Fast Algorithms for Low-Delay SBR Filterbanks in MPEG-4 AAC-ELD

Ravi K. Chivukula, Yuriy A. Reznik, *Senior Member, IEEE*, Venkat Devarajan, *Senior Member, IEEE*, and Mythreya Jayendra-Lakshman

Abstract

The MPEG committee has recently finished development of a new audio coding standard "MPEG-4 Advanced Audio Coding - Enhanced Low Delay" (AAC-ELD). AAC-ELD is targeted towards high quality, full-duplex communication applications such as audio and video conferencing. AAC-ELD uses Spectral Band Replication (SBR) technology together with a low delay AAC core encoder to achieve high coding efficiency and low algorithmic delays. In this paper, we present fast algorithms for computation of the low delay SBR filterbanks in AAC-ELD. The proposed fast algorithms are derived by establishing a connection between SBR filterbanks and the Discrete Cosine Transform of type IV (DCT-IV). The proposed techniques are of particular convenience for implementations, that already employ DCT-IV in the design of AAC-core filterbanks. Our presentation includes detailed explanation and flow-graphs of the proposed algorithms, complexity analysis, and comparisons with alternative implementations.

Index Terms

Low delay audio coding, AAC, SBR, MPEG, DCT, DCT-IV, filterbank, factorization, fast algorithms.

I. Introduction

Traditionally, speech and audio coding paradigms evolved in different directions. Speech coding is based on source modeling [1]. The vocal tract is modeled as a time-varying digital filter, and speech production is modeled as an excitation of such filter by a periodic impulse train. Speech codecs perform very well for speech-only / single-speaker material, operating at bit rates as low as 4-8 Kbps, and with round-trip algorithmic delays as low as 20ms. Such low algorithmic delays make speech codecs suitable for full-duplex communication scenarios.

On the other hand, audio coding is based on modeling the psychoacoustic characteristics of the human auditory system [2]. The audio signal is split into several frequency bands and the masking properties of the human auditory system are used to remove perceptually irrelevant (inaudible) parts of the signal. Most commonly, audio codecs use transform-domain coding for compression. Such codecs are suitable for coding of a broad variety of audio

R. K. Chivukula and Y. A. Reznik are with Qualcomm Inc., San Diego, CA, 92121 USA emails: {rchivuku, yreznik}@qualcomm.com.

V. Devarajan and M. Jayendra-Lakshman are with the Department of Electrical Engineering, The University of Texas at Arlington, Arlington, TX, 76019 USA e-mails: venkat@uta.edu, mythreya.jayendralakshman@mavs.uta.edu.

Manuscript received August 30, 2010.

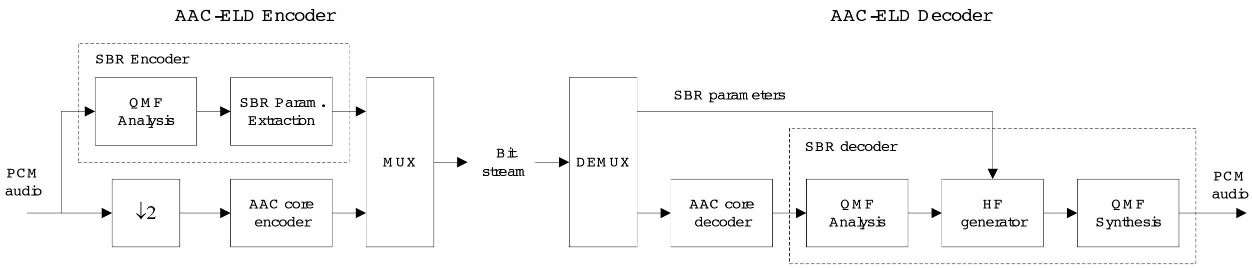

Fig. 1. Structure of AAC-ELD encoder and decoder.

material, but their algorithmic delays are usually much larger, since long transforms are needed for good frequency selectivity. This makes them unsuitable for full-duplex communications.

However, the need for high-quality, low bit rate, full-duplex audio communication applications is constantly growing [3]. MPEG first attempted to address this need by developing a modification of the MPEG-4 Advanced Audio Codec (AAC), called AAC - Low Delay (AAC-LD) [5]. The reduction in delay is obtained by reducing the frame length, avoiding block switching and minimizing the use of bit reservoir in the encoder. While the delay was reduced to 20ms, the coding bitrates of 64kbps and higher were needed to achieve high quality audio. Recently, MPEG has revisited this topic, and developed a new standard, MPEG-4 AAC-Enhanced Low Delay (AAC-ELD) [4]. Coding efficiency of AAC-ELD is increased by operating the core AAC coder at half the original sampling rate, and using the Spectral Band Replication (SBR) tool to synthesize the upper part of the spectrum [22]. A short parametric description is used to control such synthesis. We show overall AAC-ELD processing diagram in Figure I. To minimize algorithmic delays, special low-delay versions of analysis and synthesis SBR filterbanks (LD-SBR) are defined. Further, a fixed time grid is employed. Delay is also reduced by using a modified AAC core filterbank [6]. With all these improvements incorporated, AAC-ELD achieves algorithmic delay of 31.3ms while operating at bit rates as low as 32kbps per channel [3]. This makes AAC-ELD suitable for many low-bitrate, full-duplex communication applications.

Most commonly, low-delay speech and audio codecs are used in mobile devices, where processing power and battery life are limited. In such environments, particular attention is paid to reducing the complexity of the codec, and all component algorithms. From studies of earlier MPEG audio codecs, such as High-Efficiency AAC (HE-AAC), it is known that the SBR tool may contribute between 50 and 75% to the total computational complexity of the decoder [7]. This is why HE-AAC, as well as AAC-ELD standards define two types of SBR filterbanks for the decoder: complex-domain filterbanks to be used to produce High-Quality (HQ) reconstruction, and real-domain filterbanks to be used in Low Power (LP) environments. However, even when real-domain SBR is used, its contribution to the overall complexity is significant [7]. Additional efforts towards reducing SBR complexity are therefore much needed and appreciated in practice.

In this paper, we propose fast algorithms for computing low delay SBR analysis and synthesis filterbanks in AAC-ELD. Our algorithms are derived by establishing a mapping between SBR matrix operation in AAC-ELD and

Discrete Cosine Transform of type-IV (DCT-IV). We then further isolate leading factors in DCT-IV by converting it to type-II DCT (DCT-II), and merge these factors with factors in a window, applied to a signal prior to computing SBR matrix product. This way we achieve additional computational savings. Overall, our proposed algorithms are 15 times simpler than conventional implementation of such filterbanks.

