VLSI ARCHITECTURE FOR DISCRETE WAVELET TRANSFORM BASED ON B-SPLINE FACTORIZATION

Chao-Tsung Huang, Po-Chih Tseng, and Liang-Gee Chen

DSP/IC Design Lab, Graduate Institute of Electronics Engineering and
Department of Electrical Engineering, National Taiwan University,
Taipei, Taiwan, R.O.C.
Email:{cthuang, pctseng, lgchen}@video.ee.ntu.edu.tw

ABSTRACT

Based on B-spline factorization, a new category of architectures for Discrete Wavelet Transform (DWT) is proposed in this paper. The B-spline factorization mainly consists of the B-spline part and the distributed part. The former is proposed to be constructed by use of Pascal implementation. And the latter is the only part requiring multipliers and can be implemented with Type-I or Type-II polyphase decomposition. Since the degree of the distributed part is usually designed as small as possible, the proposed architectures could need fewer multipliers than previous arts, but more adders would be required. However, many adders can be implemented with small area and low speed adders because only few adders are on the critical path. Two cases of the JPEG2000 defaulted (9,7) filter and the (6,10) filter are given to demonstrate the efficiency of the proposed architectures.

1. INTRODUCTION

DWT has been developed as an efficient and powerful tool for signal analysis and image compression. Because it would require a huge amount of computation, many VLSI architectures have been proposed, which are mainly based on convolution scheme [1, 2] and lifting scheme [3, 4]. The former is to implement two-channel filter banks directly. And the latter is to factorize the polyphase matrix into many lifting steps based on the perfect reconstruction property [5, 6]. The lifting scheme usually requires fewer multipliers and adders than the convolution scheme.

However, the intrinsic B-spline property of DWT was not used to construct VLSI architectures in literature. According to [7], any DWT filters could be B-spline factorized into the B-spline part and the distributed part. The B-spline part contributes to all important wavelet properties. And the distributed part is used to design FIR DWT filters. Since only the distributed part requires multipliers, the B-spline factorization could offer fewer multipliers than the lifting scheme but induce more adders as well.

In this paper, we propose to implement DWT based on the B-spline factorization. Instead of direct implementation, the B-spline part is proposed to be constructed with the Pascal implementation for reducing adders. And the distributed part could be implemented with Type-I or Type-II polyphase decomposition. The organization of this paper is as follows. Section 2 reviews previous arts for DWT architectures. The B-spline factorization theory is described in section 3, and the proposed architectures are presented in section 4. The case studies of the JPEG2000 defaulted (9,7) filter and the (6,10) filter are given in section 5. At last, this paper is summarized in section 6.

2. PREVIOUS DWT ARCHITECTURES

The convolution scheme is to implement DWT with the direct structures of two-channel filter banks. Many design techniques, such as folding, unfolding, and polyphase decomposition, have been adopted to implement the pair of lowpass and highpass filters. However, these two filters are considered independently in this scheme. On the other hand, lifting scheme [5] has been widely used to reduce the required multiplications and additions by exploring the relation of lowpass and highpass filters. Furthermore, any two-channel filter bank of perfect reconstruction property could be factorized into the corresponding lifting structures [6].

Thus, DWT architectures can be categorized as Fig. 1. The general two-channel filter banks can be implemented with convolution scheme. If the two-channel filter bank has the perfect reconstruction property, it could be implemented with fewer arithmetic operations by use of lifting-based architectures. DWT can be implemented with the above schemes because it can be viewed as two-channel filter banks of perfect reconstruction property. However, the B-spline factorization property of DWT has not been used to construct effi-

This work was supported in part by National Science Council under grant NSC91-2215-E-002-035 and by MOE Program for Promoting Academic Excellence of Universities under the grant number 89E-FA06-2-4-8.

0-7803-7795-8/03/ $17.00 © 2003 IEEE

cient architectures in literature, which is an important property for DWT and will be described in the next section.

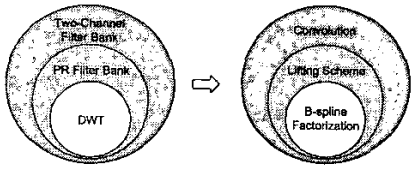


Fig. 1. Category of DWT architectures

3. B-SPLINE FACTORIZATION FOR DWT

According to [7], the lowpass filter, $H(z) = \sum_{i=0}^{P_H-1} h_i z^{-i}$, and highpass filter, $G(z) = \sum_{i=0}^{P_G-1} g_i z^{-i}$, of any DWT can be factorized as

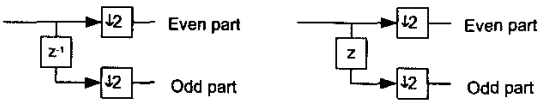
$$H(z) = (1+z^{-1})^{\gamma_H} \cdot Q(Z) \cdot h_0$$
$$G(z) = (1-z^{-1})^{\gamma_G} \cdot R(Z) \cdot g_0 \qquad (1)$$

where the first, second, and third terms of the right-hand side can be called the B-spline part, distributed part, and normalization part, respectively.

