

Highly Efficient High-Speed/Low-Power Architectures for the 1-D Discrete Wavelet Transform

Francescomaria Marino, David Guevorkian, and Jaakko T. Astola

Abstract—In this paper, we propose two scalable architectures (say, Arc_J and Arc_2^*) that perform the discrete wavelet transform (DWT) of an N_0 -sample sequence in only $N_0/2$ clock cycles. Therefore, they are at least twice as fast as the other known architectures. Also, they have an AT^2 parameter that is approximately 1/2 that of already existing devices.

This result has been achieved by means of a carefully balanced pipelining, and it has two “faces.” First, Arc_J and Arc_2^* can be employed for performing two times faster processing than allowed by other architectures working at the same clock frequency (high-speed utilization). Second, they can be employed even using a two times lower clock frequency but reaching the same performance as other architectures. This second possibility allows for reducing the supply voltage and the power dissipation, respectively, by a factor of two and four with respect to other architectures (low-power utilization).

As a final result, we show that a parallel architecture implementing an L -tap filter-based DWT with J decomposition levels [say, $\text{Arc}_{\text{OPT}}(J, L)$] can be defined, aiming at having an excellent efficiency (say, $\text{eff}[\text{Arc}_{\text{OPT}}(J, L)]$) for any value of J and L . For instance, the average value of $\text{eff}[\text{Arc}_{\text{OPT}}(J, L)]$ [computed in very wide set Σ' of “points” (J, L)] is 99.1%. The minimum value of $\text{eff}[\text{Arc}_{\text{OPT}}(J, L)]$ in Σ' is 93.8%, and, except for five “points,” in all the others, $\text{eff}[\text{Arc}_{\text{OPT}}(J, L)]$ is not lower than 96.9%.

I. INTRODUCTION

THE DISCRETE wavelet transform (DWT) [1]–[4] is a mathematical technique that decomposes a signal in the time domain by using dilated/contracted and translated versions of a single basis function, named the prototype wavelet. In the last decade, the DWT has often been found preferable to other traditional signal-processing techniques since it offers useful features such as inherent scalability, computational complexity of $O(N_0)$ (where N_0 are the samples of the processed sequence), low aliasing distortion for signal-processing applications, and adaptive time-frequency windows. Hence, the DWT has been studied and applied to a wide range of applications: numerical analysis [5], [6], biomedicine [7], different branches of image and video processing [1], [8], [9], signal-processing techniques [10], speech compression/decompression [11], etc.

Manuscript received August 1999; revised October 2000. The work of F. Marino was supported by Tampere International Center for Signal Processing (TICSP), Tampere University of Technology. This paper was recommended by Associate Editor T. Stouraitis.

F. Marino is with the Dipartimento di Elettrotecnica ed Elettronica, Facoltà di Ingegneria, Politecnico di Bari, Bari 70125 Italy (e-mail: marino@iesi.ba.cnr.it).

D. Guevorkian is with Nokia Research Center, Tampere SF-33721 Finland (e-mail: david.guevorkian@nokia.com).

J. T. Astola is with the Signal Processing Laboratory, Tampere University of Technology, Tampere FIN-33101 Finland (e-mail: jta@cs.tut.fi).

Publisher Item Identifier S 1057-7130(00)11670-2.

In many of these applications, real-time performances are required in order to achieve attractive results. Therefore, the implementation of the DWT by means of dedicated VLSI application-specific integrated circuits has recently captivated the attention of a number of researchers, and many DWT architectures have already been proposed [12]–[25]. Some of these devices have been targeted to have a low hardware complexity, but they require at least $2N_0$ clock cycles (ccs) to compute the DWT of a sequence t^0 having N_0 samples (e.g., the devices proposed in [12]–[14], the architecture A2 in [15], etc.). Nevertheless, also a large number of devices, having a period of approximately N_0 ccs, has been designed (e.g., the three architectures in [14] when they are provided with a doubled hardware, the architecture A1 in [15], the architectures in [16]–[18], the parallel filter in [19], etc.). Most of these architectures exploit the recursive pyramid algorithm (RPA) [26] or similar scheduling techniques in order both to reduce memory requirement and to employ only one or two filter units, independently from the number of decomposition levels to be computed. This is done by producing each output at the “earliest” instance that it can be produced [26].

The demand of low-power VLSI circuits in modern mobile/visual communication systems is growing all the time. On the other hand, the running progress of the VLSI technology has strongly reduced the cost of the hardware. Therefore, the possibility of reducing the period, even increasing the hardware,¹ is becoming an important issue. In fact, low-period devices are important also for low-power utilization. For instance, a device D having a period $T = N/2$ ccs could be employed not only for performing a two times faster processing than that allowed by D' having a period $T' = N$ ccs but also (clocked at a frequency f) for reaching the same performance reached by D' , when this one is clocked at a frequency $f' = 2f$. This circumstance allows D of reducing the supply voltage (linear in f) and the power dissipation (linear in f^2), respectively, by a factor of two and four with respect to D' [27].

The above considerations have motivated the work of this paper, which proposes two scalable architectures having an AT^2 parameter that is approximately 1/2 that of already existing devices and performing the DWT of an N_0 -sample sequence in only $N_0/2$ ccs, since they allow the sampling of the input sequence at a three frequency two times higher than the working clock frequency. The proposed devices are a pipeline (namely, Arc_J) and a “hybrid pipeline-RPA” architecture (namely, Arc_2^*).

¹The approach of merely using a team of K devices that cooperate in order to speed up the computation by K is not possible for most DWT applications, which require on-the-fly processing (i.e., to produce the output at the same rate that the input samples are received).

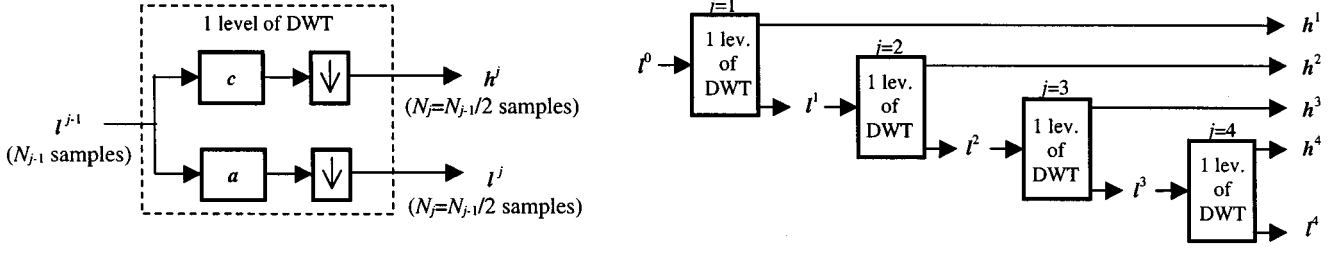


Fig. 1. One-dimensional DWT: dyadic decomposition in four levels. The symbol \downarrow denotes decimation by two.

Even though the pipeline paradigm already has been exploited by existing DWT architectures (e.g., those in [12] and [23]–[25]), to the best of our knowledge, none of them reaches the performance claimed in this paper. Also, it is worth noting that the existing pipelined devices are underutilized. In fact, the downsampling occurring in each DWT decomposition level greatly helps in designing RPA-based architectures (e.g., those in [14]–[17], [19], [20], [22], etc.) but makes the pipelined devices heavily underutilized, since the stage implementing the decomposition level k is usually clocked by a frequency 2^{k-1} times lower than the clock frequency used in the first level [24]. This underutilization comes from a low balancing of the pipelines when they implement the DWT and leads to a low efficiency. On the other hand, the architectures Arc_J and Arc_2^* are highly balanced.

In addition, because the designs of the proposed architectures depend only on the number of decomposition levels J and on the length of the filter L , we characterize the efficiencies (otherwise said *hardware utilization*) of Arc_J and Arc_2^* within a 2-D space Σ of coordinates (J, L) . Such characterization shows that the efficiency of Arc_J decreases with J , while the efficiency of Arc_2^* has an opposite behavior. This study suggests defining an architecture $\text{Arc}_{\text{OPT}}(J', L')$, which is highly efficient for any specific application identified by a point $(J', L') \in \Sigma$. $\text{Arc}_{\text{OPT}}(J', L')$ is simply the architecture (between Arc_J and Arc_2^*) having the highest efficiency in (J', L') . The efficiency of Arc_{OPT} , evaluated for a very wide subset of points in Σ , has excellent results, 99.1% being the average value.

The outline of this paper is as follows. In the next section, the one-dimensional (1-D) DWT is shortly recalled. The strategy leading to the design of Arc_J is illustrated in Section III. The computing blocks of Arc_J implementing the decomposition levels 1 and 2 (namely, B_1 and B_2) are described, respectively, in Sections III-A and III-B. Designs of blocks implementing decomposition levels higher than the second one are described in Section III-C. The “hybrid pipeline-RPA” architecture Arc_2^* is motivated and described in Section IV. Evaluations of the proposed architectures in terms of computing performances and efficiency, as well as the definition of Arc_{OPT} , are given in Section V. Conclusions are summarized in Section VI.

