ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

FPGA Implementation for Optimized Adaptive Filter Based on Distributed Arithmetic

Sangeetha B[1], Anand Kumar[2], Manikandan B[3]

PG Scholar, of ECE, Sri Venkateswara College of Engineering and Technology, Thiruvallur, India [1]

Assistant Professor, Department of ECE, Sri Venkateswara College of Engineering and Technology, Thiruvallur, India[2,3]

ABSTRACT: A novel pipelined architecture implementation of adaptive filter based on Distributed arithmetic (DA) for low-power, high-throughput, and low area. Filtering operations requires larger area and is not suited for higher order filters therefore causes reduction in the throughput. These problems have been overcome by efficient distributed formulation of Adaptive filters. Distributed arithmetic is an efficient procedure for computing inner products between a fixed and a variable data vector. Equivalen implementation of four-point inner product and weight increments unit to produce high throughput rate. Conditional signed carry-save accumulation is used in order to reduce the sampling period and area complexity for DA-basedinner-product computation. Power Is reduced by using fast bit clock for carry-save accumulation but a much slower clock for all other operations. To update the weights by using least mean square (LMS) adaptation and also minimize the mean square error between the estimated and desired output. It reduce the LUT's, occupied slices, gate count for design.

KEYWORDS: Adaptive filter, circuit optimization, distributed arithmetic (DA), least mean square (LMS) algorithm.

I. INTRODUCTION

Adaptive digital filters find wide application in several digital signal processing (DSP) areas, e.g., noise and echo cancellation, system identification,channel estimation, channel equalization, etc. filter plays a major sole for removal of unwanted signal/noise from the original signal, particularly, Adaptive FIR filter is simple to attract for many applications, where it is need to minimize computational requirement. In Adaptive filter the signal and/or noise characteristics are often non stationary and the statistical parameters vary with time. An adaptive filter has an adaptation algorithm, that is meant to monitor the environment and vary the Filter transfer function accordingly. It consists of two process. First one is filtering process and second one is Adaptation process. Adaptive filter algorithms are Least Mean Square(LMS), Recursive Least Square(RLS), Normalized Least Mean Square(RLMS). The LMS algorithm are a class of adaptive filter to mimic a desired filter by finding the filter coefficients that relate to producing the least mean squares of the error signal. The RLS algorithm involve more complicated mathematical operations and require more computational resources than LMS. The NLMS algorithm is time varying step size algorithm and it is used in system in order to improve voice quality.Finally LMS algorithm is best suitable for reducing the power and area. LMS algorithms are widely used algorithm in adaptive filtering. The main features that attracted the use of the LMS algorithm are low computational complexity. Involves a feedback connection, although LMS might seem very difficult to work due to the randomness, the feedback acts as a low-pass filter or performs averaging so that the randomness can be filtered-out. The main advantage of the LMS algorithm is its simplicity in terms of the memory requirement.DA is basically a bit-serial computational operation that forms an inner (dot) product of a pair of vectors in single direct step. DA efficiently implements the MAC using basic building blocks (Look-Up Tables) in FPGAs. One of the vector operands is fixed, as in a digital filter, distributed arithmetic preprocesses the stored data to reduce the computational complexity of the inner product calculation. By using DA instead of traditional way, a huge reduction in area can be happened. Filtering is one of the most widely used complex signal processing operations. Least Mean Square (LMS) algorithm is the most popularly used adaptive filter not only due to its simplicity but also due to its

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

satisfactory convergence performance. DA is one way to implement convolution multiplier-less, where the MAC operations are replaced by a series of LUT accesses and summations. Techniques, such as ROM decomposition and offset-binary coding (OBC), can reduce the LUT size. Large-Scale systems design in short, due to the advent of VLSI. The number of applications of integrated circuits in high-performance computing telecommunications and consumer electronics has been rising steadily and at a very fast space. One of the most important characteristics of information services is their increasing need for very high processing power and bandwidth. Typically the required computational power of these applications is the driving force for the fast development of this field. As more and more complex functions are required in various data processing and telecommunications devices the need to integrate these functions in a small system package is also increasing.As a result their design complexity is considered much higher than that of memory chips. Sophisticated computer aided design tools and methodologies are developed and applied in order to manage the rapidly increasing design complexity.