Among related prior work, we must first mention the optimization framework of G.Schuller and T. Smith [10], which was used to produce low-delay filterbanks in AAC-ELD [7]. This framework ensures that resulting filterbanks have cosine modulation functions. Similar ideas for filterbank design were also suggested by T. Nguyen and R. Koilpillai [11], and M. Harteneck, S. Weiss, and R. W. Stewart [12]. However, while the underlying framework [10] assures the existence of mappings to DCT-type transforms, the derivation of such mappings for a given final filterbank design is non-trivial. The technique that we've used to derive our fast algorithms for SBR filterbanks in AAC-ELD is motivated by an approach of K. Konstantinides [13], who has shown how to map filterbanks in MPEG-1 audio [14] to DCT-II. S.-W. Huang and T.-H. Tsai [20] also used Konstatinides' technique to derive fast algorithms for SBR filterbanks in HE-AAC [5]. However, because of an odd-number phase shift used in the low-delay SBR filterbanks in AAC-ELD, our mappings turns LD-SBR into DCT-IV instead of DCT-II. H.-W. Hsu *et al.* [21] offer a mapping of complex SBR filterbanks in HE-AAC to DCT-IV, but they don't exploit the possibility of eliminating multiplications by moving some factors to the window function. Many existing fast algorithms for computing DCT-II and DCT-IV can be found in books [15], [16].

This paper is organized as follows. In Section II, we provide definitions of LD-SBR filterbanks used in the encoder and decoder. In Sections III and IV, we present the fast algorithms for complex analysis and synthesis filterbanks. Derivations of fast algorithms for real-domain versions of these filterbanks are very similar. We bring a summary of all relevant results in Section V. An analysis of the computational complexity is provided in Section VI and the conclusions are given in Section VII.

II. Definitions

Hereafter, we use the following notation. Time domain sequences are denoted by small letters such as $x(n)$. Frequency domain sequences are denoted by capital letters such as $X(k)$. Vectors are denoted by bold-face small letters such as $\mathbf{x}$, matrices are denoted by bold-face capital letters such as $\mathbf{D}$. An element at l-th row and m-th column of a matrix $\mathbf{D}$ is denoted by $\mathbf{D}(l, m)$. The row and column indices start from 0. Symbol j denotes the imaginary unit: $j = \sqrt{-1}$. Symbols $\mathbb{R}$ and $\mathbb{C}$ denote the set of real numbers and the set of complex numbers respectively. Operators $\mathrm{Re}[.]$ and $\mathrm{Im}[.]$ return the real and imaginary parts, respectively, of a complex number.

A. SBR filterbanks in AAC-ELD

The SBR tool uses near-perfect-reconstruction pseudo QMF filterbanks (see e.g. [18]). A linear phase low pass prototype filter is designed such that the stop band attenuation is minimized. The impulse responses of the analysis and synthesis filters are cosine modulations of the prototype filter. Let $h(n)$ be the prototype filter of length N. Let

M be the number of subbands. Then, in the general form, the subband filter impulse responses are given by:

$$h_k(n) = h(n) \exp\left\{\frac{j\pi\left(k+\frac{1}{2}\right)\left(n-\frac{N-1}{2}\right)}{M}\right\},\ 0 \le k < M, 0 \le n < N,$$
$$f_k(n) = h(n) \exp\left\{\frac{j\pi\left(k+\frac{1}{2}\right)\left(n-\frac{N-1}{2}\right)}{M}\right\},\ 0 \le k < M, 0 \le n < N,$$

where f_k and h_k denote responses of analysis and synthesis filters, respectively.

AAC-ELD employs a particular design of analysis QMF in the encoder and the decoder with the following complex matrix operation [4]:

$$X(k) = \sum_{n=0}^{N-1} x(n) \exp\left\{\frac{j\pi\left(k+\frac{1}{2}\right)(2n-n_a)}{N}\right\},\ 0 \le k < \frac{N}{2}. \tag{1}$$

where, N is $1/5$-th of the filter length, and

$$n_a = \frac{3N}{2} - 1. \tag{2}$$

The synthesis QMF (SQMF) and down-sampled SQMF of LD-SBR in AAC-ELD decoder use the following matrix operations [4]:

$$\tilde{x}(n) = \mathrm{Re}\left[\sum_{k=0}^{\frac{N}{2}-1} \tilde{X}(k) \exp\left\{\frac{j\pi\left(k+\frac{1}{2}\right)(2n-n_s)}{N}\right\}\right],\ 0 \le n < N. \tag{3}$$

where

$$n_s = \frac{N}{2} - 1, \tag{4}$$

$x(n), \tilde{x}(n) \in \mathbb{R};\ X(k), \tilde{X}(k) \in \mathbb{C}$.

For applications that require low power, real analysis and synthesis filterbanks are also defined for the AAC-ELD decoder. They are defined as follows [4]:

$$X_R(k) = \sum_{n=0}^{N-1} x(n) \cos\left\{\frac{\pi\left(k+\frac{1}{2}\right)(2n-n_a)}{N}\right\},\ 0 \le k < \frac{N}{2}, \tag{5}$$

and

$$\tilde{x}_R(n) = \sum_{k=0}^{\frac{N}{2}-1} \tilde{X}_R(k) \cos\left\{\frac{\pi\left(k+\frac{1}{2}\right)(2n-n_s)}{N}\right\},\ 0 \le n < N. \tag{6}$$

where $x(n), \tilde{x}_R(n), X_R(k), \tilde{X}_R(k) \in \mathbb{R}$.

B. Cosine and Sine Transforms

In order to accelerate computation of SBR filterbanks, we map them to several standard transforms allowing fast computation [15]. The Discrete Cosine Transforms of types II - IV over a real sequence $d(n)$ of length N are

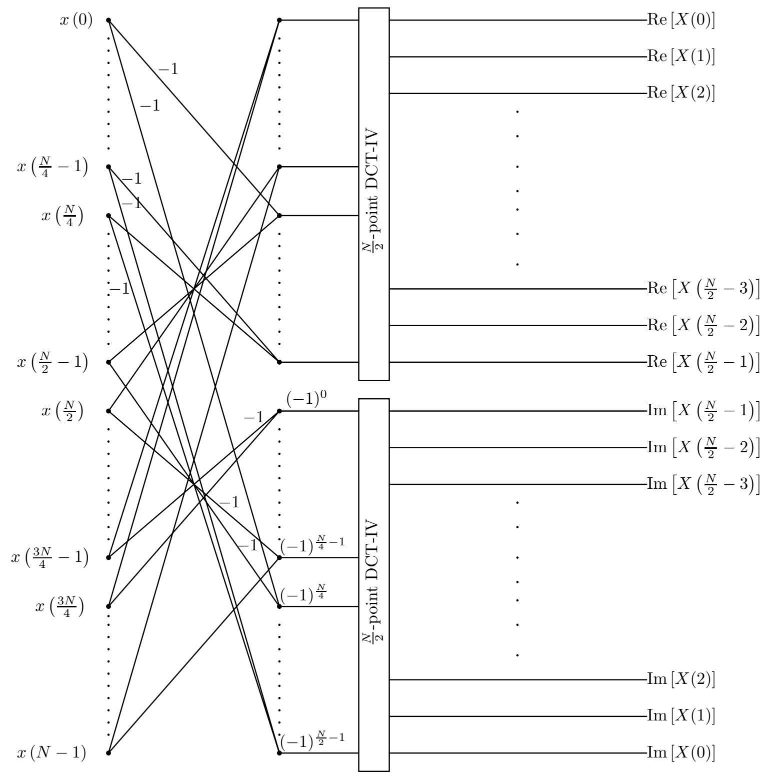

Fig. 2. Flowgraph of the proposed DCT-IV-based algorithm for computing AQMF.

defined as follows $(0 \leq k < N)$:

$$D_{C2}(k) = \sum_{n=0}^{N-1} d(n) \cos\left\{\frac{\pi(2n+1)k}{2N}\right\}, \quad (7)$$

$$D_{C3}(k) = \sum_{n=0}^{N-1} d(n) \cos\left\{\frac{\pi(2k+1)n}{2N}\right\}, \quad (8)$$

$$D_{C4}(k) = \sum_{n=0}^{N-1} d(n) \cos\left\{\frac{\pi(2k+1)(2n+1)}{4N}\right\}. \quad (9)$$

For simplicity, we omit standard normalization factors in all these definitions [15].

