Ahmed Khorsheed Al-Sulaifanie
Ph.D., Assistant Professor, Computer Engineering Department
College of Engineering, University of Mosul

Abstract

In this paper, the design of new real time integer to integer lifting based wavelet transform IWT architecture is focused. An efficient design method is proposed to construct an integrated programmable VLSI architecture that can operate as forward or backward IWT in pipeline fashion. The layout VLSI integrated structure is simple, modular, and cascadable for computation of wavelet transform based on 5/3 biorthogonal filters. The architecture is optimal with respect to both area and time and independent of the size of the input signal without necessitate to memory. The lifting steps adapted to be causal and the proposed architecture is suitable to be used in the real time processing applications. The critical path of the architecture is equal to critical path of one lifting step. The numerical precision has been established using simulink model. Experimental tests have been made with 8-bit signed two's complement integer numbers. Based on the experimental result observations, the data path width of proposed architecture is fixed at 10 bits.

Keywords: Lifting scheme, VLSI architecture, wavelet transform.
الخلاصة:
في هذا البحث اقترحت معمارية جديدة للتحويل الموجي في الزمن الحقيقي مبنية على مخطط الرفع لتحويل العدد الصحيح إلى العدد الصحيح (IWT). استخدمت طريقة تصميم كفاءة: تكنولوجيا (VLSI) متكاملة يمكنها أن تعتمد على معمارى الموجى. المعمارى المصممة هي سهلة، سببية، عملية للمحول الموجي، متكاملة على مرشح/ثنائي التعابير وهي مثالية من ناحية سهولة التطبيق في التطبيقات، بسهولة في الوقت الحقيقي وتقنية خطا النسبي، لا تعتمد على طول الإشارة الداخلية أي من دون الحاجة إلى ذاكرة خزين. تم من خلال اختبارات إعداد عملية حساسية دو طول ( ) وبصيغة المتممة لـ ( ) أظهرت الاختبارات التجريبية عرض طريقة بيانات معمارية يمكن ( ) .

Received 30 October 2005 Accepted 26 January 2006
1. Introduction

The lifting scheme is an elementary method to obtain truly loss-less non-linear integer-to-integer wavelet transforms with following properties (1) fast implementation because lifting principle based on the optimal similarities between high and low pass filters, which in consequence reduce the computation complexity (2) in-place calculation by regularly replacing the original signal with its transform, an auxiliary memory can be kept away and the hardware implementation can be compact (3) the backward transform can be realized using the inverse elementary operations of the forward one, taken in reversed order[1], [2], [3], [4].

In lifting scheme it is possible to maintain integer data after filtering operation if the input data are integer. This can be developed very simply by achieving rounding in each lifting step. In consequence the linear lifting steps are replaced by their non-linear approximation. The reversible integer-to-integer wavelet transform is called IWT. It is important to note that the filter coefficient not necessary to be integer for IWT [1].

A lot of literatures have been published concerning traditional convolution design for DWT implementations. The architecture can be broadly classified in the range from SIMD arrays to folded architectures such as systolic arrays and parallel filters. The folded architectures, implement online versions of the recursive pyramid algorithm RPA [6]. These architectures support single chip implementations in VLSI and are optimal with respect to both area and time under the word serial model [7][8][9].

Recently most of the works are done on the newly proposed lifting scheme. A numerous papers are published for efficient VLSI architectures of 1-D and 2-D lifting based DWT [10]-[15]. In [10] a lifting scheme base architecture is proposed that
perform the forward and inverse DWT for a set of filters anticipated in JPEG2000. Efficient lifting scheme VLSI architecture is proposed in [11] by flipping conventional lifting structures for improving and minimizing the critical path and memory requirement. In [12], a systematic design method for efficient pipeline VLSI architectures of lifting scheme is proposed, which includes specific lifting factorization, dependence graph formation, and systolic arrays mapping. A VLSI architecture is proposed in [13] for the IWT implementation, capable of achieving very high frame rates with moderate gate complexity. DSP-type architecture for IWT are presented in [14] dealing with optimal factorization and finite precision effects.

Although the lifting scheme has been widely studied in the literature, most of them consider non-causal systems where the whole signal is buffered.

