VLSI Architectures for Blind Equalization Based on Fractional-Order Statistics

V. Paliouras, J. Dagres, P. Tsakalides, and T. Stouraitis
Electrical & Computer Engineering Dept.,
University of Patras, GR 26 500 Patras, Greece.

Abstract

Four types of VLSI architectures for the hardware realization of the FLOS-CM algorithm are introduced in this paper. Each architecture is appropriate for a particular environment. The FLOS-CM algorithm is found to be amenable for implementation using logarithmic arithmetic. A logarithmic architecture is shown to require up to 50% less area and be 14% faster than a linear fixed-point arithmetic counterpart. In terms of Area×Time and Area×Time^2 complexities, the logarithmic architecture is up to 120% better.

1 Introduction

One of the most important issues in digital communications is the distortion of information bearing signals when transmitted through a physical channel. Distortion is induced due to the non-ideal frequency response of the channel, which can be unknown or time-variant, and it causes the intersymbol interference (ISI) problem. Further signal degradation is due to the presence of additive noise in the channel. To mitigate the ISI problem, receivers are equipped with equalizers.

In the past, a variety of adaptive linear equalizers have been proposed [1]. Most of them assume the availability of a training sequence. When the channel is time varying, a training sequence has to be sent periodically in order to allow the receiver to adapt to the new channel setting. The bandwidth loss due to the repeated transmission of a known sequence is typically too high. Therefore there is a need for receivers able to obtain the transmitted sequence of symbols without the use of a training sequence. An equalization procedure that makes no use of a training sequence, is called Blind Equalization (BE) [2].

The most frequently-used algorithm for BE is the constant modulus algorithm (CMA) [3]. CMA minimizes the cost function defined by the CM criterion, which penalizes deviations in the modulus (magnitude) of the equalized signal away from a fixed value, determined by the source constellation and its statistics. Most of the studies on these algorithms consider a noise-free channel, since ISI is claimed to be the major reason for signal degradation. In wired channels this is true, but in wireless communications the effect of additive noise is severe and cannot be neglected [4].

Recently it has been shown that the noise in certain environments is more accurately modeled as an alpha-stable process rather than as a Gaussian one [5]. BE in such a context can be performed by means of a recently introduced algorithm, the FLOS-CMA [6]. In this paper, four types of VLSI architectures for the implementation of the FLOS-CMA are introduced. Each of the proposed architectures is suitable for a particular environment depending on the p and q parameters of FLOS-CMA being constant or variable. Three of the proposed architectures are based on fixed-point arithmetic, while the fourth utilizes logarithmic arithmetic [7]. Sacha and Irwin report that logarithmic arithmetic leads to low-power architectures for the QRDRLS algorithm [8]. In this paper, it is shown that a logarithmic arithmetic-based architecture can be substantially more efficient than a linear implementation of the FLOS-CMA. The performance improvement achieved stems from both operator strength reduction and from the appropriate choice of the logarithmic base b and the logarithmic integer and fractional word lengths. The particular choice of b is utilized by an LNS adder/subtractor, also introduced in this paper.

2 Alpha-Stable Noise Model

The model commonly used to describe random noise is the Gaussian model. In practice, signal and noise sources do not always follow the Gaussian assumption. Non-Gaussianity often results in significant performance degradation for systems optimized under the Gaussian assumption, and if this degradation cannot be tolerated, a more realistic statistical model must be considered [5].

Non-Gaussian phenomena encountered in practice can be characterized as impulsive. Signals and noise in this class tend to produce large-amplitude excursions from the average value more frequently than Gaussian signals. As a result, their density functions decay in the tails less rapidly than the Gaussian density functions. A statistical model of heavy-tailed interference, based on the theory of alpha-stable processes, has been proposed by Nikias and Shao [5] to better describe these kind of phenomena. In particular it was shown that in wireless communications the first-order distribution of the interference amplitude follows a symmetric alpha stable (SaS) law. The class of symmetric alpha-

0-7803-7057-0/01/$10.00 ©2001 IEEE.

