Time-Recursive Computation, Part II: Methodology, and Application on QMF Banks and ELT

by E. Frantzeskakis, J.S. Baras, and K.J.R. Liu

T.R. 93-18r1

The Institute for Systems Research is supported by the
National Science Foundation Engineering Research Center Program (NSFD CD 8803012),
the University of Maryland, Harvard University, and Industry

Time-Recursive Computation, Part II: Methodology, and Application on QMF Banks and ELT*

Emmanuel Frantzeskakis John S. Baras† and K. J. Ray Liu‡

Electrical Engineering Department
Institute of Systems Research
University of Maryland
College Park, MD 20742

Abstract

Recent advances in ISDN have promoted applications such as video-phone, tele-conferencing and HDTV, that demand real-time processing of large volume audio, video and speech data. Being the only refuge for this intense computation, the VLSI technology favors modular and regular designs with local communication requirements. In this light, the framework for time-recursive computation, presented in part I [7] of this two-part paper, provides the background for designing efficient VLSI implementations, capable of accommodating high throughput requirements. In part II, we develop a routine that can be used for designing the time-recursive architecture of a given linear operator in a systematic manner. Three classes of QMF banks are used as design examples: the lossless QMF bank, the cosine modulated QMF bank and two Extended Lapped Transforms, one of them being the Modulated Lapped Transform (MLT). In addition to demonstrating the use of the design procedure, these examples provide novel results, interesting on their own right. In particular, the time-recursive architecture we propose for an N-point MLT, also known as Modified DCT or Time Domain Aliasing Cancellation (TDAC) transform, requires $2N+3$ multipliers, $3N+3$ adders and $N-1$ rotation circuits.

SP EDICS:
5.2. Algorithms and Application Mappings
5.1. Architectures and VLSI Hardware

*Research supported in part by grant NSFD CDR 8803012 through the Engineering Research Center's Program.
†Martin Marietta Chair in Systems Engineering
‡Research also supported in part by ONR grant N00014-93-1-0566.

1 Introduction

In part I [7] of this two part paper, we revealed the infrastructure of time-recursive computation. We also identified the fields where this model of computation has already been applied. The purpose of this part II is two-fold. First, to integrate the framework provided in part I into a *Generic Architecture Design Procedure* and to demonstrate its usage by means of application in some perfect Reconstruction (PR) Quadrature Mirror Filter (QMF) bank designs, namely the lossless QMF banks [28, 27], the cosine modulated PR QMF banks [27], and two Extended Lapped Transforms (ELT) [21, 22]. Second, to develop efficient novel architectures that will assist the custom VLSI implementation of the QMF based computations, thus responding to the demand of real-time computation that has recently arisen in a number of applications such as audio, video and speech compression [13, 25].

Fast algorithm practices have yielded efficient implementations for the computations we mentioned above [27, 22]. Nevertheless, these algorithms not always translate into efficient VLSI architectures, the main reason being that they require global communication. The time-recursive approach suggests an attractive alternative, where one seeks the minimization of the operator counts, while maintaining the communication locality. Therefore, linear area utilization is obtained in VLSI designs [18, 4, 19, 2]. This is in contrast with the fast algorithm design principles, where the goal is the minimization of the associated operation count, regardless of the locality of the communication. Consequently, a quadratic area requirement is implied by the corresponding VLSI designs [26].

The terminology that has been introduced in part I [7], such as mapping operator, kernel function, kernel group, shift property (SP), difference equation property (DEP) and periodicity property (PP) will be used throughout the text of part II without any further explanation. Furthermore, we shall use the notation I.n to reference item n from part I. For example, "Lemma I.2" should be read as "Lemma 2 in part I".

In Section 2, we draw a *Generic Design Procedure* for time-recursive realization. By encompassing the appropriate decision rules and procedures in a flow diagram, this tool enables the design of a time-recursive realization of a linear mapping operator [7] in a routine way. We distinguish categories of mapping operators based on the way they are specified and the structure they are implemented with. In particular, a mapping operator can be specified either by some close form expression or as a vector of coefficients, while its time-recursive realization may employ either lattice or IIR building modules [7]. The realization of a lossless QMF bank [28, 27] is presented in Section 3 as the first design example. In this example, the mapping operators are specified in a vector of coefficient form, while the building blocks of the resulting realization are lattice modules. In Section 4, the second design example is presented, which concerns the realization of the modulation matrix in the cosine modulated QMF bank. The mapping operators are specified by a close form expression, while IIR modules are employed by the resulting realization. In Section 5, a special case of the cosine modulated QMF banks is discussed, namely the ELT. We develop the time-recursive architectural realizations for two instances of the ELT, the Modulated Lapped Transform (MLT) and an ELT with basis functions of length $4N$, N being the number of transform coefficients. In this case, the mapping operators are specified by a close form expression, while lattice modules are the appropriate building blocks for the associated realizations. We conclude with Section 6.

2 Generic Design Procedure

In Fig. 1, we present the flow diagram of the Generic Design Procedure that addresses the problem of designing a time-recursive architecture for a linear mapping operator. The input specification can be either the coefficient vector $[h_0 \; h_1 \; \cdots \; h_{N-1}]$ or a function in the form $\phi(\cdot) = \sum_i c_i \phi_i(\cdot)$, where the kernels in $\{\phi_i(\cdot)\}$ belong in the class of functions specified by Lemma I.2. For the majority of interesting applications the specified input expression can be manipulated so that kernel groups of size $M = 2$ suffice for the architecture design. This is desirable since it implies locality and low complexity in the resulted design. Therefore, the design procedure will be focused on kernel groups of size $M = 2$, that also serves the purpose of simplicity and clarity in this presentation. Nevertheless, we believe it deserves the name *generic* since it considers all different factors that affect the architectural structure. Also, it conveys basic notions, so that the design rules and procedures in Fig. 1 can be easily modified to accommodate arbitrary values of M.

In the sequel, we present the flow diagram in Fig. 1. If the input is specified in the coefficient vector form the preprocessing described in Subsection I.3.5 has to be employed, resulting an expression of the mapping operator coefficients in terms of sine, cosine and exponential functions. This is labeled with Step 0 in the flow graph in Fig. 1. The mapping operator specified as a linear expression of kernel functions (regardless whether this is the output of Step 0 or the provided input) is fed to Step 1 described in Subsection I.3.3. In this step of the design procedure we determine the kernel groups $\{\mathbf{f}_i(\cdot)\}$ that are associated to the functions $\{\phi_i(\cdot)\}$. For each kernel group $\mathbf{f}_i$ we question the periodicity property (PP). If PP is not satisfied the lattice architecture is decided and it is determined by Step 2. Otherwise, we question whether both members of $\mathbf{f}_i$ participate in the expression $\phi(\cdot) = \sum_i c_i \phi_i(\cdot)$. If this is the case, then the lattice architecture is preferable, since it performs the computation pertinent to the second kernel function at no additional cost. Furthermore, the resulted lattice structure often comprises of a rotation circuit that can be implemented very efficiently by using a CORDIC processor [12] or distributed arithmetic [30, 24]. On the other hand, the IIR architecture is recommended if only one of the kernel functions in $\mathbf{f}_i$ is to be implemented. If this kernel function ϕ_i satisfies the difference equation property the IIR architecture parameters can easily be determined by using Corollary I.3 (see also Fig. I.6 and Eqn. (I.22)). Although given a kernel function one can construct the associated difference equation, we suggest this path only if this equation can be found in tabulated form (see for example [3]). Otherwise, a less painful way to determine the IIR architecture parameters is to determine first the corresponding lattice parameters by following Step 2 and then using Corollary I.2 (see also Fig. I.6

and Eqn. (I.19)).

In the following Sections we give a number of architecture design examples in order to demonstrate the usage of the above design procedure.

3 Lossless QMF Banks

The function of a 2-band Quadrature Mirror Filter (QMF) bank is to analyze an input signal in 2 subband components, so that reconstruction of the original signal is possible based on them [27]. Here, we consider the time-recursive implementation of a QMF bank that consists of the real coefficient, finite impulse response (FIR) filters

$$\mathbf{H} = [h_{N-1}\ h_{N-2}\ \cdots\ h_0] \text{ and } \mathbf{G} = [g_{N-1}\ g_{N-2}\ \cdots\ g_0],$$

and it exhibits the losslessness property [1][28, 27]. The sequence of the steps we follow in this Section, demonstrate the time-recursive realization for the case where the mapping operator is specified in the vector of coefficient form.

Implementing the filters **H** and **G** is equivalent to implementing the mapping operators

$$[h_0\ h_1\ \cdots\ h_{N-1}], \text{ and } [g_0\ g_1\ \cdots\ g_{N-1}].$$

Since the input specification to the Generic Design Procedure is in coefficient vector form, we follow the left branch of the flow diagram in Fig. 1.

Step 0.1: Consider the linear system with the N first Markov coefficients being equal to the N

[1] In other words, the frequency responses of the filters satisfy the relation

$$H^*(e^{j\omega})H(e^{j\omega}) + H^*(e^{j\omega})H(e^{j\omega}) = c, \ \ \forall\omega,$$

where c is a positive scalar and "$*$" denotes conjugation. Without loss of generality we assume $c = 1$.

columns of the matrix

$$\begin{bmatrix} h_0 & h_1 & \cdots & h_{N-1} \\ g_0 & g_1 & \cdots & g_{N-1} \end{bmatrix}.$$

We specify the partial realization $\{\mathbf{A}, \mathbf{b}, \mathbf{c}\}$ of minimal order M for this linear system [16, 17], so that

$$\begin{bmatrix} h_n \\ g_n \end{bmatrix} = \mathbf{c}\mathbf{A}^n\mathbf{b}, \quad n = 0, 1, \cdots, N-1. \tag{1}$$

Step 0.2: Bring the triplet $\{\mathbf{A}, \mathbf{b}, \mathbf{c}\}$ in the modal canonical form [16]. Since our system is lossless all the eigenvalues of the system matrix $\mathbf{A}$ will have the same magnitude, equal to 1 [27]. For the sake of concreteness, suppose that the order of the system is $M = 3$. The format of the matrix $\mathbf{A}$ will be

$$\mathbf{A} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & \beta \end{bmatrix},$$

where α takes values in the interval $[0, 2\pi)$ and β equals either 1 or -1. The $M \times 1$ vector $\mathbf{b}$ and the $2 \times M$ matrix $\mathbf{c}$ do not have any particular structure.

