An MDCT Hardware Accelerator for MP3 Audio

Xingdong Dai
LSI Corporation
1110 American Parkway NE
Allentown, PA 18109
xingdong.dai@lsi.com

Meghanad D. Wagh
Dept. of ECE
Lehigh University
Bethlehem, PA 18015
mdw0@lehigh.edu

***Abstract*— With the increasing popularity of MP3 audio, there is a need to develop cost and power efficient architectures for the MP3 encoder and decoder. This paper describes dedicated architectures for computing the modified discrete cosine transform (MDCT) and its inverse (IMDCT). Recent profiling studies have shown that these operations represent about 30% of the total MP3 computations. MP3 format defines two frame sizes that can occur in the same data stream. We have developed the most efficient algorithms for MDCT and IMDCT suitable for both sizes. Unlike previous algorithms, our computations can be unified in a single ASIC architecture. This unified architecture implemented in 90 nm TSMC library is still 25% smaller and 25% faster than any previous single frame size architectures designed in the same technology. In addition, at 128 Kbits/sec data rates, our algorithms save nearly 1800 multiplications per second (18%) which can help reduce the power consumption.**

I. Introduction

The MPEG-1/2 layer-III (MP3) standard is widely employed in music industry because of its efficient audio compression. A key enabler in MP3 coding is the prefect reconstruction (PR) cosine modulated filter bank based on the concept of time domain aliasing cancellation (TDAC) [1]. For the encoder, this analysis filter bank is realized by applying the *modified discrete cosine transform* (MDCT) to sliding blocks of data, while the synthesis filter bank of the decoder uses the *inverse modified discrete cosine transform* (IMDCT). Since the MDCT and IMDCT require intensive computations, fast algorithms and efficient implementations for theses transforms is important to the realization of high quality audio compression, especially when most MP3 audio players are battery operated [2]–[4].

The N point MDCT of a sequence $\{x(i)\}$ is defined as

$$X(k) = \sum_{i=0}^{N-1} x(i) \cos(\frac{\pi(2i+1+\frac{N}{2})(2k+1)}{2N}), \quad 0 \le k < N/2.$$

The inverse MDCT (IMDCT) is defined as

$$x(i) = \frac{2}{N} \sum_{k=0}^{\frac{N}{2}-1} X(k) \cos(\frac{\pi(2i+1+\frac{N}{2})(2k+1)}{2N}), \quad 0 \le i < N.$$

Note that MDCT converts N signal samples into only $N/2$ transform samples. MP3 standard defines two data frames of 1152 and 384 samples. These frames are further divided in 32 subbands to be processed by the MDCT/IMDCT, resulting a long block of 36 samples and a short block of 12 samples. The switch from a long block to a short block is called window switching and it is used to supress distortions frequently associated with frequency doamin coding of an audio signal. The ability to process both long and short blocks is crucial to efficient implementations of MP3 audio encoder and decoder.

Because the MP3 block lengths are not power of two, only a few fast algorithms are published and the focus has been on software implementations [5]–[7]. However, over the years the cost of application specific architectures has decreased substantially. In addition, such dedicated architectures can be embedded in field programmable gate arrays making them just as flexible as software routines. Such an approach permits one to utilize a low-cost and low-power processor to work on data management and other computationally simple tasks while the dedicated hardware can take care of the computationally challenging tasks.

This paper is devoted to the development of the MDCT/ IMDCT hardware accelerator for MP3 audio applications. Based on a group theoretic partitioning of the transform kernel, our solutions employ carefully crafted bilinear algorithms which can be directly mapped to hardware architectures. We show for the first time that both the long and short MP3 audio blocks can be seamlessly processed with a single hardware architecture. Our MDCT/IMDCT algorithms require only 9 multiplications for the short block and 36 multiplications for the long block. These may be compared with the current best algorithms for the same tasks which use at least 11 and 43 multiplications respectively [5]–[7]. However, the main advantage of the proposed algorithms is its bilinearity which implies that all the multiplications are independent and can be carried out concurrently. Thus when implemented in hardware, our algorithms have only one multiplication on the critical path for both the long and short block MDCT/IMDCT. Previous reported algorithms have at least 2 multiplications along the critical path [5]–[7]. As a result, our VLSI implementations reduce the critial path delay by about 25% while simultaneously saving about 25% of the chip area.

