

# A Class of Low Power Error Compensation Iterative Decoders

Amr M. A. Hussien, Muhammad S. Khairy, Amin Khajeh, Ahmed M. Eltawil, and Fadi J. Kurdahi

University of California Irvine, Irvine, CA, USA  
 { ahussien, mkhairi, akhaje, aeltawil, kurdahi }@uci.edu

**Abstract**—Recent power reduction techniques aggressively modulate the supply voltage of embedded buffering memories allowing acceptable hardware errors to flow through the processing chain. In this paper, we introduce a class of modified Turbo and LDPC decoders that provide significant improvements over standard decoders in the presence of hardware noise. Simulation results show a consistent improvement in the BER performance of the modified decoders across all SNRs with very small area and power overheads as compared to the conventional decoders.

## I. INTRODUCTION

Recently, dynamic voltage and frequency scaling (DVFS) [1] has been widely used to reduce both the dynamic and leakage components of power consumption, which is a key factor for mobile applications. To improve the effectiveness of DVFS, aggressive voltage scaling (AVS)[3] has been proposed where the range of voltage reduction is increased to a point that allows some degree of controlled hardware errors to propagate through the system (as long as an overall quality of service (QoS) metric is achieved). This approach allows hardware error tolerance based on the measured Signal to Noise Ratio (SNR); i.e for high SNR, we can have higher acceptable hardware errors with the benefit of additional power saving.

In modern Systems on Chip (SoC) applications, a large portion of the design consists of embedded memory cores as documented by the International Technology Roadmap for Semiconductors (ITRS) report [2]. Thus, a significant portion of the SoC performance and area statistics are dominated by embedded memory statistics. Moreover, memories are highly structured both in their architecture and access patterns, and therefore, provide structured means of identifying the impact of hardware induced errors on the overall behavior of the system. With these facts in mind, several publications addressed the failure mechanism of embedded memories under aggressive voltage scaling [4-6] for the purposes of low power operation.

In previous publications [3,4], the authors have shown that by incorporating application knowledge in determining the required operational failure rates of memories, an overall system power savings of 20% can be achieved by relaxing the circuit constraints on the data buffering memories in a WCDMA modem. While these facts illustrate the benefits of AVS, they did not consider the impact of hardware errors on the Forward Error Correction (FEC) schemes that are traditionally optimized for Gaussian noise and do not account

for the noise introduced by memory failures. In this paper, we address this issue by presenting a unified statistical model for the combined channel noise and memory failure noise. Additionally, we introduce the necessary modification in a class of iterative decoders; namely Turbo and LDPC decoders (which are widely used decoders in communication systems) to accommodate the new distribution. We show that with negligible overhead in the decoder structure, a significant gain in the bit error rate (BER) performance in the presence of faulty memories can be achieved.

The paper is organized as follows: Section II presents the system overview, a review of the memory error modeling and the general formulation of the combined distribution of the memory error and channel noise. This includes the general formulation of the resulting distribution. In sections III and IV, we present the modification in two widely-used classes of iterative decoders; namely Turbo and LDPC decoders respectively. Sections V and VI provides the simulations results and implementation overheads of the proposed decoders, and section VII provides the concluding remarks.

## II. SYSTEM OVERVIEW

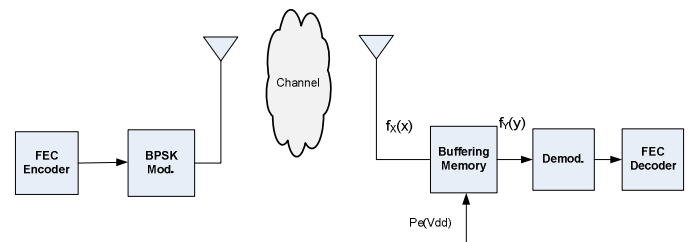


Figure 1: System Model

To illustrate the idea, we introduce a simple communication system consisting of FEC encoding and BPSK modulation as shown in Figure 1. At the front end of the receiver side, a buffering memory exists before the decoder. The communication channel can be either an AWGN or a fading channel. The buffering memory, at the receiver front end, introduces random uniform errors depending on the supply voltage scaling. The distribution of memory errors due to voltage over-scaling has been independently verified by numerous publications and confirmed by silicon trials [3]-[5]. For this work, we set up an HSPICE circuit simulation in 65nm technology using predictive models to quantify the effect of voltage over-scaling on embedded memories error rates. Figure 2 illustrates the memory error rate as well as the

power saving for different values of the supply voltage. It is shown that reducing the supply voltage by 25% from the nominal voltage (1.0 V) results in a power saving of 40% and a memory error probability of  $10^{-6}$ .

