ELSEVIER
SIGNAL
PROCESSING
An International Journal
A publication of the European Association for Signal Processing (EURASIP)
EURASIP

Signal Processing 93 (2013) 1162–1171

Contents lists available at SciVerse ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

A block floating point treatment to finite precision realization of the adaptive decision feedback equalizer

Rafi Ahamed Shaik [a], Mrityunjoy Chakraborty [b,*]

[a] Department of Electronics and Electrical Engineering,, Indian Institute of Technology, Guwahati-781 039, India
[b] Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology, Kharagpur-721 302, India

ARTICLE INFO

Article history:
Received 6 July 2012
Received in revised form
23 November 2012
Accepted 26 November 2012
Available online 4 December 2012

Keywords:
Block floating point
Adaptive decision feedback equalizer
Mantissa
Exponent
Scaling
Overflow

ABSTRACT

A scheme for efficient realization of the adaptive decision feedback equalizer (ADFE) is presented using the block floating point (BFP) data format which enables the ADFE to process rapidly varying data over a wide dynamic range, at a fixed point like complexity. The proposed scheme adopts appropriate BFP format for the data as well as the filter weights and works out separate update relations for the filter weight mantissas and exponents. Overflows at the feed forward and the feedback filter output are prevented by certain dynamic scaling of the respective input, while overflow in weight update calculations is avoided by imposing certain upper bound on the algorithm step size μ which is shown to be less than the convergence bound. The proposed scheme deploys mostly simple fixed point operations and is shown to achieve considerable computational gain over its floating point based counterpart.

1. Introduction

In many communication systems, one often encounters channels with long impulse response (IR) that results in inter symbol interference (ISI) over several symbol periods. A typical example is given by multipath channels, where under the same delay spread conditions, the channel IR length increases with increase in symbol transmission rate. Also, on many occasions, one comes across channels that exhibit spectral nulls. The linear equalizer is not a very effective option in such cases for cancelation of the ISI, due to very large order requirement and also due to the possibility of substantial noise enhancement by the spectral peaks of the equalizer. A more effective solution in such cases is provided by the adaptive decision feedback equalizer (ADFE). The ADFE consists of a feed forward filter (FFF) and a feedback filter (FBF). The FFF, working directly on the received data, tries to equalize the anticausal part of the channel impulse response. The residual ISI at the FFF output is then canceled by passing the past decisions through an appropriately designed FBF and subtracting the FBF output from the FFF output. Both the FFF and the FBF coefficients are trained by some suitable adaptive algorithm, e.g., the LMS algorithm [1]. In practice, however, the ADFE is often required to operate in a resource constrained (e.g., low power, low chip area) environment while maintaining high throughput rate. This makes it important to devise methods for reducing the complexity of the ADFE realization. Several attempts such as [2–5] have come up in recent years which try to meet this objective by means of suitable algorithmic and architectural transformations. In this paper, we propose a different approach to the complexity reduction by adopting suitable data format for the input and the equalizer coefficients.

In a practical communication receiver, the received signal level is usually very weak which also fluctuates randomly due to effects like fading. In such cases, the input to the ADFE is obtained by first processing the received sample through a

[*] Corresponding author. Tel.: +91 9434026874; fax: +91 3222 255303.
E-mail addresses: rafiahamed@iitg.ernet.in (R.A. Shaik), mrityun@ece.iitkgp.ernet.in (M. Chakraborty).

0165-1684/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.sigpro.2012.11.024

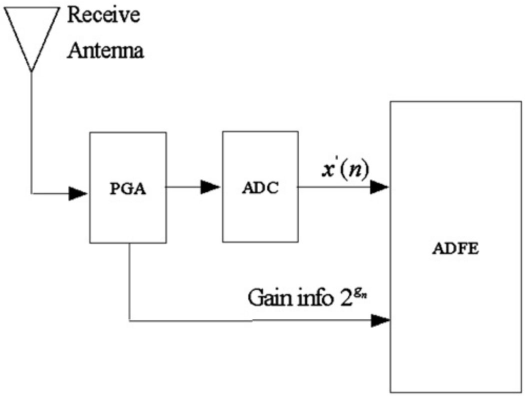

Fig. 1. Front end of an adaptive decision feedback equalizer.

programmable gain amplifier (PGA) as shown in Fig. 1. The PGA continuously adjusts its gain (by a power of two) so as to utilize the available dynamic range of the ADC maximally. This, however, gives rise to a floating point (FP) representation of the ADFE input, with the mantissa and the exponent given respectively by the ADC output and the negative of the PGA gain. A direct FP based implementation of the ADFE would, however, require much higher processing complexity than a typical fixed point (FxP) based realization, as, in FP, each stage of computation requires several additional steps not required in FxP. The block floating point (BFP) data format, in this context, is a viable alternative to the FP system. In BFP, a common exponent is assigned to a block of data. As a result, computations involving these data require only simple FxP operations, while presence of the exponent provides a FP like high dynamic range. Over decades, the BFP format has been used for efficient implementation of many signal processing algorithms. These include various forms of fixed coefficient digital filters [6–11], adaptive filters [12] and unitary transforms [13–15] on one hand and several audio data transmission standards like NICAM (stereophonic sound system for PAL TV standard), the audio part of MUSE (Japanese HDTV standard) and DSR (German Digital Satellite Radio System) on the other.

In this paper, we present a BFP treatment to finite precision implementation of the LMS based ADFE[1]. Such a realization is intrinsically more difficult than a BFP based realization [12] of transversal adaptive filters, since, unlike the latter, the ADFE consists of a decision feedback loop with a non-linear decision device. The proposed scheme provides a viable solution to this problem by effectively modifying and extending the framework of [12]. For this, first, appropriate BFP formats are adopted for the FFF and the FBF coefficients. Separate update relations for the mantissas as well as the exponents for each set of coefficients are worked out next. For the FFF, the input is block formatted by an efficient block formatting algorithm which also includes certain dynamic scaling of the data for preventing overflow at the FFF output. For the FBF, however, no block processing of the corresponding input (i.e., decisions) is possible, as that makes the system non-causal. Instead, the data stored in the FBF memory is block formatted at each time index, by appropriately modifying the proposed block formatting algorithm. It is also required to prevent overflow in the weight update computations of the FFF and the FBF. This gives rise to two upper bounds for the step size μ, one coming from the FFF and the other from the FBF considerations. The two bounds are related by a simple constant and the lesser of them is used as an upper limit of μ, which is interestingly seen to be less than $2/\mathrm{tr}\,\mathbf{R}$, (where $\mathbf{R}$ is the autocorrelation matrix of the input signal $x(n)$ and is given by $R=E[\mathbf{x}(\mathbf{n})\mathbf{x}^t(\mathbf{n})]$), i.e., upper bound of μ for convergence of the LMS iteration. The proposed scheme relies largely on simple FxP operations and thus achieves considerable speed up over a direct FP based realization. Also, simulation results show no appreciable degrading effect on the ADFE performance due to block formatting of data and filter coefficients in finite precision.

The organization of the paper is as follows: Section 2 presents a background of the BFP concept. Section 3 presents the proposed implementation scheme where BFP treatments to all the computational stages are worked out in detail. Computational complexity analysis of all the schemes proposed is carried out in Section 4 showing superiority of the proposed method over traditional FP based implementation. Simulation studies on the effects of block formatting on ADFE performance are presented in Section 5. Throughout the paper, characters with an overbar are used to indicate mantissas and the symbol Z_m for any integer m, $m\geq 0$ is used to denote the set $\{0,1,\ldots,m-1\}$.

2. BFP background

The BFP representation can be viewed as a special case of the FP format, where every f-overlapping block of N incoming data has a joint scaling factor determined by the data sample with the highest magnitude in the block. In other words, given a block $[x_0,\ldots,x_{N-1}]$, we represent it in BFP as $[x_0,\ldots,x_{N-1}]=[\bar{x}_0,\ldots,\bar{x}_{N-1}]2^{\gamma}$ where $\bar{x}_l(=x_l 2^{-\gamma})$ represents the mantissa of x_l for $l\in Z_N$ and the block exponent γ is defined as $\gamma=\lfloor\log_2 Max\rfloor+1+S$ where $Max=max(|x_0|,\ldots,|x_{N-1}|)$,

[1] Some preliminary results of this paper had been presented by the authors at ISCAS-2007, New Orleans, USA, 2007 [16].

