Turbo Equalization for Receivers with Unreliable Buffer Memory

Jan Geldmacher, Klaus Hueske, and Jürgen Götze
Information Processing Lab
TU Dortmund University, Otto-Hahn-Strasse 4, 44227 Dortmund, Germany.
Correspondence: jan.geldmacher@ieee.org

***Abstract*—In this paper the effect of unreliable buffer memory on Turbo equalization is investigated. Under the assumption that the resulting bit errors are uniformly and independently distributed on receive and LLR buffer output, the effect of unreliable memory on the channel capacity is analyzed. It is further shown, that extrinsic information transfer and bit error rate (BER) performance of a conventional Turbo Equalizer are heavily degraded. Based on these observations a fault tolerant (FT) Turbo Equalizer is derived, which considers the memory error characteristics by using a modified transition metric for the involved MAP equalizer and decoder. It is demonstrated, that the FT Turbo Equalizer can effectively compensate memory errors and thus yields a significantly improved BER performance.**

I. Introduction

Power efficiency of baseband signal processing components is an important topic – especially in mobile devices, where the power consumption of signal processing circuitry directly affects battery lifetime. Reducing algorithmic complexity, for example by applying approximations [1], [2], is one way to go. However, it is well-known that besides logic a significant portion of energy is consumed by memory access. This is especially true for iterative algorithms, like Turbo decoders or Turbo Equalizers, that require large buffer memories to store received data and the extrinsic information, that is exchanged between the components.

The idea of aggressive voltage scaling tackles this problem [3]: By scaling the supply voltage of the memory below the specification, a reduction of power consumption can be achieved. Of course, as a result, the operation of the memory on bit-level becomes faulty. This eventually leads to unexpected behavior or serious performance degradation of the involved signal processing algorithms.

However, the authors of [3] show that by proper modeling of the resulting memory failures, and co-designing algorithm and hardware-circuitry, the resulting errors can be compensated up to a certain level. For the example of a 3GPP modem, they show that power savings of about 17.5% can be achieved. Further, in [4] a Viterbi Decoder with modified branch metric is presented, that is resilient to uniformly and independently distributed errors in the received soft bits, resulting from reduced supply voltage of the receive buffer.

Under a more general perspective, unreliable memory may also be the result of process-, time- or environment-dependent parameter variations of the involved integrated circuits [5].

In this paper the impact of faulty buffer memory on Turbo equalization is analyzed. Turbo equalization [6], [7] is an iterative method to detect coded data, that has been transmitted over an intersymbol interference (ISI) channel. Under the assumption that faulty memory results in uniformly distributed bit errors in the received values and the extrinsic information, the performance of a conventional Turbo Equalizer is heavily degraded. However, we show that by modifying the transition metrics of MAP equalizer and decoder, a fault tolerant (FT) Turbo Equalizer can be derived, which effectively compensates this memory error.

This paper is organized as follows. After the principle of Turbo equalization is briefly described in Sec. II-A, Sec. II-B and II-C describe the effect of the faulty memory in detail. In Sec. III the FT Turbo Equalizer is described. Performance analysis in terms of BER and EXIT charts is provided in Sec. IV. Conclusions are drawn in Sec. V.

II. Problem Description

A. Turbo Equalization System

We assume a convolutionally encoded transmission over an ISI channel as shown in Fig. 1. A binary information sequence $\mathbf{u}$ of length $k_c T$ is encoded using a rate $R = k_c/n_c$ convolutional code. The resulting code sequence $\mathbf{v} = \{v_j^{(t)}\}$, $j = 1 \dots n_c$, $t = 1 \dots T$ is interleaved, mapped to unit-energy modulation symbols $\mathbf{x} = \{x^{(l)}\}$, $l = 1 \dots n_c T$, and transmitted over a frequency selective channel. The following discussion is restricted to BPSK modulation, i.e. $x^{(l)} = \{\pm 1\}$ – the described concept can however be extended to higher modulation orders. The received sequence can be described as

$$y^{(l)} = \sum_{i=1}^{M} h^{(M-i+1)} x^{(l-M+i)} + n = s + n, \; n \sim \mathcal{N}(0, \sigma_y^2),$$

where $\mathbf{h} = \{h^{(i)}\}$ is the length M channel impulse response, s the ISI channel output, and n represents additive white Gaussian noise (AWGN) with noise power σ_y^2.