II. ADAPTIVE FILTER BASED ON DA

LMS based adaptive filters are preferred for most of the DSP applications. The goal of the adaptation is to adjust the characteristics of the filter through an interaction with the environment in order to reach the desired values. The operation of adaptive filters is based on the estimation of the statistical properties of the signal in its environment, while modifying the value of its parameters in order to minimize a certain criterion function. The criterion function may be determined in a number of ways, depending on the particular purpose of the adaptive filter, but usually it is a function of some reference signal. The reference signal may be defined as the desired response of the adaptive filter, and in the role of the adaptive algorithm is to adjust the parameters of the adaptive filter in such a way to minimize the error signal, which represents the difference between the signal at the output of the adaptive filter and the reference signal. When doing the direct form configuration of the filters it leads to long critical path because of the inner product computation to get the filter output. Hence for high sampling rate, the critical path of structure should not exceed the sampling period. DA is basically a bit serial computational operation that forms an inner (dot) product of a pair of vectors in a single direct step. The advantage of DA is its efficiency of mechanization. In the multiplier-less distributed arithmetic (DA)-based technique has gained substantial popularity for its high-throughput processing, which result in less cost and area efficient computing structure. This brief proposes a novel DA-based architecture for low-power, low-area and high throughput pipelined implementation of adaptive filter with very low adaptation delay. Conventional adder-based shift accumulation is replaced by a conditional carry-save accumulation of signed partial linear products to reduce the sampling period. Finite-impulse-response (FIR) filters are basic processing elements in applications such as video signal processing and audio signal processing. Adaptive digital filters have been applied to a wide variety of important problems in recent years. Perhaps one of the most well known adaptive algorithms is the (LMS) algorithm, which updates the weights of a transversal filter using an approximate technique of steepest descent Many applications in digital communication (channel equalization, frequency channelization), speech processing (adaptive noise cancelation), seismic signal processing (noise elimination), and several other areas of signal processing require large order FIR filters. Since the number of multiply-accumulate (MAC) operations required per filter output increases linearly with the filter order, real-time Implementation of these filters of large orders is a challenging task.

Filters are used in signal processing applications weights of the filter in Adaptive filters, which significantly like channel equalization, interference, echo cancellation reduces the overall area of the adaptive filter. Influence of system identification and noise cancellation, etc. dividing the larger LUT into smaller LUTs was introduced Filter output is the weighted sum of the past and , but additional adders were required to combine the present input samples, which is realized through the smaller LUTs; which increases the dynamic power of the Multiply and Accumulate (MAC) unit in general DSP design. But the MAC units consume more area for multipliers Other than the DA based architectures common to provide better system performance and it leads to common sub-expression elimination (CSE) methods were also used high system cost.

III. LITERATURE REVIEW

Jesmin Joy. “Adaptive Fir Filter with Optimized Area and Power using Modified Inner-Product Block” International Journal of Innovative Research in Computerand Communication Engineering ,Vol. 3, Issue 9, An adaptive filter is a digital filter that can adjust its coefficients to give the best match to a given desired signal. When an

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

adaptive filter operates in a changeable environment the filter coefficients can adapt in response to changes in the applied input signals. The choice of the filter structure and the criterion function used during the adaptation process, have the crucial influence to the characteristics of the adaptive Filter as a whole. A new design and implementation of FIR filters using Distributed Arithmetic. Distributed Arithmetic structure is used to increase the resource usage while pipeline structure is also used to increase the system speed. The memory size can be reduced by decomposing the LUT. FIR filter is designed using multiplexer which is used to select the filter coefficients. The techniques used to provide reduction in LUT size compared with the conventional Look up Table (LUT) of adaptive FIR filter. By the proposed method, area efficiency, low power and high throughput is achieved.

Doss.B, Soundararajan.K, Narasimha.Y, Murthy. "Low-Power and Low-Area Adaptive FIR Filter Based on DA Using FPGA" International Journal of scientific research and management (IJSRM)Volume 3,Issue,2015.This presents an innovative pipelinedarchitecture for the implementation of adaptive filterbased on distributed arithmetic (DA) with low-area, low-power. The high throughput rate of the proposed design is achieved by updating the lookup table simultaneously and parallel implementation of filtering and weight-update operations. In order to reduce the sampling period and area complexity, the proposed method uses conditional signed carry-save accumulation for the purpose of DA-based inner-product computation in the place of conventional adder-based shift accumulation. In order to reduce the power consumption, the proposed design uses a faster bit clock for carry-save accumulation but it uses a much slower clock for the remaining operations. The proposed design involves the same number of multiplexers but a smaller LUT and the number of adders used reduces to half when compared to the existing DA-based design.Murali.L, Chitra.D and Manigandan.T, "Low Power Distributed Arithmetic Based Fir Filter"Middle-East Journal of Scientific Research-2014, IDOSI Publications, 2014 Data path architectures are the critical components in computational intense applications and their architectural changes leads to changes in VLSI design constraints like area, performance and power. And the modern automated world, the power constraint has been the major requirement; hence an effort has been applied regarding the necessary. This brief implements the Low power Finite Impulse Response (FIR) filter for the Digital signal processors (DSP) applications. Since it's a data path arithmetic architectural change, the proposed architecture can be applied to any hierarchical architecture where power is the major constraint. Designs were developed and modeled with Verilog HDL and synthesized using Cadence RTL compiler by mapping to TSMC's 65nm technology. The proposed arithmetic has reduced the filter power by 10.62 % when benchmarked with standard ASIC design methodology. Usha.M ,Ramadoss.R , Scholar.P.G "An Efficient Adaptive Fir Filter Based On Distributed Arithmetic" International Journal of Engineering Science Invention ISSN (Online). Volume 3, Issue 4 April 2014 PP.15-20. Adaptive filtering constitutes an important class of DSP algorithms employed in several hand held mobile devices for applications such as echo cancellation, signal de-noising, and channel equalization. The throughput of the proposed design is increased by parallel lookup table (LUT) update. The 16:1 multiplexer is replace by a 8:1 and 2:1 MUX. The conventional adder-based shift accumulation for DA-based inner-product computation is replaced by conditional signed carry-save accumulation in order to reduce the area complexity; the power consumption of the proposed design is reduced by using a fast bit clock for all operations. It involves thesame number of multiplexors, smaller LUT, and nearly half the numberof adders compared to the existing DA-based design. The proposed architecture is found to involve significantly 29% less area-13% less power and throughput compared with the existing DA-based implementations of FIR filter and a increase in operating frequency of 12MHZ is achieved.

