Proceedings of ITC-CSCC 2000, Pusan, Korea

Representation of Wavelet Transform using a Matrix Form and Its Implementation

Masayuki KUROSAKI Kiyoshi NISHIKAWA and Hitoshi KIYA

Department of Electrical Eng., Tokyo Metropolitan Univ.
1-1 Minami-Osawa, Hachioji-shi, Tokyo 192-0397, JAPAN
Phone: +81-426-77-2745, Fax: +81-426-77-2756
E-mail: kurosaki@isys.eei.metro-u.ac.jp

Abstract : Three representations are known to implement the discrete wavelet transform (DWT); i.e., direct, lifting and matrix forms. In these representations, direct and lifting forms are well known so far. This paper derives the matrix form of the DWT from the direct form. Then, we implement these three representations on a programmable digital signal processor (in the following, DSP processor) and compare them in terms of the number of calculations and instruction cycles. As a result, we confirm that the lifting form has the lowest number of calculations and cycles, and the matrix form has an effective decrease in the number of cycles than other representations on the DSP processor.

1. Introduction

The discrete wavelet transform (DWT) is considered as a superior method than the discrete cosine transform (DCT) for image coding. For example, the JPEG2000, a new still image coding standard, is based on the DWT[1]. Several representations can be used to implement the DWT. Among them, three representations are known, namely, direct, lifting and matrix forms.

Usually, the DWT is implemented using two-channel filter banks in the tree structure[2]. We call this structure the direct form in this report. This structure is made up from several stages: a stage is composed of a two-channel filter bank. The same process is repeated on each stages. One of the advantages of this structure is that it does not require huge amount of memory to store filter coefficients. On the other hand, the data path of the form would be longer as the number of stage increases because the output of previous stage is used in the calculations at next stage. Thus it could not operate at high speed because of the long data-flow.

The lifting form is a special case of the direct form[3] and is adopted for the implementation of the DWT in the JPEG2000. As the filter order of this form is shorter than that of the direct form, the amount of memory required to store filter coefficients is less than that of the direct form. Using this form, we can reduce the number of calculations than other forms. However, the lifting form inherits disaverages from the direct form.

The representation using matrix form is known as an alternative method to implement the DWT[4]. It enables us to calculate the wavelet coefficients directly and, hence, it is unnecessary to wait for data from the previous stage. The matrix form can process in parallel, and therefore, it can operate more efficiently than the other two forms if its parallelism can be used at implementation time. On the other hand, this form requires the large number of calculations because of the computational redundancy.

In this report, we compare these three forms. First we derive the matrix form of the DWT from the direct form to confirm its parallelism. Then, we implement three representations on a programmable digital signal processor (in the following, DSP processor) and compare them from two points of view: the number of calculations and instruction cycles on a DSP processor.

As a result, we confirm that the lifting form can be calculated with the smallest number of calculations and can be implemented with the smallest instruction cycles. Then, we show that, when the number of stages is one or two, the matrix form enables us to decrease the number of instruction cycles than the other forms, if we can use its parallel nature.

2. Direct form and lifting form of the DWT

In this section, we review the procedure of the DWT. First we show the direct form of the DWT and then the lifting form. Note that we explain one dimensional DWT for simplicity.

2.1 Direct form of the DWT

Usually, the DWT is calculated using tree structure of two-channel filter banks and its block diagram is shown in Figure 1. In this figure, the down arrow shows a down sampler that decimates the input signal. $H_L(z)$ and $H_H(z)$ are a low-pass and a high-pass filters. The input signal $X(z)$ are filtered by $H_L(z)$ and $H_H(z)$ independently, then they are decimated. As a result, data sequence of L_band_1 and H_band_1, called the wavelet coefficients, are obtained as the outputs of the stage 1. When the number of stages is 2, as Figure 1 shows that wavelet coefficients L_band_2, H_band_2 are calculated using L_band_1. Repeating this procedure, wavelet coefficients of the ar-