The task of the receiver is to compute a suitable estimate $\hat{\mathbf{u}}$ of the original information sequence. A Turbo Equalizer achieves this by iterating between a soft-input/soft-output

Fig. 1. Convolutionally encoded transmission over an ISI channel.

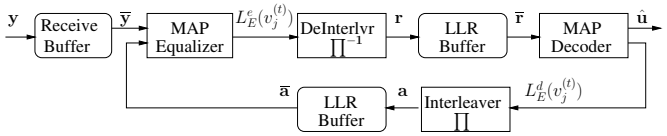


Fig. 2. Turbo Equalizer with receive and LLR buffers.

(SISO) equalizer and a SISO channel decoder, which exchange their beliefs about the probabilities of each coded bit $v_j^{(t)}$. The probabilities are expressed using Log-Likelihood Ratios (LLRs), denoted by $L(\cdot)$. The important underlying principle is that both components generate *extrinsic* LLRs $L_E(\cdot)$.

Fig. 2 shows the structure of the Turbo Equalizer: The quantized received sequence $\mathbf{y}$ is stored in the receive buffer. Using the sequence $\overline{\mathbf{y}}$ and the *a priori* LLRs $\overline{\mathbf{a}}$, which are read from the corresponding buffers, the MAP equalizer generates extrinsic LLRs $L_E^e(v_j^{(t)})$ of the coded bits. Their quantized deinterleaved representation $\mathbf{r}$ is written to a buffer. The decoder reads the values $\overline{\mathbf{r}}$ from the buffer and generates its own extrinsic estimate $L_E^d(v_j^{(t)})$ of the coded bits. The quantized interleaved sequence $\mathbf{a}$ of $L_E^d(v_j^{(t)})$ is written to the second LLR buffer, from which the equalizer reads the sequence $\overline{\mathbf{a}}$ in the following iteration. In the final iteration, the decoder will generate the *a posteriori* estimate $\hat{\mathbf{u}}$.

B. The Memory Channel

The quantized received values $\mathbf{y}$ and the LLRs $\mathbf{r}$ and $\mathbf{a}$ are stored in memory using a binary fixed-point representation. Commonly 2-complement representation or signed representation are used, where each soft value is stored using $N = d+f$ bits and d denotes decimal and f fractional bits. For example, for some value y, the 2-complement representation (2R) is written as

$$y = -2^{d-1}b_0 + \sum_{i=1}^{d-1} 2^{i-1}b_i + \sum_{i=1}^{f} 2^{-i}b_{d+i-1}, \qquad (1)$$

while the signed representation (SR) can be written as

$$y = -(2b_0 - 1)\left(\sum_{i=1}^{d-1} 2^{i-1}b_i + \sum_{i=1}^{f} 2^{-i}b_{d+i-1}\right), \qquad (2)$$

where b_i represents the stored bits.

If the buffer memories work perfectly, we have $\overline{\mathbf{y}} = \mathbf{y}$, $\overline{\mathbf{r}} = \mathbf{r}$ and $\overline{\mathbf{a}} = \mathbf{a}$, i.e. the read values match the written values. However, if the buffer memory is faulty, the read values will be distorted and $\overline{\mathbf{y}} \neq \mathbf{y}$, $\overline{\mathbf{r}} \neq \mathbf{r}$, and $\overline{\mathbf{a}} \neq \mathbf{a}$. Note that $\overline{\mathbf{y}}$, $\overline{\mathbf{r}}$ and $\overline{\mathbf{a}}$ can differ on each read access.

Following [3], [4] we assume that the distortion results from *uniformly and independently distributed* bit errors on the stored values. Before we analyze the effect of this distortion on the Turbo Equalizer, we will characterize it using the simple model shown in Fig. 3. The BPSK modulated values $x \in \{\pm 1\}$ are multiplied by the gain μ_y and transmitted over an AWGN channel with noise power σ_y^2. The noisy value y,

$$y = \mu_y x + n, \quad n \sim \mathcal{N}(0, \sigma_y^2), \qquad (3)$$

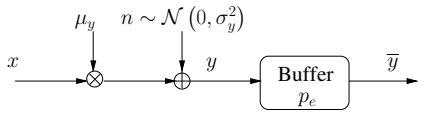


Fig. 3. Simplified model of BPSK transmission over AWGN channel with faulty buffer and bit error probability p_e.

is written to a buffer. The faulty buffer causes bit errors in the read values $\overline{y}$, where the probability of one bit being flipped is denoted by p_e. The channel from y to $\overline{y}$ is referred to as the *memory channel*.

