A SURVEY OF ARCHITECTURES FOR THE DISCRETE AND CONTINUOUS WAVELET TRANSFORMS

Chaitali Chakrabarti
Dept. of Electrical Engg.
Arizona State University
Tempe, AZ 85287, USA
chaitali@asu.edu

Mohan Vishwanath
Computer Science Lab
Xerox Palo Alto Res. Center
Palo Alto CA, 94304, USA
mohan@parc.xerox.com

Robert M. Owens
Dept. of Computer Science
Pennsylvania State University
University Park, PA 16802, USA
owens@guardian.cse.psu.edu

ABSTRACT

Wavelet transforms have proven to be useful tools for several applications, including signal analysis, signal coding, and image compression. This paper surveys the VLSI architectures that have been proposed for computing the Discrete and Continuous Wavelet Transforms for 1-D and 2-D signals. The proposed architectures range from SIMD arrays to folded architectures such as systolic arrays and parallel filters. The SIMD arrays have a size that is proportional to that of the data sequence and are optimal with respect to time. The folded architectures, on the other hand, support single chip implementations and are optimal with respect to both area and time under the word-serial model.

1. INTRODUCTION

In the last few years there has been a great amount of interest in wavelet transforms, and the interest has been highly inter-disciplinary. The wavelet transform can be viewed as a decomposition of a signal in the time-scale plane. There are several types of wavelet transforms depending on the nature of the signal (continuous or discrete) and the nature of the time and scale parameters (continuous or discrete). In this paper we focus on the realizations of the Discrete Wavelet Transform (DWT) and the Continuous Wavelet Transform (CWT) [8] [3]. We assume that the time and scale parameters as well as the input and output signals of both the transforms are discrete; the two transforms differ mainly in the manner in which the time-scale plane is tiled [10]. The CWT and the DWT have a very wide range of applications: from signal analysis and signal coding to numerical analysis. The large application-domain of these transforms makes the study of their implementations in VLSI very important.

In this paper we present a survey of VLSI architectures for computing both the DWT and the CWT. The architectures range from SIMD arrays to folded architectures such as systolic arrays and parallel filters. The SIMD arrays implement the existing pyramid algorithm for DWT and the a'trous algorithm for CWT. These arrays have a size that is proportional to that of the input data sequence and are optimal with respect to time. The folded architectures, on the other hand, implement on line versions of the pyramid and a'trous algorithms. These architectures support single chip implementations in VLSI and are optimal with respect to both area and time under the word serial model.

The rest of the paper is organized as follows. In Section 2 the wavelet transform is briefly described. Sections 3 and 4 describe architectures for the 1-D and 2-D DWT, while sections 5 and 6 describe architectures for the 1-D and 2-D CWT. In each section, we first describe implementations on a SIMD array of processors and then show how it can be folded into a smaller architecture for single chip implementation. The SIMD array under study has reconfigurable interconnection, ie., the interconnection between processors is controlled by a reconfigurable switch, which if set to 1, allows the data to pass through it without any delay.

2. PRELIMINARIES

Given a time varying sequence $\hat{x}(t)$, the wavelet transform consists of computing coefiicients that are inner products of the signal and a family of 'wavelets'. The Wavelet Transform of a sequence $x(i)$ (sampled version of the continuous signal $\hat{x}(t)$), discretized on a grid whose samples are arbitrarily spaced both in time b and scale a is given by [10]

$$W(b,a) = \frac{1}{|a|^{1/2}} \sum_{i=b}^{i=aL+b-1} x(i) h\left(\frac{i-b}{a}\right) \qquad (1)$$

where h is obtained by sampling the prototype wavelet, and L is the size of the support of h. In addition, N is the number of input samples and J is the number of scales. While the properties of the wavelet transform are heavily dependent on the properties of the basic wavelet, the architectures listed in this paper are independent of the wavelet function, making them highly flexible.

DWT: The structure of the DWT is due to the dyadic nature of its time-scale grid. Here $a = 2^k$, where k is an integer, and b is a multiple of a. If N is a power of 2, then $J \leq \log N$, and $1 \leq k \leq J$. At each scale k, the number of samples in the time dimension is $N/2^k$.