In this paper, we address new real time integer to integer lifting based wavelet transform IWT architecture. An efficient design method is proposed to construct an integrated programmable VLSI architecture that can be operating as forward or backward IWT. The architecture is casual and no memory is needed for buffering.

The paper is organized as follows. In Section 2, the theory of lifting scheme factorization polyphase matrix is reviewed. The design issues of real time forward and backward IWT are given in section 3. In section 4 the fixed-point lifting structure with numerical precision analysis is described. The design procedures for integrated VLSI architecture are provided in section 5. Finally, in section 6, conclusions are drawn.

2. Lifting Scheme

The polyphase representation of a discrete-time FIR filter h[n] can be decomposed in z-transform domain into two parallel filters as

\[ H(z) = H_e(z^2) + z^{-1}H_o(z^2) \]
where $H_e(z)$ encloses the even filter coefficients and $H_o(z)$ encloses the odd filter coefficients of the FIR filter $H(z)$. The $z$-transform of the decomposed filters can be expressed as

$$H_e(z) = \sum_{k} h[2k] z^{-k}, \quad H_o(z) = \sum_{n} h[2k+1] z^{-k},$$

where $h[2k]$ contains even coefficients and $h[2k+1]$ contains the odd coefficients.

The general block scheme of the DWT is analogous to classical subband system as shown in figure 1. If the sets of filters $\{H_a(z), G_a(z)\}$ and $\{H_s(z), G_s(z)\}$ represent analysis and synthesis lowpass and highpass filters respectively. The corresponding polyphase matrices are defined as [4]

$$P_a(z) = \begin{bmatrix} H_{ae}(z) & H_{ao}(z) \\ G_{ae}(z) & G_{ao}(z) \end{bmatrix}, \quad P_s(z) = \begin{bmatrix} H_{ae}(z) & G_{ae}(z) \\ H_{so}(z) & G_{so}(z) \end{bmatrix}$$

(1)

The forward DWT can be expressed in terms of polyphase matrix as

$$\begin{bmatrix} A(z) \\ D(z) \end{bmatrix} = P_a(z) \begin{bmatrix} X_e(z) \\ z^{-1} X_o(z) \end{bmatrix}$$

(2)

and backward is represented as

$$\begin{bmatrix} X_e(z) \\ z^{-1} X_o(z) \end{bmatrix} = P_s(z) \begin{bmatrix} A(z) \\ D(z) \end{bmatrix}$$

(3)
The Lazy wavelet transform split the input $X(z)$ into even samples $X_e(z)$ and odd samples $X_o(z)$, where $X_e(z) = \sum_{k} x[2k] z^{-k}$ stands for even samples, and $X_o(z) = \sum_{k} x[2k+1] z^{-k}$ stands for odd samples of the input $X(z)$.

The input can be recovered through inverse Lazy wavelet transform that join the even samples $X_e(z)$ and odd samples $X_o(z)$ as

$$X(z) = X_e(z^2) + z^{-1} X_o(z^2)$$

(4)

For perfect reconstruction, $P_a(z) P_s(z)^T = I$ where $T$ is matrix transpose operator and $I$ is the 2 x 2 identity matrix.

It has been shown in [5] that for a given complementary pair of filters \{H_{a}(z), G_{a}(z)\}, there are always exist Laurent polynomials $S_{ai}(z)$ and $T_{ai}(z)$ for $1 \leq i \leq q$ and a non-zero constant $K$, such that

$$P_a = \prod_{i=1}^{q} \begin{bmatrix} 1 & S_{ai}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ T_{ai}(z) & 1 \end{bmatrix} \begin{bmatrix} K & 0 \\ 0 & 1/K \end{bmatrix}$$

(5)

This means that the polyphase matrix $P_a(z)$ can be factorized into finite sequence of alternating upper and lower triangular matrices. This factorization is not unique, several pairs of \{S_{ai}(z)\} and \{T_{ai}(z)\} filters are allowable. However, all possible choices give the same result for DWT realization. In practice the set \{ S_{ai}(z), T_{ai}(z) \} of filter pairs are usually of 1 to 3 taps FIR filters [1]. Computing with $S_{ai}(z)$ filter is called primal lifting or simply lifting while computing with $T_{ai}(z)$ filter is
known as dual lifting. The forward and backward lifting schemes are shown in figure 2.

