Low-Power Configurable Processor Array for DLMS Adaptive Filtering

S. Ramanathan
raman@serc.iisc.ernet.in

V. Visvanathan
vish@serc.iisc.ernet.in

Supercomputer Education and Research Centre
Indian Institute of Science
Bangalore-560012.

Abstract

In this paper, we first present a pipelined delayed least mean square (DLMS) adaptive filter architecture whose power dissipation meets a specified budget. This low-power architecture exploits the parallelism in the DLMS algorithm to meet the required computational throughput. The architecture exhibits a novel tradeoff between algorithmic performance and power dissipation. This architecture is then extended to derive a configurable processor array (CPA), which is configurable for filter order, sample period and power reduction factor. The hardware overhead incurred for configurability is minimal.

1. Introduction

The least mean squares (LMS) adaptive filtering algorithm used in a broad range of engineering applications is not amenable to pipelining [1], a technique indispensable in the design of low-power or high-speed architectures [2]. The problem stems from the absence of a sufficient number of delays (registers) in the error feedback loop which provides the filter's prediction error to all the taps in order to update the filter coefficients.

The use of delayed coefficient adaptation in the LMS algorithm [1] has been an important contribution towards developing pipelined LMS adaptive filter architectures. The basic idea is to update the filter coefficients using a delayed value of the error, which provides the required registers in the error feedback loop to pipeline the filter architecture. However, as has been documented in [1], the convergence speed of the delayed least mean squares (DLMS) adaptive filtering algorithm degrades progressively with increase in the adaptation delay. This is of particular concern in a non-stationary environment as it could lead to a loss of tracking capability. Long et.al [1] therefore recommend that "every effort should still be made to keep the delay as small as possible if it is not avoidable".

Based on the DLMS algorithm, many researchers have proposed pipelined architectures [3-5]. The architecture reported in [5] uses significantly fewer adaptation delays compared with other existing architectures, thus resulting in superior convergence speed. We in this paper, use this architecture to derive a low-power CMOS configurable processor array (CPA) for DLMS adaptive filtering. Towards this end, we extend the synthesis procedure reported in [5] to synthesize pipelined architectures with minimal adaptation delay subject to a power constraint. In order to meet the given power constraint, the supply voltage is treated as a free variable. The resulting low-power architecture exploits the parallelism in the DLMS algorithm to meet the required computational throughput. An interesting and novel tradeoff between algorithmic performance and power dissipation is exhibited by this architecture. This tradeoff is illustrated through a system identification example.

Based on the low-power architecture which is designed for a specific filter order (N), sample period (T_s), and power reduction factor (β), we derive a CPA configurable for N, T_s and β. Both, the programmable clock frequency required for the configurability of the CPA, and the variable power supply needed to meet the required power reduction, are realized by a phase locked loop based design [6]. The CPA has minimal hardware overhead for the configurability and hence will not dissipate significantly more power than the corresponding hard-wired array for a given N, T_s and β. The CPA has two phases of operation viz., the configure phase - wherein the processor array is pro-