The B-spline part is responsible for important properties of DWT, such as order of approximation, reproduction of polynomials, vanishing moments, and multiscale differentiation property. And the distributed part is used to derive efficient FIR DWT filters [7]. As for the normalization part, it is extracted in this paper for implementation issues. It can be implemented independently from the other two parts and further together with the following quantization if image compression is needed. This normalization part is very similar with the normalization step in the lifting scheme. Based on the B-spline factorization, the output of highpass filter is the γ_G-th order difference of the input signals.

4. PROPOSED B-SPLINE FACTORIZED ARCHITECTURE

We propose to implement DWT architectures based on the B-spline factorization of equation (1). The direct implementation of polyphase decomposition of the B-spline part will require $2\gamma_H + 2\gamma_G$ adders for a pair of lowpass and highpass outputs. Instead, the Pascal implementation is proposed for the B-spline part. For example, the implementation of $(1+z^{-1})^4 = 1 + 4z^{-1} + 6z^{-2} + 4z^{-3} + z^{-4}$ and $(1-z^{-1})^4 = 1 - 4z^{-1} + 6z^{-2} - 4z^{-3} + z^{-4}$ can be computed $1 + 6z^{-2} + z^{-4}$ and $4z^{-1} + 4z^{-3}$ first. Then the sum of them is $(1+z^{-1})^4$, and the difference is $(1-z^{-1})^4$. Furthermore, the integer multiplications of the B-spline part are implemented with shifters and adders, instead of multipliers. In this example, the Pascal implementation only requires 12 adders, but the direct implementation will need 16 adders. The distributed part is the only part with multipliers and is implemented by polyphase decomposition. There are two possible types of polyphase decomposition as shown in Fig. 2. Since the normalization part can be implemented independently from the other two parts, it will be excluded in the following discussion.



(a) Type-I (b) Type-II

Fig. 2. Two polyphase decomposition types

The main advantage of the B-spline factorized architectures is that possibly fewer multipliers are required than the convolution and lifting scheme. This is because the degrees of $Q(z)$ and $R(z)$ (γ_Q and γ_R) are designed as small as possible for given γ_H and γ_G, which dominates all wavelet properties. The convolution scheme will require about $\gamma_H + \gamma_G + \gamma_Q + \gamma_R$ multipliers, while the lifting scheme could possibly save a half number of multipliers. But the B-spline factorized architecture only require $\gamma_Q + \gamma_R$ multipliers which are fewer than $\frac{\gamma_H+\gamma_G+\gamma_Q+\gamma_R}{2}$ if $\gamma_Q + \gamma_R < \gamma_H + \gamma_G$. However, the disadvantage is that more adders may be required.

5. CASE STUDIES

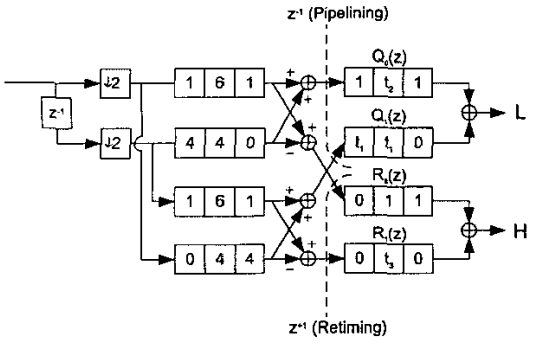
In this section, the JPEG2000 defaulted (9,7) filter and the (6,10) filter [3] are studied and implemented by use of proposed B-spline factorized architectures.