II. Algorithm and architecture

A bilinear algorithm is made up of an addition stage followed by a stage of independent multiplications and a final addition stage. Our procedure consists of (a) converting the

1-4244-2334-7/08/$20.00

MDCT/IMDCT computation to a DCT of type IV (using additions and subtractions only), (b) decomposing the DCT kernel into cyclic convolutions and Hankel matrix products, and finally, (c) employing bilinear algorithms for each of the convolutions and Hankel matrix products to obtain the required bilinear algorithm for the MDCT. We illustrate each of these steps in this section.

A. Conversion of MDCT into DCT

To obtain MDCT using a DCT, introduce a new data sequence

$$y(i) = \begin{cases} -x(i+3N/4), & \text{if } 0 \le i < N/4, \\ x(i-N/4), & \text{if } N/4 \le i < N. \end{cases}$$

Then defining

$$z(i) = y(i) - y(N-1-i), \quad 0 \le i < N/2,$$

an N point MDCT can be expressed as an $N/2$ point DCT-IV as

$$X(k) = \sum_{i=0}^{N/2-1} z(i) \cos\left(\frac{\pi(2i+1)(2k+1)}{2N}\right), \quad 0 \le k < N/2.$$

Thus the MDCT computation of short and long blocks is transformed into DCTs of 6 and 18 points respectively.

B. Conversion of IMDCT into DCT

To obtain the IMDCT, we first compute the $N/2$ point type-IV DCT of X as

$$z(i) = \frac{2}{N} \sum_{i=0}^{N/2-1} X(k) \cos\left(\frac{\pi(2i+1)(2k+1)}{2N}\right), \quad 0 \le i < N/2.$$

Then defining a new data sequence

$$y(i) = \begin{cases} z(i), & \text{if } 0 \le i \le N/2-1, \\ -z(N-1-i), & \text{if } N/2 \le i < N, \end{cases}$$

One can recover the signal sequence as

$$x(i) = \begin{cases} y(i+N/4), & \text{if } 0 \le i < 3N/4, \\ -y(i-3N/4), & \text{if } 3N/4 \le i < N. \end{cases}$$

Thus the IMDCT computation of short and long blocks is also transformed into DCTs of 6 and 18 points respectively.

C. DCT algorithm for MP3 short block

Recall that a DCT-IV of an N point sequence $\{x(i)\}$ is given by

$$X(k) = \sum_{i=0}^{N-1} x(i) \cos(\frac{\pi(2i+1)(2k+1)}{4N}), \ 0 \le k < N. \quad (1)$$

For MP3 short block, the DCT length $N = 6$. To compute (1) for this N,we partition the signal and transform indices in two sets: set $S_1 = \{1, 4\}$ is made up of those i's for which $(2i+1)$ is a multiple of 3 and set $S_2 = \{0, 2, 3, 5\}$, of the rest. We compute those transform components together whose indices are in the same set. Further, the summation in (1) is carried out over the two signal index sets separately. We will denote the computation of $X(k)$ with signal component indices restricted to sets S_1 and S_2 by $X_1(k)$ and $X_2(k)$ respectively. Clearly, $X(k) = X_1(k) + X_2(k)$.

Let p denote the DCT kernel element $\cos(\pi p/4N)$ and $\overline{p}$, the element $-\cos(\pi p/4N)$. Then from (1), the transform components $X_1(k)$, $k \in S_1$ are given by:

$$\begin{bmatrix} X_1(1) \\ X_1(4) \end{bmatrix} = \begin{bmatrix} 9 & \overline{3} \\ \overline{3} & \overline{9} \end{bmatrix} \begin{bmatrix} x(1) \\ x(4) \end{bmatrix}. \quad (2)$$

Since the constant matrix in (2) is a Hankel matrix, one can use a bilinear algorithm to obtain this product.