IV. DISTRIBUTED ARITHMETIC (DA)

Distributed arithmetic (DA) is commonly used for signal processing algorithms where computing the inner product of two vectors comprises most of the computational work load. This type of computing profile describes a large portion of signal processing algorithms, so the potential usage of distributed arithmetic is tremendous. The inner product is commonly computed using multipliers and adders. Distributed Arithmetic, along with Modulo Arithmetic, are computation algorithms that perform multiplication with look-up table based schemes. Both stirred some interest over two decades ago but have languished ever since. Indeed, DA specifically targets the sum of products (sometimes referred to as the vector dot product) computation that covers many of the important DSP filtering and frequency transforming functions. Ironically, many DSP designers have never heard of this algorithm. Inspired by the potential of the Xilinx FPGA look-up table architecture, the DA algorithm was resurrected in the early 90's and shown to produce very efficient filter designs. The derivation of the DA algorithm is extremely simple but its applications are extremely broad. The mathematics includes a mix of Boolean and ordinary algebra and require no prior preparation - even for the

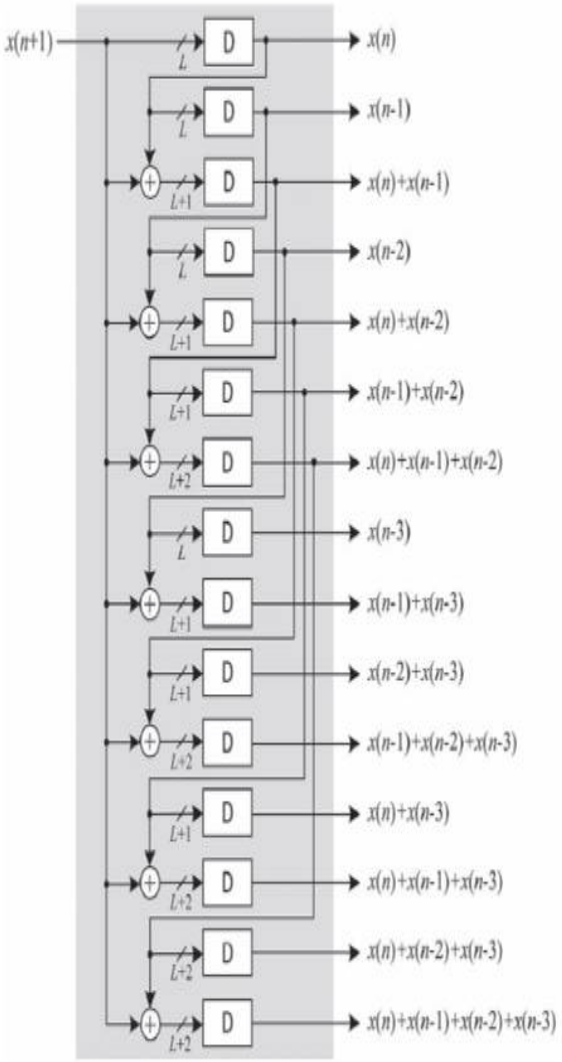
ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

logic designer. DA significantly reduces the number of additions needed for filtering. This reduction is particularly noticeable for filters with high bit precision. This reduction in the computational workload is a result of storing the pre-computed partial sums of the filter co-efficient in the memory table. When compared with other alternatives, distributed arithmetic requires fewer arithmetic computing resources and no multipliers.This aspect of distributed arithmetic is a favorable one for computing environments with limited computational resources, especially multipliers. These type of computing environments can be found on older field-programmable gate arrays (FPGAs) and low-end, low-cost FPGAs



Note that cK for $0 \leq k \leq 2N - 1$ can be pre computed and stored in RAM-based LUT of 2N words. However, instead of storing 2N words in LUT, store $(2N - 1)$ words in a DA table of $2N - 1$ registers. An example of such a DA table for $N = 4$ is shown in Fig 3.1. It contains only 15 registers to store the pre computed sums of input words. Seven new values of ck are computed by seven adders in parallel Fig 3.2. AK is the include in Sum when Xkb =1 that is the least significant bit. The content of the kth LUT location can be expressed as kj.kj where kj is the (j + 1)th bit of N-bit binary representation of integer k for $0 \leq k \leq 2N - 1$. A 4-tap (K = 4) implementation of the DA filter is shown in Figure 3.1.The contains all 16 possible combination sums of the filter weights w0;w1;w2; and;w3.The bank of shift registers in Figure 3.1 stores 4 consecutive input samples (x[n - i]; i =0……..3). The concatenation of rightmost bits of the shift registers becomes the address of the memory table. The shift register is shifted right at every clock cycle.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

The arithmetic sum of products that defines the response of linear, time-invariant networks can be expressed as:

$$y(n) = \sum_{k=1}^{K} A_k \, x_k(n)$$

where:
y(n) = response of network at time n.
$x_k(n)$ = k^{th} input variable at time n.
A_k = weighting factor of k^{th} input variable that is constant for all n, and so it remains time-invariant.

