TWO LEVEL COST-QUALITY OPTIMIZATION OF 9-7 LIFTING-BASED DISCRETE WAVELET TRANSFORM

Alireza Aminlou, Fatemeh Refan, Mahmoud Reza Hashemi, Omid Fatemi

Multimedia Processing Laboratory, School of Electrical and Computer Engineering,
University of Tehran, Tehran, Iran

ABSTRACT

Implementing the Discrete Wavelet Transform, which is being increasingly recognized in image/video compression standards, in hardware is highly area-consuming. In this paper, a new high-performance lifting-based architecture with optimized error vs. hardware cost is proposed for the 9-7 DWT. In the proposed architecture each constant coefficient multiplier of the conventional lifting structure is split into two new constant multipliers in order to minimize the hardware implementation cost and quantization error. Using an optimization process the appropriate coefficients are determined according to the hardware cost and quality requirements of each application. Simulation results indicate an average quality improvement of 13.5 dB with the same hardware resources. For achieving the same quality, it requires 40% less hardware resources, which makes it suitable for embedded systems.

Index Terms— Discrete wavelet transform, constant multiplier, lifting-based architecture, Embedded systems

1 INTRODUCTION

The rapid growth of visual media in many applications has led to a variety of image and video compression standards. The Discrete Wavelet Transform (DWT), a popular domain transform, separates high and low frequency characteristics of an image to further improve the coding efficiency. Convolution is the conventional method to implement DWT, while the lifting scheme is an efficient DWT implementation method [1]. The lower computational complexity and reduced memory requirements of lifting-based DWT have made it the best choice for hardware implementations.
Many researches have focused on the architectural complexities including several convolution-based architectures introduced in [2], and DWT architectures based on the lifting scheme [3], including one-dimensional (1-D) and two dimensional (2-D) implementations [4]. On the other hand, in order to optimize the lifting structure of DWT, flipping architecture [5] has been introduced, in which the critical path and memory requirements are reduced by scaling the constant coefficients. [6], [7] and [8] have also focused on the efficient quantization and its effect on the performance of the lifting structure.
Although many studies have been performed on the lifting structure, only few of them have focused on either optimizing the computation engine on the basis of modifying the constant coefficients [5], or the effect of quantizing them [6], [7] and [8]. The computation engine of the lifting scheme consists of a number of constant multipliers, whose hardware implementation is area and power consuming. In this paper, a split structure is proposed which offers a flexible method for designing an optimized cost-error architecture for the computation engine of the lifting method. In the proposed technique original lifting coefficients are first split into two constant multipliers and optimized by applying a local optimization. Then, the lifting structure is optimized by analyzing different combinations of the introduced constant multipliers.
The paper is organized as follows: In section 2, the proposed architecture including split constant multiplier and optimized lifting-based DWT architecture, are proposed. The achieved performance is demonstrated in section 3, followed by conclusions.

2 PROPOSED METHOD

According to the fact that the constant multiplier is the most important and area-consuming module of 9-7 lifting structure, the proposed method in this paper optimizes 2-D DWT by optimizing its constant multipliers. First the hardware cost and error of each constant coefficient multiplier is optimized individually; next the obtained constant coefficients are used to improve the overall hardware cost and quality of the lifting structure of 2D 9-7 DWT.

2-1 Split Constant multiplier

An array multiplier as the primary architecture of constant multipliers consists of a number of rows of n-bit adders, where *n* is the bit-width of the input. The number of rows of n-bit adder is equal to the number of '1' bits of the constant coefficient minus one. The area occupied by the constant multiplier is equal to the total number of 1-bit full adder cells multiplied by the area of a 1-bit adder. To put it more simply, the hardware cost of a constant multiplier is defined as the number of 1-bit full adder cells, which depends on the bit-width of input and the number of adders, which in turn depends on the '1' bits of the constant value. In order to minimize the number of '1' bits, the Canonical Signed Digit

(CSD) representation has been proposed [9]. CSD takes advantage of the two common methods, normal and booth multiplier. As a result, the exact hardware cost can be modeled by (1) in terms of 1-bit full adder cells, where *const_ones* is the number of '1's of the constant coefficient in the CSD representation.