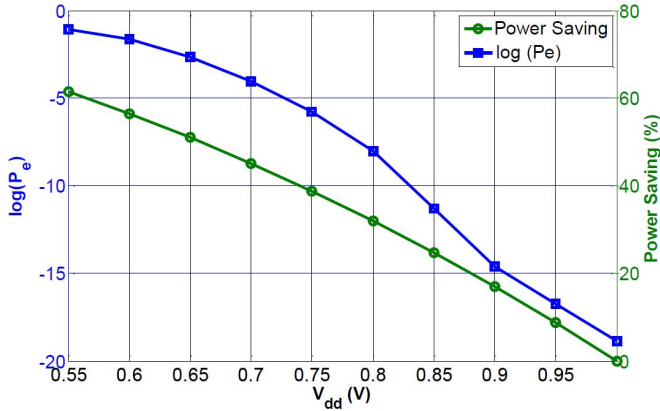


Figure 2 Memory error rate vs supply voltage

The operating supply voltage can be adjusted based on the signal to noise ratio and the required QoS. At high SNR, a controllable amount of error can be introduced in the hardware (by voltage over-scaling) with the goal of reducing power, while maintaining the required BER. This additional hardware error can be considered as an extra contribution to the channel noise.

Depending on the noise distribution introduced from the channel  $f_X(x)$ , and the noise introduced by memory, the data at the memory output has a new distribution  $f_Y(y)$ . The noise introduced by memory is dependent on the memory probability of error  $P_e(V_{dd})$  as shown in Figure 2. Then, based on the resultant distribution  $f_Y(y)$ , the FEC decoder is modified to accommodate for the hardware error introduced in the memory.

#### Statistical Noise Model

The memory model assumes one quantized signed word per memory location with  $(N, r)$  quantization. The integer part consists of  $d$  bits while the fractional part has  $r$  bits such that  $N = d + r$ . We assume that the values stored in memory ( $X$ ) have a general distribution of  $f_X(x)$ . These numbers are the channel outputs after the A/D conversion. As a result, the distribution function  $f_X(x)$  is indeed a Probability Mass Function (PMF) that demonstrates the probability of  $X$  taking any of the  $2^N$  possible values. In [4], it is shown that the error in the memory (as a result of voltage over-scaling) is spatially uniformly distributed. Therefore, all bits stored in memory have the same probability of  $P_e(V_{dd})$  to be flipped over to the wrong value. Thus, considering one location of  $N$  bits, the entire bits located from 0 to  $N - 1$  at any location inside the memory, have the same probability of  $P_e$  to be flipped to an incorrect bit value.

Considering one memory location with  $N$  bits, we can have one or more bit flipped and the distribution of the read data from that faulty memory is expressed as:

$$f_Y(y) = \sum_{k=0}^N P(k) f_Y^k(y) \quad (1)$$

where  $P(k)$  is the probability of having  $k$  bit flips simultaneously and  $f_Y^k(y)$  is the distribution of data when  $k$  bit flips occur at one word, with  $0 \leq k \leq N$ . The probability  $P(k)$  is calculated easily such that the probability of having  $k$  errors or  $k$  bit flips, out of  $N$  bits is given by

$$P(k) = P_e^k (1 - P_e)^{N-k} \quad (2)$$

Equation (2) considers only one combination of  $k$  bit flips. The overall effect of all  $k$ -bit flips is considered in (3). The detailed derivation of the memory output distribution is found in [7]. The generic  $k$ -bit flips distribution is

$$f_Y^k(y) = \frac{1}{k!} \sum_{n_1=0}^{N-1} \sum_{\substack{n_2=0, \\ n_2 \neq n_1}}^{N-1} \dots \sum_{\substack{n_k=0, \\ n_k \neq n_1, \\ \dots, n_k \neq n_{k-1}}}^{N-1} f_{Y_{n_1, n_2, \dots, n_k}}(y) \quad (3)$$

where in general  $f_{Y_{n_1, n_2, \dots, n_k}}(y)$  is the resultant distribution when bit flips at specific bit locations  $n_1, n_2, \dots, n_k$  occur.

It is important to note that the effect of memory error becomes more significant at higher SNR values as illustrated in Figure 3, which depicts the effect on the data distribution with one bit flip. Furthermore, errors in the sign bit and most significant bits are more severe than errors in other locations. As indicated by the irregularities in the tail of the distributions shown in Figure 3, these bit-flip/errors are projected in the data distribution as scaled, decoupled and shifted (at the location of the bit flip  $\pm 2^{N-r}$ ) versions of the original data distribution [7]. More details about the efficiency and accuracy of this model are mentioned in [7].