The constant factors, A_k, need not be so restricted, nor are they required to match the data word length, as is the case for the microprocessor. The constants may have a mixed integer and fractional format; they need not be defined at this time. The variable, x_k, may be written in the fractional format as shown in

$$x_k = -x_{k0} + \sum_{b=1}^{B-1} x_{kb} \, 2^{-b}$$

where$x_{kb}$is a binary variable and can assume only values of 0 and 1. A sign bit of
value -1 is indicated by xk0. Note that the time index, n, has been dropped since it is not needed to continue the derivation. The final result is obtained by first substituting equ.2 into equ.1. K $_{B-1}$

$$y = \sum_{k=1}^{K} A_k \left[-x_{k0} + \sum_{b=1}^{B-1} x_{kb} \, 2^{-b} \right] = \sum_{k=1}^{K} x_{k0} \bullet A_k + \sum_{k=1}^{K} \sum_{b=1}^{B-1} x_{kb} \bullet A_k \, 2^{-b}$$

and then explicitly expressing all the product terms under the summation symbols:

$$\begin{aligned} y = -[&x_{10} \bullet A_1 + x_{20} \bullet A_2 + x_{30} \bullet A_3 + \ldots + x_{K0} \bullet A_K] \\ &+ [x_{11} \bullet A_1 + x_{21} \bullet A_2 + x_{31} \bullet A_3 + \ldots + x_{K1} \bullet A_K] \, 2^{-1} \\ &+ [x_{12} \bullet A_1 + x_{22} \bullet A_2 + x_{32} \bullet A_3 + \ldots + x_{K2} \bullet A_K] \, 2^{-2} \\ &\bullet \\ &\bullet \\ &\bullet \\ &+ [x_{1(B-2)} \bullet A_1 + x_{2(B-2)} \bullet A_2 + x_{3(B-2)} \bullet A_3 + \ldots + x_{K(B-2)} \bullet A_K] \, 2^{-(B-2)} \\ &+ [x_{1(B-1)} \bullet A_1 + x_{2(B-1)} \bullet A_2 + x_{3(B-1)} \bullet A_3 + \ldots + x_{K(B-1)} \bullet A_K] \, 2^{-(B-1)} \end{aligned}$$

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

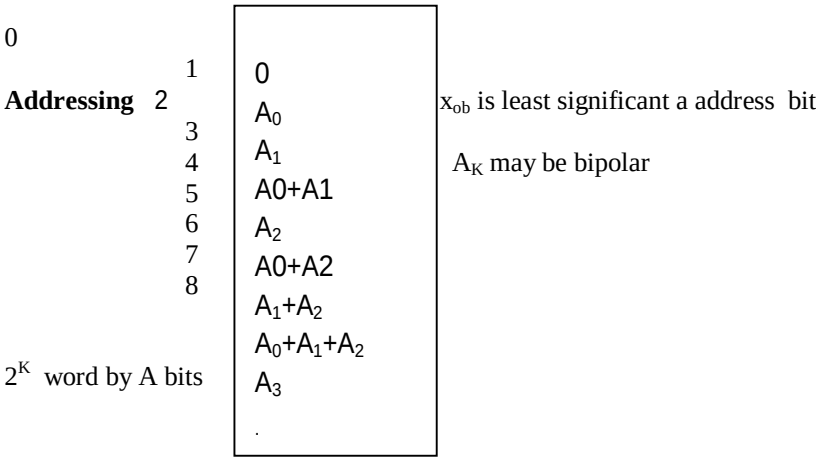
(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

Each term within the brackets denotes a binary AND operation involving a bit of the input variable and all the bits of the constant. The plus signs denote arithmetic sum operations. The exponential factors denote the scaled contributions of the bracketed pairs to the total sum and will henceforth be referred to as a Distributed Arithmetic look-up table or DALUT. Fig 3.2 Now construct a look-up table that can be addressed by the same scaled bit of all the input variables and can access the sum of the terms within each pair of brackets.

Such a table is shown in Fig 3.2 and will henceforth be referred to as a Distributed Arithmetic look-up table or DALUT. The same DALUT can be time-shared in a serially organized computation or can be replicated B times for a parallel computation scheme, as described later.

DULT content



DLUT Addressing

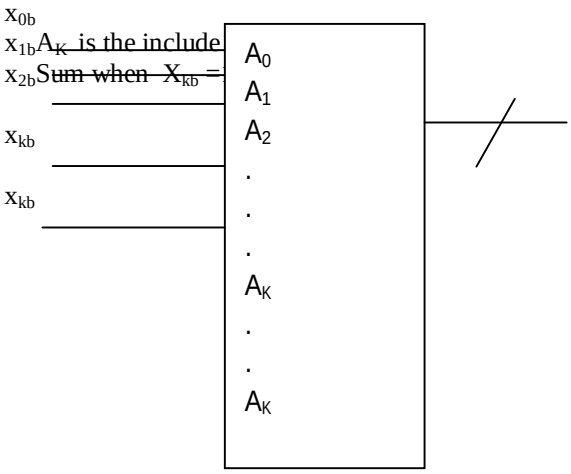


Fig 3.2 The distributed arithmetic look-up table

ADAPTIVE FILTERS USING DISTRIBUTED ARITHMETIC

Adaptive filtering is extensively used in several signal processing applications including acoustic echo cancellation, signal de-noising, sonar signal processing, clutter rejection in radars, and channel equalization for communication and

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

networking systems. For adaptive filtering applications, distributed arithmetic has not worked well for the following issues. First, the typical computational flow of distributed arithmetic for adaptive filtering requires a significant increase in the computational workload over the non-adaptive case and a noticeable increase in the computational time when constrained with limited computing resources.