Step 0.3: By substituting the above expression of $\mathbf{A}$ in (1) and expanding the matrix notation we obtain

$$\begin{bmatrix} h_n \\ g_n \end{bmatrix} = \begin{bmatrix} c_{00} & c_{01} & c_{02} \\ c_{10} & c_{11} & c_{12} \end{bmatrix} \begin{bmatrix} \cos\alpha n & \sin\alpha n & 0 \\ -\sin\alpha n & \cos\alpha n & 0 \\ 0 & 0 & \beta^n \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix}$$

and consequently

$$\begin{bmatrix} h_n \\ g_n \end{bmatrix} = \begin{bmatrix} c_{00}b_0 + c_{01}b_1 \\ c_{10}b_0 + c_{11}b_1 \end{bmatrix} \cos\alpha n + \begin{bmatrix} -c_{01}b_0 + c_{00}b_1 \\ c_{11}b_0 + c_{10}b_1 \end{bmatrix} \sin\alpha n + \begin{bmatrix} c_{02}b_2 \\ c_{12}b_2 \end{bmatrix} \beta^n. \tag{2}$$

Step 1: The kernel groups we need to implement are

$$\mathbf{f}_0(n) = \begin{bmatrix} f_{00}(n) \\ f_{01}(n) \end{bmatrix} = \begin{bmatrix} \cos\alpha n \\ \sin\alpha n \end{bmatrix} \quad \text{and} \quad \mathbf{f}_1(n) = f_{10}(n) = \beta^n.$$

Step 2: For the kernel group $\mathbf{f}_0(\cdot)$ we have $\mathbf{f}_0(n-1) = \mathbf{R}\mathbf{f}_0(n)$ with

$$\mathbf{R} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix}.$$

We also have

$$\begin{bmatrix} f_{00}(0) \\ f_{01}(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} f_{00}(N) \\ f_{01}(N) \end{bmatrix} = \begin{bmatrix} \cos\alpha N \\ \sin\alpha N \end{bmatrix}.$$

The resulted architecture is depicted in Fig. 2, module M_0. On the other hand, for the singleton kernel group $\mathbf{f}_1(\cdot)$ we have $\mathbf{f}_1(n-1) = \mathbf{R}\mathbf{f}_1(n)$ with $\mathbf{R} = \frac{1}{\beta}$ and also $f_{10}(0) = 1$, $f_{10}(N) = \beta^N$. The associated architecture is demonstrated by module M_1 in Fig. 2. The architectural implementation of the given pair of mapping operators for the case when $M = 3$ is shown in Fig. 2. In general, we need to implement M kernel functions. Among these functions no more than two are in the form of $\mathbf{f}_1(\cdot)$ seen above (since only two distinct such functions exist with $\beta = 1$ and $\beta = -1$) and they are implemented by module M_1. The rest of the kernel functions will group into pairs of cosine-sine functions specified by the parameter α, as dictated by $\mathbf{f}_0(\cdot)$ in the above example, and they can be implemented by module M_0. For example, consider the pair of wavelet filters $\mathbf{H}$ and $\mathbf{G}$, whose coefficients, obtained in [1], are given in Table 1. The lengths of the filters $\mathbf{H}$ and $\mathbf{G}$ are 9 and 7 respectively. The size of the kernel group we have to implement (that is the order of the associated linear system) is $M = 6$. The resulted architecture is shown in Fig. 3 and it involves two copies of module M_0 and two copies of module M_1. The values of the parameters α and β, as well as the output weights are given in Table 1.

The reconstruction of the original signal is obtained by using the mirror filters $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{G}}$ of $\mathbf{H}$ and $\mathbf{G}$ [1]. The architectural implementation of the reconstruction filter bank is obtained from the analysis structure by simply replacing the parameters α and β by $\pi - \alpha$ and $-\beta$ respectively.

Note that the procedure we have discussed insofar for realizing the 2-band lossless QMF bank can be used without any alterations for the case of K bands with K taking arbitrary values.

Concerning the implementation cost, module M_0 in Fig. 2 requires 2 multipliers, 3 adders and one rotation circuit. For the implementation of module M_1 we need 2 adders. We implement the desired pair of mapping operators as two weighted sums of the outputs of the above described parts. If the size of the associated kernel group is M the cost of this interconnection is $2M$ multipliers and $2(M-1)$ adders. The overall cost of the design is not higher than $3M$ multipliers, $\lfloor 7M/2 \rfloor$ adders and $M/2$ rotation circuits. For example, the circuit in Fig. 3 requires 15 multipliers, 20 adders and 2 rotation circuits. Apparently, the cost of the time-recursive realization does not depend on the filter length N, but rather on the number of decomposition kernels M, as we conjectured in part I [7].

Since the design we present in this Section is optimal (with respect to kernel group size M) it conveys information on whether the time-recursive computation is appropriate for a given lossless QMF bank and more general for a mapping operator. Nevertheless, there are special cases of lossless QMF banks exhibiting a structure that can be exploited by the use of time-recursive computation in different ways. An example of such filter banks is the cosine modulated QMF banks presented in the following Section.

4 Cosine Modulated PR QMF Banks

The cosine modulated Perfect Reconstruction (PR) QMF banks exhibit a performance characteristics similar to the other QMF banks with an easier design and cost effective algorithmic implementations [27, 22]. In this Section, we address the problem of time-recursive realization of the modulation matrix. This modification suppresses the global communication requirement of the QMF bank, giving rise to an architecture very appropriate for VLSI implementation. On the other hand, this second design example serves the purpose of demonstrating the use of the IIR building modules.

4.1 Preliminaries

An N-band cosine modulated PR QMF bank is characterized by the following property: the filters $H_0(z), H_1(z), \cdots, H_{N-1}(z)$ in the analysis bank are modulated versions of a symmetric prototype filter $P_0(z)$. A cosine matrix is used for performing the modulation operation. More specifically, the analysis filters are:

$$H_k(z) = \sum_{n=0}^{2N-1} c_{kn} z^{-n} E_n(-z^{2N}), \quad k = 0, 1, \cdots, N-1,$$

where $E_n(z), n = 0, 1, \cdots, 2N-1$ are the $2N$ polyphase components of a prototype filter $P_0(z)$ and the coefficients c_{kn} are

$$c_{kn} = 2\cos\left[\frac{\pi}{N}\left(k + \frac{1}{2}\right)\left(n - \frac{L-1}{2}\right) + \theta_k\right], \tag{3}$$

where $k = 0, 1, \cdots, N-1$, $n = 0, 1, \cdots, 2N-1$ and $\theta_k = (-1)^k \frac{\pi}{4}$ [27]. If the length L of the prototype filter $P_0(z)$ is an even multiple of the filter bank of size N, then the modulation matrix can be implemented based on a fast Discrete Cosine Transform algorithm. The resulting QMF analysis structure is depicted in Fig. 4, where the polyphase components $E_k(z), E_{k+N}(z), k = 0, 1, \cdots, N-1$ need to be pairwise lossless. The data processing evolves at the minimum rate, that is N times lower than the input data rate.

4.2 Time-Recursive Computation of the Cosine Modulation Matrix

For the cosine modulation matrix specified by (3) we need to implement N mapping operators $\mathbf{c}_k$ each one of length $2N$

$$\mathbf{c}_k = \left[c_{k0}\ c_{k1}\ \cdots\ c_{k,2N-1}\right], \qquad k = 0, 1, \cdots, N-1.$$

Since we have

$$\begin{bmatrix} \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n-1-\frac{L-1}{2}\right)+\theta_k\right] \\ \sin\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n-1-\frac{L-1}{2}\right)+\theta_k\right] \end{bmatrix} =$$

$$\begin{bmatrix} \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right) & \sin\frac{\pi}{N}\left(k+\frac{1}{2}\right) \\ -\sin\frac{\pi}{N}\left(k+\frac{1}{2}\right) & \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right) \end{bmatrix} \cdot \begin{bmatrix} \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n-\frac{L-1}{2}\right)+\theta_k\right] \\ \sin\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n-\frac{L-1}{2}\right)+\theta_k\right] \end{bmatrix}$$

the kernel group $\mathbf{f}_k(n)$ associated to the k^{th} mapping operator

$$\mathbf{f}_k(n) = \begin{bmatrix} f_{k,0}(n) \\ f_{k,1}(n) \end{bmatrix} = \begin{bmatrix} \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n-\frac{L-1}{2}\right)+\theta_k\right] \\ \sin\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n-\frac{L-1}{2}\right)+\theta_k\right] \end{bmatrix}, \qquad \theta_k = (-1)^k\frac{\pi}{4}$$

satisfies the shift property (cf. Subsection I.3.1). Thus, the k^{th} mapping operator is specified by the expression $c_{k,n} = f_{k,0}(n)$. This completes Step 1 of the design procedure. For the kernel group $\mathbf{f}_k(\cdot)$ we have $\mathbf{f}_k(n-1) = \mathbf{R}_k\mathbf{f}_k(n)$, where

$$\mathbf{R}_k = \begin{bmatrix} \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right) & \sin\frac{\pi}{N}\left(k+\frac{1}{2}\right) \\ -\sin\frac{\pi}{N}\left(k+\frac{1}{2}\right) & \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right) \end{bmatrix}$$

and

$$\mathbf{f}_k(0) = S\mathbf{f}_k(2N) = \begin{bmatrix} \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\frac{L-1}{2}-\theta_k\right] \\ -\sin\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\frac{L-1}{2}-\theta_k\right] \end{bmatrix},$$

where $S = -1$. Consequently, $\mathbf{f}_k(n)$ satisfies the periodicity property (cf. Subsection I4.2). Also, since only one of the two kernel functions in $\mathbf{f}_k(\cdot)$ is sufficient for our computation (that is $f_{k,0}(\cdot)$), the flow graph in Fig. 1 dictates that the IIR architecture should be adopted. The latter is derived based on Fig. I.8 and (I.19) and it is shown in Fig. 5.

4.3 Modified Realization of Cosine Modulated QMF Bank

The circuit inside the dotted area in Fig. 4 can be implemented with a bank of IIR modules $M_0, M_1, \cdots, M_{N-1}$ preceded by a switch as depicted in Fig. 6. This switch operates at twice the input data rate and it periodically repeats the schedule $0, 2, \cdots, 2N-2, 1, 3, \cdots, 2N-1$. One of the three multipliers in the module M_k, $k = 0, 1, \cdots, N-1$ needs to operate at the same rate, while the remaining two multipliers, as well as the polyphase components $E_0(-z^2), E_1(-z^2), \cdots, E_{2N-1}(-z^2)$, operate at the minimum data rate. This non-uniformity of the data processing rate can be compensated by using a parallel implementation for the multiplier that operates at the higher rate, while using digit serial operators [10] for all the other components of the circuit.