$$Cost = (const\_ones - 1) \times n \tag{1}$$

The error at output of a constant multiplier, that exists due to the quantization of the constant coefficient, is defined as the difference of the real output and the ideal output and calculated by (2). The ideal output (z) is the result of the multiplication with ideal coefficient, and the real output (z') is achieved from the quantized coefficient.

$$Error = \sum (z - z')^2 \tag{2}$$

2-1-1 The main idea of Split Multiplier

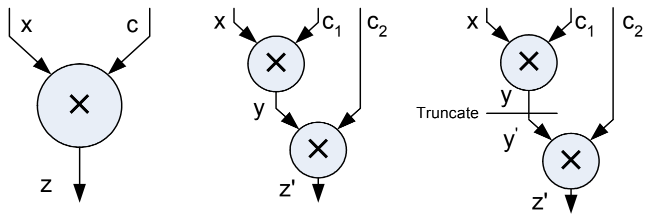
The main idea behind the Split method is to break the constant coefficient, *c*, into two new constant coefficients such that *c* is equal to c_1 multiplied by c_2. Using this technique and internal node quantization, discussed later, three changing factors are introduced to optimize the hardware cost and quantization error of the multiplier. As shown in Figure 1, the multiplication is being performed in two phases; in the first phase the input variable is multiplied by c_1, and in the second phase the result is multiplied by c_2. In this figure *y* is the partial result (internal node), *y'* is the quantized value of *y*, and *z* and *z'* are the outputs of the original and the proposed architecture, respectively. Quantizing the partial result, which means reducing its bit-width, imposes error on the final output while reducing the hardware cost of the second multiplier, c_2. As simulation results show, the trade-off between the cost and the error, imposed by partial result quantization, introduces improved cost-error points.

The hardware cost of the proposed architecture is equal to the summation of hardware cost of constant multipliers c_1 and c_2. Hardware cost of the first multiplier is equal to number of '1' bits of first coefficient minus one ($n_{c1}-1$) multiplied by the bit width of the inputs (n_x). As, the bit width of its output is ($n_{c1}+ n_x$), the bit width of the second multiplier input after applying the truncation is ($n_{c1}+n_x-n_q$). As a result, hardware cost of the proposed structure can be calculated using (3) where n_q is the number of truncated bits of the partial result.

According to (2), error is calculated using (4) where e_q is the error introduced by performing quantization on the partial result, *x* is the input of constant multiplier, *p(x)* is its probability distribution and *N* is the largest value of *x*.

$$Cost = (n_{c1} - 1)n_x + (n_{c2} - 1)(n_x + n_{c1} - n_q) \tag{3}$$

$$Error = \sum_{x=0}^{N-1} \Big(p(x)\big(cx - c_2\big(c_1 x - e_q\big)\big)^2 \Big) \tag{4}$$



(a) (b) (c)

Figure 1.Constant multiplier, (a) Standard, (b) Split main idea (c) Split structure

2-1-2 Optimization process

There are three parameters that are changed during the optimization procedure by using individual loops for each of them, as shown in Figure 2. The first parameter is the value of c_1 that is chosen according to its bit width (n_{c1}), and varies from 0.5 to 1.0, with steps of $1/2^{nc1}$. The second parameter is the number of bits truncated from the partial result (n_q), and the third is the bit width of c_2, (n_{c2}) which is iterated in the last loop. Then in the body of these nested loops, c_2 is calculated and error and cost are estimated using (3) and (4). Finally, the corresponding cost-error point is added to the optimum solution set if it offers a better cost or error comparing to existing results. Using the optimization process the best coefficients (c_1 and c_2) and their corresponding bit widths (n_{c1}, n_{c2}, n_q) will be obtained for various cost or error constraints.

