AN ARCHITECTURE FOR FAST WAVELET TRANSFORM

Charles C. Chang[‡] and Jyh–Charn Liu[†]

[‡]Department of Electrical Engineering
[†]Department of Computer Science
Texas A&M University
College Station, TX 77843–3128

ABSTRACT

In this paper we propose a simple yet effective architecture for fast wavelet transform. Our system incorporates the use of fixed point numbers which results in a simpler design than the conventional processor with a floating point arithmetic unit. Calculations are further reduced using our *thresholding-in-calculations* (TIC) technique to omit calculations of small terms that are negligible to the accumulated sum. This reduction exhibits a significant performance gain while maintaining reasonable accuracy.

Key Words — wavelet transform, fixed point numbers, scaled coefficients, thresholding-in-calculations.

1 Introduction

Wavelet transform has gained much interests from researchers in recent years. Unlike the Fourier transform, wavelet transform is localized in time domain which makes this unique property very useful in applications such as data compression, feature detection, and noise removal from signals. Many of these applications demand so much computation power that special hardware dedicated to wavelet transform is called for. A superior processor design should be either faster or less complex than the conventional design. In this paper, we will propose a simple architecture for wavelet transform that performs faster than the conventional method. The increase of speed in our system is attributed to two reasons: a simpler data representation and thresholding of the less significant intermediate results. Our system incorporates a thresholding-in-calculations (TIC) technique that performs thresholding while computing intermediate results; therefore, a significant speed gain results from avoiding unnecessary calculations. The details of how these tasks are done will be explained in the next few sections.

This paper is organized as follows: section 2 briefly explains the basics of discrete wavelet transform, section 3 discusses the nature of the data representation used in our system, section 4 continues the discussion on how to further increase the performance by omitting the less significant terms, section 5 gives an overview of the system architecture, section 6 discusses the simulation results, and section 7 concludes the discussion of this paper.

2 Wavelet Analysis Basics

The concept of wavelet transform is basically to represent an arbitrary function by super-positioning wavelets at different scales, or resolutions. For a discrete function, the discrete wavelet transform (DWT) is based on this two-scale relation

$$\phi(2^{j+1}t - l) = \sum_k a_{2k-l}\phi(2^j t - k) + b_{2k-l}\psi(2^j t - k). \tag{1}$$

This decomposition projects the original signal to two sub-bands: the band-pass sub-band, represented by

$$c_k^j = \sum_l a_{2k-l} c_l^{j+1}, \tag{2}$$

and the high-pass sub-band,

$$d_k^j = \sum_l b_{2k-l} c_l^{j+1}. \tag{3}$$

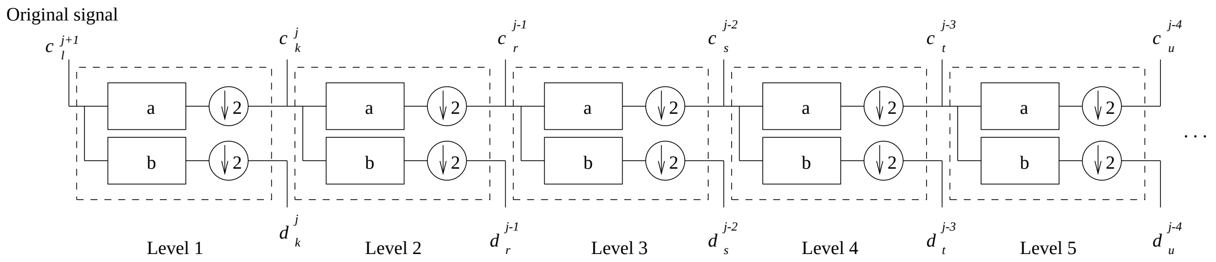

Figure 1: The hierarchy of the pyramid scheme for wavelet decomposition.

If c_k^j is used to obtain c_r^{j-1} and d_r^{j-1}, and c_r^{j-1} is further decomposed into c_s^{j-2} and d_s^{j-2}, a hierarchy of coefficients like the one shown in figure 1 is generated. The method to generate this structure is known as the *pyramid scheme* [1, 2, 3].

For wavelets with compact support such as the Daubechies wavelets [3], the number of coefficients a_i and b_i are finite. The implementation of this type of wavelet transform is similar to those of finite impulse response filters (FIR). Other types of wavelets with compact support such as the Spline Variant with Less Dissimilar Length [4], can also be implemented in a similar fashion. For more details and proofs about these wavelets, one can refer to [3, 4, 5].

