Journal of VLSI Signal Processing 39, 49–62, 2005

Energy Efficient VLSI Architecture for Linear Turbo Equalizer*

SEOK-JUN LEE, NARESH R. SHANBHAG AND ANDREW C. SINGER
Department of Electrical and Computer Engineering, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, 1308 West Main Street, Urbana, IL 61801, USA

Received March 20, 2003; Revised May 28, 2003; Accepted June 3, 2003

First online version published in September, 2004

Abstract. In this paper, energy efficient VLSI architectures for linear turbo equalization are studied. Linear turbo equalizers exhibit dramatic bit error rate (BER) improvement over conventional equalizers by enabling a form of joint equalization and decoding in which soft information is iteratively exchanged between the equalizer and decoder. However, turbo equalizers can be computationally complex and hence require significant power consumption. In this paper, we present an energy-efficient VLSI architecture for such linear turbo equalizers. Key architectural techniques include elimination of redundant operations and early termination. Early termination enables powering down parts of the soft-input soft-output (SISO) equalizer and decoder thereby saving power. Simulation results show that energy savings in the range 30–60% and 10–60% are achieved in equalization and decoding, respectively. Furthermore, we present finite precision requirements of the linear turbo equalizer and an efficient rescaling method to prevent overflow.

Keywords: iterative equalizer, iterative decoder, SISO, low-power, turbo,architecture

1. Introduction

Most high bit-rate communication systems suffer from intersymbol interference (ISI) in addition to noise during transmission over frequency selective channels. Conventional solutions (e.g., see Fig. 1) use separate equalization and decoding functions to compensate for the ISI and noise, respectively. Recently, the concept of *turbo equalization* has been proposed [1–5], where the equalization and decoding functions are iteratively combined through the exchange of soft information between the constituent devices. In turbo equalization, a soft-input soft-output (SISO) equalizer and a SISO decoder exchange soft information to jointly estimate the transmitted data. In [1], a maximum *a posteriori* probability (MAP) detector combined with a MAP decoder has been shown to provide dramatically improved bit error rate (BER) performance for many transmission channels. However, the complexity of such MAP-based schemes is prohibitive for modulation with high spectral efficiency and large delay spreads. For example, the number of states in a MAP SISO equalizer is $16^4 = 65536$ for a channel with memory 4, and 16 quadrature amplitude modulation (QAM). The MAP SISO equalizer can be replaced by a lower complexity linear SISO equalizer such as an ISI canceller [2, 3] or a minimum mean square error (MMSE) equalizer [4, 5]. These equalizers can combat ISI with a higher degree of computational efficiency even in the presence of high spectral efficiency modulation schemes. For highly degraded channels [6], as shown in the example in Fig. 2, we see that these linear turbo equalizers can provide substantial gain over conventional, separately equalized and decoded, systems. Note that for very high-speed digital subscriber line (VDSL)

*This material is based upon work supported by National Science Foundation under Grant CCR 99-79381, CCR-0092598, and ITR 00-85929. Part of the paper has been presented at IEEE Workshop on Signal Processing Systems 2002.

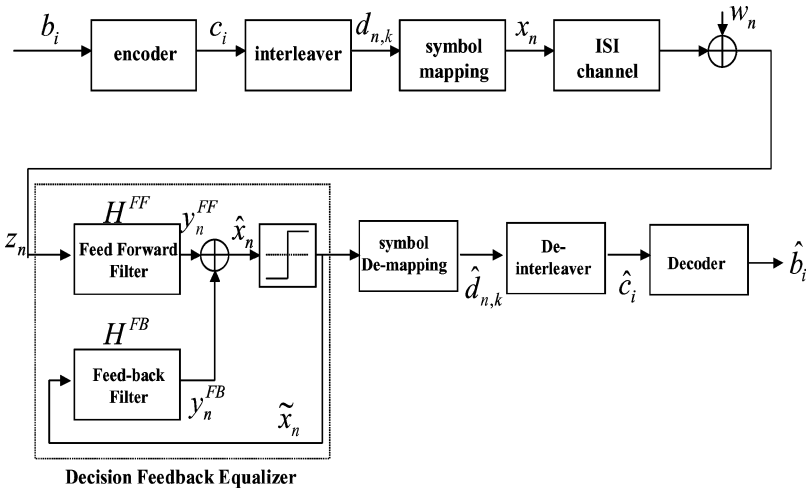

Figure 1. Block diagram of a conventional communication system with separate equalization and decoding.

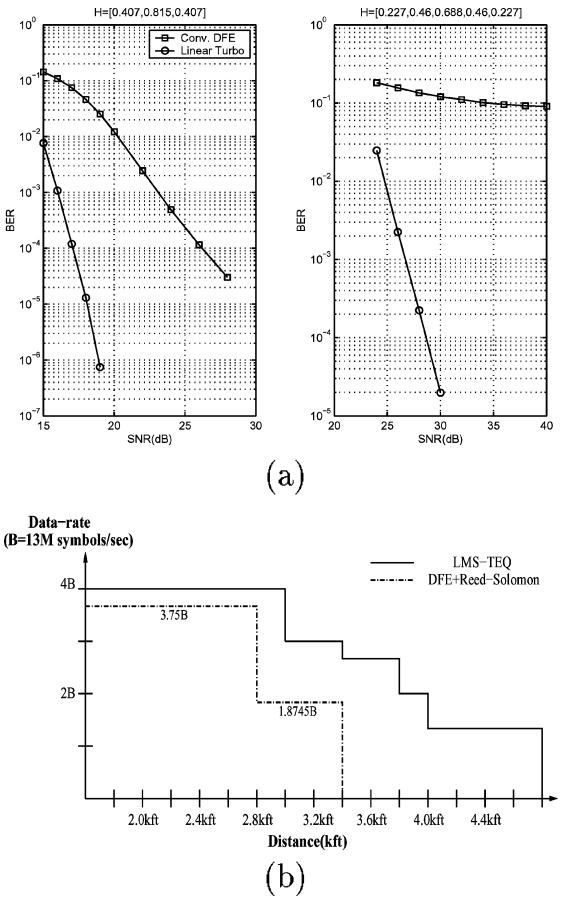

Figure 2. Comparison of turbo equalization and conventional equalization: (a) BER advantage for severely attenuated channels $\mathbf{H} = [0.407, 0.815, 0.407]$ and $\mathbf{H} = [0.227, 0.46, 0.688, 0.46, 0.227]$, and (b) data-rate advantage for VDSL channels.

systems, turbo equalization can increase the transmission data rate by up to 60% while transmitting data over longer lines ($\geq$3.5 kft) where non-iterative schemes [7] can fail to achieve the recommended BER $= 10^{-7}$.

The system level benefits of joint equalization and decoding motivate our research into low-complexity and low-power VLSI architectures for linear turbo equalizers. Related work includes low-power design techniques proposed for turbo code decoders [8–16]. In [8–11], the effects of quantization of the log-likelihood ratio (LLR) from each SISO module are discussed, and the required precisions are determined empirically. In [8, 12, 13], various techniques for early termination of iterative decoding of turbo codes are introduced and some experimental results are given. In [14, 15], the optimization of memory accesses has been studied, and in [16], a sliding window log-domain MAP algorithm is proposed to reduce the memory storage requirements. However, most published research on turbo equalization has focussed on algorithmic issues [1–5, 17–19] rather than implementation issues. In this paper, we investigate low power VLSI architectures for the constituent SISO equalizer and decoder blocks.

In a turbo equalizer, it is known that different portions of a data block may require a large number of iterations between the equalizer and decoder to converge. This is similar in nature to the behavior observed in decoding of turbo codes [8, 11–13]. We exploit this property to eliminate unnecessary operations thereby reducing power. We further reduce power in the system via early termination techniques [11–13] as used previously in low-power turbo decoders. We can

achieve early termination by monitoring the soft information passed between the SISO devices, as opposed to requiring a fixed, large number of iterations to complete. Experimental results on typical channels show that the proposed complexity reduction techniques for linear turbo equalization can achieve significant energy savings.

