Systolic arrays have regular computational structures, having less complex data routing and control for efficient VLSI implementation, when the length of the filter is large. The data movement and transfer play a key role in determining the efficiency of the VLSI implementation of lifting based DWT. Systolic arrays have regular computational structures as given in Kung (1982); Parhi (1999). Systolic arrays have less complex data routing and control for efficient VLSI implementation, when the length of the filter is large. Systolic arrays represent an appropriate architectural design for constructing wavelet lifting schemes. Pan & Park (1997) proposed a unified systolic array for the computation of the DST, DCT, and DHT. Denk & Parhi (1998) presented a systolic mapping technique and dependence graph to derive the DWT architectures. Chiper et al (2002) proposed architecture for the implementation of Discrete Sine Transforms. Meher et al (2008a) proposed hardware-efficient systolic-like modular design, using a new data-access scheme and a novel folding technique for the implementation of the convolutional DWT. Meher et al (2008) proposed efficient systolization of FIR filters using Distributed arithmetic. Mohanty et al (2012) proposed block processing and systolic arrays for lifting based 2-D DWT.

The block based high speed systolic VLSI architecture for the given image coefficients, for the high speed implementation of lifting based DWT is
proposed. The lifting algorithm for one and two dimensional systolic arrays is coded in VHDL, and the implementation results in Altera cyclone II FPGA show, that the hardware implementation is suitable for resource constrained high speed embedded multimedia applications.

5.1 MATHEMATICS IN THE 9/7 LOSSY LIFTING DWT FOR THE IMPLEMENTATION OF THE SYSTOLIC ARRAY

The lifting steps for the lossy 9/7 standard are used for the representation of the systolic array with four processing elements, connected as shown in Figure 5.1. Each of the processing elements is connected to the next element with a register delay. The samples \( X_o(n) \) and \( X_e(n) \) represent the odd and even input image coefficients.

![Figure 5.1 Representation of the systolic array for the 9/7 lifting DWT](image)

The processing element \( PE_0 \) computes the predict stage \( P_1 \) given by,

\[
D(n) = X_o(n) + a[X_e(n) + X_e(n + 1)]
\]

(5.1)

\( PE_1 \) computes the update stage \( U_1 \) given by,

\[
S(n) = X_e(n) + b[D(n - 1) + D(n)]
\]

(5.2)

\( PE_2 \) computes the predict stage \( P_2 \) resulting in high pass filtering,

\[
HP(n) = D(n) + C[S(n) + S(n - 1)]
\]

(5.3)
PE₃ computes the update stage U₂ resulting in low pass filtering,

\[ LP(n) = S(n) + d[HP(n-1) + HP(n)] \]  (5.4)

After N lifting steps, the scaling with \( K_1 \) and \( 1/K_1 \) is done, for normalizing the magnitude. The values of the constants of the lifting coefficients are \( a = -1.586134 \), \( b = 0.0529801185 \), \( c = 0.882911076 \), \( d = -0.443506852 \) and \( K_1 = 1.149604398 \).

5.1.1 Processor Space-Time Representation of the Lifting DWT

The systolic array architecture discussed is the B1 design, with broadcast inputs, move results, and weight stay, in the processing element. The data dependencies, data operations, and the control complexity used to derive 1-D and 2-D systolic arrays for the 9/7 lifting filter, are derived from the processor space time representation of the lifting algorithm, as shown in Figure 5.2.

Figure 5.2 Processor Space Time Representation of the 9/7 Lifting Systolic Array

\( X₀, X₁, X₂{...}X₇ \) Original image block of Coefficients

○ Intermediate Coefficients

○ Final high and low pass Coefficients
The input image coefficients \((X_0, X_1 \ldots X_N)\) are available at all the processors at the same time. The input data is broadcast to the adjacent processors with a delay. The weighting coefficients \(a, b, c, d\) appear at the same spatial coordinates. The outputs appear in the processor in different spaces and times, and move from one processing element to another, yielding high and low pass coefficients, with a delay of four clock pulses. The architecture has a regular data flow with 100% hardware utilization. The projection space vector \((P^T)\), processor space vector \((S^T)\), scheduling vector \((S^T_d)\) and Hardware Utilization Efficiency (HUE) for the proposed 9/7 lifting algorithm are:

\[
d = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad P^T = (0 1 0 0) \quad S^T = (1 0 0 0)
\]

i. **Projection space vector**

Two adjacent processing nodes are displaced by \(d\) or multiples of \(d\), and executed by the same processor. Any node with index \(I^T = (i, j)\) is mapped to the same processor.