These additional resources are needed for updating the contents of the memory table associated with distributed arithmetic. For these applications, one of the typical advantages of distributed arithmetic that is the low computing requirement is significantly diminished or eliminated Several attempts have been done to accelerate the process of updating the memory. Although these approaches do reduce the amount of processing necessary to update the memory, this reduction is gained at the expense of additional memory usage and of convergence speed. To address these issues, a new type of adaptive distributed arithmetic filter is proposed.

APPLICATIONS OF DISTRIBUTED ARITHMETIC

Mixed-Signal DA filters the described so far have used digital components. Filters can also be implemented using analog components. Typically, analog circuits use less power and less chip area than their digital counterparts for low precision computations. However, even with these advantages, the digital components are often preferred over the analog ones because of the ease of reprogram ability of the digital systems. A common method to achieve reprogram ability for analog systems is to utilize a bank of components to emulate the variability of key components such resistors or capacitors. A straight forward example of constructing a variable capacitor is to use a bank of capacitors whose size is scaled by factors of two and connected in parallel through a bank of switches. This type of variable component is reprogrammable using digital words.

ADAPTIVE FILTERS

Most adaptive filters are dynamic filters which change their characteristics to achieve desired output. Adaptive filter has various parameters which need to be changed in order to maintain optimal filtering. The two types algorithms for adaptive filtering i.e. Least Mean Square (LMS) and Recursive Least Squares (RLS) optimal filtering. Noise cancellation is one of the applications of adaptive filter.

Adaptive filter have two important properties: first, it can work effectively in unknown environment; second, it is used to track the input signal of time-varying characteristics. The DSP has serial architecture, so it can't process high sampling rate applications. It can use only for extremely complex math-intensive tasks .An example of system identification is shown in Fig. 3.3. The goal of the adaptive algorithm is to adjust the coefficient of the adaptive filter, w[n], in order to minimize the error term $e[n]$ in the mean-squared sense. The adaptive algorithm essentially identifies a vector of coefficient w[n] that minimizes the following quadratic equation,

$$\varepsilon[n] = E\{|e[n]|^2$$

where $e[n] = d[n]$ □$y[n]$, $d[n]$ is the output from the unknown system (desired signal), $y[n]$ is the output from the adaptive filter, the expected value, and _[n] is the mean squared error (MSE). Many methods exist to solve Eqn. 3.3, the most common being the method of steepest descent. This process continues until the adaptive algorithm reaches steady state, where the deference between the current solution vector and the optimal solution, or MSE, is at its minimum. In general, adaptive algorithms can be divided into four steps that are performed sequentially: filtering, computing the error, calculating the coefficient
updates, and updating the coefficients. When split into this form, the primary deference prediction and interference canceling.

Adaptive filtering comprises two basic operations: the filtering process and the adaptation process. In the filtering process an output signal is generated from an input data signal using a digital filter, whereas the adaptation process consists of an algorithm which adjusts the coefficients of the filter to minimize a desired cost function. We first classify the adaptive filters into two main categories:

- The Adaptive Finite Impulse Response (AFIR) and
- The Adaptive Infinite Impulse Response (AIIR) filters.

There are numerous methods for the performing weight update of an adaptive filter. There is the wiener filter, which is the optimum liner filter in terms of mean squared error, and several algorithms that attempt to approximate it,

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

such as the method of steepest descent. There is also least-mean square algorithm, he most commonly used adaptive algorithm is

- LMS(least mean square) and
- RLS (Recursive least square) algorithm.

LEAST-MEAN-SQUARE (LMS) ALGORITHM

LMS algorithms are used to maintain a specified system power budget as well as to limit the maximum output level to prevent overdriving the transducer. The Least-mean-square (LMS) algorithm is same as the method of steepest-descent in that weights are adapted by repeatedly approaching the MSE(Mean Square Error) minimum.

Least Square Algorithm, At Every Iteration N do:

1.Form the input vector $x(n)=[x(n).x(n-1)\ldots\ldots x(n-N+1)]^t$ form the inpute

2.Compute the output of the adaptive filter: $y(n)=x^t(n)h(n)=h^t(n)x(n)$;

3.Compute the output error: $e(n)=d(n)-y(n)$;

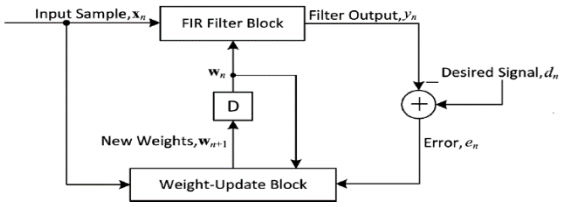
4. update the coefficient of the adaptive filter: $h(n+1)=h(n)+\mu_e(n)x(n)$;

APPLICATION

- Noise cancellation
- Signal prediction
- Adaptive feedback
- Echo cancellation

LEAST MEAN SQUARE ALGORITHM

LMS algorithm is introduced. LMS stands for Least-Mean-Square. This algorithm was developed by Bernard Widrow in the 1960s, and is the first widely used adaptive algorithm. It is still widely used in adaptive digital signal processing and adaptive antenna arrays, primarily because of its simplicity, ease of implementation and good convergence properties.