This paper is organized as follows. After a review of linear turbo equalization in Section 2, a VLSI architecture for linear turbo equalizers is presented and possible architectural approaches for achieving energy savings are explored in Section 3. In Section 4, finite precision analysis via empirical experiments is summarized and simulation results are presented to validate the proposed low-power techniques. Section 5 provides some concluding remarks.

2. Linear Turbo Equalization

In this section, we first describe the operation of the conventional decision feedback equalizer (DFE). We then highlight the differences that arise in a SISO equalizer and the associated SISO decoder.

2.1. Decision Feedback Equalizer

Figure 1 describes the transmitter and the conventional receiver model in which a DFE is used. The binary data b_i is encoded to yield the coded data c_i, the interleaver permutes the coded data c_i, and then m interleaved bits at time index n, $d_{n,0}, d_{n,1}, \ldots, d_{n,m-1}$, are grouped into one of 2^m symbols, x_n, which is transmitted over the ISI channel with additive white Gaussian noise (AWGN). The channel output z_n is given by

$$z_n = \sum_{k=-M_1}^{M_2} h_k x_{n-k} + w_n, \tag{1}$$

for an equivalent baseband discrete-time channel response h_k and noise sequence w_n. The DFE computes the estimate $\hat{x}_n$ from z_n and the past hard decisions from the slicer, $\tilde{x}_n = \arg\min_{x \in X} ||\hat{x}_n - x_n||^2$, as follows:

$$\hat{x}_n = \sum_{k=-L}^{L} H_k^{\mathrm{FF}} \cdot z_{n-k} + \sum_{k=1}^{M} H_k^{\mathrm{FB}} \cdot \tilde{x}_{n-k}, \tag{2}$$

where $2L+1$ and M are the number of taps in the feedforward (FF) and feedback (FB) filters, respectively. The slicer outputs are then processed to yield bit estimates $\hat{b}_i$ by the channel decoder.

2.2. SISO Equalizer

The linear turbo equalizer shown in Fig. 3 improves receiver performance by exchanging soft information between each of the SISO blocks [2–5]. The turbo equalizer operates on a block of N coded bits, in which the SISO equalizer operates on $\frac{N}{m}$ symbols, while the SISO decoder operates on the block of N bits. In the SISO equalizer, soft symbols $\bar{x}_n$ from the previous iteration of the decoder are processed in the feedback filter, rather than hard symbols from the slicer $\tilde{x}_n$. The channel output z_n and soft symbol $\bar{x}_n$ are processed as follows,

$$\hat{x}_n = \sum_{k=-L}^{L} H_k^{\mathrm{FF}} \cdot z_{n-k} + \sum_{k=-M}^{M} H_k^{\mathrm{FB}} \cdot \bar{x}_{n-k}, \tag{3}$$

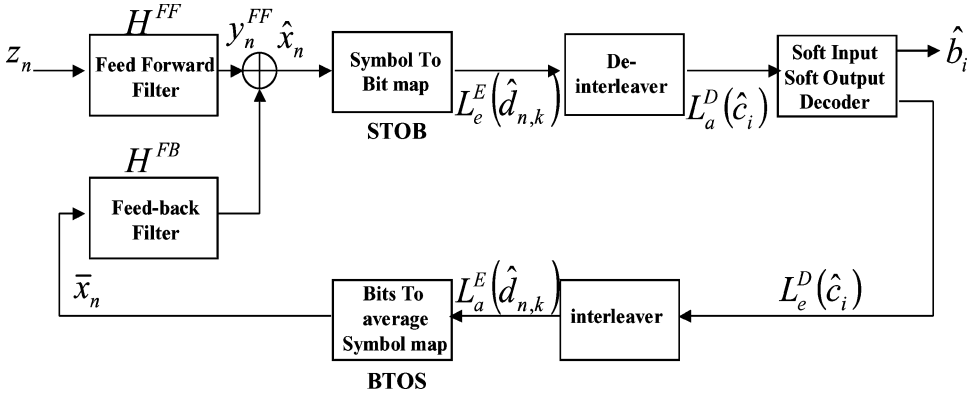

Figure 3. Block diagram of a linear turbo-equalizer.

where H_0^{FB} is equal to zero, in order to generate the estimated symbol $\hat{x}_n$ and to avoid the short feedback cycle of soft information; this is analogous to the so called extrinsic information used in turbo decoding and MAP-based turbo equalization. The symbol-to-bit (STOB) mapper [3] converts each estimated symbol $\hat{x}_n$ to a LLR on $d_{n,k}$ denoted by $L_e^E(d_{n,k})$, where $k = 0, 1, \ldots, m-1$ is the bit index within a symbol. The LLR is approximated as:

$$L_e^E(d_{n,k}) = K \ln \frac{\Pr(d_{n,k} = +1 \mid \hat{x}_n)}{\Pr(d_{n,k} = -1 \mid \hat{x}_n)}, \tag{4}$$

where K is a constant to be defined later. By defining the vector, $\mathbf{d_n} = \{d_{n,0}, \ldots, d_{n,m-1}\}$ and applying the law of total probability [20], (4) can be written as

$$L_e^E(d_{n,k}) = K \ln \frac{\sum_{\mathbf{d_n}: d_{n,k}=1} P(\hat{x}_n \mid \mathbf{d_n})}{\sum_{\mathbf{d_n}: d_{n,k}=0} P(\hat{x}_n \mid \mathbf{d_n})}, \tag{5}$$

where, $k = 0, 1, \ldots, m-1$. By approximating $P(\hat{x}_n|\mathbf{d_n})$ as a Gaussian probability density function (pdf), (5) can be expressed as

$$L_e^E(d_{n,k}) = K \ln \frac{\sum_{\mathbf{d_n}\{d_{n,k}=1\}} \exp\left(-\frac{1}{2\sigma_w^2}(\hat{x}_n - x_n\{d_{n,k}=1\})^2\right)}{\sum_{\mathbf{d_n}\{d_{n,k}=0\}} \exp\left(-\frac{1}{2\sigma_w^2}(\hat{x}_n - x_n\{d_{n,k}=0\})^2\right)}, \tag{6}$$

where, $x_n\{d_{n,k} = 1\}$ denotes a symbol mapping in which $d_{n,k}$ is equal to one. However, the LLR computations in (6) cannot be easily mapped to VLSI because the exponential terms require a large dynamic range and are non-linear. However, by employing the logarithm Jacobian approximation [3]

$$\begin{aligned} \ln\{e^{-x} + e^{-y}\} &\approx -\min(x, y) + \ln\left\{1 + e^{-|x-y|}\right\} \\ &= \overset{*}{\min}(x, y), \end{aligned} \tag{7}$$

we can express (6) as

$$L_e^E(d_{n,k}) \approx \frac{K}{2\sigma_w^2}\Big[\overset{*}{\min}_{\mathbf{d_n}\{d_{n,k}=0\}}(\hat{x}_n - x_n\{d_{n,k}=0\})^2 - \overset{*}{\min}_{\mathbf{d_n}\{d_{n,k}=1\}}(\hat{x}_n - x_n\{d_{n,k}=1\})^2\Big]. \tag{8}$$

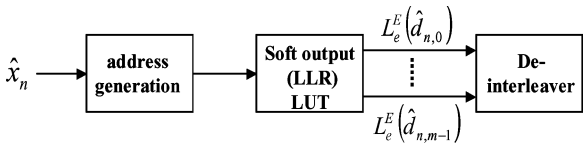

Figure 4. Symbol to binary (STOB) mapping logic.

