Abstract—Several input-traversal schedules have been proposed for the computation of the 2D Discrete Wavelet Transform (DWT). In this paper, the row-column, the line-based, and the block-based schedules for the 2-D DWT computation are compared with respect to their execution time on a Very Long Instruction Word (VLIW) digital signal processor (DSP). Implementations of the wavelet transform according to the considered schedules have been developed. They are parameterized with respect to filter pair, image size, and number of decomposition levels. All implementations have been mapped on a VLIW DSP. Performance metrics for the implementations for a complete set of parameters have been obtained and compared. The experimental results show that each implementation performs better for different points of the parameter space.

I. INTRODUCTION

One application area where the 2-D DWT has demonstrated good algorithmic performance is image compression [1]. This fact resulted in the inclusion of the DWT into image and video compression standards, namely JPEG-2000 [3] and MPEG-4 (visual texture coding) [2].

In general, two approaches exist for the realization of the DWT filtering: the conventional convolution-based implementation [4] and the lifting-based implementation [5], which employs both low-pass and high-pass filters through a ladder-structure of adders and multipliers. For optimized implementations, the lifting-based implementation is generally preferable as it reduces the required arithmetic operations [5]. In this work, the lifting implementation of two filter-pairs included in JPEG-2000 was considered, namely the 5/3 and the 9/7 filter-pairs.

Due to its computational complexity, early proposals for the realization of the wavelet transform called for application-specific integrated circuit (ASIC) realizations. Process technology and architecture developments led to programmable (instruction set) digital signal and multimedia processors with high clock rates and processing capabilities. An important advantage of (domain specific) digital signal processors compared to ASICs and general-purpose microprocessors is the good balance between flexibility and implementation efficiency. Furthermore, the application development cycles for digital signal processors are far smaller than those of ASICs.

In this paper, a comparison of the performance of various schedules for the computation of the two-dimensional DWT on a programmable VLIW digital signal processor is presented. Comparisons of 2-D DWT computation schedules have been presented in [6], [7], [8]. The comparisons presented in [6] and [7] are rather abstract and ASIC oriented. In [8] a comparison of wavelet transform schedules on programmable architectures is presented. The various schedules are compared with respect to data-cache performance, which is related to overall performance but not always in a straightforward manner. Furthermore, the experimental comparison is performed (a) on a simple-scalar processor simulator and (b) on a general-purpose processor. Consequently, although well-suited for such type of algorithms, a DSP realization was not considered in [8].

The major contribution of our work is in filling this gap by presenting the performance comparison of wavelet transform computation schedules on a real VLIW DSP architecture that forms an attractive alternative for the realization of multimedia-processing algorithms. Specifically in our work, the TriMedia TM1 processor [11] introduced by Philips has been used for the realization of the different variants of the 2-D DWT. TriMedia is a 32-bit dedicated media processor for high performance multimedia applications that deal with high quality video and audio.

The rest of the paper is organized as follows: In section II, the wavelet transform computation schedules considered in our comparison are presented. The results of the performance comparison of the wavelet transform computation schedules and the related analysis are presented in section III. Finally, in section IV, some conclusions are drawn. Extended analysis and results can be found in [12], which consists of an extended version of this work.
II. WAVELET TRANSFORM COMPUTATION SCHEDULES

Three major computation schedules have been proposed for the 2-D DWT, namely the row-column (RC) [4], the line-based (LB) [9] and the block-based (BB) [10]. The RC approach exploits the fact that the 2-D DWT is a separable transform and thus it can be computed along the rows and columns of each decomposition level. According to the RC computation schedule, the transform proceeds to the next decomposition level only when all the input data to the current level have been processed [4].

Both line-based (LB) and block-based (BB) approaches have been proposed to improve the issues of memory utilization and memory-access locality of the conventional row-column approach. The LB-approach operates with a streaming input of non-overlapping sets of input image lines to produce the wavelet coefficients at all the decomposition levels. The BB-approach operates with a raster-scan of the input image with non-overlapping blocks and produces each block’s wavelet coefficients at all the decomposition levels. As a result, the main difference in the two methods is the selected image traversal method (based on complete image rows or on blocks). The reader is referred to our analysis in [12] for pseudocode and the particular details of the implementation of the three methods.