CWT: As in [10], we consider a commonly used version of the CWT where the scale resolution is logarithmic as in the case of DWT. In other words, $a = 2^k$ for $1 \leq k \leq J$, and $J \leq \log N$. At each scale k, the number of samples in the time dimension is N.

Lower bounds: The lower bounds for computing the Wavelet Transforms have been derived in [11] for single chip implementations when the I/O is unilocal, place-determinate and

0-7803-2431-5/95 $4.00 © 1995 IEEE

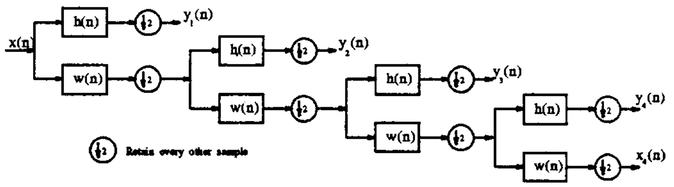

Figure 1: The DWT pyramid for $N = 16$ and $J = 4$. $h(n)$ is the high pass filter and $w(n)$ is the low pass filter.

word-local. If A is the area and T is the computation delay or period [1], then

- For 1-D DWT, $AT^2 \geq (J^2L^2k^2)$, and for the word-serial model, $A \geq (JLk)$ and $T \geq N$.
- For 1-D CWT, $AT^2 \geq (N^2L^2k^2)$, and for the word-serial model, $A \geq (NLk)$ and $T \geq NJ$.

Lower bounds for the 2-D case can be obtained by simply replacing N by N^2 and L by L^2 in the bounds for the 1-D case. For 2-D DWT, it is conjectured that when the image is available in the raster-scan format, the area is bounded by $A \geq (NLk)$, and the time is bounded by $T \geq N^2$.

3. 1-D DISCRETE WAVELET TRANSFORM

1-D DWT can be implemented by the filter bank structure shown in Figure 1. The structure is based on the recursive application of the two channel (low frequency and high frequency) subband decomposition. The filter bank structure for data size N can be implemented in a straight-forward manner on a SIMD architecture consisting of a linear array of N processors with reconfigurable interconnection [2]. For the 1st octave computation, all N processors are used. For the 2nd octave computation, the N processor array is reconfigured to a $N/2$ processor array with processors $P(2j)$ being active, $0 \leq j \leq \lfloor N/2 \rfloor$. In general, for the mth octave computation, the N processor array is reconfigured to an array of size $\frac{N}{2^{m-1}}$ with processors $P(2^{m-1}j)$ being active, $0 \leq j \leq \lfloor \frac{N}{2^{m-1}} \rfloor$. All the outputs of an octave are computed at the same time. The weights are broadcast and the partial results move from one processor to the other. Since the time to compute an output is L, the computation time for computing J octaves is LJ and the period is also LJ. The period can be reduced to L in a multigrid architecture of size $2N-1$ (there are $N/2^i$ processors in level i, $0 \leq i < logN$), and to $O(1)$ in a modified multigrid architecture of size $L(2N-1)$.

The SIMD architecture can be folded into a systolic array or a parallel filter architecture with $O(L)$ multipliers. The resulting architectures satisfy the word serial model and are optimal with respect to area and time. The systolic array architecture of Vishwanath, Owens and Irwin [13] implements an on-line algorithm called the Recursive Pyramid Algorithm [12]. In RPA, the first octave outputs are computed every 4 cycles, and all higher octave outputs are computed between two first octave output computations. The architecture in [13] consists of a linear systolic array of size L to compute both the low pass and the high pass outputs, and a storage unit to store the inputs for higher octave computations. The x-inputs that are required for the first octave computation are fed in alternate cycles to one end of the array, while the inputs that are required for the higher octave computations are fed in parallel from the storage unit. The latency of this architecture is 1 clock cycle. The x-inputs can be fed in every cycle if the architecture is modified to consist of two linear arrays, one to compute the low pass outputs and the other to compute the high pass outputs. The x-inputs are now loaded first into the storage unit and fed in parallel to the two arrays. The hardware components of the modified architecture include $2L$ MACs, LJ storage units and a simple control unit to generate the appropriate control signals. The computation time as well as the period is $\approx N$.