Similarly, using the same shorthand notation, transform components $X_2(k)$, $k \in S_1$ are given by:

$$\begin{bmatrix} X_2(1) \\ X_2(4) \end{bmatrix} = \begin{bmatrix} 3 & 9 \\ 9 & \overline{3} \end{bmatrix} \begin{bmatrix} x(0) - x(3) \\ -x(5) - x(2) \end{bmatrix}. \quad (3)$$

Note that (3) exploits the fact that some kernel matrix elements are equal in magnitude and therefore the signal sequence may be folded to reduce the number of multiplications. Since the constant matrix in (3) is a Hankel matrix, one can once again use a bilinear algorithm to obtain this product.

When both the signal and transform indices are restricted to set S_2, the computation can be transformed into a multidimensional convolution. To do this, observe that indices in S_2 are elements of $A(8N)$, a group formed by non-negative integers less than and relatively prime to $8N$ under the operation of multiplication modulo $8N$. The structure of this group can be used to order the indices in S_2 as well as to define a sign function. By using the order $\{0, 5, 3, 2\}$ suggested by $A(8N)$ and the corresponding sign function $\{1, -1, 1, 1\}$, one can express $X_2(k)$, $k \in S_2$ as:

$$\left[\begin{array}{c} X_2(0) \\ -X_2(5) \\ \hline X_2(3) \\ X_2(2) \end{array}\right] = \left[\begin{array}{cc|cc} 1 & \overline{11} & 7 & 5 \\ \overline{11} & 1 & 5 & \overline{7} \\ \hline 7 & 5 & 1 & \overline{11} \\ 5 & \overline{7} & \overline{11} & 1 \end{array}\right] \left[\begin{array}{c} x(0) \\ -x(5) \\ \hline x(3) \\ x(2) \end{array}\right]. \quad (4)$$

The structure of the matrix in (4) is predicted by the structure of the group $A(8N)$. By partitioning the matrix in (4) as shown, one can see that it is a block cyclic matrix with each block being a Hankel matrix. Thus this computation corresponds to a two dimensional convolution with a 2 point cyclic convolution along one dimension and a 2 point Hankel product along the other. Again, appropriate bilinear algorithms for these small lengths can be combined to obtain the bilinear algorithm for (4).

Finally, when the signal indices are restricted to the set S_1, one the computation of $X_2(k)$, $k \in S_2$ gives:

$$\begin{bmatrix} X_1(0) \\ X_1(5) \\ X_1(3) \\ X_1(2) \end{bmatrix} = \begin{bmatrix} 3 & 9 \\ \overline{9} & 3 \\ \overline{3} & \overline{9} \\ \overline{9} & 3 \end{bmatrix} \begin{bmatrix} x(1) \\ x(4) \end{bmatrix}. \quad (5)$$

By comparing (5) with (2) one can see that

$$\begin{bmatrix} X_1(0) \\ X_1(5) \\ X_1(3) \\ X_1(2) \end{bmatrix} = \begin{bmatrix} -X_1(4) \\ -X_1(1) \\ X_1(4) \\ -X_1(1) \end{bmatrix}. \quad (6)$$

Equation (5) shows that $X_1(k)$, $k \in S_2$ need not be computed separately.

As a last simplification, note that because of the following trigonometric identities for $N = 6$,

$$\begin{aligned} \cos(9\pi/4N) &= \cos(\pi/4N) - \cos(7\pi/4N), \\ -\cos(3\pi/4N) &= -\cos(11\pi/4N) - \cos(5\pi/4N), \end{aligned} \quad (7)$$

one gets

$$\begin{aligned} X_2(1) &= -2X_2(2), \\ X_2(4) &= 2X_2(3). \end{aligned} \quad (8)$$

Therefore computation for (3) can be absorbed into the that for (4). The operation of multiplication by 2 may be counted as one addition. However, frequently in hardware design, this scaling by 2 can be realized as a trivial left shift with negligible impact on area and speed of the implementation.