It should be noted that c_2 is the quantized value of c/c_1. If truncation is used, the lower bits of c_2 are dropped, while when using rounding, c_2 is approximated to its nearest number (up or down) according to its bit width (n_{c2}). Simulation results show that rounding is more precise than truncation and leads to better cost-error results.

```
for all c1 values
 for all truncation bits of y′ ($n_q$)
  for all bit_width of c2 ($n_{c2}$)
   Calculate c2
   Calculate cost-error //(3) and (4)
   Add to optimum curve if qualified
```

Figure 2. Optimization process for a constant multiplier

2-1-3 Experimental results of Split multiplier

The experimental results of optimizing one of the constant multipliers in the lifting structure (α) is shown in Figure 3. In this figure, there are three cost-PSNR curves corresponding to a standard multiplier, and split multipliers using truncation and rounding. The standard curve is obtained by changing the bit width dedicated to the constant coefficient (α) and calculating its hardware cost and error. The two other curves are achieved using split idea and the optimization process, shown in Figure 2. It is shown that both curves of split method have better PSNR comparing to the standard multiplier. Also, the rounding method shows a better performance than a simple truncation. Hence, the rounding technique is used to optimize the lifting coefficients, discussed later.

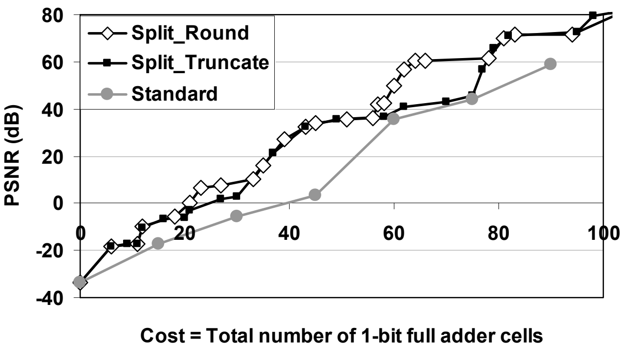


Figure 3. Cost-PSNR of the proposed and standard structures for α = -1.56613 = 1.10010000111011

2-2 Optimizing the DWT Architecture

Our main goal in this paper is to perform optimization on cost-error of the computation core of 9-7 DWT by focusing on the constant multipliers of lifting structure. This paper does not tend to propose a new architecture; instead a general method is proposed that can be applied to any system level designed architecture. Our main idea for optimizing cost and error of the lifting structure is to use a two level optimization, local and global. In the local step, constant multipliers of lifting coefficients are optimized using the split method proposed in the previous section, resulting in an optimum cost-error curve for each coefficient. Then in the global step, cost and error of DWT lifting structure is optimized using different combinations of the optimized coefficients.

2-2-1 DWT Architecture

During the wavelet step of JPEG2000 encoder [10], image is decomposed into four distinct LL, HL, LH and HH subbands, as shown in Figure 4. LL which is the low-resolution subband can be decomposed into four subbands, recursively. The wavelet subbands are individually coded and transmitted to the decoder where by applying the inverse wavelet transform, the image is reconstructed.

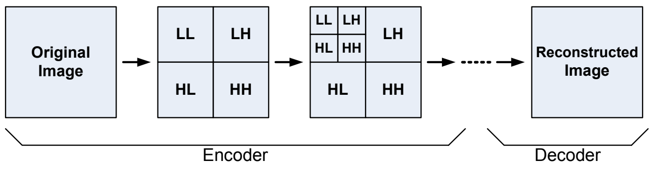


Figure 4. 2D multi resolution wavelet and Subbands

2-D DWT consists of two one-dimensional (1-D) wavelet transforms which act as the computational engine of DWT module with lifting structure, as shown in Figure 5. Outputs of this module, Y_i, are calculated in six steps, where X_i are inputs of the engine, α, β, γ, δ, k and k^{-1} are constant coefficients, and P, Q, R and S are internal nodes. These constant multipliers are area and power-consuming which highlights the importance of optimizing this structure.

