

AREA DELAY POWER EFFICIENT FIXED POINT LMS ADAPTIVE FILTER WITH LOLAYW ADAPTATION

1. S.NARESH BABU, 2. M.VARA PRASAD

1. PG Scholar, Dept of ECE, Anubose Institute of Technology, Palwancha, khammam 2. Dept of ECE,
Anubose Institute of Technology, Palwancha, khammam

ABSTRACT:

This project presents a novel partial product generator and proposes a strategy for optimized balanced pipelining across the time-consuming combinational blocks of the structure. To achieve lower adaptation-delay and area-delay-power efficient implementation proposed technique is adapted. In proposed method we use large processing elements (PEs) for achieving a lower adaptation delay with the critical path of one MAC operation. They have proposed a fine-grained pipelined design to limit the critical path to the maximum of one addition time, which supports high sampling frequency, but involves a lot of area overhead for pipelining and higher power consumption due to large number of pipeline latches. An efficient adder tree for pipelined inner-product computation is used to minimize the critical path and silicon area without increasing the number of adaptation delays.

INTRODUCTION:

The Least Mean Square (LMS) adaptive filter is the most popular and most widely used adaptive filter because of its satisfactory convergence performance. The direct-form LMS adaptive filter involves a long critical path due to an inner-product computation to obtain the filter output. The critical path is reduced by pipelined implementation. The conventional LMS algorithm does not support pipelined implementation because of its recursive behavior. It is modified then it called by delayed LMS (DLMS) algorithm which allows pipelined implementation of the filter. A lot of work has been done to implement the DLMS algorithm in systolic architectures to increase the maximum usable frequency but, they involve an adaptation delay of $\sim N$ cycles for filter length N , which is quite high for large order filters. Since the convergence performance degrades considerably for a large adaptation delay has proposed a modified systolic architecture to reduce the adaptation delay. A transpose-form LMS adaptive filter, where the filter output at any instant depends on the delayed versions of weights and the number of delays in weights varies from 1 to N . In systolic architecture, they have used relatively large processing elements (PEs) for achieving a lower adaptation delay with the critical path of one MAC operation. A fine-grained pipelined design to limit the critical path to the maximum of one addition time, which supports high sampling frequency, but involves a lot of area overhead for pipelining and higher power consumption than due to its large number of pipeline latches. Further effort has been made to

reduce the number of adaptation delays. With an efficient adder tree for pipelined inner-product computation to minimize the critical path and silicon area without increasing the number of adaptation delays. Adaptive digital filters have a wide range of communication in area and DSP applications such as adaptive equalization, system identification, and image restoration and noise cancelling and flip-flop clustering. In adaptive filter mostly used algorithm is the least mean-square (LMS) algorithm because of its extra performance and simple calculation.

LITERATURE SYRVEYE:

In the transposed form in, the operands of the multipliers in the MCM module are the current input signal $x[n]$ and coefficients. The results of individual constant multiplications go through structure adders (SAs) and delay elements. In the past decades, there are many papers on the designs and implementations of low-cost or high-speed LMS filters [1]–[13], [15]–[19]. In order to avoid costly multipliers, most prior hardware implementations of digital LMS filters can be divided into two categories: multiplierless based and memory based. Multiplierless-based designs realize MCM with shift-and-add operations and share the common suboperations using canonical signed digit (CSD) recoding and common subexpression elimination (CSE) to minimize the adder cost of MCM [1]–[10]. In [18] and [19], more area savings are achieved by jointly considering the optimization of coefficient quantization and CSE. Most multiplierless MCM-