\[
P^T I = (0 1 0 0) \begin{pmatrix} i \\ j \end{pmatrix} = j
\]

Therefore, all nodes on a horizontal line are mapped to the same processor.

ii. **Processor space vector**

The scheduling of the processor with index \(I^T = (i, j)\) is executed at the time.

\[
S^T I = (1 0 0 0) \begin{pmatrix} i \\ j \end{pmatrix} = i
\]

(5.6)
iii. **Scheduling vector**

Any node with index I would be executed at time, $S^T d$

$$S^T d = (1 \ 0 \ 0 \ 0) \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = 1$$  \hspace{1cm} (5.7)

iv. **Hardware Utilization Efficiency (HUE) = $S^T d$**,

$$\frac{1}{S^T d} = 1$$  \hspace{1cm} (5.8)

To estimate the speed performance and parallelism involved in the computation of the lifting steps, the critical path is computed, which represents the longest path necessary for the coefficients to move from the input to the output.

### 5.2 HARDWARE IMPLEMENTATION OF THE PROPOSED LIFTING ALGORITHM

The systolic array architecture for the 9/7 lossy lifting DWT architecture consists of processing elements arranged in a systolic manner. Each of the processing elements (PE) consists of 2 adders, 1 register and one multiplier. The individual processing element processes two blocks of samples for every clock period, as shown in Figure 5.3. The delay in the execution of the processing module is $Tm+2Ta$, which represents the critical path delay for the architecture, where $Tm$ and $Ta$ are the execution times of the multipliers and adders respectively.

![Figure 5.3](image)

**Figure 5.3** Representation of the single processing node in the systolic array
### 5.2.1 One Dimensional Systolic Array Architecture

The first level representation of the systolic array is done, based on the lifting steps as discussed in lifting steps (5.1-5.4) in section 5.1, and from the dependence graph shown in Figure 5.2. The construction of the one dimensional systolic processing array requires one multiplier, two adders and one delay register, as shown in Figure 5.4. The architecture can be pipelined by adding the pipeline registers to speed up the execution time.

![Pipeline Architecture](image)

**Figure 5.4 Pipelined architecture for the 9/7 lifting DWT**

Each processing element performs two additions and one multiplication, as shown in Figure 5.5. The lifting steps have similar computing steps. The one dimensional systolic array architecture consists of L rows, and each row consists of four processing elements, where L represents the required DWT levels. The pixel coefficients of an image are read from the external Static Random Access Memory. The image coefficients can be read either line-wise as discussed by Dillen et al (2003) or block-wise as discussed by Salehi & Amirtafattahi (2011). The advantage of the block based scanning of the image coefficients is that, the memory required for computing the intermediate coefficients is reduced.
Figure 5.5 Hardware requirements of the Single Processing element in the systolic array

In this architecture, the image coefficients are read in a block or group of pixel coefficients, and used for computing the lifting steps $P_1$, $U_1$, $P_2$, and $U_2$ as discussed in steps (5.1-5.4) of section 5.1, and are computed by the processing element, as shown in Figure 5.6.

Figure 5.6 Construction of the one dimensional systolic array
At each clock pulse, the processing element 1 takes a pair of odd and even coefficients, and computes the predict stage (P1). This output is given to the next processing element to compute the update stage (U1) with a delay of one clock pulse. The processing element 3 receives the coefficients from PE2 to compute the predict stage (P2). The processing element 4 computes the fourth lifting step to compute the update stage U2. The output coefficients are scaled by a factor of $K_1$ and $\frac{1}{K_1}$ to give the final first level high and low pass coefficients. A similar idea can be extended to the N dimensional input image coefficients, by properly scheduling the input coefficients. The arrays of the processing elements are arranged in such a way, as to achieve pipelining with efficient hardware utilization and high speed. After L level of decomposition of an N-dimensional image, the desired filter output is obtained.