By setting K to $2\sigma_w^2$, $L_e^E(d_{n,k})$ is only a function of the linear equalizer output,

$$L_e^E(d_{n,k}) \approx \Big[\overset{*}{\min}_{\mathbf{d_n}\{d_{n,k}=0\}}(\hat{x}_n - x_n\{d_{n,k}=0\})^2 - \overset{*}{\min}_{\mathbf{d_n}\{d_{n,k}=1\}}(\hat{x}_n - x_n\{d_{n,k}=1\})^2\Big], \tag{9}$$

and the receiver complexity is simplified since the noise variance disappears and its estimation is no longer needed.

Figure 4 shows a potential hardware implementation for computing $L_e^E(\cdot)$ using a look-up table (LUT). The SISO equalizer outputs $L_e^E(\cdot)$ are passed through a de-interleaver and then fed to the SISO decoder, which is described next.

2.3. SISO Decoder

The LLR's updated by the linear SISO equalizer are taken by the SISO decoder as inputs and used to produce the reliability metric $L_e^D(\cdot)$ on coded bits [3]. In this work, a sliding window log-domain MAP algorithm is employed to make the overall metric be a sum rather than a product and to minimize the metric storage requirements. If $\mathbf{c}$ is the coded block and $\mathbf{L}_a^D(\mathbf{c})$ is the updated LLR sequence from SISO equalizer, then the *a posteriori* LLR for the kth bit is given by

$$\begin{aligned} L_e^D(c_k) &= \frac{P\left(c_k = +1 \,\middle|\, \mathbf{L}_a^D(\mathbf{c})\right)}{P\left(c_k = -1 \,\middle|\, \mathbf{L}_a^D(\mathbf{c})\right)} \\ &= \frac{\sum_{\mathbf{c}\{c_k=+1\}} p\left(\mathbf{c}, \mathbf{L}_a^D(\mathbf{c})\right)}{\sum_{\mathbf{c}\{c_k=-1\}} p\left(\mathbf{c}, \mathbf{L}_a^D(\mathbf{c})\right)}. \end{aligned} \tag{10}$$

The sliding window Log-MAP algorithm can be used to compute (10) [16]. In this algorithm, we define the forward metric $a_k(s)$, backward metric $b_k(s)$, and branch metric $\gamma_k(s', s)$ as

$$\gamma_k(s', s) = \frac{1}{2} \sum c_k \cdot L_a^D(c_k) \tag{11}$$

$$a_k(s) = \overset{*}{\max_{s'}}[a_{k-1}(s') + \gamma_k(s', s)] \tag{12}$$

$$b_{j-1}(s) = \overset{*}{\max_{s}}[b_j(s) + \gamma_j(s', s)], \tag{13}$$

Table 1. 4-PAM binary to symbol mapping.

Binary	01	00	10	11
Symbol	−3	−1	+1	+3

where the $\overset{*}{\max}$ operation is implemented as

$$\overset{*}{\max}(x, y) = \max(x, y) + \ln\left(1 + e^{-|x-y|}\right), \quad (14)$$

and the approximate LLR of the k-th bit as

$$L_e^D(c_k) \approx \overset{*}{\max}_{s',s:c_k=+1}[a_{k-1}(s') + \gamma_k(s', s) + b_k(s)] - \overset{*}{\max}_{s',s:c_k=-1}[a_{k-1}(s') + \gamma_k(s', s) + b_k(s)]. \quad (15)$$

The updated $L_e^D(\cdot)$ is passed through an interleaver and used as input to the bits-to-average-symbol (BTOS) mapping block, which computes the average symbol value, $\bar{x}_n$, by taking a statistical expectation. For BPSK modulation, the average symbol can be expressed as

$$\bar{x}_n = \Pr\{x_n = 1\} - \Pr\{x_n = -1\} = \frac{\exp\left(L_a^E(d_n)\right) - 1}{1 + \exp\left(L_a^E(d_n)\right)}, \quad (16)$$

and for a 4-level pulse amplitude modulation (PAM) where the symbol mapping is defined in Table 1,

$$\begin{aligned}\bar{x}_n &= 3 \cdot \Pr\{d_{n,0} = 1, d_{n,1} = 1\} \\ &\quad + 1 \cdot \Pr\{d_{n,0} = 1, d_{n,1} = 0\} \\ &\quad - 1 \cdot \Pr\{d_{n,0} = 0, d_{n,1} = 0\} \\ &\quad - 3 \cdot \Pr\{d_{n,0} = 0, d_{n,1} = 1\} \\ &= \frac{e^{L_a^E(d_{n,0})}\left[3e^{L_a^E(d_{n,1})} + 1\right] - 1 - 3e^{L_a^E(d_{n,1})}}{\left[1 + e^{L_a^E(d_{n,0})}\right]\left[1 + e^{L_a^E(d_{n,1})}\right]}. \end{aligned} \quad (17)$$

The averaged symbols $\bar{x}_n$ are then fed back into the SISO equalizer for the next iteration. Unlike a conventional DFE, the use of soft symbols in a turbo equalizer helps mitigate error propagation. However, the exponential terms in both the numerator and denominator of (17) are not easily implemented in VLSI circuits.

For modest signal-to-noise ratios (SNR) and typical channels, we have found empirically that $|L_e^D(\cdot)|$ typically lies between 0.0 and 30.0. Therefore, we would need at least six bits for the integer part and a few bits for the fractional part in order to represent $L_e^D(\cdot)$. From (16), we see that if $L_e^D(\cdot) = \pm 6.0$, then $\bar{x} = \pm 0.99$, which implies that the absolute values of $L_e^D(\cdot)$ greater than 6.0 would make little difference in the average symbol computation. Furthermore, for the 4-PAM constellation example, the output average symbols depend on only $L_a^E(d_{n,0})$ if $L_a^E(d_{n,1})$ is more than 6.0 and vice versa.

This property can be exploited to reduce the lookup table size, and the logic required to transfer the LLR value of each coded bit into soft symbols is depicted in Fig. 5, where $L_a^E(\cdot)$ is passed through saturation logic, and then becomes the look-up table access address. Although the dynamic range is much smaller, numerical experiments verify that the BER performance loss is negligible.

3. Turbo Equalizer VLSI Architectures

Figure 6 describes the VLSI architecture of a linear turbo equalizer, which is made up of a linear SISO equalizer and SISO decoder employing sliding window Log-MAP decoding. In the linear SISO equalizer, the feedforward and feedback filters are implemented using multiply-accumulate (MAC) based architectures. The estimated symbol $\hat{x}_n$ is the input to the $L_e^E(\cdot)$ LUT address generation logic. The LUT outputs are de-interleaved and passed to the SISO decoder.

In the SISO decoder, the de-interleaved LLR outputs $L_a^D(\cdot)$ are used in the branch metric ($\gamma_k(s', s)$) unit. There are two backward metric computation units and one forward metric computation unit which contain an array of add-compare-select (ACS) circuits.

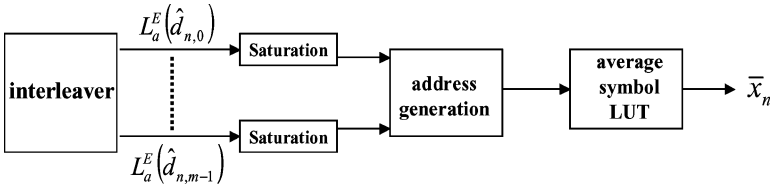

Figure 5. Binary to average symbol (BTOS) mapping logic.

