PERFORMANCE OF THE STRENGTH-REDUCED ADAPTIVE FILTER ARCHITECTURE FOR 51.84 Mb/s ATM-LAN[1]

Manish Goel and Naresh R. Shanbhag

Coordinated Science Laboratory/ECE Department,
University of Illinois at Urbana-Champaign,
1308 W. Main, Urbana, IL 61801.
E-mail : [mgoel,shanbhag]@uivlsi.csl.uiuc.edu

Abstract — **In this paper, a pipelined strength-reduced (PIPSR) adaptive filter architecture is employed as a receive equalizer for** 51.84 *Mb/s* **ATM-LAN over unshielded twisted pair category-3 (UTP-3) wiring. This architecture provides the advantage of low-power dissipation and high-speed operation. Simulation results are presented to investigate the effect of level of pipelining on the steady-state signal-to-noise ratio at the slicer** (SNR_{slicer})**. Simulation results indicate that speed-ups of up to** 160 **can be achieved with about** 0.8 **dB loss in** SNR_{slicer}.

1. Introduction

Strength reduction is an algebraic transformation, which has been proposed [1] to trade-off multipliers with adders in a complex multiplication thereby achieving power reduction. In [3], we proposed the application of *strength reduction* transformation at the algorithmic level to adaptive systems involving complex signals and filters. It was shown in [3] that the strength-reduced (**SR**) filter enables power savings of $21 - 25\%$ over the traditional cross-coupled (**CC**) filter with no loss in performance. However, the application of strength reduction increases the critical path and hence an inherently pipelined **SR** (**PIPSR**) architecture was also presented. Furthermore, by trading off the throughput gained through pipelining with power supply scaling [1], it was demonstrated that additional power savings of $40 - 69\%$ are feasible. Clearly, the **SR** and **PIPSR** architectures is an attractive alternative to the traditional **CC** architecture for high bit-rate communications and digital signal processing applications.

In this paper, we demonstrate an application of the **SR** and **PIPSR** architectures as an equalizer for quadrature amplitude multiplex (QAM) [5] receivers. We employ QAM receiver for 51.84 *Mb/s* [4] ATM-LAN over 100 meters of unshielded twisted pair category-3 (UTP-3) cable employing 16-CAP (carrierless amplitude/phase) modulation scheme. It must be mentioned that the 16-CAP line code was chosen as ATM-LAN standard over UTP-3 at 51.84 Mb/s [4]. Therefore, this study is of great interest as it can lead to low-power and cost-effective ATM-LAN transceivers. We present the simulation results for this application in order to investigate the effect of the level of pipelining on the steady-state signal-to-noise ratio at the slicer (SNR_{slicer}).

[1]This research was supported by Analog Devices Inc. and University of Illinois.

2. Pipelined Strength-reduced (PIPSR) Equalizer

In this section, we review the strength reduction transformation and development of the **PIPSR** architecture [3] from the **CC** architecture. The reader is referred to [3] for more details, while we will present only the final results here.

A *Strength Reduction Transformation*

Consider the problem of computing the product of two complex numbers $(a + jb)$ and $(c + jd)$ as shown below,

$$(a + jb)(c + jd) = (ac - bd) + j(ad + bc). \qquad (2\ .\ 1)$$

From (2.1), a direct-mapped architectural implementation would require a total of four real multiplications and two real additions to compute the complex product. Application of strength reduction involves reformulating (2.1) as follows,

$$\begin{aligned}(a - b)d + a(c - d) &= ac - bd \\ (a - b)d + b(c + d) &= ad + bc,\end{aligned} \qquad (2\ .\ 2)$$

where we see that strength reduction reduces the number of multipliers by one at the expense of three additional adders. Typically, multiplications are more expensive than additions and hence we achieve an overall savings in hardware.

B *Strength-reduced (SR) Architecture*

The **SR** architecture [3] is obtained by applying strength reduction transformation at the algorithmic level instead of at the multiply-add level described in the previous subsection. Starting with the complex LMS algorithm, assume that the filter input is a complex signal $\mathbf{X}(n)$ given by $\mathbf{X}(n) = \mathbf{X}_r(n) + j\mathbf{X}_i(n)$, where $\mathbf{X}_r(n)$ and $\mathbf{X}_i(n)$ are the real and the imaginary parts of the input signal vector $\mathbf{X}(n)$. Furthermore, if the filter $\mathbf{W}(n)$ is also complex ($\mathbf{W}(n) = \mathbf{c}(n) + j\mathbf{d}(n)$), then the complex LMS algorithm is given by,

$$\begin{aligned}e(n) &= d(n) - \mathbf{W}^H(n-1)\mathbf{X}(n) \\ \mathbf{W}(n) &= \mathbf{W}(n-1) + \mu e^*(n)\mathbf{X}(n),\end{aligned} \qquad (2\ .\ 3)$$