5.1. JPEG2000 defaulted (9,7) filter

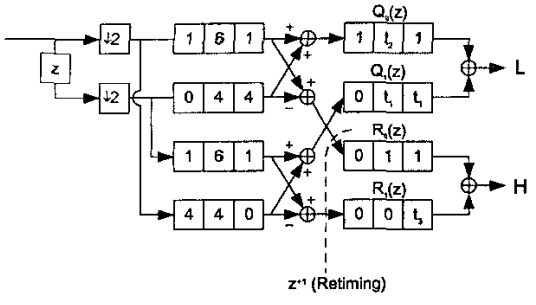
The B-spline factorization of the (9,7) filter can be expressed as:

$$H(z) = (1+z^{-1})^4(1 + t_1 z^{-1} + t_2 z^{-2} + t_1 z^{-3} + z^{-4})h_0$$
$$G(z) = (1-z^{-1})^4(1 + t_3 z^{-1} + z^{-2})g_0 \qquad (2)$$

where $t_1 = -4.630464, t_2 = 9.597484$, and $t_3 = 3.369536$. Thus the B-spline factorized architecture of the (9,7) filter will only need three multipliers, excluding the normalization part h_0 and g_0. However, 18 adders are required, of which 12 adders for the B-spline part and 6 adders for the distributed part. The proposed B-spline factorized architectures are shown in Fig. 3, where Fig. 3(a) and (b) represent Type-I and Type-II polyphase decomposition, respectively. And the notation for FIR filters is described as Fig. 4.



(a) Type-I



(b) Type-II

Fig. 3. Proposed B-spline factorized architectures for (9,7) filter

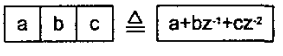


Fig. 4. Notation for filters

The original Type-I architecture requires ten registers, and the critical path is $T_m + 5T_a$, where T_m is the time taken for a multiplication operation, and T_a is the time needed for an addition operation. If retiming is performed through the downside dot line of Fig. 3(a), the number of registers will be decreased from ten to eight without modifying the critical path. On the other hand, if pipelining is performed through the upside dot line, the critical path can be shortened to $T_m + 2T_a$ with totally 12 registers. However, the critical path of the retiming Type-II architecture is $T_m + 2T_a$ with only 10 registers.

5.1.1. Comparison

By extracting the normalization part h_0 and g_0 and utilizing the symmetric property, the convolution-based architecture of the (9,7) filter can be implemented by use of 7 multipliers, 14 adders, and 7 registers. And the critical path is $T_m + 3T_a$ if adder tree is used to connect adders.

The lifting scheme of the (9,7) filter can be factorized as:

$$P(z) = \begin{bmatrix} 1 & a(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ b(1+z) & 1 \end{bmatrix} \begin{bmatrix} 1 & c(1+z^{-1}) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ d(1+z) & 1 \end{bmatrix} \begin{bmatrix} K & 0 \\ 0 & 1/K \end{bmatrix} \quad (3)$$

where P(z) is the polyphase matrix, and the coefficients are given as $a = -1.586134342$, $b = -0.052980118$, $c = 0.882911076$, $d = 0.443506852$, and $K = 1.149604398$. The corresponding signal flow graph is shown in Fig. 5. Thus, the lifting-based architecture would require 4 multipliers and 8 adders if the normalization steps K and $1/K$ are excluded. The critical path $4T_m + 8T_a$ is quite long with only 4 registers and can be reduced to $T_m + 2T_a$ by pipelining through the dot lines with totally 10 registers.

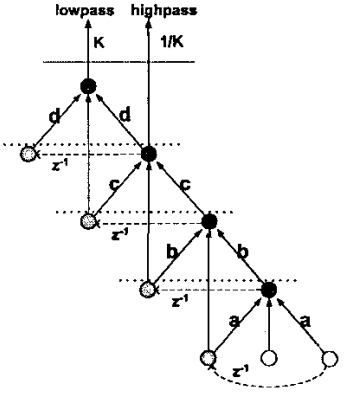


Fig. 5. Lifting scheme for the (9,7) filter

The proposed B-spline factorized architectures as well as the aforementioned convolution-based and lifting-based ones have been verified by use of Verilog-XL and synthesized into gate-level netlists by Synopsys Design Compiler with standard cells from Artisan 0.25-μm cell library. The comparison and synthesis results are shown in Table 1, where the internal bit-widths are all 16-bit, the multipliers are all 16-by-12 multiplications, and the adders are also 16-bit for comparison. The gate counts are given with combinational and non-combinational gate counts separately. The former contributes to the multipliers and adders while the latter is responsible to the registers. For circuit synthesis, the timing constraints are set as tight as possible.

According to Table 1, the proposed architectures could require fewer gate counts under the same timing constraints. Although the proposed architectures need more adders, most of the adders are not on the critical path such that they can be implemented with smaller area and lower speed adders. Furthermore, the saving of gate counts will be more signifi-

cant if the multipliers are required to be more floating-point precision.