The modified realizations of the cosine modulated filter banks we described have a number of advantages:

1. They maintain the locality throughout the realization of the filter bank, that is very important in VLSI implementations, since a short clock cycle can be used.

2. They are modular, regular, scalable and they imply linear implementation cost, as opposed to more irregular design and quadratic cost for the fast algorithm based approaches [26]. Consequently, the VLSI implementation of filter banks with large number of filters N becomes more feasible.

3. The design is efficient for arbitrary values of the filter bank size N, unlike the fast algorithm designs that are efficient only for N being a power of 2.

In conclusion, we anticipate that a single-chip implementation of the cosine modulated PR QMF bank will be feasible based on the time-recursive realization of the modulation matrix.

5 Extended Lapped Transforms

A subfamily of the cosine modulated PR QMF banks is the family of the Extended Lapped Transforms (ELT) [21, 22]. In this case of QMF banks the coefficients of the prototype design filter $P_0(z)$ are sampled sinusoids, resulting into a very efficient algorithmic implementation of the associated filter bank. In this Section, we show that the special structure of the prototype filter $P_0(z)$ can be exploited by the time-recursive design principles, resulting in very efficient architectural implementations too. At the same time, we continue the demonstration of the design procedure by providing examples of time-recursive architectures utilizing lattice modules. We also demonstrate the use of time-recursive computation for realizing inverse block transforms.

5.1 Preliminaries

The most popular member of the ELT family is the Modulated Lapped Transform (MLT) [20, 22], oftentimes called Modified Discrete Cosine Transform (MDCT) in the data compression community [13]. The basis functions of the N-point MLT constitutes an N-band PR cosine modulated QMF bank, for which the prototype filter is a sinusoidal function of length equal to $2N$ [22] or equivalently, the polyphase components in Fig. 4 are length-1 FIR filters. The magnitude responses of the MLT basis functions exhibit a stopband attenuation of approximately $24dB$ [22], yielding high capability of resolving weak signals despite the presence of stronger ones. Furthermore, being a Lapped Orthogonal Transform (LOT), the MLT diminishes *the blocking effect* that appears at low bit rate data coding with transform techniques. The MLT has been incorporated by the ISO-MPEG and ASPEC standards (with the name MDCT) for audio data compression [14] and it is also used for image data compression [13, 22, 23].

The performance characteristics of the MLT can be further enhanced by using an Extended Lapped Transform (ELT) with longer kernel functions [21, 22]. For example, an N-point ELT with basis length equal to $4N$ has been used by a high fidelity audio coding scheme [29]. This ELT can be viewed as a cosine modulated QMF bank with length-2 polyphase components, generated by a prototype filter $P_0(z)$ of length $4N$. What is important with the time-recursive design we propose is that the increased length of the prototype filter translates mostly in communication cost rather than in increasing the operator counts, while communication locality is maintained.

5.2 Architecture for the Forward MLT

The MLT operates data segments of length $2N$, $x(t+n-2N+1), n = 0, 1, \cdots, 2N-1$ and it produces N output coefficients $X_{MLT}(k,t), k = 0, 1, \cdots, N-1$ as follows [20]:

$$X_{MLT}(k,t) = c_k\sqrt{\frac{2}{N}}\sum_{n=0}^{2N-1} \sin\frac{\pi}{2N}\left(n+\frac{1}{2}\right)\cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{N}{2}\right)\right] x(t+n-2N+1), \quad (4)$$

where $t = 0, 1, \cdots$ and $c_k = (-1)^{(k+2)/2}$ if k is even and $c_k = (-1)^{(k-1)/2}$ if k is odd. The sequence of the k^{th} output coefficients $X_{MLT}(k,t), t = 0, 1, \cdots$ can be thought of as the output of the mapping operator

$$h_{k,n} = \left(c_k\sqrt{\frac{2}{N}}\right)\sin\frac{\pi}{2N}\left(n+\frac{1}{2}\right)\cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{N}{2}\right)\right]. \quad (5)$$

After a few algebraic manipulations, we derive the following decomposition of this mapping operator

$$h_{k,n} = -\left(c_k\sqrt{\frac{1}{2N}}\right) f_{k+1,0}(n) - \left(c_k\sqrt{\frac{1}{2N}}\right) f_{k,1}(n), \quad k = 0, 1, \cdots, N-1, \quad (6)$$

where

$$\begin{bmatrix} f_{k,0}(n) \\ f_{k,1}(n) \end{bmatrix} = \begin{bmatrix} \cos\left[\frac{\pi}{N}k\left(n+\frac{1}{2}\right)+\left(k+\frac{1}{2}\right)\frac{\pi}{2}\right] \\ \sin\left[\frac{\pi}{N}k\left(n+\frac{1}{2}\right)+\left(k+\frac{1}{2}\right)\frac{\pi}{2}\right] \end{bmatrix} \triangleq \mathbf{f}_k(n)$$

is the associated kernel group. For this kernel group we have $\mathbf{f}_k(n-1) = \mathbf{R}_k \mathbf{f}_k(n)$, where

$$\mathbf{R}_k = \begin{bmatrix} \cos\frac{k\pi}{N} & \sin\frac{k\pi}{N} \\ -\sin\frac{k\pi}{N} & \cos\frac{k\pi}{N} \end{bmatrix}. \tag{7}$$

We also have

$$\begin{bmatrix} f_{k,0}(0) \\ f_{k,1}(0) \end{bmatrix} = S \begin{bmatrix} f_{k,0}(2N) \\ f_{k,1}(2N) \end{bmatrix} = \begin{bmatrix} \cos\left[\frac{k\pi}{2N} + \left(k+\frac{1}{2}\right)\frac{\pi}{2}\right] \\ \sin\left[\frac{k\pi}{2N} + \left(k+\frac{1}{2}\right)\frac{\pi}{2}\right] \end{bmatrix}, \tag{8}$$

where $S = 1$. Therefore, the periodicity property is satisfied. Since both member functions of the kernel group appear in the decomposition (6), the lattice architecture is recommended (cf. Fig. 1). In Fig. 7, we provide the lattice module that is used as the building block in the MLT architectural implementation. The latter is depicted in Fig. 8 for the case of $N = 8$. Rotation circuits are suggested for the implementation of the lattice structure [24].

5.3 Architecture for the Inverse MLT

The inverse MLT (IMLT) is specified by transposing the coefficient matrix of the direct transform. In close form, the IMLT is

$$x(Nt+n+1) = \sqrt{\frac{2}{N}} \sin\frac{\pi}{2N}\left(n+\frac{1}{2}\right) \sum_{k=0}^{N-1} c_k \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{N}{2}\right)\right] X_{MLT}(k, Nt),$$

for $t = 1$ and

$$x(Nt+n+1) = \sqrt{\frac{2}{N}} \sin\frac{\pi}{2N}\left(n+\frac{1}{2}\right) \sum_{k=0}^{N-1} c_k \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{N}{2}\right)\right] X_{MLT}(k, Nt) +$$

$$\sqrt{\frac{2}{N}} \sin\frac{\pi}{2N}\left(n+\frac{1}{2}+\frac{N}{2}\right) \sum_{k=0}^{N-1} c_k \cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{3N}{2}\right)\right] X_{MLT}(k, N(t-1)),$$

for $t \geq 2$. Since we have

$$\cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{3N}{2}\right)\right] = (-1)^{k+1}\sin\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{N}{2}\right)\right]$$

we obtain

$$x(Nt+n+1) = \begin{cases} \sqrt{\frac{2}{N}}\sin\frac{\pi}{2N}\left(n+\frac{1}{2}\right)\sum_{k=0}^{N-1} c_k y_{k,0}(Nt+n+1), & t=1 \\ \begin{array}{l} \sqrt{\frac{2}{N}}\sin\frac{\pi}{2N}\left(n+\frac{1}{2}\right)\sum_{k=0}^{N-1} c_k y_{k,0}(Nt+n+1)+ \\ \sqrt{\frac{2}{N}}\sin\frac{\pi}{2N}\left(n+\frac{1}{2}+\frac{N}{2}\right)\sum_{k=0}^{N-1}(-1)^{k+1} c_k y_{k+1,1}(N(t-1)+n+1) \end{array}, & t \geq 2, \end{cases}$$

where

$$\begin{bmatrix} y_{k,0}(Nt+n+1) \\ y_{k,1}(Nt+n+1) \end{bmatrix} = \mathbf{f}_k(n) X_{MLT}(k,Nt) \triangleq \mathbf{y}_k(Nt+n+1) \tag{9}$$

and

$$\mathbf{f}_k(n) = \begin{bmatrix} f_{k,0}(n) \\ f_{k,1}(n) \end{bmatrix} = \begin{bmatrix} \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{N}{2}\right) \\ \sin\frac{\pi}{N}\left(k+\frac{1}{2}\right)\left(n+\frac{1}{2}+\frac{N}{2}\right) \end{bmatrix}. \tag{10}$$

For the kernel group $\mathbf{f}_k(\cdot)$ we have

$$\mathbf{f}_k(n-1) = \mathbf{R}_k \mathbf{f}_k(n), \quad n = 1, 2, \cdots, N, \tag{11}$$

with

$$\mathbf{R}_k = \begin{bmatrix} \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right) & \sin\frac{\pi}{N}\left(k+\frac{1}{2}\right) \\ -\sin\frac{\pi}{N}\left(k+\frac{1}{2}\right) & \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right) \end{bmatrix} \tag{12}$$

and

$$\begin{bmatrix} f_{k,0}(0) \\ f_{k,1}(0) \end{bmatrix} = \begin{bmatrix} \cos\frac{(N+1)\pi}{2N}\left(k+\frac{1}{2}\right) \\ \sin\frac{(N+1)\pi}{2N}\left(k+\frac{1}{2}\right) \end{bmatrix}. \tag{13}$$

Consequently, the shift property is satisfied.

In the sequel, we show how we can evaluate the expression in (9) in a time-recursive way. For the sake of a simpler notation, we drop the subscript k except when it is necessary for avoiding confusion.