Discrete Sine Transform of type IV (DST-IV), $D_{S4}(k)$, over a sequence $d(n)$ is defined as follows [15]:

$$D_{S4}(k) = \sum_{n=0}^{N-1} d(n) \sin\left\{\frac{\pi(2k+1)(2n+1)}{4N}\right\}, \; 0 \leq k < N. \quad (10)$$

It is known that DST-IV is connected to DCT-IV by means of sign changes and reversals of the input and output sequences [15]:

$$D_{S4}(N-1-k) = \sum_{n=0}^{N-1} (-1)^n d(n) \cos\left\{\frac{\pi(2k+1)(2n+1)}{4N}\right\}, \; 0 \leq k < N. \quad (11)$$

III. Fast Algorithm for Complex AQMF

We first the present mapping of complex-domain AQMF (1) to DCT-IV.

Theorem 1: The AAC-ELD Complex AQMF (1) can be computed as follows:

$$X(k) = X_1(k) + jX_2\left(\frac{N}{2} - 1 - k\right),\ 0 \le k < \frac{N}{2} \tag{12}$$

where

$$X_1(k) = \sum_{n=0}^{\frac{N}{2}-1} x_1(n) \cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\},\ 0 \le k < \frac{N}{2} \tag{13}$$

$$X_2(k) = \sum_{n=0}^{\frac{N}{2}-1} (-1)^n x_2(n) \cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\},\ 0 \le k < \frac{N}{2}. \tag{14}$$

are DCT-IV transforms over the following intermediate sequences:

$$x_1(n) = \begin{cases} x\left(n + \frac{3N}{4}\right) + x\left(\frac{3N}{4} - 1 - n\right) & \text{for} \quad 0 \le n < \frac{N}{4} \\ x\left(\frac{3N}{4} - 1 - n\right) - x\left(n - \frac{N}{4}\right) & \text{for} \quad \frac{N}{4} \le n < \frac{N}{2}. \end{cases} \tag{15}$$

and

$$x_2(n) \triangleq \begin{cases} x(n + \frac{3N}{4}) - x(\frac{3N}{4} - 1 - n) & \text{for} \quad 0 \le n < \frac{N}{4} \\ -x(\frac{3N}{4} - 1 - n) - x(n - \frac{N}{4}) & \text{for} \quad \frac{N}{4} \le n < \frac{N}{2}. \end{cases} \tag{16}$$

The proof of this statement can be found in Appendix A. It can be observed that only additions, sign changes, and reordering operations are needed to reduce this filterbank to computing 2 $N/2$-point DCT-IV transforms. We show the flowgraph of this process in Figure III.

A. Mapping to DCT-II and Further Optimization

In this section, we show how to map the aforementioned DCT-IV-based fast algorithm to the DCT-II. We also show how some of the multiplications can be absorbed into the windowing operation that precedes the AQMF.

We start with the matrix formulation of the algorithm from Theorem 1. Define the following vectors:

$$\mathbf{x_1} = \left[x_1(0) \quad \dots \quad x_1\left(\tfrac{N}{2} - 1\right)\right]^T, \quad \mathbf{x_2} = \left[x_2(0) \quad \dots \quad x_2\left(\tfrac{N}{2} - 1\right)\right]^T,$$
$$\mathbf{x} = \left[x(0) \quad \dots \quad x\left(N - 1\right)\right]^T, \qquad \mathbf{X} = \left[X(0) \quad \dots \quad X\left(\tfrac{N}{2} - 1\right)\right]^T.$$

Let $\mathbf{B_N^I}$ and $\mathbf{B_N^{II}}$ be $\frac{N}{2} \times N$ matrices that split $\mathbf{x}$ into $\mathbf{x_1}$ and $\mathbf{x_2}$ respectively (cf. (15) and (16)):

$$\mathbf{B_N^I}\left(n, n+\frac{3N}{4}\right) = 1 \qquad \text{for } 0 \le n < \frac{N}{4} \tag{17}$$

$$\mathbf{B_N^I}\left(n, \frac{3N}{4}-1-n\right) = 1 \qquad \text{for } 0 \le n < \frac{N}{2} \tag{18}$$

$$\mathbf{B_N^I}\left(n, n-\frac{N}{4}\right) = -1 \qquad \text{for } \frac{N}{4} \le n < \frac{N}{2} \tag{19}$$

$$\mathbf{B_N^I}(n,k) = 0 \qquad \text{for all other combinations of } n \text{ and } k \tag{20}$$

$$\mathbf{B_N^{II}}\left(n, n+\frac{3N}{4}\right) = 1 \qquad \text{for } 0 \le n < \frac{N}{4} \tag{21}$$

$$\mathbf{B_N^{II}}\left(n, \frac{3N}{4}-1-n\right) = -1 \qquad \text{for } 0 \le n < \frac{N}{2} \tag{22}$$

$$\mathbf{B_N^{II}}\left(n, n-\frac{N}{4}\right) = -1 \qquad \text{for } \frac{N}{4} \le n < \frac{N}{2} \tag{23}$$

$$\mathbf{B_N^{II}}(n,k) = 0 \qquad \text{for all other combinations of } n \text{ and } k \tag{24}$$

Let $\mathbf{C^{IV}_{\frac{N}{2}}}$ be the $\frac{N}{2} \times \frac{N}{2}$ DCT-IV transform matrix:

$$\mathbf{C^{IV}_{\frac{N}{2}}}(k,n) = \cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\}, \qquad 0 \le n < \frac{N}{2}, 0 \le k < \frac{N}{2}, \tag{25}$$

$\mathbf{A_{\frac{N}{2}}}$ denote an $\frac{N}{2} \times \frac{N}{2}$ diagonal matrix that inverts signs of odd-indexed elements:

$$\mathbf{A_{\frac{N}{2}}}(n,n) = (-1)^n, \qquad 0 \le n < \frac{N}{2}, \tag{26}$$

and $\mathbf{J_{\frac{N}{2}}}$ denote an $\frac{N}{2} \times \frac{N}{2}$ order reversal matrix

$$\mathbf{J_{\frac{N}{2}}}\left(n, \frac{N}{2}-1-n\right) = 1, \qquad 0 \le n < \frac{N}{2}, \tag{27}$$

$$\mathbf{J_{\frac{N}{2}}}(n,k) = 0, \qquad \text{for all other combinations of } n, k$$

respectively.