It is easy to see, that the probability of $\overline{y} = y$, i.e. none of the N bits of y is affected by an error, is given as $(1-p_e)^N$. More generally, the probability α_i that i out of N bits are hit by an error is given as

$$\alpha_i = \binom{N}{i} p_e^i (1-p_e)^{N-i} \quad \text{for} \quad i = 0 \ldots N. \qquad (4)$$

In order to characterize the distortion, the capacity of the combination of Gaussian and memory channel is analyzed. For equally probable inputs $x = \pm 1$ and quantized output $\overline{y}$ the capacity can be computed by measuring the probability mass functions (PMFs) $P(\overline{y}|x=-1)$ and $P(\overline{y}|x=+1)$ and computing the mutual information [8]

$$I(X;\overline{Y}) = \frac{1}{2}\sum_{\overline{y}\in\mathcal{Y}} \left[P(\overline{y}|x=1)\mathrm{ld}\frac{2P(\overline{y}|x=1)}{P(\overline{y}|x=1)+P(\overline{y}|x=-1)} + P(\overline{y}|x=-1)\mathrm{ld}\frac{2P(\overline{y}|x=-1)}{P(\overline{y}|x=1)+P(\overline{y}|x=-1)} \right], \qquad (5)$$

where $\mathcal{Y}$ denotes the set of 2^N quantization values.

Fig. 4 shows the capacity of the BPSK input, quantized output AWGN channel (BAWGNQ), which is $I(X;Y)$ and serves as a reference, along with the capacities of combined BAWGNQ and memory channel for 2R and SR. The capacities are given for $p_e = 0.1$ and $p_e = 0.01$. For all cases, the gain μ_y is set to $\mu_y = 1$ and the binary representation is selected as $N = 8$ bit per quantized value with $d = f = 4$, such that there is no significant influence of quantization on the capacity.

Two conclusions can be drawn from Fig. 4: Firstly, an increased bit error probability p_e results in a decrease of the channel capacity. This is obvious, because the memory channel adds additional noise to the value y. The second observation

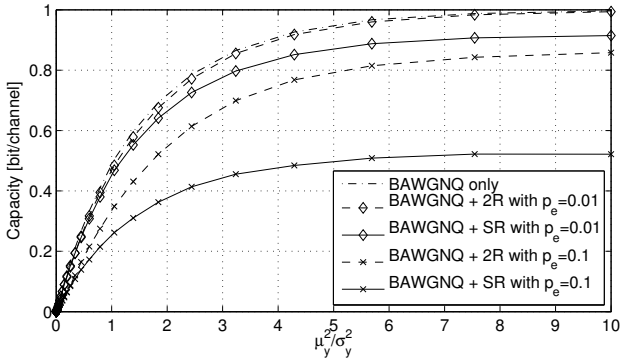
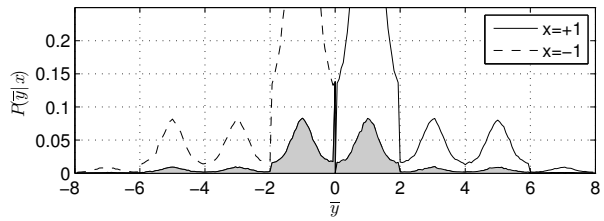
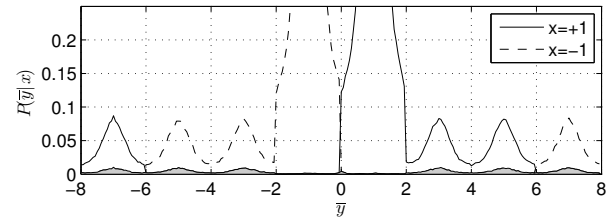


Fig. 4. Capacity of binary input AWGN channel for error-free and 8Bit-quantized erroneous storage. Comparison of 2-complement (2R) and sign-based (SR) representations with different bit error probabilities p_e ($\mu_y^2 = 1$).