based LMS filter designs use the transposed structure to allow for cross-coefficient sharing and tend to be faster, particularly when the filter order is large. However, the area of delay elements is larger compared with that of the direct form due to the range expansion of the constant multiplications and the subsequent additions in the SAs. In [17], Blad and Gustafsson presented high-throughput (TP) LMS filter designs by pipelining the carry-save adder trees in the constant multiplications using integer linear programming to minimize the area cost of full adders (FAs), half adders (HAs), and registers (algorithmic and pipelined registers). The existing work on the DLMS adaptive filter does not discuss the fixed-point implementation issues, e.g., location of radix point, choice of word length, and quantization at various stages of computation, although they directly affect the convergence performance, particularly due to the recursive behavior of the LMS algorithm. Therefore, fixed-point implementation issues are given adequate emphasis in this paper. Besides, we present here the optimization of our previously reported design [13], [14] to reduce the number of pipeline delays along with the area, sampling period, and energy consumption. The proposed design is found to be more efficient in terms of the power-delay product (PDP) and energy-delay product (EDP) compared to the existing structures

FILTER:

In signal processing, a **filter** is a device or process that removes from a signal some unwanted component or feature. Filtering is a class of signal processing, the defining feature of filters being the complete or partial suppression of some aspect of the signal. Most often, this means removing some frequencies and not others in order to suppress interfering signals and reduce background noise. However, filters do not exclusively act in the frequency domain; especially in the field of image processing many other targets for filtering exist. Correlations can be removed for certain frequency components and not for others without having to act in the frequency domain.

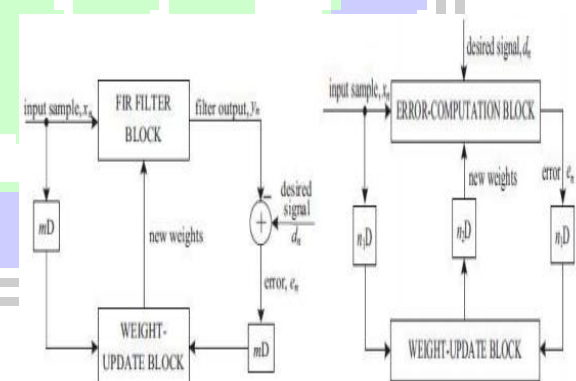
There are many different bases of classifying filters and these overlap in many different ways; there is no simple hierarchical classification. Filters may be:

- linear or non-linear
- time-invariant or time-variant, also known as shift invariance. If the filter operates in a

spatial domain then the characterization is space invariance. causal or not-causal: depending if present output depends or not on "future" input; of course, for time related signals processed in real-time all the filters are causal; it is not necessarily so for filters acting on space-related signals or for deferred-time processing of time-related signals. analog or digital discrete-time (sampled) or continuous-time passive or active type of continuous-time filter infinite impulse response (IIR) or finite impulse response (FIR) type of discrete-time or digital filter.

BLOCK DIAGRAM OF LMS ADAPTIVE FILTER :

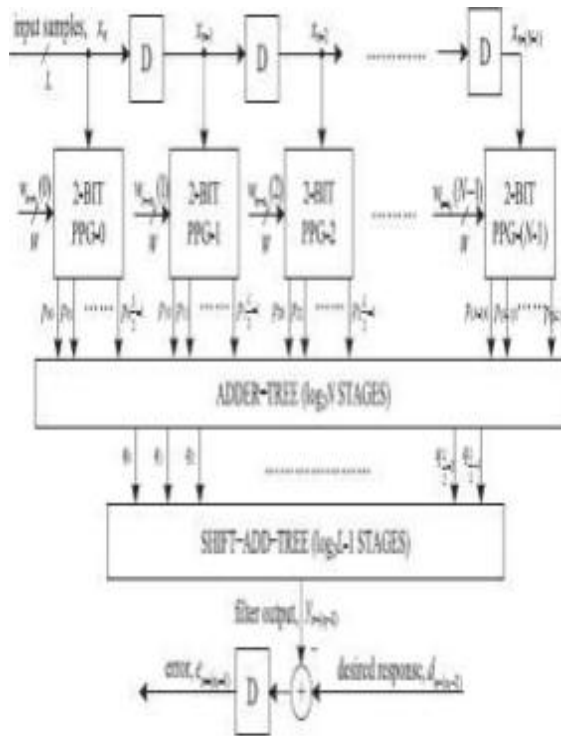
There are two main computing blocks in the general block adaptive filter is Error-Computation block and Weight Update block. The general block diagram of the Delayed LMS adaptive filter is shown in and the delay introduced by the whole of adaptive filters structure. In proposed method the adaptation delay of conventional LMS can be decomposed into two parts: one part of the delay is introduced by the pipeline stages in LMS filtering, and another part of the delay involved in pipelining of weight update process. It can perform optimal pipelining by feed forward cutest retiming of both of these sections separately. Because it is used to minimize the number of pipeline stages and adaptation delay in each of the condition. The Filter output $y(n)$ and the desired signal should be compared and the error signal given to the weight update block.