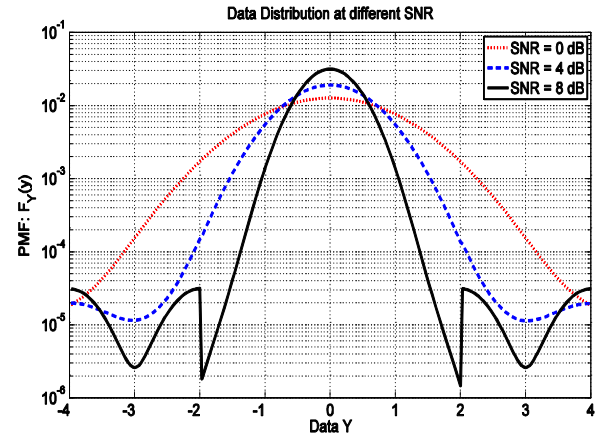


Figure 3 Data distribution after memory for one bit flip

### III. MODIFIED TURBO DECODER

In prior work, the authors considered the impact of memory errors on convolutional decoders [8]. In this paper, we consider iterative decoders. Specifically, this section focuses on Turbo decoders.

Generally speaking, a generic Turbo decoder architecture consists of two Soft Input Soft Output (SISO) decoders with interleavers. In the iterative decoding process, each SISO decoder generates a soft estimation about the information bit in a form of Log Likelihood Ratio (LLR). Extrinsic

information is generated by each SISO decoder and then passed to the other decoder after (de)interleaving. There are two widely used Turbo decoding schemes; namely Maximum A-posteriori (MAP) [9] and Soft Output Viterbi Algorithm (SOVA) [10]. The SOVA based decoder has less computation compared to the original MAP decoder at the cost of slightly degraded performance. A commonly used MAP approximation approach is the MAX-Log MAP algorithm [11] which can be considered as two Viterbi decoders; one in the forward direction and one in the backward direction.

The basic idea of the MAX Log MAP algorithm is to calculate the branch metrics (BM<sub>s</sub>) based on the channel values  $r_{ch}$  and the extrinsic likelihoods  $L_{ex}$ .

$$BM = f_1(r_{ch}, L_{ex}) \quad (4)$$

Following that, the forward and backward state metrics ( $FW$  and  $BK$ ) are updated based on the previous (next) forward (backward) metrics as well as the current branch metrics

$$FW_{M,t} = f_2(FW_{M,t-1}, BM_t) \quad (5)$$

$$BK_{M,t-1} = f_3(BK_{M,t}, BM_t) \quad (6)$$

Where  $M=\{0,1,\dots,N-1\}$  for  $N$  states. The calculation of the forward (backward) metrics is a simple Add-Compare and Select (ACS) operation which is radix-2 or radix-4 for binary and double binary codes respectively. The log likelihood ratios (LLR) are computed for different symbols based on equations (4)-(6) at each time. For binary Turbo codes, one LLR value is computed at each time and three LLR values are computed for double binary Turbo codes. In the next subsection, we introduce the memory error resilient MAP algorithm then show how this modification reflects the MAX Log MAP algorithm.

#### A. Modified MAP algorithm

To simplify the discussion, we consider only the effect of the dominant error, i.e. a single error in the sign bit. The same analysis can be easily generalized to a higher number of bit flips, although as discussed in section II, the contribution of lower order bits (LSBs) is progressively insignificant.

The LLR of an information bit  $u_k$  at  $k^{th}$  time is computed as

$$LLR = \frac{\sum_{(s'_{k-1}, s_k \leftrightarrow u_k = +1)} (\alpha_{k-1}(s') \cdot \gamma_{k-1}(s' \rightarrow s) \cdot \beta_k(s))}{\sum_{(s'_{k-1}, s_k \leftrightarrow u_k = -1)} (\alpha_{k-1}(s') \cdot \gamma_{k-1}(s' \rightarrow s) \cdot \beta_k(s))} \quad (7)$$

The channel probability contributes to the calculation of the branch metric  $\gamma_k(s' \rightarrow s)$  term as shown in (8).