II. THE 1-D DISCRETE WAVELET TRANSFORM

The 1-D DWT [1]–[4] is a multilevel decomposition technique. In it, each decomposition level j ($1 \leq j \leq J$; $N_0 = p2^J$, N_0 , and p being, respectively, the length of the input sequence l^0 and an integer) can be seen as the further decomposi-

tion of the sequence l^{j-1} (having N_{j-1} samples) into two subbands l^j and h^j (both having $N_j = N_{j-1}/2$ samples). Such a decomposition is produced by two convolutions followed by a decimation by two, as depicted in Fig. 1 and formalized by

$$l_n^j = \sum_{i=0}^{L-1} a_i \cdot l_{2n-i}^{j-1}; \quad 0 \leq n < N_j \quad (1a)$$

$$h_n^j = \sum_{i=0}^{L-1} c_i \cdot l_{2n-i}^{j-1}; \quad 0 \leq n < N_j \quad (1b)$$

where a_i and b_i denote coefficients, respectively, of low-pass and high-pass L -tap filters (say, \mathbf{a} and \mathbf{c}),² and we have assumed $l_n^j = 0$ for $n < 0$ or $n \geq N_j$.

A direct consequence of the decimation by two in (1a) and (1b) is the following Property P, which assumes particular importance in the body of this paper.

Property P

Let $l^{j-\text{EVEN}}$, \mathbf{a}^{EVEN} and \mathbf{c}^{EVEN} be the sequences consisting of the even-numbered samples, respectively, of l^j , \mathbf{a} , and \mathbf{c} (i.e., $l_m^{j-\text{EVEN}} = l_{2m}^j$; $a_i^{\text{EVEN}} = a_{2i}$; and $c_i^{\text{EVEN}} = c_{2i}$; $0 \leq m < N_j/2$; $0 \leq i < L/2$). Moreover, let $l^{j-\text{ODD}}$, \mathbf{a}^{ODD} , and \mathbf{c}^{ODD} be the sequences consisting of the odd-numbered samples, respectively, of l^j , \mathbf{a} , and \mathbf{c} (i.e., $l_m^{j-\text{ODD}} = l_{2m+1}^j$; $a_i^{\text{ODD}} = a_{2i+1}$; and $c_i^{\text{ODD}} = c_{2i+1}$; $0 \leq m < N_j/2$; $0 \leq i < L - \lceil L/2 \rceil$). Therefore, l^{j+1} and h^{j+1} can be expressed as

$$l_n^{j+1} = \sum_{i=0}^{\lceil L/2 \rceil - 1} a_i^{\text{EVEN}} \cdot l_{n-i}^{j-\text{EVEN}} + \sum_{i=0}^{L - \lceil L/2 \rceil - 1} a_i^{\text{ODD}} \cdot l_{n-i}^{j-\text{ODD}}; \quad 0 \leq n < N_{j+1} \quad (1a')$$

$$h_n^{j+1} = \sum_{i=0}^{\lceil L/2 \rceil - 1} c_i^{\text{EVEN}} \cdot l_{n-i}^{j-\text{EVEN}} + \sum_{i=0}^{L - \lceil L/2 \rceil - 1} c_i^{\text{ODD}} \cdot l_{n-i}^{j-\text{ODD}}; \quad 0 \leq n < N_{j+1}. \quad (1b')$$

²The filters \mathbf{a} and \mathbf{c} may also have different number of taps. Nevertheless, in the literature on DWT architectures, they are considered to have an equal number of taps, since, in any case, the shorter filter can be suitably zero padded.

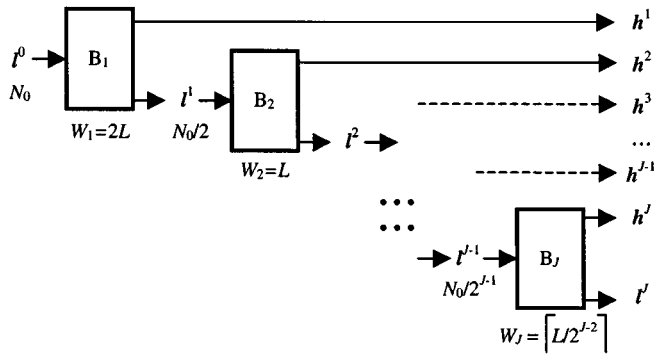


Fig. 2. Arc_J : top-level architectural scheme.

III. ARC_J : A BALANCED PIPELINED APPROACH TO THE 1-D DWT

Because of its structure, 1-D DWT can be straightforwardly pipelined into J blocks B_j ($1 \leq j \leq J$), each block B_j being devoted to compute the decomposition level j . Nevertheless, from (1a) and (1b), we observe that the complexity (say, C_j) of the decomposition level j is linear in N_j with a factor depending on L . Therefore, because of the decimation by two performed in each decomposition level

$$C_j = 2C_{j+1}. \quad (2)$$

As a clear consequence of (2), in order to balance a pipelined DWT architecture, each block B_j should employ a number W_j of processing elements (PEs) (each PE will be basically constituted by one multiplier and one adder)

$$W_j = 2W_{j+1}. \quad (3)$$

Therefore, our idea is to build a balanced architecture Arc_J constituted by a pipeline $\{B_1 \rightarrow B_2 \rightarrow \dots \rightarrow B_J\}$. Each block B_j is designed carefully taking into account (3). A top-level scheme of Arc_J is given in Fig. 2.

We assume $W_1 = 2L$ since, as we shall see in Section III, this choice leads to design of a block B_1 having a period of only $N_0/2$ ccs that is 100% efficient and *scalable with any value of L* . Because of this choice, in order to balance Arc_J , from (3), we must have $W_2 = L$ and, in general

$$W_k = \left\lceil \frac{L}{2^{k-2}} \right\rceil \quad (4)$$

where $\lceil z \rceil$ denotes the rounding to the smallest integer not smaller than z , and takes into account the discrete nature of the PEs.

Designs of blocks B_1 and B_2 employing, respectively, $2L$ and L PEs, are introduced in Section III-A and -B, respectively. Blocks B_k ($3 \leq k \leq J$) having W_k PEs as defined in (4) are described in Section III-C.

A. First Level of Decomposition: B_1

In order to design a high-speed block B_1 performing the first level of DWT decomposition, we consider Property P introduced in Section II. For $j = 0$, Property P means that $t^1[h^1]$ can be computed by the point-wise sum of two different and

independent convolutions having, as filters, \mathbf{a}^{EVEN} and \mathbf{a}^{ODD} [\mathbf{c}^{EVEN} and \mathbf{c}^{ODD}] and, as input, $t^{0\text{-EVEN}}$ and $t^{0\text{-ODD}}$, respectively.

Therefore, four independent filter units (say, \mathbf{A}^{EVEN} , \mathbf{A}^{ODD} , \mathbf{C}^{EVEN} , and \mathbf{C}^{ODD}), globally employing $2L$ PEs [say, $w_{a(0)}, w_{a(1)}, \dots, w_{a(L-1)}, w_{c(0)}, w_{c(1)}, \dots, w_{c(L-1)}$], can be designed and arranged in order to process in parallel $t^{0\text{-EVEN}}$ and $t^{0\text{-ODD}}$, and therefore, perform the first level of DWT at the rate of 2 samples per cc, supposing that 1 cc is the time needed by one PE to compute one product and one sum.³ Block B_1 is shown in Fig. 3 for $L = 6$. The data dependence graph (DDG) of \mathbf{A}^{EVEN} and \mathbf{A}^{ODD} computing t^1 is provided in Table I. From this, the DDG of \mathbf{C}^{EVEN} and \mathbf{C}^{ODD} can be straightforwardly derived. In this DDG, and in the other DDGs that will be considered in the following, the columns related to the I/O data lines [e.g., $I_E, I_O, I, O_{L(k-1)}, O, O_{L(k)},$ and $O_{H(k)}$], as well as the columns related to the select signals (e.g., S, S', S''), show the data present on that line in a specific cc [identified by the clock cycles counter $\tau(k)$, $1 \leq k \leq J$]. The products computed in each cc by the multipliers inside each PE are shown in the related columns. Arrows are used to denote how these products have to be added in order to achieve the final results.

The block B_1 has two input lines I_E and I_O . I_E feeds $t^{0\text{-EVEN}}$ both into \mathbf{A}^{EVEN} and into \mathbf{C}^{EVEN} , while I_O feeds $t^{0\text{-ODD}}$ both into \mathbf{A}^{ODD} and into \mathbf{C}^{ODD} . Because of the independence of the computations, $t_i^{0\text{-EVEN}}$ and $t_{i-1}^{0\text{-ODD}}$ are fed in parallel during the same cc $\tau(1) = i$ at the rate of 1 sample per cc. Therefore, the input sequence t^0 can be sampled at a frequency twice higher than the working frequency of the device and can be “consumed” by B_1 in only $(N_0/2) + 1$ ccs (this additional cc is due to the fact that the input of $t^{0\text{-ODD}}$ is delayed by 1 cc). The subband t^1 (i.e., the point-wise sum of the outputs from \mathbf{A}^{EVEN} and from \mathbf{A}^{ODD}) is generated by the adder S_L and output by means of the line $O_{L(1)}$ at the rate of 1 sample per cc, starting on $\tau(1) = 1$. Similarly, in parallel and at the same rate, the subband h^1 (i.e., the point-wise sum of the outputs from \mathbf{C}^{EVEN} and from \mathbf{C}^{ODD}) is generated by the adder S_H and output by means of the line $O_{H(1)}$. The filter units have been designed according to a well-known scheme [28], but any scheme for serial input/output convolution could be used. The four latches (shown in gray in Fig. 3) at the input of the adders S_L and S_H have no functional reason but have been inserted in order to make the critical paths limited to one multiplier and one adder. By this way, we can satisfy (in a very first approximation) the assumption that the clock period (i.e., the duration in time of 1 cc) is bounded by the latency of one multiplier and one adder, which is the standard clock period for DWT architectures [13]–[24]. Additional latches inserted at the output of each multiplier could further decrease the minimum allowed clock period.