ERROR COMPUTATION BLOCKS:

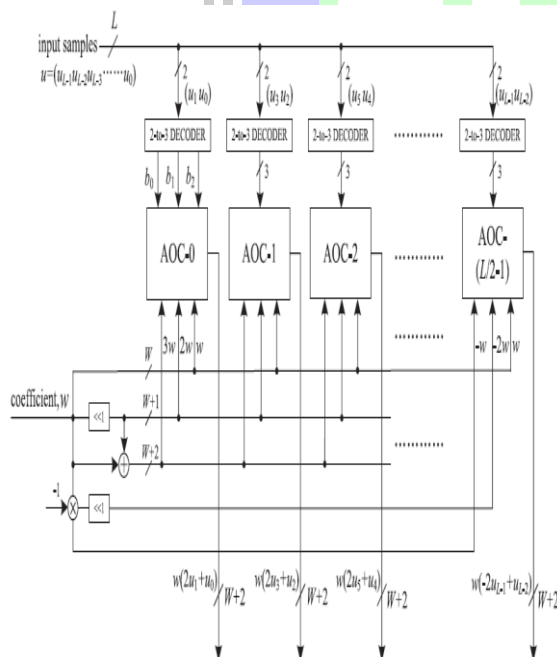
In the error computation block the each of the input are given to the D and the 2-BIT PPG and each of the condition to check the co efficient of each sample of inputs. N number of 2-b partial product generators (PPG) with respect to N num of

multipliers and a cluster of $L/2$ binary adder trees. It should be based on adder tree and shift adder tree.



Structure of PPG:

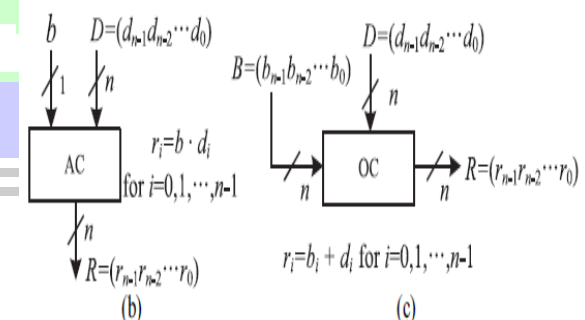
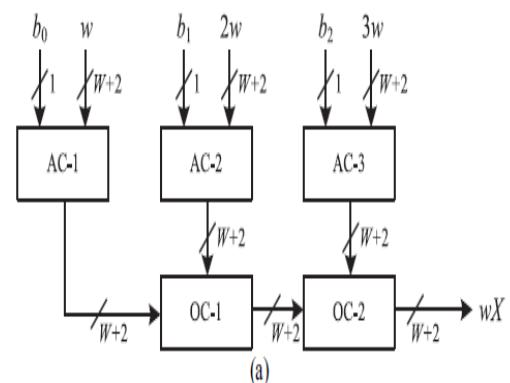
The structure of each PPG is shown in below figure. It consists of $L/2$ number of 2-to-3 decoders and the same