'$\lfloor . \rfloor$' is the so-called floor function, meaning rounding down to the closest integer and the integer S is a scaling factor, used for preventing overflow during filtering operation.

In practice, if the data is given in a FP format, i.e., $x_l = M_l 2^{e_l}, l \in Z_N$ with $0 < |M_l| < 1$, and the 2's complement system is used, the above block formatting may be carried out by the block formatting algorithm presented by the authors in [16].

Note that due to the presence of S, the range of each mantissa is given as $0 \le |\bar{x}_l| < 2^{-S}$. The scaling factor S can be calculated from the inner product computation [8] representing filtering operation, given as $y(n) = \mathbf{w}^t \mathbf{x}(n) = [w_0 \bar{x}(n) + \cdots + w_{L-1}\bar{x}(n-L+1)]2^{\gamma} = \bar{y}(n)2^{\gamma}$, where $\mathbf{w}$ is a length L, fixed point filter coefficient vector and $\mathbf{x}(n)$ is the data vector at the n-th index, represented in the aforesaid BFP format. For no overflow in $y(n)$, we need $|\bar{y}(n)| < 1$. Since $|\bar{y}(n)| \le \sum_{k=0}^{k=L-1} |w_k||\bar{x}(n-k)|$ and $0 \le |\bar{x}(n-k)| < 2^{-S}, 0 \le k \le L-1$, this implies that it is sufficient to have $S \ge \lceil \log_2(\sum_{k=0}^{L-1} |w_k|) \rceil$ in order to have $|\bar{y}(n)| < 1$ satisfied, where $\lceil . \rceil$ denotes the so-called ceiling function, meaning rounding up to the closest integer.

3. The proposed implementation

Consider the ADFE shown in Fig. 2 that processes an input signal $x(n)$ and generates the output decision $\hat{y}(n)$ as per the following:

$$\hat{y}(n) = \mathrm{Q}[y(n)], \tag{1}$$

$$y(n) = \mathbf{w}^t(n)\boldsymbol{\Phi}(n), \tag{2}$$

$$\mathbf{w}(n) = [\mathbf{w}^{ft}(n)\mathbf{w}^{bt}(n)]^t, \tag{3}$$

$$\boldsymbol{\Phi}(n) = [x(n), \ldots, x(n-p+1), v(n-1), \ldots, v(n-q)]^t, \tag{4}$$

where Q[.] represents quantization, $\mathbf{w}^f(n) = [w_0^f(n), w_1^f(n), \ \ldots, w_{p-1}^f(n)]^t$ is a p-th order feed forward filter (FFF) and $\mathbf{w}^b(n) = [w_1^b(n), w_2^b(n), \ldots, w_q^b(n)]^t$ is a q-th order feedback filter (FBF). The signal $v(n)$ is given by a desired response $d(n)$ during the initial training phase and by $\hat{y}(n)$ during the subsequent decision directed phase. The equalizer coefficients are updated by the LMS algorithm as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \Phi(n) e(n), \tag{5}$$

where $e(n) = v(n) - y(n)$ is the error signal and μ is an appropriate step size. The input $x(n)$ is actually obtained from the received sample, which is, however, first passed through a PGA-ADC combine. The PGA amplifies the received sample by a power of two, say, by 2^{g_n} for effective utilization of the available dynamic range of the ADC. The gain factor g_n is adjusted from time to time to take care of fluctuating signal level, caused by, say, fading. It is assumed that an $r(+1\ sign)$-bit ADC is used, representing the 2^r positive and 2^r negative, discrete voltage levels transmitted by the transmitter. For the present treatment, the ADC output will be treated as a F × P word with magnitude less than one, i.e., the binary point will be assumed after the MSB (i.e., the sign bit). The input $x(n)$ then has the following scaled representation: $x(n) = \bar{x}'(n)2^{-g_n+r}$, where the mantissa $\bar{x}'(n)$ is the ADC output, with $|\bar{x}'(n)| < 1$. [For the special case where the ADC output is $10 \cdots 0$, we do sign extension by one bit and write $x(n) = 1.10 \cdots 0\, 2^{-g_n+r+1}$, thus maintaining $|x'(n)| < 1$.] In the proposed scheme,

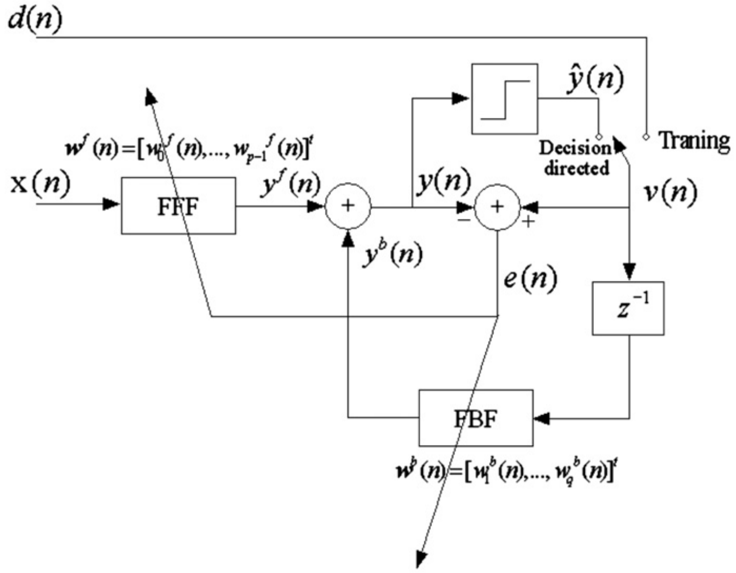

Fig. 2. The basic adaptive decision feedback equalizer.

the equalizer weight vector $\mathbf{w}(n)$ is represented in a BFP format as:

$$\mathbf{w}(n) = [\overline{\mathbf{w}}^{ft}(n)\ \overline{\mathbf{w}}^{bt}(n)]^t 2^{\psi_n}, \tag{6}$$

where $\overline{\mathbf{w}}^f(n)$ and $\overline{\mathbf{w}}^b(n)$ are the mantissa vectors for the FFF and the FBF respectively. The integer ψ_n is the time-varying block exponent which needs to be updated at each index n and is chosen to ensure that $|\overline{w}_m^f(n)| < \frac{1}{2}$ for $m \in Z_p$ and $|\overline{w}_{l+1}^b(n)| < \frac{1}{2}$ for $l \in Z_q$.

The proposed implementation: The proposed BFP realization consists of three stages, namely

(i) *Buffering*: Here, the input sequence $x(n)$ is partitioned into non-overlapping blocks of length N each, with the i-th block given by $\{x(n)|n \in Z_i'\}$, where $Z_i' = \{iN, iN+1, \ldots, iN+N-1\}, i \in Z$. For this, the input is shifted into a buffer of size N. We take $N \geq p-1$, as otherwise, the input vector $\mathbf{x}(n)$ will involve data from three or more adjacent blocks and thus the complexity of implementation would go up. The buffer is cleared and its contents transferred to a block formatter once in every N input clock cycles.

