Power & Throughput Optimized Lifting Architecture for Wavelet Packet Transform

Masab Ahmad, Awais Mehmood Kamboh, and Rehan Hafiz
School of Electrical Engineering and Computer Science (SEECS)
National University of Sciences and Technology (NUST), Islamabad, Pakistan
{09beemahmad, awais.kamboh, rehan.hafiz}@seecs.edu.pk

***Abstract*—This paper presents area-power efficient architectures for the lifting based Wavelet Packet Transform (WPT). Using Daubechies 6 as an example, three different approaches to the lifting scheme implementation are optimized. For higher level decompositions, a novel Fibonacci based technique to optimally compute the number of processing elements per level is presented. Comparisons between FPGA implementations of various architectures show a throughput-to-power ratio improvement of 62% over previously implemented WPT architectures. The architecture consumes a smaller area, while consuming a dynamic power of 46mW at a maximum throughput of 342 Mbits/sec per level.**

I. Introduction

The Wavelet Packet Transform (WPT) and the Discrete Wavelet Transform (DWT) have become popular tools for time-frequency domain data analysis. The DWT being a subset of the WPT, both have found extensive use in a wide range of applications, including signal compression [1], denoising [2], feature extraction [3], and pattern recognition. The WPT is computed by passing input data through a set of filters to get the high frequency and low frequency sub-bands of information. The DWT only analyzes the lower frequency sub-bands, implicitly ignoring any information embedded in the higher frequency components. WPT on the other hand analyzes all the sub-bands providing a better analysis enabling it to achieve higher compression ratios and better scaling effects within an analysis framework, but requires more computations. Finding efficient hardware implementations of WPT is thus of great interest to signal processing community.

A WPT is typically implemented using the lifting scheme [2]. In comparison to the direct filter implementation, the lifting scheme drastically reduces the number of computations. Even though the lifting scheme is an optimized solution in itself, it still has a noticeable regularity that can be exploited to further condense and minimize required hardware. Previous efforts have mostly focused on optimizing the area consumption of their architectures implementing the WPT, while maintaining the power consumption and data throughput. Shi *et al*. [4] proposed a folded architecture based on the *9/7* filter that has reduced area and control complexity requirements, while also achieving 100% hardware utilization. However this is achieved at the cost of halving the throughput rate. Wang *et al*. [5] also proposed a folded architecture, based on the Daubechies (DB) 4 wavelet, which minimized the processing time to *N/2* cycles, where *N* is number of resulting wavelet coefficients. Aroutchelvame *et al*. [6] presented an efficient lifting scheme for the *9/7* filter that removed intermediate memory elements, thereby reducing area required for transitional storage. Further reduction in area was done in [7], which introduced a folded Computation Core (CC) architecture that time multiplexed the lifting steps using only a single unit. Unfortunately, all these architectures have tried to optimize and reduce area or memory over power and throughput.

The 'Directly Mapped Architecture', discussed in [8], has a high throughput, but consumes a hefty area due to its large scale usage of multipliers, and is therefore unsustainable for resource starved applications. A folded multiplierless architecture designed specifically for the *5/3* and *9/7* filter implementation [9] does not detail the power requirements. Also, most of these architectures have been designed for DWT up to 3 or 4 decomposition levels, and do not cater for multi-level decompositions of the WPT. Most are designed for specific wavelets and do not support other wavelet basis functions. Multi-level decompositions provide better quality analysis [1, 4], but require parallel processing elements to improve throughput, which degrades due to inefficient switching between nodes at higher levels.

This work presents three different pipelined approaches to the lifting scheme. The directly mapped architecture, the folded architecture, and the multiplierless architectures and their implementations within a Fibonacci based approach are presented. The analysis is based on the DB6 wavelet and explains power, throughput, and area requirements of these architectures. Optimal processing requirements per level for the WPT of up to 8 levels are compared upon mapping on a Field Programmable Gate Array (FPGA).