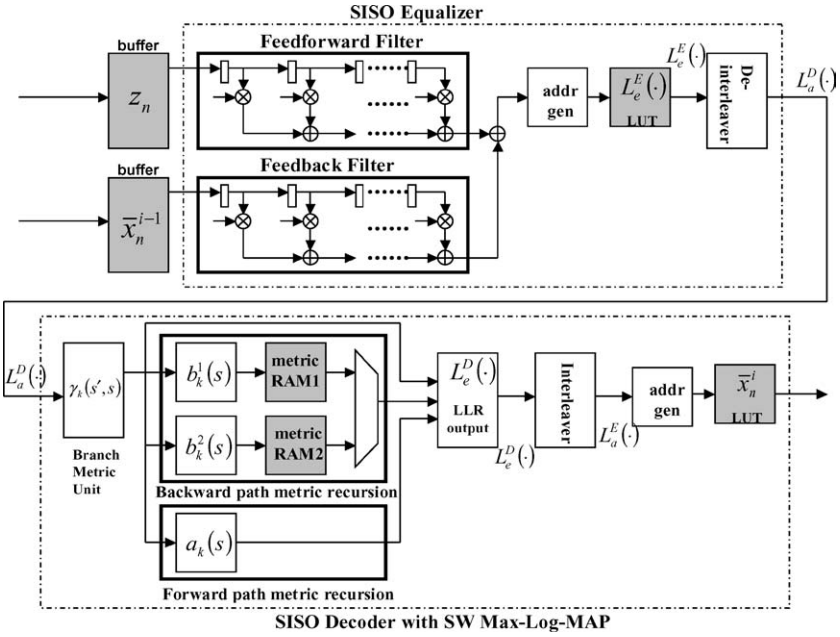

Figure 6. VLSI architecture of linear turbo equalizers.

Equation (15) is implemented in the $L_e^D(\cdot)$ LLR output block. The $L_e^D(\cdot)$ LLR's are interleaved bit-by-bit, and then passed to the address generation block to access the soft symbol LUT, whose outputs will be used in the next iteration of SISO equalization.

There are a wide range of known architectural techniques [21–25] to parallelize both the MAC-based finite impulse response (FIR) filter implementation [21, 22] and the computation of $L_e^D(\cdot)$ LLR's [23–25] thereby improving the equalizer and decoding throughput.

A snapshot of averaged and estimated symbols over all iterations reveals that different symbols often converge at different rates, that is, different regions within a data block require more iterations than others. This motivates several key techniques which not only attack the computational inefficiency of SISO equalization by eliminating unnecessary operations, but also enable the termination of iterations by employing a novel stopping criterion to detect convergence. In this section, energy efficient architectural solutions for the linear turbo equalizer are described in more detail.

3.1. Low-Power SISO Equalization and Decoding

Various stopping criteria in turbo decoders [12, 13] were proposed to achieve low-power VLSI implementations. However, most of these techniques cannot be directly applied to the linear turbo equalization because the soft information exchanged between the equalizer and decoder are different.

In the linear turbo equalizer, the average symbols are updated using (16) or (17) at every iteration and can provide clues for convergence. For example, the difference between average symbols of consecutive iterations can be checked over one block as follows:

$$\left|\bar{x}_n^i - \bar{x}_n^{i-1}\right| = \left|\frac{e^{L_a^E(\hat{d}_n^i)} - 1}{e^{L_a^E(\hat{d}_n^i)} + 1} - \frac{e^{L_a^E(\hat{d}_n^{i-1})} - 1}{e^{L_a^E(\hat{d}_n^{i-1})} + 1}\right| < \delta_S, \forall n, \tag{18}$$

where i is the iteration index, $\delta_S > 0$ is a constant, and BPSK modulation is assumed. As the SISO decoder converges, the magnitudes of the LLR values become large and Eq. (18) can be used as a stopping criterion.

The turbo equalizer may initially have some symbols converge to desired values, while others do not. In each iteration of the SISO equalizer, we compute

$$\hat{x}_n^i = \sum_{k=-L}^{L} H_k^{\mathrm{FF}} \cdot z_{n-k} + \sum_{k=-M}^{M} H_k^{\mathrm{FB}} \cdot \bar{x}_{n-k}^{i-1}. \tag{19}$$

If $|\bar{x}_n^{i-1} - \bar{x}_n^{i-2}| \leq \delta_S$ over the filtering window, then the estimate $\hat{x}_n^i$ does not need to be updated. From

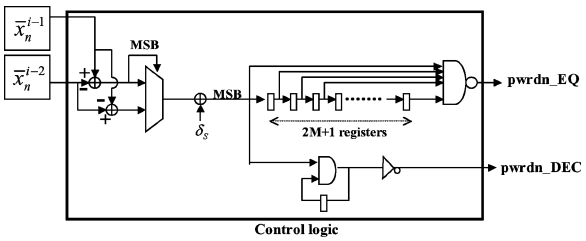

Figure 7. Control logic for low-power VLSI implementation.

the second iteration, the equalizer operations are selectively carried out only if the update of $L_e^E(\hat{d}_n^i)$ is needed. Control logic checks the differences between soft symbols of two consecutive iterations over the filtering window, and decides whether $L_e^E(\hat{d}_n^i)$ is updated or not.

Figure 7 depicts the details of the control logic to generate *pwrdn_EQ* and *pwrdn_DEC* for the SISO equalizer and decoder, respectively. The *pwrdn_EQ* signal turns down SISO equalizer operations if the update of $L_e^E(\hat{d}_n^i)$ is not necessary and the *pwrdn_DEC* signal terminates iterations by turning off the SISO decoder block. The most significant bit (MSB) of $|\bar{x}_n^{i-1} - \bar{x}_n^{i-2}| - \delta_S$ is computed and stored in the 1-bit register over the $2M+1$ filtering window. If all MSB's are equal to '1' which implies all difference values over the $2M+1$ filtering window are less than δ_S, then the control signal will turn off equalization of the current sample. For generating the decoder control signal, at the first symbol of one block, the 1-bit register is reset to one. For each symbol, the MSB of $|\bar{x}_n^{i-1} - \bar{x}_n^{i-2}| - \delta_S$ updates the *pwrdn_DEC* signal stored in the 1-bit register. After processing one block, if the *pwrdn_DEC* signal is equal to '1' meaning all the difference values are less than the stopping threshold, the iteration is terminated. It should be noted that the logic complexity of this control block is marginal compared to that of the SISO equalizer and decoder blocks. Figure 8 shows the modified data-path including the necessary control logic.

3.2. *Determination of Threshold Value δ_S*

The threshold value, δ_S, enables us to trade-off BER versus energy efficiency. As in [26], the mutual information (I) between LLR's of the outputs of SISO decoder and the transmitted symbols can be used to test convergence. Therefore, we determine δ_S such that

$$I_{\text{org}} - I_{\text{LP}} < I_T \tag{20}$$

where I_{org} and I_{LP} are the mutual information of the original and low-power turbo equalizer, respectively, and I_T is the threshold value of the difference between

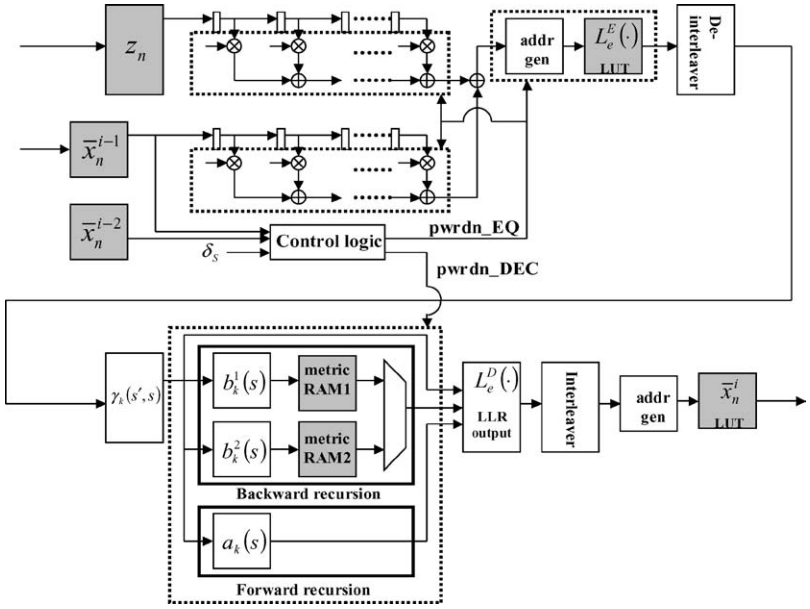

Figure 8. Modified architecture for energy saving.