3 The Drawbacks of Floating Point Arithmetics

Floating point numbers provide a great dynamic range of numbers for arithmetic operations. Because of this flexibility, floating point arithmetic units have become the engine of most digital signal processing (DSP) hardware. However, floating point computations involve additional procedures compared to integer computations; therefore, a floating point arithmetic unit contains more hardware than an integer arithmetic unit. If we use only the integer portion to compute wavelet transform, calculations using the filter coefficients whose range is between -1 and 1 will produce inaccurate results since the fractions are not accounted for. Therefore, floating point arithmetic has been an essential element in wavelet transform since integer operations do not provide accurate results.

An observation made from implementing the wavelet transform algorithm with compact support coefficients is that the length of each filter is finite and thus small enough to be easily managed. Because of the compact support and the small magnitudes of coefficients, the wavelet transform

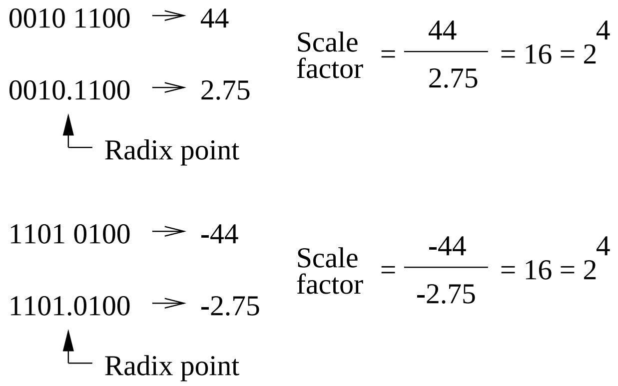

Figure 2: Examples of how to represent fixed point numbers using integers.

using these types of coefficients does not fully utilize the wide dynamic range provided by the floating point numbers. An alternative approach is to use fixed point numbers instead to calculate the wavelet transform. Fixed point numbers have arithmetic properties similar to integers, yet fixed point numbers also include fractions that integers lack.

The representation of a positive fixed point number is basically the same as an integer. The fraction part of a number is to the right of the radix point; for example, $2.75_{10} = 2^1 + 2^{-1} + 2^{-2}$ can be represented as 10.11_2. On the other hand, a negative fixed point number is represented by the two's complement of its magnitude. The two's complement is obtained by adding a 1 to the least significant bit after the bitwise inversion of the fixed point number. For example, 2.75_{10} can be represented by 0010.1100_2. After the bitwise inversion, one is added to the least significant bit which the resultant then becomes 1101.0100_2. Therefore, -2.75_{10} is represented by 1101.0100_2 in this case.

A nice property of fixed point numbers is that if the radix point is removed, the remaining digits represent an integer that is 2^N times larger than the original number, where N is the number of bits used to represent the fraction of that fixed point number. In other words, a fixed point number can be represented by an integer with the same digits and a scale factor; moreover, the value of this scale factor depends on where the radix point is located. This means that we can treat fixed point numbers like integers as long as we maintain a proper scale factor. Figure 2 illustrates some examples of the scaling relationships between the integers and the fixed point numbers they represent.

	Dec.	p_k	αp_k	$p'_k = \lceil \alpha p_k \rceil$
	Hex.	IEEE floating point standard	IEEE floating point standard	2's complement integer
p_0	Dec.	0.48296291	4051386.5	4051387
	Hex.	3ef746ea	4a7746ea	003dd1bb
p_1	Dec.	0.83651630	7017207.5	7017208
	Hex.	3f5625ef	4ad625ef	006b12f8
p_2	Dec.	0.22414386	1880255.0	1880255
	Hex.	3e6585f8	49e585f8	001cb0bf
p_3	Dec.	-0.12940952	-10885565.8	-10885565.0
	Hex.	be0483ee	c98483ee	ffef6f83

Table 1: The original and scaled Daubechies wavelet coefficients (four-tap) ($\alpha = 2^{23}$).