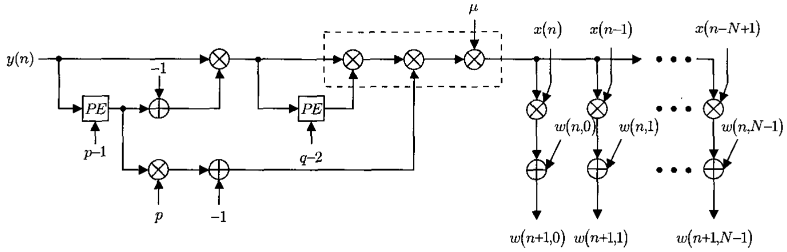


Figure 1: Type I architecture. The multipliers in the dashed rectangle can be rearranged into a tree to reduce the delay.

stable distributions can be described by its characteristic function as follows

$$\varphi(\omega) = \exp(j\lambda\omega - \gamma\,|\omega|^{\alpha}), \quad (1)$$

where α is the characteristic exponent restricted to the values $0 < \alpha \leq 2$, λ $(-\infty < \delta < \infty)$ is the location parameter, and γ $(\gamma > 0)$ is the dispersion of the distribution. The characteristic exponent α measures the "thickness" of the tails of the distribution. Dispersion γ is similar to the variance of the Gaussian distribution and equals half the variance in the Gaussian case, i.e., when $\alpha = 2$. It is known that for a non-Gaussian stable distribution with characteristic exponent α, only moments of order less than α are finite [5].

3 CMA and FLOS-CMA

A common choice for blind equalization is the CMA [1]. The original form of the cost function that the CMA is trying to minimize, is given by

$$J_{pq}^{\mathrm{CM}} = E\left[\left|\|y(n)\|^{p} - \delta\right|^{q}\right], \quad (2)$$

where $y(n)$ is the output of the equalizer, $E[\cdot]$ denotes statistical expectation, p and q are positive integers, and δ is a positive constant related to the constant modulus of the signal constellation. Assuming that $\mathbf{x}(n)$ denotes the receiving data vector

$$\mathbf{x}(n) = [x(n)\; x(n-1) \ldots x(n-N+1)]^{T} \quad (3)$$

and $\mathbf{w}(i)$ is the vector of adjustable coefficients

$$\mathbf{w}(i) = [w(i,0)\; w(i,1) \ldots w(i,N-1)]^{T}, \quad (4)$$

the equalizer output can be written as

$$y(n) = \mathbf{x}^{T}(n)\mathbf{w}(i). \quad (5)$$

Various adaptive equalizers can be obtained by choosing a (p,q)-variant of the cost function in (2) and an iterative technique to minimize it with respect to the filter coefficients. CMA uses p and q equal to two and attempts to minimize (2) by following the path of steepest descent. The adaptation equation is given by

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\mathbf{x}^{*}(n)e(n) \quad (6)$$

$$e(n) = y(n)\left(\|y(n)\|^{2} - 1\right). \quad (7)$$

As it can be seen from (6) and (7), CMA involves fourth-order moments of the measures. In the presence of heavily-tailed noise, the use of second or higher-order statistics in effect amplifies the noise. Rupi *et al.* [6] proposed a new criterion based on fractional-order moments that mitigates the noise and introduced a new FLOS-CM cost function

$$J_{p,q}^{\text{FLOS-CM}} = \frac{1}{2}E\left[\left|\|y(n)\|^{p-1}y(n) - \delta y(n)\right|^{q}\right]. \quad (8)$$

The pair (p,q) now takes fractional values between 0 and α, where α is the characteristic exponent of the alpha-stable distribution that best describes the additive noise component. To minimize (8), Rupi *et al.* used the following stochastic gradient descent algorithm

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\nabla_{w}J_{p,q}^{\text{FLOS-CM}}, \quad (9)$$

where μ is the step parameter and ∇_{w} is the gradient with respect to the equalizer taps w. The instantaneous estimate of the gradient leads to [6]

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\mathbf{x}^{*}(n)\zeta(n), \quad (10)$$

where

$$\zeta(n) = A(n)\,\|A(n)\|^{q-2}\left(p\,\|y(n)\|^{p-1} - 1\right) \quad (11)$$

$$A(n) = y(n)\left(\|y(n)\|^{p-1} - 1\right). \quad (12)$$

A major disadvantage of using the FLOS-CMA algorithm has been the complexity of hardware units that compute fractional powers. In the following section the particular hardware complexity is quantified and low-complexity VLSI architectures are introduced.