The proposed design of B_1 is fully scalable with any value of L and has 100% efficiency since during each cc, all the PEs are employed in effective computations.

³Actually, (1a') and (1b') require globally $2L$ products and $2L-2$ sums, but we approximate these operations to those performed in 1 cc by $2L$ PEs. This approximation makes simpler the efficiency evaluation. Moreover, it will be compensated by an opposite approximation that will be assumed in Section V.

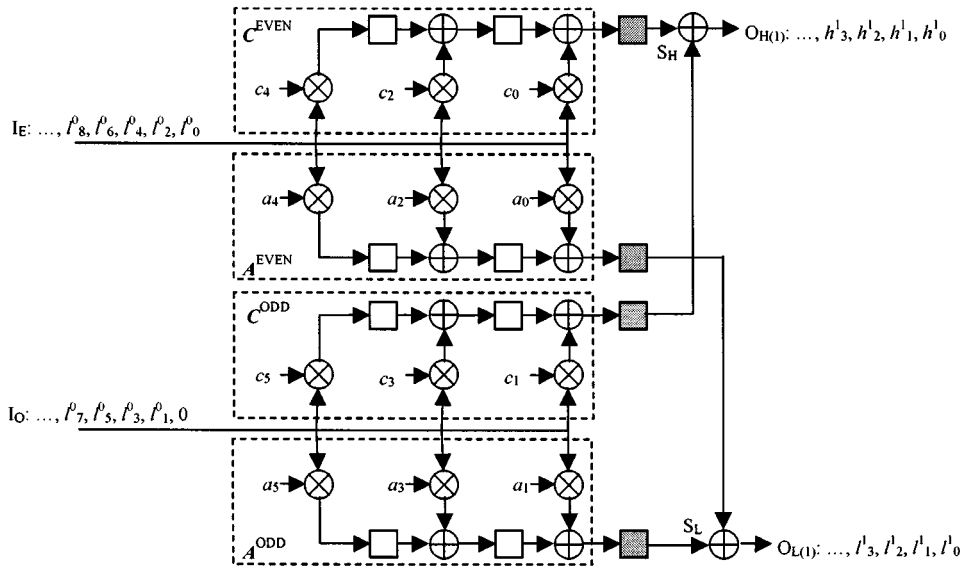


Fig. 3. Block B_1 : architectural scheme ($L = 6$). The gray latches have no functional reason but have been inserted in order to make the critical paths limited to one multiplier and one adder.

TABLE I
BLOCK B_1 : DDG OF A^{EVEN} AND A^{ODD} COMPUTING l^1 (NINE PERIODS).
THE DDG OF C^{EVEN} AND C^{ODD} CAN BE STRAIGHTFORWARDLY DERIVED

$\tau_{(1)}$	I_E	I_O	$w_{a(4)}$	$w_{a(2)}$	$w_{a(0)}$	$O_{L(1)}$	$w_{a(1)}$	$w_{a(3)}$	$w_{a(5)}$
0	l_0^0	0	$a_4 l_0^0$	$a_2 l_0^0$	$a_0 l_0^0$	0	$a_1 0$	$a_3 0$	$a_5 0$
1	l_2^1	l_1^1	$a_4 l_2^1$	$a_2 l_2^1$	$a_0 l_2^1$	l_0^1	$a_1 l_1^1$	$a_3 l_1^1$	$a_5 l_1^1$
2	l_4^2	l_3^2	$a_4 l_4^2$	$a_2 l_4^2$	$a_0 l_4^2$	l_1^2	$a_1 l_3^2$	$a_3 l_3^2$	$a_5 l_3^2$
3	l_6^3	l_5^3	$a_4 l_6^3$	$a_2 l_6^3$	$a_0 l_6^3$	l_2^3	$a_1 l_5^3$	$a_3 l_5^3$	$a_5 l_5^3$
4	l_8^4	l_7^4	$a_4 l_8^4$	$a_2 l_8^4$	$a_0 l_8^4$	l_3^4	$a_1 l_7^4$	$a_3 l_7^4$	$a_5 l_7^4$
5	l_{10}^5	l_9^5	$a_4 l_{10}^5$	$a_2 l_{10}^5$	$a_0 l_{10}^5$	l_4^5	$a_1 l_9^5$	$a_3 l_9^5$	$a_5 l_9^5$
6	l_{12}^6	l_{11}^6	$a_4 l_{12}^6$	$a_2 l_{12}^6$	$a_0 l_{12}^6$	l_5^6	$a_1 l_{11}^6$	$a_3 l_{11}^6$	$a_5 l_{11}^6$
7	l_{14}^7	l_{13}^7	$a_4 l_{14}^7$	$a_2 l_{14}^7$	$a_0 l_{14}^7$	l_6^7	$a_1 l_{13}^7$	$a_3 l_{13}^7$	$a_5 l_{13}^7$
8	l_{16}^8	l_{15}^8	$a_4 l_{16}^8$	$a_2 l_{16}^8$	$a_0 l_{16}^8$	l_7^8	$a_1 l_{15}^8$	$a_3 l_{15}^8$	$a_5 l_{15}^8$
...

B. Second Level of Decomposition: B_2

When more than one level of decomposition is needed, as in most of the applications, the subband l^1 produced by B_1 has to be further transformed. This requires a second block B_2 , which has to be pipelined to the output line $O_{L(1)}$ coming from B_1 . In this section, we describe how such a block B_2 , having 100% efficiency for any value of L , can be designed.

As previously stated, B_2 has to be provided with L PEs w_i ($0 \leq i < L$) in order to constitute a balanced team with B_1 . Therefore, a possible way of employing these PEs is to assign to w_i the computations related both to a_i and to c_i . This strategy is made possible by Property P introduced in the previous section.

In fact, as shown in the DDG of B_1 (Table I), the line $O_{L(1)}$ outputs the samples of l^1 in a sequential data stream at a rate of 1 sample per cc. Therefore, by splitting $O_{L(1)}$ into two different lines I_E and I_O carrying, respectively, $l^{1-\text{EVEN}}$ and $l^{1-\text{ODD}}$ into B_2 , we have the possibility of duplicating on I_E [I_O] the single sample $l_{2n}^1[l^{1-2n+1}]$ ($0 \leq n < N_1/2$) before a new sample $l_{2n+2}^1[l^{1-2n+3}]$ is sent by $O_{L(1)}$ onto I_E [I_O], without increasing

the period. By this way, $l_{2n}^1[l^{1-2n+1}]$ is available for 2 ccs and $w_{2m}[w_{2m+1}]$ ($0 \leq m < L/2$) has the time for processing it both by a_{2m} and by c_{2m} [by a_{2m+1} and by c_{2m+1}]. In order to do this, an *Adapter* such as that shown in Fig. 4 has to be inserted between $O_{L(1)}$ and the lines I_E [I_O]. Such an adapter allows the following strategy:

$$I_E(\tau_{(2)}) = \begin{cases} O_{L(1)}(\tau_{(2)}), & \text{if } S(\tau_{(2)}) = 0 \\ I_E(\tau_{(2)} - 1), & \text{if } S(\tau_{(2)}) = 1 \end{cases} \quad (5a)$$

$$I_O(\tau_{(2)}) = \begin{cases} I_O(\tau_{(2)} - 1), & \text{if } S(\tau_{(2)}) = 0 \\ O_{L(1)}(\tau_{(2)}), & \text{if } S(\tau_{(2)}) = 1 \end{cases} \quad (5b)$$

where the clock cycles counter $\tau_{(2)}$ is here adopted only in order to make transparent the propagation delay introduced in the pipe by B_1 [i.e., $\tau_{(2)} = 0$ when l_0^1 is output by B_1] and $S(\tau_{(2)})$ is a select signal, which is zero [1] for the even [odd] values of $\tau_{(2)}$.⁴

The functionality of B_2 becomes clear by an analysis of its DDG, which is given in Table II. Starting on $\tau_{(2)} = 0$, the input line I_E [I_O] provides the even-numbered [odd-numbered] PEs of B_2 with the even-numbered [odd-numbered] samples of l^1 , which are duplicated as shown in the I_E [I_O] column of the DDG. The multipliers in the even-numbered PEs w_{2m} are provided with two-cell circular shift registers (CSRs) pre-loaded in such a way that the filter coefficients a_{2m} are used in the even-numbered ccs ($S = 0$) and the filter coefficients c_{2m} are used in the odd-numbered ccs ($S = 1$). Conversely, CSRs connected in input to the multipliers of the odd-numbered PEs w_{2m+1} operate in the opposite way, i.e., the coefficients of a are used in the odd-numbered ccs, and the coefficients of c are used in the even-numbered ccs.⁵ By this way, l^2 and h^2 can be pro-

⁴An alternative way of achieving the desired data flow onto I_E and I_O is to replace the two multiplexers in the *Adapter* with two latches triggered at a rate twice slower than the clock signal.

⁵In practice, these circular shift registers work as multiplexers, but they are more compact. Their use, instead of the use of multiplexers, will be even more convenient in the other levels of the pipe, since, as we shall show in Section III-C, each multiplier in B_k will need a (2^{k-1}) -input multiplexer, suitably replaced by a (2^{k-1}) -cell CSR.