number of AND/OR cells

In the PPG $L/2$ number of 2-to-3 decoders and also the same number of AND/OR cells (AOC).the 2 to 3 decoders to produce the three outputs are b_0, b_1 and b_2 .the co efficient value of each $w, 2w, 3w$ are fed to the AOC. (AOC).1 Each of the 2-to-3 decoders takes a 2-b digit (u_1u_0) as input and produces three outputs $b_0 = u_0 \cdot u_1, b_1 = u_0 \cdot u_1$, and $b_2 = u_0 \cdot u_1$, such that $b_0 = 1$ for $(u_1u_0) = 1, b_1 = 1$ for $(u_1u_0) = 2$, and $b_2 = 1$ for $(u_1u_0) = 3$. The decoder output b_0, b_1 and b_2 along with $w, 2w$, and $3w$ are fed to an AOC, where $w, 2w$, and $3w$ are in 2's complement representation and sign-extended to have $(W + 2)$ bits each. To take care of the sign of the input samples while computing the partial product corresponding to the most significant digit (MSD), i.e., $(u_{L-1}u_{L-2})$ of the input sample, the AOC $(L/2 - 1)$ is fed with $w, -2w$, and $-w$ as input since $(u_{L-1}u_{L-2})$ can have four possible values 0, 1, -2, and -1.

STRUCTURE OF AOCs: The structure and function of an AOC are depicted in below figure. Each AOC consists of three AND cells and two OR cells. The structure and function of AND cells and

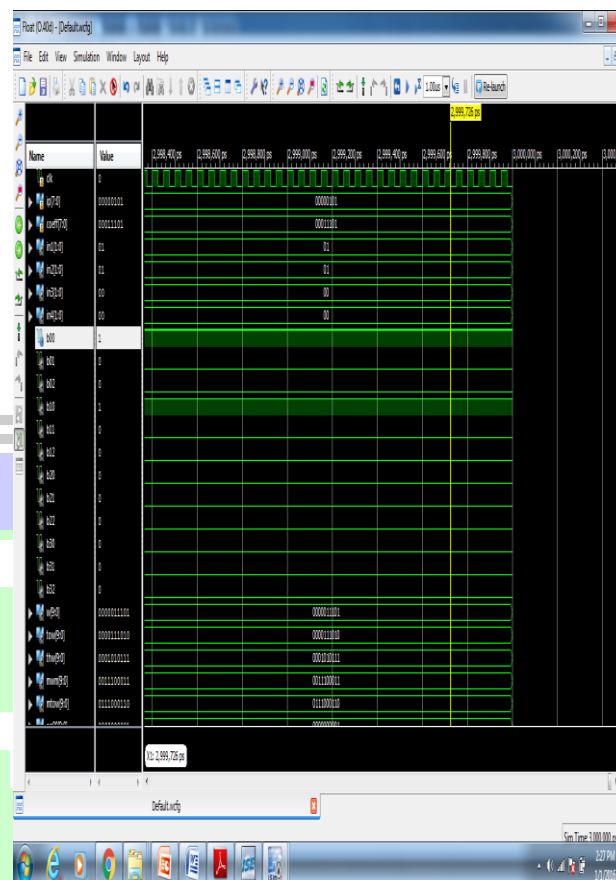


OR cells are depicted by above figure (b) and (c), respectively. Each AND cell takes an n -bit input D and a single bit input b , and consists of n AND gates. It distributes all the n bits of input D to its n AND gates as one of the inputs. The other inputs of all the n AND gates are fed with the single-bit input b . As shown in (c), each OR cell similarly takes a

pair of n -bit input words and has n OR gates. A pair of bits in the same bit position in B and D is fed to the same OR gate. The output of an AOC is w , $2w$, and $3w$ corresponding to the decimal values 1, 2, and 3 of the 2-b input ($u1u0$), respectively. The decoder along with the AOC performs a multiplication of input operand w with a 2-b digit ($u1u0$), such that the PPG of above figure performs $L/2$ parallel multiplications of input word w with a 2-b digit to produce $L/2$ partial products of the product word wu .