(ii) *Block formatting of input*: Here, the data samples $x(n)$ constituting the i-th block, $i \in Z$ and available in FP form, are block formatted following the treatment of Section 2, resulting in the BFP representation: $x(n) = \overline{x}(n)2^{\gamma_i}$, $n \in Z_i'$, where $\gamma_i = ex_i + S_i$, $ex_i = \lfloor \log_2 M_i \rfloor + 1$, $M_i = \max\{|x(n)| \,|n \in Z_i'\}$. Then, for the case when each element of the vector $\mathbf{x}(n)$ originates from the i-th block, we can express $\mathbf{x}(n)$ in BFP form as $\mathbf{x}(n) = \overline{\mathbf{x}}(n)2^{\gamma_i}$ ($\overline{\mathbf{x}}(n) = [\overline{x}(n), \cdots, \overline{x}(n-p+1)]$) and the FFF output $y^f(n)$ as $y^f(n) = \overline{y}^f(n)2^{\gamma_i+\psi_n}$, where $\overline{y}^f(n) = \overline{\mathbf{w}}^{ft}(n)\overline{\mathbf{x}}(n)$ denotes the FFF output mantissa. For no overflow in $\overline{y}^f(n)$, it is required that $|\overline{y}^f(n)| < 1$. However, in the proposed scheme, we restrict $\overline{y}^f(n)$ to lie between $+\frac{1}{4}$ and $-\frac{1}{4}$. From Section 2 and also from the fact that $|\overline{w}_m^f(n)| < \frac{1}{2}$, $m \in Z_p$, this implies a lower limit of S_i as $S_{min} = \lceil \log_2 2p \rceil$. During the block-to-block transition phase, i.e.,for the indices $n = iN$ to $n = iN+p-2$, however, part of $\mathbf{x}(n)$ comes from the i-th block and part from the $(i-1)$-th block. To retain the BFP form of $\mathbf{x}(n)$ as $\mathbf{x}(n) = \overline{\mathbf{x}}(n)2^{\gamma_i}$ at these indices as well, we rescale the elements $\overline{x}(iN-p+1), \cdots, \overline{x}(iN-1)$ by dividing them by $2^{\Delta\gamma_i}$, where $\Delta\gamma_i = \gamma_i - \gamma_{i-1}$. Equivalently, for the elements $x(iN-p+1), \cdots, x(iN-1)$, the scaling factor S_{i-1} is changed to an effective scaling factor of $S_{i-1}' = S_{i-1} + \Delta\gamma_i$. The scaling factor for the i-th block, S_i is assigned by taking it to be the *minimum integer required to satisfy both* $S_i \geq S_{min}$ *and* $S_{i-1}' \geq S_{min}$. This is realized by using the exponent assignment algorithm presented in [12], which sets $S_i = S_{min}$ (i.e., $\gamma_i = ex_i + S_{min}$) if $ex_i \geq ex_{i-1}$, else (i.e., for $ex_i < ex_{i-1}$) $S_i = (ex_{i-1} - ex_i) + S_{min}$, (i.e., $\gamma_i = ex_{i-1} + S_{min}$).

(iii) *Equalization and weight updating*: This consists of four main computations, namely

(a) *FFF output*: The block formatter inputs (i) $\overline{x}(n)$, $n \in Z_i'$, (ii) the rescaled mantissas for $x(iN-k)$, $k = 1, 2, \ldots, p-1$, and (iii) the block exponent γ_i to the FFF, which computes the output exponent as $\gamma_i + \psi_n$, and the output mantissa $\overline{y}^f(n) = \overline{\mathbf{w}}^{ft}(n)\overline{\mathbf{x}}(n), \forall n \in Z_i'$. The FFF output $y^f(n)$ is then given as $y^f(n) = \overline{y}^f(n)2^{\gamma_i+\psi_n}, n \in Z_i'$.

(b) *FBF output*: Unlike the FFF, BFP computation of the FBF output $y^b(n) = \mathbf{w}^{bt}(n)\mathbf{v}(n-1)$ would require block formatting of the decision vector $\mathbf{v}(n-1) = [v(n-1), \ldots, v(n-q)]^t$ at each index n. This is done by modifying the block formatting algorithm of [16] suitably as discussed below. However, like the FFF, the FBF output is also constrained to satisfy $|\overline{y}^b(n)| < \frac{1}{4}$ where $\overline{y}^b(n)$ denotes the mantissa of $y^b(n)$. The corresponding scaling factor, say, S' is then required to satisfy $S' \geq S_{min}' = \lceil \log_2 2q \rceil$. For minimum loss of bits, we choose $S' = S_{min}'$. If $v(n)$ is represented by $r(+1\ sign)$ bit FxP numbers, right shift of $v(n)$ by S_{min}' may, however, result in the loss of many significant bits, or, even flushing of the register to zero, for small values of $|v(n)|$. To avoid this, we assume that the discrete levels of the quantizer and also the samples of the desired response $d(n)$, both representing the transmitted symbols, are stored in a normalized, scaled format, with mantissas scaled down by $2^{S_{min}'}$. This means that $v(n)$ is available in a normalized FP form as $v(n) = \overline{v}(n)2^{e_{v(n)}+S_{min}'}$ with $2^{-S_{min}'-1} \leq |\overline{v}(n)| < 2^{-S_{min}'}$. From above, it follows that at each index n, computation of $y^b(n)$ involves the following data:

(a) $\{v(n-l), l = 2, 3, \ldots, q\}$, stored in the FBF in a block formatted form as $v(n-l) = \overline{v}_{n-1}(n-l)2^{v_{n-1}}, |\overline{v}_{n-1}(n-l)| < 2^{-S_{min}'}$ with v_{n-1} and $\overline{v}_{n-1}(n-l)$ denoting respectively the block exponent at the $(n-1)$-th index and the corresponding mantissa of $v(n-l)$.

(b) The latest input to the FBF, $v(n-1) = \overline{v}(n-1)2^{e_{v(n-1)}+S_{min}'}$ with $2^{-S_{min}'-1} \leq |\overline{v}(n-1)| < 2^{-S_{min}'}$. Block formatting of the data in (a) and (b), namely $\{v(n-l)|l = 1, 2, \ldots, q\}$ can then be carried out by appropriately modifying the "Block Formatting Algorithm" of [16], generating new block exponent v_n and corresponding mantissas, $\{\overline{v}_n(n-l)|l = 1, 2, \ldots, q\}$, as explained below:

Algorithm for block formatting the FBF input Given $v(n-l) = M_l 2^{e_l}$, $l = 1, 2, \ldots, q$, where, for $l = 1$, $M_l = \overline{v}(n-1)$, $e_l = e_{v(n-1)} + S_{min}'$ with $2^{-S_{min}'-1} \leq |M_l| < 2^{-S_{min}'}$, and, for $l = 2, 3, \ldots, q$, $M_l = \overline{v}_{n-1}(n-l)$, $e_l = v_{n-1}$ with $|M_l| < 2^{-S_{min}'}$, carry out the following steps:

(i) Count the number, say, n_l of binary 0's (if $v(n-l)$ is positive) or binary 1's (if $v(n-l)$ is negative) between the binary point of M_l and the first binary 1 or binary 0 from left respectively. Clearly, $n_l = S_{min}'$ for $l=1$ and $n_l \geq S_{min}'$ for $l = 2, 3, \ldots, q$. Define $n_l' = n_l - S_{min}'$.

(ii) Compute $v_n = \max\{e_{\nu(n-1)} + S'_{min},\ v_{n-1} - n'_2, \ldots, v_{n-1} - n_q{}'\} = \max\{e_{\nu(n-1)} + S'_{min},\ \max\{v_{n-1} - n_l{}' | l = 2,3,\ldots,q\}\} = \max\{e_{\nu(n-1)} + S'_{min},\ v_{n-1} - \min\{n_l{}' | l = 2,3,\ldots,q\}\}$. It is easy to verify from above that $v_n \geq e_{\nu(n-1)} + S'_{min}$.

(iii) Shift each M_l right or left by $(v_n - e_l)$ bits depending on whether $|v_n - e_l|$ is positive or negative respectively, thus generating $\overline{v}_n(n-l)$.

The FBF output is then obtained as $y^b(n) = \overline{y}^b(n) 2^{\psi_n + v_n}$, with the mantissa given as $\overline{y}^b(n) = \overline{\mathbf{w}}^{bt}(n)\overline{\mathbf{v}}_n(n-1)$, $\overline{\mathbf{v}}_n(n-1) = [\overline{v}_n(n-1), \overline{v}_n(n-2), \ldots, \overline{v}_n(n-q)]^t$.