Using the above notation, our algorithm (12) can be formulated as

$$\mathbf{X} = \left(\mathbf{C^{IV}_{\frac{N}{2}}} \mathbf{B_N^I} + j \mathbf{J_{\frac{N}{2}}} \mathbf{C^{IV}_{\frac{N}{2}}} \mathbf{A_{\frac{N}{2}}} \mathbf{B_N^{II}}\right) \mathbf{x}. \tag{28}$$

We now note that vector $\mathbf{x}$ in SBR matrix operation is usually derived from input data sequence

$$\mathbf{u} = \begin{bmatrix} u(0) & \dots & u(5N-1) \end{bmatrix}^T$$

using the following operation (cf. [4], [5]):

$$\mathbf{x} = \mathbf{S_N} \mathbf{C_{5N}}\, \mathbf{u}, \tag{29}$$

where, $\mathbf{C_{5N}}$ is a $5N \times 5N$ diagonal matrix of constants (window factors) and $\mathbf{S_N}$ is a $N \times 5N$ overlap-add matrix

$$\mathbf{S_N}(n, n+Nk) = 1, \qquad 0 \le n < N;\ 0 \le k < 5 \tag{30}$$

$$\mathbf{S_N}(n,k) = 0, \qquad \text{for all other combinations of } 0 \le n < N;\ 0 \le k < 5N. \tag{31}$$

From [19], it is known that the DCT-IV matrix $\mathbf{C}^{\mathbf{IV}}_{\frac{\mathbf{N}}{\mathbf{2}}}$ can be factorized as

$$\mathbf{C}^{\mathbf{IV}}_{\frac{\mathbf{N}}{\mathbf{2}}} = \mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}} \tag{32}$$

where, $\mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}}$ is the $\frac{N}{2} \times \frac{N}{2}$ DCT-II matrix defined as follows:

$$\mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}}(k,n) = \cos\left\{\frac{\pi(2n+1)k}{N}\right\}, \qquad 0 \le n < \frac{N}{2}, 0 \le k < \frac{N}{2}, \tag{33}$$

$\mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}}$ is a $\frac{N}{2} \times \frac{N}{2}$ recursive addition matrix defined as follows (assuming $\frac{N}{2}$ is even):

$$\mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 & \dots & 0 \\ -\frac{1}{2} & 1 & 0 & 0 & \dots & 0 \\ \frac{1}{2} & -1 & 1 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ -\frac{1}{2} & 1 & -1 & 1 & \dots & 1 \end{bmatrix}_{\frac{N}{2} \times \frac{N}{2}} \tag{34}$$

and $\mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}}$ is a $\frac{N}{2} \times \frac{N}{2}$ diagonal matrix of factors

$$\mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}}(n,n) = 2\cos\left\{\frac{\pi(2n+1)}{2N}\right\}, \qquad 0 \le n < \frac{N}{2}. \tag{35}$$

It is known that the computational complexity (in terms of number of multiplications) of this factorization is equal to the theoretical minimum for DCT-IV [17]. It is also known that recursive additions defined by matrix $\mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}}$ may cause an increase in the dynamic range. This is usually a concern for the design of large transforms. However, since AAC-ELD employs small SBR matrices ($N/2 = 32$ or 64), we felt that it can be used in this case.
Using (29) and (32) in (28), we obtain:

$$\mathbf{X} = \left(\mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{I}}_{\mathbf{N}} + j\mathbf{J}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{A}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{II}}_{\mathbf{N}}\right) \mathbf{S}_{\mathbf{N}} \mathbf{C}_{\mathbf{5N}} \mathbf{u} \tag{36}$$

Since $\mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}}$ and $\mathbf{A}_{\frac{\mathbf{N}}{\mathbf{2}}}$ are diagonal matrices $\mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{A}_{\frac{\mathbf{N}}{\mathbf{2}}} = \mathbf{A}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}}$. Hence

$$\mathbf{X} = \left(\mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{I}}_{\mathbf{N}} + j\mathbf{J}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{A}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{II}}_{\mathbf{N}}\right) \mathbf{S}_{\mathbf{N}} \mathbf{C}_{\mathbf{5N}} \mathbf{u} \tag{37}$$

We notice from (17) - (24) that each column of $\mathbf{B}^{\mathbf{I}}_{\mathbf{N}}$ and $\mathbf{B}^{\mathbf{II}}_{\mathbf{N}}$ has exactly one non-zero element. Also, for every column, the non-zero element is at the same position for both $\mathbf{B}^{\mathbf{I}}_{\mathbf{N}}$ and $\mathbf{B}^{\mathbf{II}}_{\mathbf{N}}$. Further, the magnitude of every non-zero element is the same (in this case, 1). Hence, it can be shown that

$$\mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{I}}_{\mathbf{N}} = \mathbf{B}^{\mathbf{I}}_{\mathbf{N}} \mathbf{D}'_{\mathbf{N}} \tag{38}$$

$$\mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{II}}_{\mathbf{N}} = \mathbf{B}^{\mathbf{II}}_{\mathbf{N}} \mathbf{D}'_{\mathbf{N}} \tag{39}$$

where, $\mathbf{D}'_{\mathbf{N}}$ is a $N \times N$ diagonal matrix defined as:

$$\text{if } \mathbf{B}^{\mathbf{I}}_{\mathbf{N}}(n,k) = 1 \text{ or } -1, \text{ then } \mathbf{D}'_{\mathbf{N}}(k,k) = \mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}}(n,n), \qquad 0 \le n < \frac{N}{2}, 0 \le k < N. \tag{40}$$

By plugging these results in (37), we obtain:

$$\mathbf{X} = \left(\mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{I}}_{\mathbf{N}} + j\mathbf{J}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{A}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{II}}_{\mathbf{N}}\right) \mathbf{D}'_{\mathbf{N}} \mathbf{S}_{\mathbf{N}} \mathbf{C}_{\mathbf{5N}} \mathbf{u} \tag{41}$$

Again, we notice from (30) - (31) that each column of $\mathbf{S_N}$ has exactly one non-zero element, and that all non-zero elements in this matrix are equal to 1. Hence,

$$\mathbf{D'_N S_N} = \mathbf{S_N D''_{5N}} \tag{42}$$

where, $\mathbf{D''_{5N}}$ is a $5N \times 5N$ diagonal matrix defined as:

$$\text{if } \mathbf{S_N}(n,k) = 1, \text{ then } \mathbf{D''_{5N}}(k,k) = \mathbf{D'_N}(n,n), 0 \leq n < N; 0 \leq k < 5N \tag{43}$$

By using (42) in (41), we finally arrive at

$$\mathbf{X} = \left(\mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{I}}_{\mathbf{N}} + j\mathbf{J}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{L}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{II}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{A}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{B}^{\mathbf{II}}_{\mathbf{N}}\right) \mathbf{S_N C'_{5N} u} \tag{44}$$

where, $\mathbf{C'_{5N}}$ is a $5N \times 5N$ diagonal matrix of modified window factors:

$$\mathbf{C'_{5N}} = \mathbf{D''_{5N} C_{5N}} \tag{45}$$

The central idea of this section can be grasped by comparing (44) with (37). The merging of the transform factors (contributed by $\mathbf{D}_{\frac{\mathbf{N}}{\mathbf{2}}}$) with the SBR window coefficients (contributed by $\mathbf{C_{5N}}$) saves $\frac{N}{2} + \frac{N}{2} = N$ multiplications.