The complete bilinear algorithm for 6 point DCT is shown in Fig. 1. The multiplication constants used in Fig. 1 have the

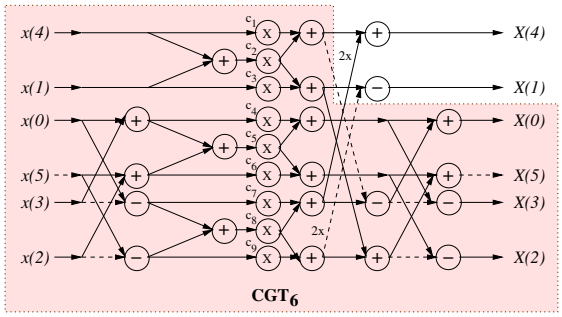

Fig. 1. Proposed bilinear implementation of the 6 point DCT-IV.

following values: $c_1 = 0.5412$, $c_2 = -0.9239$, $c_3 = 1.3066$, $c_4 = 0.4687$, $c_5 = 0.3314$, $c_6 = -1.1315$, $c_7 = 0.6533$, $c_8 = -0.4619$ and $c_9 = 0.2706$.

The arithmetic complexity and hardware delay of vaious MDCT/IMDCT algorithms are compared in Table I. One can see that the new proposed algorithm has about 20% less multiplications than other algorithms currently available. Because of the bilinear structure, we have only one multiplication on the critical path which results into a very fast implementation. Figures 2 and 3 show a comparison of various fixed point implementations in TSMC 90nm technology. It is evident from this figure that the proposed bilinear algorithm provides the smallest and fastest MDCT/IMDCT implementation for MP3 audio short block.

D. DCT algorithm for MP3 long block

We now extend the procedure of Subsection II-C to length $N = 18$ which is used in MP3 audio long block. As before, we

TABLE I

COMPLEXITIES AND CRITICAL DELAYS OF VARIOUS 12 POINT MDCT AND IMDCT ALGORITHMS. NOTE THAT **M** AND **A** REFER TO MULTIPLICATION AND ADDITION RESPECTIVELY.

Algorithm		complexity	Critical delay
MDCT	Proposed	$9M + 29A$†	$M + 6A$
	Ref. [5]	$13M + 39A$	$2M + 6A$
	Ref. [6]	$13M + 27A$	$2M + 5A$
	Ref. [7]	$11M + 29A$	$3M + 7A$
IMDCT	Proposed	$9M + 23A$†	$M + 5A$
	Ref. [5]	$13M + 33A$	$2M + 5A$
	Ref. [6]	$13M + 21A$	$2M + 4A$
	Ref. [7]	$11M + 23A$	$3M + 6A$

† Two of these additions can be converted to shifts.

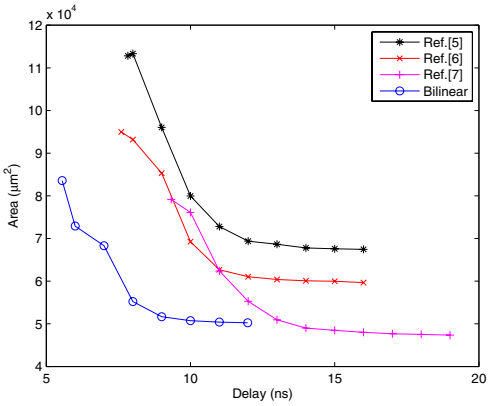

Fig. 2. Delay and area for various implementations of 12 point MDCT for MP3 short block.