The goal of the LMS algorithm is to produce the mean square error for the given environment and weights that minimize the mean-squared error between a desired signal and the arrays output, loosely speaking, it tries to maximize reception in the direction of the desired signal (who or what the array is trying to communicate with) and minimize reception from the interfering or undesirable signals in the Fig 4.1. some information is needed before optimal weights can be determined.

The weights of LMS adaptive filter during the nth iteration are updated according to the following equation. The updated weight is

$$\mathbf{w_{n+1}= w_n+ \mu e_n x_n}$$

$\mathbf{e_n = d_n - y_n}$

$\mathbf{y_n = w^T_n x}$

Where $\mathbf{d_n}$ is the desired response, $\mathbf{e_n}$ is the error computed during nth iteration, $\boldsymbol{\mu}$is convergence factor or step size. The weight vector W_n and input vector $x_n$is given by

$$\mathbf{x}_n = [x_n, x_{n-1}, \ldots, x_{n-N+1}]^T$$
$$\mathbf{w}_n = [w_n(0), w_n(1), \ldots, w_n(N-1)]^T$$

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

Where
N is the number of weights used in the LMS adaptive filter. It is well known that the LMS algorithm has a slow convergence for correlated inputs.

Another situation that can appear in identification applications is when the coefficients of the model are time-varying. The adaptive algorithm should provide a mechanism to track the changes of the model. Fig 4.1 in the LMS algorithm described has a very low computational complexity (number of additions, subtractions, divisions, multiplications per iteration) and memory load, which makes it very attractive for practical implementations. It is well known that the step-size μ influences the performances of the adaptive filter. Despite its low complexity, the LMS has also some drawback.

Several adaptive algorithms have expanded upon ideas used in the original LMS algorithm. Most of these algorithms seek to produce improved convergence properties at the expense of increased computational complexity.

The recursive least-square (RLS) algorithm seeks to minimize the MSE just as in the LMS algorithm. Typically, after each sample, the coefficients of the FIR filter are adjusted as follows (Widrow). For μ is called the convergence factor.

$$l = 0 \ldots L$$
$$w_{l,k+1} = w_{lk} + 2\mu\, \epsilon_k\, x_{k-l},$$

The LMS algorithm does not require that the X values have any particular relationship; there for it can be used to adapt a linear combiner as well as an FIR filter. In this case the update formula is written as:

$$w_{l,k+1} = w_{lk} + 2\mu\, \epsilon_k\, x_{lk}$$

The effect of the LMS algorithm is at each time, k, to make a small change in each weight. The direction of the change is such that it would decrease the error if it had been applied at time k. The magnitude of the change in each weight depends on μ, the associated X value and the error at time k. The weights making the largest contribution to the output, y_k, are changed the most. If the error is zero, then there should be no change in the weights. If the associated value of X is zero, then changing the weight makes no difference, so it is not changed.

The DLMS adaptive FIR filters need more registers than LMS adaptive FIR filters for providing the delay lines. Assuming that the error-computation path is implemented in *m* pipelined stages, the latency of error computation is m cycles so that the error computed by the structure at the nth cycle is e_{n-m} and is used with the input samples delayed by *m* cycles to generate the weight-increment term. The weight-update equation of the DLMS algorithm in Fig 4.2 is given by

$$lw_{n+1} = w_n + \mu\, e_{n-m}\, x_{n-m}$$

$e_{n-m} = d_{n-m} - y_{n-m}$
$y_{n-m} = w^T_n x_n$
and $\mathbf{d_{n-m}}$ is the desired response, $\mathbf{e_{n-m}}$ is the error computed during nth iteration

The number of delays corresponds to the *m* pipeline delays introduced due to pipelining of the error-computation block. A transpose form LMS adaptive filter is suggested in, where the filter output at any instant depends on the delayed version of weights and the number of delays.

Since all weights are updated concurrently in every cycle to compute the output according to direct-form realization of the FIR filter is a natural candidate for implementation. However, the direct-form LMS adaptive filter is often believed to have a long critical path due to an inner product computation to obtain the filter output.