(a) Signed representation.



(b) 2-complement representation.

Fig. 5. Probability mass function $P(\overline{y}|x)$ for signed and 2-complement representation with $p_e = 0.1$, $N = 8$, $d = f = 4$, $\sigma_y^2 = 0.1$.

is that the capacity reduction is less severe in the 2R case than in the SR case: For low noise, $1/\sigma_y^2 = 10$, and $p_e = 0.1$ it can be seen that the capacity is reduced to 0.52 for SR, but only to 0.91 for 2R. An intuitive explanation can be found by comparing the PMFs of $\overline{y}$ for SR and 2R. For both cases, Fig. 5 shows the measured PMFs for transmitted $+1$ and -1. Additionally the gray area marks the overlapping regions of $P(\overline{y}|x = +1)$ and $P(\overline{y}|x = -1)$. This area is significantly smaller for the 2R then for SR, which, according to (5), explains the reduced capacity. Another interpretation is that for 2R the redundancy due to the relatively large quantization range is used more effectively, and thus making its PMF more distinctive than the PMF of SR. A suitably designed algorithm may use this redundancy to compensate memory errors. Thus, the following discussion will be restricted to 2R.

C. PMF for Memory Channel

In order to derive transition metrics for the MAP algorithm, an analytical expression for the PMF $P(\overline{y}|x)$ is required. In general, this PMF can be written as

$$P(\overline{y}|x) = \sum_{i=0}^{N} \alpha_i P_i(y|x),$$

where $P_i(y|x)$ denotes the PMF for the case that i out of N bits are erroneous and α_i is given in (4). Then $P_0(y|x)$ is the PMF for the error-free case which is given as

$$P_0(y|x) = P(y|x) = \frac{1}{\sqrt{2\pi}\sigma_y} \exp\left(-\frac{1}{2\sigma_y^2}(y - \mu_y x)^2\right).$$

For the remaining $P_i(y|x)$, $i > 0$, the expressions can be found by considering each possible bit error constellation and treating errors that result from this constellation as a shift of the mean μ_y of the original PMF $P(y|x)$ [4].

For example, according to (1) a single error in the MSB b_0 of y causes a shift of $\pm 2^{d-1}$ of y. This can be written as

$$\overline{y} = \left\{ \begin{array}{l} y - 2^{d-1} \text{ for } b_0 = 0 \\ y + 2^{d-1} \text{ for } b_0 = 1 \end{array} \right\} = y + (2b_0 - 1)2^{d-1}.$$

Corresponding expressions can be found for single errors in b_1 to b_{N-1}. The PMF $P_1(y|x)$ can then be found to be

$$P_1(y|x) = \frac{1}{\sqrt{2\pi}\sigma_y N}\left[\exp\left(-\frac{1}{2\sigma_y^2}(y - (\mu_y x - \tilde{b}_0 2^{d-1}))^2\right) + \sum_{m=1}^{N-1} \exp\left(-\frac{1}{2\sigma_y^2}(y - (\mu_y x + \tilde{b}_m 2^{d-1-m}))^2\right)\right], \quad (6)$$

where $\tilde{b}_m = 2b_m - 1$ is the m-th bit of $\overline{y}$ modulated to ± 1.

Expressions for $P_2(y|x)$ to $P_N(y|x)$ can be derived in a similar way. However, the complexity of $P(\overline{y}|x)$ directly affects the computational complexity of the MAP algorithm. Therefore, given (4), we assume that $p_e \ll 0.5$, such that the probability of more than one bit error is negligible, i.e. $\alpha_2 = \ldots = \alpha_N \approx 0$. Then the resulting approximated PMF becomes

$$P(\overline{y}|x) \approx \frac{\alpha_0}{\sqrt{2\pi}\sigma_y} \exp\left(-\frac{1}{2\sigma_y^2}(\overline{y} - \mu_y x)^2\right) + \frac{\alpha_1}{\sqrt{2\pi}\sigma_y N}\left[\exp\left(-\frac{1}{2\sigma_y^2}(\overline{y} - (\mu_y x - \tilde{b}_0 2^{d-1}))^2\right) + \sum_{m=1}^{N-1} \exp\left(-\frac{1}{2\sigma_y^2}(\overline{y} - (\mu_y x + \tilde{b}_m 2^{d-1-m}))^2\right)\right]. \quad (7)$$