IV. Fast Algorithm for Complex SQMF

We now turn our attention to complex-domain SQMF in AAC-ELD (3).

Theorem 2: The matrix operation of complex SQMF (3) can be computed as follows:

$$\tilde{x}(n) = \tilde{x}_1(n) - \tilde{x}_2(n), \qquad 0 \leq n < N. \tag{46}$$

where, intermediate sequences $\tilde{x}_1(n)$ and $\tilde{x}_2(n)$ are obtained as

$$\tilde{x}_1\left(n + \frac{N}{4}\right) = \sum_{k=0}^{\frac{N}{2}-1} \operatorname{Re}\left[\tilde{X}(k)\right] \cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\} \qquad 0 \leq n < \frac{N}{2}, \tag{47}$$

$$\tilde{x}_1\left(n + \frac{3N}{4}\right) = -\tilde{x}_1\left(\frac{3N}{4} - 1 - n\right) \qquad 0 \leq n < \frac{N}{4}, \tag{48}$$

$$\tilde{x}_1\left(n - \frac{N}{4}\right) = \tilde{x}_1\left(\frac{3N}{4} - 1 - n\right) \qquad \frac{N}{4} \leq n < \frac{N}{2}, \tag{49}$$

and

$$\tilde{x}_2\left(n + \frac{N}{4}\right) = (-1)^n \sum_{k=0}^{\frac{N}{2}-1} \operatorname{Im}\left[\tilde{X}\left(\frac{N}{2} - 1 - k\right)\right] \cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\} \qquad 0 \leq n < \frac{N}{2}, \tag{50}$$

$$\tilde{x}_2\left(n + \frac{3N}{4}\right) = \tilde{x}_2\left(\frac{3N}{4} - 1 - n\right) \qquad 0 \leq n < \frac{N}{4}, \tag{51}$$

$$\tilde{x}_2\left(n - \frac{N}{4}\right) = -\tilde{x}_2\left(\frac{3N}{4} - 1 - n\right) \qquad \frac{N}{4} \leq n < \frac{N}{2}. \tag{52}$$

The proof of this statement is obtained by using essentially the same technique described in Appendix A. It can be seen that only additions, sign changes and reordering operations are needed to reduce this filterbank to computing 2 $N/2$-point DCT-IV transforms. We show the flowgraph of this process in Figure IV.

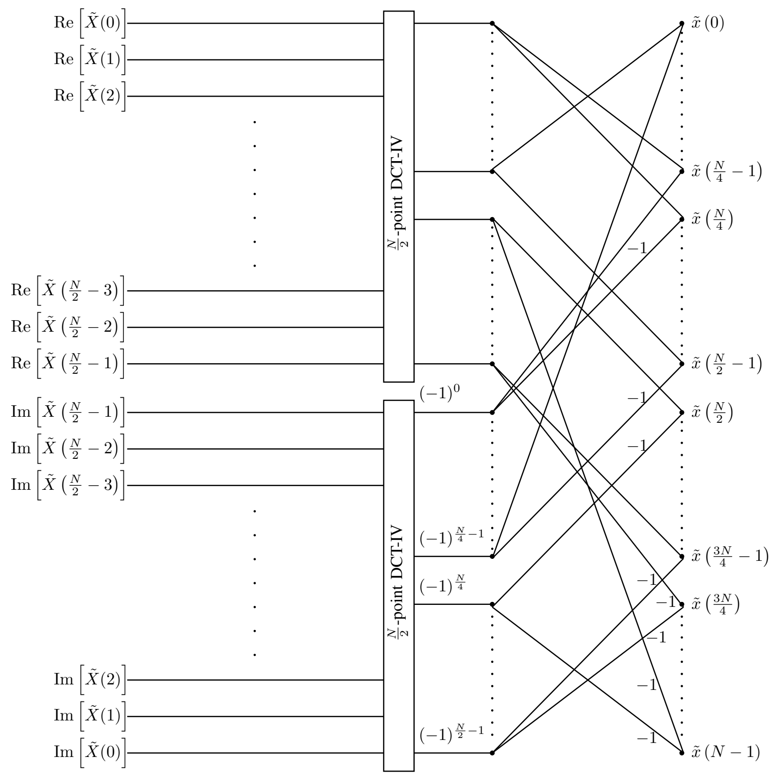

Fig. 3. Flowgraph of the proposed DCT-IV-based algorithm for computing SQMF.

A. Mapping to DCT-II and Further Optimization

Similar to Section III-A, we develop a matrix description of the synthesis filterbank involving the matrix operation, followed by the windowing and overlap-addition operations. In the process, we can merge the multiplications in the post-processing stage of DCT-IV implementation with the windowing stage, thereby, further reducing the computational complexity.

Define the following vectors:

$$\tilde{\mathbf{x}}_{\mathbf{1}} = \left[\tilde{x}_1(0) \quad \ldots \quad \tilde{x}_1(N-1)\right]^T, \quad \tilde{\mathbf{x}}_{\mathbf{2}} = \left[\tilde{x}_2(0) \quad \ldots \quad \tilde{x}_2(N-1)\right]^T, \quad \tilde{\mathbf{x}} = \left[\tilde{x}(0) \quad \ldots \quad \tilde{x}(N-1)\right]^T,$$
$$\tilde{\mathbf{X}} = \left[\tilde{X}(0) \quad \ldots \quad \tilde{X}\left(\frac{N}{2}-1\right)\right]^T, \quad \tilde{\mathbf{X}}_{\mathbf{r}} = \mathrm{Re}\left[\tilde{\mathbf{X}}\right], \quad \tilde{\mathbf{X}}_{\mathbf{i}} = \mathrm{Im}\left[\tilde{\mathbf{X}}\right]$$