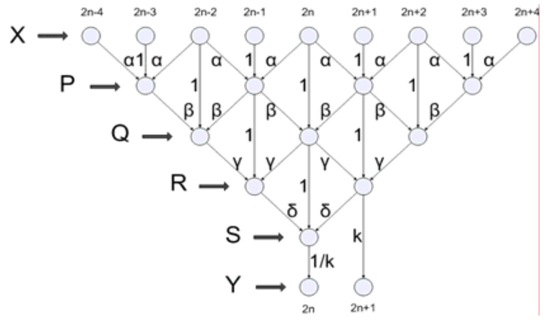


Figure 5. The lifting-based structure of 9-7 DWT filter

2-2-2 Cost and Error Estimation

As it is expected the overall cost is simply equal to the summation of all constant coefficient multipliers cost. It is assumed that the output bit widths of the internal multiplier nodes in lifting structure are the same as their input bit widths. This means that the quantization errors of internal nodes of the lifting are ignored for both the standard and the proposed structures.

Overall error of DWT is defined as the quality difference of the input image that is applied to the encoder and the reconstructed image after the completion of decoding. To focus on the effect of lifting optimization, it is assumed that the other steps of the encoder and decoder are lossless. Also, it is supposed that the reconstructed image using the high precision coefficients is the reference image to estimate the error. Following the above assumptions, error calculation consists of two phases: first the effect of coefficient's error on the subbands' error is calculated for the encoding step. Then error of the reconstructed image is estimated using subbands' error, propagated through the decoding step.

The error introduced by splitting each of the six coefficients in 9-7 DWT, causes error on each subband with different weights. So, in the first phase, the subband error is calculated as the weighted sum of the coefficients' error by (5). The error values introduced by coefficients cancel each other, therefore some of the weight factors are positive, and some of them are negative. These weights are extracted for each subband using simulation, as reported in Table 1. In the second phase, the error in the reconstructed image is calculated as the weighted sum of the subband errors using (6). These weights are L2Norm factors that show the effect of each subband on the reconstructed image [10].

$$E_s = \left(w_\alpha \Delta\alpha + w_\beta \Delta\beta + w_\gamma \Delta\gamma + w_\delta \Delta\delta + w_k \Delta k + w_{rk} \Delta rk\right)^2 \qquad (5)$$

$$\Delta\alpha = \alpha - \alpha_1\alpha_2 \quad \Delta\beta = \beta - \beta_1\beta_2 \quad \Delta\gamma = \gamma - \gamma_1\gamma_2 \quad \cdots$$

$$E_{total} = \sum\left(w_s E_s\right) \qquad s = LL, HL, LH, HH \qquad (6)$$

Table 1. Weight of coefficients' error on subband error

	w_α	w_β	w_γ	w_δ	w_k	w_{rk}
LL	98	-870	172	8.5	125	0
HL	50	-240	70	2.8	6	8
LH	50	-240	70	3.2	4	5.5
HH	5	-23	6	0	0	6

2-2-3 DWT Optimization

The process of optimizing the DWT structure consists of two phases, as shown in Figure 6. First all coefficients of lifting structure are optimized using the split method, described in Figure 2. Then six constant coefficients are iterated on the optimum choices using individual loops for each. In the body of these nested loops, total cost and overall error are calculated. The achieved coefficients are added to the optimum coefficient set if they offer a better cost or error comparing to existing ones.

```
for coefficients (α, β, γ, δ, k, k^-1)
 run split method (Figure 2)
for all optimum points of α
 for all optimum points of β
  for all optimum points of γ
   for all optimum points of δ
    for all optimum points of k
     for all optimum points of k^-1
       Calc cost-error
       Add to optimum list if qualified
```

Figure 6. The optimization process of DWT lifting structure

3 SIMULATION RESULTS

In this section, the cost-PSNR of the 9-7 DWT structure, optimized by split method, is compared to that of the standard structure with original coefficients. For the split method the optimization process of Figure 6 is used to derive coefficients, while in the case of standard structure, the original coefficients of lifting are selected by changing the number of quantized bits. The ranges of bit width applied in the simulation of each method, and the bit width assumed for the input of each multiplier are shown in Table 2.

In order to calculate the improvement of the proposed method, the cost-error results of these two simulations are compared in Figure 7. In this figure, the horizontal axis is hardware cost which is equal to the total number of full adder cells as defined in (3), while the vertical axis represents the corresponding PSNR quality measured in dB unit, calculated by (6). Figure 7 shows that the proposed method improves the quality by 13.5 dB in average (3.0-30.0dB), without increasing the hardware cost. Similarly, for achieving the same quality the proposed architecture is 40% area consuming in terms of 1-bit full adder cell.