3. Lifting Structure realization

The lifting scheme realizes analysis or synthesis filter bank as factorized polyphase matrix which are convenient both for design and implementation of wavelet transform. In the literature, lifting scheme architectures have been proposed [2], especially in the very last years due to increasing interest gathered by JPEG2000 deliver. The well known 5/3 bi-orthogonal is a default filter employed by JPEG2000 for lossless transforms. The analysis biorthogonal 5/3 filters \{H_a(z), G_a(z)\} have the following coefficients[1]:

\[
\begin{align*}
\text{Lowpass } H_a(z) & = -1/8 z^2 + 1/4 z^{-1} + 3/4 + 1/4 z - 1/8 z^2 \\
\text{Highpass } G_a(z) & = -1/2 z^2 + z^{-1} - 1/2
\end{align*}
\]

(7)

The polyphase matrix of above filters is

\[
P_a(z) = \begin{bmatrix}
-1/8 z^{-1} + 3/4 & -1/8 z & 1 + 1/4 z \\
-1/2 z^{-1} - 1/2 & 1/4 & 4 \\
\end{bmatrix}
\]

(8)
A probable factorization of $P_a(z)$ using two lifting steps is

$$P_a(z) = \begin{bmatrix} 1 & 0.25 (1 + z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -0.5 (1 + z^{-1}) & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$  \hspace{1cm} (9)$$

Where $S_{a1}(z) = -0.5 \, (1 + z^{-1})$ and $T_{a1}(z) = 0.25 \, (1 + z)$. The lifting steps lead to the lowpass output $A(z)$ and highpass output $D(z)$ of the filter bank shown in figure 1.

Define the even and odd samples as $x_e[k] = x[2k]$ and $x_o[k] = x[2k+1]$ then above matrix can interpret in discrete time domain as:

Predict $p_1$: $d[k] = x[2k-1] - 1/2 \, (x[2k] + x[2k-2])$  \hspace{1cm} (10)

Update $u_1$: $a[k] = x[2k] + 1/4 \, (d[k] + d[k+1])$  \hspace{1cm} (11)

where $0 \leq k \leq N/2$ for input stream data $x$ of length $N$.

The given system is a multi-rate system; the input sampling rate is $F_s$ while the output sampling rate is half or $F_s/2$. It’s visible that each of the lifting steps has alike computing outline, the disparity are in the values of input samples and multiplier factors. The determined lifting scheme should be causal for real times processing applications. It’s obvious that predict lifting is causal while update lifting is not causal. Usually this is not really a problem. The processing operations can delayed to make the system causal. In order that the result lifting scheme become adapted in real time applications the computation process is delayed by one unit time as:
\[ d[k-1] = x[2k-3] - \frac{1}{2}(x[2k-2] + x[2k-4]) \tag{12} \]

\[ a[k-1] = x[2k-2] + \frac{1}{4}(d[k-1] + d[k]) \tag{13} \]

The resultant dependence graph (DG) can be drawn for the corresponding lifting as shown in figure 3. It is important to note that \( d[k] \) and \( a[k-1] \) occurred in the same time slot. The corresponding signal flow graph (SFG) of the real time forward lifting scheme wavelet transform is depicted using DG as shown in figure 4.

Figure 3: Data dependence graph (DG)
Figure 4: The real time forward lifting scheme architecture.

The real time backward lifting scheme can be determined from forward equations as

\[ x[2k-2] = a[k-1] - \frac{1}{4} (d[k-1] + d[k]) \]

(14)

\[ x[2k-3] = d[k-1] + \frac{1}{2} (x[2k-2] + x[2k-4]) \]

(15)

The above equations are projected to get the SFG of the real time backward lifting for reconstructing the original signal as shown in figure 5.
4. Fixed-point Reversible Lifting Structure

The invertible transform means that the transform is calculated using exact arithmetic. In practice finite-precision arithmetic is usually employed, and such arithmetic is inherently inaccurate due to rounding error. In this case the transforms are reversible (i.e. invertible in finite-precision arithmetic). It is possible to create transforms that are not only invertible, but reversible as well [15]. The reversible transform map integers to integers, and approximate linear wavelet transforms. Although reversible wavelet transforms map integers to integers, such transforms are not fundamentally integer in nature. That is, these transforms are based on arithmetic over the real numbers in conjunction with rounding operations [16].