From (11) we have

$$\mathbf{f}(n+1) = \tilde{\mathbf{R}}\mathbf{f}(n), \quad n = 0, 1, \cdots, N-1, \tag{14}$$

where

$$\tilde{\mathbf{R}} \triangleq \mathbf{R}^{-1} = \begin{bmatrix} \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right) & -\sin\frac{\pi}{N}\left(k+\frac{1}{2}\right) \\ \sin\frac{\pi}{N}\left(k+\frac{1}{2}\right) & \cos\frac{\pi}{N}\left(k+\frac{1}{2}\right) \end{bmatrix}.$$

Also, (9) is equivalent to

$$\mathbf{y}(Nt+n) = \mathbf{f}(n-1)X(0, Nt), \quad t = 1, 2, \cdots, \quad n = 1, 2, \cdots, N, \tag{15}$$

where $X(0,t), t = 0, 1, \cdots$ is the transform sequence that corresponds to the 0^{th} kernel function of $\mathbf{f}(\cdot)$. (14) and (15) imply

$$\mathbf{y}(Nt+n+1) = \tilde{\mathbf{R}}X(0, Nt)$$

and therefore, the quantities $\mathbf{y}(Nt+n+1), n = 0, 1, \cdots, N-1$ can be evaluated by the following time-recursive algorithm:

$$\mathbf{y}(Nt+1) = \mathbf{f}(0)X(0, Nt) \tag{16}$$

$$\mathbf{y}(Nt+n+1) = \tilde{\mathbf{R}}\mathbf{y}(Nt+n), \quad n = 1, 2, \cdots, N-1. \tag{17}$$

This algorithm can be implemented by the lattice architecture in Fig. 9. Note that the parameters $\tilde{r}_{ij}$, $i = 0, 1,\ j = 0, 1$ are the elements of the matrix $\tilde{\mathbf{R}}$. In particular, the lattice module for IMLT is depicted in Fig. 10. The architectural implementation of the overall IMLT for $N = 8$ is shown in Fig. 11.

5.4 Architecture for Forward ELT

In this Subsection, we consider the ELT [2] that operates on segments of data of length $4N$, $x(t+n-4N+1), n = 0, 1, \cdots, 4N-1$ and it produces N output coefficients $X_{ELT}(k,t), k = 0, 1, \cdots, N-1$ as follows [22]:

$$X_{ELT}(k,t) = \sqrt{\frac{2}{N}} \sum_{n=0}^{4N-1} \left[-\frac{1}{2\sqrt{2}} + \frac{1}{2} \cos \frac{\pi}{N} \left(n + \frac{1}{2} \right) \right] \cos \left[\frac{\pi}{N} \left(k + \frac{1}{2} \right) \left(n + \frac{1}{2} + \frac{N}{2} \right) \right] x(t+n-4N+1), \tag{18}$$

where $t = 0, 1, \cdots$. The sequence of the k^{th} output coefficients $X_{ELT}(k,t), t = 0, 1, \cdots$ can be thought of as the output of the mapping operator

$$h_{k,n} = \frac{1}{2\sqrt{2N}} \left[\alpha + 2 \cos \frac{\pi}{N} \left(n + \frac{1}{2} \right) \right] \cos \left[\frac{\pi}{N} \left(k + \frac{1}{2} \right) \left(n + \frac{1}{2} \right) + \left(k + \frac{1}{2} \right) \frac{\pi}{2} \right], \tag{19}$$

where $\alpha = -\sqrt{2}$. After a few algebraic manipulations, we derive the following decomposition of the mapping operator:

$$h_{k,n} = \frac{1}{2\sqrt{2N}} \left[-f_{k-1,1}(n) + \alpha f_{k,0}(n) + f_{k+1,1}(n) \right], \quad k = 0, 1, \cdots, N-1, \tag{20}$$

where

$$\begin{bmatrix} f_{k,0}(n) \\ f_{k,1}(n) \end{bmatrix} = \begin{bmatrix} \cos \left[\frac{\pi}{N} \left(k + \frac{1}{2} \right) \left(n + \frac{1}{2} \right) + \left(k + \frac{1}{2} \right) \frac{\pi}{2} \right] \\ \sin \left[\frac{\pi}{N} \left(k + \frac{1}{2} \right) \left(n + \frac{1}{2} \right) + \left(k + \frac{1}{2} \right) \frac{\pi}{2} \right] \end{bmatrix}$$

is the associated kernel group $\mathbf{f}_k(n)$. For this kernel group we have $\mathbf{f}_k(n-1) = \mathbf{R}_k \mathbf{f}_k(n)$, where

$$\mathbf{R}_k = \begin{bmatrix} \cos \frac{\pi}{N} \left(k + \frac{1}{2} \right) & \sin \frac{\pi}{N} \left(k + \frac{1}{2} \right) \\ -\sin \frac{\pi}{N} \left(k + \frac{1}{2} \right) & \cos \frac{\pi}{N} \left(k + \frac{1}{2} \right) \end{bmatrix}. \tag{21}$$

[2] Although the Extended Lapped Transforms are a family of transforms (including MLT), in the sequel we shall reserve the name ELT discussed in this Subsection.

We also have

$$\begin{bmatrix} f_{k,0}(0) \\ f_{k,1}(0) \end{bmatrix} = S \begin{bmatrix} f_{k,0}(4N) \\ f_{k,1}(4N) \end{bmatrix} = \begin{bmatrix} \cos\frac{\pi}{2}\left(1+\frac{1}{N}\right)\left(k+\frac{1}{2}\right) \\ \sin\frac{\pi}{2}\left(1+\frac{1}{N}\right)\left(k+\frac{1}{2}\right) \end{bmatrix}, \tag{22}$$

where $S = 1$. Therefore, the periodicity property is satisfied. Since both member functions of the kernel group appear in the decomposition (20), the lattice architecture is recommended (cf. Fig. 1). In Fig. 12, we provide the time-recursive architecture for the case of $N = 8$. The details of the lattice modules in Fig. 12 can be obtained by a simple parameter substitution.

5.5 Cost Issues

The cost of the time-recursive implementation of the MLT is $N-1$ rotation circuits, $2N+3$ multipliers and $3N+3$ adders, while the cost of the time-recursive implementation of the ELT is $N+2$ rotation circuits, $3N+4$ multipliers, and $4N+4$ adders.

For the derivation and proper interpretation of the throughput rate, we need to consider the data model from a closer perspective. In the applications of interest, such as communication of audio, video, sonar and radar data, the input is provided in a serial way. Let us denote with u the time-unit, that is the time lapsing between two adjacent input data. Let us recall also, that the locality property of the time-recursive design makes the bit-parallel implementation of the arithmetic operators feasible. We will denote with R_P, M_P and A_P the time required for a rotation, a multiplication and an addition respectively, as opposed to R_S, M_S and A_S that will denote the time needed for the same operations when implemented in a bit-serial way. Note that the architectures based on fast algorithms usually must employ serial implementations because of the extensive use of global communication. For real-time applications the throughput rate th, that is the number of data samples to be processed per time-unit, needs to be equal to $th = 1$. Otherwise, if $th < 1$, the circuit will not be able to handle all the arriving data, so storage of data and off-line processing will be needed. On the other hand, if $th > 1$, the circuit will be periodically idle since no input data will be available. Under this light of timing information and throughput requirements we will next consider the MLT implementation.

First, for the time-recursive architecture of the sliding MLT circuit the data flow of the structures in Fig. 7 and Fig. 8 dictate that between two adjacent output data samples one multiplication, three additions and one rotation have to be performed. By introducing buffers at the points A, B, C and D in Fig. 7 pipeline processing is possible, so that the computation time is specified by

$$\max\{M_P + A_P, A_P + R_P\}.$$

Consequently, the real-time processing requirement is equivalent to

$$th = \frac{u}{\max\{M_P + A_P, A_P + R_P\}} = 1.$$

In other words, the implementation should meet the constraint

$$\max\{M_P + A_P, A_P + R_P\} = u. \tag{23}$$

Furthermore, if this is true the latency of the circuit will be equal to $3Nu$. Consider now the fast algorithm design for the sliding MLT that has an inherently parallel-input parallel-output nature [20, 22]. At each time instant an input data arrives, and a sliding window of length N moves to the next position. The real-time processing requirement dictates that before the next input data arrives an N-point MLT must be performed. An architecture based fast transform consists of a sequence of $\log_2 N + 1$ alternating stages of multiply-add pairs and butterfly interconnection networks. The latency time from input to output for a fully parallel structure will be equal to $(\log_2 N + 1)(M_S + A_S)$. For a fully parallel and pipelined structure each stage can operate on a different set of data, so at a time interval equal to the latency time above a number of output data sets equal to $\log_2 N + 1$ will be available. Therefore, the real-time processing requirement will be equivalent to

$$M_S + A_S = u. \tag{24}$$

A comparison of (23) with (24), as well as the latencies of the time-recursive and the architecture based on a fast algorithm yields a strong superiority to the former. In other words, (23) is substantially easier to meet than (24). There are two reasons for this:

1. The locality property of the time-recursive architecture allows a shorter internal clock cycle than the one employed by the architecture based on a fast algorithm since the latter makes extensive use of global interconnections.

2. Even if the same clock cycle intervals are used for the two circuits, the ratio $\frac{X_S}{X_P}$ will be of the order of the wordlength used in the finite word length implementation [11]. Here, X_S and X_P denote the time necessary to carry out an operation (either addition or multiplication) with a bit-serial and a bit-parallel implementation respectively. It worths mentioning that the use of distributed arithmetic [30, 24] in implementing bit-parallel operations has been proved to be very effective in the applications of interest [4]. In particular, it is possible to execute one multiplication per clock cycle with this approach.

For the time-recursive implementation of the block MLT the output of the circuit depicted in Fig. 7 needs to be decimated at the points C and D. By employing the pipeline processing described above we obtain the time lapsing between two adjacent output samples: $N \max\{M_P + A_P, A_P + R_P\}$. For real-time data processing the time available for this computation is equal to Nu. Consequently, the real-time processing requirement is again specified by (23). Also, the latency time is $3Nu$ as in the case of the sliding MLT. On the other hand, an architecture based on a fast transform operates as follows. The data arrive serially, and stored in a buffer of length N, which feeds the fully parallel and pipelined structure described above. This structure produces a new set of output data in a time interval equal to $M_S + A_S$. The available time for this computation is equal to Nu. Consequently, the real-time processing requirement dictates

$$M_S + A_S = Nu. \tag{25}$$

The latency time in the pipelined structure will be $(\log_2 N + 1)u$. The waiting time of the data in the input buffer needs to be added on the latter. This ranges from u to Nu with an average (approximately) $\frac{N}{2}u$. In total, the (average) latency time is $(\log_2 N + \frac{N}{2} + 1)u$.

The throughput rate constraints and the operator counts for both the MLT and ELT circuits are summarized in Table 2. As long as VLSI implementation is concerned, this information needs to be viewed in conjunction with the layout features of the associated circuits and their consequences listed in Subsection 4.3. In conclusion, the time-recursive approach suggests very competitive designs for the VLSI implementation of the MLT and ELT that have a significant impact on the real-time audio data compression [14, 29] and other real-time signal processing applications [22].