$$\gamma_k(s' \rightarrow s) = P_{y|x}(y_k|x_k) \cdot P_x(x_k) \quad (8)$$

Where  $\gamma_k(s' \rightarrow s)$  is the branch metric at  $k^{th}$  time for a specific transition from state  $s'$  to state  $s$  in the trellis diagram,  $\alpha_k(s')$  is the forward metric of state  $s'$  at  $k^{th}$  time,  $\beta_k(s)$  is the backward metric of state  $s$  at  $k^{th}$  time,  $x_k$  is the transmitted symbol corresponding to this specific transition, and  $y_k$  is the received symbol at  $k^{th}$  time. In fact, at each time slot, we have multiple transmitted and received symbols represented in systematic and parity bits. For simplicity, our notation  $y_k$  refers to any received symbol at  $k^{th}$  time. Now, consider  $y_k$  is the received symbol at the memory output instead. For a fixed

precision of  $(N, r)$  and  $d = N - r$ , the effect of a *single error at the sign bit* is the dominant error over all other error combinations. This error introduces two artifacts in the distribution at  $-2^{d/2}$  and  $2^{d/2} - 2^{-r/2}$  with height of  $P_e$ . Thus, the distribution is dominated by the mainlobe around  $y_k - x_k = 0$  and the sidelobes around  $y_k - x_k = \pm 2^{d/2}$  neglecting the term  $2^{-r/2}$ . So, from (1) the approximated new channel probability is represented by three terms as follows:

$$\begin{aligned} P_{y|x}(y_k|x_k) &\approx P(0)f_Y^0(y_k) + P(1)f_Y(y_k - 2^{d/2}) \\ &\quad + P(1)f_Y(y_k + 2^{d/2}) \\ P_{y|x}(y_k|x_k) &= \frac{\alpha_1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_k-x_k)^2}{2\sigma^2}} + \frac{\alpha_2}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_k-x_k-\mu_1)^2}{2\sigma^2}} \\ &\quad + \frac{\alpha_2}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_k-x_k+\mu_1)^2}{2\sigma^2}} \end{aligned} \quad (9)$$

where  $\mu_1 = 2^{d/2}$ ,  $\alpha_1 = P(0) = (1 - P_e)^N$ , and  $\alpha_2 = P(1) = P_e(1 - P_e)^{N-1}$

The remainder of the computations remains unchanged including the forward and backward metrics.

#### B. Architectural Modifications

When the MAX Log MAP algorithm is applied, the change in the distribution affects only the computation of the branch metrics and the extrinsic likelihood metrics. In the modified decoder, an additional MAX Log approximation step is applied prior to the conventional one in MAX Log algorithm. The log likelihood function is represented as

$$\ln P_{y|x}(y_k|x_k) = \ln \left\{ \begin{aligned} &\frac{\alpha_1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_k-x_k)^2}{2\sigma^2}} \\ &+ \frac{\alpha_2}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_k-x_k-\mu_1)^2}{2\sigma^2}} \\ &+ \frac{\alpha_2}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_k-x_k+\mu_1)^2}{2\sigma^2}} \end{aligned} \right\} \quad (10)$$

$$\ln P_{y|x}(y_k|x_k) = \ln \left\{ \begin{aligned} &e^{\ln \frac{\alpha_1}{\sqrt{2\pi\sigma^2}} - \frac{(y_k-x_k)^2}{2\sigma^2}} \\ &+ e^{\ln \frac{\alpha_2}{\sqrt{2\pi\sigma^2}} - \frac{(y_k-x_k-\mu_1)^2}{2\sigma^2}} \\ &+ e^{\ln \frac{\alpha_2}{\sqrt{2\pi\sigma^2}} - \frac{(y_k-x_k+\mu_1)^2}{2\sigma^2}} \end{aligned} \right\} \quad (11)$$

Applying the first step of MAX Log approximation to the log likelihood function results in

$$\ln P_{y|x}(y_k|x_k) \approx \max \left\{ \begin{aligned} &\ln \frac{\alpha_1}{\sqrt{2\pi\sigma^2}} - \frac{(y_k-x_k)^2}{2\sigma^2}, \\ &\ln \frac{\alpha_2}{\sqrt{2\pi\sigma^2}} - \frac{(y_k-x_k-\mu_1)^2}{2\sigma^2}, \\ &\ln \frac{\alpha_2}{\sqrt{2\pi\sigma^2}} - \frac{(y_k-x_k+\mu_1)^2}{2\sigma^2} \end{aligned} \right\} \quad (12)$$