0-8186-7755-4/96 $05.00 © 1996 IEEE

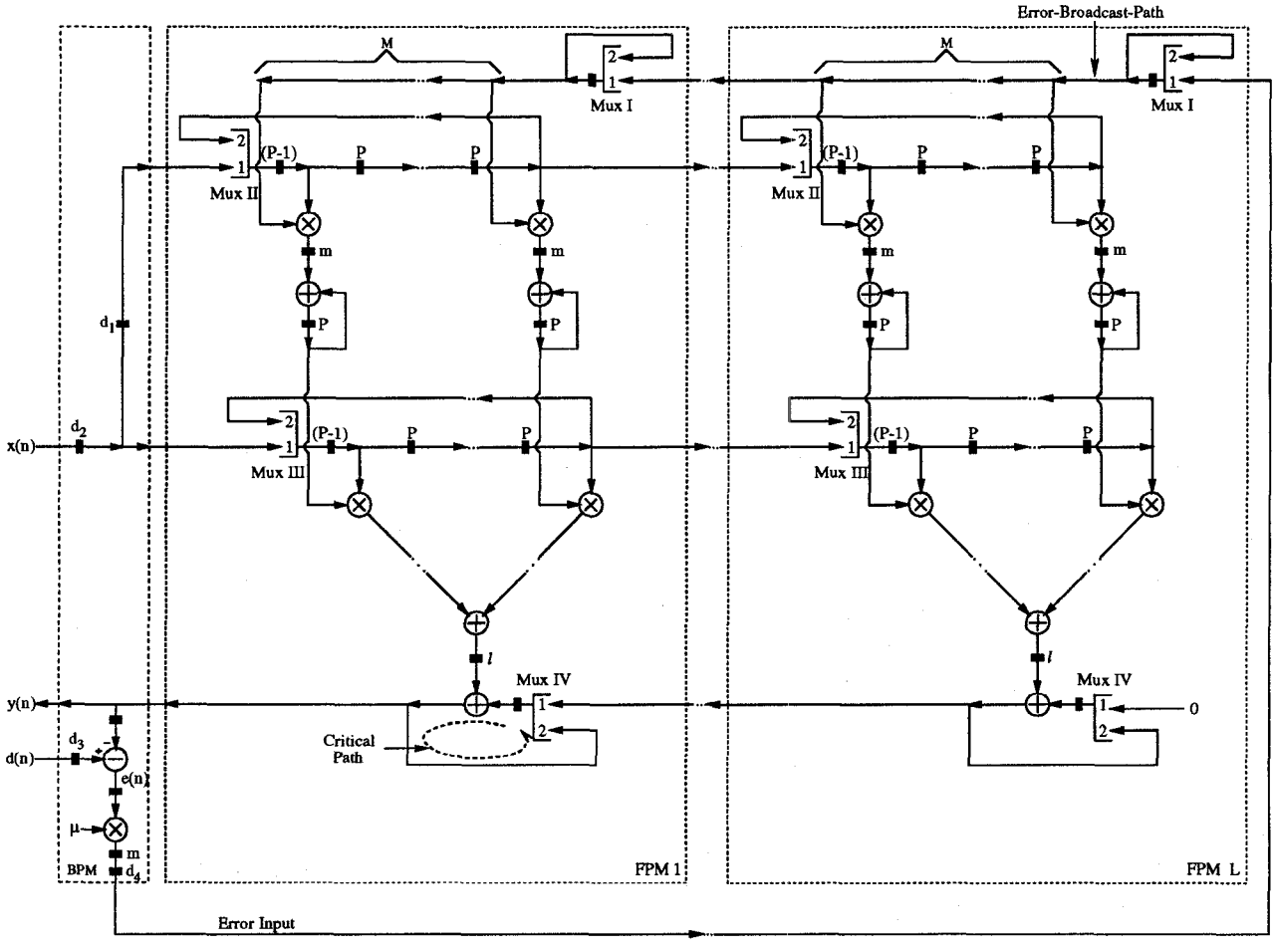

Figure 1. Pipelined DLMS Architecture.

grammed for a specific N, T_s and β, and the execution phase - wherein the processor array performs the desired DLMS adaptive filtering operation.

The organization of the paper is as follows. Section 2 reviews the pipelined DLMS architecture with minimal adaptation delay reported in [5]. In Section 3, we extend the synthesis methodology of [5] to incorporate a power constraint. We also discuss the novel tradeoff between algorithmic performance and power dissipation exhibited by the resulting low-power architecture, and substantiate this tradeoff through a system identification example. This low-power architecture is designed for a specific N, T_s and β. In Section 4, we extend this design to realize a processor array configurable for N, T_s and β. We summarize the work in Section 5.