III. Fault Tolerant Turbo Equalization

A Turbo Equalizer can be implemented by using the BCJR algorithm [9] for equalizer and decoder. In this case the extrinsic LLRs $L_E^e(v_j^{(t)})$ and $L_E^d(v_j^{(t)})$ are generated by doing forward and backward recursions on the underlying trellis. For the equalizer this trellis is defined by the channel, which has in case of BPSK 2^{M-1} states. For the decoder, the trellis is usually the 2^ν state trellis defined by the encoder of the constraint length ν convolutional code.

A. Conventional Log MAP Algorithm

Usually the BCJR algorithm is implemented in Log-domain using the Log MAP approach [10]. For the equalizer, the transition metric $\gamma^{(l)}(p,q)$ of a transition from trellis state p to state q at time instant l is given by [11]

$$\gamma^{(l)}(p,q) = \left\{ \begin{array}{l} \frac{(y^{(l)} - s(p,q))^2}{2\sigma_y^2} + \frac{1}{2}a^{(l)}x(p,q),\ 1 < l \le n_c T \\ \frac{(y^{(l)} - s(p,q))^2}{2\sigma_y^2},\ n_c T < l \le n_c T + M - 1, \end{array} \right. \quad (8)$$

where $s(p,q)$ and $x(p,q)$ are channel output and input associated with the transition from p to q, $y^{(l)}$ and $a^{(l)}$ are received soft bit from the channel and the corresponding a priori value generated by the decoder, and σ_y^2 is the channel noise power.

For the decoder, the transition metric is given by

$$\gamma^{(t)}(p,q) = \frac{1}{2}\sum_{j=1}^{n_c} r_j^{(t)} \tilde{v}_j(p,q),\ 1 \le t \le T, \quad (9)$$

where $\tilde{v}_j(p,q) = 2v_j(p,q) - 1$ is the BPSK-modulated j-th code bit associated with the transition from p to q, and $r_j^{(t)}$ is the corresponding extrinsic value from the equalizer. Note, that (8) and (9) can be derived using the assumption that $a^{(l)}$ and $r_j^{(t)}$ are samples of a consistent Gaussian distribution.

B. Fault Tolerant MAP Algorithm

In case of a system with faulty buffers, the input values of equalizer and decoder are distorted by the memory channel. The transition metric $\overline{\gamma}(p,q)$, which considers this additional distortion, can be derived with the help of (7) as follows: For the equalizer, $\overline{\gamma}^{(l)}(p,q)$ can be written as

$$\overline{\gamma}^{(l)}(p,q) = \log P(\overline{y}^{(l)}|x(p,q)) + \log P(\overline{a}^{(l)}|x(p,q)), \quad (10)$$

where $P(\overline{y}^{(l)}|x(p,q))$ is the PMF of the combination of communication and memory channel, and $P(\overline{a}^{(l)}|x(p,q))$ the PMF of a priori LLR and memory channel. Based on (7) the first summand in (10) is given by

$$\log P(\overline{y}^{(l)}|x(p,q)) = \log\left[\frac{\alpha_0}{\sqrt{2\pi}\sigma_y}\exp\left(-\frac{(\overline{y}^{(l)}-s(p,q))^2}{2\sigma_y^2}\right) + \frac{\alpha_1}{\sqrt{2\pi}\sigma_y N_y}\left[\exp\left(-\frac{(\overline{y}^{(l)}-(s(p,q)-\tilde{b}_0 2^{d-1}))^2}{2\sigma_y^2}\right) + \sum_{m=1}^{N_y-1}\exp\left(-\frac{(\overline{y}^{(l)}-(s(p,q)+\tilde{b}_m 2^{d-1-m}))^2}{2\sigma_y^2}\right)\right]\right].$$