6 Conclusion

In the second part of this two part paper, a design procedure for time-recursive realization under different conditions has been presented and it has been applied to derive novel architectures for a number of PR QMF banks.

Given a linear, discrete time, time invariant, compactly supported but otherwise arbitrary mapping operator, a Generic Design Procedure we propose dictates first, whether the time-recursive computation is appropriate for a given mapping operator and second, how to derive an optimized time-recursive design. Furthermore, the flow graph of this design procedure provides the way for specifying the parameters of the available IIR and lattice building blocks and combining them into the time-recursive architecture.

The use of time-recursive computation in the cosine modulated QMF banks, as well as the MLT and ELT, yields architectures that are very appropriate for VLSI implementation. These architectures are modular, regular and they require only local communication. Also, they are very efficient in terms of area utilization because: first, they have linear requirements in operation counts, second, a substantial part of the computation is carried out by rotation circuits that can be implemented very efficiently [24] and third, the designs are based on locally interconnected building modules that are almost as low in number and complexity as the corresponding ones of the DCT [18]. Furthermore, they operate in a Single Input Multiple Output (SIMO) manner, that is very convenient for signal processing applications in communications. Exhibiting these features, the designs we propose have an impact in diverse applications in real-time signal processing, such as audio data compression, sampling rate conversion, channel equalization and echo cancellation. Furthermore, the combination of the aforementioned Generic Design Procedure with an induction procedure for realizing multi-dimensional computational cores [19] yields a very powerful technique for real-time multi-dimensional signal processing.

In summary, we have provided evidence that the time-recursive computation carries a potential (mainly in VLSI computing) that has not been fully exploited by the signal processing community yet. We also have developed a tool that facilitates the design of the time-recursive realizations, thus providing the means of exploiting this potential.

References

[1] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies. Image Coding Using Wavelet Transform. *IEEE Transactions on Signal Processing*, 1992.

[2] J. Canaris. A VLSI Architecture for the Real-Time Computation of Discrete Trigonometric Transforms. *Journal of VLSI Signal Processing*, 5(1):95–104, Jan. 1993.

[3] T.S. Chihara. *An Introduction to Orthogonal Polynomials.* Gordon and Breach Science Pub., New York, 1978.

[4] C.T. Chiu and K.J.R. Liu. Real-Time Parallel and Fully Pipelined Two-Dimentional DCT Lattice Structures with Application to HDTV Systems. *IEEE Transactions on Circuits and Systems for Video Technology*, 2(1):25–37, March 1992.

[5] E. Frantzeskakis. *An Architectural Framework for VLSI Time-Recursive Computation with Applications.* PhD thesis, The University of Maryland at College Park, 1993.

[6] E. Frantzeskakis, J.S. Baras, and K.J.R. Liu. Time-Recursive Architectures and Wavelet Transform. In *Proc. IEEE ICASSP*, pages I.445–448, 1993.

[7] E. Frantzeskakis, J.S. Baras, and K.J.R. Liu. Time-Recursive Computation and Real-Time Parallel Architectures, Part I: Framework. *submitted to IEEE Trans. on SP*, July 1993.

[8] E. Frantzeskakis, J.S. Baras, and K.J.R. Liu. Time-Recursive Computation and Real-Time Parallel Architectures, with Application on the Modulated Lapped Transform. In *Proc. SPIE, International Symposium on Optical Applied Science and Engineering, Advanced Signal Processing Algorithms, Architectures, and Implementations*, 1993.

[9] E. Frantzeskakis and H. Karathanasis. On Computing the 2-D Modulated Lapped Transform in Real-Time. In *Accepted in 1993 IEEE Workshop on VLSI Signal Processing*, 1993.

[10] R. Hartley and P. Corbett. Digit-Serial Processing Techniques. *IEEE Trans. on Circuits and Systems*, 37(6):707–719, June 1990.

[11] J.P. Hayes. *Computer Architecture and Organization.* McGraw-Hill,Inc., New York, second edition, 1988.

[12] Y.H. Hu. CORDIC-Based VLSI Architectures for Digital Signal Processing. *IEEE Signal Processing Magasine*, pages 16–35, July 1992.

[13] N. Jayant. Signal Compression: Technology Targets and Research Directions. *IEEE Journal on Selected Areas in Communications*, 10(5):796–818, June 1992.

[14] N. Jayant. Digital Coding of Wideband Audio. Technical Report Tutorial No3, Bell Labs, IEEE/ICASSP, Minessota, Minneapolis, April 1993. (Tutorial lecture notes).

[15] N.S. Jaynant and P. Noll. *Digital Coding of Waveforms.* Prentice Hall, Englewood Cliffs,NJ, 1984.

[16] T. Kailath. *Linear Systems.* Prentice Hall, London, 1980.

[17] S.Y. Kung. *Multivariable and Multidimentional Systems: Analysis and Design.* PhD thesis, Stanford University, June 1977.

[18] K.J.R. Liu and C.T. Chiu. Unified Parallel Lattice Structures for Time-Recursive Discrete Cosine/Sine/Hartley Transforms. *IEEE Trans. on SP*, 41(3):1357–1377, May 1993.

[19] K.J.R. Liu, C.T. Chiu, R.K. Kolagolta, and J.F. Jaja. Optimal Unified Architectures for the Real-Time Computation of Time-Recursive Discrete Sinusoidal Transforms. *Submitted to IEEE Transactions on Circuits and Systems for Video Technology*, 1992.

[20] H.S. Malvar. Lapped Transforms for Efficient Transform/Subband Coding. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(6):969–978, June 1990.

[21] H.S. Malvar. Extended Lapped Transforms: Properties, Applications, and Fast Algorithms. *IEEE Transactions on Signal Processing*, 40(11):2703–2714, Nov. 1992.

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

[23] E.M. Rubino and H.S. Malvar. Improved Chen-Smith Image Coder. In *Proc. IEEE ISCAS*, 1993.

[24] S.G. Smith and S.A. White. Hardware Approaches to Vector Plane Rotation. In *Proc. IEEE ICASSP*, pages 2128–2131, 1988.

[25] R. Stedman, H. Gharavi, L. Hanzo, and R. Steele. Transmission of Subband-Coded Images via Mobile Channels. *IEEE Trans. on Circuits and Systems for Video Technology*, 3(1):15–26, Feb. 1993.

[26] J. Ullman. *Computational Aspect of VLSI.* Computer Science Press, Rockville, MD, 1984.

[27] P.P. Vaidyanathan. *Multirate Filters and Filter Banks.* Signal Processing. Prentice Hall, Eglewood Cliffs, NJ, 1993.

[28] P.P. Vaidyanathan and Z. Doganata. The Role of Lossless Systems in Modern Digital Signal Processing: A Tutorial. *IEEE Transactions on Education*, 32(3):181–197, Aug. 1989.

[29] L.F.C. Vargas and H.S. Malvar. ELT-Based Wavelet Coding of High-Fidelity Audio Signals. In *Proc. IEEE ISCAS*, 1993.

[30] S.A. White. Applications of Distributed Arithmetic to Digital Signal Processing: A Tutorial Review. *IEEE Signal Processing Magasine*, 6:4–19, July 1989.

h_n	g_n	α or β	$weight_h$	$weight_g$
0.0267	0.0000	1.0	0.1781	-0.0032
-0.0168	0.0456	-1.0	-0.0056	0.1778
-0.0782	-0.0287	1.1085	-0.0881	-0.0235
0.2668	-0.2956		-0.3089	-0.0781
0.6029	0.5575	2.0929	-0.0575	-0.1511
0.2668	-0.2936		0.0998	0.2673
-0.0782	-0.0287			
-0.0168	0.0456			
0.0267	0.0000			

Table 1: Example of lossless QMF filter coefficients and the associated architecture parameters.

		rate constraint	implementation cost (mult, add, rotation)
MLT	time-rec.,sliding	$\max\{M_P + A_P, A_P + R_P\} = u$	$2N+3, 3N+3, N-1$
	time-rec.,block	$\max\{M_P + A_P, A_P + R_P\} = u$	$2N+3, 3N+2, N-1$
	fast algo.,sliding	$M_S + A_S = u$	$\frac{N}{2}(\log_2 N + 5)$, $\frac{3N}{2}\log_2 N + \frac{5N}{2}, 0$
	fast algo.,block	$M_S + A_S = Nu$	
ELT	time-rec.,sliding	$\max\{M_P + 2A_P, A_P + R_P\} = u$	$3N+4, 4N+4, N+2$
	time-rec.,block	$\max\{M_P + 2A_P, A_P + R_P\} = u$	$3N+4, 4N+3, N+2$
	fast algo.,sliding	$M_S + A_S = u$	$\frac{N}{2}(\log_2 N + 5)$, $\frac{3N}{2}\log_2 N + 4N, 0$
	fast algo.,block	$M_S + A_S = Nu$	

Table 2: Cost metrics for the architectural implementation of block transforms. M_P, A_P and R_P denote the time delays associated with a bit-parallel implementation of the multiplier, the adder and the rotation circuit respectively. M_S, A_S and R_S denote the corresponding time delays for a bit-serial implementation.

