Novel Pipelined DWT Architecture for Dual-Line Scan

Jinook Song and In-Cheol Park
Dept. of Electrical Engineering
KAIST
Daejeon, Korea
Email: jinook.song@gmail.com, icpark@ee.kaist.ac.kr

Abstract—A new discrete wavelet transform (DWT) architecture is proposed in this paper to realize a memory-efficient 2D DWT unit. The proposed DWT architecture alternately processes two lines to remove the transpose buffer whose size is proportional to the image row size. As a result, the hardware complexity of 2D DWT is significantly reduced. To maintain the same critical path delay as that of the previous pipelined DWT, serially concatenated additions are optimized by changing computation topology and applying arithmetic optimization.

I. INTRODUCTION

Considerable attention has been paid to the development of efficient architectures for 2-dimensional discrete wavelet transform (2D-DWT) since the JPEG2000 was revealed as a new still-image compression standard [1]. The DWT is composed of high-pass and low-pass filters. In the JPEG 2000 standard, 5/3 and 9/7 filters are adopted. The filters are conventionally designed based on the convolution-based architecture. Though some previous works have proposed convolution-based DWT architectures based on the convolution filters [2], they require many operators and complex operations. To reduce the hardware complexity, a systematic structure called lifting-based DWT was proposed in [3], which separates the convolution into predict and update steps. The 5/3 filter consists of one predict and one update step, while the 9/7 filter can be performed by applying the predict and update steps twice. As the lifting-based architecture can be computed with regular equations and a less number of operators, it has a merit in hardware implementation.

The 2D-DWT is generally realized by applying 1D-DWT two times as the row-wise processing and the column-wise processing are separable. In other words, all the rows are processed first, and then the results are processed in the column-wise direction. Accordingly, the simplest implementation of the 2D-DWT is directly performing 1D-DWT in the row direction and then the column direction. However, this direct architecture necessitates a buffer called a transpose buffer to store all the results of the row-wise DWT. The buffer size is proportional to O(N^2) if the image size is N by N. To reduce the transpose buffer size, the speed difference of being filled and being consumed was utilized in [4]. Furthermore, the dual-line scan architectures have been proposed to reduce the transpose buffer size to O(N) [5]-[7]. In [5], two lines are simultaneously processed by scaling up the number of row-wise processors (RPs) and column-wise processors (CPs) double [5], while in [6] and [7], two lines are processed with a single RP by alternating the processing order every cycle. The latter approach called tile-based processing is most efficient because it can reduce the number of required processors as well as the transpose buffer size. However, alternate processing of two lines is against the nature of the lifting-based DWT, so it requires complex control and additional buffers in the RP and CP. As all the 1-D-DWT architectures presented in previous literatures are dedicated to single-line processing, complex control and additional buffers are indispensable if the 2D-DWT is designed based on the 1D-DWT optimized for single-line processing.

This paper proposes a novel pipelined DWT architecture developed for scanning two lines alternately. Accordingly, the proposed DWT architecture is suitable for the tile-based processing. The rest of this paper is organized as follows. The proposed DWT architecture is described in Section II. The proposed DWT architecture is described in Section III, and is compared with other works in Section IV. Finally, concluding remarks are made in Section V.

II. LIFTING-BASED DISCRETE WAVELET TRANSFORM ARCHITECTURE

Fig. 1(a) shows the lifting-based DWT structure for 5/3 and 9/7 filters adopted in the JPEG2000 standard, and their coefficients are listed in the figure. The 5/3 filter consists of one predict step and one update step, while the 9/7 filter can be performed by applying the predict and update steps twice and then scaling the outputs. As the 5/3 filter is a reversible transform, it is composed of only adders and shifters, and has a short delay. However, the 9/7 filter consists of several adders and multipliers since it is an irreversible transform. From now on, we will focus on the 9/7 filter, for the 5/3 filter is much simpler and can be regarded as a subset of the 9/7 filter. As shown in Fig. 1(a), the input data set involved in generating a filter output is overlapped with those involved in the next filter output. It is a common technique to incrementally calculate

This work was supported by KIPO and IDEC
the filter outputs by considering two adjacent input data. Accordingly, two input data generate one low-pass filter output and one high-pass filter output. Therefore, the critical path delay of the 9/7 filter comes from 5 multiplications and four 3-input additions that are serially concatenated. The delay is 5Tm+8Ta if implemented straightforwardly, where Tm and Ta are the delays of multiplication and 2-input addition, respectively. Recently, a new version of the lifting-based DWT was proposed in [9], which is called the flipping-based architecture. The computational part of the flipping-based architecture, called a slice in this paper which is indicated with dotted lines in Fig. 1(b), decreases the number of multiplications and the critical path delay. As shown in Fig. 1(b), the multiplication is changed to the reciprocal multiplication and the scale factors are also changed accordingly. The flipping-based architecture reduces the critical path delay to Tm+5Ta because the multiplications can be executed in parallel. [9].