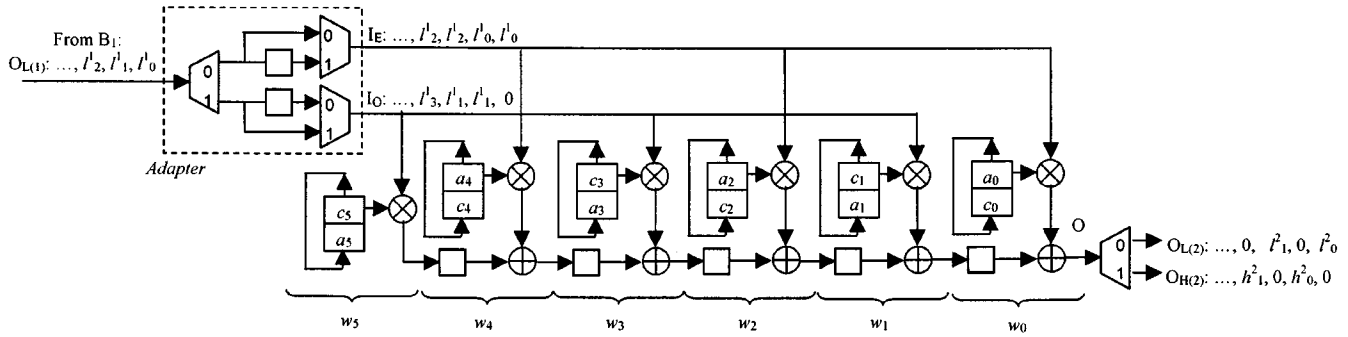


Fig. 4. Block B_2 : architectural scheme ($L = 6$). The select signal S , controlling multiplexers and demultiplexers, is not explicitly shown. The two-cell CSRs storing the filter coefficients work as two-input multiplexers.

TABLE II
BLOCK B_2 : DDG (FOUR PERIODS)

$\tau_{(2)}$	$O_{L(1)}$	S	I_E	I_O	w_5	w_4	w_3	w_2	w_1	w_0	O	$O_{L(2)}$	$O_{H(2)}$
0	l'_0	0	l'_0	0	$c_5 0$	$a_4 l'_0$	$c_3 0$	$a_2 l'_0$	$c_1 0$	$a_0 l'_0$	\hat{l}'_0	\hat{l}'_0	0
1	l'_1	1	l'_0	l'_1	$a_5 l'_1$	$c_4 l'_0$	$a_3 l'_1$	$c_2 l'_0$	$a_1 l'_1$	$c_0 l'_0$	\hat{h}'_0	0	\hat{h}'_0
2	l'_2	0	l'_2	l'_1	$c_5 l'_1$	$a_4 l'_2$	$c_3 l'_1$	$a_2 l'_2$	$c_1 l'_1$	$a_0 l'_2$	\hat{l}'_1	\hat{l}'_1	0
3	l'_3	1	l'_2	l'_3	$a_5 l'_3$	$c_4 l'_2$	$a_3 l'_3$	$c_2 l'_2$	$a_1 l'_3$	$c_0 l'_2$	\hat{h}'_1	0	\hat{h}'_1
4	l'_4	0	l'_4	l'_3	$c_5 l'_3$	$a_4 l'_4$	$c_3 l'_3$	$a_2 l'_4$	$c_1 l'_3$	$a_0 l'_4$	\hat{l}'_2	\hat{l}'_2	0
5	l'_5	1	l'_4	l'_5	$a_5 l'_5$	$c_4 l'_4$	$a_3 l'_5$	$c_2 l'_4$	$a_1 l'_5$	$c_0 l'_4$	\hat{h}'_2	0	\hat{h}'_2
6	l'_6	0	l'_6	l'_5	$c_5 l'_5$	$a_4 l'_6$	$c_3 l'_5$	$a_2 l'_6$	$c_1 l'_5$	$a_0 l'_6$	\hat{l}'_3	\hat{l}'_3	0
7	l'_7	1	l'_6	l'_7	$a_5 l'_7$	$c_4 l'_6$	$a_3 l'_7$	$c_2 l'_6$	$a_1 l'_7$	$c_0 l'_6$	\hat{h}'_3	0	\hat{h}'_3
...

duced and output via the lines $O_{L(2)}$ and $O_{H(2)}$, respectively, in the even-numbered and in the odd-numbered ccs.

Block B_2 is balanced with B_1 since its design has been dimensioned on L PE's. Also, it is fully scalable with L and 100% efficiency.

C. Higher Levels of Decomposition: B_k Having Less than L PEs

In this section, we describe the blocks B_k ($k > 2$), which have to be implemented in Arc_J for any $k \leq J$.

In order to be "at the best" balanced with the above-described B_1 and B_2 , a block B_k should employ a number of PEs

$$W_k = \left\lceil \frac{L}{2^{k-2}} \right\rceil. \quad (4)$$

Since L is at least two, for $k > 2$, $W_k < L$. Therefore, in this section, we deal with the designs of L -tap filter structures employing less than L PEs.

To solve this problem, we will consider a semisystolic approach based on arrays similar to those used by B_1 and B_2 , where, at a given cc $\tau(2^{k-2}t \leq \tau < 2^{k-2}(t+1), 0 \leq t < N_k)$, each PE processes the same input sample l_t^{k-1} .

An alternative approach based on arrays can be found in [29], where at a given cc $\tau(2^{k-2}t \leq \tau < 2^{k-2}(t+1), 0 \leq t < N_k)$, the PEs process different input samples and, specifically, w_i process l_{t-i}^{k-1} .

Independently from the approaches, a direct consequence of (4) is that in B_k , each single PE will perform the computations related both to 2^{k-2} coefficients of \mathbf{a} and to 2^{k-2} coefficients of \mathbf{c} . This "folded-like computation" is made possible because of

TABLE III
BLOCK B_3 : DDG (THREE PERIODS). "—" DENOTES THAT ONLY PARTIAL RESULTS ARE AVAILABLE ON O

$\tau_{(3)}$	$O_{L(2)}$	S	S'	I	w_2	w_1	w_0	O	$O_{L(3)}$	$O_{H(3)}$
0	\hat{l}'_0	0	0	\hat{l}'_0	$a_4 \hat{l}'_0$	$a_2 \hat{l}'_0$	$a_0 \hat{l}'_0$	\hat{l}'_0	\hat{l}'_0	0
1	0	0	1	\hat{l}'_0	$c_4 \hat{l}'_0$	$c_2 \hat{l}'_0$	$c_0 \hat{l}'_0$	\hat{h}'_0	0	\hat{h}'_0
2	\hat{l}'_1	1	0	\hat{l}'_1	$a_5 \hat{l}'_1$	$a_3 \hat{l}'_1$	$a_1 \hat{l}'_1$	—	0	0
3	0	1	1	\hat{l}'_1	$c_5 \hat{l}'_1$	$c_3 \hat{l}'_1$	$c_1 \hat{l}'_1$	—	0	0
4	\hat{l}'_2	0	0	\hat{l}'_2	$a_4 \hat{l}'_2$	$a_2 \hat{l}'_2$	$a_0 \hat{l}'_2$	\hat{l}'_1	\hat{l}'_1	0
5	0	0	1	\hat{l}'_2	$c_4 \hat{l}'_2$	$c_2 \hat{l}'_2$	$c_0 \hat{l}'_2$	\hat{h}'_1	0	\hat{h}'_1
6	\hat{l}'_3	1	0	\hat{l}'_3	$a_5 \hat{l}'_3$	$a_3 \hat{l}'_3$	$a_1 \hat{l}'_3$	—	0	0
7	0	1	1	\hat{l}'_3	$c_5 \hat{l}'_3$	$c_3 \hat{l}'_3$	$c_1 \hat{l}'_3$	—	0	0
8	\hat{l}'_4	0	0	\hat{l}'_4	$a_4 \hat{l}'_4$	$a_2 \hat{l}'_4$	$a_0 \hat{l}'_4$	\hat{l}'_2	\hat{l}'_2	0
9	0	0	1	\hat{l}'_4	$c_4 \hat{l}'_4$	$c_2 \hat{l}'_4$	$c_0 \hat{l}'_4$	\hat{h}'_2	0	\hat{h}'_2
10	\hat{l}'_5	1	0	\hat{l}'_5	$a_5 \hat{l}'_5$	$a_3 \hat{l}'_5$	$a_1 \hat{l}'_5$	—	0	0
11	0	1	1	\hat{l}'_5	$c_5 \hat{l}'_5$	$c_3 \hat{l}'_5$	$c_1 \hat{l}'_5$	—	0	0
...

the down-sampling, since the line $O_{L(k-1)}$ (coming from B_{k-1}) feeds into B_k the samples of l^{k-1} at the rate of 1 sample every 2^{k-2} ccs. Therefore, such input data can be suitably replicated, without increasing the period, in order to allow each processor to perform the needed number of operations, before new data are input from B_{k-1} . Also, because of this folded-like computation, the communications among the PEs will not be exclusively systolic, since also a feedback of the partial results will be required. In the following, we will explicitly refer to the case of $k = 3$, but we will also describe how other cases can be derived.

Table III shows the DDG of block B_3 for $W_3 = 3, 5 \leq L \leq 6$.⁶ The computation is periodic with a period of 2^{k-1} ccs (in this case, 4 ccs). The periods are subdivided into two semiperiods identified by a binary select signal S . S is zero [1] when an even-numbered [odd-numbered] input sample is input into B_k . In the semiperiod $S = 0$, the PE w_i ($0 \leq i < W_k$) performs "for $j = 0$ to $(2^{k-3}-1)$ " the products of the input sample by the filter coefficients a_{2i+2jW_k} and c_{2i+2jW_k} (in this order) and adds these products to the data produced by itself during the previous semiperiod (feedback). In the semiperiod $S = 1$, w_i performs "for $j = 0$ to $(2^{k-3}-1)$ " the products of the input sample by the filter coefficients a_{2i+2jW_k+1} and c_{2i+2jW_k+1}

⁶When a particular application requires a filter length $L < L'_k = W_k 2^{k-2}$, the exceeding $L'_k - L$ coefficients have to be replaced by zeroes. As we shall show in Section VI, such a situation will imply underutilization, leading to an efficiency lower than 100%.