5.2. The (6,10) filter

The B-spline factorization of the (6,10) filter [3] can be expressed as:

$$\begin{aligned} H(z) =& (1+z^{-1})^3(1+s_3z^{-1}+z^{-2})h_0 \\ G(z) =& (1-z^{-1})^3(1-z^{-1})^2 \\ & (1+s_1z^{-1}+s_2z^{-2}+s_1z^{-3}+z^{-4})g_0 \qquad (4) \\ =& (1-z^{-1})^3 \\ & (1+r_1z^{-1}+r_2z^{-2}+r_3z^{-3}+r_2z^{-4}+r_1z^{-5}+z^{-6})g_0 \qquad (5) \end{aligned}$$

where $s_1 = -t_1$, $s_2 = t_2$, $s_3 = -t_3$, $r_1 = 2.630464$, $r_2 = 1.336557$, and $r_3 = -9.934042$. However, the Pascal implementation can only cover $(1 \pm z^{-1})^3$, and there are two solutions, Solution-1 and Solution-2, for the left part $(1-z^{-1})^2$ of $G(z)$, which are corresponding to equations (4) and (5). The proposed architectures are shown in Fig. 6, where the parts marked with '*' and '**' can be shared. Thus, the Solution-1 of the B-spline factorized architecture would require 3 multipliers and 20 adders while the Solution-2 would need 4 multipliers and 18 adders.

The critical path of the Solution-1 architecture could be $T_m + 6T_a$, $T_m + 4T_a$, or $T_m + 2T_a$ by retiming, pipelining, or retiming and pipelining together. The corresponding numbers of registers are 9, 11, and 13. On the other hand, the Solution-2 architecture can be retimed to obtain a critical path of $T_m + 5T_a$ with totally 9 registers.

5.2.1. Comparison

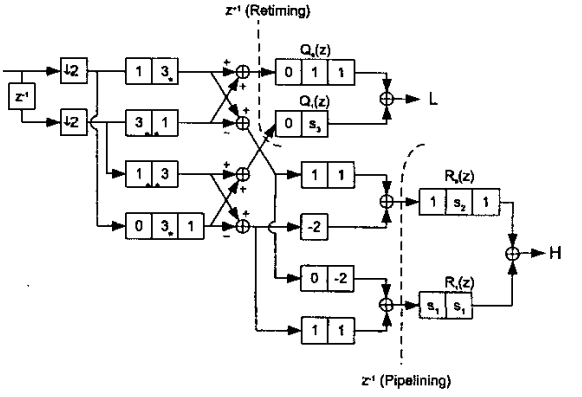
By extracting the normalization part h_0 and g_0 and utilizing the anti-symmetric property, the convolution-based architecture of the (6,10) filter can be implemented by use of 6 multipliers, 14 adders, and 8 registers. And the critical path is $T_m + 4T_a$ if the adder tree is used.

In contrast to the odd symmetric (9,7) filter, the polyphase matrix of the even anti-symmetric (6,10) filter can be decomposed as follows:

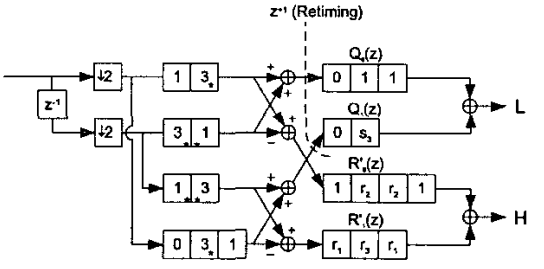
$$\begin{aligned} P(z) = & \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ b+cz^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & e+dz \\ 0 & 1 \end{bmatrix} \\ & \begin{bmatrix} 1 & 0 \\ -f+gz^{-1}+fz^{-2} & 1 \end{bmatrix} \begin{bmatrix} K_2 & 0 \\ 0 & K_1 \end{bmatrix} \end{aligned} \qquad (6)$$

where the coefficients are given as $a = -0.369536$, $b = -0.42780$, $c = -0.119532$, $d = -0.090075$, $e = 0.872739$, $g = -0.572909$, $f = 0.224338$, $K_1 = 0.874919$, and $K_2 = 1.142963$ [3]. Thus, the lifting-based architecture can be shown as Fig. 7, where 7 multipliers, 8 adders, and 5 registers are required if K_1 and K_2 are excluded.

As the case of (9,7) filter, the proposed, convolution-based, and lifting-based architectures have been verified and synthesized. The bit-width and timing constraints are all the same as the case of (9,7) filter. The results are listed in Table 2. In this case, the lifting-based architecture requires more multipliers than the convolution-based one because the lifting scheme of even anti-symmetric DWT filters is not as efficient as that of odd symmetric filters. However, the proposed B-spline architecture can still reduce the number of multipliers to three. Table 2 shows that the proposed architectures can achieve the same timing constraints with fewer gate counts than the other two architectures.