II. Lifting Architectures and Optimizations

WPTs have generally been implemented using recursive Finite Impulse Response (FIR) filters [1]. The lifting scheme reduces complexity by decomposing the conventional FIR architecture into *Predict* and *Update* steps, which halves the hardware cost. This section describes three different architectures for implementing lifting WPT. The DB6 wavelet, used as an example in this paper, contains 12 coefficients in its low-pass and high-pass filters.

A. Directly Mapped Architecture

For the 'Directly Mapped' implementation, each of the lifting predict and update steps of a wavelet can be applied separately in hardware [8]. This requires 12 multipliers and 12 adders, as exhibited in Fig. 1, where T_i and S_i represent predict and update filters respectively. The architecture can be pipelined between the T_i and S_i filters to increase hardware utilization and to reduce the overall critical path delay.

978-1-4799-3432-4/14/$31.00 ©2014 IEEE

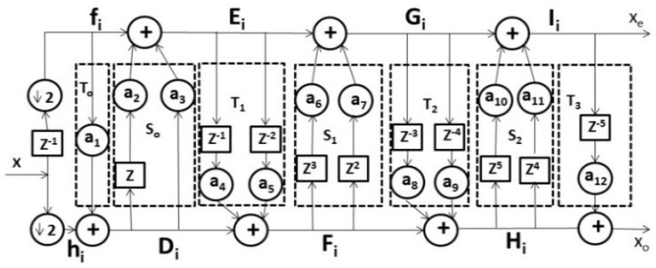

Figure 1. Directly Mapped Architecture for the DB6 Wavelet

B. Modified Folded Architecture

The filters in Fig. 1 exhibit a noticeable regularity that can be exploited to reduce area requirements. Using the folding technique [4], all of the T_i and S_i filters can be applied using a single generalized 'computation core', similar to the one constructed in [7], requiring only two multipliers and two adders as shown in Fig. 2, where a_{even} and a_{odd} represent the filter coefficients from Fig 1. *Out* denotes the result from each lifting step, *X*, *Y,* and *Z* denote the data inputs, and D_i and *P* represent the delay and pipelining registers respectively.

In [7] the hardware utilization is not optimized as computations for T_0 and T_3 filters in Fig 1 will only utilize one multiplier and one adder, reducing the overall utilization to 85.71%. The similarity, however, of the first lifting step, T_0, and the last lifting step, T_3, in the case of the DB6 wavelet's lifting scheme can be exploited. The data from these filters can be time-multiplexed onto the arithmetic hardware to maximize hardware utilization. This would mean introducing input data from the first lifting step to the non-utilized multiplier and adder, while the last lifting step is taking place. This improvement increases the hardware utilization to about 98%, while also saving 2 clock cycles to compute each output.

C. Multiplierless Architecture

Although folded architectures reduce area requirements for all algorithm implementations, they also decrease throughput, while increasing the power consumption [9]. Keeping the clock speed constant, the sequential computation of lifting steps results in a reduced throughput. Owing to the reduction in critical path, an increase in clock speed is possible, but results in an increase in power consumption. An alternative architecture is thus desirable which maximizes the throughput, at lower area and power consumption costs.

