADD cycles in pre- and post processing stages. In fixed point performance analysis, also assume the coefficients and all the data variables are truncated to a fixed word length ***R*** and conduct fixed point simulation in MATLAB to determine its PSNR value. PSNR (peak signal to noise ratio) is defined as

$$PSNR = 10 * log\left(N * (maxF_i)^2 \Big/ \sum_i^N (F_i - f_i)^2 \right) dB$$

TABLE 4 MDCT/IMDCT Computing Latency Analysis

Method	MDCT		IMDCT	
	complexity	N= 16	complexity	N= 16
[7]	N^2	256	N^2	256
[8]	N^2	256	N^2	256
Ours	Nv	64	$N(v-1)$	48

From table 5, our scheme has a significant performance edge over the regressive approach scheme at different word lengths.

TABLE 5 Fixed point performance analysis for MDCT /IMDCT

Method		R=16 bits		R=20 bits	
		MDCT	IMDCT	MDCT	IMDCT
[7]	MSE	-	-	-	-
	PSNR	overflow	overflow	17.36	-
ours	MSE	0.0865	0.0162	0.0001	0.01380
	PSNR	36.82	32.04	62.406	32.75

VI. CONCLUSION:

In conclusion, in this paper implemented an efficient MDCT/IMDCT scheme and its hardware design targeting for audio application. In this paper first derived a fast computing algorithm with lowest computing complexity and memory storage size among competitive works. Based on the derived scheme, an efficient memory access scheme was devised which features simple address generation, small temporary storage size and low access bandwidth. Only four size NI4 single port memory modules are required and no memory access conflict occurs to stall the computation. And also conduct extensive performance Analyses among different schemes and the results show the superiority of the proposed design.

REFERENCES:

[1]. J.P. Princen and A.B. Bradley, "Analysis/synthesis filters bank designs based on time domain aliasing cancellation", IEEE Trans. Acoust. Speech Signal Process**.** ASSP 34, October 1986, pp. 1153–1161.

[2]. J.P. Princen, A.W. Johnson, A.B. Bradley, "Sub-band/transform coding using filter bank designs based on time domain aliasing cancellation", Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, Dallas, TX, April 1987, pp. 2161–2164.

[3]. Po-sheng Wu Yin-Tsung Hwan," Efficient IMDCT core designs for audio signal processing", Signal Processing Systems, 2003. SIPS 2003. IEEE Workshop 27- 29 August 2003, pp. 275- 280.

[4]. P. Duhamel, Y. Mahieux, J.P. Petit, "A fast algorithm for the implementation of filter banks based on time domain aliasing cancellation", Proceedings of the IEEE International Conference on Acoustics, Speech, Signal Processing 1991, Toronto, ON, Canada, May 1991, pp. 2209–2212.

[5]. C.H. Chen, B.D. Liu and J.F. Yang," Recursive architectures for realizing modified discrete cosine transform and its inverse", IEEE Trans. Circuits Systems-II: Analog Digital Signal Process. 50, January 2003, (1), pp. 38–45.

[6]. C.M. Liu and W.C. Lee, "A unified fast algorithm for cosine modulated filter banks in current audio coding standards", J. Audio Eng. Soc. 47, December 1999, (12), pp. 1061–1075.

[7]. V.B. Britanak and K.R. Rao, "An efficient implementation of the forward and inverse MDCT in MPEG audio coding", IEEE Signal Process. Lett. 8, February 2001, (2), pp. 48–51.

[8]. H. Cheng and Y.H. Hsu," Fast IMDCT and MDCT algorithms—a matrix approach", IEEE Trans. Signal Process. 51, January 2003, (1), pp. 221–229.

[9]. S.W. Lee,"Improved algorithm for efficient computation of the forward and backward MDCT in MPEG audio coder", IEEE Trans. Circuits Syst.-II: Analog Digital Signal Process. 48, October 2001, (10), pp. 990–994.

[10]. K. Konstantinides, "Fast subband filtering in MPEG audio coding" IEEE Signal Process. Lett. 1, February 1994 (2), pp. 26–28.

[11]. C. H. Chen and others "Recursive architectures for the forward and inverse modified discrete cosine transforms," in Proc. IEEE Workshop Signal Process. Syst., 2000, pp. 50–59.

[12]. Yaroslavsky, L.; Vilermo, M.; Vaananen, M. "Some Peculiar Properties of the MDCT" Wang, Y.; Signal Processing Proceedings, 2000. WCCC-ICSP 2000. 5th International Conference on Vol. 1, 21-25 August 2000, pp.61 – 64.

[13]. Yin-Tsung Hwang; Shin-Chi Lai; "A novel MDCT/IMDCT computing kernel design" Signal Processing Systems Design and Implementation, 2005. IEEE Workshop on 2-4 Nov. 2005, pp.526 – 531.

[14]. Ulises S. Mendoza, Ren´e de Jes´us Romero; "VHDL Core for the Computation of the One Dimensional Discrete Cosine Transform" Proceedings of the 2006 International Conference on Reconfigurable Computing and FPGAs, IEEE Computer Society, 20-22 September.

[15]. H.-C. Chiang, J.-C. Liu,"Regressive implementations for the forward and inverse MDCT in MPEG audio coding", IEEE Signal Processing Letters, Vol. 3, April 1996, (4), pp.116 – 118.

[16]. V. Nikolajevic, G. Fettweis, "Computation of forward and inverse MDCT using Crenshaw's recurrence formula", IEEE Transactions on Signal Processing, Vol. 51,(5), May 2003, pp. 1439 – 1444.

[17]. Shin-Chi Lai; Sheau-Fang Lei; Ching-Hsing Lou," Common Architecture Design Of Novel Recursive MDCT and IMDCT Algorithms for Application to AAC, AAC in DRM, and MP3 Codec's," IEEE Transactions on Circuits and Systems II, Vol. 56, Oct. 2009, (10), pp, 793 – 797.

[18]. Hui Li; Ping Li; Yiwen Wang," An efficient hardware Implementing fast IMDCT computation," IEEE Transactions on 26 August 2009, Revised 5 February 2010, accepted 24 February 2010, Available online 11 March 2010.

[19]. "A survey of efficient MDCT implementations in MP3 audio coding standard", Retrospective and state-of-the-art, Review Article Signal Processing, Vol. 91, April. 2011, (11), pp. 624 – 672.

[20]. Venkataramana, A. ; Raj, P.A. ," Recursive Computation of Forward Krawtchouk Moment Transform Using Clenshaw's Recurrence Formula", IEEE Transactions on Computer Vision, Pattern Recognition, Image Processing and Graphics, 2011 (NCVPRIPG), Third National Conference, Dec. 2011,(15-17), pp. 200-203.

[21]. Shin-Chi Lai ; Yi-Ping Yeh ; Wen-Chieh Tseng ; Sheau-Fang Lei ,"Low An High-Accuracy Design of Fast Recursive MDCT/MDST/IMDCT/IMDST Algorithms and Their Realization", IEEE Transactions on Signal Processing, Vol. 59, Jan 2012, (1), pp. 65 – 69.