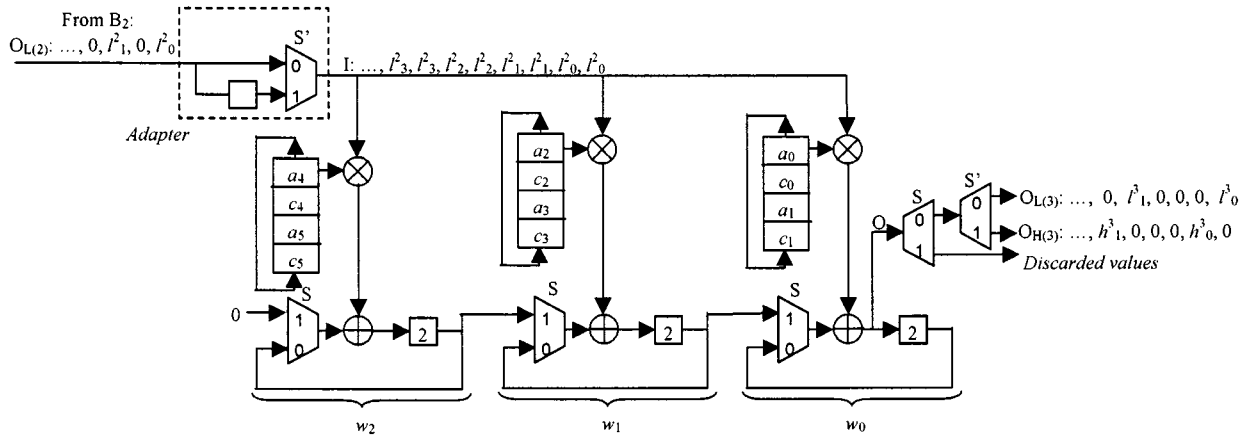


Fig. 5. Block B_3 : architectural scheme ($5 \leq L \leq 6$). The *Adapter* can be replaced by a single latch triggered by a signal having a frequency twice as low as the frequency of the clock. The four-cell CSRs storing the filter coefficients work as four-input multiplexers.

(in this order) and adds these products to the data produced by w_{i+1} during the previous semiperiod (systolic communication). The correct use of the filter coefficients by each multiplier is simply achieved by storing the coefficients in (2^{k-1}) -cell CSRs (one CSR for each PE) in the above-specified order. For $k = 3$, the operations performed in each period by w_i are simply those related to a_{2i} and c_{2i} (semiperiod $S = 0$) and those related to a_{2i+1} and c_{2i+1} (semiperiod $S = 1$). This scheduling takes into account Property P. The final results are produced at the same rate of the input, i.e., two samples per period, which means two samples every 2^{k-1} ccs (in this case, 4 ccs). The architecture implementing B_3 according to this data DDG is shown in Fig. 5. The *Adapter* duplicates for one semiperiod, i.e., for 2^{k-2} ccs (for $k = 3$, 2 ccs) the input samples onto the line I , as soon as they are produced by B_{k-1} . The multiplexers in input to the adders select (by means of the signal S) the feedback or the systolic communication among the processors.

The proposed scheme assumes a substantial difference for $k > 3$. In fact, since w_i has to be shared by 2^{k-3} coefficients (which are at least two, for $k > 3$) for each one of the filters \mathbf{a}^{EVEN} , \mathbf{a}^{ODD} , \mathbf{c}^{EVEN} and \mathbf{c}^{ODD} , the same input data will be multiplied (in each PE) at least by two low-pass and at least by two high-pass filter coefficients. Therefore, the composition of the final result needs an external feedback between the first and the last PE of the systolic array. These concepts are clearer from an exam of the DDG of block B_4 , which is provided in Table IV ($W_4 = 2$, $5 \leq L \leq 8$). The above-mentioned external feedback (denoted by gray arrows in the graphs) concerns $(2^{k-2}-2)$ every 2^{k-2} data accumulated by the adder in w_0 in the first semiperiod $S = 0$ (the remaining two data constitute a sample of l^k and a sample of h^k and do not need further processing). Specifically, these data are delayed by $(2^{k-2}-2)$ ccs and sent in input to the adder of the last PE of the array (i.e., w_{W_k-1}). Therefore, in the semiperiod $S = 1$, the adder in w_{W_k-1} receives the data that were generated in w_0 during the previous semiperiod. This external feedback is enabled by the combination $\{S = 1, S' = 0\}$ and does not involve the last 2 ccs of the semiperiod $S = 1$, since in those ccs w_{W_k-1} is employing the filter coefficients having index $(L'_k - 1) = (2^{k-2}W_k - 1)$.

The architecture that realizes such a data graph is represented in Fig. 6. From the presented examples and from the above de-

TABLE IV
BLOCK B_4 : DDG (TWO PERIODS). “-” DENOTES THAT ONLY PARTIAL RESULTS ARE AVAILABLE ON O . THE CONDITION $\{S = 1, S' = 0\}$, WHICH ENABLES THE EXTERNAL FEEDBACK (GRAY ARROWS), HAS BEEN EVIDENCED IN BOLD

$\tau_{(4)}$	$O_{L(3)}$	S	S'	S''	l	w_1	w_0	O	$O_{L(4)}$	$O_{H(4)}$
0	l^0_0	0	0	0	l^0_0	$a_2 l^0_0$	$a_0 l^0_0$	l^0_0	l^0_0	0
1	0	0	0	1	l^0_0	$c_2 l^0_0$	$c_0 l^0_0$	h^0_0	0	h^0_0
2	0	0	1	0	l^0_0	$a_6 l^0_0$	$a_4 l^0_0$	-	0	0
3	0	0	1	1	l^0_0	$c_6 l^0_0$	$c_4 l^0_0$	-	0	0
4	l^1_1	1	0	0	l^1_1	$a_3 l^1_1$	$a_1 l^1_1$	-	0	0
5	0	1	0	1	l^1_1	$c_3 l^1_1$	$c_1 l^1_1$	-	0	0
6	0	1	1	0	l^1_1	$a_7 l^1_1$	$a_5 l^1_1$	-	0	0
7	0	1	1	1	l^1_1	$c_7 l^1_1$	$c_5 l^1_1$	-	0	0
8	l^2_2	0	0	0	l^2_2	$a_2 l^2_2$	$a_0 l^2_2$	l^2_2	l^2_2	0
9	0	0	0	1	l^2_2	$c_2 l^2_2$	$c_0 l^2_2$	h^2_2	0	h^2_2
10	0	0	1	0	l^2_2	$a_6 l^2_2$	$a_4 l^2_2$	-	0	0
11	0	0	1	1	l^2_2	$c_6 l^2_2$	$c_4 l^2_2$	-	0	0
12	l^3_3	1	0	0	l^3_3	$a_3 l^3_3$	$a_1 l^3_3$	-	0	0
13	0	1	0	1	l^3_3	$c_3 l^3_3$	$c_1 l^3_3$	-	0	0
14	0	1	1	0	l^3_3	$a_7 l^3_3$	$a_5 l^3_3$	-	0	0
15	0	1	1	1	l^3_3	$c_7 l^3_3$	$c_5 l^3_3$	-	0	0
...

scriptions, designs of B_k , with $k > 4$, can be easily derived. Also, the extension to B_k having higher values of W_k is trivial. In the case of $W_k = 1$ (i.e., $L \leq L'_k = 2^{k-2}$), only one PE is employed, and therefore, the external loop connecting the first PE with the last one becomes a loop connecting the only w_0 with itself. Note that the *Adapters* at the input of the blocks shown in Figs. 5 and 6, as well as the tree of demultiplexers at the output of the blocks, can be avoided by a suitable sampling of the data lines.