Since the slice still has a long critical path, several researches have applied the pipelining technique to reduce the critical path further by placing registers in the DWT slice [8], [9]. Although the pipelining reduces the critical path delay, it increases the number of registers and the control complexity. The pipelined flipping-based architecture is shown Fig. 1(c), where the slice is divided into five stages to reduce the critical path delay to Tm. The pipelined slice produces not only DWT outputs but also intermediate values denoted by the gray circles in Fig. 1(c). Once a slice is completed for an input pair of a row, the intermediate values are stored in the intermediate buffers for the processing of the next input pair belonging to the row. Thus, the conventional pipelined architectures are appropriate for single-line processing.

A. 2-Dimensional Discrete Wavelet Transform

The typical 2D-DWT architecture is shown in Fig. 2, which contains a RP, a CP, a transpose buffer and intermediate buffers. The transpose buffer size is proportional to the latency between the RP and CP. Since the previous 2D-DWT architecture is developed for the line-based processing, it is hard to remove the buffer. Note that a slice of the DWT can be started if at least a pair of two input data is provided. Therefore, the CP can start when the RP completes the row data related to the column-wise processing. Since the CP needs two column-wise data, the CP should wait until the RP finishes the first two data in the second row, as shown in Fig. 3. During the latency, the results of the RP must be stored into the transpose buffer. Detailed analysis reveals that the minimum size of the transpose buffer is 1.5N, where N represents the row size.

To reduce the latency between the RP and CP, we can employ two RPs each of which is responsible for one of two adjacent rows. In this processing approach that performs two rows concurrently, the latency between the RP and CP can be reduced to a constant that is independent of N, which means that the transpose buffer can be removed. However, it has the number of RPs and CPs increased by a factor of two. Another way of reducing the transpose buffer is to scan two lines alternately as shown in Fig. 4, where two data are read from the first row and then from the second row alternately. In this scheme, the CP can start as soon as the two data of both rows are processed. This approach called tile-based DWT processing can remove the transpose buffer without increasing the number of RPs and CPs. The tile-based processing renders an image being segmented into a set of rectangular tiles whose size is 2-by-2 pixels as shown in Fig. 4. For each tile, the RP performs the partial rows, and then the CP starts processing.
for the partial column data resulting from the RP as shown in Fig. 5. If the RP finishes the processing of a tile, it will move to the next adjacent tile. Since the latency between the RP and CP is proportional to the tile size, only a small sized buffer of 2-by-2 is sufficient, removing the transpose buffer effectively. Therefore, the tile-based processing is much efficient in terms of the hardware resource than scanning two rows concurrently [7]. However, it requires a special pipelined DWT architecture that can process two rows alternately. Such a DWT architecture is not reported yet. If the previous pipelined DWT, which is optimized for the line-based processing, is used to realize the tile-based processing, additional intermediate buffers should be included because the data come from two rows not from a single row. In fact, we have to double the pipeline registers, as the input data are not belonging to a row.

III. PROPOSED DWT ARCHITECTURE

Let us suppose that a pair of two input data contained in an even row is provided at every even cycle, and a pair of two input data contained in the next odd row is alternately provided at every odd cycle. As the intermediate data resulting from the even row are used for the next data pair coming from the even row, the current data are dependent only the data provided two cycles earlier, as depicted in Fig. 6(a). This relation induces a limitation on the number of pipeline stages. The desired pipelined DWT must be designed with two stages, as shown in Fig. 6(b), where the results of the second stage are fed back to the first stage. Since the data pairs separated by two cycles come from the same row, the feedback results are related to the previous data pair of the same row. Thus, if we can make such a two-stage pipelined DWT architecture, it is possible to process two rows alternately without appending additional buffers. The previous pipelined DWT architectures have 5 stages to reduce the critical path delay, thereby it is not suitable for the tile-based processing.

