Grenze International Journal of Engineering and Technology, January Issue

Design and Implementation of Adaptive Filters

Padmavathi M[1], Umashankara Raja R[2], Abhinav Kumar[3], Dheeraj Kumar Reddy[4], Pranav Kumar[5] and Srinithin K[6]

[1]Assistant Professor, Department of Electronics & Communication Engineering, Dayananda Sagar College of Engineering, Bangalore, Karnataka

[2]Assistant Professor, Department of Physics, Sapthagiri NPS University, Bangalore

[3-6]Under Graduate BE (ECE) Students, Final Year, VII Semester, Department of Electronics & Communication Engineering, Dayananda Sagar College of Engineering, Bangalore, Karnataka

***Abstract*— This paper presents ReAdapt, a reconfigurable data path architecture that enables dynamic scaling of the energy- quality trade-off in adaptive filtering systems at runtime. Adaptive filtering is a crucial technique in many digital signal processing (DSP) applications, such as biomedical signal processing, where system accuracy requirements fluctuate depending on noise and signal strength.**

ReAdapt is designed to dynamically switch between four adaptive filtering algorithms, each offering a different level of computational complexity: Least Mean Square (LMS), Partial Update Normalized LMS (PU-NLMS), Set-Membership Normalized LMS (SM-NLMS), and Normalized LMS (NLMS). By reorganizing the data path flow and utilizing data-gating to reduce switching activity in unused modules, the system reduces energy consumption without compromising signal quality.

The ReAdapt architecture was implemented using a Field Programmable Gate Array (FPGA) on the Xilinx Vertex-5 platform, employing a reversible Wallace Tree multiplier to further optimize the design's power and delay characteristics. A case study on interference mitigation for electroencephalogram (EEG) signal processing demonstrated that ReAdapt offers substantial improvements in both energy efficiency and throughput compared to conventional architectures. Specifically, the design achieved a throughput increase of 6.8 times and an energy reduction of at least 2.84 times per operation. Additionally, this study explores the effect of runtime reconfiguration based on signal-to-noise ratio (SNR) variations. The hardware synthesis results illustrate that dynamic algorithm selection at runtime achieves superior energy- quality trade-offs compared to static single-mode systems.

The system's compact VLSI implementation makes it suitable for power- constrained environments such as wearable biomedical devices or low-latency communication systems. This work demonstrates that the proposed architecture is an efficient solution for real-time applications requiring both high performance and energy optimization. The paper also highlights the dynamic energy-quality trade-off, proving the architecture's efficiency for high-performance applications such as 5G signal processing. The matter presented in this article is the review of a paper and is the seminar / project work that was done by the under-graduate students under the guidance of the authors and there is no novelty in it and the staffs have just guided it and is a collection of the articles written by different authors as mentioned in the references.

***Index Terms*—FPGA, Reconfigurable Datapath, Adaptive Filters, Energy-Quality Trade-off, FPGA, LMS Algorithm, Reversible Multiplier.**

Grenze ID: 01.GIJET.11.1.556_2

I. INTRODUCTION

Growing demand for efficient digital signal processing (DSP) solutions This is especially true in error-prone applications such as wireless communications. biomedical signal processing and real- time data collection. The role of energy-efficient and scalable systems needs to be developed so as to isolate the desired signals from noisy environments. But as these algorithms become more complex to accommodate different signal-to-noise ratios (SNR), Therefore, it requires significant computational resources. This results in power consumption, circuit area, and… a balance between increased energy efficiency and algorithm quality, especially during runtime. It is a major challenge in designing Very Large Scale Integration (VLSI) systems... Least Mean Square (LMS), Normalized LMS (NLMS), Partially Updated NLMS (PU- NLMS), and Set Membership NLMS (SM-NLMS) to deal with noise reduction in different noisy contexts, etc. Filtering techniques. Adaptive models are widely used. The complexity of these algorithms increases as the noise level in the input signal increases. Highly complex algorithms tend to provide better noise reduction in high-noise environments. But this is not necessary by choosing highly complex algorithms in low-noise situations can use energy consumption, which may cause inefficient energy use. Therefore, a system is required that can switch between different filtering algorithms. Dynamically according to the noise context to achieve the optimal power quality trade-off [1].