Let a'_k, b'_k be the integers representing the scaled sequences, i.e. $a'_k = \alpha a_k$, $b'_k = \alpha b_k$, and $\alpha = 2^N$, where N is the number of bits used to represent the fraction part of a_k and b_k. After substituting a'_k and b'_k into (2) and (3), we get

$$c'^j_k = \sum_l \frac{a'_{2k-l} c'^{j+1}_l}{\alpha} \tag{4}$$

, and

$$d'^j_k = \sum_l \frac{b'_{2k-l} c'^{j+1}_l}{\alpha}, \tag{5}$$

where $c'^j_k = \alpha c^j_k$, $d'^j_k = \alpha d^j_k$, and $c'^{j+1}_l = \alpha c^{j+1}_l$. Since the scaled coefficients a'_k, b'_k, c'^j_k, d'^j_k, and c'^{j+1}_l are all integers, we can perform wavelet transforms with only integer additions and multiplications. Furthermore, since α is a power of 2, the division of α can be realized by right shifting the integer N times.

Let us look at an example of how the coefficients are converted. If we allocate 23 bits for the fraction part of the coefficients, we get a new set of coefficients like the ones in table 1. The digitized input signal can also be converted in a similar fashion. The division of α is done by shifting the product to the right by $\log_2 \alpha$, or 23 times in this case.

4 Further Reduction of Calculations

Computing wavelet transform using fixed point numbers simplifies the calculations which also reduces the hardware complexity. The next question one can ask is that if the calculation can be further simplified. One approach taken is to omit calculations of the less significant products produced by the coefficients and the data to be transformed. In other words, thresholding the small terms produced by $a_{2k-l}c_l^{j+1}$ and $b_{2k-l}c_l^{j+1}$ before they are actually computed. If the terms $a_{2k-l}c_l^{j+1}$ or $b_{2k-l}c_l^{j+1}$ are so small that they make negligible differences when added to the sum, the multiplication and addition of these terms can be omitted to further reduce the calculation time.

In order to quickly decide whether a multiplication shall take place, a thresholding detector is used for such task. Several steps are involved to generate this detector. First, the magnitude of each possible multiplication operand is quantized into L levels; therefore, there are $L \times L$ combinations for multiplications. The estimated magnitude from the multiplication in each combination is then compared to the threshold value. If the estimated magnitude is less than the threshold value, the corresponding entry in the $L \times L$ table is marked to indicate that no multiplication will be executed for this combination. During the computation of wavelet transform, the thresholding detector will generate a signal from the quantized magnitudes of multiplication operands to control the multiplier.

To implement the quantizer, m most significant bits from the fraction part of the fixed point numbers are selected to distinguish $L = 2^{m-1}$ quantized levels. For example, 4 bits provides 2^4, or 16 quantized levels, and one extra bit is used to quantize the number to the nearest level; thus, 5 bits total are needed to quantize each operand to 16 levels. Since the thresholding of small terms is done while the wavelet transform is computed, this technique is given the name of *thresholding-in-calculations*, or TIC.

5 System architecture

The major components of our system are illustrated in figures 3 through 5. The basic operations of wavelet transform are multiplications and sum accumulations. The data to be transformed is stored in a temporary buffer. The flow of input data through the temporary buffer can be viewed as if a sliding window moves along the input sequence. Thus, the temporary buffer is given the name of SW for Sliding Window. The data from SW multiplies a coefficient from CT (Coefficient Table); the finished product is then added to the accumulated sum. Each time the processor produces a new pair of coefficients (c_k^{j-1} and d_k^{j-1}) from c_l^j, the processor fetches two more input coefficients from memory with the starting address stored in SAR (Starting Address Register). The computed c_k^{j-1} and d_k^{j-1}, temporarily stored in C and D, respectively, will be written to the memory at the address stored in DAR (Destination Address Register). FLR (Filter Length Register) and ISL (Input Signal Length) are user input parameters that store the filter length and input signal length, respectively. The filter size stored in FLR not only controls how many registers will be used in SW, but it is also used to generate the coefficient address, *c_addr*, which is used to locate corresponding wavelet coefficient for each input data. The control unit supplies the necessary control signals to manipulate the data flow and to initiate multiplications/sum-accumulations.

The Multiplier/Sum-accumulator combination performs the multiplications and sum accumulation tasks of the wavelet transform. The multiplications are done by a Booth multiplier that adds all the partial products; the final product is then added to the accumulated sum. Since the principle operations of a Booth multiplier is basically a series of additions, the Booth multiplier can be integrated with the sum accumulation stage so that minimum hardware is used.

6 The Experiment and Results

The accuracy of wavelet decomposition using fixed point numbers and TIC technique is studied for both speech and musical signals. Speech and musical signals are digitized in 16 bits, two's complement integers at the sampling rates of 8 kHz and 22.5 kHz, respectively. Then the digitized signals are converted to proper fixed point format and decomposed using the pyramid scheme.