0-7803-3583-X/97 $10.00 ©1997 IEEE

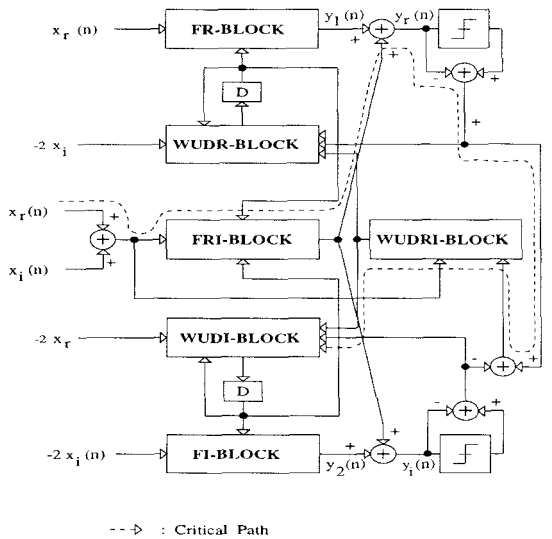

Figure 1: Low-power strength reduced adaptive filter architecture

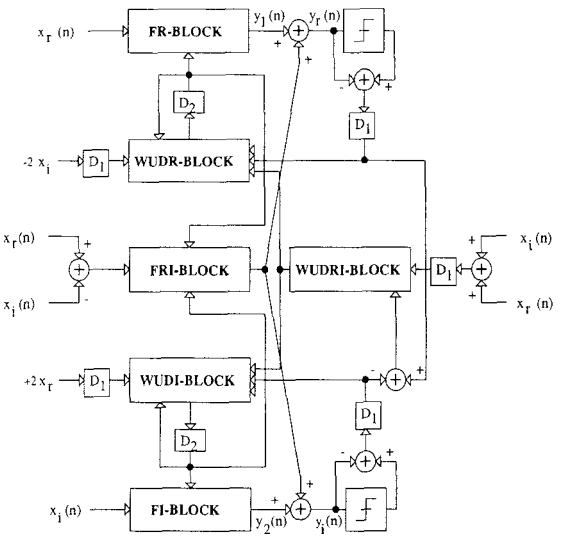

Figure 2: Pipelined strength reduced adaptive filter architecture

where μ is the step-size, $d(n)$ is the desired signal, $e(n)$ is the error and $\mathbf{W}(n)$ is the coefficient vector. Also, $e^*(n)$ represents the complex conjugate of the signal $e(n)$ and $\mathbf{W}^H(n)$ represents the hermitian of $\mathbf{W}(n)$.

Traditionally, the complex LMS algorithm is implemented via the **CC** architecture, which are described by the following equations,

$$y_r(n) = \mathbf{c}^T(n-1)\mathbf{X}_r(n) + \mathbf{d}^T(n-1)\mathbf{X}_i(n) \quad (2.4a)$$

$$y_i(n) = \mathbf{c}^T(n-1)\mathbf{X}_i(n) - \mathbf{d}^T(n-1)\mathbf{X}_r(n), \quad (2.4b)$$

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mu\left[e_r(n)\mathbf{X}_r(n) + e_i(n)\mathbf{X}_i(n)\right] \quad (2.4c)$$

$$\mathbf{d}(n) = \mathbf{d}(n-1) + \mu\left[e_r(n)\mathbf{X}_i(n) - e_i(n)\mathbf{X}_r(n)\right], \quad (2.4d)$$

where $e(n) = e_r(n) + je_i(n)$ and the **F**-block output is given by $y(n) = y_r(n) + jy_i(n)$. Equations (2.4(a)-(b)) and (2.4(c)-(d)) define the computations in the **F**-block and the **WUD**-block, respectively. A direct-mapped implementation of (2.4) would require $8N$ multipliers and adders.