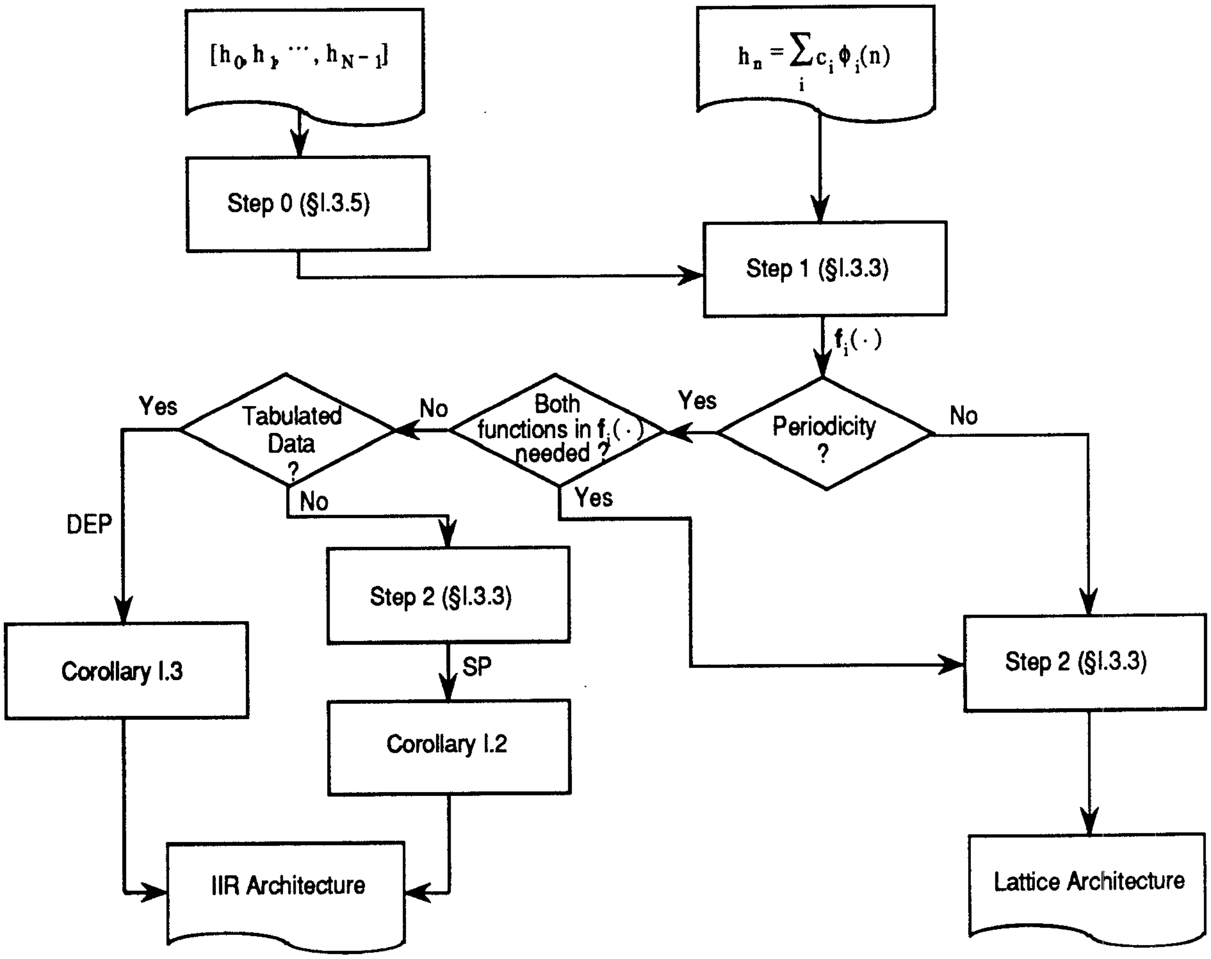


Figure 1: Architecture design procedure.

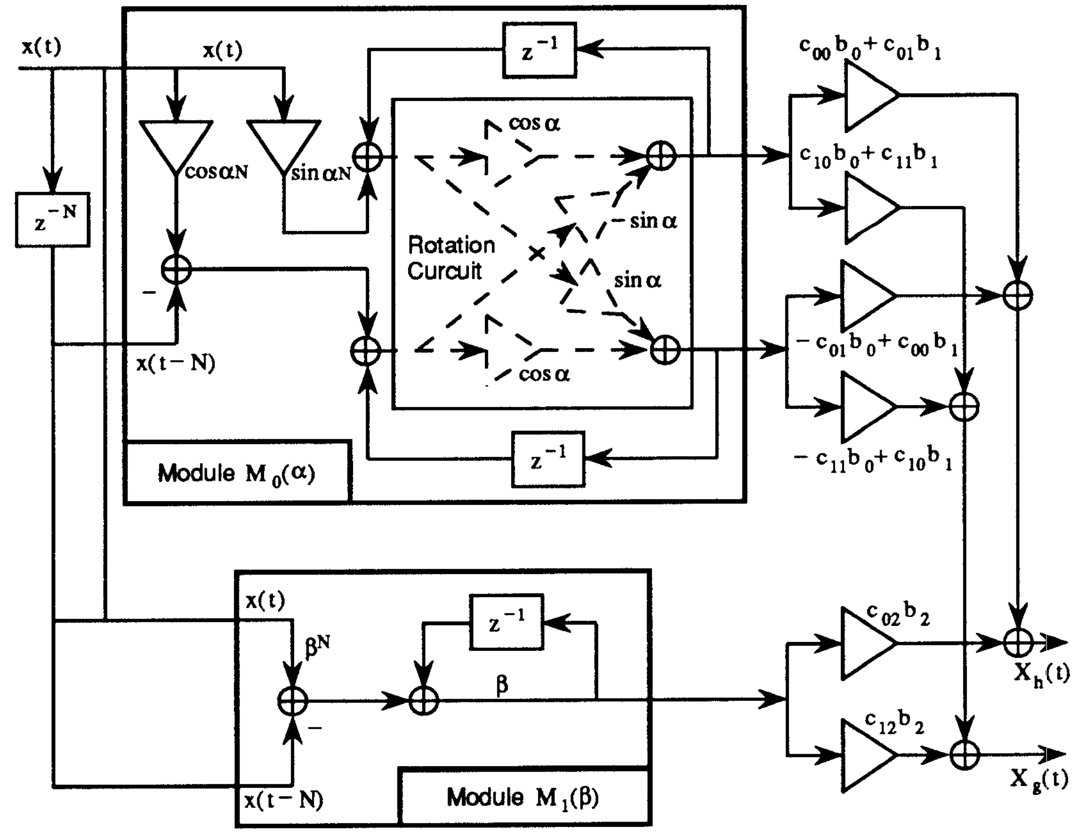


Figure 2: The architectural modules used for DWT.

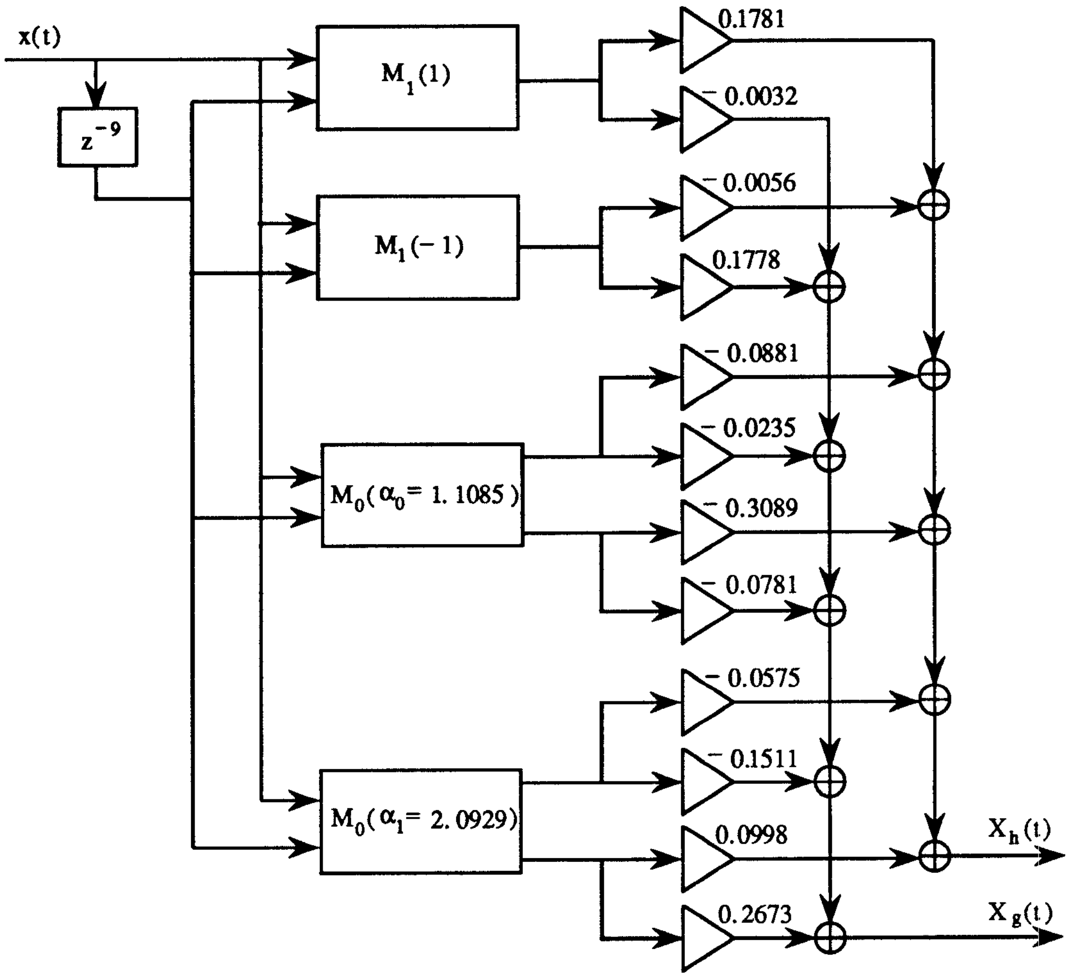


Figure 3: Architecture for the DWT filters specified in Table 1.

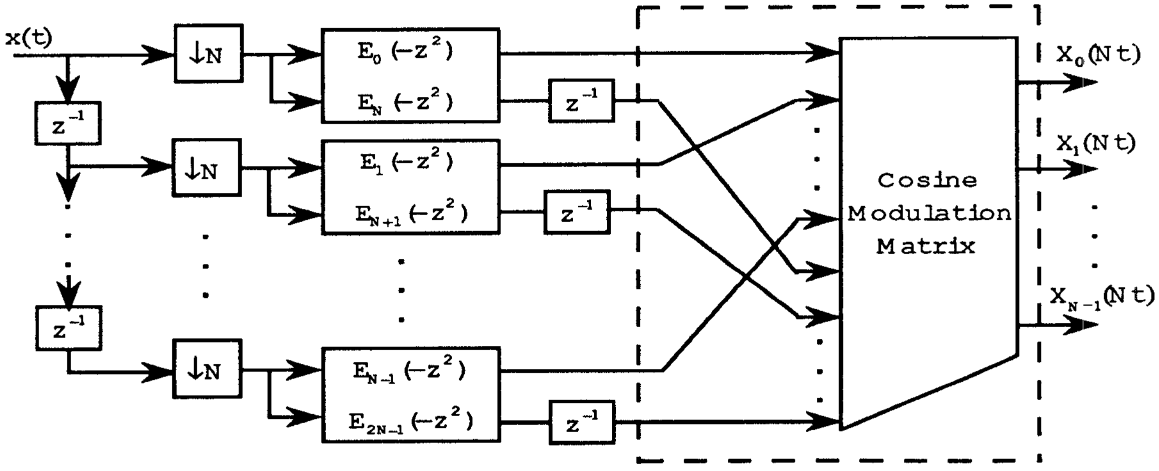
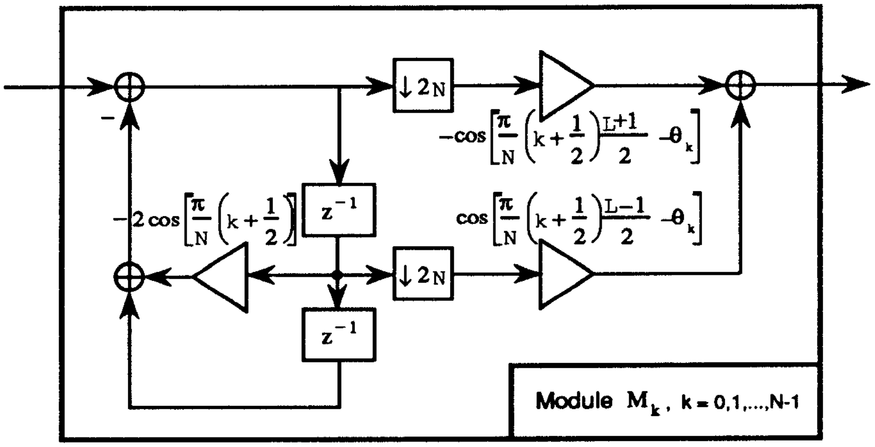


Figure 4: PR analysis bank based on cosine modulation.