Recent Advances in Power Quality Scalable Systems (EQ- scalable) has shown the possibility of balancing power quality between the design and abstraction layers of the computing stack. Scalable EQ systems aim to reduce the quality of the output signal in exchange for energy savings. Especially in fault-tolerant applications such as machine learning and multimedia processing But the application of such systems to adaptive filtering in DSP is still relatively unknown. Real-time dynamic scaling of the adaptive filtering algorithm can significantly reduce energy consumption. while maintaining acceptable output quality [2].

In this paper, we propose ReAdapt, a reconfigurable data routing architecture. It is designed to address the power quality disadvantages of runtime adaptive filtering. The ReAdapt architecture can dynamically select from four distinct adaptive filtering algorithms—LMS, NLMS, PU-NLMS, and SM-NLMS— based on the complexity required for the current noise context. By reorganizing the processing flow within the datapath and employing data-gating techniques, the ReAdapt system block, which reduces energy consumption. dynamic This approach helps the ReAdapt system achieve significant energy savings. while maintaining optimal filtering performance for various SNR conditions [3].

The ReAdapt architecture is designed to reuse common modules in all filtering algorithms. This results in the implementation of compact VLSI hardware. This modular approach allows the system to scale across different filter lengths and bit widths without excessive overhead. The main contribution of this work is the demonstration of monotonic power quality scalability during runtime. The system dynamically adjusts the filtering method based on the SNR of the input signal. Thus achieving an optimal balance between energy efficiency and filtration precision. To evaluate the performance of the ReAdapt architecture, we use a case study focused on electroencephalographic (EEG) signal processing, where noise reduction is critical for accurate data analysis. Hardware synthesis of the ReAdapt system on the Vertex-5 FPGA shows a significant increase in energy per operation. and energy per operation is significantly reduced. This is compared to existing state-of-the-art adjustable filtration systems [4].

II. ADAPTIVE FILTERS OVERVIEW

This section overviews the LMS, NLMS, PU-NLMS, and SM- NLMS algorithms and presents the main related works present in the literature [5].

A. Least Mean Squares (LMS) Algorithm

Least mean squares (LMS) algorithms are widely used in adaptive filtering. Due to its simplicity and ease of use. Its complexity can be described in terms of the number of multiplications and maps required to iterate [6].

B. Readapt: A Reconfigurable Datapath For Eq- Scalable Adaptive Filters

This section demonstrates our ReAdapt architecture proposal– the reconfigurable data path for EQ-scalable adaptive filter systems. Our ReAdapt architecture can work in four different operation modes, and each one implements a particular LMS-based AF: LMS, NLMS, PU-NLMS, and SM-NLMS. This gives the top view generic scheme to the W-tap ReAdapt filter architecture proposal. The top module of our ReAdapt architecture is generalized to be implemented at design time for W number of the filtering coefficients and n-bit input/output signals [25]. Here, the W- tap ReAdapt filter generalization applies a finite state machine (FSM) with four states for any filter length. Four blocks compose the ReAdapt architecture proposal. Those blocks are named Core, SM

NLMS comparison (SM.C.), Error block, and Update Control (U.C.). The Core block is responsible for the principal operations of the filters, i.e., output calculation and coefficients update. To scale the ReAdapt design for W-tap only needs the replication of the Core blocks W 2 times and the addition of its results by an adder tree. The Core blocks are interconnected to create the input delay sequences and appropriately update the filter weight. Our work proposes the study of a 2-tap ReAdapt architecture to verify its precision and hardware synthesis results [7].