By applying the logarithm and removing constant terms, this can be simplified to

$$\log P(\overline{y}^{(l)}|x(p,q)) = \overset{*}{\max_{1\le m<N_y}}\left\{\kappa_y - \frac{(\overline{y}^{(l)}-s(p,q))^2}{2\sigma_y^2}, -\frac{(\overline{y}^{(l)}-(s(p,q)-\tilde{b}_0 2^{d-1}))^2}{2\sigma_y^2}, -\frac{(\overline{y}^{(l)}-(s(p,q)+\tilde{b}_m 2^{d-1-m}))^2}{2\sigma_y^2}\right\}, \quad (11)$$

where $\overset{*}{\max}$ denotes the $\max$-operation with Jacobian correction term [10], $\tilde{b}_0$ and $\tilde{b}_m$ are the BPSK-modulated bits of the 2R of $\overline{y}^{(l)}$ and $\kappa_y = \log\frac{\alpha_0}{\alpha_1/N_y}$. The number of bits used for $\overline{y}^{(l)}$ is denoted by N_y, and α_0 and α_1 are given by (4).

For the second summand in (10), under the assumption, that $a^{(l)}$ is a sample of a consistent Gaussian distribution, i.e. the mean μ_a of $a^{(l)}$ equals half of the variance σ_a^2, $\mu_a = \sigma_a^2/2$, the simplified expression can be determined as

$$\log P(\overline{a}^{(l)}|x(p,q)) = \overset{*}{\max_{1\le m<N_a}}\left\{\kappa_a - \frac{(\overline{a}^{(l)}-\frac{\sigma_a^2}{2}x(p,q))^2}{2\sigma_a^2}, -\frac{(\overline{a}^{(l)}-(\frac{\sigma_a^2}{2}x(p,q)-\tilde{b}_0 2^{d-1}))^2}{2\sigma_a^2}, -\frac{(\overline{a}^{(l)}-(\frac{\sigma_a^2}{2}x(p,q)+\tilde{b}_m 2^{d-1-m}))^2}{2\sigma_a^2}\right\}, \quad (12)$$

where $\tilde{b}_0$ and $\tilde{b}_m$ are the BPSK-modulated bits of $\overline{a}^{(l)}$.

Using the same approach, a fault tolerant Log MAP decoder can be implemented. In this case the transition metric becomes

$$\overline{\gamma}^{(t)}(p,q) = \sum_{j=1}^{n_c}\overset{*}{\max_{1\le m<N_r}}\left\{\kappa_r - \frac{(\overline{r}_j^{(t)}-\frac{\sigma_r^2}{2}\tilde{v}_j(p,q))^2}{2\sigma_r^2}, -\frac{(\overline{r}_j^{(t)}-(\frac{\sigma_r^2}{2}\tilde{v}_j(p,q)-\tilde{b}_0 2^{d-1}))^2}{2\sigma_r^2}, -\frac{(\overline{r}_j^{(t)}-(\frac{\sigma_r^2}{2}\tilde{v}_j(p,q)+\tilde{b}_m 2^{d-1-m}))^2}{2\sigma_r^2}\right\}, \quad (13)$$

where $\tilde{b}_0$ and $\tilde{b}_m$ are the BPSK-modulated bits of the 2R of the corresponding $\overline{r}_j^{(t)}$, and σ_r^2 the variance of $r_j^{(t)}$.

FT Turbo equalization, which provides a compensation for the bit errors resulting from faulty buffer memory, is based on the Log MAP equalizer using (10), (11) and (12), and the Log MAP decoder using (13). The approximations used in the derivation, are the assumption in (7), that only single errors occur, and the approximation of the sum of exponentials by the Jacobian logarithm in (11), (12) and (13). Further simplification can be achieved by only considering errors in the decimal part, such that $1 \le m < d$ in (11), (12) and (13). The impact of these approximations is negligible, as long as p_e is sufficiently small ($p_e \le 0.1$).

Finally, $\overset{*}{\max}$ may be replaced by a simple $\max$, which results in about $0.1dB$ to $0.2dB$ loss of BER.

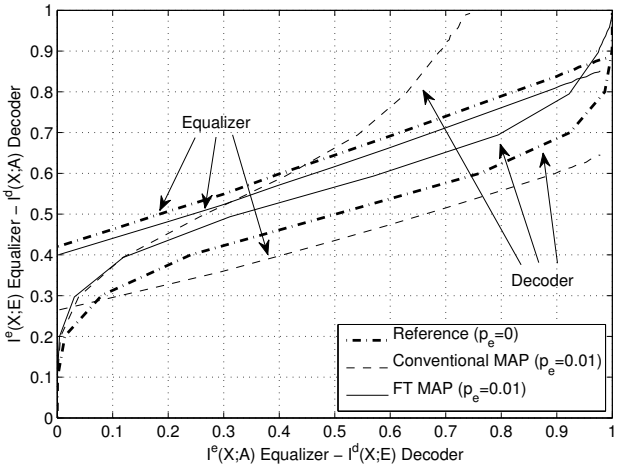


Fig. 6. Comparison of EXIT charts at $E_b/N_0 = 5.5dB$.