4 The Proposed Architectures

Four types of VLSI architectures are proposed, for the implementation of the FLOS-CMA defined by (10), (11), and (12). Each of the proposed architectures exhibits different performance characteristics to match particular channel characteristics and application idiosyncrasies.

Type I architecture, shown in Fig. 1, occurs by directly mapping each node of a dataflow graph (DFG) description of the FLOS-CMA to a dedicated hardware unit. The main feature of a Type I architecture, is its applicability to situations where p and q are variable. Several architectures can be obtained from the Type I architecture, by partitioning and scheduling the DFG [9].

The Type I architecture utilizes linear arithmetic. The architecture comprises two power evaluation (PE) units, for the computation of x^{y}, two adders with the constant -1, $5 + N$ multipliers, and N subtractors. x^{y} can be computed via $x^{y} = (e^{\ln x})^{y} = e^{y\ln x}$. [10]. Therefore, each of the PE units requires a look-up table for computing $\ln x$, a multiplication, and a LUT for the computation of e^{x}. Hence, due to power evaluation, the number of multiplications required for the FLOS-CMA, increases to $7 + N$, while four LUT operations are introduced.

In case $y(n)$ is of word length less than 14 bits, two different techniques can be employed for the derivation of $\zeta(n)$. .Both techniques require the storage of $\zeta(n)$ values for all possible inputs $y(n)$. In particular a Type II

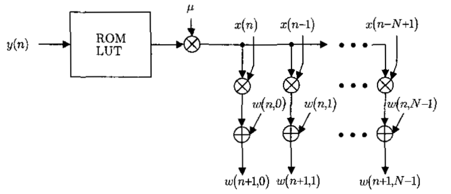


Figure 2: Type II structure: p and q are constant.

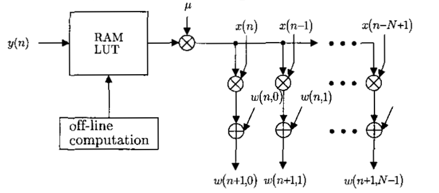


Figure 3: Type III architecture: Off-line computation of $\zeta(n)$ for particular p and q.

architecture, applicable in the case that both p and q are constant, replaces the computation of $\zeta(n)$ with a LUT operation, as depicted in Fig. 2.

A more versatile architecture can be obtained by replacing the ROM LUT with a RAM LUT, the contents of which as computed off-line, for example, during the initialization of the equalizer. This scheme is adopted by the Type III architecture, depicted in Fig. 3.

A different approach is shown in Fig. 4. The Type IV architecture utilizes the properties of the logarithmic arithmetic [7] or logarithmic number system (LNS) to reduce hardware complexity in the case of variable p and q. Since the FLOS-CMA requires several multiplications and the evaluation of x^y, where y is fractional, a logarithmic implementation becomes of interest, because it simplifies the particular operations. By replacing operands with their logarithmic images, multiplications are reduced to additions, while the operation of x^y, which is awkward in linear arithmetic, is reduced to a simple multiplication. Let X and Y be $X = \log_b x$ and $Y = \log_b y$, assuming for simplicity, $x,\ y > 0$ and b is the base of the logarithm. Then, $x^y = (b^{\log_b x})^y = b^{y \log_b x} = b^{yX}$. Hence, the sought logarithmic image of x^y, $P = \log_b(x^y)$, can be as $P = \log_b(x^y) = yX$. By exploiting the complexity reduction in the particular arithmetic operations and for the general case of variable p and q, a logarithmic architecture for the FLOS-CMA can be derived that executes two multiplications, N logarithmic subtractions, $4 + N$ two-operand additions, one addition with a constant, and two logarithmic subtractions with a constant. Notice that LNS addition and subtraction require the evaluation of $f_s(d) = \log_b(1 - b^{-d})$ and $f_a(d) = \log_b(1 + b^{-d})$ respectively. In this paper, the evaluation of $f_s(d)$ and $f_a(d)$ is performed as a LUT operation. A comparison of the Type I and Type IV architectures in terms of the operations required, is shown in Table I, where it has been assumed that a logarithmic subtraction is equivalent to two additions and a LUT operation. The complexity of the LNS operations depends on the world lengths allocated to represent the integral and the fractional part of the logarithm, expressed in number of bits as k and l respectively, as well as the often neglected base of the logarithm b [11]. In order to qualitatively compare the performance of the linear Type-I architecture to the logarithmic Type-IV architecture, extensive simulations have been carried out with the objective to determine an adequate linear fixed-point word length n and the corresponding LNS parameters k, l, and b, that guarantee equivalence performance to the fixed-point implementation. These simulations are exploited to measure the average MSE in the steady-state case due to quantization errors using the best choice of the algorithm's step parameter μ, with respect to the steady-state error.