I_{org} and I_{LP}. Note [26] that the mutual information I is given by

$$I = \frac{1}{2} \sum_{x \in \{\pm 1\}} \sum_{l} \left\{ f_{L_e^p}(l \mid x) \cdot \log_2 \frac{2 f_{L_e^p}(l \mid x)}{f L_e^D(l \mid +1) + f_{L_e^p}(l \mid -1)} \right\}, \quad (21)$$

where, $f_{L_e^p}(l \mid x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(l - \frac{x\sigma^2}{2})^2}{2\sigma^2})$ and σ denotes the variance of the LLR value $L_e^D(\cdot)$. We have determined empirically that $I_T = 10^{-4}$ is a reasonable value for channel SNR's in the range from 15 dB to 20 dB for typical channels.

4. Summary of Experimental Results

4.1. Simulation Setup

We employ a recursive systematic convolutional (RSC) encoder at the transmitter with a generator polynomial $(23, 35)_8$. The coded bit stream is first passed through a random interleaver [27] followed by 4 level pulse amplitude modulation (PAM). For purposes of comparison, we considered three static channel models (channel **A**, **B** and **C**),

$$\begin{aligned}
\mathbf{H}_A(z) &= 0.04z^5 - 0.05z^4 + 0.07z^3 - 0.21z^2 - 0.5z \\
&\quad + 0.72 + 0.36z^{-1} + 0.21z^{-3} + 0.03z^{-4} \\
&\quad + 0.07z^{-5} \\
\mathbf{H}_B(z) &= 0.407z + 0.815 + 0.407z^{-1} \\
\mathbf{H}_C(z) &= 0.227z^2 + 0.46z + 0.688 + 0.46z^{-1} \\
&\quad + 0.227z^{-2},
\end{aligned}$$

where $\mathbf{H}_A(z)$ is a "good" channel, $\mathbf{H}_B(z)$ has "medium ISI", and $\mathbf{H}_C(z)$ has "severe ISI" [6]. We employ a 1024-bit interleaver for channel **A** and **B** while a 4096-bit interleaver is employed for channel **C** because the larger interleaver overcomes the worst ISI channel, $\mathbf{H}_C(z)$, more effectively by decorrelating a burst of errors better. To determine the coefficients of the feedforward and feedback filters (H^{FF} and H^{FB}), we employ a least-mean-square (LMS) adaptive algorithm. The number of taps used in the SISO equalizer is summarized in Table 2, where N_{FF} denotes the number of feedforward filter taps, N_{FB1} and N_{FB2} are the number of feedback filter taps in the first and subsequent iterations, respectively. A sliding window Log-MAP algorithm [16] is employed and 5 iterations are used for channel **A** and **B** and 8 iterations for channel **C** are carried out. For performance comparison, the BER is measured by transmitting as many blocks as are needed so that at least ten erroneous blocks are received.

Table 2. Number of taps in SISO equalization.

Channel	N_{FF}	N_{FB1}	N_{FB2}
A	15	7	14
B	7	4	6
C	11	5	10

4.2. Optimization of Word-Lengths

By employing the proposed soft input mappings in (9) and (17), an integer-based linear turbo equalizer is implemented. Word-lengths of internal variables of the linear turbo equalizer were determined empirically in order to minimize loss in BER. A comparison with a floating point implementation is given in Fig. 9 for the channel $\mathbf{H}_B(z)$, where the performance loss is negligible at BER of 10^{-6} after 5 iterations. The precisions used for each internal variable are summarized in Table 3.

The precisions of path metrics in SISO decoding should be carefully determined to avoid overflow.

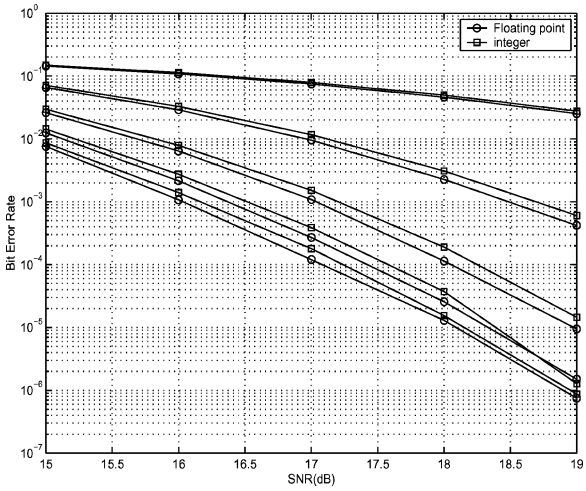

Figure 9. BER performance comparison of integer implementation over $\mathbf{H}_B(z)$.

Table 3. Empirically determined precision summary in linear turbo equalizer for 4-PAM.

Variables	Precision	
$z(n)$	12-bit	Received symbol
$W_{\mathrm{FF}}(n)$	11-bit	FF filter coefficient
$y_{\mathrm{FF}}(n)$	16-bit	Accumulation of FF filter
$W_{\mathrm{FB}}(n)$	11-bit	FB filter coefficient
$y_{\mathrm{FB}}(n)$	16-bit	Accumulation of FB filter
$\bar{x}(n)$	5-bit	Average symbol
$\lambda_{\max}$, c_k	7-bit	Maximum branch metric in SISO decoder
a_k, b_k	10-bit	Forward & backward metrics
$L_e^E(\cdot)$	7-bit	LLR of SISO equalizer
$L_e^D(\cdot)$	5-bit	LLR of SISO decoder
LUT in (9)	128 entry ROM	L_e^E mapping LUT
LUT in (17)	725 entry ROM	Average symbol mapping LUT

Metric values may be corrupted due to overflow since the forward metrics, a_k, are accumulated over a block, and the backward metrics, b_k, are computed in a recursive manner over a sliding window [16]. The path metric normalization method developed via a modulo arithmetic [28] is not applicable here because, in SISO decoding, the difference values between path metrics are used as a measure of reliability rather than the comparison results on which path metric is larger in Viterbi decoders. In this paper, we subtract a constant scale from all the path metrics to avoid overflow [29].

The updated path metric at time index k is bounded by

$$\begin{aligned} a_k(s) &= \max_{s'}[a_{k-1}(s') + \gamma_k(s', s)] \\ &\le Q + \lambda_{\max}, \end{aligned} \tag{22}$$

where $a_{k-1} \le Q$ and $\lambda_{\max}$ is the maximum branch metric. In practice, Q is not known but the upper bound Δ_{pm} on the path metric difference is known as $\Delta_{\mathrm{pm}} \le \lambda_{\max} \upsilon$ [30], where υ is the memory order of the encoder. Therefore, we choose q, the path metric precision, to satisfy $2^{q-2} > \Delta_{\mathrm{pm}} = \lambda_{\max}\upsilon$. This enables us to check for overflow of each path metric by using the circuit in Fig. 11, where q-bit metric registers are employed and N_s is the number of trellis states in the SISO decoder. If all metrics are greater than 2^{q-1}, then we subtract 2^{q-1} from all state path metrics. In our design, the maximum branch metric, $\lambda_{\max}$, is less than 2^6, υ is 4, and $2^8 > \Delta_{\mathrm{pm}}$. Hence, 10 bits are assigned to the path metrics, a_k and b_k. We also found empirically that 10 bits of precision do not cause overflow due to the rescaling. Figure 10 shows the maximum values of the a_k and b_k metrics during the transmission of 10^8 information bits.

Furthermore, the path metrics are positive quantities, and hence the constant subtraction can be done efficiently by setting q-th bit of all path metric registers to zero. The logic diagram implementing the proposed normalization method is shown in Fig. 12 and only one multiplexer is required for rescaling each metric register.