The proposed pipelined DWT is shown in Fig. 7(a). Compared to the previous pipelined DWT shown in Fig. 3, the proposed DWT is sliced in different direction. In the first stage, all the multiplications are performed, and they are summed in the second stage. The first stage has no difficulty in implementation, as the multiplications can be performed in parallel. The second stage may have long delay as several adders are serially connected. If the second stage is implemented straightforwardly, the serial path makes the cycle time long, losing lots of throughput. An efficient structure of the second stage is shown in Fig. 7(b), where the three input adder is changed to two input adders so as to reduce the serial path. To further reduce the remaining serial path, arithmetic optimization is applied by employing the carry-save adder (CSA). The CSA has a delay of the single-bit full adder and can compress three inputs to two outputs. For example, the delays of the CSA and a 16-bit adder are 0.13ns and 0.9ns in a 0.18 um CMOS technology. As a matter of fact, the CSA is functionally equivalent to the full adder. Looking at the serial path, we can find that its objective is to sum five values. Therefore the serial path can be optimized by using the CSA, as depicted in Fig. 7(c). The resulting second stage has a delay lower than the multiplication delay. In the above technology, the second stage has a delay of about 1.5ns, because one addition and four CSAs are concatenated, and the multiplier in the first stage has a delay of 1.9ns. Therefore, the critical delay of the proposed pipelined architecture is Tm, which is the same as that of the conventional architecture [9]. Note that the proposed DWT has only two stages, while the previous one has 5 stages. The proposed DWT architecture is designed for scanning dual lines alternately.

IV. COMPARISON

Table I compares the hardware complexity of the RP and the transpose buffer size needed to realize a 2D-DWT hardware unit. In the table, the 2D-DWT unit based on the proposed pipelined DWT is compared with three different implementations. The direct implementation is designed for the lifting-based DWT, and is fully pipelined to minimize the critical path delay. Huang’s implementation employs the flipping-based architecture [9]. This architecture is effective in reducing registers and achieving a high throughput. Wu’s implementation merges the predict and update step of the lifting-based DWT to achieve a throughput of 1 input/output.
Though this architecture reduces the number of operators, it needs many registers and the throughput is reduced by half. All the previous DWT architectures process line by line, thus it is not possible to remove the transpose buffer whose size is proportional to at least $O(N)$. The proposed architecture does not require the transpose buffer proportional to $O(N)$, instead it needs only a small constant-sized buffer for 2D-DWT. In addition, we can see in the table that the proposed one increases the number of the registers slightly compared to Huang's architecture, but it reduces the number of operators. As three CSAs are usually equivalent to one adder in terms of hardware complexity, the total adder complexity is reduced while maintaining the same number of multipliers.

### V. CONCLUSION

This paper has presented a new DWT architecture developed for dual-line scan. We have derived the property required to process two lines alternately, and then proposed the two-stage pipelined DWT. To maintain the same critical path as the previous pipelined DWT, the second stage consisting of serially concatenated additions is optimized by changing the computation topology and applying arithmetic optimization. The proposed DWT solves the long critical path of the serial additions by employing the CSA. The resulting critical path is not increased even compared to the previous state-of-the-art architectures developed for single-line processing. As the proposed DWT removes the transpose buffer whose size is proportional to the row size of the image, it reduces the hardware complexity of 2D DWT significantly.

### REFERENCES


### TABLE I. COMPARISON OF VARIOUS DWT ARCHITECTURES OF 9/7 FILTER

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Register</th>
<th>Multipliers</th>
<th>Adders</th>
<th>Critical Path</th>
<th>Processing throughput (per cycle)</th>
<th>Transpose Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct Implementation</td>
<td>32</td>
<td>4</td>
<td>8</td>
<td>$T_m$</td>
<td>2 input/output</td>
<td>$N$</td>
</tr>
<tr>
<td>Huang [9]</td>
<td>11</td>
<td>4</td>
<td>8</td>
<td>$T_m$</td>
<td>2 input/output</td>
<td>1.5N</td>
</tr>
<tr>
<td>Wu [8]</td>
<td>20</td>
<td>2</td>
<td>4</td>
<td>$T_m$</td>
<td>1 input/output</td>
<td>1.5N</td>
</tr>
<tr>
<td>Proposed Pipelined</td>
<td>13</td>
<td>4</td>
<td>$4+7CSA≈6.3$</td>
<td>$T_m$</td>
<td>2 input/output</td>
<td>4</td>
</tr>
</tbody>
</table>