We see that (2.4(a)-(b)) has two complex multiplications (inner products) and hence can benefit from the application of strength reduction. Doing so results in the following equations, which describe the **F**-block computations of the **SR** architecture [3],

$$\begin{aligned} y_1(n) &= \mathbf{c}_1^T(n-1)\mathbf{X}_r(n) \\ y_2(n) &= \mathbf{d}_1^T(n-1)\mathbf{X}_i(n) \\ y_3(n) &= -\mathbf{d}^T(n-1)\mathbf{X}_1(n) \\ y_r(n) &= y_1(n) + y_3(n) \\ y_i(n) &= y_2(n) + y_3(n), \end{aligned} \quad (2.5)$$

where $\mathbf{X}_1(n) = \mathbf{X}_r(n) - \mathbf{X}_i(n)$, $\mathbf{c}_1(n) = \mathbf{c}(n) + \mathbf{d}(n)$, and $\mathbf{d}_1(n) = \mathbf{c}(n) - \mathbf{d}(n)$. Similarly, the **WUD** computation is described by,

$$\begin{aligned} \mathbf{c}_1(n) &= \mathbf{c}_1(n-1) + \mu[\mathbf{eX}_1(n) + \mathbf{eX}_3(n)] \\ \mathbf{d}_1(n) &= \mathbf{d}_1(n-1) + \mu[\mathbf{eX}_2(n) + \mathbf{eX}_3(n)], \end{aligned} \quad (2.6)$$

where $\mathbf{eX}_1(n) = 2e_r(n)\mathbf{X}_i(n)$, $\mathbf{eX}_2(n) = 2e_i(n)\mathbf{X}_r(n)$, $\mathbf{eX}_3(n) = e_1(n)\mathbf{X}_1(n)$, $e_1(n) = e_r(n) - e_i(n)$, $\mathbf{X}_1(n) = \mathbf{X}_r(n) - \mathbf{X}_i(n)$. It is easy to show that the **SR** architecture (see Fig. 1) requires only $6N$ multipliers and $8N+3$ adders. This is the reason why the **SR** architecture results in $21-25\%$ power savings [3] over the **CC** architecture.

C *Pipelined Strength-reduced (***PIPSR***) Architecture*

The dotted line in Fig. 1 indicates the critical path of the **SR** architecture. As explained in [3], both the **SR** as well as **CC** architectures are bounded by a maximum possible clock rate due the computations in this critical path. This throughput limitation is eliminated via the application of the *relaxed look-ahead transformation* [7] to the **SR** architecture (see (2.5-2.6)). The relaxed look-ahead transformation is an approximation of the *look-ahead transformation* [6] and it results in hardware efficient pipelined adaptive filter architectures. Application of relaxed look-ahead to the **SR** architecture in (2.5-2.6) results in the following equations that describe the **F**-block computations in the **PIPSR** architecture,

$$\begin{aligned} y_1(n) &= \mathbf{c}_1^T(n-D_2)\mathbf{X}_r(n) \\ y_2(n) &= \mathbf{d}_1^T(n-D_2)\mathbf{X}_i(n) \\ y_3(n) &= -\mathbf{d}^T(n-D_2)\mathbf{X}_1(n) \\ y_r(n) &= y_1(n) + y_3(n) \\ y_i(n) &= y_2(n) + y_3(n), \end{aligned} \quad (2.7)$$

where D_2 is the number of delays introduced before feeding the filter coefficients into the **F**-block. Similarly, the computation of the **WUD** block of the **PIPSR** architecture are given by,

$$\begin{aligned} \mathbf{c}_1(n) &= \mathbf{c}_1(n-D_2) + \mu \sum_{i=0}^{LA-1} [\mathbf{eX}_1(n-D_1-i) + \\ &\quad \mathbf{eX}_3(n-D_1-i)] \\ \mathbf{d}_1(n) &= \mathbf{d}_1(n-D_2) + \mu \sum_{i=0}^{LA-1} [\mathbf{eX}_2(n-D_1-i) + \\ &\quad \mathbf{eX}_3(n-D_1-i)] \end{aligned} \qquad (2.8)$$

where $\mathbf{eX}_1(n)$, $\mathbf{eX}_2(n)$ and $\mathbf{eX}_3(n)$ are defined in the previous subsection and $D_1 \geq 0$ are the delays introduced into the error feedback loop. A block level implementation of the **PIPSR** architecture is shown in Fig. 2 where D_1 and D_2 delays will be employed to pipeline the various operators such as adders and multipliers at a fine-grain level. The high-throughput of the **PIPSR** architecture can be traded-off with supply voltage reduction resulting in additional power savings [3] of $40-69\%$. Therefore, the **PIPSR** architecture results in $60-90\%$ power savings as compared to the serial **CC** architecture.

3. Application to 51.84 Mb/s ATM-LAN

In this section, we will study the performance of the proposed low-power adaptive filter architecture in a high-speed digital communication system. In particular, we will employ the proposed architecture as an adaptive equalizer in a CAP-QAM modulation scheme for a data-rate of 51.84 Mb/s over 100 meter of unshielded twisted-pair(UTP-3) wiring. 16-CAP is currently the line-code of choice in this application [4].