2. Pipelined DLMS Architecture

The pipelined architecture shown in Fig. 1 is derived by applying a sequence of function preserving transformations (as described in [5]) on the standard signal flow graph representation of the DLMS adaptive filtering algorithm. In the figure, μ represents the step-size, and x, y and d are respectively, the input, the output and the desired response of an N-th order adaptive filter. Further, m and l are respectively, the number of pipeline registers for a multiplier and a multiplier followed by an adder tree of depth $\lceil \log_2 M \rceil$. Following the standard notation, the pipeline registers are shown at the output of the arithmetic units. M is chosen such that, the broadcast delay of the M fan-out lines in the error-broadcast-path is less than the critical path delay (T_{crit}) of the architecture. The precise expressions for $d_1, \cdots, d_4$ in Fig. 1 are available in [7].

The circular systolic array shown in Fig. 1 consists of a boundary processor module (BPM) and L $(= \frac{N}{MP})$ folded processor modules (FPMs), where P indicates the number of computations performed by an FPM in any given sample period T_s. For ease of exposition we shall assume throughout this paper that N is a multiple

of MP. The extension of the architecture to the more general case of arbitrary N, is straightforward and is available in [7].

The critical path of the architecture consists of an adder and a 2-to-1 multiplexer as indicated in Fig. 1. Since the multiplexer delay is negligible compared with an adder delay, $T_{crit} \approx T_a$, where T_a indicates the delay of an adder. The clock period T_c is matched to T_{crit}, hence

$$P = \left\lfloor \frac{T_s}{T_{crit}} \right\rfloor \quad (1)$$

$$\text{and } T_c = \frac{T_s}{P} \quad (2)$$

The precise definitions of the various multiplexers are available in [7]. The expression for the adaptation delay (D_A) and the holdup delay (D_H) are given as follows [5]:

$$D_A = \frac{N}{MP} + \left\lceil \frac{2m+l+2}{P} \right\rceil \quad (3)$$

$$D_H = \left\lceil \frac{l}{P} \right\rceil \quad (4)$$

Since, minimizing the adaptation delay (D_A) directly contributes to improved algorithmic performance (convergence speed) [1], it is beneficial to minimize D_A. However, we show in the following section that D_A is a monotonically decreasing function of the supply voltage V. This brings about a novel tradeoff between algorithmic performance and power dissipation.

3. Algorithmic Performance vs. Power Dissipation

We first establish the relationship between the adaptation delay D_A and the supply voltage V. Towards this end, we require the following expression for the critical path delay (T_{crit}) in CMOS technology [2],

$$T_{crit}(V) = \frac{C\,V}{k(V - V_T)^2} \quad (5)$$

where, C is the capacitive load of the critical path, k and V_T are device level model parameters and V is the supply voltage.

As we know from [1], minimizing the adaptation delay directly contributes to improved algorithmic performance. However, for a given N and T_s, decreasing D_A

$\Longrightarrow$ increasing P (from (3); m, l and M are constants for a given critical path)

$\Longrightarrow$ decreasing T_{crit} (from (1))

$\Longrightarrow$ increasing V (from (5))

Thus, D_A is a monotonically decreasing function of the supply voltage V. In other words, the minimum adaptation delay solution needs maximum supply voltage (say V_{max}). Further, the critical path delay corresponding to the supply voltage V_{max} is given by

$$T_{crit}(V_{max}) = \frac{C\,V_{max}}{k\,(V_{max} - V_T)^2} \quad (6)$$

We now derive the expression for power dissipation as a function of the supply voltage and the algorithm specifications. In CMOS technology [2],

Power Dissipation

$= C_{eff}\,V^2\,f_c$
where, C_{eff} is the effective switched capacitance of the realization and f_c $\left(= \frac{1}{T_c}\right)$ is the clock frequency

$\approx ((2LM)(C_M + C_A))\,V^2\,(P\,f_s)$
where, C_M and C_A are the effective switched capacitance of a multiplier and an adder respectively (for ease of exposition, we neglect the power dissipated by the control circuitry and the boundary processor module)

$$= 2N(C_M + C_A)\,V^2\,f_s \quad (7)$$

In (7), for the given algorithm specifications and technology, the only free variable is the supply voltage V. Thus, the architecture corresponding to the minimum adaptation delay solution, dissipates the peak power of $2N(C_M + C_A)V_{max}^2 f_s$.