III. EXECUTION TIME COMPARISONS

In this section, the performance of the row-column, line-based and block-based 2-D DWT computation schedules on the TriMedia TM1 processor is analyzed and compared. ANSI C codes have been developed for the three different implementations. The developed C codes are parameterized with respect to the filter pair, the number of decomposition levels and the input image size [12]. The C codes have been mapped on the TriMedia TM1 processor. Their performance for the processing of (various sizes of) four test images has been measured (and averaged) using processor’s simulator.

The following parameters were investigated in the evaluation studies of this paper: (a) both the 5/3 and 9/7 filter pairs (included in the JPEG 2000 standard); (b) (grayscale) image sizes of 128×128, 256×256, 512×512, and 1024×1024 pixels; and (c) four different total decomposition levels ranging from 3 to 6. For the smaller image sizes and for a large number of decomposition levels, the size of the wavelet subbands in the last levels may become equal to the filter size. For such cases, no performance results have been obtained.

A. 128×128-pixel images

In this case, the BB-schedule always leads to worst performance for both the 5/3 and 9/7 filter pairs. The LB-schedule always leads to the best performance except one case (5/3 filter pair and 3 decomposition levels). For both the 5/3 and 9/7 filter pairs, as the number of decomposition levels increases, the performance of the RC-schedule becomes worse, while that of the BB-schedule improves. The performance of the LB-schedule improves as the number of decomposition levels increases up to 5 and then degrades. On average, for the various numbers of decomposition levels, the LB-schedule is 3.3% and 11.3% faster than the RC- and BB-schedules, respectively, for the 5/3 filter pair; for the 9/7 filter pair, the LB-schedule is 5.2% and 13% faster than the RC- and BB-schedules, respectively. The execution time results for implementations based on the 5/3 and 9/7 filter pairs are presented in Figure 1 and Figure 2 respectively.

figure 1. Execution time for the 5/3 filter pair implementations based on image size of 128×128 pixels

<table>
<thead>
<tr>
<th>#decomposition levels</th>
<th>RC</th>
<th>LB</th>
<th>BB</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5000000</td>
<td>4000000</td>
<td>3000000</td>
</tr>
<tr>
<td>4</td>
<td>4000000</td>
<td>3000000</td>
<td>2000000</td>
</tr>
<tr>
<td>5</td>
<td>3000000</td>
<td>2000000</td>
<td>1000000</td>
</tr>
<tr>
<td>6</td>
<td>2000000</td>
<td>1000000</td>
<td>0</td>
</tr>
</tbody>
</table>

figure 2. Execution time for the 9/7 filter pair implementations based on image size of 128×128 pixels

<table>
<thead>
<tr>
<th>#decomposition levels</th>
<th>RC</th>
<th>LB</th>
<th>BB</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>6000000</td>
<td>5000000</td>
<td>4000000</td>
</tr>
<tr>
<td>4</td>
<td>5000000</td>
<td>4000000</td>
<td>3000000</td>
</tr>
<tr>
<td>5</td>
<td>4000000</td>
<td>3000000</td>
<td>2000000</td>
</tr>
<tr>
<td>6</td>
<td>3000000</td>
<td>2000000</td>
<td>1000000</td>
</tr>
</tbody>
</table>