(c) *Error e(n)*: Evaluation of $e(n)$ requires two FP based additions, first of them being $y(n) = y^f(n) + y^b(n) = \overline{y}(n) 2^{\xi_n}$, where the mantissa $\overline{y}(n)$ and the exponent ξ_n are computed as,
If $\gamma_i + \psi_n > v_n + \psi_n$
$\overline{y}(n) = \overline{y}^f(n) + \overline{y}^b(n) 2^{v_n - \gamma_i}$, $\xi_n = \gamma_i + \psi_n$,
else
$\overline{y}(n) = \overline{y}^b(n) + \overline{y}^f(n) 2^{\gamma_i - v_n}$, $\xi_n = v_n + \psi_n$. It is easy to check that $|\overline{y}(n)| < \frac{1}{2}$, meaning there is no overflow in $\overline{y}(n)$. The other FP addition computes $e(n) = \nu(n) - y(n) = \overline{v}(n) 2^{e_{\nu(n)} + S'_{min}} - \overline{y}(n) 2^{\xi_n} = \overline{e}(n) 2^{\theta_n}$, where the mantissa $\overline{e}(n)$ and the exponent θ_n are evaluated as,
If $e_{\nu(n)} + S'_{min} > \xi_n$
$\overline{e}(n) = \overline{v}(n) - \overline{y}(n) 2^{\xi_n - e_{\nu(n)} - S'_{min}}, \theta_n = e_{\nu(n)} + S'_{min}$,
else
$\overline{e}(n) = \overline{v}(n) 2^{e_{\nu(n)} + S'_{min} - \xi_n} - \overline{y}(n), \theta_n = \xi_n$.
In either case, $|\overline{e}(n)| < |\overline{v}(n)| + |\overline{y}(n)|$. Since $|\overline{v}(n)| < 2^{-S'_{min}}$ and $S'_{min} = \lceil \log_2 2q \rceil$, we have, $|\overline{v}(n)| < 1/2q$. Thus, $|\overline{e}(n)| < (1/2q) + (1/2) \leq 1$, meaning $\overline{e}(n)$ is free of overflows.

(d) *Weight updating*: For updating $\mathbf{w}(n)$, we first try to express $\mathbf{w}^f(n+1)$ and $\mathbf{w}^b(n+1)$ as $\mathbf{w}^f(n+1) = \overline{\mathbf{u}}^f(n) 2^{\psi_n}$ and $\mathbf{w}^b(n+1) = \overline{\mathbf{u}}^b(n) 2^{\psi_n}$ for some appropriate $\overline{\mathbf{u}}^f(n)$ and $\overline{\mathbf{u}}^b(n)$ that are constrained as $|\overline{u}^f_m(n)| < 1, |\overline{u}^b_{l+1}(n)| < 1$, $m \in Z_p$, $l \in Z_q$. Then, if each $\overline{u}^f_m(n)$ and each $\overline{u}^b_{l+1}(n)$ lie within $\pm \frac{1}{2}$, we make the assignments

$$\overline{\mathbf{w}}(n+1) = [\overline{\mathbf{u}}^f(n) \quad \overline{\mathbf{u}}^b(n)]^t, \quad \psi_{n+1} = \psi_n. \tag{7}$$

Otherwise, we scale down $\overline{\mathbf{u}}^f(n)$ and $\overline{\mathbf{u}}^b(n)$ by 2, meaning,

$$\overline{\mathbf{w}}(n+1) = \tfrac{1}{2}[\overline{\mathbf{u}}^f(n)\ \overline{\mathbf{u}}^b(n)]^t, \quad \psi_{n+1} = \psi_n + 1. \tag{8}$$

To express $\mathbf{w}^f(n+1)$ and $\mathbf{w}^b(n+1)$ in the above form, we need to substitute $e(n)$, $\mathbf{x}(n)$, $\mathbf{v}(n-1)$, $\mathbf{w}^f(n)$ and $\mathbf{w}^b(n)$ by their respective scaled representations in (5). This results in the following general expressions for $\overline{\mathbf{u}}^f(n)$ and $\overline{\mathbf{u}}^b(n)$:

$$\overline{\mathbf{u}}^f(n) = \overline{\mathbf{w}}^f(n) + \mu \overline{\mathbf{x}}(n) 2^{\gamma_i} \overline{e}(n) 2^{\theta_n - \psi_n}, \tag{9}$$

$$\overline{\mathbf{u}}^b(n) = \overline{\mathbf{w}}^b(n) + \mu \overline{\mathbf{v}}_n(n-1) 2^{v_n} \overline{e}(n) 2^{\theta_n - \psi_n}. \tag{10}$$

To ensure that $|\overline{u}^f_m(n)| < 1$, $m \in Z_p$, we observe that $|\overline{u}^f_m(n)| \leq |\overline{w}^f_m(n)| + \mu |\overline{x}(n-m)| 2^{\gamma_i} |\overline{e}(n)| 2^{\theta_n - \psi_n}$. Since $|\overline{w}^f_m(n)| < \frac{1}{2}$, $m \in Z_p$, it is sufficient to have $\mu |\overline{x}(n-m)| 2^{\gamma_i} |\overline{e}(n)|\ 2^{\theta_n - \psi_n} \leq \frac{1}{2}$ in order to satisfy $|\overline{u}^f_m(n)| < 1$.Now, $|\overline{e}(n)| 2^{\theta_n} \equiv |e(n)| \leq |\overline{v}(n)| 2^{e_{\nu(n)} + S'_{min}} + |\overline{y}(n)| 2^{\xi_n} < 2^{e_{\nu(n)}} + |\overline{y}(n)| 2^{\xi_n}$. Again, $|\overline{y}(n)| 2^{\xi_n} \equiv |y(n)| \leq |\overline{y}^f(n)| 2^{\gamma_i + \psi_n} + |\overline{y}^b(n)| 2^{v_n + \psi_n} < \frac{p}{2} 2^{ex_i + \psi_n} + \frac{q}{2} 2^{-S'_{min} + v_n + \psi_n}$, meaning

$$|\overline{e}(n)| 2^{\theta_n} < 2^{e_{\nu(n)}} + \frac{p}{2} 2^{ex_i + \psi_n} + \frac{q}{2} 2^{-S'_{min} + v_n + \psi_n}. \tag{11}$$

From this and noting that $|\overline{x}(n-m)| 2^{\gamma_i} < 2^{ex_i}$, it is then sufficient to have

$$\mu \leq \frac{1}{2^{ex_i - \psi_n + e_{\nu(n)} + 1} + p 2^{2ex_i} + q 2^{ex_i - S'_{min} + v_n}}, \tag{12}$$

for satisfying $\mu |\overline{x}(n-m)| 2^{\gamma_i} |\overline{e}(n)| 2^{\theta_n - \psi_n} \leq \frac{1}{2}$, $m \in Z_p$ and thus $|\overline{u}^f_m(n)| < 1$. It is easy to check that the above bound for μ is valid not only when each element of $\overline{\mathbf{x}}(n)$ comes purely from the i-th block but also during transition from the $(i-1)$-th to

Table 1
The expressions for $\overline{\mathbf{u}}^f(n)$ and $\overline{\mathbf{u}}^b(n)$ for four different cases $\overline{e}'(n) = \overline{e}(n) 2^{S'_{min} + e_{\nu(n)} - \psi_n}$.