While the standard does specify the line code to be 16-CAP, there is a flexibility in choosing the transmitter and receiver structure. For the purpose of demonstration of the low-power adaptive filter, we have assumed a CAP transmitter [4] and QAM receiver [5]. In addition to channel distortion, the received signal has near-end cross talk (NEXT) signal superimposed upon it. The NEXT impairment occurs due to the physical proximity of wire pairs used for duplex mode operation. We present a brief description of the CAP transmitter [4] and QAM receiver [2, 5].

A The CAP Transmitter

The block diagram of a digital CAP transmitter is shown in Fig. 3. The bit stream is first passed through a scrambler. The scrambled bits are then fed into 16-CAP encoder, which maps block of 4 bits into one of 16 different complex symbols shown in Fig. 4. The real and imaginary symbol streams are processed by digital shaping filters. This requires the shaping filters to be operated at a sampling frequency f_s, which is at least twice the maximum frequency component of the

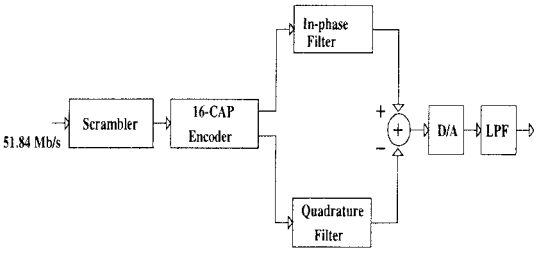

Figure 3: The CAP transmitter

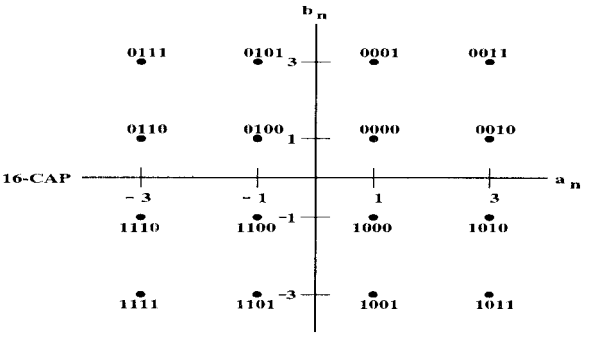

Figure 4: The 16-CAP signal constellation

transmit spectrum. The outputs of the filters are subtracted and the result is passed through a digital-to-analog (D/A) converter, which is followed by an interpolating low-pass filter.

The output spectrum is braodband with a bandwidth of 25.92 MHz. The bit rate of 51.84 Mb/s and 16-CAP signal constellation imply a symbol rate of 12.96 $Mbaud$. Hence, the chosen transmit spectrum has 100 % excess bandwidth.

B The QAM Receiver

The QAM receiver in Fig. 5, first demodulates the received signal (which is sampled at 51.84 $Msamples/s$), such that the output of the low-pass filters (LPF) has energy from DC to 12.96 MHz. This allows us to downsample the LPF output by a factor of two. The resulting complex signal can then be filtered via the adaptive equalizer, which can be implemented as the traditional **CC** architecture or the proposed **PIPSR** architecture. The equalizer output is sampled at the symbol rate of 12.96 MHz, which is then passed through the slicer to generate the detected symbols. The error across the slicer is employed to adapt the equalizer coefficients once every symbol period. The detected symbols are also decoded to generate the received bit-stream.

C Simulation Results

A signal-to-noise ratio (SNR) of 23.25 dB at the slicer makes sure that the bit-error probability with 16-CAP is less than 10^{-10}. The values of step-size, μ employed in the simulations were deliberately made powers of two so that the hardware implementation re-