B. 256×256-pixel images

In this case, for the 5/3 filter pair, the performance of the RC is best for 3 decomposition levels; the performance of the LB is best for 4 and 5 decomposition levels, while the performance of the BB is best for 6 decomposition levels. For the 9/7 filter pair, the performance of the LB is best for 3 and 4 decomposition levels, while the BB is best for 5 and 6 decomposition levels. The performance of the RC degrades as the number of decomposition levels increases (in almost all cases). In principle, the performance of the BB improves as the number of decomposition levels increases (for the case of the 9/7 filter pair up to the 5 decomposition levels). There is no clear conclusion for performance change versus the number of decomposition levels for the LB for both 5/3 and 9/7-filter pair realizations. On average, for the various
numbers of decomposition levels, the LB is 3.5% and 5.5% faster than the RC and BB, respectively, for the 5/3 filter pair. For the 9/7 filter pair, the LB is 3.6% and 4.9% faster than the RC and BB, respectively (faster on the average for the various numbers of decomposition levels). The execution time results for implementations based on the 5/3 and 9/7 filter pairs are presented in Figure 3 and Figure 4 respectively.

![Figure 3](image-url) - Execution time for the 5/3 filter pair implementations based on image size of 256×256 pixels

![Figure 4](image-url) - Execution time for the 9/7 filter pair implementations based on image size of 256×256 pixels

C. 512×512-pixel images

The BB has the best performance for 5 and 6 decomposition levels for both the 5/3 and 9/7 filter pairs. The LB has the best performance for 3 and 4 decomposition levels for the 5/3 filter pair, while the RC has the best performance for 3 and 4 decomposition levels for the 9/7 filter pair. In principle, the performance of the BB improves as the number of decomposition levels increases (for the case of the 9/7 filter pair pair up to the 5 decomposition levels). There is no clear relation between performance changes and increase of the number of decomposition levels for the RC and the LB for both 5/3 and 9/7 filter pair realizations. On average, for various numbers of decomposition levels, the LB is 3.4% and 5.6% faster than the RC and BB, respectively, for the 5/3 filter pair. For the 9/7 filter pair the RC is 0.4% and 1.3% faster than the LB and BB respectively (faster on the average for the various numbers of decomposition levels). The execution time results for implementations based on the 5/3 and 9/7 filter pairs are presented in Figure 5 and Figure 6 respectively.

![Figure 5](image-url) - Execution time for the 5/3 filter pair implementations based on image size of 512×512 pixels

![Figure 6](image-url) - Execution time for the 9/7 filter pair implementations based on image size of 512×512 pixels

D. 1024×1024-pixel images

In this case, the BB transform has the best performance for the larger number of decomposition levels i.e. for 4, 5, 6 decomposition levels for the 5/3 filter pair and for 5, 6 decomposition levels for the 9/7 filter pair. The LB has the best performance for 3 decomposition levels for the 5/3 filter pair. The RC has the best performance for 3, 4 decomposition levels for the 9/7 filter pair. As in the case of 512×512 pixel images, the performance of the BB improves as the number of decomposition levels increases in all cases except one (for the case of the 9/7 filter pair when the number of decomposition levels increases from 5 to 6). No clear relation between performance and the number of decomposition levels for the RC and the LB can be identified for both the 5/3 and the 9/7 filter pair realizations. On average, for the various numbers of decomposition levels the BB is 2.9% and 3% faster than the RC and the LB respectively for the 5/3 filter pair. For the 9/7 filter pair, the RC is 1.9% and 2.3% faster than the LB and BB respectively (faster on the average for the various numbers of decomposition levels). The execution time results for implementations based on the 5/3 and 9/7 filter pairs are presented in Figure 7 and Figure 8 respectively.
mainly for image sizes below 1024\times1024 pixels. The RC performs well for the 9/7 filter pair and larger image sizes (512\times512, 1024\times1024 pixels) for small numbers of decomposition levels (3, 4), while, for the 5/3 filter pair, it performs well for small images (128\times128, 256\times256 pixels) and small number of decomposition levels (3).

Independent on the computation schedule, the execution time may improve with an increase of the number of decomposition levels. The last point is an interesting and somewhat counter-intuitive result, since one expects the execution time to increase with the number of decomposition levels. However, this can be explained in the LB and BB schedules based on the increased locality of the processing.

V. ACKNOWLEDGEMENT

This work was supported in part by the SESE and by the Pythagoras II program of the EPEAEK project of the Greek government.

REFERENCES