4.3. *Power Savings*

Threshold values satisfying (20) were determined empirically to be $\delta_S = 1.0$ for 4-level PAM. Figure 13 illustrates the relative BERs of the floating point and integer implementations with the low-power schemes and thresholding methods explained in the previous section. We note that the impact on the BER performance is relatively small.

To measure the energy efficiency of the proposed scheme, we compute the overall energy savings in SISO decoding and equalization. The energy consumed by the SISO decoders is largely proportional to the number of executed iterations and hence the energy saving of the SISO decoder can be computed by

$$\begin{aligned} ES_{\mathrm{DEC}} &= \frac{E_{\mathrm{DEC,org}} - E_{\mathrm{DEC,LP}}}{E_{\mathrm{DEC,org}}} \times 100 \\ &= \frac{N_{\mathrm{iter,org}} - N_{\mathrm{iter,LP}}}{N_{\mathrm{iter,org}}} \times 100, \end{aligned} \tag{23}$$

where $N_{\mathrm{iter,org}}$ denotes the number of iterations in the original decoder, $N_{\mathrm{iter,LP}}$ is the number of iterations in the low-power decoder, and $E_{\mathrm{DEC,org}}$ and $E_{\mathrm{DEC,LP}}$ are the consumed energy of the original decoder and the low-power decoder, respectively.

Further, we assume that the energy consumed by the multiplier-accumulate (MAC) of SISO equalization is proportional to the precisions of a multiplier ($b_1 \times b_2$) and an accumulator (b_3) as shown below,

$$E_{\mathrm{MAC}} \propto K' \cdot (b_1 \times b_2 + b_3), \tag{24}$$

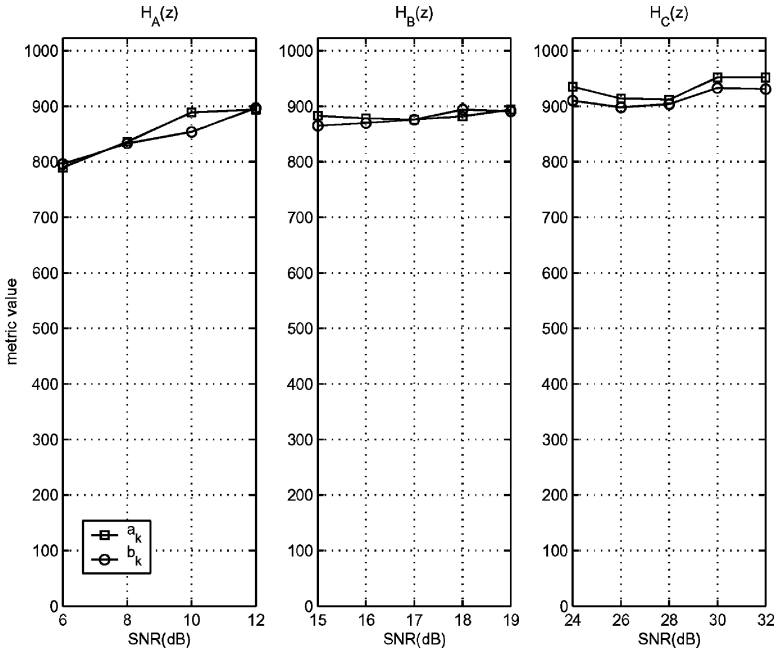

Figure 10. The maximum value in a_k and b_k metrics.

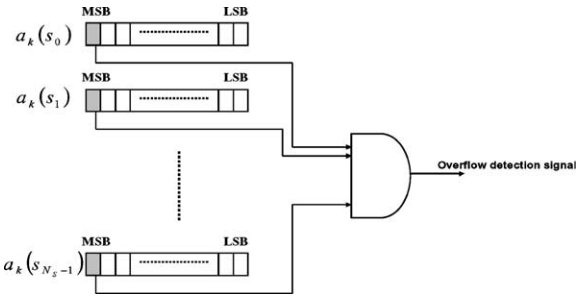

Figure 11. Overflow detection logic.

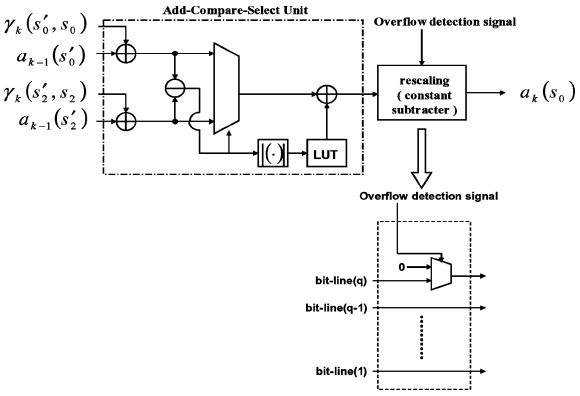

Figure 12. Add-Compare-Select unit and rescaling block diagram.

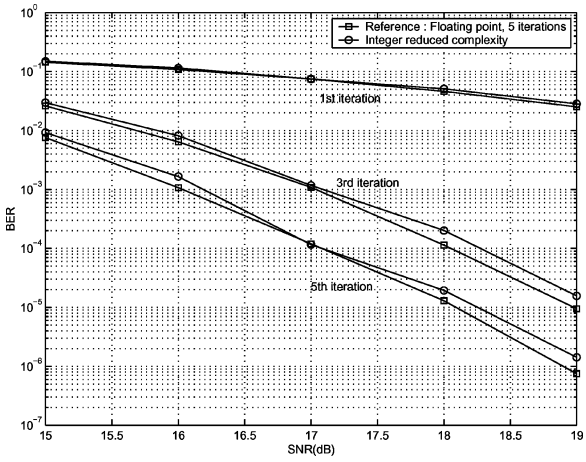

Figure 13. BER performance comparison versus channel SNR.

and hence the power consumed by a MAC operation can be expressed as

$$P = \alpha \cdot C_L \cdot V_{\mathrm{dd}}^2 \cdot f_S \propto K' \cdot (b_1 \times b_2 + b_3) \cdot V_{\mathrm{dd}}^2 \cdot f_S \tag{25}$$

where C_L is the load capacitance, V_{dd} is the supply voltage, and f_S is the operating frequency. The number of MAC operations required for the linear SISO

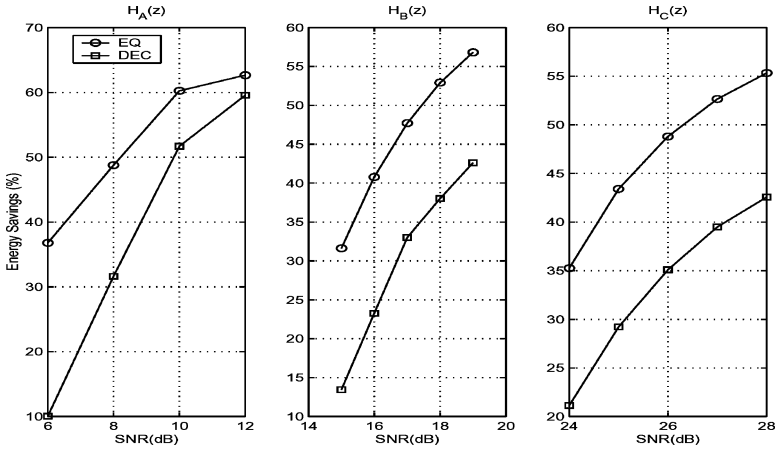

Figure 14. Energy savings of SISO equalizer and decoder.

equalizer is summarized in Table 4 and the energy consumed by the linear SISO equalizer can be expressed as

$$\begin{aligned} E_{\mathrm{EQ}} = K'\{&N_B(N_{\mathrm{FF}}(11\times 12+16)\\ &+N_{\mathrm{FB1}}(11\times 2+16))\\ &+N_{\mathrm{sym}}(N_{\mathrm{FF}}(11\times 12+16)\\ &+N_{\mathrm{FB2}}(11\times 5+16))\}, \end{aligned} \tag{26}$$

where N_B is the number of symbols in one block and N_{sym} denotes the number of symbols processed after the first iteration. The energy saving in the equalizer can be defined as

$$ES_{\mathrm{EQ}} = \frac{E_{\mathrm{EQ,org}} - E_{\mathrm{EQ,LP}}}{E_{\mathrm{EQ,org}}} \times 100, \tag{27}$$

where $E_{\mathrm{EQ,org}}$ and $E_{\mathrm{EQ,LP}}$ are the consumed energy of the original equalizer and the low-power equalizer, respectively, and $E_{\mathrm{EQ,LP}}$ includes the control logic complexity, which is a 5-bit subtract per symbol. Figure 14 shows the energy savings of the SISO equalizer and decoder; the proposed low-power scheme saves 30–60%

Table 4. MAC complexity analysis of SISO equalizer.