We now synthesize low-power architectures by treating the supply voltage V as a free variable and using parallelism to meet the required computational throughput [2]. Towards synthesizing the architecture under a power constraint, let us define the power reduction factor β as:

$$\beta = \frac{\text{Desired Power Dissipation}}{\text{Peak Power Dissipation}} = \frac{V^2}{V_{max}^2} \quad \text{(From (7))} \quad (8)$$

Note that, β equals 1 for the minimum adaptation delay solution. The supply voltage V for the architecture which meets the given β (< 1) can be calculated

Metric	This Work		Previous
	$\beta = 1$	$\beta = 0.25$	Work [4]
V	3.3 volts	1.65 volts	3.3 volts
P	8	2	1
D_A	4	16	64
D_H	1	3	64
# of M'pliers	17	65	129
# of Adders	17	65	129
# of Registers	307	513	571
# of Muxes	8	32	0

Table 1. Comparison Table.

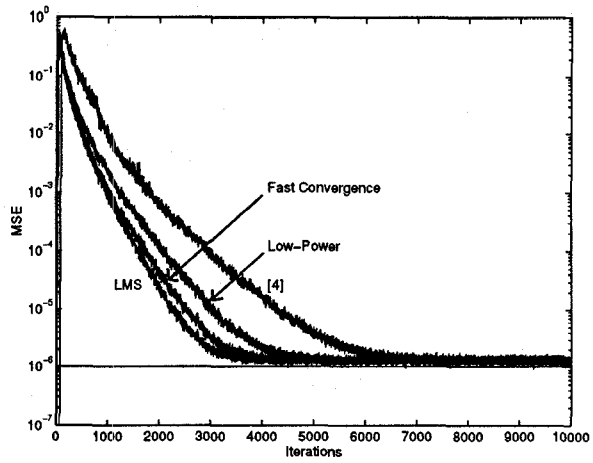

Figure 2. Convergence Plots.

from (8). Further, from (5) and (6), the critical path delay for the supply voltage V is given by

$$T_{crit}(V) = \left(\frac{T_{crit}(V_{max})\ (V_{max} - V_T)^2}{V_{max}}\right) \frac{V}{(V - V_T)^2} \quad (9)$$

Hence from (1), the required value of P can be obtained. The number of required FPMs can now be obtained as $L = \frac{N}{MP}$. Since we use parallelism to reduce power, for any given power reduction factor β, we need more number of FPMs than compared with the minimum D_A solution. Further note that, for any $\beta < 1$, the adaptation delay increases because P decreases. This brings about the interesting tradeoff between algorithmic performance and power dissipation. This tradeoff is illustrated through a system identification example described below.

System Identification Example: The task is to identify the coefficients of a linear-phase FIR filter of order 64 by using a DLMS adaptive filter of the same order. The input, whose sample period is 80ns, is a 90-th order moving average source of unit power. The measurement noise is an independent additive zero-mean white noise of power 10^{-6}. The critical path delay at the peak supply voltage (V_{max}) of 3.3 volts and threshold voltage (V_T) of 0.6 volt is assumed to be 10ns. Further, let $m = 4$, $l = 6$ and $M = 4$.

Table 1 highlights the two cases of this work, namely the synthesis solution without power constraint ($\beta = 1$) and the synthesis solution with a power constraint $\beta = 0.25$, for various metrics. Note that, the synthesis solution with a power constraint uses extra hardware in order to achieve the power reduction. Further, the adaptation delay is more than the minimum adaptation delay solution (or synthesis solution without power constraint).