Case	$\overline{\mathbf{u}}^f(n)$	$\overline{\mathbf{u}}^b(n)$
I: $(\gamma_i + \Psi_n) > (\nu_n + \Psi_n)$ and $\xi_n > e_\nu(n) + S'_{min}$	$\overline{\mathbf{w}}^f(n) + \mu \overline{x}(n) \overline{e}(n) 2^{2^{\gamma_i}}$	$\overline{\mathbf{w}}^b(n) + \mu \overline{\nu}_n(n-1) \overline{e}(n) 2^{r_i + \nu_n}$
II: $(\gamma_i + \Psi_n) \leq (\nu_n + \Psi_n)$ and $\xi > e_\nu(n) + S'_{min}$	$\overline{\mathbf{w}}^f(n) + \mu \overline{x}(n) \overline{e}(n) 2^{r_i + \nu_n}$	$\overline{\mathbf{w}}^b(n) + \mu \overline{\nu}_n(n-1) \overline{e}(n) 2^{2\nu_n}$
III,IV: $(\gamma_i + \Psi_n) > (\leq)(\nu_n + \Psi_n)$ and $\xi_n \leq e_\nu(n) + S'_{min}$	$\overline{\mathbf{w}}^f(n) + \mu \overline{x}(n) \overline{e}'(n) 2^{r_i}$	$\overline{\mathbf{w}}^b(n) + \mu \overline{\nu}(n-1) \overline{e}'(n) 2^{\nu_n}$

i-th block with $ex_i \geq ex_{i-1}$, for which after necessary rescaling, we have $S'_{i-1} \geq S_i = S_{min}$, implying $|\overline{x}(n-m)| < 2^{-S_i}$ and thus $|\overline{y}^f(n)| < (p/2)2^{-S_i}$. For $ex_i < ex_{i-1}$, however the upper bound expression (12) gets modified with ex_i replaced by ex_{i-1}, as in that case, we have $\gamma_i = ex_{i-1} + S'_{i-1}$ with $S'_{i-1} = S_{min} < S_i$ meaning $|\overline{x}(n-m)| < 2^{-S'_{i-1}}$ and thus $|\overline{y}^f(n)| < (p/2)2^{-S'_{i-1}}$. From above, we obtain a general upper bound for μ by equating ψ_n to its lowest value of zero, ν_n and $e_{\nu(n)}$ to their highest values of $r+1+S'_{min}$ and $r+1$ respectively and replacing ex_i by $ex_{max} = \max\{ex_i \mid i \in Z\}$. The general upper bound is given by

$$\mu \leq \mu^f = \frac{1}{2^{ex_{max}+r+2} + p2^{2ex_{max}} + q2^{ex_{max}+r+1}}. \tag{13}$$

Similarly, from (10) and using analogous arguments, it follows that in order to satisfy $|\overline{u}^b_{l+1}(n)| < 1$, $l \in Z_q$, it is sufficient to have $\mu|\overline{v}_n(n-l-1)|2^{\nu_n}|\overline{e}(n)|2^{\theta_n - \psi_n} \leq \frac{1}{2}$. From (11) and recalling that $|\overline{v}_n(n-l-1)| < 2^{-S'_{min}}$, $l \in Z_q$, it is then sufficient to have

$$\mu \leq \frac{1}{2^{-S'_{min}+\nu_n}[2^{-\psi_n + e_{\nu(n)}+1} + p2^{ex_i} + q2^{-S'_{min}+\nu_n}]}, \tag{14}$$

for satisfying $\mu|\overline{v}_n(n-l-1)|2^{\nu_n}|\overline{e}(n)|2^{\theta_n - \psi_n} \leq \frac{1}{2}, l \in Z_q$ and thus $|\overline{u}^b_{l+1}(n)| < 1$. Again, following the arguments used above, the exponent ex_i in (14) is to be replaced by ex_{i-1} for the block-to-block transition case with $ex_i < ex_{i-1}$. The general upper bound in this case is given by

$$\mu \leq \mu^b = \frac{1}{2^{2r+3} + p2^{ex_{max}+r+1} + q2^{2r+2}}. \tag{15}$$

The final choice of μ will be made following $\mu \leq \min\{\mu^f, \mu^b\}$.

Table 2

Summary of the LMS based ADFE realized in BFP format (initial value of $\psi_n = 0$, $\overline{\mathbf{w}}(n) = [\overline{\mathbf{w}}^{ft}(n)\overline{\mathbf{w}}^{bt}(n)]^t = \mathbf{0}$).

```
1. Preprocessing:
Using the data x(n) for the i-th block, $n \in z'_i = \{iN, iN+1, \ldots, iN+N-1\}$ (stored during processing of the (i−1)-th block):
(a) Evaluate block exponent $\gamma_i$ as per the Block Formatting Algorithm of [12] and express x(n), $n \in z'_i$ as $x(n) = \overline{x}(n)2^{\gamma i}$
(b) Rescale the following elements of the (i−1)-th block:
$\{\overline{x}(n) | n = \{iN-p+1, \ldots, iN-1\}$ as $\overline{x}(n) \to \overline{x}(n)2^{-\Delta\gamma_i}, \Delta\gamma_i = \gamma_i - \gamma_{i-1}$
2. Processing for the i-th block: (i.e., $n \in z'_i$)
(a) (i) Feed Forward Filter output: $\overline{y}^f(n) = \overline{w}^{ft}(n)\overline{x}(n), ex^f(n) = \gamma_i + \Psi_n(ex^f(n)$ is the FFF output exponent).
(ii) Feedback Filter output:
Using the data stored in the FBF barrel, $\nu(n-1) = \overline{v}_{n-1}(n-l)2^{\nu_{n-1}}, l = 2,3,\ldots,q$ and the latest decision $\nu(n-1) = \overline{v}(n-1)2^{e_{\nu(n-1)}+S'_{min}}$ generate new block exponent $\nu_n$ and corresponding mantisses, $\{\overline{v}(n-l) | l = 1,2,\ldots,q\}$ as per the "Algorithm for Block Formatting the FBF input" (Section 3).
$\overline{y}^b(n) = \overline{w}^{bt}(n)\overline{v}(n-1), ex^b(n) = \nu_n + \Psi_n(ex^f(n)$ is the FFF output exponent).
(iii) output (pre-decision) y(n):
Compute y(n) as $y(n) = \overline{y}(n)2^{\xi_n}$ where
if $ex^f(n) > ex^b(n)$
$\overline{y}(n) = \overline{y}^f(n) + \overline{y}^b(n)2^{\nu_n - \gamma_i}$   and   $\xi_n = \Psi_n + \gamma_i$
else
$\overline{y}(n) = \overline{y}^f(n)2^{\gamma_i - \nu_n} + \overline{y}^b(n)$   and   $\xi_n = \Psi_n + \nu_i$
end
(b) Output error computation:
$e(n) = \nu(n) - y(n) = \overline{e}(n)2^{\theta_n}$, where
if $e_\nu(n) + S'_{min} > \xi_n$
$\overline{e}(n) = \overline{v}(n) - \overline{y}(n)2^{\xi_n - e_\nu(n) - S'_{min}}$   and   $\Theta_n = e_{\nu(n)} + S'_{min}$
else
$\overline{e}(n) = \overline{v}(n)2^{e_\nu(n) - S'_{min} - \xi_n} - \overline{y}(n)$   and   $\Theta_n = \xi_n$
end.
(c) Weight updating:
Compute $\mu\overline{x}(n)2^{\gamma_i}\overline{e}(n)2^{\theta_n - \Psi_n} = \overline{x}(n)$ (say),
as per the steps (lf)–(3b) of Section 3.
Compute $\mu\overline{v}(n-1)2^{\nu_n}\overline{e}(n)2^{\theta_n - \Psi_n} = \overline{v}_1(n-1)$ (say),
as per the steps (lb)–(3b) of Section 3.
Evaluate $\overline{u}^f(n) = \overline{w}^f(n) + \overline{x}^1(n)$,
Evaluate $\overline{u}^b(n) = \overline{w}^b(n) + \overline{v}^1(n-1)$,
if $|\overline{u}^{-f}_m(n)| < \frac{1}{2}$   for all $m \in Z_p$   and $|\overline{u}^{-b}_{l+1}(n)| < \frac{1}{2}$   for all $l \in Z_l$
$\overline{w}(n+1) = [\overline{u}^f(n)\overline{u}^b(n)]^t, \Psi_{n+1} = \Psi_n$
else
$\overline{w}(n+1) = \frac{1}{2}[\overline{u}^f(n)\overline{u}^b(n)]^t, \Psi_{n+1} = \Psi_n + 1$
end
i=i+1
Repeat steps 1–2
```

The two bounds, μ^f and μ^b are easily seen to be related by a simple constant, i.e., $\mu^f/\mu^b = 2^{r+1-ex_{max}}$.