The 5/3, transforms are truly multiplierless (i.e., their underlying lifting filters all have coefficients that are strictly powers of two). Evidently, each of the resultant architecture in figures 4 and 5 has computation complexity of 4 additions and 2 shifts. The total delay between the input signal and reconstructed signal are three clocks or 3/F_s. The critical path
of each architecture is equal to the delay of the predict step plus the delay of the update step. The critical path of each lifting step is given by

\[ T_{L} = 2T_{A} + T_{S} \]

(16)

where \( T_{A} \) is the latency of the adder and \( T_{S} \) is the latency of the arithmetic shifter.

The reversible implementation of the forward and backward operations of equations (12), (13), (14), and (15) are approximated by nonlinear operations which map integers to integers. The forward IWT equations are put into practice as

\[
\begin{align*}
d[k-1] &= x[2k-3] - \left\lfloor \frac{x[2k-2] + x[2k-4]}{2} \right\rfloor \\
a[k-1] &= x[2k-2] + \left\lfloor \frac{d[k-1] + d[k]}{4} \right\rfloor
\end{align*}
\]

(17)

(18)

While the backward IWT equations are executed as

\[
\begin{align*}
x[2k-2] &= a[k-1] - \left\lfloor \frac{d[k-1] + d[k]}{4} \right\rfloor \\
x[2k-3] &= d[k-1] + \left\lfloor \frac{x[2k-2] + x[2k-4]}{2} \right\rfloor
\end{align*}
\]

(19)

(20)

Where the symbol \( \lfloor \cdot \rfloor \) means floor function. In this work all the arithmetic operations considered are fixed-point arithmetic and operands are represented as two’s complement signed integer. Under these conditions
the arithmetic right shift (symbolized by ») of a number \( V \) by \( p \) bits is equivalent to \( \lfloor V/2^p \rfloor \) or floor function that results into largest integer not larger than \( V/2^p \).

### 4.1 Numerical precision analysis

A comparison study has been implemented using Simulink of Matlab 7 to determine the number of bits required for satisfied fixed-point implementations. The study started by examining the BIBO (Bounded Input Bounded Output) gain of lifting implementation of the 5/3 biorthogonal filters. Considering the cascade equivalence relations obtained by means of the interchanging between a filter and down sampling facilities the way to compute the BIBO [2]. The equivalent low-pass filter obtained after \( j \) stage of the basic filter bank structure is

\[
H_{aj}(z) = \prod_{i=0}^{j} H_s(z^{2^i})
\]

(21)

and the equivalent high-pass filter is

\[
G_{aj}(z) = G_s(z^{2^{j-1}}) \prod_{i=0}^{j-2} H_s(z^{2^i})
\]

(22)

The BIBO analysis gain for lowpass and highpass output subbands at stage \( j \) are given respectively by

\[
B_{Lj} = \sum_n |h_{aj}[n]|
\]

(23)

\[
B_{Hj} = \sum_n |g_{aj}[n]|
\]

(24)
Where the $h_{aj}[n]$ and $g_{aj}[n]$ are inverse z-transform of $H_{aj}(z)$ and $G_{aj}(z)$ respectively.

Using the above equations the estimated values of $B_{Lj}$ and $B_{Hj}$ up to five levels of decompositions are shown in table 1. The bit-depth expansion is defined as the number of extra bits required and known as base 2 logarithmic of BIBO gain. The bit-depth figures at each level are also illustrated in table 1.

It is apparent from table 1 that the worst-case bit-depth expansion intended for lifting implementation of the 5/3 biorthogonal filters is 2 bits up to five level of decompositions.