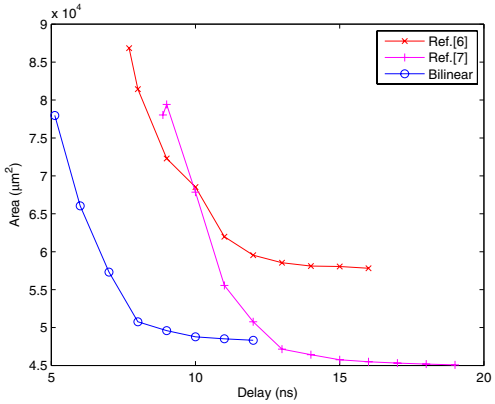

Fig. 3. Delay and area for various implementations of 12 point IMDCT for MP3 short block.

partition the transform components into two sets and compute each set of components together. The partition of transform indices is based on the set of integers $A(8N) = \{0 \leq i < 8N \,|\, \gcd(i, 8N) = 1\}$, i.e., a set of positive integers less than $8N$ which are relatively prime to $8N$. The transform components $X(k)$ with $(2k+1) \notin A(8N)$, can be directly obtained from a DCT of a 6 point sequence $y(i)$ obtained by folding of the input as

$$y(i) = x(i) - x(11-i) - x(12+i), \quad 0 \leq i < 6.$$

On the other hand, the transform components $X(k)$ with $(2k+1) \in A(8N)$ require a computation that we call the N point *Cosine Group Transform*, CGT_N. This overall structure of the computation for $N = 18$ is shown in Fig. 4.

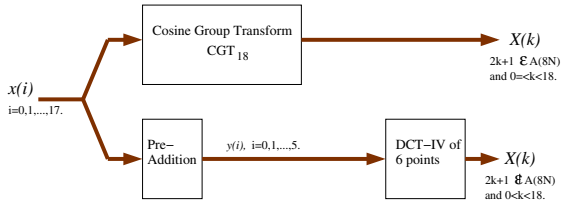

Fig. 4. Flow graph for 18 point bilinear DCT-IV.

The *Cosine Group Transform*, CGT_N, can itself be computed recursively by dividing the signal components into sets using the group $A(N)$. In particular, the portion of the CGT_N that uses signal components $x(i)$, with $i \notin A(N)$ forms $CGT_{N/3}$ while the portion using $x(i)$, with $i \in A(N)$ can be structured as a multidimensional cyclic convolution. The output of this convolution may be added to the output of $CGT_{N/3}$ to obtain the output of CGT_N. The decomposition of CGT_N is summarized in Fig. 5.

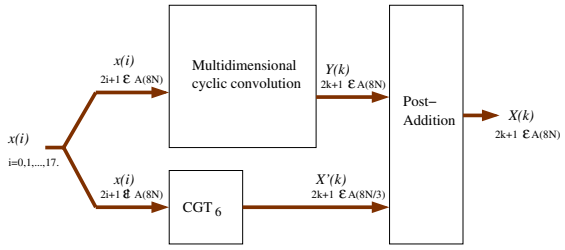

Fig. 5. Flow graph for cosine group transform CGT_{18}.

The computation of 18 point DCT (hence 36 point MDCT/ IMDCT) for the MP3 long block requires one to obtain CGT_{18} and a 6 point DCT. From Fig. 5, CGT_{18} computation in turn involves that of a CGT_6 and a large computational block (multidimensional convolution). We already have the 6 point bilinear DCT algorithm from Subsection II-C. Further, the CGT_6 is also a part of the 6 point DCT as indicated in Fig. 1. Thus we only need to obtain the additional algorithm for the multidimensional convolution in Fig. 5.

As in the previous case, this block can be converted into a multi-dimensional convolution with the use of the group $A(8N)$ where $N = 18$. By permuting the inputs and outputs as per the index sequence $\{0, 9, 11, 2, 12, 14, 8, 17, 15, 6, 3, 5\}$ and using the sign function values $\{1, 1, -1, -1, -1, 1, 1, 1, 1, -1, -1\}$, one can translate the computation to a three-dimensional convolution with a 2 point cyclic convolution, a 3 point cyclic convolution and a 2 point Hankel matrix multiplication along the three dimensions. The flow-chart of this multi-dimensional convolution is shown in Fig. 2. Note that both the index permutation and the sign function sequence are obtained systematically from the structure of the group $A(8N)$. The details are omitted for brevity.