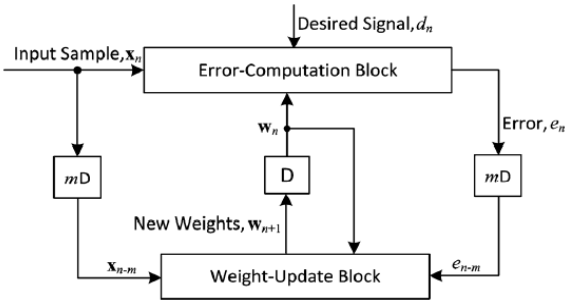
This is mainly based on the assumption that an arithmetic operation starts only after the complete input operand words are available/generated. In the existing literature on implementation of LMS adaptive filters, it is assumed that the addition in a multiply-addoperation. An area–delay-power efficient with low adaptation-delay architecture for fixed-point implementation of DLMS adaptive filter are achieved by using a novel PPG for efficient implementation of general multiplications and inner-product computation by common sub expression. Such an assumption could be valid when multipliers and adders are used as discrete components, which is not the case in ASIC and FPGA implementation these days. On the other hand, accordingly, the propagation delay for the multiply-add operation can be calculated.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016



The generalized DLMS is introduced in order to obtain an efficient LMS algorithm in VLSI circuits. It is well known that a pipelined architecture helps the efficient implementation of the LMS algorithm. The influence of these delays on the algorithm behavior is analyzed theoretically and by simulation. The proposed design is found to be more efficient in terms of power-delay Product and energy-delay-product compared with the existing structures.

ERROR COMPUTATION BLOCK

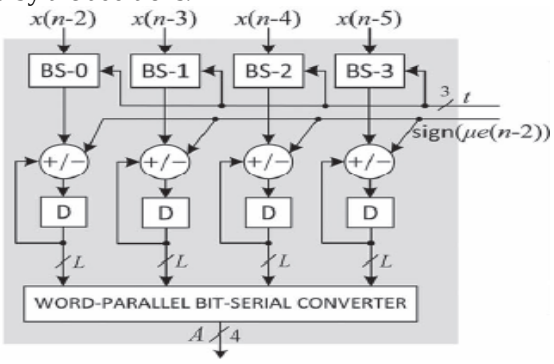
The structure for error-computation unit of an *N*-map DLMS adaptive filter. It the Fig 4.3 is consists of *N* number of 2-b partial product generators (PPG) corresponding to *N* multipliers and a cluster of *L*/2 binary adder trees, followed by a single shift–add tree.

Its consists of four barrel shifters and four adder/subtractor cells in the Fig.5.3. The barrel shifter shifts the different input values $x_k$for k =0, 1, . . .,N− 1 by appropriate number of locations (determined by the location of the most significant one in the estimated error). The barrel shifter yields the desired increments to be added with or subtracted from the current weights. The sign bit of the error is used as the control for adder/subtractor cells such that, when sign bit is zero or one, the barrel-shifter output is respectively added with or subtracted from the content of the corresponding current value in the weight register.

Realize the corresponding multiplication of a shift operation. The weight-update block consists of n carry-save units to update n weights. Each of those carry-save units performs the multiplication of shifted error values with the delayed input samples along with the addition with the old weights. Note that the addition of old weight with weight increment term is merged with multiplication pertaining to the calculation of weight-increment term.

The barrel shifter shifts the different input values $x_k$for k =0, 1, . . .,N− 1 by appropriate number of locations (determined by the location of the most significant one in the estimated error). The barrel shifter yields the desired increments to be added with or subtracted from the current weights.

The sign bit of the error is used as the control for adder/subtractor cells such that, when sign bit is zero or one, the barrel-shifter output is respectively added with or subtracted from the content of the corresponding current value in the weight register. Each of the MAC units therefore performs the multiplication of the shifted value of error with the delayed input samples x_{n-m} followed by the additions.



ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

The final outputs of the carry-save units constitute the updated weights which serve as an input to the error computation block as well as the weight-update block for the next iterations. A pair of delays are introduced before and after the final addition of the weight-update block to keep the critical-path equal to one addition time.

CONTROL WORD T FOR THE BARREL SHIFTER

Adder/subtractor Block is a digital circuit that is capable of adding or subtracting numbers. The circuit does the adding or subtracting process depending on a control signal. When Sign Bit = '0' the circuit perform addition .When Sign Bit = '1' the circuit perform subtraction. Word Parallel Bit Serial Converter is used to find convert parallel bit to serial. Control word t for the barrel shifter:

if r_6=1 then t="000";
else if r_5=1 then t="001";
else if r_4=1 then t="010";
else if r_3=1 then t="011";
else if r_2=1 then t="100";
else if r_1=1 then t="101";
else if r_0=1 then t="110";
else then t="111"

r=abs((n-2))