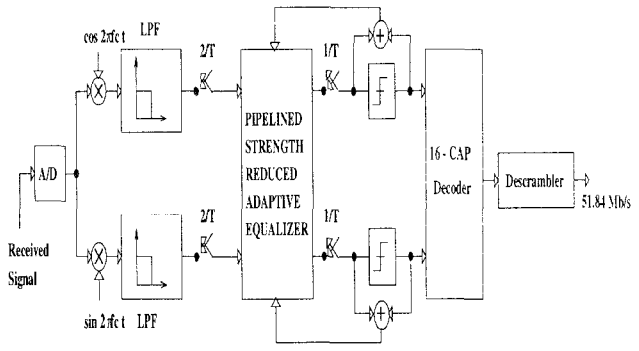

Figure 5: The QAM receiver

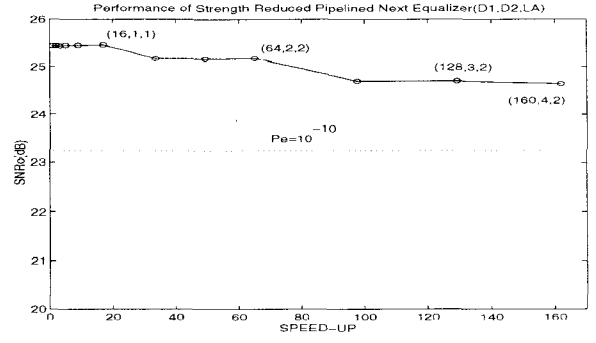

Figure 6: SNR_o vs. speed-up

quires only shift-right operations while implementing the multiplications by μ in the weight-update (**WUD**) block. In general, we employed gear-shifting after the first 120,000 symbols and again after 240,000 symbols. Furthermore, the receive equalizer was chosen to have a span of 32 symbol periods, which implies 128 complex coefficients in the adaptive equalizer.

In Fig. 6, we plot the SNR_{slicer} with respect to the speed-up, where the speed-up is defined as the ratio of the $T_{\mathbf{SR}}$ to $T_{\mathbf{PIPSR}}$. It is clear that SNR_{slicer} degrades by less than 0.8 dB for speed-ups of upto 160. Speed-ups upto 50 or 60 maybe sufficient for most of the practical applications for which the performance loss is less than 0.27 dB. Thus, the proposed structure has substantial speed-ups with negligible performance loss. Clearly, the SNR_{slicer} is greater than $25dB$ (which means margin of approximately 2 dB) for most of the levels of pipelining.

For all cases, SNR_{slicer} increases to 20 dB within 4 ms (approximately 50,000 symbols). This is indicated in Fig. 7, where the convergence plot for **SR** and **PIPSR** (speed-up of 160) for 360,000 symbols is shown. It can be seen that the degradation in the steady state SNR_{slicer} due to the pipelining is less than 1.0 dB. This is an attractive result, given that the pipelined architecture enables power-savings of $60 - 90\%$. Also worth noting is fact that the total convergence time is 28 ms, while a few hundred milliseconds is acceptable for this

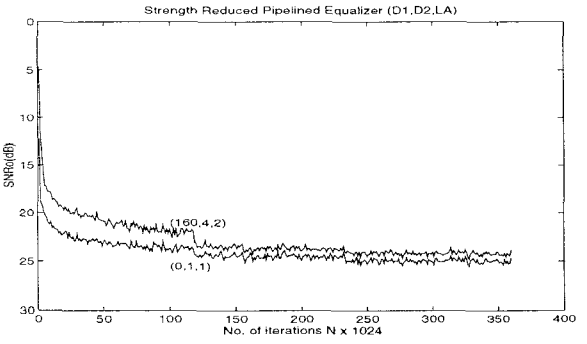

Figure 7: Convergence curves for the error across the slicer

application. Thus, we conclude that the proposed architecture is a viable alternative for QAM based receivers especially in an ATM-LAN environment

It must be mentioned that the low-power architecture presented in this paper is applicable to any communication system which employs two dimensional signal constellations. While we have demonstrated the application of the proposed architecture for 51.84 Mb/s ATM-LAN, numerous other applications exist. The finite precision analysis of the pipelined strength-reduced architecture was also carried out, and it was found that the precision requirements of the low-power architectures are similar to that of the traditional cross-coupled architecture.

References

[1] A. Chandrakasan *et al.*, "Minimizing power using transformations," *IEEE Trans. Comp.-Aided Design*, vol. 14, no. 1, pp. 12–31, Jan. 1995.

[2] R. D. Gitlin, J. F. Hayes, and S. B. Weinstien, *Data Communications Principles.* NY: Plenum Press, 1992.

[3] M. Goel and N. R. Shanbhag, "Low-power adaptive filter architectures via strength reduction," in *Proceedings of International Symposium on Low Power Electronics and Design*, (Monterey, CA), pp. 217–220, August 1996.

[4] G. H. Im and J. J. Werner, "51.84 Mb/s 16-CAP ATM-LAN standard," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 4, pp. 620–632, May 1995.

[5] E. A. Lee and D. G. Messerschmitt, *Digital Communication.* Boston, MA: Kluwer Academic Publishers, 1994.

[6] K. K. Parhi and D. G. Messerschmitt, "Pipeline interleaving and parallelism in recursive digital filters - part II : Pipelined incremental block filtering," *IEEE Trans. Acoust., Speech, and Signal Process.*, vol. 37, pp. 1118–1134, July 1989.

[7] N. R. Shanbhag and K. K. Parhi, *Pipelined Adaptive Digital Filters.* Kluwer Academic Publishers, 1994.