To estimate ex_{max}, we can write from Fig. 1, $ex_{max} = \max\{r+1-g_n | n \in Z\} \equiv r+1-\min\{g_n | n \in Z\}$. Since in all practical applications, $g_n \geq 0$, we have, $ex_{max} \leq r+1$, meaning $\mu^f/\mu^b = 2^{r+1-ex_{max}} \geq 1$, or equivalently, $\mu^f \geq \mu^b$.

The lower of the two bounds μ^f and μ^b, namely μ^b is actually less than $2/\text{tr}\,\mathbf{R}$ which is the upper bound for μ for convergence of the LMS based weight update iteration (6). To see this, we note that $|x(n)| < 2^{ex_{max}}$ and thus $E[x^2(n)] < 2^{2ex_{max}}$. Again, $|\nu(n)| \leq 2^r$ meaning $E[\nu^2(n)] < 2^{2r}$. Together, this implies $\text{tr}\,\mathbf{R} < p2^{2ex_{max}} + q2^{2r}$, meaning $(2/\text{tr}\,\mathbf{R}) > 1/(p2^{2ex_{max}} + q2^{2r}) > \mu_b$, where we have used the fact that $ex_{max} \leq r+1$. Eqs. (9) and (10) provide general expressions for $\overline{\mathbf{u}}^f(n)$ and $\overline{\mathbf{u}}^b(n)$ respectively. However, since θ_n assumes four different values depending on whether $(\gamma_i + \psi_n) > (\leq)\ (\nu_n + \psi_n)$ and whether $\xi_n > (\leq)\ e_{\nu(n)} + S'_{min}$, in practice, we can have four different expressions for $\overline{\mathbf{u}}^f(n)$ and $\overline{\mathbf{u}}^b(n)$ each. These are listed in Table 1, where, it may be seen that under cases III and IV, $\overline{\mathbf{u}}^f(n)$ and also $\overline{\mathbf{u}}^b(n)$ assume the same form. Also note that in (9) and (10), computation of the update terms $\mu\overline{\mathbf{x}}(n)2^{\gamma_i}\overline{e}(n)2^{\theta_n-\psi_n}$ and $\mu\overline{\mathbf{v}}_n(n-1)2^{\nu_n}\overline{e}(n)2^{\theta_n-\psi_n}$ involve several multiplications. These need to be carried out in such a way that no overflow occurs in any of the intermediate products or shift operations involved. At the same time, we need to avoid direct product of quantities which could be very small, as that may lead to loss of several useful bits via truncation. One possible way to realize this is given by the following steps ($m \in Z_p$, $l \in Z_q$):

Feed forward filter:
Weight update term: $\mu\overline{\mathbf{x}}(n)2^{\gamma_i}\overline{e}(n)2^{\theta_n-\psi_n}$
step (1f): $\mu 2^{ex_i} = \mu_1^f$(say),
step (2f): $[\mu_1^f\overline{e}(n)]2^{\theta_n-\psi_n} = \overline{e}_1^f(n)$(say), and
step (3f): $\overline{e}_1^f(n)[\overline{x}(n-m)2^{S_i}]$. [For the block-to-block transition case with $ex_i < ex_{i-1}$, replace ex_i in step (1f) and S_i in step (3f) by ex_{i-1} and $S'_{i-1} = S_{min}$ respectively.]
Feedback filter:
Weight update term: $\mu\overline{\mathbf{v}}_n(n-1)2^{\nu_n}\overline{e}(n)2^{\theta_n-\psi_n}$
step (1b): $\mu 2^{\nu_n - S'_{min}} = \mu_1^b$ (say),
step (2b): $[\mu_1^b\overline{e}(n)]2^{\theta_n-\psi_n} = \overline{e}_1^b(n)$(say), and
step (3b): $\overline{e}_1^b(n)[\overline{\nu}(n-l-1)2^{S'_{min}}]$.

The proposed BFP treatment to the ADFE is summarized in Table 2. Note that the three stages: (i) buffering, (ii) block formatting (of input) and (iii) equalization and weight updating are *pipelined*, with their relative timing shown in Fig. 3.

4. Complexity issues

The proposed schemes require processing complexity much less than that required by their FP-based counterparts, as they rely mostly on FxP arithmetic and thus are largely free from the usual FP operations like shift, exponent comparison,

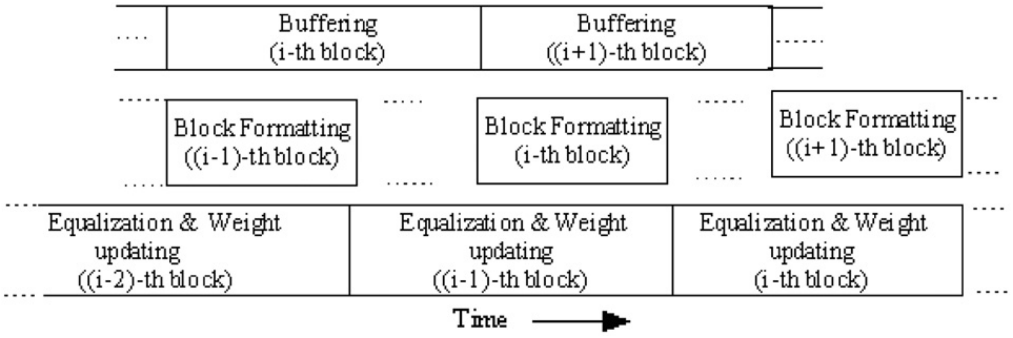
Buffering (i-th block)
Buffering ((i+1)-th block)
Block Formatting ((i-1)-th block)
Block Formatting (i-th block)
Block Formatting ((i+1)-th block)
Equalization & Weight updating ((i-2)-th block)
Equalization & Weight updating ((i-1)-th block)
Equalization & Weight updating (i-th block)
Time

Fig. 3. The relative timing of the three units.

Table 3
A comparison between the BFP vis-à-vis the FP-based realizations of the ADFE. Number of operations required per iteration for (a) weight updating and (b) filtering are shown. Unless specified otherwise, all the general operations indicate mantissa operations. ($l=p+q$, MAC=Multiply and Accumulate, MC=Magnitude Check, EC=Exponent Check, EA=Exponent Addition).

(a)	MAC	Shift	MC	EC	EA
BFP	$l+2$	$2l+4$	l	Nil	1
FP	$l+1$	$2l+1$	Nil	l	$2l+2$
(b)	MAC	Shift		EC	EA
BFP	l	q − 1		q − 1	2
FP	l	$2l$		l	$2l$