IV. ARC $_2^*$: A HYBRID PIPELINE-RPA ARCHITECTURE FOR THE 1-D DWT

As we shall show in the next section, Arc $_J$ is 100% efficient for any value of L when $J < 3$. For $J \geq 3$, the efficiency of Arc $_J$, say, $\text{eff}[\text{Arc}_J]$, is still 100%, but only if

$$L = L'_k (\equiv W_k \cdot 2^{k-2}) \quad (6)$$

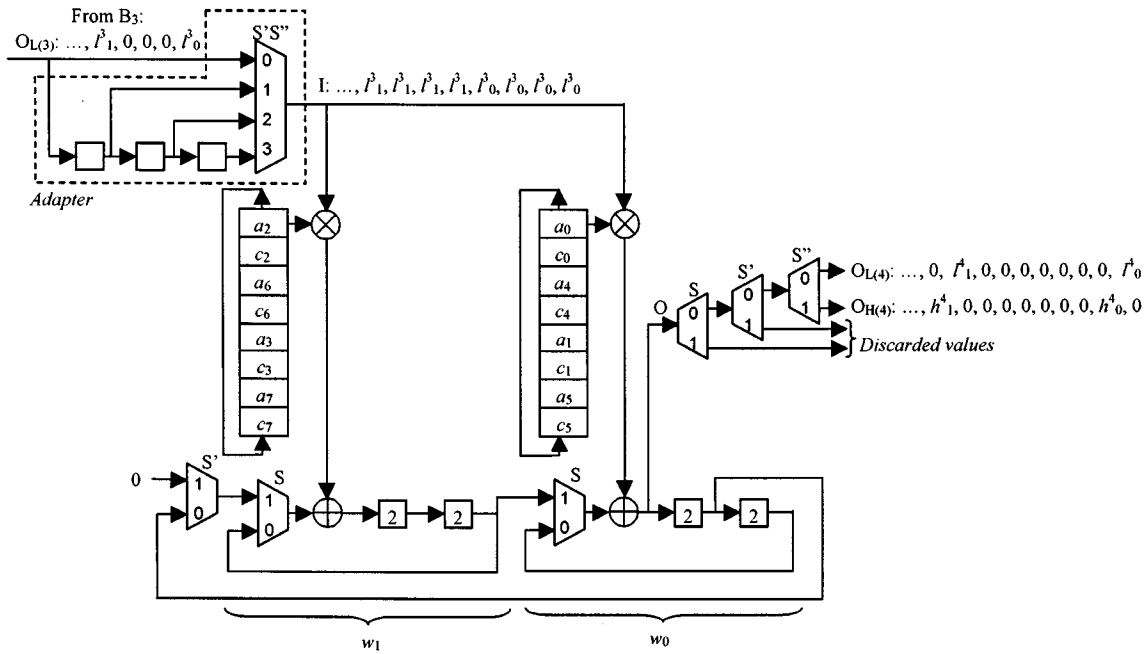


Fig. 6. Block B_4 : architectural scheme ($5 \leq L \leq 8$). The *Adapter* can be replaced by a single latch triggered by a signal having a frequency four times lower than the frequency of the clock. The eight-cell CSRs storing the filter coefficients work as eight-input multiplexers.

for any $k \leq J$. This restriction is due to the rounding in (4) that leads to implementation in B_k two filters having L'_k taps, where L'_k may also be greater than L . As we have seen in Section III, when (6) is not verified, $2(L'_k - L)$ filter coefficients, among those preloaded in B_k , have to be zeroes⁶ and therefore, B_k will be underutilized.

Another consequence of the rounding in (4) is that some J and some L may exist such that⁷

$$\left(\sum_{k=2}^J W_k = \right) \sum_{k=2}^J \left\lceil \frac{L}{2^{k-2}} \right\rceil > 2L \quad (7)$$

which means that the blocks B_2, B_3, \dots, B_{J-1} , and B_J globally require more than $2L$ PEs.

On the other hand, from (2) in Section III, we observe that

$$C_{\bar{j}} > \sum_{k=\bar{j}+1}^J C_k \quad (8a)$$

for any $1 \leq \bar{j} < J$. In particular, for $\bar{j} = 1$

$$C_1 > \sum_{k=2}^J C_k. \quad (8b)$$

Therefore, it seems reasonable to us to design also a second architecture (say, Arc_2^*), which is constituted by a pipeline $\{B_1 \rightarrow B_2^*\}$ having only two blocks for any value of $J > 2$. In Arc_2^* , the decomposition level 1 is produced by B_1 exactly as in Arc_J , while all the levels k ($2 \leq k \leq J$) are performed by a new block B_2^* . Because of (8b), a number W_2^* of PEs not bigger

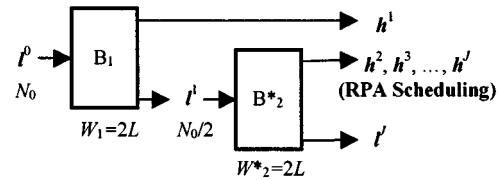


Fig. 7. Arc_2^* : top-level architectural scheme.

than W_1 is needed by B_2^* in order to not deteriorate the performance allowed by B_1 . Therefore, we choose $W_2^* = W_1 = 2L$. As an effect of this choice, Arc_2^* will be more efficient than Arc_J in any application requiring J decomposition levels and employing L -tap filter bases when J and L satisfy (7).

The design of Arc_2^* does not require additional descriptive details, since we observe that the N_1 samples of l^1 are fed into B_2^* by the line $O_{L(1)}$ in a sequential data stream and at the rate of 1 sample per cc. This means that B_2^* can simply be an RPA-based architecture able to decompose in $J-1$ levels a sequence of N_1 samples in N_1 ccs, employing $2L$ PEs. Many of these devices have already appeared in the literature: for instance, the architectures described in [16]–[18], the parallel filter proposed in [19], the architecture “A1” proposed in [15], the three devices described in [14] when, as the same authors suggest, they are provided with a double number of processors, etc.

Therefore, for our purposes, a scheme of Arc_2^* can be simply proposed in the form of Fig. 7, where B_1 is the block described in Section III-A and B_2^* is any RPA-based architecture employing $2L$ PEs and transforming the N_1 samples of l^1 in N_1 ccs. It is worth noting that Arc_2^* (i.e., the above-described “coupled use” of B_1 with any already known RPA device B_2^*) allows two times faster processing than does the classical RPA-based device B_2^* .

⁷For instance, (7) is trivially verified for any $J > L + 2$.

V. PERFORMANCE EVALUATIONS AND DEFINITION OF $\text{ARC}_{\text{OPT}}(J, L)$

A. Period and AT^2 Parameter

In order to evaluate the global time required by the proposed architectures to perform the DWT of a sequence \mathbf{t}^0 having N_0 samples, we observe that each block B_k has a period $T_k = 2^{k-1}$ ccs, since it produces a sample h_i^k every 2^{k-1} ccs (immediately followed by h_i^k). As a consequence, each block needs $T_k N_k / 2 = N_0 / 2$ ccs to perform its task (balanced computation). Therefore, the period of Arc_J is

$$T[\text{Arc}_J] = \frac{N_0}{2} + \Delta_J \quad [\text{ccs}] \quad (9)$$

where Δ_J takes into account the “startup” delay due to the propagation through the pipe (i.e., the number of ccs needed by B_J to output h_0^J). Nevertheless, it can be neglected since $\Delta_J \approx J$, which, in practical applications, is very much smaller than N_0 . Approximately $N_0/2$ ccs is also the value of $T[\text{Arc}_2^*]$, when B_2^* is implemented by devices as those referenced in Section IV.

Measures of $T[\text{Arc}_J]$ and $T[\text{Arc}_2^*]$ can be known only taking into account the particular adopted technology. However, to be independent of the technology, in a very first approximation (considering a latch inserted between each pair of adjacent blocks), we can assume the cc period as the latency of one multiplier plus the latency of one adder, which is the same assumption made by other authors. Therefore, as we have already claimed in the introduction, Arc_J and Arc_2^* are approximately two times faster than all the other “single chip” known architectures.⁸

Our architectures allow approximately the same improvement (i.e., by a factor of two) in terms of the AT^2 parameter, as summarized in Table V. In such a table, the VLSI area has been characterized only by means of the number of multipliers and adders: such a measure is therefore only indicative. Nevertheless, it should be remarked that in DWT applications, the required precision grows with the levels. Therefore, while the only processing unit in classical RPA-based devices must achieve the precision required by the level J (i.e., the highest one), pipelines might be realized by blocks that in lower levels achieve lower precision. This possibility could provide further reduction to the VLSI area of Arc_J and Arc_2^* . Moreover, Arc_J (and Arc_2^* , in part) are semisystolic (i.e., they do not require complex routing, as many RPA devices do) and use a controller simpler than that needed by fully RPA architectures. In fact, the multiplexers and demultiplexers employed in any block B_k of Arc_J need a set of $(k-1)$ select signals, which are the $(k-1)$ least significant outputs of a counter modulo 2^{J-1} . Therefore, this counter is the only control unit needed by Arc_J , since it provides all the needed select signals.

B. Efficiency

Another important issue in parallel architectures is the effective utilization of the processors, which is characterized by the *efficiency*. We can define the efficiency of a parallel architecture A having $W[A]$ PEs as the ratio between the period $T[B]$ of a

⁸Even though single-input multiple data realizations have been shown in [19], performing decompositions either in LJ or in L ccs, these devices require, respectively, $2N_0$ and N_0 PEs, and in most applications cannot be implemented in a single chip.

TABLE V
COMPARATIVE EVALUATION OF DWT ARCHITECTURES: PERIOD AND INDICATIVE ESTIMATION OF AT^2 PARAMETER

Architectures	Number of “PEs”	Period	AT^2
Arc_2^*	$4L$	$N_0/2$	α
Arc_J	$\sum_{j=1}^J \left\lceil \frac{L}{2^{j-2}} \right\rceil$	$N_0/2$	$\approx \alpha$
Architectures in [14] (doubled hardware)	$2L$	N_0	$\approx 2\alpha$
Architecture A1 in [15]	$2L$	N_0	$\approx 2\alpha$
Architectures in [16]	$2L$	N_0	$\approx 2\alpha$
Architectures in [17]	$2L$	N_0	$\approx 2\alpha$
Architecture in [18]	$2L$	N_0	$\approx 2\alpha$
Parallel Filter in [19]	$2L$	N_0	$\approx 2\alpha$
Architecture in [23]	$2L$	N_0	$\approx 2\alpha$
Architecture in [13]	L	$2N_0$	$\approx 4\alpha$
Architectures in [14]	L	$2N_0$	$\approx 4\alpha$
Architecture A2 in [15]	L	$2N_0$	$\approx 4\alpha$
Pipeline in [24]	JL	N_0	$\approx J\alpha$
Pipeline in [12]	JL	$2N_0$	$\approx 4J\alpha$

nonparallel architecture B (i.e., $W[B] = 1$) and $W[A]$ times the period $T[A]$ of the parallel architecture A . From the efficiency $\text{eff}[A]$, the speedup of an architecture A (say, $\text{speedup}[A]$) can be straightforwardly derived as $\text{speedup}[A] = \text{eff}[A]W[A]$.