Instead of the direct cross-correlation in the conventional branch metrics calculations, a new metric based on the new distribution is calculated for each symbol for both cases of transmitted symbols  $x_j = \pm 1$ . Consider the  $j^{th}$  transmitted symbol (systematic or parity) at any time, the new metric is:

$$M_j = x_j y_j + \max \left\{ \begin{aligned} &\sigma^2 \ln \frac{\alpha_1}{\alpha_2} \cdot \mu_1 (y_j - x_j) - \frac{\mu_1^2}{2}, \\ &-\mu_1 (y_j - x_j) - \frac{\mu_1^2}{2} \end{aligned} \right\} \quad (13)$$

For the approximated case,  $\ln \frac{\alpha_1}{\alpha_2} = \ln \frac{1-P_e}{P_e} \approx -\ln P_e$

$$\gamma_k(s' \rightarrow s) = \sum_j M_{k,j} + l_{\text{extrin},k}(i) \quad , s' \rightarrow s \leftrightarrow u_k = i \quad (14)$$

The summation in (14) is performed over all systematic and parity metrics at the  $k^{\text{th}}$  time corresponding to the transition from  $s'$  to  $s$ . The forward  $\alpha_k(s')$  and backward  $\beta_k(s)$  metric updates are modified as follows

$$\alpha_k(s) = \max_{s'} \{\gamma_{s' \rightarrow s,k} + \alpha_{k-1}(s')\} \quad (15)$$

$$\beta_k(s') = \max_s \{\gamma_{s' \rightarrow s,k+1} + \beta_{k+1}(s)\} \quad (16)$$

While the LLR computation becomes

$$llr_k(i) = \max_{k, \forall s' \rightarrow s \leftrightarrow u_k = i} \{\alpha_k(s') + \gamma_k(s' \rightarrow s) + \beta_{k+1}(s)\} \quad (17)$$

The extrinsic likelihood computation is modified as follows:

$$l_{\text{ex},k}(i) = llr_k(i) - l_{\text{extrin},k}(i) - \sum_j M_{sj,k}(i) \quad (18)$$

where  $l_{\text{ex},k}(i)$  is the output extrinsic likelihood for symbol  $i$  at time  $k$ ,  $l_{\text{extrin},k}(i)$  is the input extrinsic likelihood for symbol  $i$  at time  $k$ , and  $M_{sj,k}(i)$  is the metric for the  $j^{\text{th}}$  systematic bit at time  $k$  corresponding to an input value of  $i$ . Thus, the modified decoder requires a change in the branch metric computation unit as well as a change in the extrinsic likelihood computation unit. The rest of the decoder remains unchanged.

#### IV. MODIFIED LDPC DECODER

In this section, we propose the necessary changes in the LDPC decoder to accommodate the combined channel and memory error effect. A binary  $(n,k)$  LDPC is a linear block code represented by a sparse parity check matrix  $H$  of size  $m \times n$ . The  $H$  matrix is represented by a tanner graph with variable nodes and check nodes and a value of 1 is presented by a connection between the corresponding variable node and check node. The original decoding scheme is based on the belief propagation approach [12], where soft messages are exchanged among the variable and check nodes. Some approximations were adopted to simplify the hardware implementation such as the min-sum decoding [13] and the offset min-sum decoding [14]. Recently, a modified architecture for LDPC codes was proposed in [15] and [16] to enable Turbo-like decoding of LDPC codes. It is known as the Turbo Decoding Message Passing (TDMP) algorithm. It assumes a certain structure of the  $H$  matrix. A summary of belief propagation, min-sum and TDMP decoding as well as the modified versions for the combined error distribution is shown in the next subsections.

##### A. Belief propagation algorithm

Let  $\mathcal{M}(n)$  denotes the set of check nodes connected to the variable node  $n$  and  $\mathcal{N}(m)$  denotes the set of variable nodes connected to the check node  $m$ . Let  $q_{n \rightarrow m}(0)$  and  $q_{n \rightarrow m}(1)$  denote the message from the variable node  $n$  to the check node  $m$  indicating the likelihood of the variable node  $n$  being zero or one respectively. Similarly, let  $r_{n \rightarrow m}(0)$  and  $r_{n \rightarrow m}(1)$  denote the message from the  $m^{\text{th}}$  check node to the  $n^{\text{th}}$  variable node indicating the likelihood of the  $n^{\text{th}}$  variable node being

zero or one respectively. Assume that the transmitted sequence is  $[x_1, x_2 \dots x_n]$ , and the received sequence is  $[y_1, y_2 \dots y_n]$ . In the log-domain, the exchanged LLRs between variable and check nodes and check to variable nodes are defined as follows:

$$Z_{n \rightarrow m}(x_n) = \ln(q_{n \rightarrow m}(0)/q_{n \rightarrow m}(1)), \text{ and}$$

$$L_{m \rightarrow n}(x_n) = \ln(r_{m \rightarrow n}(0)/r_{m \rightarrow n}(1))$$

##### 1) Initialization:

Initialize LLR  $Z_{n \rightarrow m}(x_n) = L(x_n|y_n) = (4/N_o) \cdot y_n$ , where  $N_o$  represents the noise variance of the channel.