IV. Simulation Results

This section presents simulation results for the proposed FT Turbo Equalizer. For the simulations, a $R = 1/2$ rate non-systematic, terminated convolutional code with generators $\mathbf{G} = (5_8, 7_8)$, blocklength $T = 2^{15}$, BPSK-modulation, and random interleaver is used. The impulse response of the length $M = 5$ ISI channel is $\mathbf{h} = [0.227, 0.46, 0.688, 0.46, 0.227]$. 2R is used for $\mathbf{y}$, $\mathbf{r}$ and $\mathbf{a}$, where the quantization is set to $N_y = d + f = 4 + 4$, $N_r = 6 + 4$, $N_a = 7 + 4$. For the simulations perfect knowledge of the channel and of the variances σ_y^2, σ_r^2, and σ_a^2 is assumed. In a real system, σ_y^2, σ_r^2, and σ_a^2 have to be estimated before the corresponding values are written to the buffer.

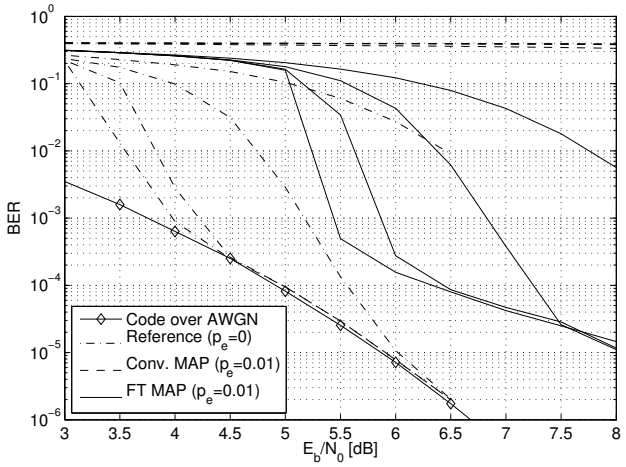


Fig. 7. BERs of conventional and FT Turbo Equalizer with $p_e = 0.01$.

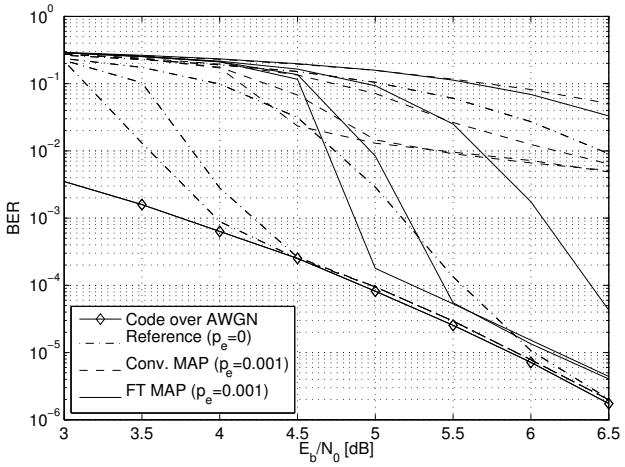


Fig. 8. BERs of conventional and FT Turbo Equalizer with $p_e = 0.001$.

A. EXIT chart

Firstly, Fig. 6 shows the extrinsic information transfer (EXIT) chart of the Turbo Equalizer at $E_b/N_0 = 5.5dB$, where mutual information of a priori and extrinsic information of equalizer and decoder, $(I^e(X;A), I^e(X;E))$ and $(I^d(X;A), I^d(X;E))$, are plotted against each other. The reference is the chart of the Turbo Equalizer with perfect memory ($p_e = 0$). For the case of faulty memory ($p_e = 0.01$), the charts of Turbo Equalizer using conventional MAP equalization/decoding is compared to FT MAP equalization/decoding. For the conventional MAP the curves cross very early, which indicates, that the Turbo Equalizer will not converge. The chart of the FT MAP, however, indicates that convergence is possible in this case, although the reference cannot be reached.