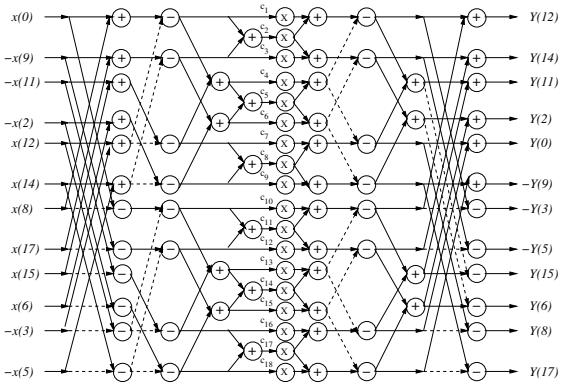

Fig. 6. Proposed bilinear implementation of multidimensional convolution involved in the 18 point DCT-IV.

The multiplication constants used in Fig. 6 are as follows: $c_1 = -0.9231$, $c_2 = -0.6528$, $c_3 = 2.2287$, $c_4 = -0.5086$, $c_5 = -0.3596$, $c_6 = 1.2278$, $c_7 = -0.6025$, $c_8 = -0.4261$, $c_9 = 1.4546$, $c_{10} = -0.1628$, $c_{11} = 0.2779$, $c_{12} = -0.3930$, $c_{13} = 0.1851$, $c_{14} = -0.3160$, $c_{15} = 0.4469$, $c_{16} = 0.7181$, $c_{17} = -1.2258$ and $c_{18} = 1.7336$.

The arithmetic complexity and hardware delay of vaious MDCT/IMDCT algorithms are compared in Table II. The proposed bilinear MDCT/IMDCT algorithm has the lowest multiplication requirement for MP3 audio short block. For 128Kbps MP3 audio stream, our algorithms represent a saving

TABLE II

COMPLEXITIES AND CRITICAL DELAYS OF VARIOUS 36 POINT MDCT AND IMDCT ALGORITHMS. NOTE THAT **M** AND **A** REFER TO MULTIPLICATION AND ADDITION RESPECTIVELY.

Algorithm		Complexity	Critical delay
MDCT	Proposed	$36M + 150A$†	$M + 10A$
	Ref. [5]	$47M + 165A$	$2M + 9A$
	Ref. [6]	$47M + 129A$	$2M + 8A$
	Ref. [7]	$43M + 133A$	$3M + 22A$
IMDCT	Proposed	$36M + 132A$†	$M + 9A$
	Ref. [5]	$51M + 151A$	$2M + 8A$
	Ref. [6]	$51M + 115A$	$2M + 7A$
	Ref. [7]	$43M + 115A$	$3M + 21A$

† Two of these additions can be converted to shifts.

of nearly 1800 multiplications per second. This can instantly translate into significant power savings for MP3 players, now mostly battery operated. Moreover, because the proposed algorithm is bilinear, a directly mapped hardware will have fastest implementation due to the minimization of multiplications on the critical path.

III. Unified architecture

The computation of MDCT/IMDCT of MP3 long block requires one to design for a multi-dimensional convolution, a CGT_6 and a 6 point DCT. Note that both the CGT_6 and the 6 point DCT are used in the computation of MDCT/IMDCT of MP3 short block. Therefore without introducing any extra computations, we can integrate the computation of MP3 short block with that of the long block. Three different unified MDCT/IMDCT architectures are proposed, providing the processing power of one long or one short block (architecture A), one long or two short blocks (architecture B), or half long or one short block (folding, architecture C). It is worth emphasizing that the ability to process both block sizes is of central importance to a low cost and low power implementation of MP3 audio player.