##### 2) Check node update:

For each check node  $m$  and variable node  $n$ , we compute

$$L_{m \rightarrow n}(x_n) = \left( \prod_{n' \in \mathcal{N}(m) \setminus n} \text{sign}(Z_{n' \rightarrow m}(x_{n'})) \right) \times 2 \tanh^{-1} \left( \prod_{n' \in \mathcal{N}(m) \setminus n} \tanh \frac{|Z_{n' \rightarrow m}(x_{n'})|}{2} \right) \quad (19)$$

##### 3) Variable node update:

$$Z_{n \rightarrow m}(x_n) = L(x_n|y_n) + \sum_{m' \in \mathcal{M}(n) \setminus m} L_{m' \rightarrow n}(x_n) \quad (20)$$

The decision is based on the overall LLR for the  $n^{\text{th}}$  variable node as follows:

$$Z_n(x_n) = L(x_n|y_n) + \sum_{m' \in \mathcal{M}(n)} L_{m' \rightarrow n}(x_n) \quad (21)$$

The value of  $Z_n(x_n)$  is quantized such that  $\hat{x}_n = 1$  if  $Z_n(x_n) < 0$  and  $\hat{x}_n = 0$  if  $Z_n(x_n) > 0$

In the min-sum approximation, the check node update is approximated as follows:

$$\tilde{L}_{m \rightarrow n}(x_n) = \left( \prod_{n' \in \mathcal{N}(m) \setminus n} \text{sign}(Z_{n' \rightarrow m}(x_{n'})) \right) \times \min_{n' \in \mathcal{N}(m) \setminus n} |Z_{n' \rightarrow m}(x_{n'})| \quad (22)$$

##### B. Modified Belief propagation algorithm

In order to modify the decoder to accommodate the combined error distribution, the modification takes place only in the initialization of the LLRs i.e in the  $Z_{n \rightarrow m}(x_n)$ . For the same fixed precision analysis presented in section III, and considering only the effect of a single error in the MSB, The initialization of  $Z_{n \rightarrow m}(x_n)$  is:

$$I_1 = \max \left\{ \begin{array}{l} -\ln(P_e), \frac{1}{N_o} \left( -2\mu(y_n - 1) - \frac{\mu^2}{2} \right), \\ \frac{1}{N_o} \left( 2\mu(y_n - 1) - \frac{\mu^2}{2} \right) \end{array} \right\} \quad (23.a)$$

$$I_0 = \max \left\{ \begin{array}{l} -\ln(P_e), \frac{1}{N_o} \left( -2\mu(y_n + 1) - \frac{\mu^2}{2} \right), \\ \frac{1}{N_o} \left( 2\mu(y_n + 1) - \frac{\mu^2}{2} \right) \end{array} \right\} \quad (23.b)$$

$$Z_{n \rightarrow m}(x_n) = (4/N_o) \cdot y_n + I_1 - I_0 \quad (24)$$

This modification is considered as adding a preprocessing block prior to the decoder as shown in Figure 4. The additional block adjusts the initial LLRs based on (24) while the rest of the decoder remains unchanged. As will be discussed later in the paper, the overhead of the computation is insignificant. This architecture illustrates that it is not necessary to build a new decoder from scratch; but rather add a small block at the decoder front-end to adjust the value of LLRs based on the new distribution.

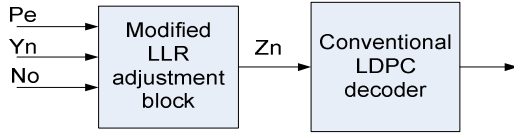


Figure 4 Modified decoder block diagram

### C. Modified TDMP LDPC decoding

The TDMP architecture assumes a specific arrangement for the  $H$  matrix such that it consists of several sub-matrices. For each sub-matrix, each column and each row contains at most one nonzero element. This architecture enables the representation of the LDPC code as a parallel set of constituent codes (sub-codes) connected to each other through a set of interleavers. Hence, Turbo-like decoding is applicable. The details of the original TDMP algorithm are found in [15]. The same modification as in (24) is exactly applied to the front-end of the TDMP decoder to obtain the combined noise resilient TDMP decoder. The LLR vector is initialized as follows:

$$\underline{\gamma} = (4/N_o) * \underline{y} + \underline{I}_1 - \underline{I}_0 \quad (25)$$

where  $\underline{I}_1$  and  $\underline{I}_0$  are computed according to (23.a) and (23.b) respectively.