Given the fact that for increased E_b/N_0 the curve of the equalizer is roughly just shifted upwards, while the decoder curve remains unchanged, it can be seen, that the conventional MAP for $p_e = 0.01$ cannot achieve good performance even for high E_b/N_0: Even for very good a priori $I^d(X;A) \approx 1$, the decoder will not produce good extrinsic information, but will even degrade the quality of its extrinsic values ($I^d(X;E) \approx 0.73$). On the other hand, the FT MAP can reach the point $(1,1)$ given suitable E_b/N_0.

B. Bit Error Rate (BER)

Fig. 7 and 8 show the BER performance of Turbo equalization with conventional and FT MAP algorithm and memory error probability of $p_e = 0.01$ and $p_e = 0.001$. The BER of Turbo equalization with error-free memory is provided as a reference. In all cases, the maximum number of iterations is 14, but only the BERs after iterations 2, 4, 8, and 14 iterations are shown.

For $p_e = 0.01$ it can be seen, that the conventional Turbo Equalizer does not converge in the considered E_b/N_0-range. The FT Turbo Equalizer, however, converges and shows a loss of about $2dB$ compared to the Turbo Equalizer over error-free memory. In case of smaller error probability of $p_e = 0.001$, the conventional Turbo Equalizer still reveals a significant error floor. The FT Turbo Equalizer on the other hand can almost reach the floor of the reference, while the threshold is increased by about $1.5dB$ compared to the reference.

V. Conclusions

In order to compensate the effects of unreliable buffer memory and to avoid performance degradation of the signal processing algorithms a co-design of hardware and algorithm is required. This co-design has been studied for the case of Turbo equalization with uniformly and independently distributed bit errors in the buffer memories. A fault tolerant Turbo Equalizer has been described, which effectively compensates memory errors. For example, given a memory bit error probability of $p_e = 0.001$, the FT Turbo Equalizer shows a floor loss of only $0.2dB$ compared to the reference, while the conventional Turbo Equalizer exhibits a strongly increased error floor.

References

[1] M. Vollmer, M. Haardt, and J. Götze, "Comparative study of joint-detection techniques for TD-CDMA based mobile radio systems," *IEEE JSAC*, vol. 19, no. 8, pp. 1461 –1475, Aug. 2001.

[2] J. Geldmacher, K. Hueske, and J. Goetze, "An adaptive and complexity reduced decoding algorithm for convolutional codes and its application to digital broadcasting systems," in *Int. Conf. on Ultra Modern Telecommunications (ICUMT2009)*, St. Petersburg, Russia, Oct. 2009.

[3] A. K. Djahromi, A. M. Eltawil, F. J. Kurdahi, and R. Kanj, "Cross layer error exploitation for aggressive voltage scaling," in *8th Int. Symp. on Quality Electronic Design (ISQED '07)*, 2007, pp. 192 –197.

[4] A. Hussien, M. Khairy, A. Khajeh, A. Eltawil, and F. Kurdahi, "Combined channel and hardware noise resilient viterbi decoder," in *Asilomar Conf. on SS&C*, Pacific Grove, CA, Nov 2010.

[5] S. Ghosh and K. Roy, "Parameter variation tolerance and error resiliency: New design paradigm for the nanoscale era," *Proceedings of the IEEE*, vol. 98, no. 10, pp. 1718 –1751, 2010.

[6] C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *ETT*, vol. 6, no. 5, pp. 507–511, 1995.

[7] M. Tuchler and A. C. Singer, "Turbo equalization: An overview," *IEEE Trans. on Inf. Theory*, vol. 57, no. 2, pp. 920 –952, 2011.

[8] T. M. Cover and J. A. Thomas, *Elements of Inf. Theory*. Wiley, 2006.

[9] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284 – 287, March 1974.

[10] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal map decoding algorithms operating in the log domain," in *IEEE ICC'95*, vol. 2, Seattle, June 1995, pp. 1009–1013.

[11] G. Bauch, H. Khorram, and J. Hagenauer, "Iterative equalization and decoding in mobile communications systems," in *Proc. European Personal Mobile Commun. Conf, ITG Fachberichte*, 1997.