exponent addition, etc. For example, to compute the FFF output $y^f(n)$, the BFP method requires only p FxP based "Multiply and Accumulate (MAC)" operations for evaluating $\overline{y}^f(n) = \overline{\mathbf{w}}^{ft}(n)\overline{\mathbf{x}}(n)$ and, at the most, one exponent addition for evaluating the exponent $\gamma_i + \psi_n$, at each index n. In FP, this would, however, require p FP-based MAC operations. Note that given three numbers in FP (normalized) format: $A = \overline{A}2^{e_a}, B = \overline{B}2^{e_b}, C = \overline{C}2^{e_c}$, the MAC operation $A+BC$ requires the following steps: (i) $e_b + e_c$, i.e., exponent addition (EA), (ii) exponent comparison (EC) between e_a and $e_b + e_c$, (iii) Shifting either $\overline{A}$ or $\overline{B}/\overline{C}$, (iv) FxP-based MAC, and finally (v) renormalization, requiring shift and exponent addition. In other words, in FP, computation of $y^f(n)$ will require the following *additional* operations over the BFP-based realization: (a) $2p$ shifts (assuming availability of single cycle barrel shifters), (b) p exponent comparisons, and (c) $2p-1$ exponent additions. Similar advantages exist in the FBF output computation also except that $q-1$ exponent comparisons and a maximum of $q-1$ shifts are required for block formatting the FBF data at each index. For updating the FFF and the FBF weights, the primary computation involves evaluation of $\overline{\mathbf{u}}^f(n)$ and $\overline{\mathbf{u}}^b(n)$. For the proposed LMS-ADFE realization, this requires p and q FxP based MAC operations respectively apart from a few shift operations, as detailed in steps (1f)–(3f) and (1b)–(3b) at the end of Section 3. Additionally, $p+q$ magnitude check operations are needed to check whether each $|\overline{u}^f_m(n)|$, $m \in Z_p$ and $|\overline{u}^b_{l+1}(n)|$, $l \in Z_q$ lies within $\frac{1}{2}$ or not, and another $p+q$ shift operations are necessary for scaling down $\overline{\mathbf{u}}^f(n)$ and $\overline{\mathbf{u}}^b(n)$ by 2 if any $|\overline{u}^f_m(n)|$ or $|\overline{u}^f_{l+1}(n)| > \frac{1}{2}$. In contrast, a FP based realization would require p and q FP based MAC operations respectively for computing the same. For the LMS-ADFE, Table 3 provides a comparativeaccount of the two approaches in terms of number of operations required at each index n. It is easily seen from this table that given a low cost, simple FxP processor with single cycle MAC and barrel shifter units, the proposed scheme is between *two to two and a half times faster* than a FP based implementation.

5. Simulation studies

The proposed scheme developed in Section 3 and given in Table 2 was simulated to study the effects of block formatting of the data and the equalizer coefficients in finite precision on the ADFE performance and to assess it vis-a-vis a FP based realization. For this, a 9-tap FIR channel with a spectral null at 0.8π radian was considered, having impulse response

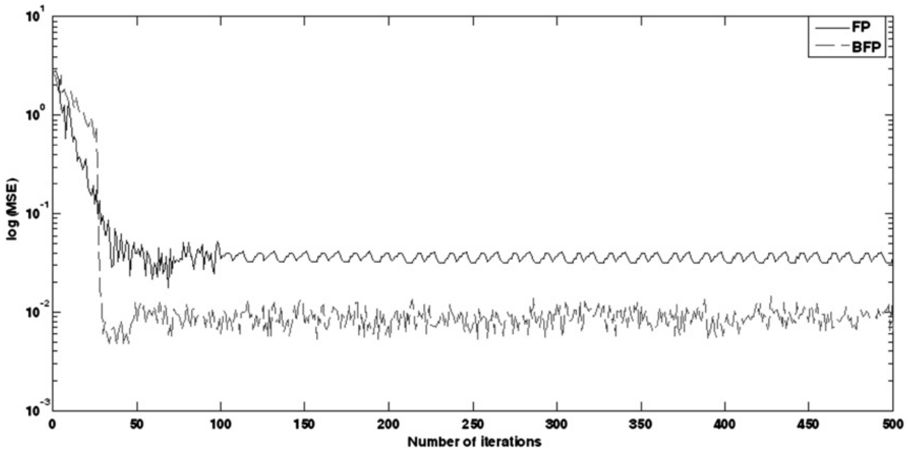

Fig. 4. Learning curves for FP and BFP based ADFE (N=25, register length R=10).

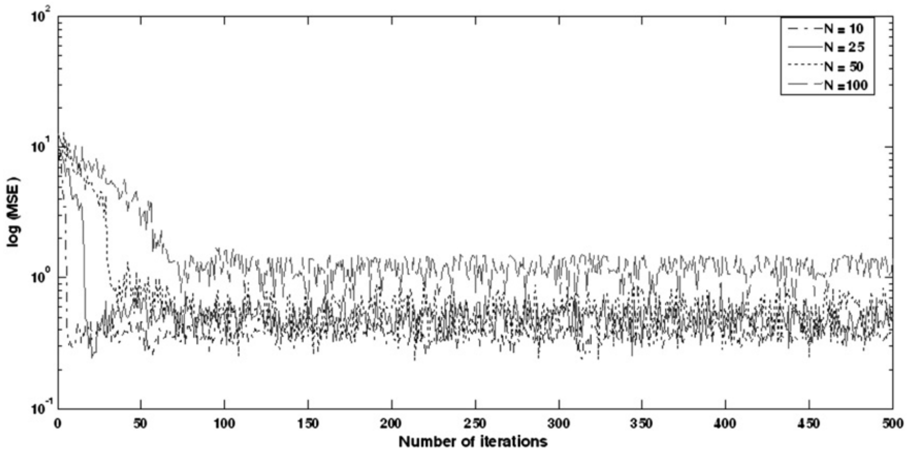

Fig. 5. MSE plots for different block lengths with R=10.

{0.0675 0.103 0.227 0.460 0.688 0.460 0.227 0.103 0.0675}, with an accompanying AWGN of variance 0.1. The transmitted symbols were chosen from an alphabet of eight equispaced, equiprobable discrete amplitude levels, with transmitted signal power of 6 dB. For simplicity, same precision was assumed for the ADC and the quantizer, or, equivalently, for the FFF and the FBF mantissas. The lengths of the FFF and the FBF were chosen as $p=3$ and $q=3$. To evaluate the upper bound of μ, one needs the knowledge of ex_{max}. For this, we considered a received signal model where the minimum g_n needed to raise the signal level to the full dynamic range of the ADC was $(r-1)$, where $r(+1\ sign)$ is the ADC precision. This implies $ex_{max}=r+1-\min\{g_n|n\in Z\}=2$, resulting in $\mu^f=0.0048$ from (13), $\mu^b=0.0024$ from (15) and thus, $\mu\le\min\{\mu^f,\mu^b\}=0.0024$. Taking $\mu=0.001$, the ADFE was first simulated using the proposed BFP scheme, choosing block length N as 25 and allocating 10 (i.e., $1+9$) bits to the FFF and to the FBF coefficient mantissas as well as to the data mantissas, and 4 (i.e., $1+3$) bits to the exponents of all. The ADFE was operated in training mode for the first 100 iterations and then, switched over to the decision directed mode for the subsequent iterations. The corresponding learning curve is obtained by plotting the MSE (dB) versus the number of iterations. This is shown by the dotted line in Fig. 4, which also displays the learning curve obtained under a FP based realization by the solid line. It is easily seen from Fig. 4 that the rates of convergence under FP and BFP are comparable. However, the steady state MSE in the case of BFP, though within acceptable ranges, is larger than under FP, which is expected since the quantization noise effects are more pronounced in BFP than in FP, owing to the block formatting of data and filter coefficients. Next, keeping the register length constant at $R=10$, the MSE curves were plotted for different block lengths of $N=10$, 25, 50 and 100 in Fig. 5. Clearly, the steady state MSE increases with N. This can be explained from the fact that with increasing block length, it is more likely to have larger spread in the magnitudes of the data samples in the block, specially for data samples that are situated far away from each other, resulting in more pronounced quantization noise effects via the block formatting process. Again, keeping the block length constant at $N=25$, the MSE curves were plotted for different register lengths of $R=6$, 8, 10 and 12 in Fig. 6. Clearly, the performance becomes better both in convergence speed as well as excess MSE with increasing R which is due to the fact that increasing register length implies higher signal to quantization noise ratio. It is also easily observed from Fig. 6 that the improvement in convergence speed and excess MSE is prominent mostly in the range of $R=6$–10, while the improvement is marginal as R is increased further. In steady state, ideally the weights remain constant and the ADFE

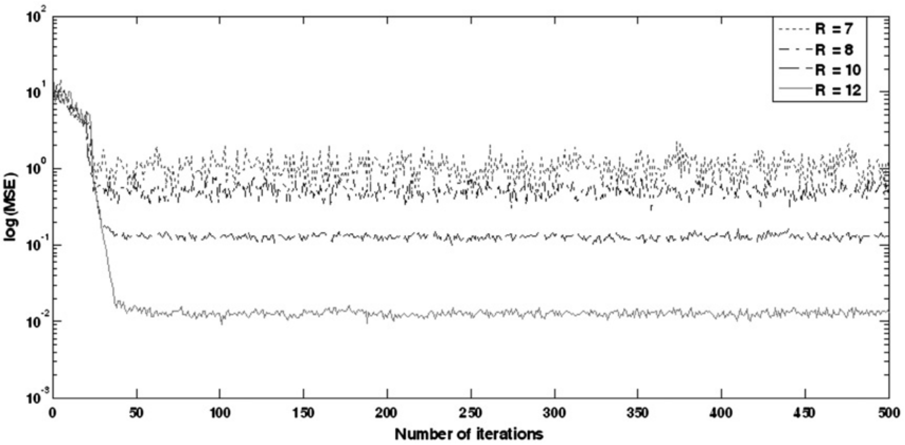

Fig. 6. MSE plots for different register lengths with $N=25$.