Then

$$\tilde{\mathbf{x}} = \tilde{\mathbf{x}}_{\mathbf{1}} - \tilde{\mathbf{x}}_{\mathbf{2}} = \mathbf{B}^{\mathbf{r}}_{\mathbf{N}} \mathbf{C}^{\mathbf{IV}}_{\frac{\mathbf{N}}{\mathbf{2}}} \tilde{\mathbf{X}}_{\mathbf{r}} - \mathbf{B}^{\mathbf{i}}_{\mathbf{N}} \mathbf{A}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{C}^{\mathbf{IV}}_{\frac{\mathbf{N}}{\mathbf{2}}} \mathbf{J}_{\frac{\mathbf{N}}{\mathbf{2}}} \tilde{\mathbf{X}}_{\mathbf{i}} \tag{53}$$

where, $\mathbf{B_N^r}$ and $\mathbf{B_N^i}$ denote $N \times \frac{N}{2}$ matrices establishing relation between $\tilde{\mathbf{x}}_1$, $\tilde{\mathbf{x}}_2$ and $\tilde{\mathbf{x}}$ (cf. (47) - (52)):

$$\mathbf{B_N^r}\left(n, \frac{N}{4} - 1 - n\right) = 1 \qquad \text{for } 0 \le n < \frac{N}{4} \tag{54}$$

$$\mathbf{B_N^r}\left(n + \frac{N}{4}, n\right) = 1 \qquad \text{for } 0 \le n < \frac{N}{2} \tag{55}$$

$$\mathbf{B_N^r}\left(n + \frac{3N}{4}, \frac{N}{2} - 1 - n\right) = -1 \qquad \text{for } 0 \le n < \frac{N}{4} \tag{56}$$

$$\mathbf{B_N^r}(n,k) = 0 \qquad \text{for all other combinations of } n \text{ and } k \tag{57}$$

$$\mathbf{B_N^i}\left(n, \frac{N}{4} - 1 - n\right) = -1 \qquad \text{for } 0 \le n < \frac{N}{4} \tag{58}$$

$$\mathbf{B_N^i}\left(n + \frac{N}{4}, n\right) = 1 \qquad \text{for } 0 \le n < \frac{N}{2} \tag{59}$$

$$\mathbf{B_N^i}\left(n + \frac{3N}{4}, \frac{N}{2} - 1 - n\right) = 1 \qquad \text{for } 0 \le n < \frac{N}{4} \tag{60}$$

$$\mathbf{B_N^i}(n,k) = 0 \qquad \text{for all other combinations of } n \text{ and } k. \tag{61}$$

Using the relationship between DCT-IV and DCT-II (32) (cf. [19]), we obtain

$$\tilde{\mathbf{x}} = \mathbf{B_N^r} \mathbf{D_{\frac{N}{2}}} \mathbf{C^{III}_{\frac{N}{2}}} \mathbf{L_{\frac{N}{2}}}^T \tilde{\mathbf{X}}_\mathbf{r} - \mathbf{B_N^i} \mathbf{A_{\frac{N}{2}}} \mathbf{D_{\frac{N}{2}}} \mathbf{C^{III}_{\frac{N}{2}}} \mathbf{L_{\frac{N}{2}}}^T \mathbf{J_{\frac{N}{2}}} \tilde{\mathbf{X}}_\mathbf{i} \tag{62}$$

where, $\mathbf{C^{III}_{\frac{N}{2}}} = \mathbf{C^{II}_{\frac{N}{2}}}^T$ is matrix of DCT-III.

We note that $\mathbf{B_N^r}$ and $\mathbf{B_N^i}$ have non-zero elements in the same row and column positions. We also note that every row has exactly one non-zero element. Further $\mathbf{A_{\frac{N}{2}}}$ and $\mathbf{D_{\frac{N}{2}}}$ commute. Hence,

$$\tilde{\mathbf{x}} = \mathbf{D_N^I} \left(\mathbf{B_N^r} \mathbf{C^{III}_{\frac{N}{2}}} \mathbf{L_{\frac{N}{2}}}^T \tilde{\mathbf{X}}_\mathbf{r} - \mathbf{B_N^i} \mathbf{A_{\frac{N}{2}}} \mathbf{C^{III}_{\frac{N}{2}}} \mathbf{L_{\frac{N}{2}}}^T \mathbf{J_{\frac{N}{2}}} \tilde{\mathbf{X}}_\mathbf{i} \right) \tag{63}$$

where, $\mathbf{D_N^I}$ is a $N \times N$ diagonal matrix

$$\text{if } \mathbf{B_N^r}(n,k) = 1 \text{ or } -1, \text{ then } \mathbf{D_N^I}(n,n) = \mathbf{D_{\frac{N}{2}}}(k,k), \qquad 0 \le n < N, 0 \le k < \frac{N}{2}. \tag{64}$$

In the decoder, this operation of computing $\tilde{\mathbf{x}}$ is repeated 10 times to form a vector of length $10N$, from which, a vector of length $5N$ is formed. Essentially, this operation is equivalent to applying the matrix $\mathbf{S_{5N}}^T$ to the vector $\tilde{\mathbf{x}}$. That is,

$$\mathbf{g} = \mathbf{S_{5N}}^T \tilde{\mathbf{x}} \tag{65}$$

Vector $\mathbf{g}$ is multiplied by a diagonal matrix of constants $\mathbf{C^S_{5N}}$ to form a vector $\mathbf{w}$:

$$\mathbf{w} = \mathbf{C^S_{5N}}\, \mathbf{g} \tag{66}$$

The output audio sample vector $\mathbf{y}$ of length $\frac{N}{2}$ is formed by applying an addition matrix $\mathbf{E}$ of dimensions $\frac{N}{2} \times 5N$ to the vector $\mathbf{w}$:

$$\mathbf{y} = \mathbf{E}\, \mathbf{w} \tag{67}$$

where,

$$\mathbf{E}\left(n, n+\frac{N}{2}k\right) = 1, \qquad 0 \le n < \frac{N}{2},\ 0 \le k < 10 \tag{68}$$

$$\mathbf{E}(n,k) = 0, \qquad \text{for all other combinations of } 0 \le n < \frac{N}{2},\ 0 \le k < 5N. \tag{69}$$

Hence, based on (63), (65), (66) and (67):

$$\mathbf{y} = \mathbf{E}\mathbf{C^S_{5N}}\mathbf{S_{5N}}^T\mathbf{D^I_N}\left(\mathbf{B^r_N}\mathbf{C^{III}_{\frac{N}{2}}}\mathbf{L_{\frac{N}{2}}}^T\tilde{\mathbf{X}}_\mathbf{r} - \mathbf{B^i_N}\mathbf{A_{\frac{N}{2}}}\mathbf{C^{III}_{\frac{N}{2}}}\mathbf{L_{\frac{N}{2}}}^T\mathbf{J_{\frac{N}{2}}}\tilde{\mathbf{X}}_\mathbf{i}\right). \tag{70}$$

As in the case of the analysis filterbank, we can change the order of the matrix multiplication and obtain the following result:

$$\mathbf{y} = \mathbf{E}\mathbf{C^S_{5N}}\mathbf{D^{II}_{5N}}\mathbf{S_{5N}}^T\left(\mathbf{B^r_N}\mathbf{C^{III}_{\frac{N}{2}}}\mathbf{L_{\frac{N}{2}}}^T\tilde{\mathbf{X}}_\mathbf{r} - \mathbf{B^i_N}\mathbf{A_{\frac{N}{2}}}\mathbf{C^{III}_{\frac{N}{2}}}\mathbf{L_{\frac{N}{2}}}^T\mathbf{J_{\frac{N}{2}}}\tilde{\mathbf{X}}_\mathbf{i}\right) \tag{71}$$

where, $\mathbf{D^{II}_{5N}}$ is a $5N \times 5N$ diagonal matrix of modified window factors

$$\mathbf{D^{II}_{5N}}\left(nN+k, nN+k\right) = \mathbf{D^I_N}(k,k) \qquad 0 \le n < 5, 0 \le k < N. \tag{72}$$

V. Fast Algorithms for Other SBR Filterbanks

The derivations for the other filterbanks introduced in section IV closely follow the previous two sections. Hence, we present the results without proofs.