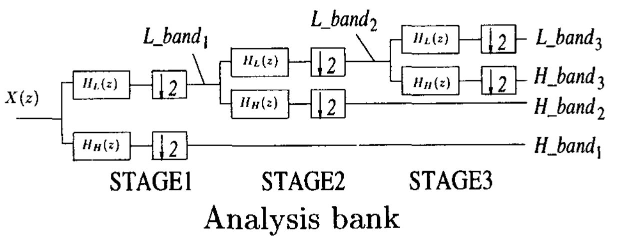

Figure 1: Direct form based on two-channel filter banks

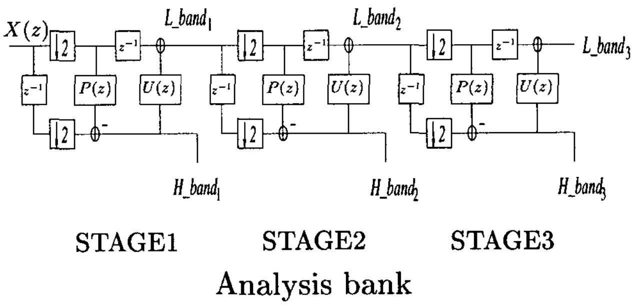

Figure 2: Lifting form of the DWT

bitrary number of stages can be obtained. We call this way of implementation the direct form.

We can implement the direct form of the DWT with relatively small amount of memorys to store filter coefficients. However, as the number of stages increases its data-flow become longer. Consequently, the latency of the system will be long, and this may arise a problem in some applications.

2.2 Lifting form of the DWT

Next, we explain the lifting form of the DWT. The JPEG2000, a next generation coding standard for still images, uses the lifting form as one of the implementation methods of the DWT.

This form is a special case of the direct form. It satisfies the perfect reconstruction property with filters of short order. In addition, this property is not affected by quantization.

Figure 2 shows lifting form of three stage DWT. In this figure, z^{-1} shows a delay; $P(z)$ and $U(z)$ show filters derived from $H_L(z)$ and $H_H(z)$ respectively. As the order of filters of this form are short, the amount of memory required to store filter coefficients can be further reduced. Besides, we can reduce the amount of calculations required than other forms because of shorter filters. However, the lifting form suffers from the same disaverages as the direct form.

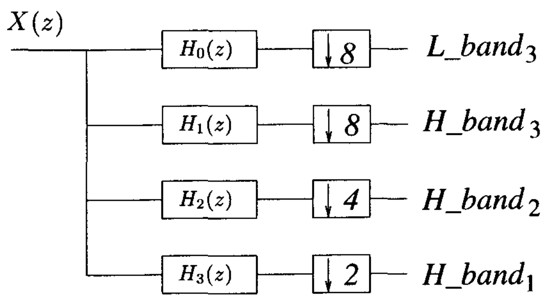

Figure 3: Representation using parallel form

3. Matrix form of the DWT

Here, we derive the matrix form of the DWT from direct form of Figure 1 to confirm its parallelism. First we show a representation derived from Figure 1 using filter banks of the DWT and then we show the matrix form.

3.1 Representation using filter bank

Using a property of multi-rate signal processing[4], we can derive the representation of Figure 3 from that of Figure 1.

In this figure, each filter is expressed as follows:

$$H_0(z) = H_L(z^4)H_L(z^2)H_L(z) \quad (1)$$
$$H_1(z) = H_H(z^4)H_L(z^2)H_L(z) \quad (2)$$
$$H_2(z) = H_H(z^2)H_L(z) \quad (3)$$
$$H_3(z) = H_H(z) \quad (4)$$

where $H(z^N)$ shows to insert $N-1$ zeros between each sample of the impulse response of $H(z)$.

It is possible to derive Figure 4 from Figure 3, using the serial/parallel transform and a property of multirate processing[4].