The sample period of the systolic array implementation is bound by the time that it takes to do a multiply-accumulate. If, however, smaller sample periods have to be supported, the systolic arrays have to be replaced by pipelined parallel filters. The pipelining latches in the parallel filter introduce a latency which is greater than 1 cycle, and consequently, a different scheduling algorithm has to be used. Chakrabarti and Vishwanath [2] have implemented a scheduling algorithm called the Modified RPA (MRPA) for large latencies. The architecture in [2] consists of 2 parallel filters each consisting of L fixed multipliers and a tree of $L-1$ adders, and a storage unit consisting of J serial-in parallel-out shift registers, each of length L. The ith shift register of the storage unit stores the inputs required for the ith octave computation $1 \leq i \leq J$. The hardware components of this architecture include $2L$ multipliers, $2(L-1)$ adders, JL storage units and a control unit to generate the appropriate control signals. The computation time as well as the period is $\approx N$. The utilization of both the systolic and parallel architectures is 100%.

Parhi and Nishitani [9] have also proposed folded architectures for 1-D forward and inverse DWT. Their architectures make use of register minimization techniques to reduce the amount of storage. The forward-backward allocation scheme used there makes the architecture irregular (ie., they have complex interconnections). This architecture supports pipelining to any level but is not easily scalable. The authors have also proposed a digit-serial architecture where each level is implemented using a different digit-sized processor. The resulting architecture achieves 100% hardware utilization and has simpler routing.

Recently, Fridman and Manolakos [5] have proposed a systematic way of generating regular computation structures using index space transformations on the dependence graph for 1-D DWT. The resulting architecture consists of L processors, and implements a schedule that is very similar to the RPA. Each processor consists of 2 multipliers, a storage of size J, and multiplexers and demultiplexers to route the data between storage units of neighboring processors. While the number of hardware components of this architec-

[1]The period is defined as the time between the initiation of two consecutive sets of computations.

ture is comparable to that of the other folded architectures, the control circuitry required to route the data is quite complex. This is because the storage unit is now distributed over L processors and so additional control is required to gate the outputs to the right processors. The sample period of this architecture is bound by the time that is required to do a multiply-accumulate.

Other architectures for 1-D DWT include the one by Knowles [6] and the one by Aware Inc. [1]. The architecture in [6] is historically the first architecture proposed for DWT. Unfortunately, it is not well suited for VLSI since it uses large multiplexers. The Wavelet Transform Processor [1] is the only commercial processor that we are aware of. The processor consists of 4 MACs and external memory and relies on software for computing the DWT.

Comparisons: A study of the 1-D DWT architectures show that while the SIMD array has an area complexity of $O(Nk)$ and period of LJ, all the folded architectures have an area complexity of $O(LJk)$ and a period of $\approx N$. This makes the folded architectures optimal under the word-serial model.

4. 2-D DISCRETE WAVELET TRANSFORM

The four channel subband decomposition of 2-D DWT can be obtained by separable applications of the two channel decomposition of 1-D DWT in the horizontal and vertical dimensions. The subband structure for $N \times N$ data can be mapped very easily onto a SIMD array of $N \times N$ processors. If row computations are followed by column computations in each octave, then for the mth octave, the array is reconfigured to form $N/2^{m-1}$ row arrays of size $N/2^{m-1}$ each for the row computations and is reconfigured to form $N/2^{m}$ column arrays of size $N/2^{m-1}$ each for the column computations. Since all the outputs of a particular octave are computed at the same time, the computation time as well as the period is $2LJ$.

The SIMD architecture has been folded into an architecture which consists of 2 systolic arrays to do the computations along the rows and and 2 parallel filters to do the computations along the columns in [13]. Two rows of inputs are fed in to the two systolic arrays every 2 cycles. The systolic arrays as well as the parallel filters each consist of L programmable multipliers. The scheduling scheme of this architecture is based on RPA in 2-D. The hardware components of this architecture include $4L$ MACs, $2N(L+1)+LJ$ storage cells, and a control unit. The computation time as well as the period is $\approx N^2$ cycles. For high sample rate applications, the systolic arrays can be replaced by pipelined parallel filters. The resulting architecture consists of 2 parallel filters to do the row computations, 2 parallel filters to do the column computations, a storage unit of size $2N(L+1)$ between the row and the column filters and a storage unit of size LJ between the column and the row filters. Compared to the architecture in [13], only one (instead of two) row of input is fed into one row filter. The computations are scheduled using MRPA in 2-D resulting in a computation time of $\approx N^2$.