↓2N
$-\cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\frac{L+1}{2}-\theta_k\right]$
−
$-2\cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\right]$
z^{-1}
$\cos\left[\frac{\pi}{N}\left(k+\frac{1}{2}\right)\frac{L-1}{2}-\theta_k\right]$
↓2N
z^{-1}
Module M_k, k = 0,1,...,N-1

Figure 5: Cosine modulation module

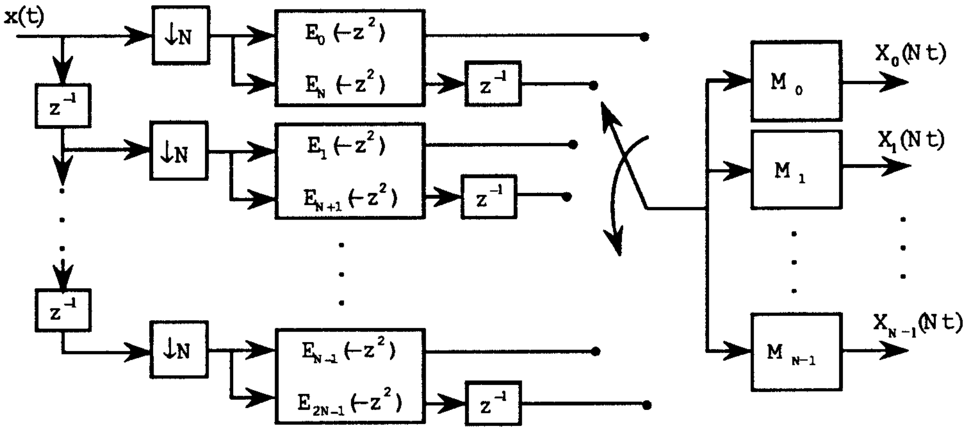
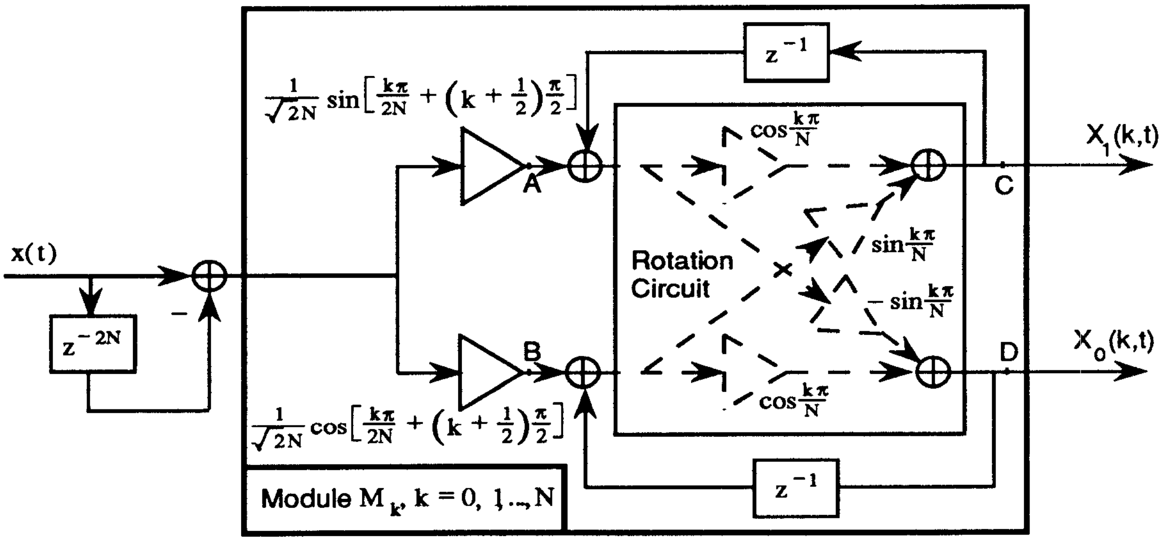


Figure 6: Modified realization of the cosine modulated PR QMF bank.



z^{-1}
$\frac{1}{\sqrt{2N}}\sin\left[\frac{k\pi}{2N}+\left(k+\frac{1}{2}\right)\frac{\pi}{2}\right]$
A
$\cos\frac{k\pi}{N}$
$X_1(k,t)$
C
x(t)
Rotation Circuit
$\sin\frac{k\pi}{N}$
$-\sin\frac{k\pi}{N}$
z^{-2N}
−
B
D
$X_0(k,t)$
$\cos\frac{k\pi}{N}$
$\frac{1}{\sqrt{2N}}\cos\left[\frac{k\pi}{2N}+\left(k+\frac{1}{2}\right)\frac{\pi}{2}\right]$
Module M_k, k = 0, 1..,N
z^{-1}

Figure 7: Lattice architecture for the MLT module.

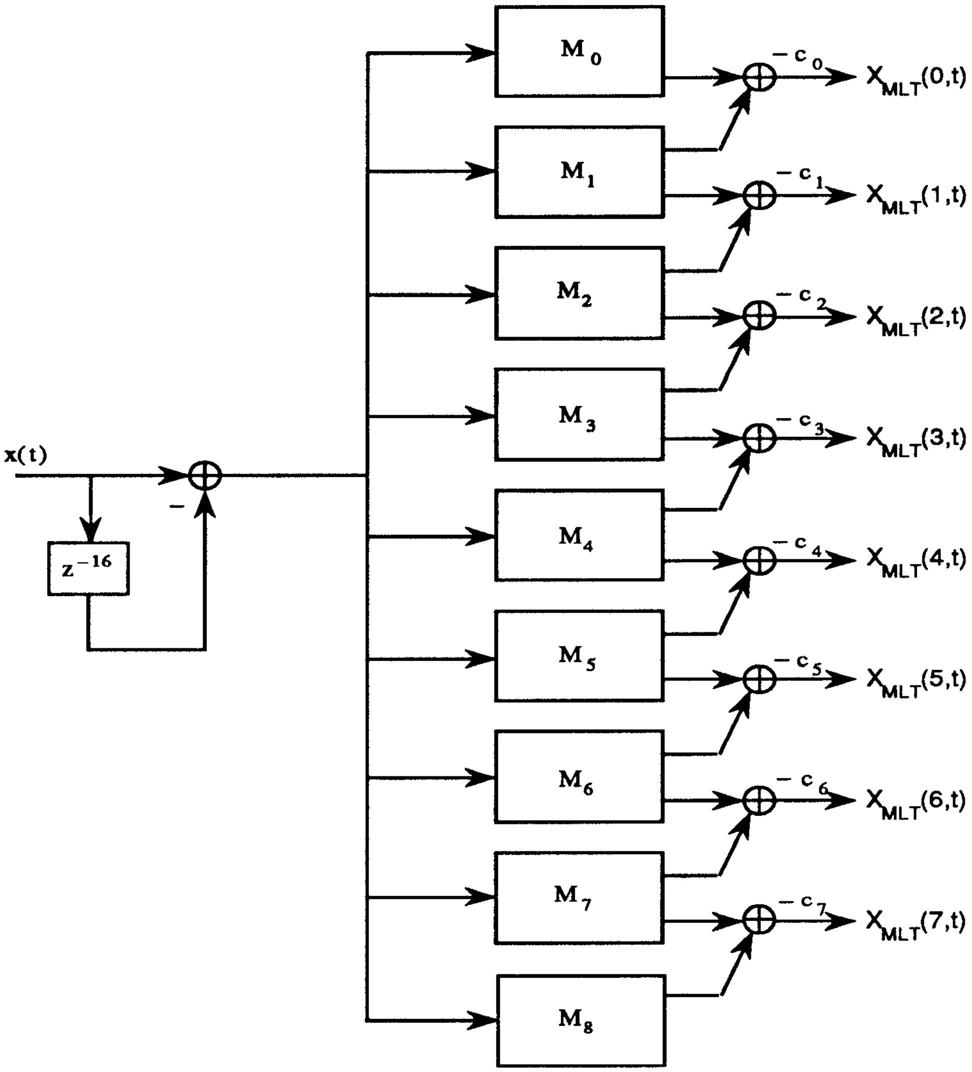


Figure 8: Time-recursive architecture for the MLT.

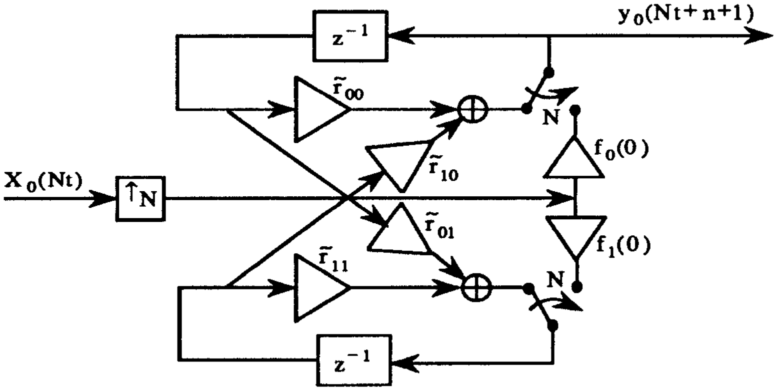


Figure 9: Inverse transform module.