In order to simplify the following analysis, we define one product and one sum as one “basic computation” occurring in the 1-D DWT. Since we assumed 1 cc as the time needed by a single PE to perform one “basic computation,” the period $T[B]$ measured in ccs is simply the number of basic computations required by the 1-D DWT. Because the production of each coefficient of a given subband requires L basic computations,⁹ we have

$$T[B] = L \sum_{j=1}^J \frac{N_0}{2^{j-1}} \quad [\text{ccs}]. \quad (10)$$

On the other hand, the global number of PEs employed by Arc_J and Arc_2^* is¹⁰

$$W[\text{Arc}_J] = \sum_{j=1}^J \left\lceil \frac{L}{2^{j-2}} \right\rceil \quad (11a)$$

$$W[\text{Arc}_2^*] = 4L \quad \text{for } J \geq 4. \quad (11b)$$

Therefore, considering $T[\text{Arc}_J] = T[\text{Arc}_2^*] = N_0/2$ ccs, $\text{eff}[\text{Arc}_J]$ and $\text{eff}[\text{Arc}_2^*]$ result

$$\text{eff}[\text{Arc}_J] = \frac{\sum_{j=1}^J \frac{L}{2^{j-2}}}{\sum_{j=1}^J \left\lceil \frac{L}{2^{j-2}} \right\rceil} \quad (12a)$$

$$\text{eff}[\text{Arc}_2^*] = \sum_{j=1}^J \frac{1}{2^j} \quad \text{for } J \geq 4. \quad (12b)$$

⁹Actually, any subband coefficient requires L products and $L-1$ sums, but we approximate these operations to L basic computations. On the other hand, this approximation compensates the approximation made in the opposite sense, when blocks B_k having W_k multipliers and $W_k - 1$ adders were considered having W_k PEs³.

¹⁰ Arc_2^* has to be considered only for $J \geq 4$. In fact, Arc_J , for $J < 4$ and for any $L \geq 2$, requires fewer PEs than Arc_2^* does.

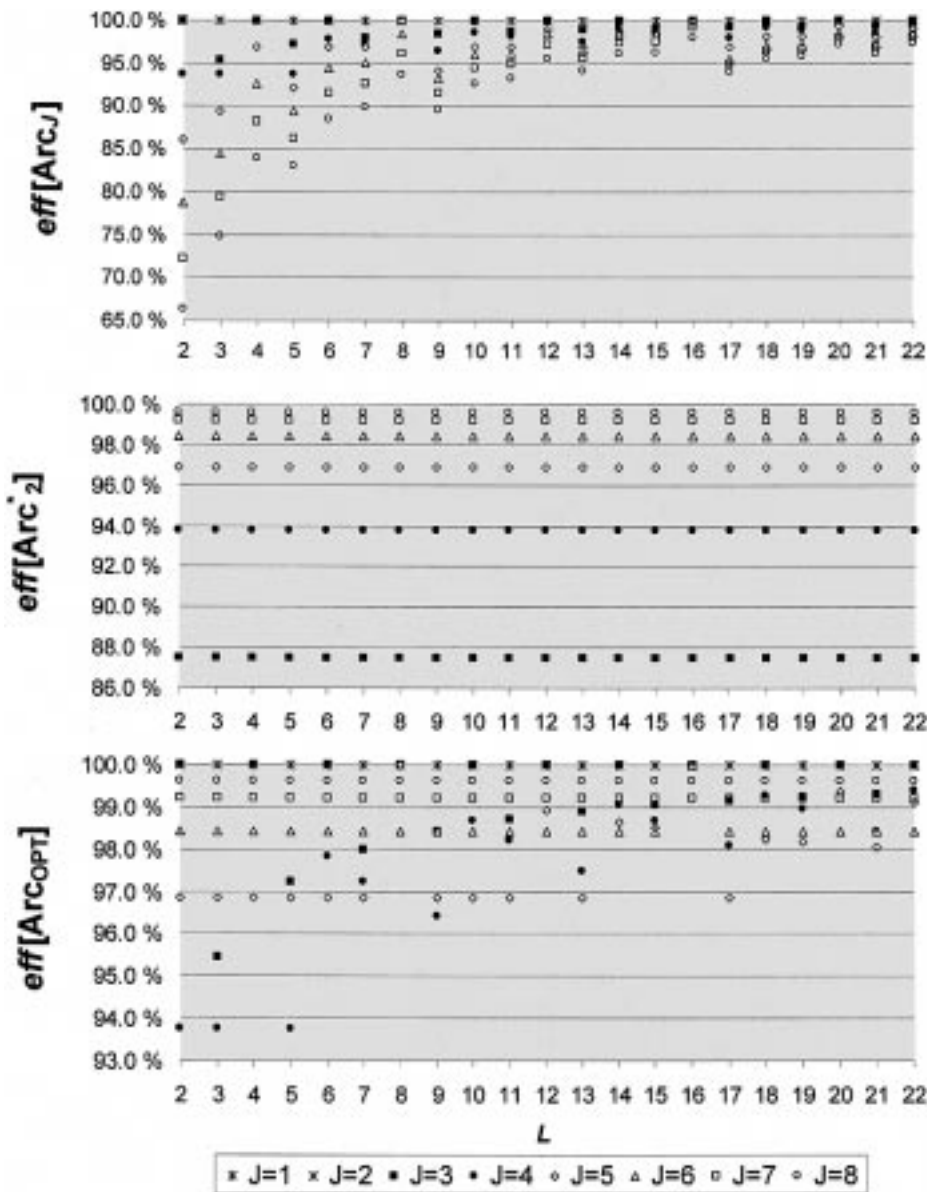


Fig. 8. Efficiencies for different values of J and L .

As was expected, the only parameters that affect the efficiencies are J and L (in particular, $\text{eff}[\text{Arc}_2^*]$ depends only on J), since the designs of both the architectures are independent on N_0 . Therefore, we can characterize any specific problem or application by the coordinates (J, L) of a 2-D space Σ , and plot $\text{eff}[\text{Arc}_J]$ and $\text{eff}[\text{Arc}_2^*]$ for any point of Σ . Fig. 8(a) and (b) shows such a study for a wide subset $\Sigma' = \{(J, L), 1 \leq J \leq 8, 2 \leq L \leq 22\} \subset \Sigma$. $\text{eff}[\text{Arc}_J]$ is 100% in all the points $(1, L)$ and $(2, L)$, which are denoted by “*” in Fig. 8(a). Nevertheless it is worthwhile to note the following.

- 1) For a given value of L , $\text{eff}[\text{Arc}_J]$ decreases as J increases.
- 2) $\text{eff}[\text{Arc}_2^*]$ is independent on L and increases with J .

In other words, Arc_J and Arc_2^* compensate themselves each other in terms of efficiency. This circumstance is visually evident in Fig. 8(a) and (b). In fact, in Fig. 8(a), the white symbols (corresponding to $5 \leq J \leq 8$) label lower efficiencies than those labeled by the black symbols (corresponding to $3 \leq J \leq 4$), which is the opposite of Fig. 8(b).

Therefore, given the parameters of a particular problem or application [i.e., a point $(J', L') \in \Sigma$], the designer can choose that architecture (between $\text{Arc}_{J'}$ and Arc_2^*) that is the most efficient in (J', L') . We call this architecture the *optimal architecture* for (J', L') , and we denote it as $\text{Arc}_{\text{OPT}}(J', L')$, which is simply given by

$$\text{Arc}_{\text{OPT}} \equiv \begin{cases} \text{Arc}_{J'}, & \text{if } \text{eff}[\text{Arc}_{J'}] \geq \text{eff}[\text{Arc}_2^*] \\ & \text{i.e., inequality (7) is not satisfied by } J' \text{ and } L' \\ \text{Arc}_2^*, & \text{if } \text{eff}[\text{Arc}_{J'}] < \text{eff}[\text{Arc}_2^*] \\ & \text{i.e., inequality (7) is satisfied by } J' \text{ and } L' \end{cases} \quad (13)$$

Fig. 8(c) shows $\text{eff}[\text{Arc}_{\text{OPT}}]$ in Σ' . The average value of $\text{eff}[\text{Arc}_{\text{OPT}}]$ in Σ' is 99.1%, and its minimum value is 93.8%.

It is remarkable that except for five points, $\text{eff}[\text{Arc}_{\text{OPT}}]$ is not lower than 96.9%.

VI. CONCLUSION

In this paper, we have proposed two scalable architectures (Arc_J and Arc_2^*) that perform a DWT of an N_0 -sample sequence in only $N_0/2$ ccs. Therefore, the proposed devices are at least twice as fast as the other known architectures. Moreover, they have an AT^2 parameter that is approximately 1/2 that of already existing devices.