Fig. 2 shows the simulation results obtained using SIMULINK [8] with $D_A = 4$ and $D_H = 1$ (minimum D_A solution or fast convergence solution) and $D_A = 16$ and $D_H = 3$ (synthesis solution with power constraint or low-power solution). SIMULINK provides cycle true simulation and thus the architecture is verified at the RTL level and hence can be easily synthesized onto silicon. For each plot, the step-size was chosen to get a critically damped convergence of the mean squared error for a fixed misadjustment of 30%. The ideal LMS convergence plot (ie., $D_A = D_H = 0$) is shown for reference. It is clear from Fig. 2 that algorithmic performance (namely convergence speed) is traded for power.

Also note that, in the same plot, the performance of the architecture reported in [4] is shown. The convergence speed of this architecture operating at the maximum supply voltage V_{max} and hence dissipating the peak power of $2N(C_M + C_A)V_{max}^2 f_s$, is significantly worse than that of the low-power solution of this work. Further, note the shift in the convergence plot of [4] due to the large holdup delay which is a function of the filter order. Table 1 further highlights the significant improvement of this work compared with [4] for various metrics.

It is interesting to note the impact of the following delay model, described in [9] for sub-micron CMOS technology, on the trade-off between algorithmic performance and power dissipation:

$$T_{crit}(V) = \frac{C\ V}{k(V - V_T)^\alpha} \quad (10)$$

where, α ranges between 1 and 2. For example, in a 0.6μm CMOS technology [10] wherein $\alpha = 1.6$, D_A and D_H for the low-power solution are 12 and 2 respectively. This would then imply that, the degradation in the convergence speed for the low-power solu-