A. Real Analysis QMF

$$X_R(k) = \sum_{n=0}^{\frac{N}{2}-1} x_1(n)\cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\}, \quad \text{for } 0 \le k < \frac{N}{2}, \tag{73}$$

where,

$$x_1(n) = \begin{cases} x\left(n+\frac{3N}{4}\right) + x\left(\frac{3N}{4}-1-n\right), & 0 \le n < \frac{N}{4} \\ x\left(\frac{3N}{4}-1-n\right) - x\left(n-\frac{N}{4}\right), & \frac{N}{4} \le n < \frac{N}{2}. \end{cases} \tag{74}$$

B. Real Synthesis QMF

$$\tilde{x}_R\left(n+\frac{N}{4}\right) = \sum_{k=0}^{\frac{N}{2}-1} \tilde{X}_R(k)\cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\}, \qquad 0 \le n < \frac{N}{2}, \tag{75}$$

$$\tilde{x}_R\left(n+\frac{3N}{4}\right) = -\tilde{x}_R\left(\frac{3N}{4}-1-n\right), \qquad 0 \le n < \frac{N}{4}, \tag{76}$$

$$\tilde{x}_R\left(n-\frac{N}{4}\right) = \tilde{x}_R\left(\frac{3N}{4}-1-n\right), \qquad \frac{N}{4} \le n < \frac{N}{2}. \tag{77}$$

VI. Computational Complexity

In this section, we analyze the computational complexity of the proposed algorithms by computing the required number of additions and multiplications. Note that when the lengths of analysis and synthesis filterbanks are equal, the required number of additions and multiplications for their computation will be the same. This can be observed from the almost tranpose-like relationship between the flowgraphs of the analysis and synthesis filterbanks. Therefore, the presented formulas for the number of additions and multiplications are valid for both analysis and synthesis filterbanks.

A. Complex Filterbanks

In this case, multiplications are contributed by the two DCT-II blocks and by the diagonal matrix $\mathbf{C'_{5N}}$ defined in (45). In LD-SBR filterbanks, N is a power of 2. So, if we use one of the several fast algorithms available for DCT-II, such as [19], the number of multiplications contributed by each DCT-II block in the figure would be $\frac{N}{4}\left(\log_2 N - 1\right)$. See [19] for details. The matrix $\mathbf{C'_{5N}}$ contributes $5N$ multiplications. Therefore, the total number of multiplications is given by

$$M\left(N\right) = \frac{9N}{2} + \frac{N}{2}\log_2 N. \tag{78}$$

Additions are contributed by the DCT-II blocks and the matrices $\mathbf{L_{\frac{N}{2}}}$, $\mathbf{B^{I}_{N}}$, $\mathbf{B^{II}_{N}}$ and $\mathbf{S_N}$. Matrix $\mathbf{L_{\frac{N}{2}}}$ contributes $\left(\frac{N}{2} - 1\right)$ additions. $\mathbf{B^{I}_{N}}$ and $\mathbf{B^{II}_{N}}$ each contribute $\frac{N}{2}$ additions. $\mathbf{S_N}$ contributes $4N$ additions. Each DCT-II block contributes $\left(\frac{3N}{4}\log_2\left(\frac{N}{2}\right) - \frac{N}{2} + 1\right)$ additions. See [19] for details. The total number of additions is given by

$$A\left(N\right) = \frac{7N}{2} + \frac{3N}{2}\log_2 N. \tag{79}$$

B. Real Filterbanks

The real-valued filterbanks have lesser computational complexity because of the absence of the imaginary component. Following the previous subsection, we have:

$$M\left(N\right) = \frac{19N}{4} + \frac{N}{4}\log_2 N, \tag{80}$$

$$A\left(N\right) = \frac{15N}{4} + \frac{3N}{4}\log_2 N. \tag{81}$$

The numbers of required additions and multiplications for the computation of LD-SBR filterbanks in AAC-ELD is summarized in Table 1. The first column shows the complexity of straightforward computation of these filterbanks. The second column shows the complexity of algorithms utilizing mappings to DCT-IV. The last column shows the numbers for the proposed DCT-II-based algorithms, absorbing factors in the windowing stage.

TABLE I
COMPUTATIONAL COMPLEXITY OF PROPOSED ALGORITHMS

Filterbank	N	No Optimizations		Algorithms using DCT-IV		Algorithm using DCT-II	
		Multiplications	Additions	Multiplications	Additions	Multiplications	Additions
Complex AQMF in Encoder	128	17024	16832	1152	1792	1024	1792
Complex AQMF in Decoder	64	4416	4320	544	800	480	800
Real AQMF in Decoder	64	2368	2272	432	528	400	528
Complex SQMF in Decoder	128	17024	16832	1152	1792	1024	1792
Complex Downsampled SQMF in Decoder	64	4416	4320	544	800	480	800
Real SQMF in Decoder	128	8832	8640	896	1152	832	1152
Real Downsampled SQMF in Decoder	64	2368	2272	432	528	400	528

VII. CONCLUSIONS

In this paper, we derived fast algorithms for low-delay SBR filterbanks used in MPEG-4 AAC-ELD. The fast algorithms are based on the mapping of the analysis and synthesis SBR filterbanks to DCT-IV. Since several fast algorithms exist for DCT-IV, this mapping provides us with a fast algorithm for the filterbanks. We have also shown that by mapping DCT-IV to DCT-II, the multiplications in pre/post processing can be absorbed into the windowing stage that precedes/succeeds the SBR filterbank, thereby further reducing the number of multiplications. Finally, we have presented a complexity analysis of our algorithms, showing that they are appreciably faster than other possible implementations.