3.2 Matrix form of the DWT

We review the case of the CDF(5,3)[1] filter as an example. The impulse response is shown as follows:

$$H_L(z) = \frac{1}{8}(-1+2z^{-1}+6z^{-2}+2z^{-3}-z^{-4}) \quad (5)$$
$$H_H(z) = \frac{1}{2}(-1+2z^{-1}-z^{-2}) \quad (6)$$

The matrix of the one stage DWT is derived as follows:

$$P = \frac{1}{8}\begin{pmatrix} -1 & 2 & 6 & 2 & -1 \\ -4 & 8 & -4 & 0 & 0 \end{pmatrix} \quad (7)$$

Similarly, the matrix of two stage DWT is shown as follows:

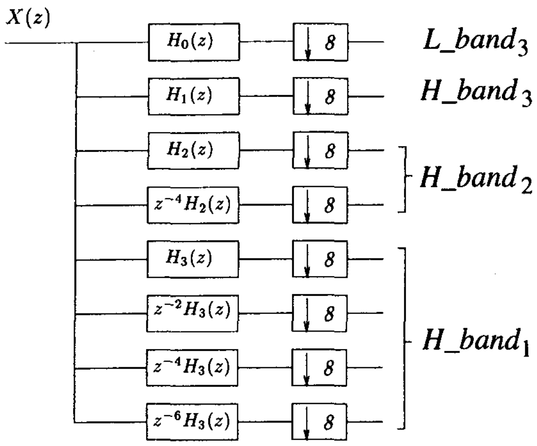

Figure 4: Representation using uniform filter banks of the DWT

$$P = \frac{1}{64} \times \begin{pmatrix} 1 & -2 & -8 & 2 & 7 & 16 & 32 & 16 & 7 & 2 & -8 & -2 & 1 \\ 4 & -8 & -32 & 8 & 56 & 8 & -32 & -8 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & -32 & 64 & -32 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -32 & 64 & -32 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (8)$$

where P shows the transform matrix from the input data to wavelet coefficients.

Using this matrix, wavelet coefficients are calculated directly from input data and we call this method for calculating wavelet coefficients as the matrix form. The advantage of the matrix form is that it is unnecessary to wait for the output data of the previous stage. This feature contributes to reduce the latency of the system. Moreover, it is possible to implement in parallel on the hardware.

4. Comparison

We simulated the three representations on a DSP processor and compared them in the two points of views: the amount of calculations and the instruction cycles.

A. Comparison of the amount of calculations
We estimated the amount of calculations required to implement the three forms. We assume computational complexity of both one addition and one multiplication are same because the DSP processor we used for implementation requires the equal number of cycles for those operations.

Table 1 and Figure 5 show the result of estimating the amount of calculations for 100 input data. We can confirm that the lifting form requires the smallest amount of calculations and the matrix form the largest.

Table 1: Amount of calculatons

Stages	1	2	3	4
Direct form	723	975	1119	1209
Lifting form	400	600	700	750
Matrix form	728	1456	2432	4000

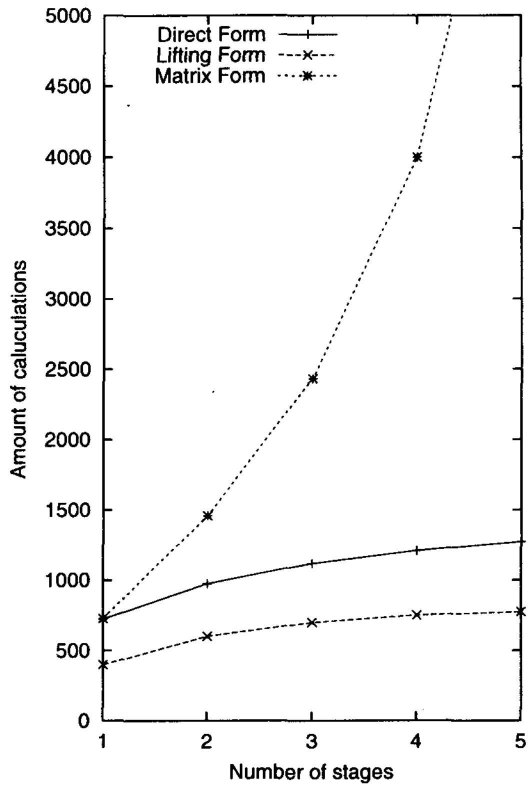

Figure 5: Amount of calculations

B. Comparison of the number of instruction cycles
Next, we implemented three forms of the DWT on a DSP processor. The target DSP processor was the TMS320C6701[5] produced by Texas Instruments. This DSP processor has two independent processing units. Each units can connect independent data path. We used the Code Composer Studio as a simulation tool and instruction cycle time was set as 60 ns. Other conditions of the simulation are shown as follows:

- We used the CDF(5,3) filter.
- Matrix coefficients were prepared and stored in the memory.
- Parallel instructions were used to process the data on the DSP processor efficiently.