	Number of MAC's
Feed Forward section	$N_{\mathrm{FF}} \cdot MAC_{11\times 12+16}$ per symbol
Feed Back section in the first iteration	$N_{\mathrm{FB1}} \cdot MAC_{11\times 2+16}$ per symbol
Feed Back section in later iterations	$N_{\mathrm{FB2}} \cdot MAC_{11\times 5+16}$ per symbol
Control logic: Equation (18)	5-bit subtracter per symbol

Table 5. Average number of iterations in low-power scheme.

E_b/N_o	$\mathbf{H}_A(z)$	E_b/N_o	$\mathbf{H}_B(z)$	E_b/N_o	$\mathbf{H}_C(z)$
6 dB	4.5	15 dB	4.33	24 dB	6.30
8 dB	3.42	16 dB	3.84	26 dB	5.66
10 dB	2.42	17 dB	3.35	28 dB	5.19
12 dB	2.02	18 dB	3.10	30 dB	4.84
		19 dB	2.87	32 dB	4.60

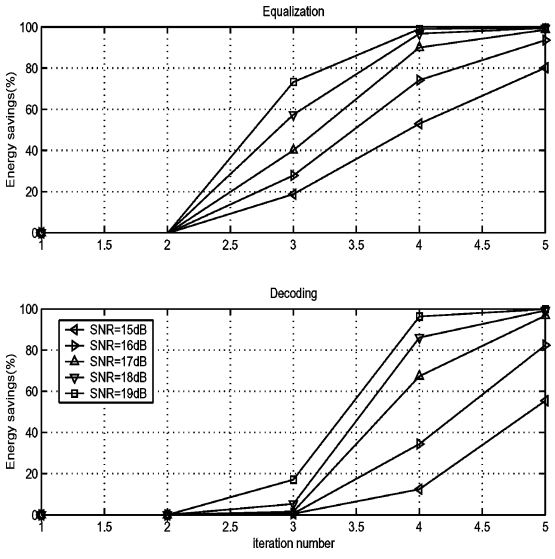

Figure 15. Energy savings of SISO equalization and decoding vs. iteration index for channel **B**.

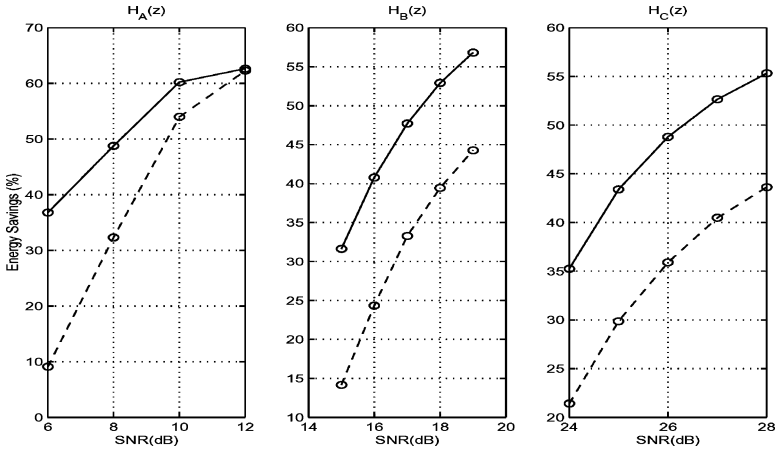

Figure 16. SISO equalizer energy saving comparison in case that only stopping criterion is applied.

and 10–60% energy, in the equalizer and decoder, respectively, depending upon the channel SNR.

Figure 15 shows the energy savings for channel **B** versus the iteration number. We see that the complexity is reduced much more in the high SNR region than the low SNR region. In particular, the 4th and 5th iterations need little computation, which results in greater energy savings. Table 5 shows the average number of iterations for different channel SNRs. We find that the low-power turbo equalizer achieves a reduction of 10 to 60% in the number of iterations.

Figure 16 shows the energy savings of two different schemes. The first scheme (solid line) employs low-power techniques in both the SISO equalizer and decoder as explained in the previous section. The other scheme (dotted line) applies only to the early termination of the linear turbo equalizer. It appears that the SISO equalizer energy is saved more by using low-power techniques than by applying only early termination.

5. Concluding Remarks

In this paper, we explore architectural approaches for energy-efficient linear turbo equalization by eliminating unnecessary operations and employing early termination. The low-power techniques need marginal control logic overhead compared with the main computation logic and are easily implemented in VLSI. Computer simulations over various channel conditions reveal that our integer-based implementation shows negligible BER performance loss and provide energy savings of 30–60% in the equalizer and 10–60% in the decoder, depending on the channel and SNR. Moreover, we studied finite word-length effects of symbol-to-binary and binary-to-symbol conversions inherent to the linear turbo equalizer and presented an efficient rescaling method for a SISO decoder.

References

1. C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative Correction of Intersymbol Interference: Turbo-Equalization," *European Transactions on Telecommunications*, vol. 6, 1995, pp. 507–511.
2. A. Glavieux, C. Laot, and J. Labat, "Turbo Equalization Over a Frequency Selective Channel," in *Symposium on Turbo-Codes*, Brest, France, Sept. 1997, pp. 96 – 102.
3. C. Laot, A. Glavieux, and J. Labat, "Turbo Equalization: Adaptive Equalization and Channel Decoding Jointly Optimized," *IEEE Journal of Selected Areas in Comm.,* vol. 19, Sept. 2001, pp. 1744–1752.
4. M. Tüchler, A. Singer, and R. Koetter, "Minimum Mean Squared Error Equalization Using A-Priori Information," *IEEE Trans. on Signal Processing*, vol. 50, no. 3, 2002, pp. 673–683.
5. M. Tüchler, R. Koetter, and A. Singer, "Turbo Equalization: Principles and New Results," *IEEE Trans. on Comm.,* vol. 50, no. 5, 2002, pp. 754–767.
6. J.G. Proakis, *Digital Communications*, 3rd ed., McGraw-Hill, 1995.
7. M. Goel and N.R. Shanbhag, "Low-Power Equalizers for 51.84 Mb/s Very High-Speed Digital Subscriber Loop (VDSL) modems," in *Proc. of IEEE Signal Processing Systems(SiPS): Design and Implementation*, Oct. 1998, pp. 317–326.
8. Z. Wang, H. Suzuki, and K.K. Parhi, "VLSI Implementation Issues of Turbo Decoder Design for Wireless Applications," in *Proc. of IEEE Signal Processing Systems(SiPS): Design and Implementation,* Oct. 1999, pp. 503–512.
9. G. Montorsi and S. Benedetto, "Design of Fixed-Point Iterative Decoders for Concatenated Codes with Interleavers," *IEEE Journal on Selected Areas in Comm.,* vol. 19, no. 5, May 2001.