Structure of Adder Tree: Conventionally, we should have performed the shift-add operation on the partial products of each PPG separately to obtain the product value and then added all the N product values to compute the desired inner product. However, the shift-add operation to obtain the product value increases the word length, and consequently increases the adder size of $N - 1$ additions of the product values. To avoid such increase in word size of the adders, we add all the N partial products of the same place value from all the N PPGs by one adder tree. All the $L/2$ partial products generated by each of the N PPGs are thus added by $(L/2)$ binary adder trees. The outputs of the $L/2$ adder trees are then added by a shift-add tree according to their place values. Each of the binary adder trees require $\log_2 N$ stages of adders to add N partial product, and the shift-add tree requires $\log_2 L - 1$ stages of adders to add $L/2$ output of $L/2$ binary adder trees.² The addition scheme for the error-computation block for a four-tap filter and input word size $L = 8$ is shown in below figure. For $N = 4$ and $L = 8$, the adder network requires four binary adder trees of two stages each and a two-stage shift-add tree. In this figure, we have shown all possible locations of pipeline latches by dashed lines, to reduce the critical path to one addition time. If we introduce pipeline latches after every addition, it would require $L(N - 1)/2 + L/2 - 1$ latches in $\log_2 N + \log_2 L - 1$ stages, which would lead to a high adaptation delay and introduce a large overhead of area and power consumption for large values of N and L . On the other hand, some of those pipeline latches are redundant in the sense that they are not required to maintain a critical path of one addition time. The final adder in the shift-add tree contributes to the maximum delay to the critical path. Based on that observation, we have identified the pipeline latches that do not contribute significantly to the critical path and could exclude those without any noticeable increase of the critical path.

RESULT:



CONCLUSION:

We proposed an area-delay-power efficient low adaptation delay architecture for fixed-point implementation of LMS adaptive filter. So we used a novel PPG for efficient implementation of general multiplications and inner product computation by common sub expression sharing. We have proposed an efficient addition scheme for inner product computation to reduce the adaptation delay significantly in order to achieve faster convergence performance and to reduce the critical path. We proposed a strategy for optimized balanced pipelining across the time-consuming blocks of the structure to reduce the adaptation delay and power consumption. The proposed structure involved significantly less adaptation delay and provided significant saving of ADP and EDP compared to the existing structures. We proposed a fixed-point implementation of the proposed architecture, and derived the expression for steady-state error. We also discussed a pruning scheme that provides nearly 25% saving in the ADP and 10% saving in EDP over the proposed structure before pruning, without a noticeable degradation of steady-state error performance. The

adaptive filter is required to be operated at a lower sampling rate, with a clock slower than the maximum usable frequency and a lower operating voltage to reduce the power consumption.

REFERENCES :

- [1] S. Haykin and B. Widrow, Least-Mean-Square Adaptive Filters. Hoboken, NJ, USA: Wiley, 2003
- [2] S. Ramanathan and V. Visvanathan, "A systolic architecture for LMS adaptive filtering with minimal adaptation delay," in Proc. Int. Conf. Very Large Scale Integr. (VLSI) Design, Jan. 1996, pp. 286–289.
- [3] Y. Yi, R. Woods, L.-K. Ting, and C. F. N. Cowan, "High speed FPGA-based implementations of delayed-LMS filters," J. Very Large Scale Integr. (VLSI) Signal Process., vol. 39, nos. 1–2, pp. 113–131, Jan. 2005. [4] L. D. Van and W. S. Feng, "An efficient systolic architecture for the DLMS adaptive filter and its applications," IEEE Trans. Circuits Syst. II, Analog Digital Signal Process., vol. 48, no. 4, pp. 359–366, Apr. 2001.
- [5] FPGA Implementation of an Adaptive Noise Canceller ,Tian Lan1, Jinlin Zhang ,IEEE.
- [6] Digit-Serial Architecture For VLSI Implementation of Delayed LMS FIR Adaptive Filters. Basant Kumar Mohanty