## V. SIMULATION RESULTS

To illustrate the benefits of the proposed system, we set up two simulation scenarios for two state of the art standards; the first one is a WiMAX double binary turbo decoder with block size  $N=240$  and coding rate  $R=1/2$ . The second one is a DVB-S2 LDPC decoder with block size of 64800 and coding rate of  $1/2$ . The idea can be applied to any other standard with any coding rate and block size. The simulation illustrates the decoders' performance for both AWGN as well as single tap Rayleigh fading channel environment. The simulation assumes a memory error probability  $P_e$  of  $5 \times 10^{-3}$ . As shown in Figure 2 this value of  $P_e$  achieves a considerable savings in the memory power (more than 40%).

### A. Performance in AWGN environment

As shown in Figure 5, the dashed and solid lines represent the performance of the conventional and modified decoders respectively. The system is simulated for (8,5) precision. As shown, the modified decoder enhances the performance for all values of SNR. Furthermore, it does not suffer from the noise flooring effects observed in the conventional decoders. It is shown that the modified decoder is approximately 0.5 dB off from the ideal one (with  $P_e=0$ ) at BER of  $10^{-6}$ , while consuming much less power (a reduction of  $\sim 40\%$  in power). This deviation from the ideal decoder can be smaller for lower values of  $P_e$ . The value of  $P_e$ , which depends on the voltage over-scaling can be controlled based on the available slack in the received SNR. For example, if the received SNR is higher than the minimum required by the decoder to achieve the target BER,  $P_e$  can be increased as long as the target BER is maintained. Additionally, further improvement in performance can be achieved by considering higher order bits, albeit at a cost of decoder complexity. This adaptive power management

is the subject of current research by the authors. The same trends can be observed for the DVB-S2 LDPC decoder as shown in Figure 6.

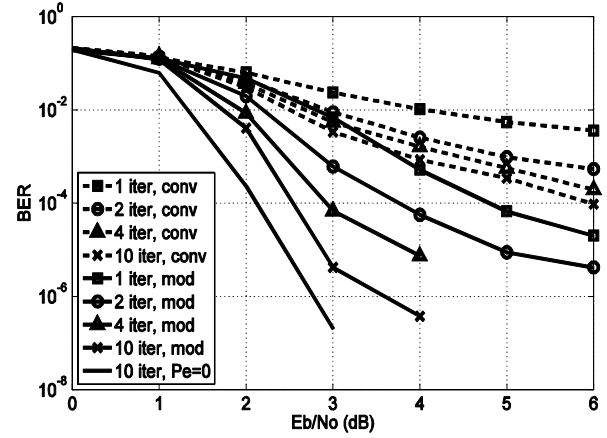


Figure 5 Conventional vs modified decoder performance for WiMAX Turbo decoder for  $N=240$ , and  $R=1/2$

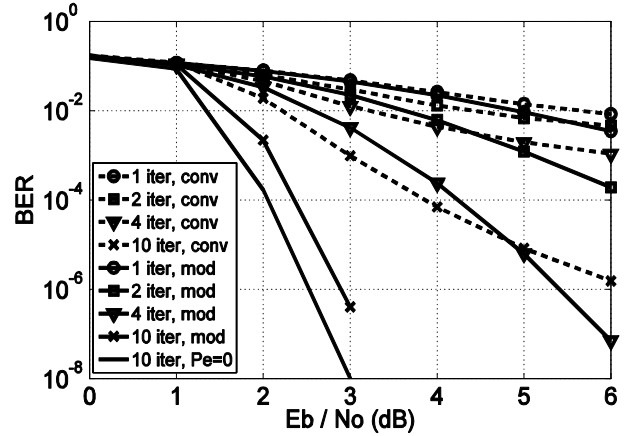


Figure 6 Conventional vs modified decoder performance for DVB-S2 LDPC decoder, rate  $1/2$

### B. Performance in Rayleigh Fading Channel

The decoder performance was tested under a Rayleigh fading channel, assuming perfect channel estimation and MMSE channel equalization. Figure 7 and 8 depict the BER performance for both Turbo and LDPC decoders respectively for multiple iterations. The same trends as in the AWGN case are observed.