2-tap filter is the basis for Core block replication. Proving its high efficiency in the error metrics and synthesis results, we confirm the feasibility of implementing larger circuits to obtain energy-efficient circuits. The paragraph shows, in detail, our ReAdapt datapath architecture proposal which has two signal inputs (reference signal x(k), and desired signal d(k)) and three control inputs named Selector, γ, and Step. The inputs γ and Step define the variables of the filters SM-NLMS and PU NLMS, respectively. The filter mode is defined by the Select input. The SM-NLMS comparison block decides the most suitable value of μ(k) for the SM-NLMS filters. The Error block calculates the filter error based on the input d (k and the filter output y(k. The Update Control delivers the appropriate updating factor depending on the input selector [24]. The blocks employ two multipliers (X), two adders (+), one subtraction (-), one divider (/), one shift operator ( ), and a comparator block (≥) to control the SM-NLMS updates condition. The blocks are controlled by a FSM with four clock cycles. The operations employ a fixed-point representation using the K least signification bits (LSBs) to represent the fractional part. The most significant bit (MSB) is the sign value, and the n−K following MSBs are the integer part. Figs. 2(b-d) present the VLSI architecture behavior in the different modes. Then, it shows the architecture in operation for the LMS mode that disables the SM- NLMS comparison and Update Control blocks. It comprises the architecture for NLMS and PU-NLMS modes that exclude just the SM-NLMS block. The difference between these two modes is the FSM. Finally, the circuit also implements the architecture configured for SM-NLMS mode, which utilizes the whole architecture and those respective control signals. Data-gating technique is implemented in each mode to block switching activity (i.e., reducing dynamic power) of unused modules represented as shaded blocks [8].

The ReAdapt FSM works with four clock cycles per sample.In the four operation modes, the first cycle realizes the filter output calculation (y(k)) and the second cycle calculates the system error (e(k)) and variables necessary to update the coefficients (e(k) μ and e(k) γ). The last two cycles are required to update the coefficients once per cycle (W(k)) [23]. We adopt the following strategies to decrease the architecture complexity, aiming for an efficient operation with reduced circuit area and energy. Firstly, we reuse the same architecturefor the FIR Filter and coefficients adaptation algorithms to minimize the number of arithmetic units and critical path delay. Then, we define the μ assuming power of two values tosimplify the multiplication μ e(k) for just a shift operation, realized in the Error block. Another strategy is applied when the LMS operates, data-gated the divisor inputs skipping its result to reduce dynamic power. The last technique is related tothe SM-NL [9].

The last technique is related to the SM-NLMS. The condition to update the filter coefficients for SM-NLMS, which needs a division and one subtraction,is presented in (4). If we directly apply the SM-NLMS updatecondition (4) in (2) to reduce the complexity, equation (11) is found. Using the distributive property and factoring for the variable x(k) we found (12). Solving the multiplication the division γe(k) / |e(k)| and simplifying the resultant equation, we found a new update equation (13) which eliminates the division γ / |e(k)| and simplifies the two multiplications of μ∗e(k)∗x(k) for one sum and one, multiplication, x(k)(e(k)±γ). As the update condition equation (4) uses a modular value of error, the operation γe(k) / |e(k)| can create a negative sign in case of error e(k) is negative [10].

The sign ± guarantees the proper filter operation, creating a condition such that, if the error is positive, apply the subtraction; otherwise, apply the sum. However, we can simplify it by changing just the sign of γ by 2's complement when necessary and using a single adder. We can see the simplification in the SM-NLMS comparison block. We use an extra input defined as Step in the PU-NLMS mode operation to change the coefficients updating frequency. The input Step decides the iteration that the registers W0 and W1 enable the update of the filter coefficients values. Usually, in the literature, the adaptive filters are decomposedinto two systems, the FIR filter and the adaptive algorithms [11].

However, one observes an increase in hardware requirements once the replication of arithmetic units without reusing. There-fore, our architecture implements the filter as a single system to deal with these issues and reuses the arithmetic units to calculate the filter output and the coefficient update. Table I shows all properties for each ReAdapt operating mode, such as control signal Select definition, steps, operations, and blocks used [22]. Worth mentioning that the Select sign is a priority for the architecture, and one can switch between modes at any moment independent of the FSM. As the modes present different processing modes, Table II shows the number of arithmetic operations realized for each operating mode for a single sample. [21] highlights the mathematical

complexity difference, considering the number of arithmetic operations between the filters (i.e., sum/subtraction, multiplication, and division) for each sample. Working with 2-tap, the calculation of filter output y(k) in all modes and the α calculation for NLMS, PU-NLMS, and SM-NLMS modes implement two multiplication and one sum each. The results highlight the lower LMS mathematical complexity when compared to the others due to the divider operation. Even the PU-NLMS and SM-NLMS have a higher number of arithmetic operations than NLMS, the extra adder units are used to reduce the switching activity of the multipliers by fixing the same coefficients for many cycles during its runtime operation [12].