Figure 6 shows the mean square error (MSE) of a typical speech signal using fixed point numbers for wavelet decomposition. The parameter N is the number of bits used to represent fractions of fixed point numbers. For each set of N, there are two curves; the upper curve is the MSE of the bandpass sub-bands at each level of decomposition, and the lower curve represents the MSE of highpass sub-bands. As N increased by one, the resolution of the fixed point fraction should double; therefore, the MSE should be halved. This relationship is verified in our experiment, and it is shown in figure 6.

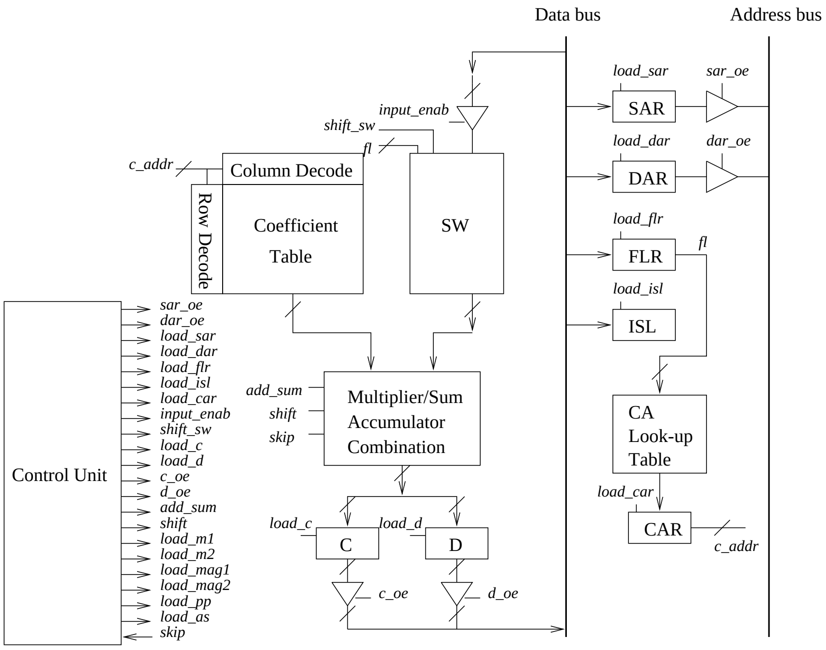

Figure 3: An architectural overview of a wavelet analyzer.

The effect of thresholding non-crucial multiplications is also studied. For the case of the Spline Variant with Dissimilar Length wavelet, we use the smallest case (shortest a and $\tilde{a}$, or h and $\tilde{h}$ as described in [4]) in its family. From our experiment, we have found that a very good approximation of the wavelet transform can be obtained at $N = 20$, $L = 16$, and $th = 2.51e-3$. N is the number of bits used for the fraction of a fixed point number; L is the levels of quantization for the magnitude of multiplication operands; th is the threshold value.

The speedup gained from thresholding non-crucial multiplications is measured by this formula:

$$speedup = \frac{m_{before}}{m_{actual}}.$$

m_{before} is the number of multiplications computed without thresholding, and m_{actual} is the number of actual multiplications with thresholding. With the musical signal applied to the same control parameters, the experiment yields similar results.