<table>
<thead>
<tr>
<th>level ( j )</th>
<th>( B_{Lj} )</th>
<th>( B_{Hj} )</th>
<th>( \log_2(B_{Lj}) )-bit</th>
<th>( \log_2(B_{Hj}) )-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.5000</td>
<td>2.0000</td>
<td>0.5850</td>
<td>1.0000</td>
</tr>
<tr>
<td>2</td>
<td>1.6250</td>
<td>2.5000</td>
<td>0.7004</td>
<td>1.3219</td>
</tr>
<tr>
<td>3</td>
<td>1.6875</td>
<td>2.7500</td>
<td>0.7549</td>
<td>1.4594</td>
</tr>
<tr>
<td>4</td>
<td>1.6963</td>
<td>2.8047</td>
<td>0.7624</td>
<td>1.4878</td>
</tr>
<tr>
<td>5</td>
<td>1.7067</td>
<td>2.8198</td>
<td>0.7712</td>
<td>1.4956</td>
</tr>
</tbody>
</table>

The above computed values at different levels refer to filter bank or lifting implementation wavelet transform using of 5/3 biorthogonal filters. Now the case of the integer-to-integer mapping wavelet transform IWT is taken into account for the purpose of hardware completion of the proposed architecture. If the sample values of the original input signal are \( b \)-bit two’s complement integer numbers in the range

\[-2^{b−1} \leq x[n] \leq 2^{b−1} − 1\]

(25)

then the sample values range IWT lowpass and highpass subband outputs at level \( j \) are also integer numbers bounded by
\[ |a_j| \leq B_{lj} 2^{b-1} + \varepsilon_L \]  
(26)

\[ |d_j| \leq B_{lj} 2^{b-1} + \varepsilon_H \]  
(27)

Where \( \varepsilon \) bounds the effect of the floor operations used in each lifting step. It is noted in [2] that \( \varepsilon \) has negligible impact on the number bits required for representing the subband sample values. Consequently, if the signal input samples are \( b \)-bit two’s complement integer numbers, then \((b+2)\)-bit integers are sufficient to represent the reversibly transformed output subbands up to five levels of decompositions.

4.2 Experimental Results

Four input test signals are used to extract the performance of the proposed IWT architecture. The signals are shown in figure 6 and named as blocks, bumps, quad-chirp, and white gaussian noise. The input samples of each tested signal set apart as 8-bit signed two’s complement integers. All signals were 1024 samples long.

The SNR in decibels is used to measure the performance as

\[
\text{SNR} = 10 \log \left( \frac{\sum (x)^2}{\sum (x - x_r)^2} \right) \quad \text{dB}
\]  
(28)

Where \( x \) is the original input data represented as 8-bit signed two’s complement integers, \( x_r \) is the reconstructed output data.
The SNR values for input signals after five levels of forward and backward IWT are given in table 2, 3, 4, and 5. The lossless transform is happened for all input test signals at 10-bit data width, where infinity (Inf.) SNR values are gained. Therefore the data path width of the proposed architecture is fixed at 10-bit

Table 2: SNR values in dB for blocks signal.

<table>
<thead>
<tr>
<th>Bit width</th>
<th>j=1</th>
<th>j=2</th>
<th>j=3</th>
<th>j=4</th>
<th>j=5</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Inf.</td>
<td>Inf.</td>
<td>Inf.</td>
<td>Inf.</td>
<td>Inf.</td>
</tr>
</tbody>
</table>

Table 3: SNR values in dB for bumps signal.

<table>
<thead>
<tr>
<th>Bit width</th>
<th>j=1</th>
<th>j=2</th>
<th>j=3</th>
<th>j=4</th>
<th>j=5</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>24.53</td>
<td>18.26</td>
<td>15.27</td>
<td>14.07</td>
<td>13.65</td>
</tr>
<tr>
<td>9</td>
<td>Inf.</td>
<td>Inf.</td>
<td>Inf.</td>
<td>Inf.</td>
<td>Inf.</td>
</tr>
</tbody>
</table>

Table 4: SNR values in dB for quad-chirp signal.
### Table 5: SNR values in dB for white gaussian noise signal.