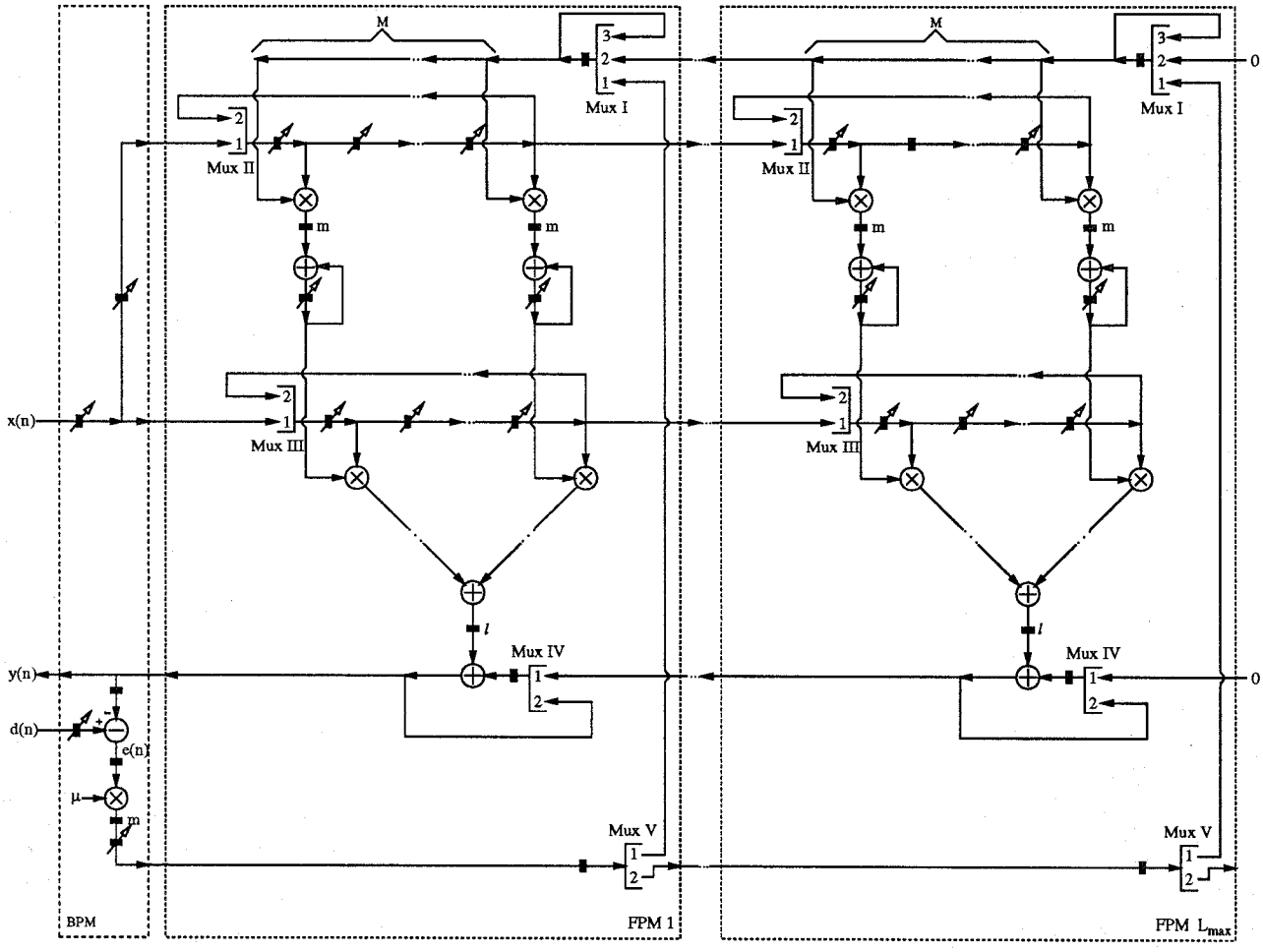

Figure 3. Configurable Processor Array.

tion ($\beta = 0.25$) with respect to the fast convergence solution ($\beta = 1$) is less for $\alpha = 1.6$ than for $\alpha = 2$. □

Note that, since (8) and (9) are simple high level equations, the precise power reduction in an actual implementation may be some what different from β. Nevertheless, it is significant that such a tradeoff can be made at the highest level of abstraction viz., the algorithmic level.

4. Configurable Processor Array

In this section, we describe a low-power configurable processor array (CPA) for DLMS adaptive filtering, configurable for filter order N, sample period T_s and power reduction factor β. We see from the previous section, that the low-power architecture for a given N, T_s and β is precisely specified by the values of P, L and V. Hence, the low-power CPA should be made programmable for P, L and V. In other words, the number of FPMs, the delay lines, the control for multiplexers, and the clock frequency and the supply voltage should be made programmable.

We conceive of a CPA having L_{max} FPMs. Since, the number of FPMs used ($L \leq L_{max}$) is programmable, the error input (refer Fig. 1) should be broadcast to each of the L_{max} FPMs separately. But, this would distort the systolic nature of the architecture. This is overcome by introducing a register and a 2-to-1 multiplexer in each FPM (refer Mux-V in Fig. 3), at a minimal cost of a few adaptation delays. This then results in a CPA which is a linear systolic array as shown in Fig. 3. In order to incorporate the above change, mux-I in each FPM (refer Fig. 3) is now a 3-to-1 multiplexer. The precise multiplexer definitions are available in [7]. The adaptation delay required by this linear systolic array for a specified N, T_s and β is given by the expression $D_A = \frac{N}{MP} + \left\lceil \frac{2m+l+2+L}{P} \right\rceil$. Note that, this value of D_A is marginally more than

that given by (3). The $(L_{max} - L)$ unused FPMs are powered down to minimize power dissipation [2].

The delay lines in the CPA which are a function of P, are indicated by an arrow across them (refer Fig. 3). Since P is programmable, each of these delay lines have to be replaced by programmable delay lines. Further, the multiplexers except mux-V in the CPA are P-periodic. Hence the control for these multiplexers should be programmable. These can be achieved by straightforward design procedures, the details of which are available in [11].