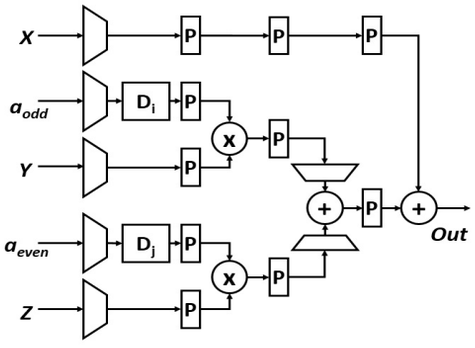

Figure 2. Modified Folded Computation Core Architecture

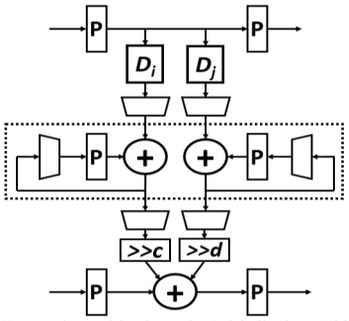

Figure 3. A Recursive adder based Multiplierless Lifting Step

Multiplication with constants can be implemented using shift and add operations. Multiplication with wavelet coefficients, which remain constant for a given wavelet basis function, can thus be implemented using multiplierless architectures. The filter coefficients are first converted to the rational form, as displayed in Table 1 for the DB6 wavelet. For a Multiplierless operation, any input is left-shifted to multiply, then added to itself the number of additions that remain after left shifting, after which it is right-shifted to divide. The folded multiplierless architecture used in [9] achieved 100% utilization, but the same technique requires several times more adders in the case of the DB6 wavelet. It's fully pipelined version increases power consumption as well.

To maintain high throughput rates, the directly mapped architecture was converted to a new pipelined multiplierless architecture. Further reduction in area was done by computing the required additions using a recursive accumulator adder, which reduces a multiplier to a single adder, along with shift registers and multiplexers. Fig. 3 illustrates a generalized recursive adder based multiplierless lifting step. D_i and D_j indicate the delay registers, *P* denoted the pipeline registers, while *c* and *d* show the shifting orders respectively. DB6 results in seven lifting steps, each one implemented separately as in Fig. 3. Additional area is consumed by control and memory blocks. The hardware utilization can be maximized, depending on the storage of the delayed samples in the delay registers. Storage of the delayed samples means that each of the lifting steps from (1) can be simultaneously applied, increasing the hardware utilization to about 100%. The critical path delay is also changed from D_m to jD_a+D_s, where D_m is the delay of a multiplier, jD_a is the delay of an adder scaled with the numerator integer of the rational filter coefficient, and D_s being the delay of a shifter respectively. A hardware comparison of each of these architectures is given below in Table 2.

Table 2. DB6 Lifting Hardware Architecture Comparison per Level

Variable	Directly Mapped [8]	Modified Folded CC. [7]	Multiplierless
Multipliers	12	2	0
Adders	12	2	29
Delay + Pipelining Registers	31 + 10	31 + 10	31 + 29
Cycles to Output	41	146	60
Critical Path Delay	D_m+D_a	D_m	$jD_a + D_s$
Throughput Rate (per cycle)	2 input/output	1 input/output	2 input/output
Control Complexity	Simple	Complex	Medium

Table 1. Original and Rational Lifting coefficients for the DB6 Wavelet

Coefficients	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
Original	-4.4345	-0.0633	0.2146	9.9700	-4.4931	-0.0237	0.0574	2.3565	-0.6788	-0.0009	0.0072	0.0941
Rational	-9/2	-1/16	7/32	10	-9/2	-1/32	1/16	5/2	-11/16	0	0	3/32

III. Wavelet Packet Transform Implementations

A. Fibonacci Derived Wavelet Packet Architecture

Previous works [4]–[8], have mostly described decompositions of up to 3 or 4 levels. The Folded CC Architecture used just one PE sequentially for all levels [7]. Shi *et al.* [4] proposed time multiplexing a single PE per level, as shown in Fig 1–3, to improve throughput of a 3 level decomposition. Both these methods reduce area requirements, but degrade the throughput significantly for higher level decompositions. This not only restricts the number of channels that can be processed, but also limits analysis of signals that are digitized at higher sample rates. According to [4], in order maintain throughput levels, each level must be computed by its own PE. However, this method requires that a single PE be time-multiplexed multiple times per level. This implies that clock cycles would be used to switch and store data between any two nodes per level. Thus 2 clock cycles are redundantly used at the first level, 6 at the second level, and so on. These latent clock cycles are independent of the type and the latency of the PE used. If an m level decomposition is required, and each level is denoted by n, where $0 \leq n \leq m$, then the latent clock cycles, L_n, at any level can be specified by (1).