This result is twofold. First, Arc_J and Arc_2^* can be employed for performing two times faster processing than that allowed by other architectures working at the same clock frequency (high-speed utilization). Second, they can be employed, even using a two times lower clock frequency, but reaching the same performance of other architectures. This second possibility allows for reducing the supply voltage and the power dissipation, respectively, by a factor of two and four with respect to the other architectures (low-power utilization). These results have been achieved by means of pipelining. Even though other architectures have already exploited the pipelined approach, they do not reach the performance of the proposed architectures, and they result in heavy underutilization. In fact, the stage implementing the decomposition 13 level k in pipelined architectures is usually clocked at a frequency 2^{k-1} times lower than the clock used in the stage performing decomposition level 1. Conversely, our architectures have been designed taking into account the balancing of the pipeline. Therefore, they are highly efficient. Specifically, we have shown that the efficiency of Arc_J decreases with the number of levels J , while the efficiency of Arc_2^* grows with J . Therefore, as a conclusive result, an impressively efficient architecture has been defined [say, $\text{Arc}_{\text{OPT}}(J', L')$], which is simply the architecture between Arc_J and Arc_2^* having the highest efficiency when it implements an L' -tap filter based DWT in J' decomposition levels.

The efficiency of $\text{Arc}_{\text{OPT}}(J', L')$ is excellent. For instance, its average value [computed in very wide set of points (J', L')] is 99.1%. Its minimum value is 93.8%, and, except for five points, the efficiency is not lower than 96.9%.

ACKNOWLEDGMENT

D. Guevorkian would like to express his deepest gratitude to his colleagues from the Signal Processing Laboratory, Tampere University of Technology, where he has been working on this paper.

REFERENCES

- [1] S. G. Mallat, "A theory for multiresolution signal decomposition: The wavelet representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 2, pp. 674–693, Dec. 1989.
- [2] M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding*. Englewood Cliffs, NJ: Prentice-Hall, 1995.
- [3] I. Daubachies, *Ten Lectures on Wavelets*. Philadelphia, PA: SIAM, 1992.
- [4] —, "The wavelet transform, time frequency, localization and signal analysis," *IEEE Trans. Inform. Theory*, vol. 36, pp. 961–1005, Sept. 1990.

- [5] G. Beylkin, R. Coifman, and V. Rokhlin, *Wavelet in Numerical Analysis in Wavelets and their Applications*. New York: Jones and Bartlett, 1992, pp. 181–210.
- [6] —, *Fast Wavelet Transforms and Numerical Algorithms*. New Haven, CT: Yale Univ. Press, 1989.
- [7] L. Senhadji, G. Carrault, and J. J. Bellanger, "Interictal EEG spike detection: A new framework based on the wavelet transforms," in *Proc. IEEE-SP Int. Symp. Time-Frequency Time-Scale Anal.*, Philadelphia, PA, Oct. 1994, pp. 548–551.
- [8] S. G. Mallat, "Multifrequency channel decompositions of images and wavelet models," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 12, pp. 2091–2110, 1989.
- [9] Z. Mou and P. Duhamel, "Short-length FIR filters and their use in fast non recursive filtering," *IEEE Trans. Signal Processing*, vol. 39, pp. 1322–1332, June 1991.
- [10] A. N. Akansu and R. A. Haddad, *Multiresolution Signal Decomposition: Transforms, Subbands and Wavelets*. New York: Academic, 1992.
- [11] R. Kronland-Martinet, J. Morlet, and A. Grossman, "Analysis of sound patterns through wavelet transforms," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 1, no. 2, pp. 273–302, 1987.
- [12] S. B. Pan and R. H. Park, "New systolic arrays for computation of the 1-D discrete wavelet transform," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 5, 1997, pp. 4113–4116.
- [13] A. B. Premkumar and A. S. Madhukumar, "An efficient VLSI architecture for the computation of 1-D discrete wavelet transform," in *Proc. IEEE Int. Conf. Information, Communications and Signal Processing*, 1997, pp. 1180–1184.
- [14] M. Vishwanath, R. M. Owens, and M. J. Irwin, "VLSI architectures for the discrete wavelet transform," *IEEE Trans. Circuits Syst. II*, vol. 42, pp. 305–316, May 1995.
- [15] J. Fridman and S. Manolakos, "Discrete wavelet transform: Data dependence analysis and synthesis of distributed memory and control array architectures," *IEEE Trans. Signal Processing*, vol. 45, pp. 1291–1308, May 1997.
- [16] K. K. Parhi and T. Nishitani, "VLSI architectures for discrete wavelet transforms," *IEEE Trans. VLSI Syst.*, vol. 1, no. 2, pp. 191–202, 1993.
- [17] T. C. Denk and K. K. Parhi, "Systolic VLSI architectures for 1-D discrete wavelet transform," in *Proc. 32nd Asilomar Conf. Signals, Systems & Computers*, vol. 2, 1998, pp. 1220–1224.
- [18] G. Knowles, "VLSI architecture for the discrete wavelet transform," *Electron. Lett.*, vol. 26, no. 15, pp. 1184–1185, 1990.
- [19] C. Chakrabarti and M. Vishwanath, "Efficient realizations of discrete and continuous wavelet transforms: From single chip implementations to mappings on SIMD array computers," *IEEE Trans. Signal Processing*, vol. 43, no. 3, pp. 759–771, 1995.
- [20] C. Chakrabarti, M. Vishwanath, and R. M. Owens, "Architectures for wavelet transforms: A survey," *J. VLSI Signal Process.*, vol. 14, no. 2, pp. 171–192, 1996.
- [21] A. Grzeszczak, M. K. Mandal, and S. Panchanatan, "VLSI implementation of discrete wavelet transform," *IEEE Trans. VLSI Syst.*, vol. 4, pp. 421–433, Dec. 1996.
- [22] M. Vishwanath and R. M. Owens, "A common architecture for the DWT and IDWT," in *Proc. IEEE Int. Conf. Application Specific Systems, Architectures and Processors*, 1996, pp. 193–198.
- [23] C. Yu, C. H. Hsieh, and S. J. Chen, "Design and implementation of a highly efficient VLSI architecture for discrete wavelet transforms," in *Proc. IEEE Int. Conf. Custom Integrated Circuits*, 1997, pp. 237–240.
- [24] S. B. Syed and M. A. Bayoumi, "A scalable architecture for discrete wavelet transform," in *Proc. 15 Computer Architectures for Machine Perception (CAMP '95)*, 1995, pp. 44–50.
- [25] F. Marino, "A 'Double-Face' bit-serial architecture for the 1-D discrete wavelet transform," *IEEE Trans. Circuits Syst. II*, vol. 47, no. 1, pp. 65–71, 2000.
- [26] M. Vishwanath, "The recursive pyramid algorithm for the discrete wavelet transform," *IEEE Trans. Signal Processing*, vol. 42, no. 3, pp. 673–677, 1994.
- [27] D. Liu and C. Svensson, "Trading speed for low power by choice of supply and threshold voltages," *IEEE J. Solid-State Circuits*, vol. 28, no. 1, pp. 10–17, 1993.
- [28] H. T. Kung, "Why systolic architectures?," *IEEE Comput.*, 1982.
- [29] F. Marino, D. Gevorkian, and J. T. Astola, "Highly efficient high-speed/low-power architectures for the 1-D discrete wavelet transform (extended version)," Politecnico di Bari, Tech. Rep. DEE-N. 10/00/S, Apr. 2000.

Francescomaria Marino was born in 1968. He received the Laurea degree (*cum laude*) and the Ph.D. degree in electronic engineering from Polytechnic of Bari, Italy, in 1991 and 1996, respectively.

Since 1999, he has been Assistant Professor in the Department of Electrical and Electronic Engineering, Polytechnic of Bari. In the same year, he became an Invited Visiting Researcher in the Signal Processing Laboratory, Tampere University of Technology. Previously, he worked in the Telecommunications Research Center of the Arizona State University (1998), in the Electrical and Computer Engineering Department of the University of Texas at Austin (1997), and in the Institute of Signal and Image Processing of the Italian National Council of Researches (1996). He has coauthored two patents by Intel and one patent by CNR. His main interests are parallel algorithms and architectures for digital image and signal processing.

Dr. Marino received an award from Firestone S.p.A. as the Best Graduate of the Year in Universities of Puglia and an award from Telecom for his thesis work.

David Guevorkian received the Candidate of Sciences degree in physics and mathematics from Kiev State University, Ukraine, the M.Sc. degree in mathematics from Yerevan State University, Armenia, and the Dr.Tech. degree in signal processing from Tampere University of Technology, Finland, in 1997, where he is pursuing the Ph.D. degree.

He was a Senior Researcher with the Institute for Problems of Informatics and Automation, National Academy of Sciences of Armenia, from 1983 to 1994. He then joined the Signal Processing Laboratory of Tampere University of Technology, where he worked until March 2000. Currently he is with Nokia Research Center, Finland. His research interests include signal and image processing, parallel algorithms and architectures, and computational complexity analysis.

Jaakko T. Astola was born in Helsinki, Finland, on May 6, 1949. He received the B.Sc., M.Sc., Licentiate, and Ph.D. degrees in mathematics (specializing in error-correcting codes) from Turku University, Finland, in 1972, 1973, 1975, and 1978, respectively.

From 1976 to 1977 he was a Research Assistant at the Research Institute for Mathematical Sciences, Kyoto University, Kyoto, Japan. Between 1979 and 1987, he was with the Department of Information Technology, Lappeenranta University of Technology, Lappeenranta, Finland, holding various teaching positions in mathematics, applied mathematics, and computer science. In 1984, he was a Visiting Scientist at Eindhoven University of Technology, the Netherlands. From 1987 to 1992, he was an Associate Professor in applied mathematics at Tampere University, Tampere, Finland. Currently, he is a Professor of signal processing and Head of the Signal Processing Laboratory of Tampere University of Technology, elected by the Finnish Ministry of Education as a Center of Excellence in Research in Finland. He leads a group of about 50 scientists.