(a) Solution-1 Type-I



(b) Solution-2 Type-I

Fig. 6. Proposed B-spline factorized architectures for the (6,10) filter

6. SUMMARY

In this paper, efficient DWT architectures are proposed on the basis of B-spline factorization. The B-spline part is proposed to be implemented with Pascal implementation. And the distributed part could be implemented with Type-I or Type-II polyphase decomposition. Because the distributed part is usually designed as smaller as possible, the proposed B-spline factorized architectures consist of fewer multipliers than the convolution-based and lifting-based ones. Although more adders are required, many adders can be im-

Architecture	Multiplier	Adder	Critical Path	Register	Timing (ns)	Comb. Gate Count	Non-comb. Gate Count
Lifting + no pipelining	4	8	4Tm+8Ta	4	34	15418.4	796.0
Lifting + 4 pipelining stages	4	8	Tm+2Ta	10	9.8	15152.0	1495.3
Convolution + no pipelining	7	14	Tm+3Ta	7	10.8	17830.5	1266.7
B-spline Type-I + retiming	3	18	Tm+5Ta	8	13.6	9670.4	1271.3
B-spline Type-II + retiming	3	18	Tm+2Ta	10	10.3	9419.4	1523.3

Table 1. Comparisons for DWT architectures of the (9,7) filter

Architecture	Multiplier	Adder	Critical Path	Register	Timing (ns)	Comb. Gate Count	Non-comb. Gate Count
Lifting + no pipelining	7	8	4Tm+5Ta	5	26.6	12664.0	929.3
Lifting + 4 pipelining stages	7	8	Tm+2Ta	11	9.2	14181.7	1679.0
Convolution + no pipelining	6	14	Tm+4Ta	8	12	13685.4	1332.0
Bspline-1 Type-I + retiming	3	20	Tm+6Ta	9	14.1	9623.7	1322.3
Bspline-1 Type-I + retiming + pipe.	3	20	Tm+4Ta	11	11.5	8788.7	1571.0
Bspline-1 Type-I + pipelining	3	20	Tm+2Ta	13	9.5	8648.7	1782.0
Bspline-2 Type-I + retiming	4	18	Tm+5Ta	9	13.1	11647.7	1366.7

Table 2. Comparisons for DWT architectures of the (6,10) filter

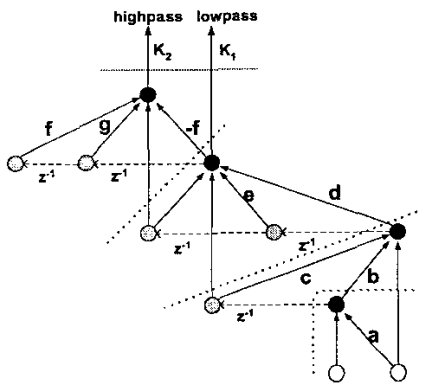


Fig. 7. Lifting scheme for the (6,10) filter

plemented with small area and low speed adders because most of them are not on the critical path. Based on two case studies of the (9,7) and (6,10) filters, the efficiency of the proposed architectures is proven.

7. REFERENCES

[1] K. K. Parhi and T. Nishitani, "VLSI architectures for discrete wavelet transforms," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 1, no. 2, pp. 191–202, June 1993.

[2] C. Chakrabarti, M. Vishwanath, and R. M. Owens, "Architectures for wavelet transforms: A survey," *Journal of VLSI Signal Processing*, vol. 14, pp. 171–192, 1996.

[3] K. Andra, C. Chakrabarti, and T. Acharya, "A VLSI architecture for lifting-based forward and inverse wavelet transform," *IEEE Transactions on Signal Processing*, vol. 50, no. 4, pp. 966–977, Apr. 2002.

[4] W. Jiang and A. Ortega, "Lifting factorization-based discrete wavelet transform architecture design," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 5, pp. 651–657, May 2001.

[5] W. Sweldens, "The lifting scheme: a custom-design construction of biorthogonal wavelets," *Applied and Computaional Harmonic Analysis*, vol. 3, no. 15, pp. 186–200, 1996.

[6] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *The Journal of Fourier Analysis and Applications*, vol. 4, pp. 247–269, 1998.

[7] M. Unser and T. Blu, "Wavelet theory demystified," *IEEE Transactions on Signal Processing*, vol. 51, no. 2, pp. 470–483, Feb. 2003.