We implemented our algorithms and those available in the literature using fixed point arithmetic in TSMC 90nm technology. The area and delay comparisons are shown in Fig. 2. We compare our unified architecture performance with that of other architectures useful for processing long block audio only. One can see that using the same technology, our unified architectures are still 25% smaller and 25% faster than any previous architectures designed for a single frame size.

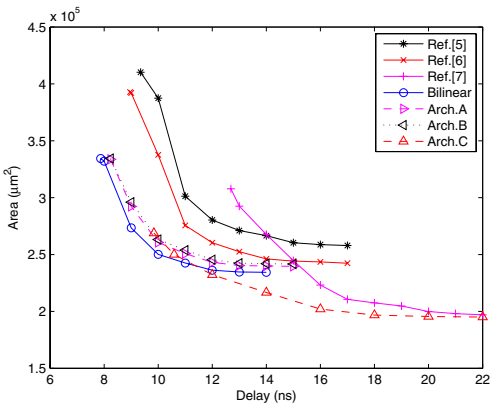

Fig. 7. Delay and area for unified 12 and 36 point MDCT and IMDCT architectures (A, B and C) for MP3 audio, with comparison to the 36 point MDCT architectures in literature.

IV. Conclusion

The modified discrete cosine transform (MDCT) and its inverse (IMDCT) are amongst the two most compute intensive operations in MP3 audio encoders and decoders. This paper has developed efficient bilinear algorithms and architectures for computing the MDCT and IMDCT of short and long MP3 blocks. Our algorithms have less multiplications than any other algorithm in literature. In mobile MP3 players this may imply a sizable power saving.

The real advantage of our algorithm is seen when it is cast in Silicon. Because our algorithm is bilinear, it has only one multiplication on the critical path. All other algorithms have at least two multiplications on the critical path. This implies a much faster MP3 encoder and decoder architecture allowing much more flexibility in trading speed for area and power of the hardware accelerator.

The structure of our algorithm allows us to create unified architectures to process both the long and short MP3 blocks. This ability to process both block sizes is crucial to MP3 encoders and decoders because both types of blocks generally abound in the same data stream. As compared with the implementations (in the same technology) of other algorithms that process only a single long block, our unified architectures processing both size blocks are still 25% smaller and 25% faster.

References

[1] J. Princen and A. Bradley, "Analysis/synthesis filter bank design based on time domain aliasing cancellation," *IEEE Trans. Acoust. Speech Signal Processing*, vol. ASSP-34, pp. 1153–1161, Oct. 1986.

[2] T. Tsai, T. Chen, and L. Chen, "An MPEG audio decoder chip," *IEEE Trans. Consumer Electronics*, vol. 41, pp. 89–96, Feb. 1995.

[3] S. Tai, C. Wang, and C. Lin, "FFT and IMDCT circuit sharing in DAB receiver," *IEEE Trans. Broadcasting*, vol. 49, pp. 124–131, June 2003.

[4] Y. Yao, Q. Yao, P. Liu, and Z. Xiao, "Embedded software optimization for MP3 decoder implemented on RISC core," *IEEE Trans. Consumer Electronics*, vol. 50, pp. 1244–1249, June 2005.

[5] V. Britanak and K. Rao, "An efficient implementation of the forward and inverse MDCT in MPEG audio coding," *IEEE Signal Processing Letters*, vol. 8, pp. 48–50, Feb. 2001.

[6] V. Nikolajevic and G. Fettweis, "Improved implementation of MDCT in MP3 audio coding," *10th Asia-Pacific Conf. Comm. and 5th Intern. Symp. Multi-Dimen. Mobile Comm.*, vol. 1, pp. 309–312, Aug. 2004.

[7] S. Lee, "Improved algorithm for efficient computation of the forward and inverse MDCT in MPEG audio coding," *IEEE Trans. Circuits Syst. II*, vol. 48, pp. 990–994, Oct. 2001.