In 2-D non-separable DWT, the computations in each level cannot be separated into row and column computations. Here the $N \times N$ SIMD array is configured to form an array of size $(\frac{N}{2^{m-1}} \times \frac{N}{2^{m-1}})$ for mth octave computation. All the outputs of a particular octave are computed at the same time and the computation time as well as the period is L^2J.

The above SIMD implementation has been folded into an architecture with $O(L^2)$ multipliers by Chakrabarti and Vishwanath in [2]. The architecture consists of 2 parallel filters, each consisting of L^2 programmable multipliers (programmable since each filter computes the outputs of 2 bands) and a storage unit of size NL. This architecture implements the MRPA algorithm in 2-D. The windows of the first octave computations are centered in even-numbered rows, while the windows for higher octave computations are centered in odd-numbered rows. The hardware components of this architecture include $2L^2$ multipliers, $2(L^2-1)$ adders, NL storage cells, and a control unit. The computation time as well as the period is $\approx N^2$.

Other architectures for 2-D DWT include the one proposed by Lewis and Knowles [7] and the one by Denk and Parhi in [4]. The architecture in [7] uses a specific wavelet, namely the Daubechies 4-tap filter, and as a result it does not work efficiently for other wavelets. The 2-D DWT architecture in [4] uses lapped block processing techniques since the transform operates on overlapping blocks of data. It uses register minimization techniques to reduce storage. This comes at the expense of scalability and complex routing.

Comparisons: A study of the asymptotic area complexities show that all the folded architectures have an area complexity of $O(NLk)$ while the SIMD architectures have a complexity of $O(N^2k)$. The period of the folded architectures is $\approx N^2$ while that of the SIMD arrays is L^2J (or $2LJ$ for the separable case).

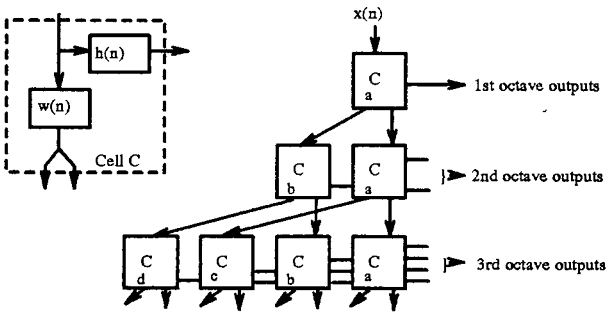

Figure 2: Reorganized 'atrous' computational structure for 1-D CWT [10]

5. 1-D CONTINUOUS WAVELET TRANSFORM

1-D CWT can be implemented by the filter bank implementation based on the reorganized "a trous" computational structure of [10] (see Figure 2). In this implementation there are 2^{j-1} cells at the jth octave, with each cell working at the rate of $1/2^{j-1}$th that of the cell of the first octave.

The above structure can be mapped onto a SIMD architecture consisting of a linear array of N processors. For the $(k+1)$th octave computation, the N processor array is reconfigured to form a set of 2^k arrays, each of size $N/2^k$. The computation time for the $(k+1)$th octave is $L2^k$ and the period is $L(2^J - 1)$. The SIMD architecture can be folded into a systolic array/parallel filter architecture. We consider two cases: when J outputs are computed every cycle and when 1 output is computed every cycle [2].

The systolic array and the parallel filter architectures for the case when J outputs are computed every cycle, consist of J computation units. The output of the low pass filter of the jth computation unit is sent to the $(j+1)$th computation unit. In both architectures, the inputs to the jth computation unit have to be subsampled by a factor of 2^{j-1}. In the parallel filter architecture this is achieved by storing the data in a delay line of size $2^{j-1}L$, and tapping the delay line every 2^{j-1} delay units. In the systolic array architecture, this is achieved by storing the partial results in registers of size 2^{j-1} in between consecutive processors. The total number of multipliers, adders and storage units are $2JL$, $2J(L-1)$ and $(2^J-1)(L-1)$ respectively. The area complexity is $O(NLk)$ and the period is $\approx N$.