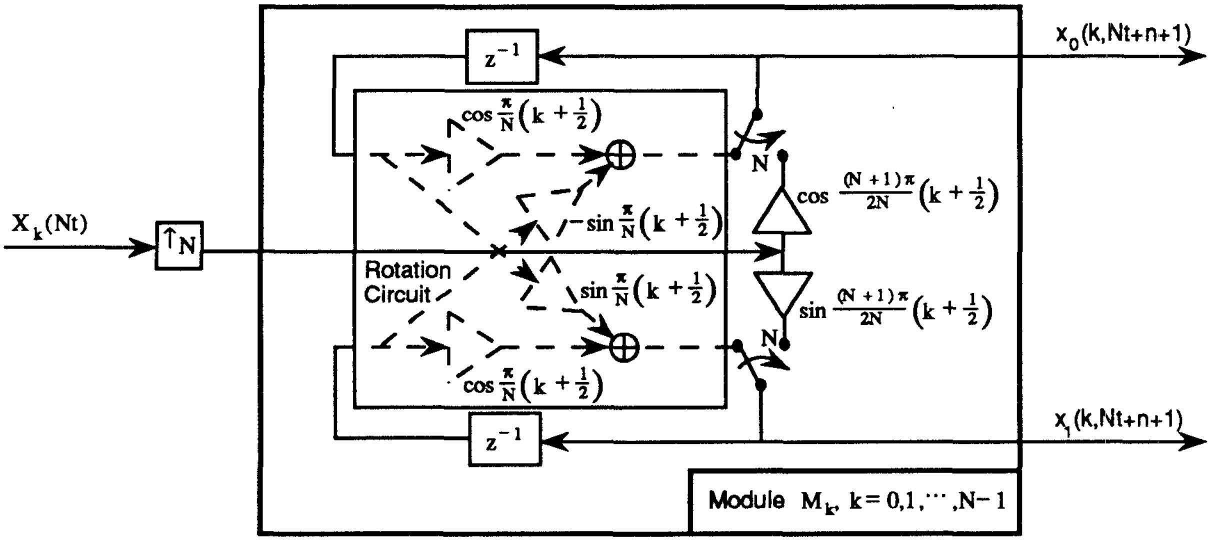

$X_k(Nt)$
$\uparrow N$
z^{-1}
$\cos\frac{\pi}{N}(k+\frac{1}{2})$
$-\sin\frac{\pi}{N}(k+\frac{1}{2})$
$\sin\frac{\pi}{N}(k+\frac{1}{2})$
$\cos\frac{\pi}{N}(k+\frac{1}{2})$
Rotation Circuit
N
N
$\cos\frac{(N+1)\pi}{2N}(k+\frac{1}{2})$
$\sin\frac{(N+1)\pi}{2N}(k+\frac{1}{2})$
$x_0(k,Nt+n+1)$
$x_1(k,Nt+n+1)$
Module M_k, $k=0,1,\cdots,N-1$

Figure 10: Lattice architecture for the IMLT module.

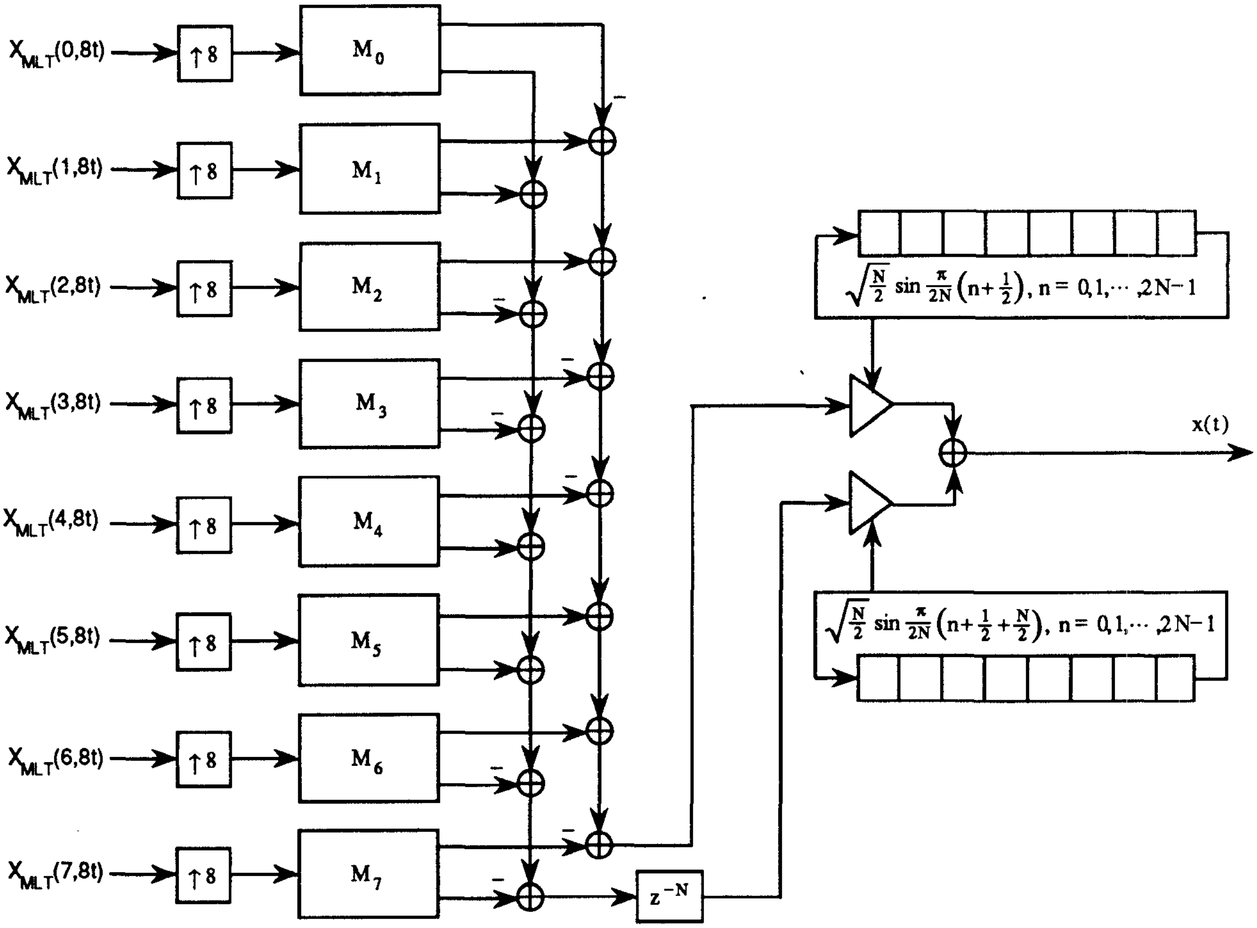


Figure 11: Time-recursive architecture for the IMLT.

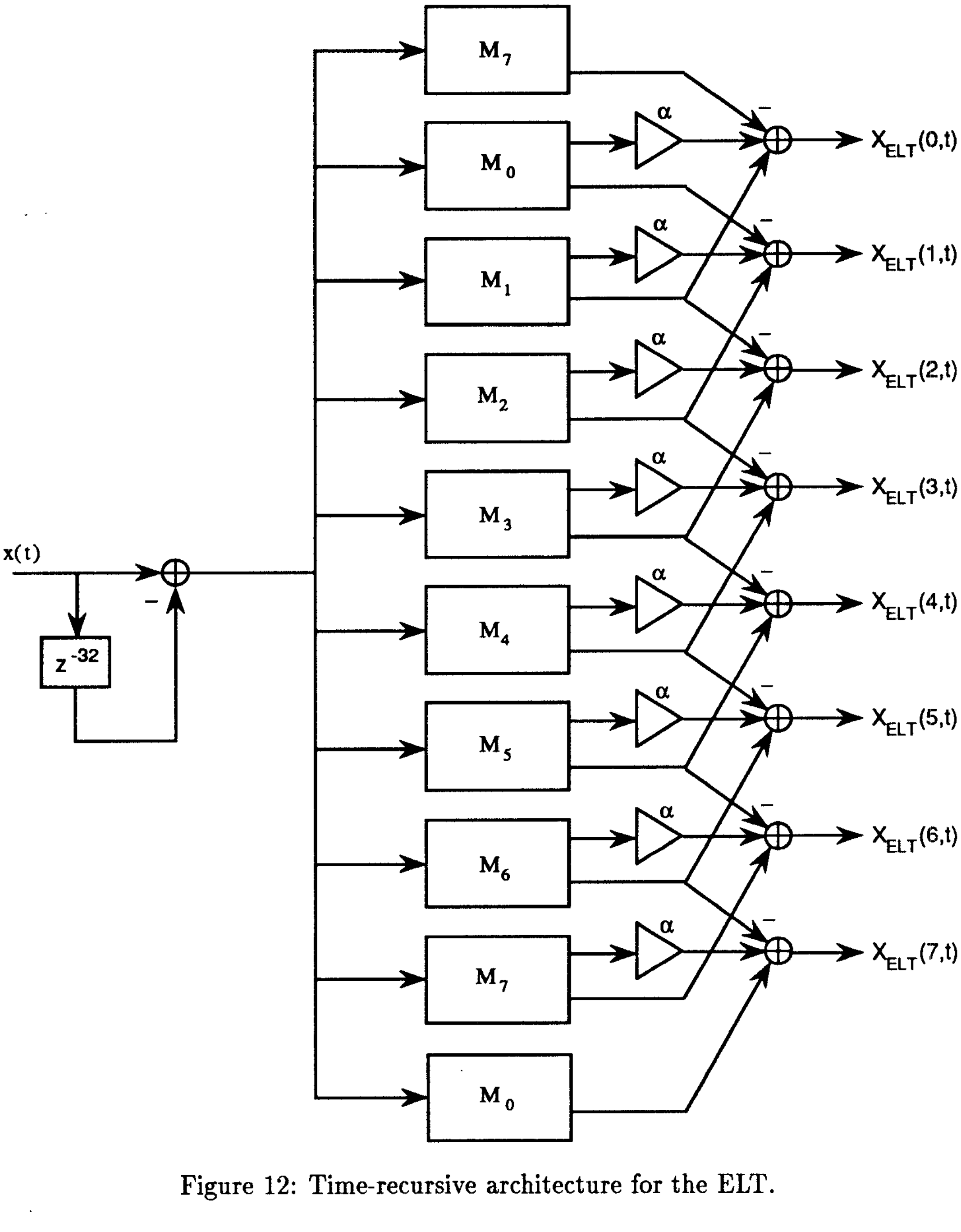


Figure 12: Time-recursive architecture for the ELT.