$$L_n = L_{n-1} + 2^n, L_0 = 0 \quad (1)$$

For an 8 level decomposition, the latency due to multiplexing and storing would reach 510 clock cycles at level 8 if a single PE was used. This clock cycle usage is more than the cycles to output of any PE used here. Moreover, a single PE used at level 8 would have to be time multiplexed 256 times to compute the output, which would greatly reduce throughput, and increase the power usage. The design therefore requires an optimal number of PEs per level to facilitate area and throughput requirements by introducing substantial parallelism at higher levels. This optimal number of PEs per level, T_n, can be derived using the Fibonacci sequence, as given by (2) and (3). Computations for levels 1 to 4 require just one PE per level, while parallel PEs are applied at subsequent levels.

$$T_{0-4} = 1 \quad (2)$$

$$T_n = T_{n-1} + T_{n-2}, 5 \leq n \leq m \quad (3)$$

This Fibonacci based improvement significantly reduces the number of redundant read/write cycles, by introducing parallel PEs from level 5. An example improvement in clock cycle usage for two nodes is shown in Fig. 4. Parallelism is introduced from level 5 because at this level the number of latent cycles using a single PE per level outnumbers the cycles to output of any PE shown in Table 2. Fig. 5 shows this improvement if the architecture was made to decompose input data up to 8 levels. Parallelism at higher levels also reduces the overall time to output, thereby increasing throughput rates at the cost of additional area requirements.

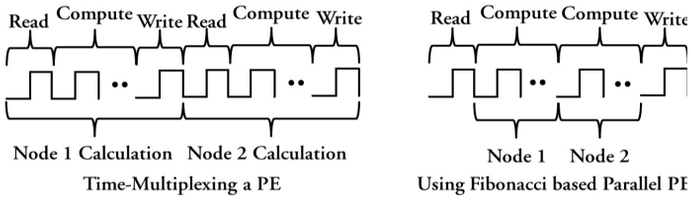

Figure 4. Clock cycle Usage Reduction using the Fibonacci based system

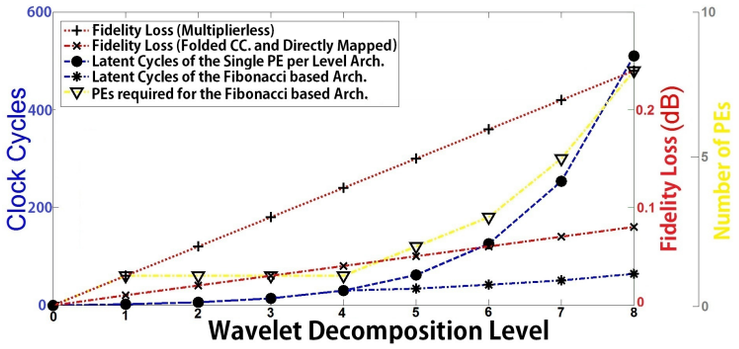

Figure 5. Performance Losses and Latent Clock Cycles of the described architectures

IV. Results and Analysis

Based on the performances of the lifting PEs discussed above in Table 2, the most suitable architecture can be selected that fits the WPT requirements. The above architectures were implemented using Verilog for the Xilinx Virtex 5 xc5vfx70t (Speed Grade -2) FPGA. A 10 bit bus width was assumed for both the data and the filter coefficients, with one sign bit, four integer bits, and five decimal bits. Table 3 presents the hardware utilizations per level, along with the control scheme for the Fibonacci based PE assignment, for each of the described designs. The directly mapped architecture uses the highest area, while the area requirements of both the folded CC and multiplierless architectures are minimal. The throughput of the folded cc architecture is the lowest, peaking at 101Mbits/sec, while that of the directly mapped architecture is the highest, with the multiplierless architecture having an intermediate output rate.

The signal fidelity loss per level must also be compared as the filter coefficient conversion to rational numbers will induce an error at each level. Thus the peak signal-to-noise ratio (PSNR) was calculated by using pseudo-random sequences as test data to each of the architectures, and comparing the outputs with their error-free results. Both the directly mapped and folded cc architectures were found to induce an error of 0.01 dB per level, while the multiplierless architecture induced an error of 0.03 dB per level, propagating a maximum error of 0.24 dB at level 8. Such a small error can be tolerated in cases of image compression [9]; hence it is assumed that it can also be tolerated in the analysis of other signals as well. Fig. 5 further gives the performance loss accumulation per level for each of the described lifting architectures for up to 8 levels.