5.2.2 Systolic Array Representation for the two Dimensional 9/7 Lifting DWT

An approach to construct a two dimensional systolic array for lifting schemes, is shown in Figure 5.7. The row processor consists of $\frac{M \times N}{2}$ processing elements arranged in a systolic manner. For column processing, using the cyclic symmetry property of the input image coefficients, the last row of the systolic array for the given levels of decomposition is overlapped with the first row of the array of processing elements, resulting in efficient hardware utilization.
The 2-D systolic array architecture for the lifting DWT is shown in Figure 5.8. The first block of image coefficients is given to the first row of processing elements. The systolic row processing is done for the input image coefficients, and it produces the intermediate matrices of $\frac{M \times N}{2}$, and they are stored in the intermediate buffer, where $M$ and $N$ are dimensions of the image. By using these intermediate coefficients, column processing is done, and the coefficients are mapped to each of the PEs in such a manner as to schedule the PEs to give a two dimensional array. The cyclic symmetry property is used for the input block of image coefficients $\{x(0)_p, x(1)_p, x(2)_p, \ldots, x(N-1)_p, x(N-2)_p, \ldots\}$, where $p$ is the given block size. The first term of the input image coefficients is overlapped with the last term of the image coefficients, in the same systolic array processing element. A 2-parallel structure of the architecture is derived for the image with a block size of $P=M \times N$, with $M=N$ with a factor of $N^2/2$. This is done to reduce the average computing time, and the number of processing elements used for the systolic array.

Figure 5.7 Block diagram of the 2-D Systolic architecture
Figure 5.8 Representation of the 2 dimensional systolic array

For computing the $j^{th}$ level DWT, $M \times N/2^{2j-1}$ samples are required in every clock cycle, where $1 \leq j \leq L$. where $L = \lceil \log_4 (M \times N) \rceil$. In each clock cycle, the PE receives $N^2$ input samples, and a row of first level sub bands is obtained with a gap of four clock cycles. Each row of matrix $A^{j-1}$ is overlapped and fed to the PE$_j$ in $2^{j-1} \times 4$ cycles.

5.2.2.1 Data access scheme of the systolic array

The images to be analyzed are read from MATLAB and loaded into the systolic VHDL file. The image size is 128x128, as shown in Figure 5.9. The architecture is generic for any square two dimensional input image.
Considering a sample of 8×8 coefficients with a block size of P=4, the block of coefficients is fed parallel to the Systolic array. The corresponding sample image coefficients with M=N=8, with the block size of P=4 of the given image coefficients, are shown in Figure.5.10.

\[
P_1 = \begin{bmatrix}
108 & 107 & 102 & 105 \\
107 & 109 & 108 & 108 \\
107 & 109 & 108 & 109 \\
107 & 108 & 108 & 110
\end{bmatrix} \quad P_2 = \begin{bmatrix}
111 & 123 & 192 & 188 \\
126 & 135 & 151 & 157 \\
111 & 116 & 131 & 184 \\
109 & 106 & 129 & 197
\end{bmatrix} \quad p_3 = \begin{bmatrix}
108 & 107 & 107 & 108 \\
146 & 109 & 108 & 109 \\
205 & 156 & 113 & 108 \\
209 & 217 & 153 & 111
\end{bmatrix}
\]

\[
P_4 = \begin{bmatrix}
109 & 106 & 112 & 146 \\
110 & 107 & 106 & 108 \\
109 & 107 & 107 & 107 \\
110 & 108 & 105 & 106
\end{bmatrix}
\]

L level decomposition has to be done for the given image of N×N with M=N. The first level works on all N^2 pixels of the original image, and each successive level works on the approximation and detail coefficients produced by the previous level, and uses only one fourth of the pixels of the previous level, resulting in fewer coefficients of the original image. Table 5.1 shows the first level of the decomposed image coefficients, after passing through the processing elements in parallel, for the blocks P_1, P_2, P_3, and P_4.
Table 5.1 First level of decomposed coefficients for the given image

<table>
<thead>
<tr>
<th>Approximation Coefficients</th>
<th>Detailed Coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LL</strong></td>
<td><strong>LH</strong></td>
</tr>
<tr>
<td>171</td>
<td>6</td>
</tr>
<tr>
<td>172</td>
<td>6</td>
</tr>
<tr>
<td>161</td>
<td>8</td>
</tr>
<tr>
<td>174</td>
<td>10</td>
</tr>
<tr>
<td><strong>HL</strong></td>
<td><strong>HH</strong></td>
</tr>
<tr>
<td>98</td>
<td>-46</td>
</tr>
<tr>
<td>128</td>
<td>-47</td>
</tr>
<tr>
<td>184</td>
<td>-48</td>
</tr>
<tr>
<td>191</td>
<td>-64</td>
</tr>
</tbody>
</table>

Each processing element in the systolic array requires two blocks of samples to compute one lifting step. Four pipeline stages are required for the efficient implementation of the 9/7 lifting filters. From the data flow Table: 5.2 it is clear, that when the block of input image coefficients is loaded into the systolic processor, it requires four clock cycles to compute the first level of lifting coefficients. A latency of eight clock cycles is needed for the implementation of the 2-D DWT.