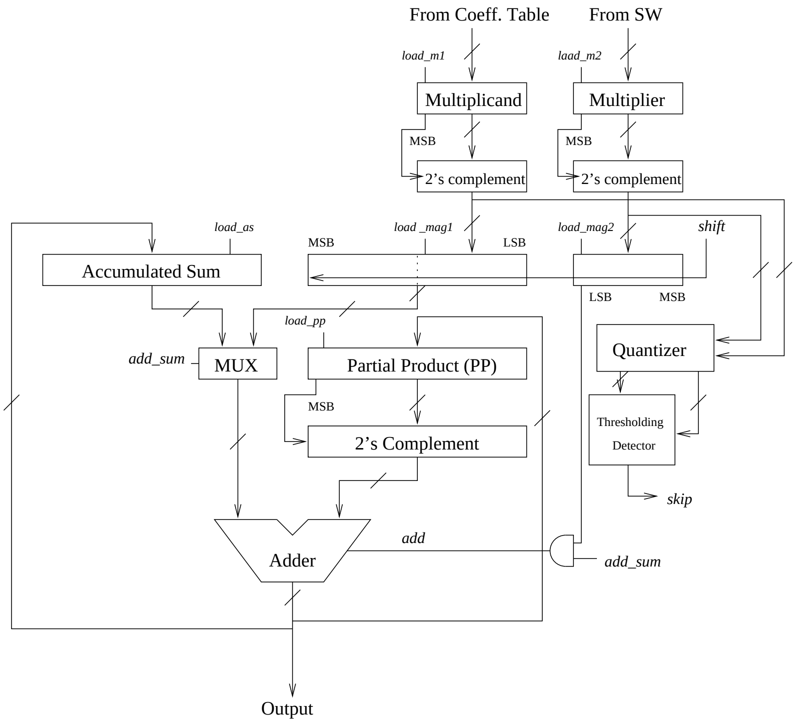

Figure 4: The Booth multiplier/Sum-accumulator combination of the wavelet analyzer.

7 Conclusion

We have presented an architecture that is simple yet effective for fast wavelet transform. The performance gain is achieved by combining the use of fixed point numbers and a fast approximation of the wavelet transform. The simulation results show that the use of fixed point numbers for wavelet transform provides very accurate result compared to single precision floating point numbers. Furthermore, the thresholding-in-calculations (TIC) technique is shown to provide a great performance gain without compromising much accuracy for wavelet decompositions up to five levels. Such capability is far more adequate for most of the signal or image processing applications using wavelet transform.

References

[1] P. Burt and E. Adelson, "The Laplacian Pyramid as a Compact Image Code," *IEEE Trans. on Communications*, Vol. 31, 1983, pp. 482–540.

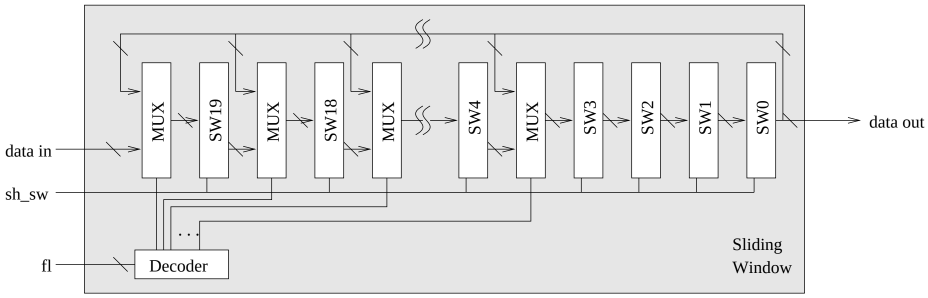

Figure 5: Implementation of the sliding window (SW).

[2] S. Mallat, "A Theory for Multiresolution Signal Decomposition: the Wavelet Representation," *IEEE Trans. on Pattern And Machine Intell.*, Vol. 7, no. 7, 1993, pp. 674–693.

[3] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets", *Comm. on Pure and Applied Math.*, Vol. XLI, 1988, pp. 909–996.

[4] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Images Coding Using Wavelet Transform," *IEEE Trans. on Image Processing*, Vol. 1, No. 2, April 1992, pp. 205–220.

[5] A. Cohen and I. Daubechies, "Biorthogonal Bases of Compactly Supported Wavelets", *Comm. on Pure and Applied Math.*, Vol. XLV, 1992, pp. 485–560.