Table 3.FPGA Implementation and Comparison of the Lifting Architectures with the Fibonacci Optimization Scheme per Level

	Directly Mapped [8]	**Modified Folded CC [7]**	**Multiplierless**
Clock Frequency	275.06 MHz	294.19 MHz	191.72 MHz
Maximum Throughput	671*Mbits/sec*	101*Mbits/sec*	342*Mbits/sec*
Slice Registers	1752	1266	1239
Slice LUTs	1451	649	2292
Slice LUT-FF Pairs	522	275	1168
Bonded IOBs	33	23	33
BUFGs	2	1	1
DSP48Es	7	2	0
Max. PSNR Loss per level (dB)	0.01	0.01	0.03

The same pseudo-random test sequences were also used to calculate the dynamic power dissipation. Clock frequencies were therefore varied from 10 MHz to the peak frequencies for each of the described architectures, and their dynamic power consumptions were noted and analyzed using the Xilinx xPower Analyzer. Fig. 6 shows the power dissipation plotted against throughput. It can be seen that the folded CC architecture consumes the highest power for its highest throughputs, while the multiplierless architecture consumes the lowest power at all the throughputs analyzed. Note that the maximum throughput of the folded CC architecture is 100Mbits/sec, while that of the directly mapped architecture can extend up to 671Mbits/sec, so the maximum range is limited to 350Mbits/sec to accurately show the power dissipation discrepancies between the architectures.

To evaluate the Fibonacci derived WPT architecture the same test sequences were used for assessment. From Fig. 5, it can be seen that clock cycle losses, and thus system throughput, depend on the decomposition levels that need to be computed. For an 8 level implementation using the DB6 multiplierless PE, our architecture saves more than 900% of latent clock cycles with only a 175% increase in area when compared with the single PE per level architecture from [4]. This propagates into an overall speed up of about 400% in terms of throughput. Power requirements, are increased per unit time as using parallel PEs will consume more energy than using single PE per level. However, given the longer latency of using single PEs per level, our Fibonacci based architecture will have a better throughput/power ratio. Fig. 7 depicts the variation of these variables per level using the multiplierless PE for the lifting architecture operating at its maximum throughput of 342 Mbits/sec per PE. Using the statistics from Fig. 7, it can be calculated that the throughput/power ratios of the Fibonacci based Architecture is approximately 62% better than that of the single PE per level/tree architecture from [4] [5] and [7]. Thus our Fibonacci derived architecture can be seen as the most efficient hardware to compute the WPT for higher decomposition levels.

In terms of area consumption, both the multiplierless and the folded CC architectures can be supported over the directly mapped architecture due to their high hardware utilizations, and lower multiplier, slice costs. On the other hand, both the directly mapped and multiplierless architecture demonstrate to be very efficient in terms of their dynamic power consumptions and throughputs. Consequently, the multiplierless architecture can be advocated for a low area, low power, and intermediate throughput implementation, while incurring minimal PSNR performance losses.