<table>
<thead>
<tr>
<th>Bit width</th>
<th>j=1</th>
<th>j=2</th>
<th>j=3</th>
<th>j=4</th>
<th>j=5</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>16.11</td>
<td>13.04</td>
<td>11.19</td>
<td>10.37</td>
<td>9.65</td>
</tr>
<tr>
<td>9</td>
<td>Inf.</td>
<td>47.07</td>
<td>35.58</td>
<td>31.10</td>
<td>27.58</td>
</tr>
</tbody>
</table>

5. **Proposed VLSI Architecture**

The predict and update lifting steps has a similar computing pattern. It is possible to design a single programmable process element (PE) with control inputs such that the PE can operate as predicts or updates lifting step. In order to configure the PE a two control inputs denoted as \( m \) (shift) and \( s \) (add/subtract) are applied as

\[
m = \begin{cases} 
0 & \text{shift right by } 2^2 \\
1 & \text{shift right by } 2^1 
\end{cases} \quad \text{and} \quad s = \begin{cases} 
0 & \text{Add} \\
1 & \text{Subtract} 
\end{cases}
\]

(29)

Table 6 shows the setting used in the case of forward/backward predict and update lifting steps.

### Table 6: The setting of control signal \( s \) and \( m \) for forward and backward lifting steps

<table>
<thead>
<tr>
<th>Lifting step</th>
<th>Forward Predict</th>
<th>Forward Update</th>
<th>Backward Update</th>
<th>Backward Predict</th>
</tr>
</thead>
</table>
The detailed structural design of the lifting step PE is shown in Figure 7. The one of the four categories of the forward predict, forward update, backward update and backward predict can be implemented using the given PE by selecting the corresponding control inputs m and s.

The programmable lifting step PE affixed regularity in overall system design. Hence the implementation of the forward lifting IWT is straightforward. On the other hand, a better performance can be achieved by using pipeline structure as shown in figure 8. Adding two latches between the PE$_0$ and PE$_1$ the principle of pipelining is attained. Now the critical path is enhanced and equal to critical path of one PE.
As mentioned before the backward transform can be realized using the inverse elementary operations of the forward one, taken in reversed order. Applying this idea and using the derived backward equations. The projection of pipeline backward IWT is as shown figure 9.

5.1 Integrated architecture

It’s clear that the functional block diagrams of the proposed forward and backward of IWT differ in the way the input data supplied to the pipelined processing elements PE₀ and PE₁ see the bold boxes in figures 8
and 9. It is possible to build an integrated structure that can functions as forward or backward IWT architecture by adding multiplexers with control signal \( u \). Such that if \( u=0 \) the integrated structure operates as forward IWT architecture, otherwise it operates as backward IWT architecture.

The block diagram of the programmed forward/backward IWT architecture is shown in figure 9. In the forward mode \( u=0 \), the input multiplexers select the input \( x \) to the architecture and the buffers corresponded to ‘a’ and ‘d’ outputs are actives. In the same time control selections of PEs are set to \( m_0=1, m_1=1, s_0=0 \) and \( s_1=1 \). While in backward mode \( u=1 \), \( m_0=1, m_1=0, s_0=1 \) and \( s_1=0 \), the input multiplexers route the ‘a’ and ‘d’ to be the inputs to the architecture. In the same moment the output multiplexer buffer is active to deliver the output \( x_r \).

![Forward/Backward IWT Integrated Architecture](image)

**Figure 10:** The integrated architecture for forward and backward IWT.

6. Conclusion
In this paper, the design of a programmable modular VLSI integrated architecture for computing 1D IWT is proposed. The proposed architecture is simple and cascadeable for computation of multi-levels decompositions and can be programmed to operate as 1D forward or backward IWT. The integrated architecture is independent of the size of input signal therefore it is not including any memory and this is an advantageous in VLSI design with respect to both area and time.

The precision data analysis is performed using simulink model in the environment of Malab7. The data path of the architecture is selected as 10-bit for the integer input samples of 8-bit using two’s complement representations. The 10-bit is sufficient for lossless reversible transform and up to five level of IWT.

The architecture is suitable to be used in the appliance of real time processing systems. A better arrangement is attained by using pipeline configuration which reduces the critical path of the architecture to critical path of one lifting step and consequently increase the speed of processing. With simple modifications the proposed architecture can be used as 2D IWT.

7. References