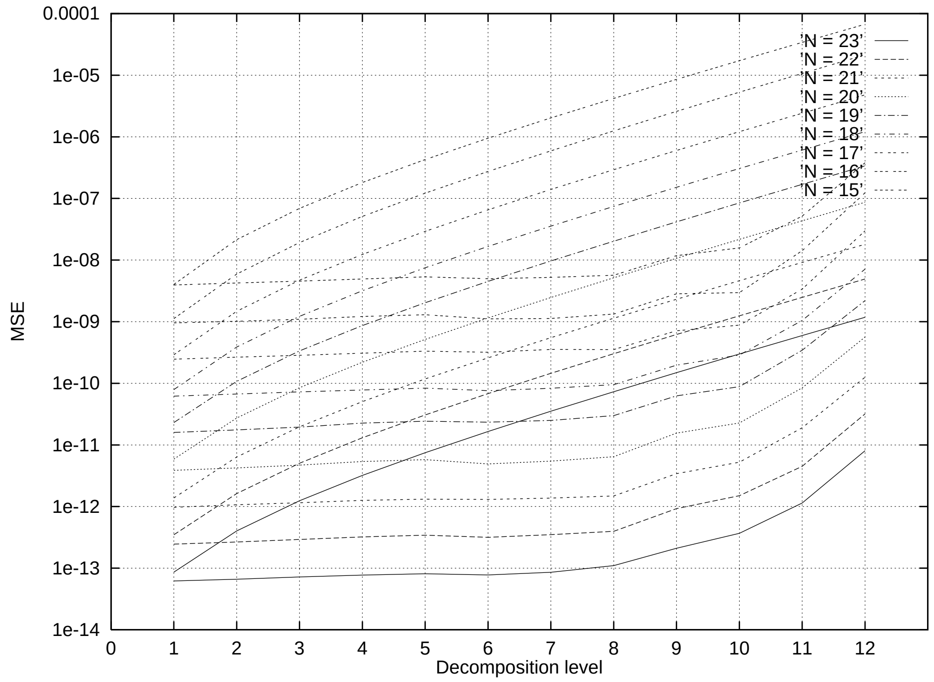

Figure 6: The MSE of wavelet transform using fixed point numbers (Daubechies wavelet with filter length = 4).