Exactly like in the case of the CPA for an FIR filter [11], we require a programmable clock $(T_c = \frac{T_s}{P})$ and a programmable power supply. This can be achieved by a phase-locked-loop (PLL) based design [6] as described in [11]. The operation of the CPA consists of two phases viz., the configure-phase and the execution-phase. During the configure-phase of the processor array, the various configurable elements are programmed. During the execution-phase, the desired DLMS adaptive filtering operation is performed. Note that, the hardware overhead for configurability is minimal. This leads us to conclude that, for a given N, T_s and β, the CPA will not dissipate significantly more power than the corresponding hard-wired array.

5. Conclusion

We have presented a low-power pipelined DLMS adaptive filter architecture which exhibits a novel tradeoff between algorithmic performance and power dissipation. This architecture exploits the parallelism in the DLMS algorithm to meet the required computational throughput. It was shown that this use of parallelism to reduce power results in an increase in the adaptation delay, thereby creating a tradeoff between algorithmic convergence speed and the power reduction factor. This tradeoff was illustrated with a system identification example.

A processor array configurable for filter order, sample period and power reduction factor, was derived from the low-power architecture. Since the CPA has minimal control overhead, it is expected that it will not dissipate significantly more power than the corresponding hard-wired counterpart.

References

[1] G. Long, F. Ling, and J. G. Proakis, "The LMS algorithm with delayed coefficient adaptation," *IEEE Trans. Acoust., Speech, and Signal Processing*, vol. 37, no. 9, pp. 1397-1405, Sept. 1989, and corrections, vol. 40, no. 1, pp. 230-232, Jan. 1992.

[2] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low-power CMOS digital design," *IEEE J. Solid-State Circuits*, vol. 27, pp. 473-484, Apr. 1992.

[3] H. Herzberg, R. Haimi-Cohen, and Y. Be'ery "A systolic array realization of an LMS adaptive filter and effects of delayed adaptation," *IEEE Trans. Signal Processing*, vol. 40, no. 11, pp. 2799-2802, Nov. 1992.

[4] M. D. Meyer and D. P. Agrawal, "A high sampling rate delayed LMS filter architecture," *IEEE Trans. Circuits and Systems - II*, vol. 40, no. 11, pp. 727-729, Nov. 1993.

[5] S. Ramanathan and V. Visvanathan, "A systolic architecture for LMS adaptive filtering with minimal adaptation delay," *Proc. 9th Intl. Conf. VLSI Design*, Bangalore, pp. 286-289, Jan. 1996.

[6] V. V. Kaenel, P. Macken, and M. G. R. Degrauwe, "A voltage reduction technique for battery-operated systems," *IEEE J. Solid-State Circuits*, vol. 25, no. 5, pp. 1136-1140, Oct. 1990.

[7] S. Ramanathan and V. Visvanathan, "Low-power architectures for DLMS adaptive filtering," *Technical Report*, SERC Dept., Indian Institute of Science, Bangalore, Sep. 1996.

[8] SIMULINK User's Guide, *MathWorks Inc.*, Dec. 1993.

[9] T. Sakurai and A. R. Newton, "Delay analysis of series-connected MOSFET circuits," *IEEE J. Solid-State Circuits*, vol. 26, pp. 122-131, Feb. 1991.

[10] Uming Ko, Poras T. Balsara, and Wai Lee, "Low-power design techniques for high-performance CMOS adders," *IEEE Trans. VLSI Systems*, vol. 3, no. 2, pp. 327-333, Jun. 1995.

[11] V. Visvanathan and S. Ramanathan, "Synthesis of energy-efficient configurable processor array," *Proc. Intl. Workshop on Parallel Processing*, Bangalore pp. 627-632, Dec. 26-31, 1994.