We divided the partial product generation circuit into four sections. These sections communicate with each other by FG gate. In each part, the input of the TG gate is provided by the outputs of FG or TG gates. This indicates that these sections need not wait to receive the input of the other sections. On the other hand, all four sections operate in parallel, and the whole delay of the circuit is equal to the maximum delay. This circuit has 24 constant inputs and 16 garbage outputs. Since quantum cost of TG gate equals to 5 and quantum cost of FG gate equals to 1, quantum cost of the proposed circuit is 88. The delay is equal to 3. In addition, its hardware complexity is 24α + 16β [13].

The second approach for producing the partial products is illustrated. As can be seen, this approach applied reversible TG and PG to compute the partial products. In this approach, we attempted to apply less number of gates, constant inputs and garbage outputs. Since the low-cost PG is used to produce AND operation, it can reduce total quantum cost of the circuit as much as possible. This circuit has 16 constant inputs and 8 garbage outputs. Since quantum cost of PG gate equals to 4 and quantum cost of TG gate equals to 5, quantum cost of the proposed circuit is 73 [20]. The delay is equal to 4. Moreover, its hardware complexity is 23α + 16β. So, it can be concluded that in the design of the proposed reversible multiplier circuit, if the circuit speed has higher priority than other criteria, it is better to use the first design. If the number of constant inputs and garbage outputs, quantum cost and hardware complexity have higher priority than delay, then it is better to utilize the second method. After computing the partial product bits, In the proposed circuit, the PG gate and Feynman's block have been used as reversible HA and FA blocks, respectively. The proposed Wallace-based reversible multiplier is shown [14].

III. Models of Approximation Restoring Divisions

An 8-bit dividend/4-bit divisor version of restoring array divider is shown, where each row performs a trial subtraction. The result of trial subtraction being positive or negative determines the quotient bit and the partial remainder. Although the array divider looks similar to an array multiplier, the latency of the array divider is much higher because of ripple-borrow subtraction in each row, and each row has to wait for the execution of the previous row. In a 2n/n restoring divider with a quotient of n bits, the basic rule to avoid overflow is that the first n bits of the 2n bits wide dividend should be less than n bits of divisor [19]. As shown in each row, n bits of subtrahend are subtracted from n + 1 bits of minuend. Subtrahend is always the n-bit divisor B . In the first row, most significant n + 1 bits of dividend A are taken as minuend. The quotient bit is 1 if there is no borrow after subtraction or if the most significant bit of minuend is 1, otherwise the quotient is 0. Each quotient bit decides whether the partial remainder of the respective stage is the result of subtraction (when the quotient bit is 1) or the least n bits of minuend. In the following row, the partial remainder with the consecutive dividend bit forms the minuend and steps are repeated until n bits of quotient are obtained [15].

Approximate restoring divider- Model 1: In the first approximation model of the restoring divider, some of the exact restoring cells are replaced with approximate restoring cells. Each exact cell in restoring division is made of a full subtractor and a 2-1 multiplexer as seen. The two outputs of the exact cell can be given by (2) and (3), where bout is the borrow-out of the restoring cell and rout is the partial remainder bit. The exact results are given in [18]. As it can be seen from the table, when some values of bout and rout are modified, logic equations can be simplified as given in (4) and (5). Modified cells are represented in square boxes in table III. For approximate divider model 1 (AD-M1) with approximation factor p, p(p + 1)/2 exact restoring cells are replaced with approximate restoring cells. A 8/4 AD-M1 with p=4 is shown, where AC represents approximate cells [16].

Approximate restoring divider- Model 2, where in approximate divider model 2 (AD-M2), a 2n/n divider is approximated to a (2n- p)/(n- p) divider. p number of cells are reduced in each row by reducing the number of bits in the divisor by a factor p. The first step is reducing the n-bit divisor to a n - p-bit divisor. A leading-one detector is used in first p bits of B to truncate p bits. After finding the leading-one starting bit-position (sbp), B can be trimmed to the number of bits trimmed at the beginning and end of the divisor and dividend can be the same, since the initial n bits of A are less than B. After trimming the bits, overflow is possible if the most significant n- p bits of A are equal to n- p bits of B. To compensate for overflow, an equality detector is used

and, when equality is detected, quotient bits are set to I. The remainder is shifted left by sbp- ((n- p)- 1) bits. A 8/4 AD-M2 with p=2 is observed [17].