10. J. Hsu and C. Wang, "On Finite-Precision Implementation of a Decoder for Turbo Codes," in *Proc. International Symposium on Circuits and Systems*, 1999, vol. 4, pp. 423–426.
11. P.H. Wu and S.M. Pisuk, "Implementation of a Low Complexity, Low Power, Integer-Based Turbo Decoder," in *Proc. Global Telecommunication Conf.*, 2001, vol. 2, pp. 946–951.
12. O. Leung, C. Yue, C. Tsui, and R. Cheng, "Reducing Power Consumption of Turbo Code Decoder Using Adaptive Iteration with Variable Supply Voltage," in *Proc. of IEEE Int. Symp. Low Power Electronics Design (ISLPED'99),* San Diego, CA, 1999, pp. 36–41.
13. D. Garrett, B. Xu, and C. Nicol, "Energy Efficient Turbo Decoding for 3G Mobile," in *Proc. of IEEE Int. Symp. Low Power Electronics Design (ISLPED'01)*, Huntington Beach, CA, 2001, pp. 328–333.
14. C. Schurgers, F. Catthoor, and M. Engels, "Memory Optimization of MAP Turbo Decoder Algorithms," *IEEE Trans. on VLSI Systems*, vol. 9, no. 2, 2001, pp. 305–312.
15. M.M. Mansour and N.R. Shanbhag, "VLSI Architectures for SISO-APP Decoders," *IEEE Trans. on VLSI Systems*, vol. 11, no. 4, 2003, pp. 627–650.
16. A.J. Viterbi, "An intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes," *IEEE Journal on Selected Areas in Comm.*, vol. 16, no. 2, 1998, pp. 260–264.
17. G. Bauch and V. Franz, "A Comparison of Soft-in/Soft-Out Algorithms for Turbo Detection," in *Proc. of IEEE Int. Conf. on Telecommunications(ICT'98),* June 1998, pp. 259–263.
18. G. Bauch, H. Khorram, and J. Hagenauer, "Iterative Equalization and Decoding in Mobile Communications Systems," in *Proc. of European Personal Mobile Comm. Conf. (EPMCC),* Oct. 1997.
19. G. Bauch, H. Khorram, and J. Hagenauer, "Iterative Equalization and Decoding for the GSM System," in *Proc. IEEE Veh. Technol. Conf.*, Ottawa, vol. 3, May 1998, pp. 2262–2266.
20. H. Stark and J.W. Woods, *Probability, Random Processes, and Estimation Theory for Engineers*, 2nd ed., Prentice Hall, 1994.
21. G.A. Clark, S.K. Mitra, and S.R. Parker, "Block Implementation of Adaptive Digital Filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. ASSP-29, no. 3, June 1981.
22. K.K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*, John Wiley & Sons, Inc., 1999.
23. J. Hsu and C. Wang, "A Parallel Decoding Scheme For Turbo Codes," in *Proc. IEEE International Symp. on Circuits and Systems*, vol. 4, 1998, pp. 445–448.
24. Z. Wang, Z. Chi, and K. Parhi, "Area-Efficient High Speed Decoding Schemes for Turbo/MAP Decoders," in *Proc. 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001, vol. 4, pp. 2633–2636.
25. M.M. Mansour and N.R. Shanbhag, "Design Methodology For High-Speed Iterative Decoder Architectures," in *Proc. 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, 2002, pp. 3085–3088.
26. S. ten Brink, "Convergence Behavior of Iteratively Decoded Parallel Concatenated Codes," *IEEE Trans. on Comm.,* vol. 49, 2001, pp. 1727–1737.
27. J. Hokfelt, O. Edfors, and T. Maseng, "Interleaver Design for Turbo Codes Based on the Performance of Iterative Decoding," in *Proc. of International Conf. on Comm.,* 1999, pp. 93–97.
28. A.P. Hekstra, "An Alternative to Metric Rescaling in Viterbi Decoders," *IEEE Trans on Comm.*, vol. 37, no. 11, pp. 1220–1222.
29. C.B. Shung, P.H. Siegel, G. Ungerboeck, and H.K. Thapar, "VLSI Architecture for Metric Normalization in the Viterbi Algorithm," in *Proceeding of IEEE International Conf. on Comm.*, 1990, vol. 4, pp. 1723–1728.
30. A.P. Hekstra, "On the Maximum Difference Between Path Metrics in a Viterbi Decoder," in *Proceeding of IEEE International Symposium on Information Theory*, 1993, pp. 21–21.

Seok-Jun Lee received his B.S. degree in Feb. 1996, and M.S. degree in Feb. 1998 all in School of Electrical Engineering from Seoul National University (SNU), Seoul, Korea. Since August 2000, he has been a research assistant at the Coordinated Science Laboratory (CSL), University of Illinois at Urbana-Champaign, where he is currently a Ph.D. candidate in Electrical and Computer Engineering. During the summer of 2001 and 2002, he worked at Texas Instruments Inc., Dallas, TX, with communication research laboratory, DSPS R&D center. His research interests include VLSI architectures for communication and signal processing systems.
slee6@uiuc.edu

Naresh R. Shanbhag received the B.Tech. degree from the Indian Institute of Technology, New Delhi, India, (1988), the M.S. degree from the Wright State University (1990) and the Ph.D. degree from the University of Minnesota, (1993), all in Electrical Engineering.

From July 1993 to August 1995, he worked at AT&T Bell Laboratories at Murray Hill, where he was the lead chip architect for AT&T's 51.84 Mb/s transceiver chips over twisted-pair wiring for Asynchronous Transfer Mode (ATM)-LAN and very high-speed digital subscriber line (VDSL) chip-sets. Since August 1995, he is with the Department of Electrical and Computer Engineering, and the Coordinated Science Laboratory where he is presently a Professor. His research interests are in low-power, high-performance, and reliable integrated circuit implementations of broadband communications and digital signal processing systems. He has published

numerous journal articles/book chapters/conference publications in this area and holds three US patents. He is also a co-author of the research monograph *Pipelined Adaptive Digital Filters* (Norwell, MA: Kluwer, 1994).

Dr. Shanbhag received the the 2001 IEEE Transactions on VLSI Systems Best Paper Award, the 1999 IEEE Leon K. Kirchmayer Best Paper Award, the 1999 Xerox Faculty Award, the National Science Foundation CAREER Award in 1996, and the 1994 Darlington best paper award from the IEEE Circuits and Systems society. From July 1997–2001, he was a Distinguished Lecturer for the IEEE Circuits and Systems Society. He served as an Associate Editor for the IEEE Transaction on Circuits and Systems: Part II, and the IEEE Transactions on VLSI systems. He was the Technical Program Chair of 2002 IEEE Workshop on Signal Processing Systems.
shanbhag@uiuc.edu

Andrew C. Singer was born in Akron, OH, in 1967. He received the S.B., S.M., and Ph.D. degrees, all in electrical engineering and computer science, from the Massachusetts Institute of Technology, Cambridge, in 1990, 1992, and 1996, respectively.

Since 1998, he has been on the faculty of the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign, where he is currently an Associate Professor in the Electrical and Computer Engineering Department, and a Research Assistant Professor in the Coordinated Science Laboratory. During the academic year 1996, he was a Post-Doctoral Research Affiliate in the Research Laboratory of Electronics at MIT. From 1996 to 1998, he was a Research Scientist at Sanders, A Lockheed Martin Company, Manchester, NH. His research interests include statistical signal processing and communication, universal prediction and data compression, and machine learning.

Prof. Singer was a Hughes Aircraft Masters Fellow and was the recipient of the Harold L. Hazen Memorial Award for excellence in teaching in 1991. In 2000, he received the National Science Foundation CAREER Award and in 2001 he received the Xerox Faculty Research Award. He is currently a member of the MIT Educational Council, Eta Kappa Nu, and Tau Beta Pi.
acsinger@uiuc.edu