r_i:i^{th}bit of 7-bit word r

RTL SCHEMATIC OF MADCOMPLEXFIR FILTER

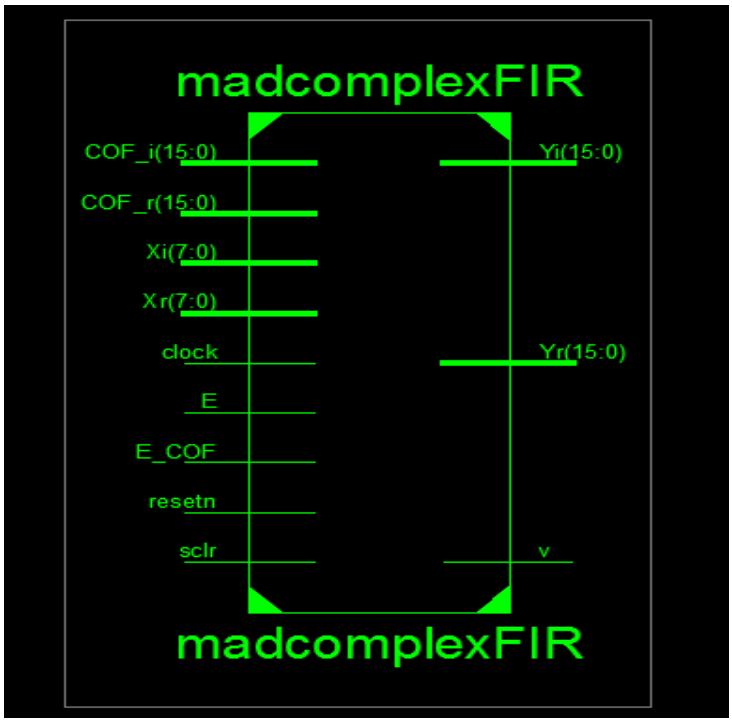


Fig 6.1 RTL Schematic mad complex FIR Filter

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

RTL Schematic of Mad complex FIR filter using adder based shift accumulation by cadence RTL compiler the power and area are estimated. The synthesis of Mad complex FIR filter was successfully carried out by using cadence RTL compiler.

SIMULATION RESULT OF MADCOMPLEXFIR

The existing DA based filter architecture.4-tap DA based Filter was designed as per the ASIC design methodology. The simulation result is obtained by using mad complex FIR filter. The inputs are given clock, reset, xr[7:0], xi[7:0]. The output is obtained of Y for y[15:0]. The power consumption is 3.422(w) and area delay product 1.54 is results of the mad complex fir filter in existing.

V. CONCLUSION

Pipelined architecture is used to achieve low-power, high-throughput, and low-area implementation of DA-based adaptive filter. Throughput rate is significantly enhanced by parallel LUT update and concurrent processing of filtering operation and weight-update operation. Adder based shift accumulation scheme of signed partial inner products for the computation of filter output obtained is 3.422. The offset binary coding is used to reduce the LUT size of area efficient implementation of DA (Distributed arithmetic).

Efficient pipelined architecture is used to achieve the high throughput implementationof DA-based adaptive filter. Throughput rate is significantly enhanced by parallel LUT update and concurrent processing implementation of four point inner product blocks and weight-update operation. A carry-save accumulation scheme is used in order to reduce the sampling period and area complexity. Conventional carry save accumulation method will improve the lower adaptation-delay, lower area and reduction of power when comparing with the adder based shift accumulation.

REFERENCES

[1] Allred.D.J, Yoo.H, Krishnan.H, Huang.W, and Anderson.D.V, "LMS adaptive filters using distributed arithmetic for high throughput," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 7, pp. 1327–1337, Jul. 2005.
[2] Guo.R and DeBrunner.L.S, "Two high-performance adaptive filter implementation schemes using distributed arithmetic," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 58, no. 9, pp. 600–604, Sep. 2011.
[3] Guo.R, and DeBrunner.D.S, "A novel adaptive filter implementation scheme using distributed arithmetic," in Proc. Asilomar Conf. Signals, Syst., Comput., Nov. 2011, pp. 160–164.
[4] Haykin.S and Widrow.B Least-Mean-Square Adaptive Filters. Hoboken, NJ, USA: Wiley, 2003.
[5] Haimi-Cohen.R, Herzberg.H, and Beery.H, ―Delayed adaptive LMS filtering: Current results,‖ in Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., Albuquerque, NM, Apr. 1990, pp. 1273–1276.
6] Haykin.S and Widrow.B Least-Mean-Square Adaptive Filters. Hoboken, NJ, USA: Wiley, 2003.
[7] Meher.P.K, and Park.S.Y, "High-throughput pipelined realization of adaptive FIR filter based on distributed arithmetic," in VLSI Symp. Tech. Dig., Oct. 2011, pp. 428–433.
[8] Meyer.M.D, and Agrawal.P, "A modular pipelined implementation of a delayed LMS transversal adaptive filter," in Proc. IEEE Int. Symp. Circuits Syst., May 1990, pp. 1943–1946.
[9] Mitra.S.K, ―Digital Signal Processing: A Computer–Base Approach‖, 2nd ed, New York: McGraw–Hill Companies, 2001.
[10] Poltmann.R.D, ―Conversion of the delayed LMS algorithm into the LMS algorithm,‖IEEE Signal Process. Lett., vol.2,p.223,Dec.1994.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 3, March 2016

[11] Prakash, M.S. and Shaik,R.A, 2013. Low-Area and High-Throughput Architecture for an Adaptive Filter Using Distributed Arithmetic, Circuits and Systems II: Express Briefs, IEEE Transactions on ,*(FPT), 2011 International Conference on*. IEEE.
[12] Paulo S.R. Diniz , Adaptive filtering Algorithms and Practical Implementations., ISBN 978-1-4614-4105
[13] Sang Yoon Park and Promod Kumar Meher, "Lowpower, High-Throughput, and Low-area Adaptive FIR Filter Based on Distributed Arithmetic," IEEE Trans. On Circuits and Systems-II, Express Briefs, Vol.60, no.6, pp.346-350, June 2013.
[14] Widrow.B,andStearns.S.D, Adaptive signal processing. Prentice Hall, Englewood Cliffs, NJ, 1985.
[15]White.S.A, "Applications of the distributed arithmetic to digital signal processing: A tutorial review," IEEE ASSP Mag., vol. 6, no. 3, pp. 4–19,