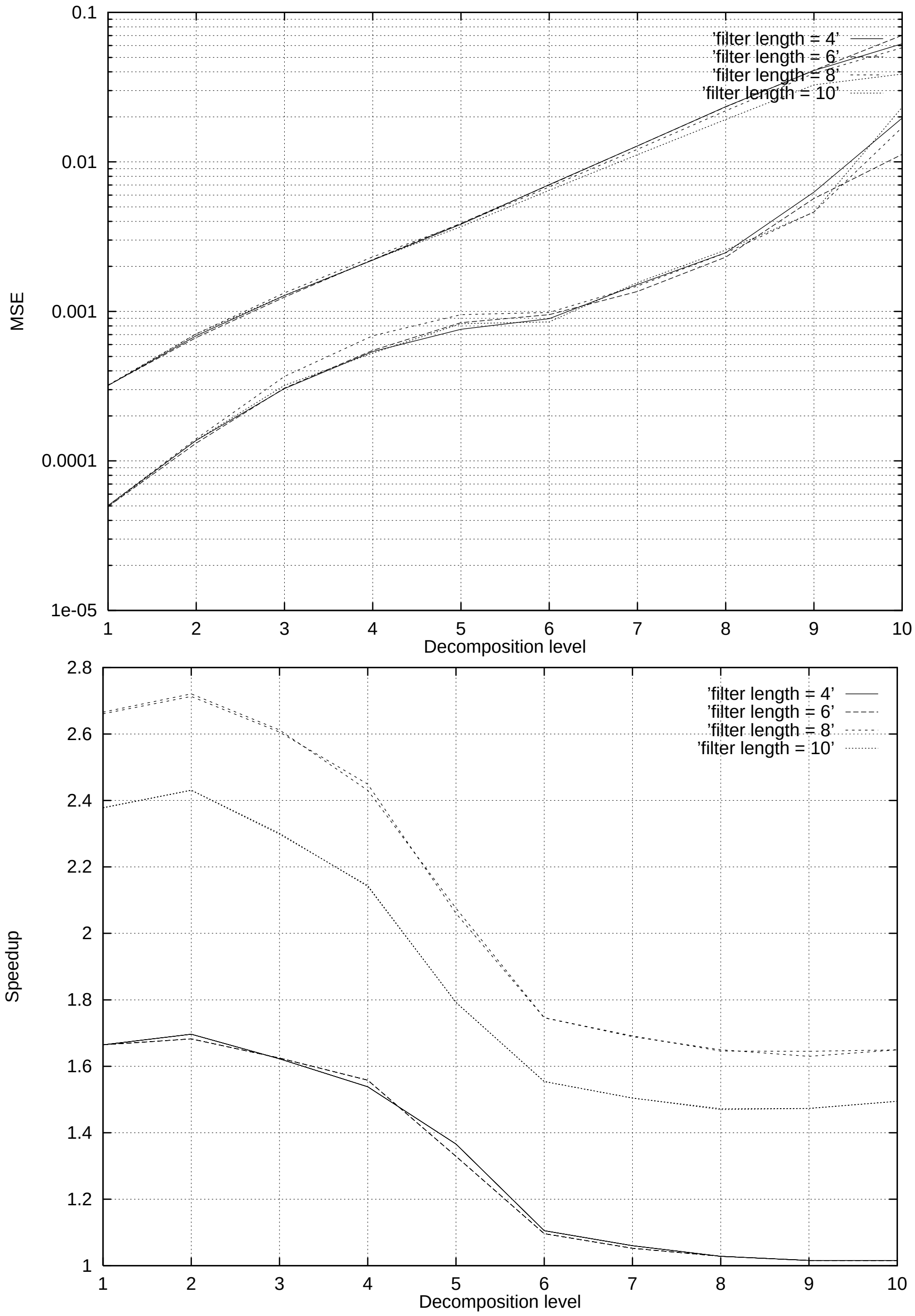

Figure 7: The MSE and speedup using thresholding-in-calculations technique (Daubechies wavelets, $N = 20$, $L = 16$, $th = 2.51e - 3$).

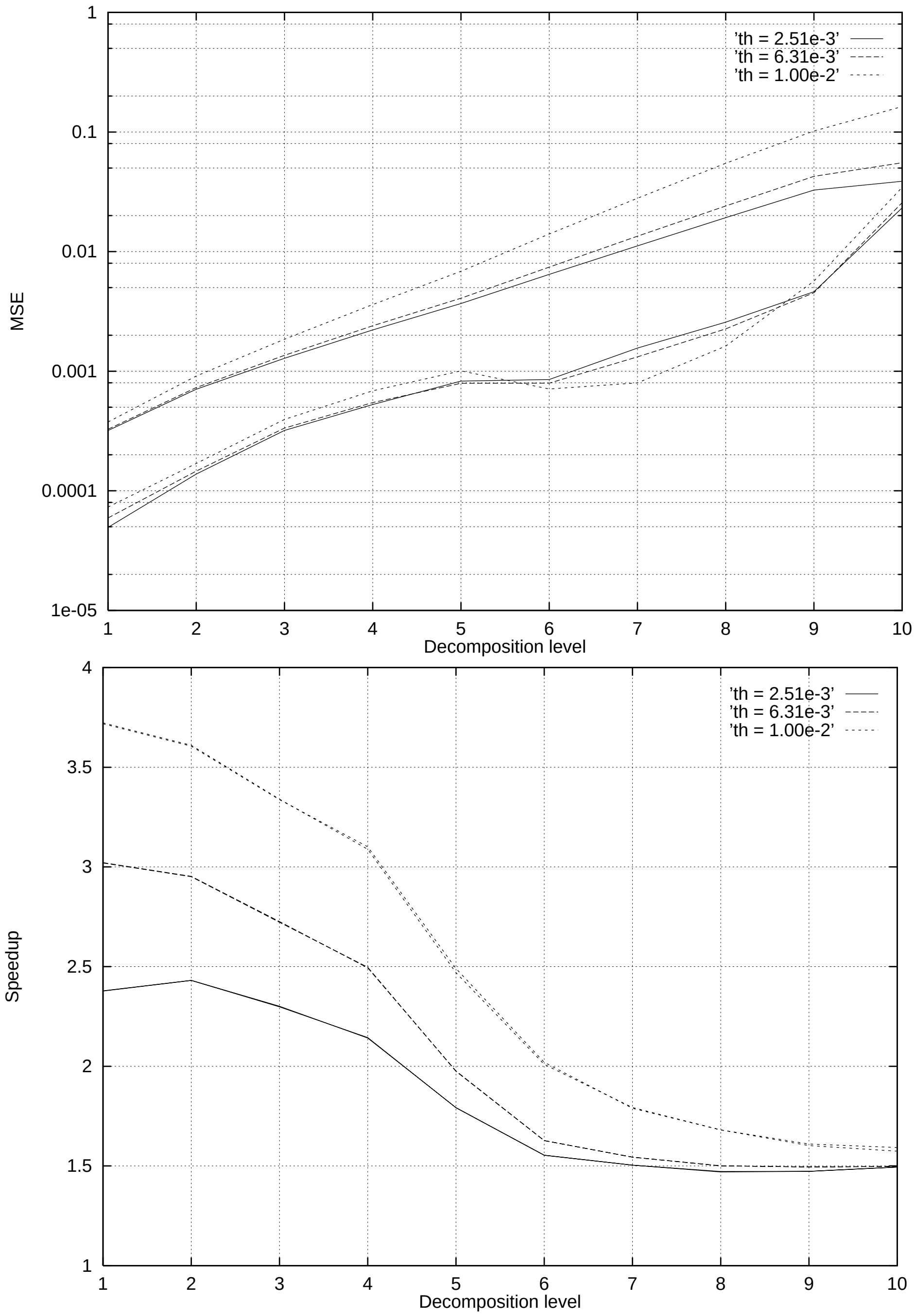

Figure 8: The MSE and speedup using thresholding-in-calculations technique (Daubechies wavelets, $N = 20$, $L = 16$, filter length $= 10$).