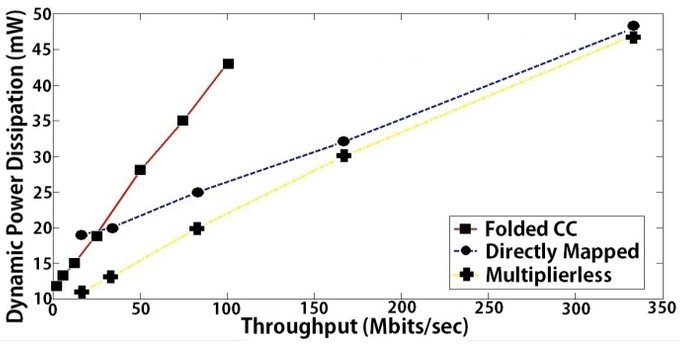

Figure 6. Dynamic Power Dissipation versus Throughput per Level

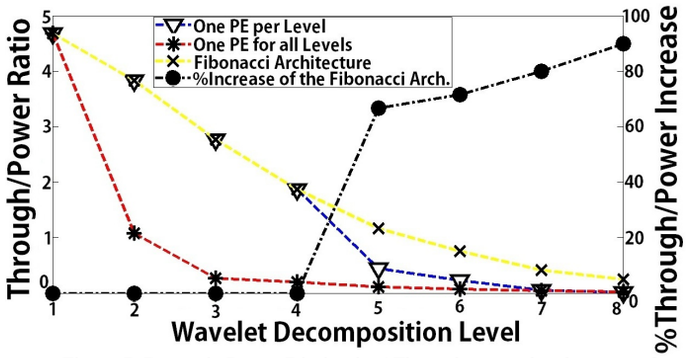

Figure 7. Dynamic Power Dissipation/ Throughput per level up to an 8 Level Implementation

V. Conclusion

In this paper three different architectures to compute the multi-level WPT have been implemented and compared. A novel Fibonacci based technique to optimize the required number of processing elements per level is also developed and implemented within the architecture. Using Daubechies 6 wavelet as an example, the analyses on area, throughput and power supports the use of multiplierless architecture to implement the lifting scheme. This architecture incurs a dynamic power dissipation of 46mW at a throughput of 342Mbits/sec per level, corresponding to a throughput/power ratio improvement of 62% over previously implemented WPT architectures. This is accomplished with a PSNR loss of only 0.24dB for an 8 level implementation on a FPGA. The presented data supports our conclusion that the generalized architecture is the most suitable approach to analyze data at higher levels of the WPT for different wavelets and basis functions.

References

[1] R. R. Coifman, Y. Meyer, and V. Wickerhauser, "Wavelet Analysis and Signal Processing," *Wavelets and their applications,* Boston, MA: Jones and Bartlett.

[2] M. Bahoura and H. Ezzaidi, "FPGA Implementation of Discrete Wavelet Transform with Application to Signal Denoising," *Circuits, Systems, and Signal Proc.*, vol. 31,issue. 3, pp. 987-1015, 2012.

[3] A. M. Kamboh and A. Mason, "Computationally Efficient Neural Feature Extraction for Spike Sorting in Implantable High-Density Recording Systems," *IEEE Trans. On Neural Systems and Rehabilitation Engineering*, vol. 21, no.1, pp. 1–9, 2013.

[4] G. Shi, W. Liu, L. Zhang, and F. Li, "An Efficient Folded Architecture for Lifting based Discrete Wavelet Transform," *IEEE Trans. On Circuits and Systems,* vol. 56, no. 4, pp. 290–294, 2006.

[5] C. Wang and W. S. Gan, "Efficient VLSI Architecture for Lifting based Discrete Wavelet Transform," *IEEE Tran. On Circuits and Systems-II: Express Briefs,* vol. 54, no. 5, pp. 422-426, 2007.

[6] S.M Aroutchelvame and K. Raahemifar, "An Efficient Architecture for Entropy-Based Best-Basis Algorithm," *Int. Conf. on Image Processing*, pp. 3281 – 3284, 2006.

[7] A. M. Kamboh, M. Raetz, K. G. Oweiss, and A. Mason, "Area-Power Efficient VLSI Implementation of Multichannel DWT for Data Compression in Implantable Neuroprosthetics," *IEEE Tran. On Biomed. Circuits and Systems*, vol. 1, no. 2, pp. 128 - 135, 2007.

[8] T. Acharya and C. Chakrabarti, "A Survey on Lifting-based Discrete Wavelet Transform Architectures," *J. of VLSI Signal Proc.*, vol. 42, issue 3, pp. 321 - 339, 2006.

[9] M. Martina and G.Masera, "Folded Multiplierless Lifting based Wavelet Pipeline," *IET Elec. Letters*, vol. 43, no. 5, pp. 27-28, 2007.