For the case when there is only 1 computation unit (1 output is computed every cycle), the parallel filter and the systolic array architectures implement an on-line algorithm [2] to schedule the outputs of J octaves. The storage unit consists of J subunits of storage cells, the number of cells in the kth subunit being $2^{k-1}L$, $1 \leq k < J$. In every cycle, a data is read into the storage unit and L data are read out from the storage unit. The parallel filter and the systolic array architectures differ in the way the data is read out. The hardware components for both the architectures include $2L$ multipliers, $2(L-1)$ adders and $(2^J-1)L$ storage cells. The area complexity of both the architectures is $O(NLk)$ and the period is $\approx NJ$.

Comparisons: A study of the asymptotic area complexities show that all the folded architectures have a complexity of $O(NLk)$ while the SIMD array has a complexity of $O(Nk)$. The folded architectures for the case when only 1 output is computed per cycle has a period of $\approx NJ$, and is optimal with respect to both area and time under the word-serial model.

6. 2-D CONTINUOUS WAVELET TRANSFORM

2-D CWT can easily be mapped onto a 2-D array of $(N \times N)$ processors. For the $(k+1)$th octave computation, the $(N \times N)$ processor array is reconfigured to form a set of 4^k arrays, each of size $N/2^k \times N/2^k$. The computation time for the $(k+1)$th octave is $L^2 4^k$ and the period is $\approx L^2(4^J-1)/3$.

The systolic array and the parallel filter architectures for 2-D CWT for the case when J outputs are computed every cycle are extensions of the ones that we proposed for 1-D CWT. In the parallel filter architecture, each parallel filter consists of L^2 multipliers grouped in L subunits with L multipliers and $L-1$ adders per subunit. Similarly, in the systolic array architecture, each filter consists of L^2 processors, grouped in L subunits with a linear array of L processors in each subunit. In both the architectures, the inputs to the jth filter are subsampled along the rows and along the columns by a factor of 2^{j-1}. The total number of multipliers, adders and storage units are $2JL^2$, $2J(L^2-1)$ and $(2^J-1)(N+L)(L-1)$ respectively. The area complexity of both the architectures is $O(N^2Lk)$ and the period is $\approx N^2$.

7. REFERENCES

[1] Aware, Inc., Cambridge, MA. *Aware Wavelet Transform Processor (WTP) Preliminary*, 1991.

[2] C. Chakrabarti and M. Vishwanath. Efficient realizations of the discrete and continuous wavelet transforms: from single chip implementations to SIMD parallel computers. To appear in the IEEE Trans. on Signal Processing.

[3] I. Daubechies. The wavelet transform, time-frequency localization and signal analysis. *IEEE Trans. Info. Theory*, 36(5):961–1005, Sept 1990.

[4] T. Denk and K. Parhi. Calculation of minimum number of registers in 2-d discrete wavelet transforms using lapped block processing. *Int. Symp. on Circuits and Systems*, pages 77–81, 1994.

[5] J. Fridman and E.S. Manolakos. Distributed Memory and Control VLSI Architectures for the 1-D Discrete Wavelet Transform. *VLSI Signal Processing VII*, Oct 1994.

[6] G. Knowles. VLSI architecture for the discrete wavelet transform. *Elec. Letters*, 26(15):1184–1185, Jul 1990.

[7] A.S. Lewis and G. Knowles. VLSI architecture for 2-d daubechies wavelet transform without multipliers. *Elec. Letters*, 27(2):171–173, Jan 1991.

[8] S. Mallat. Multifrequency channel decompositions of images and wavelet models. *IEEE Trans. Acoustics Speech and Sig. Proc*, 37(12):2091–2110, Dec 1989.

[9] K. Parhi and T. Nishitani. VLSI Architectures for Discrete Wavelet Transforms. *IEEE Trans. on VLSI Systems*, 1(2), Jun 1993

[10] O. Rioul and P. Duhamel. Fast algorithms for wavelet transforms. *IEEE Trans. on Information Theory*, 38(2):569–586, Mar 1992.

[11] M. Vishwanath. *Time-Frequency Distributions: Complexity, Algorithms and Architectures.* PhD thesis, Pennsylvania State University, May 1993.

[12] M. Vishwanath. The recursive pyramid algorithm for the discrete wavelet transform. *IEEE Trans. on Signal Processing*, March 1994.

[13] M. Vishwanath, R.M. Owens, and M.J. Irwin. VLSI architectures for the discrete wavelet transform. To appear in the IEEE Trans. Circuits and Systems.