Table 2 and Figure 6 show the number of instruction cycles required to process 512 input data on the target DSP processor. We can confirm that the lifting

Table 2: Number of instruction cycles

Stages	1	2	3	4
Direct form	131025	197548	231559	249314
Lifting form	55845	84385	99383	107501
Matrix form	96105	155100	240584	353328

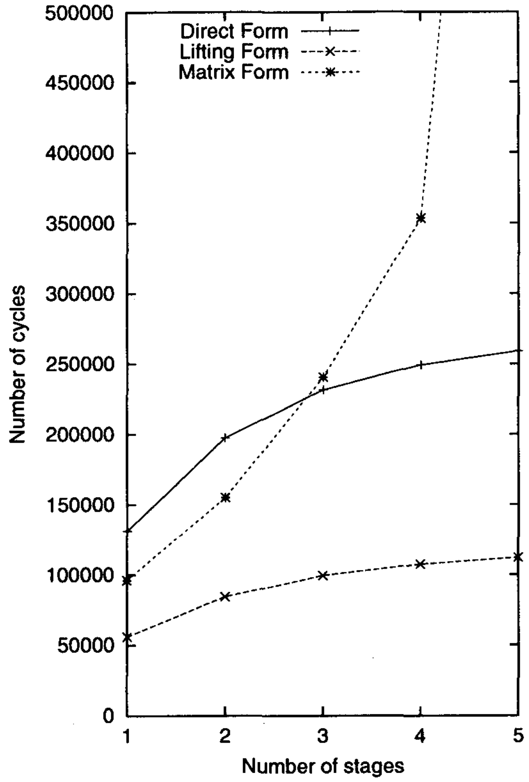

Figure 6: Number of instruction cycles

form can be implemented with the smallest instruction cycles.

By comparing Figure 5 and Figure 6, we can confirm that when the number of stages is one and two the number of instruction cycles of the matrix form is smaller than that of the direct form. This is because the number of instruction cycles is reduced effectively using the parallelism of the matrix form. Followings are confirmed from the results.

- The lifting form requires the smallest amount of calculations and can be implemented with the smallest instruction cycles.

- When multiple processing units are available, the number of instruction cycles are not in proportional to the amount of calculations.

The effect of using the matrix form is limited on processors where available number of operation units are small like a DSP processor. When enough amount of processing units are available, however, we can reduce instruction cycles more efficiently by using its parallelism.

5. Conclusion

In this report, we compared three forms for implementing the DWT: direct, lifting and matrix forms. First, we derived the matrix form of the DWT from the direct form for confirmation of its parallelism. Then, we implemented three representations on a DSP processor. As a result, we confirm that the lifting form can be calculated with the smallest amount of calculations and can be implemented with the small est instruction cycles. Then, we showed that, when the number of stages is one or two, the matrix form enables us to decrease the number of instruction cycles than the other forms, if we can use its parallelism. We think that when enough amount of processing units are available, we can reduce instruction cycles further by using the matrix form.

References

[1] ISO/IEC CD15444-1, JPEG2000 Committee Draft Version 1.0.9 Dec. 1999.

[2] H.Kiya, "Multirate signal processing," shoko-do, 1995. (in Japanese).

[3] I.Daubechies and W.Sweldens, "Factoring wavelet transforms into lifting steps," J. Fourier Anal. Appl., vol.4, pp.247–269, 1998.

[4] H.Kiya, "Wavelet Transform and its Application," ISCIE, system, control and information, vol. 41, No. 8, pp.330–337, Aug. 1997. (in Japanese).

[5] Texas Instruments, "TMS320C6000 Peripherals Reference Guide," SPRU190C, Apr. 1999.