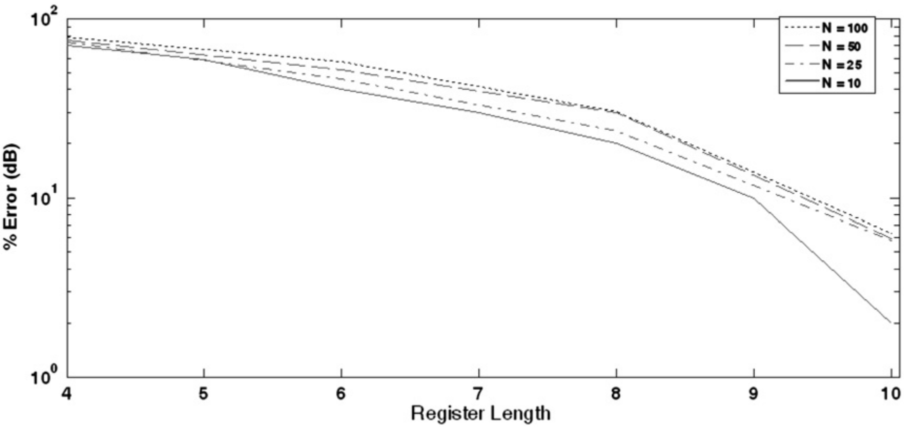

Fig. 7. Variation of error with register length for different N.

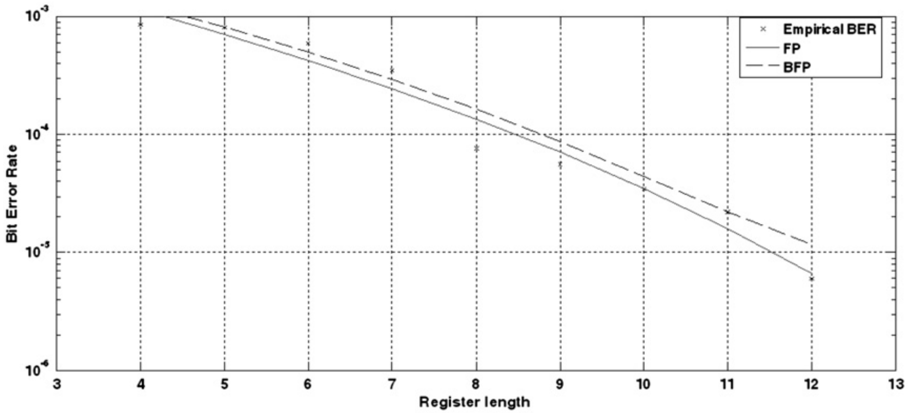

Fig. 8. BER versus register length for both the FP and the proposed BFP based method (using best curve fitting and for $N=25$).

behaves like a simple DFE. Using the steady state weights, the percentage of decision errors was plotted against register length R, with block length N as a parameter. The corresponding plots shown in Fig. 7 confirm that the percentage error decreases with decreasing N for a constant R and with increasing R for each N. In all the simulations, virtually no error was observed in the output decision for $R \geq 12$. Finally, to provide further practical insights, we also plotted the bit error rate (BER) against the register length in Fig. 8, for both the FP and the proposed BFP based method. Fig. 8 clearly shows that for any register length, the BER for the two methods are not much different.

6. Conclusion

An efficient scheme to implement the LMS-based ADFE in finite precision using BFP arithmetic is presented. The proposed scheme adopts appropriate BFP format for both data and filter coefficients and recasts the ADFE equations in terms of the respective mantissas and exponents. Block processing is applied to the input which is block formatted by an efficient algorithm. For the FBF, however, no block processing of its input is possible as that leads to non-causality. Instead, the FBF data is block formatted at every index by a suitable modification of the proposed block formatting algorithm. Overflows at the FFF and the FBF output are prevented by certain dynamic scaling of the respective data. Prevention of overflow in weight update calculations, however, imposes an upper bound on the step size μ which is shown to be less than the upper bound for convergence of the algorithm. The proposed realization deploys mostly simple FxP operations and are largely free from the usual FP operations like shift, exponent comparison, exponent addition, etc., resulting in considerable speed up over their FP based counterpart.

References

[1] S. Haykin, Adaptive Filter Theory, Prentice-Hall, Englewood Cliffs, NJ, 1986.
[2] K. Georgoulakis, G. Glentis, An efficient decision feedback equalizer for the ATSC DTV receiver, Signal Processing 91 (2011) 2671–2676.
[3] R. Arablouei, Kutluyil Doğancay, Low-complexity adaptive decision-feedback equalization of MIMO channels, Signal Processing 92 (2012) 1514–1524.
[4] Y.-C. Lin, S.-J. Jou, M.-T. Shiue, High throughput concurrent lookahead adaptive decision feedback equialiser, IET Circuits Devices System 6 (1) (January 2012) 52–62.
[5] M. Magarini, L. Barletta, A. Spalvieri, Efficient computation of the feedback filter for the hybrid decision feedback equalizer in highly dispersive channels, IEEE Transactions on Wireless Communications 11 (6) (June 2012) 2245–2253.
[6] K.R. Ralev, P.H. Bauer, New tools for localization of limit cycles in recursive block floating point systems, Signal Processing 69 (9) (September 1999) 169–175.
[7] K.R. Ralev, P.H. Bauer, Realization of block floating point digital filters and application to block implementations, IEEE Transactions on Signal Processing 47 (4) (April 1999) 1076–1086.
[8] K. Kalliojärvi, J. Astola, Roundoff errors in block-floating-point systems, IEEE Transactions on Signal Processing 44 (4) (April 1996) 783–790.
[9] P.H. Bauer, Absolute error bounds for block floating point direct form digital filters, IEEE Transactions on Signal Processing 43 (8) (August 1995) 1994–1996.
[10] S. Sridharan, G. Dickman, Block floating point implementation of digital filters using the DSP56000, Microprocessors and Microsystems 12 (July–August (6)) (1988) 299–308.
[11] F.J. Taylor, Block floating point distributed filters, IEEE Transactions on Circuits System CAS-31 (March 1984) 300–304.
[12] A. Mitra, M. Chakraborty, H. Sakai, A block floating point treatment to the LMS Algorithm: efficient realization and roundoff error analysis, IEEE Transactions on Signal Processing, (December 2005) 4536–4544.
[13] E. David, C. Lovescu, A Block Floating Point Implementation for an N-Point FFT on the TMS320C55X DSP, Texas Instruments Application Report, SPRA948 (September 2003).
[14] E. Bidet, D. Castelain, C. Joanblanq, P. Senn, A fast single-chip implementation of 8192 complex point FFT, IEEE Journal of Solid State Circuits 30 (3) (March 1995) 300–305.
[15] A. Erickson, B. Fagin, Calculating FHT in hardware, IEEE Transactions on Signal Processing 40 (June 1992) 1341–1350.
[16] R. Shaik, M. Chakraborty, An efficient finite precision realization of the adaptive decision feedback equalizer, in: Proceedings of the ISCAS-2007, May 2007, New Orleans, USA.