V. Conclusions

Some strategies in the literature implement the LMS-based filters on efficient circuits, regarding power dissipation and processing speed. The work implements approximate distributed arithmetic circuits in LMS filters by replacing the multipliers and adders with radix-8 booth encodes, Wallace tree, and carry lookahead adders. As the authors used an architecture already presented in the literature, LMS filter in parallel form, the circuit uses many mathematical operators and registers. Considering a W-tap filter, the architecture of needs 2*W multipliers, 2*W registers, and 3*W-1 adders. In the same way, proposed an LMS, NLMS, and RLS filters application in parallel form with minimum components amount of 2*W+1 multipliers, 2*W adders, and 2*N memory location considering the LMS application [19].
Regarding the NLMS case, the authors added two multipliers, one adder, and one divider. Worth mentioning that the work implements each application independently. On the other hand, presented a fully-sequential architecture for the LMS algorithm in FPGA. Even using just one multiplier, one adder, one register, and one divider for the adaptive algorithm, the authors omitted the component for FIR filter architecture and the number of clock cycles, which is much higher, making impossible a direct comparison to our work in this regard. We demonstrate in Table IV that the energy overhead of our ReAdapt architecture proposal on top of its single NLMS architecture is less than +6%. The overhead of the related work is +27.7% of total power (i.e., in energy also) on top of the NLMS to an reconfigurable architecture which implements just two algorithms (i.e., NLMS and LMS). Following the synthesis results comparison, we consider the strategy proposed by and simulate the number of arithmetic operators for a W-tap filter. Likewise, we consider a general architecture for the W-tap filter based on the NLMS architecture [18].

References

[1] S. F. Obermann and M. J. Flynn, "Division algorithms and implementations," IEEE Transactions on Computers, vol. 46, no. 8, pp. 833- 854, Aug 1997.

[2] M. D. Ercegovac and J. Muller, "Digit-recurrence algorithms for division and square root with limited precision primitives," The Thrlty-Seventli Asilomar Conference on Signals, Systems & Computers, 2003, Pacific Grove, CA, USA, 2003, pp. 1440-1444 Vol.2.

[3] M. Ercegovac and T. Lang, "Digital Arithmetic". Morgan Kaufmann Publishers, 2004.

[4] M. J. Schulte, J. Omar, and E. E. Swartzlander Jr, "Optimal Initial Ap proximations for the Newton- Raphson Division Algorithm," Computing, vol. 53, pp. 233-242, 1994.

[5] B. Parhami, "Computer Arithmetic- Algorithms and Hardware designs", Oxford University Press, 2000.

[6] S. Hashemi, R. I. Bahar and S. Reda, "A low-power dynamic divider for approximate applications," 2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC), Austin, TX, 2016, pp. 1-6.

[7] L. Chen, J. Han, W.Liu and F.Lombardi, "On the Design of Approximate Restoring Dividers for Error-Tolerant Applications," IEEE Transactions on Computers, vol. 65, no. 8, pp. 2522-2533, Aug. 1 2016.

[8] Raha, S. Venkataramani, V. Raghunathan, and A. Raghunathan, “Energy-efficient reduce-and-rank using input-adaptive approximations,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 25, no. 2, pp. 462–475, Feb. 2017.

[9] Raha and V. Raghunathan, “Approximating beyond the processor: Exploring full-system energy-accuracy tradeoffs in a smart camera system,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 26, no. 12, pp. 2884–2897, Dec. 2018.

[10] G. Paim, L. M. G. Rocha, G. M. Santana, L. B. Soares, E. A. C. da Costa, and S. Bampi, “Power-, area-, and compression efficient eight-point approximate 2-D discrete Tchebichef transform hardware design combining truncation pruning and efficient transposi tion buffers,” IEEE Trans Reg. Papers, vol. 66, no. 2, pp. 680–693, Feb. 2019.