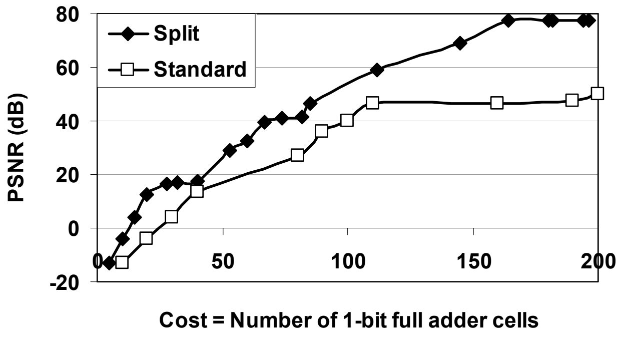


Figure 7. Cost-quality of proposed and standard structures

Table 2. Bit width of different nodes in simulation

Node	Bit	Node	bit
input	10	standard coefficients	1-15
split coefficients	1-15	internal node truncation	0-5

4 CONCLUSION

This paper addressed the trade-off in the terms of hardware cost and error of 9-7 DWT by optimizing the constant coefficient multipliers of lifting structure. In order to improve the performance of this structure we proposed a split architecture, in which each constant coefficient was substituted by two new constant coefficients to gain cost-error improvement. Then, using an optimization process, the new optimized coefficients were chosen according to the equations derived for hardware cost and PSNR calculation. Simulation results show an average quality improvement of 13.5 dB with the same hardware resources, and hardware reduction of 40% for the same PSNR quality. Offering higher quality and consuming lower hardware resources shows its superior performance for embedded applications.

REFERENCES

[1] W. Sweldens, “Wavelets and the lifting scheme: A 5 minute tour,” Z. Angew. Math. Mech., vol. 76, no. 2, pp. 41–44, 1996.

[2] Chakrabarti C, Vishwanath M, Owens RM, “Architectures for wavelet transforms: A survey”, Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology, vol. 14, no. 2, pp. 171-192, November 1996.

[3] Acharya T, Chakrabarti C, “A survey on lifting-based Discrete Wavelet Transform architectures”, Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology, vol. 42, no. 3, pp. 321-339, 2006

[4] Zervas ND, Anagnostopoulos GP, Spiliotopoulos V, et al. “Evaluation of design alternatives for the 2-D-discrete wavelet transform” IEEE Tran. on Circuits and Systems for Video Technology, vol. 11, no. 12, pp. 1246-1262, 2001.

[5] C.T. Huang, P.C. Tseng, and L.G. Chen, “Flipping structure: an efficient VLSI architecture for lifting-based discrete wavelet transform,”IEEE Transactions on Signal Processing, pp. 1080-1089, April 2004.

[6] K. A. Kotteri, A. E. Bell and J. E. Carletta, “Improving the Performance of the Quantized Biorthogonal 9-7 Wavelet Filters,” Proceedings of the 10th IEEE Digital Signal Processing Workshop, Pine Mountain, GA, October 2002.

[7] V. Spiliotopoulos, N. D. Zervas, Y. Andreopoulos, G. Anagnostopoulos, and C. E. Goutis, “Quantization Effect on VLSI Implementations for the 9-7 DWT Filters,” Proc. ICASSP, vol. 2, pp. 1197–1200, 2001.

[8] S. Barua, K. A. Kotteri, A. E. Bell and J. E. Carletta, “Optimal, Quantized Lifting Coefficients for the Biorthogonal 9-7 Wavelet," Proceedings of ICASSP, Montreal, Canada, vol. V, pp. 193-196, May 2004.

[9] Israel Koren, "Computer Arithmetic Algorithms", A. K. Peters, 2nd edition, 2001.

[10] "JPEG2000 part I final draft international standard," ISO/IEC JTC1/SC29/WG1 N1890, Sept 2000.