Operations	Linear	Logarithmic
Multiplications	$7+N$	2
LUT	4	$N+2$
Add or subtract	N	$2N+4$
Add with constant	2	0

Table I: A comparison of the Type I and IV architectures.

The word lengths of the data values stored in the LUT for $f_a(d)$ and $f_s(d)$ can be minimized by selecting a logarithm base $b = 1.66833$ and $l = 7$ fractional bits. It can be noted that as the address increases, the word length of stored data diminishes. Therefore, a folding scheme can be adopted, according to which addresses in excess of 1024 are stored in the most significant part of words located at 512 and onwards. The particular scheme is employed in the introduced LNS adder/subtractor of Fig. 5. The most or least significant part of a stored word can be retrieved, using a multiplexer. Address is generated by a 2-input OR gate, used to assert the 9th bit ($2^9 = 512$) of the original address, when the 10th bit is asserted. The 10th address bit is also used to control the output multiplexers, shown in Fig. 5. Further savings in chip area are gained by using a common physical ROM component for both logarithmic addition and subtraction functions, as replication of address generation circuitry is avoided. Again, the least significant 10-bit part of an 18-bit word contains subtraction values, while the most significant 8-bit part contains addition values. The particular memory organization requires a 1024-word × 18 bits-per-word ROM, which fits into an area of 0.866mm^2 for a 0.7-μm CMOS technology [12], with an access time of 16.25ns. For comparison, an 11-bit multiplier occupies 0.625mm^2 and exhibits a delay of 25.52ns in the same technology. The complexities of the various building blocks for the linear and the logarithmic architecture are offered in Table II. The corresponding word lengths have been defined using the experimental procedure described earlier.

It is obtained that the LNS FLOS-CMA is of sufficiently low complexity to be considered for practical applications. Savings in terms of the percentage $s =$

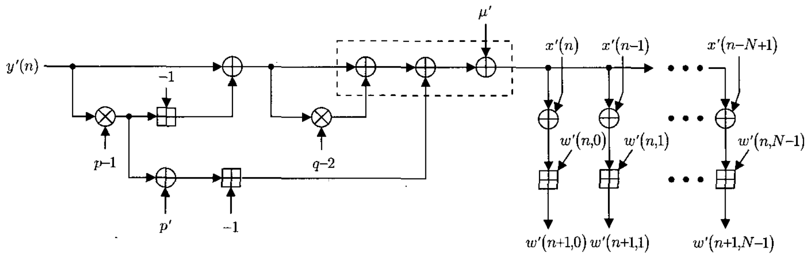


Figure 4: Type IV: The logarithmic architecture. Arranging the adders in the dashed rectangle as a tree reduces delay.

LNS		FXP	
a'_{mul}	0.777mm^2	a_{mul}	0.625mm^2
t'_{mul}	27.86ns	t_{mul}	25.52ns
a'_{LUT}	0.866mm^2	a_{LUT}	0.988mm^2
t'_{LUT}	16.25ns	t_{LUT}	14.80ns
a'_{add}	0.027mm^2	a_{add}	0.021mm^2
t'_{add}	13.19ns	t_{add}	11.37ns

Table II: Complexities of the basic building blocks.