Table 5.2 Data flow of the systolic array architecture

<table>
<thead>
<tr>
<th>Clock cycle</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input data</td>
<td>X₀₁</td>
<td>X₀₂</td>
<td>X₀₃</td>
<td>X₀₄</td>
<td>X₀₅</td>
<td>X₀₆</td>
<td>X₀₇</td>
<td>X₀₈</td>
<td>X₁₁</td>
<td>X₁₂</td>
</tr>
<tr>
<td>Predict stage 1 (P1)</td>
<td>-</td>
<td>D(0)</td>
<td>-</td>
<td>D(1)</td>
<td>-</td>
<td>D(2)</td>
<td>-</td>
<td>D(3)</td>
<td>-</td>
<td>D(4)</td>
</tr>
<tr>
<td>Update stage 1 (U1)</td>
<td>-</td>
<td>S(0)</td>
<td>-</td>
<td>S(1)</td>
<td>-</td>
<td>S(2)</td>
<td>-</td>
<td>S(3)</td>
<td>-</td>
<td>S(4)</td>
</tr>
<tr>
<td>Predict stage 2 (P2)</td>
<td></td>
<td>HP(0)</td>
<td></td>
<td>HP(1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Update stage 2 (U2)</td>
<td></td>
<td>LP(0)</td>
<td></td>
<td>LP(1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The flow-chart of the block based systolic array as implemented in this architecture, is shown in Figure 5.11.

Figure 5.11 Flowchart of the block based systolic array architecture as implemented

In this, the ROW represents the current row, and COLUMN represents the current column for the given j level of coefficients. The maximum level of DWT decomposition is L.
5.3 IMPLEMENTATION RESULTS OF THE SYSTOLIC ARRAY ARCHITECTURE

The performance analysis of the one dimensional and two dimensional systolic arrays is coded in VHDL and is generic for any square input block of image coefficients. The architectures are implemented using EP2C35F672C6 cyclone II Altera FPGA. The RTL schematic of the proposed single processing element and one dimensional systolic array is shown in Figures 5.12-5.13.

Figure 5.12 RTL schematic of single processing element as implemented

Figure 5.13 RTL schematic of one dimensional systolic array as implemented

The performance of the proposed architecture is compared in terms of the number of multipliers, number of logic elements, storage size, computing time, control complexity and hardware utilization. The computing time has been normalized to the internal clocking rate. Every two input blocks of coefficients generate two output blocks of coefficients, with a delay of four clock cycles.
Table 5.3 Synthesis report of the systolic architecture

<table>
<thead>
<tr>
<th>FPGA</th>
<th>Altera Cyclone II EP2C35F672C6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One dimensional Systolic array</td>
</tr>
<tr>
<td>Total logic elements</td>
<td>248</td>
</tr>
<tr>
<td>Total registers</td>
<td>200</td>
</tr>
<tr>
<td>Total pins</td>
<td>241</td>
</tr>
<tr>
<td>Embedded multipliers</td>
<td>8</td>
</tr>
<tr>
<td>Clock Set up period (ns)</td>
<td>4.398</td>
</tr>
<tr>
<td>Operating frequency(MHz)</td>
<td>227.38</td>
</tr>
</tbody>
</table>

From the results shown in Table 5.3 it is clear, that a critical path latency of 3.846ns with an operating frequency of 260.01MHz is achieved, for the proposed systolic array architecture, making it suitable for real time high speed multimedia applications.

5.3.1 Hardware and Timing Complexity of the Systolic Array

The systolic array representation for the first level decomposition of an image consists of 10 delay registers, four multipliers and eight adders. For N dimensional arrays the hardware requirement is given below:

(i) Number of adders

\[4N+N/2 \times 8 = 8N\]

(ii) Number of multipliers

\[2N+N/2 \times 4 = 4N\]
For an N element input image vector, the number of the computation level \( L = \log_4 N \). At each level, computations are performed through multiply and accumulate units in a pipelined fashion. The pipeline delay exists in registers for calculating the L level DWT, then for the delay \( T_d \), \( L \times T_d \) is the latency for the given input, to reach the output with the specified clock cycle. The processing time in the clock cycle for L-stage N dimensional DWT is, \( N + (L \times T_d) \). The computation time required for the individual PE in the systolic array is given by;

\[
\text{Computation time} = M \times N \times 1/L \times 1/F_{\text{max}}
\]

For an 8×8 image; \( T = 8 \times 8 \times 1/L \times 1/ (260 \times 10^6) = 0.246 \mu s \).

With \( L=1 \) for the first stage, the speed can be increased with the added stages, and is \( NT \) for the given stages. High speed can be achieved, by suitably combining multiple levels of inputs, which is desirable for high speed multimedia applications.