APPENDIX A
PROOF FOR THEOREM 1

Let $p = n - \frac{3N}{4}$ in (1). Then AQMF matrix operation (1) can be written as

$$X(k) = \sum_{p=-\frac{3N}{4}}^{-1} x\left(p + \frac{3N}{4}\right) \exp\left\{\frac{j\pi(2k+1)(2p+1)}{2N}\right\} + \sum_{p=0}^{\frac{N}{4}-1} x\left(p + \frac{3N}{4}\right) \exp\left\{\frac{j\pi(2k+1)(2p+1)}{2N}\right\}$$

Let $n = p + N$ in the first sum and also replace p by n in the second. Then

$$X(k) = \sum_{n=\frac{N}{4}}^{N-1} -x\left(n - \frac{N}{4}\right) \exp\left\{\frac{j\pi(2k+1)(2n+1)}{2N}\right\} + \sum_{n=0}^{\frac{N}{4}-1} x\left(n + \frac{3N}{4}\right) \exp\left\{\frac{j\pi(2k+1)(2n+1)}{2N}\right\} \quad (82)$$

Define a new sequence $x'(n)$ as follows:

$$x'(n) \triangleq \begin{cases} x\left(n + \frac{3N}{4}\right) & \text{for} \quad 0 \le n < \frac{N}{4} \\ -x\left(n - \frac{N}{4}\right) & \text{for} \quad \frac{N}{4} \le n < N. \end{cases} \quad (83)$$

Then (82) becomes

$$\begin{aligned} X(k) &= \sum_{n=0}^{N-1} x'(n) \exp\left\{\frac{j\pi(2k+1)(2n+1)}{2N}\right\} \\ &= \sum_{n=0}^{\frac{N}{2}-1} x'(n) \exp\left\{\frac{j\pi(2k+1)(2n+1)}{2N}\right\} + \sum_{n=\frac{N}{2}}^{N-1} x'(n) \exp\left\{\frac{j\pi(2k+1)(2n+1)}{2N}\right\} \end{aligned}$$

Replacing n by $N-1-n$ in the second sum, we obtain

$$\begin{aligned} X(k) &= \sum_{n=0}^{\frac{N}{2}-1} x'(n) \exp\left\{\frac{j\pi(2k+1)(2n+1)}{2N}\right\} + \sum_{n=0}^{\frac{N}{2}-1} -x'(N-1-n) \exp\left\{-\frac{j\pi(2k+1)(2n+1)}{2N}\right\} \\ &= \sum_{n=0}^{\frac{N}{2}-1} \{x'(n) - x'(N-1-n)\} \cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\} \\ &\quad + j\sum_{n=0}^{\frac{N}{2}-1} \{x'(n) + x'(N-1-n)\} \sin\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\} \end{aligned} \tag{84}$$

From (15), (16), (83) we see that for $0 \le n < \frac{N}{2}$,

$$x'(n) - x'(N-1-n) = x_1(n)$$

$$x'(n) + x'(N-1-n) = x_2(n)$$

Hence, (84) becomes

$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x_1(n) \cos\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\} + j\sum_{n=0}^{\frac{N}{2}-1} x_2(n) \sin\left\{\frac{\pi(2k+1)(2n+1)}{2N}\right\},\ 0 \le k < \frac{N}{2}. \tag{85}$$

We note that the first sum is a $\frac{N}{2}$-point DCT-IV and the second sum is a $\frac{N}{2}$-point DST-IV. Further, DST-IV can be mapped to DCT-IV using (11). This completes the proof of the theorem.

REFERENCES

[1] A. M. Kondoz, *Digital Speech: Coding for Low Bit Rate Communication Systems*, 2nd ed. Wiley, 2004.

[2] T. Painter and A. Spanias, “Perceptual Coding of Digital Audio”, *Proc. IEEE*, vol. 88, pp. 451-515, Apr. 2000.

[3] R. Geiger, J. Herre, M. Jander, M. Multrus, M. Schmidt, M. Schnell, G. Schuller, “Enhanced MPEG-4 Low Delay AAC - Low Bitrate High Quality Communication”, *122nd Convention of the AES*, Vienna, Austria, May 2007.

[4] ISO/IEC 14496-3:2005/FDAM9, *Information Technology - Generic Coding of Moving Pictures and Associated Audio Information - Part 3: Advanced Audio Coding (AAC), Amendment 9: Enhanced Low Delay AAC*, Oct. 2007.

[5] ISO/IEC 14496-3:2005, *Information Tecnology - Generic Coding of Moving Pictures and Associated Audio Information - Part 3: Advanced Audio Coding (AAC), Subpart 4: General Audio Coding (GA) - AAC, TwinVQ, BSAC*, 2005.

[6] M. Schnell, R. Geiger, M. Schmidt, M. Multrus, M. Mellar, J. Herre, G. Schuller, “Low Delay Filterbanks for Enhanced Low-Delay Audio Coding,” *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 235–238, Oct. 21-24, 2007.

[7] M. Wolters, K. Kjorling, D. Homm, and H. Purnhagen, “A closer look into MPEG-4 High Efficiency AAC,” *115th Convention of the AES*, New York, NY, October 10-13 2003.

[8] R. K. Chivukula, and Y. A. Reznik, “Efficient Implementation of a Class of MDCT/IMDCT Filterbanks for Speech and Audio Coding Applications,” *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pp. 213–216, 2008.

[9] R. K. Chivukula, Y. A. Reznik, and V. Devarajan, “Efficient Algorithms for MPEG-4 AAC-ELD, AAC-LD and AAC-LC Filterbanks,” *Proc. Int. Conf. Audio, Language, Image Processing*, pp. 1629–1634, 7-9 July 2008.

[10] G. Schuller and M. Smith, "New Framework for Modulated Perfect Reconstruction Filter Banks," *IEEE Trans. Signal Processing*, vol. 44, no. 8, August 1996, pp. 1941–1954.

[11] T. Q. Nguyen, and R. D. Koilpillai, "The Theory and Design of Arbitrary-Length Cosine-Modulated Filter Banks and Wavelets, Satisfying Perfect Reconstruction," *IEEE Trans. Signal Processing*, vol. 44, no. 3, March 1996, pp. 473–483.

[12] M. Harteneck, S. Weiss, and R. W. Stewart, "Design of near perfect reconstruction oversampled filter banks for subband adaptive filters," *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing*, vol.46, no.8, pp.1081-1085, Aug 1999.

[13] K. Konstantinides, "Fast Subband Filtering in MPEG Audio Coding," *IEEE Signal Processing Letters*, vol. 1, no. 2, pp. 26–28, Feb. 1994.

[14] ISO/IEC 11172-3:1993, *Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 3: Audio*, 1993.

[15] K. R. Rao, *The Transform and Data Compression Handbook*, CRC Press, Boca Raton, 2001.

[16] V. Britanak, P. Yip, and K. R. Rao, *Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations*, Academic Press, Oxford, 2007.

[17] H. S. Malvar, *Signal Processing with Lapped Transforms*, Artech House, Boston, 1992.

[18] J. H. Rothweiler, "Polyphase Quadrature Filters - A New Subband Coding Technique", *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pp. 1280–1283, 1983.

[19] C. W. Kok, "Fast Algorithm for Computing Discrete Cosine Transform," *IEEE Trans. Signal Processing*, vol. 45, no. 3, pp. 757–760, Mar. 1997.

[20] S. -W. Huang and T. -H. Tsai, "Fast Decomposition of Filterbanks for the State-of-the-Art Audio Coding," *IEEE Signal Processing Letters*, vol. 12, no. 10, pp. 693–696, Oct. 2005.

[21] H. -W. Hsu, C. -M. Liu and W. -C. Lee, "Fast Complex Quadrature Mirror Filterbanks for MPEG-4 HE-AAC," *121st Convention of the AES*, San Francisco, CA, Oct. 2006.

[22] M. Dietz, L. Liljeryd, K. Kjorling, and O. Kunz, "Spectral Band Replication, a Novel Approach in Audio Coding," *12th Convention of the AES*, Munich, Germany, Apr. 2002.