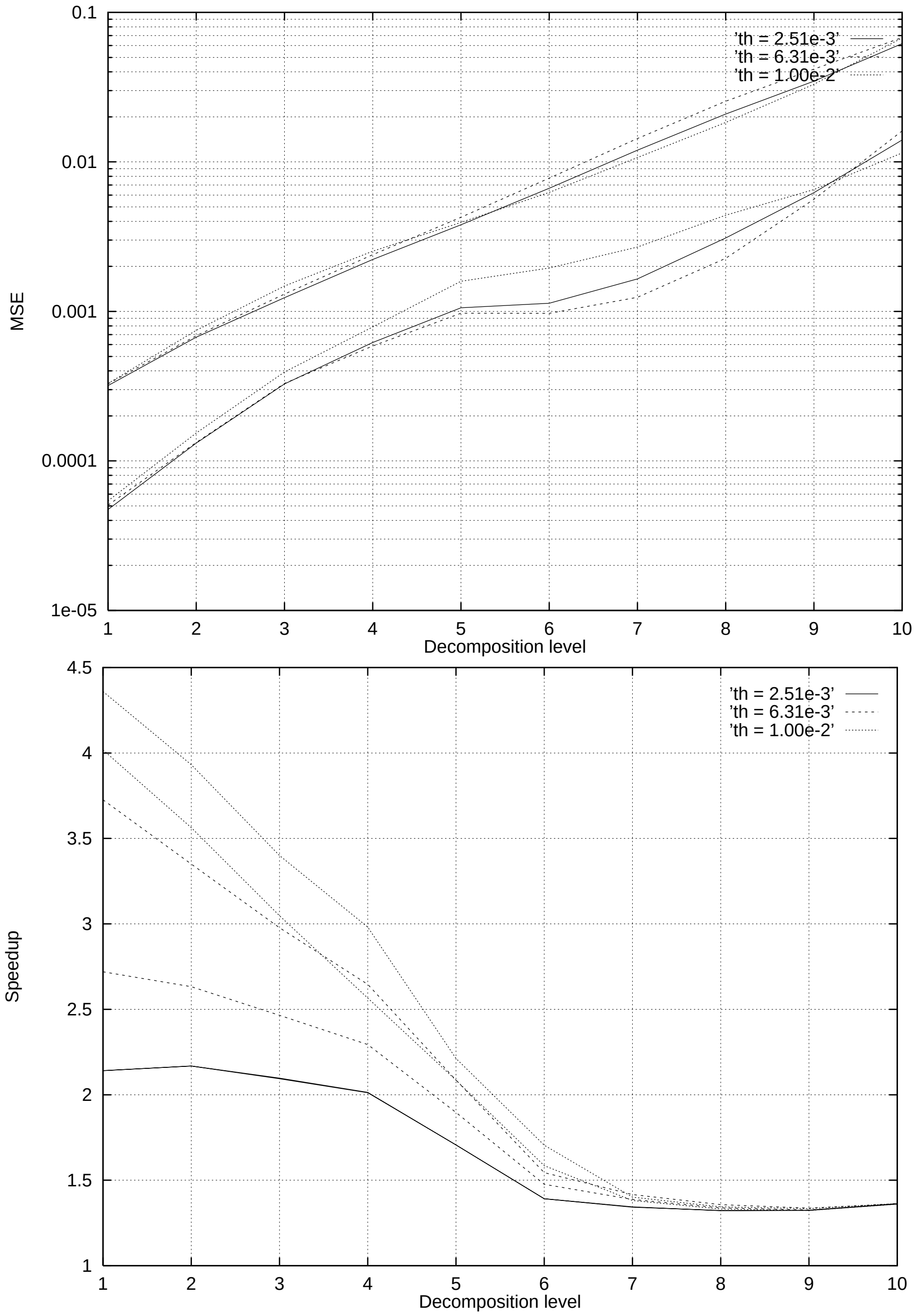

Figure 9: The MSE and speedup using thresholding-in-calculations technique (Spline Variant with Dissimilar Length wavelet, $N = 20$, $L = 16$).