[11] Cheffi, M. Djendi, and A. Guessoum, “A new correlated set membership partial-update NLMS (SM-PU-NLMSCOR) algorithm for acoustic noise reduction,” Proc. 5th Int. Conf. Electr. Eng. Boumerdes (ICEE-B), Oct. 2017, pp. 1–4.

[12] M. Rupp, W. Kellermann, A. Zoubir, and G. Schmidt, “Advances in adaptive filtering theory and applications to acoustic and speech signal processing,” EURASIP J. Adv. Signal Process., vol. 2016, no. 1, pp. 1–3, Dec. 2016.

[13] G. Akkad, A. Mansour, B. A. ElHassan, E. Inaty, R. Ayoubi, and J. A. Srar, “A pipelined reduced complexity two-stages parallel LMS structure for adaptive beamforming,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 67, no. 12, pp. 5079– 5091, Dec. 2020.

[14] H. B. M. Ajjaiah, P. V. Hunagund, and B. R. Rao, “Adaptive filters in digital transmission based on improved LMS algorithm,” Proc. Int. Conf. Wireless Commun., Signal Process. Netw. (WiSPNET), Mar. 2016, pp. 985–988.

[15] V. Guidotti, G. Paim, L. M. G. Rocha, E. Costa, S. Almeida, and S. Bampi, “Power-efficient approximate Newton–Raphson integer divider applied to NLMS adaptive filter for high-quality interference can celling,” Circuits, Syst., Signal Process., vol. 39, no. 11, pp. 5729–5757, Nov. 2020.

[16] B. La Rosa et al., “Exploring NLMS-based adaptive filter hardware architectures for eliminating power line interference in EEG signals,” Circuits, Syst., Signal Process., vol. 40, no. 7, pp. 3305–3337, Jul. 2021.
[17] P. U. da Costa, G. Paim, L. M. G. Rocha, E. A. C. da Costa, S. J. M. de Almeidaf, and S. Bampi, “Fixed-point NLMS and IPNLMS VLSI architectures for accurate FECG and FHR processing,” IEEE Trans. Biomed. Circuits Syst., vol. 15, no. 5, pp. 898–911, Oct. 2021.
[18] M. M. N. Mannan, M. A. Kamran, and M. Y. Jeong, “Identification and removal of physiological artifacts from electroencephalogram signals: A review,” IEEE Access, vol. 6, pp. 30630–30652, 2018.
[19] M. Liu, M. Guan, Z. Wu, C. Sun, W. Zhang, and M. Wang, “High-speed VLSI implementation of an improved parallel delayed LMS algorithm,” Mobile Netw. Appl., vol. 27, pp. 1–11, Jan. 2022.
[20] M. Chandra, P. Goel, A. Anand, and A. Kar, “Design and analysis of improved high-speed adaptive filter architectures for ECG sig nal denoising,” Biomed. Signal Process. Control, vol. 63, Jan. 2021, Art. no. 102221.
[21] B. K. Das and M. Chakraborty, “Sparse adaptive filtering by an adaptive convex combination of the LMS and the ZA-LMS algorithms,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 61, no. 5, pp. 1499–1507, May 2014.
[22] Ray, N. V. George, and P. K. Meher, “Analysis and design of unified architectures for zero- attraction-based sparse adaptive filters,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 28, no. 5, pp. 1321–1325, May 2020.
[23] M. J. Schulte, J. Omar, and E. E. Swartzlander Jr, "Optimal Initial Ap proximations for the Newton- Raphson Division Algorithm," Computing, vol. 53, pp. 233-242, 1994.
[24] S. Hashemi, R. I. Bahar and S. Reda, "A low-power dynamic divider for approximate applications," 2016 53nd ACM/EDAC/IEEE Design Automa tion Conference (DAC), Austin, TX, 2016, pp. 1- 6.
[25] L. Chen, J. Han, W.Liu and F.Lombardi, "On the Design of Approximate Restoring Dividers for Error-Tolerant Applications," IEEE Transactions on Computers, vol. 65, no. 8, pp. 2522-2533, Aug. I 2016.