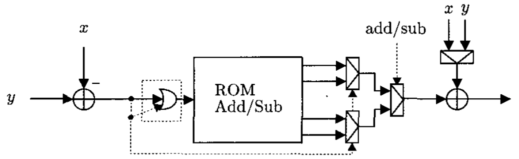
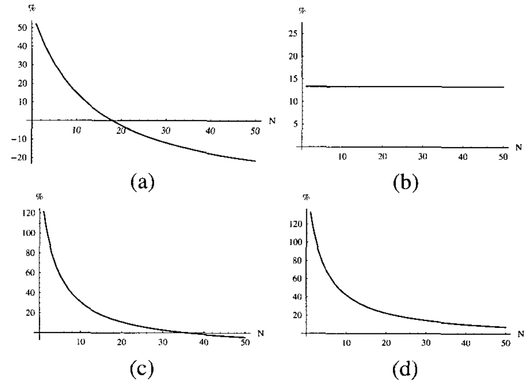


Figure 5: The proposed logarithmic adder/subtractor.

$\frac{C_{\mathrm{I}}-C_{\mathrm{IV}}}{C_{\mathrm{I}}}100\%$ are plotted in Fig. 6. Fig. 6(b) reveals that the logarithmic architecture is up to 14% faster, while it can also require less areas for several N. In fact for small N, which are of practical importance, area savings can reach over 50%, as shown in Fig. 6(a). A performance comparison in terms of the area×time and area×time^2 criteria, shown in Fig. 6(c) and (d), demonstrates that the logarithmic architectures can be up to 120% more efficient. A technique to quantify the conversion overhead has been presented in [11].

5 Conclusions

Four architectures that implement the FLOS-CMA have been introduced, each applicable to a particular environment. Among them, the logarithmic implementation is found to match the computational idiosyncrasies of the FLOS-CMA, such as the evaluation of the power function x^y, leading to efficient hardware implementations, of significant complexity savings.



(a) (b) (c) (d)

Figure 6: Percentage of complexity savings achieved by Type IV architecture over a Type I architecture, in terms of area (a), time (b), area×time (c), and area×time^2 (d), versus the number of equalizer taps N.

References

[1] J. G. Proakis, *Digital Communications*. New York: McGraw-Hill, 1995.

[2] S. Haykin, *Adaptive Filter Theory*. New Jersey: Prentice Hall, 1996.

[3] Y. Sato, "A method of self-recovering equalization for multilevel-amplitute modulation systems," *IEEE Trans. Comm.*, pp. 679–682, June 1975.

[4] I. Fijalkow, A. Touzni, and J. Treichler, "Fractionally spaced equalization using CMA:robustness to channel noise and lack of disparity," *IEEE Trans. Signal Processing*, pp. 56–66, January 1997.

[5] C. L. Nikias and M. Shao, *Signal Processing with Alpha-Stable Distributions and Applications*. New York: John Wiley and Sons, 1995.

[6] M. Rupi, P. Tsakalides, E. D. Re, and C. Nikias, "Robust constant modulus arrays based on fractional lower-order statistics," *Proceedings of ICASSP99*, 1999.

[7] E. Swartzlander and A. Alexopoulos, "The sign/logarithm number system," *IEEE Transactions on Computers*, vol. 24, pp. 1238–1242, Dec. 1975.

[8] J. R. Sacha and M. J. Irwin, "The Logarithmic Number System for strength reduction in adaptive filtering," in *Proceedings of International Symposium on Low-Power Electronics and Design, (ISLPED'98)*, (Monterey, CA), pp. 256–261, 1998.

[9] K. Parhi, *VLSI Digital Signal Processing Systems*. Wiley, 1999.

[10] P. Markstein, *IA-64 and elementary functions*. Hewlett-Packard Professional Books, 2000.

[11] V. Paliouras and T. Stouraitis, "Low-power properties of the Logarithmic Number System," in *Proceedings of 15th Symposium on Computer Arithmetic (ARITH15)*, 2001.

[12] ES2, *ES2 ECPD07 library databook*. Rousset, France: European Silicon Structures, 1992.