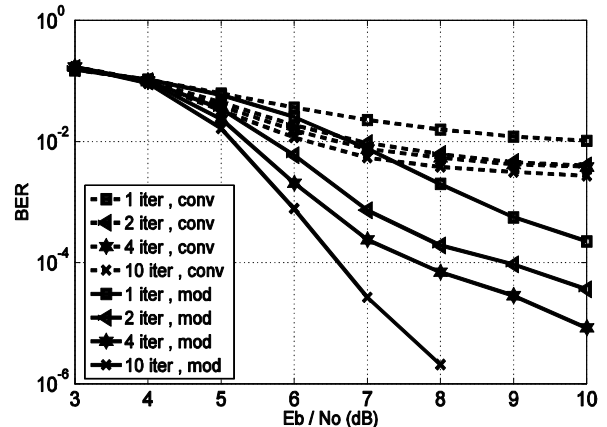


Figure 7 Conventional vs Modified Turbo decoders



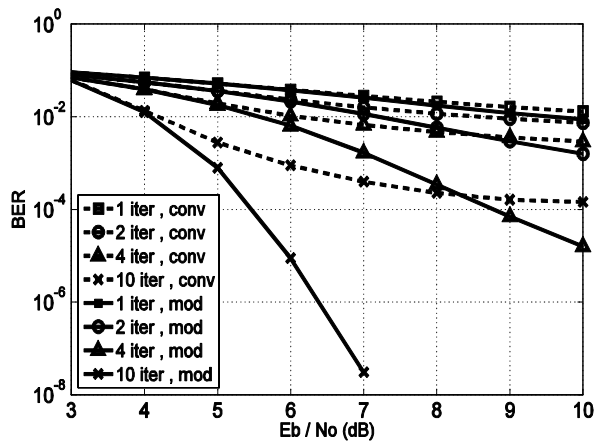


Figure 8 Conventional vs Modified LDPC decoders

## VI. IMPLEMENTATION RESULTS

To estimate the overhead in hardware area and power, the additional modules were designed and synthesized using Synopsys Design Compiler for TSMC CMOS 65nm technology, with a 1V supply and operating at 200 MHz. Table 1 illustrates the overhead as compared to previously published decoders. The area overhead is reported in KGates, where the reference gate is a 2-input NAND gate with a drive X1. To maintain a fair comparison, power is scaled appropriately to a reference technology of 65 nm at 1V, 200 MHz.

Based on the results reported in [17] and [18], the overheads in area are 0.82% and 0.013% for Turbo and LDPC respectively. The overheads in power consumption are 1.1% and 0.016%.

Table 1 Area and power overheads for conv. and modified decoders

	Conv. WiMAX turbo [17]	overhead in WiMAX turbo	Conv. DVB-S2 LDPC [18]	overhead in DVB-S2 LDPC
Area (KG)	154.484	1.266	4215.3	0.482
Est. Power (mW)	25	0.278	476	0.077

## VII. CONCLUSION

This paper presented an analytical framework for managing the combined channel and hardware noise that is induced by applying aggressive memory power management techniques. We presented the design and performance results of modified Turbo and LDPC decoders that are compatible with the new distribution. The modified decoders achieved significant performance improvement with a negligible overhead in area and power consumption.

## VIII. REFERENCES

- [1] Gutnik, V.; Chandrakasan, A.P.; , "Embedded power supply for low-power DSP," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol.5, no.4, pp.425-435, Dec 1997
- [2] International Technology Roadmap for Semiconductors, <http://www.itrs.net/>
- [3] Khajeh, A.; Shih-Yang Cheng; Eltawil, A.M.; Kurdahi, F.J.; "Power Management for Cognitive Radio Platforms," *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, pp.4066-4070, 26-30 Nov. 2007
- [4] Djahromi, Amin Khajeh; Eltawil, Ahmed M.; Kurdahi, Fadi J.; Kanj, Rouwaida; , "Cross Layer Error Exploitation for Aggressive Voltage Scaling," *Quality Electronic Design, 2007. ISQED '07. 8th International Symposium on*, pp.192-197, 26-28 March 2007
- [5] Mukhopadhyay, S.; Mahmoodi, H.; Roy, K.; , "Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol.24, no.12, pp. 1859- 1880, Dec. 2005
- [6] Agarwal, K.; Nassif, S.; , "The Impact of Random Device Variation on SRAM Cell Stability in Sub-90-nm CMOS Technologies," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol.16, no.1, pp.86-97, Jan. 2008
- [7] Khajeh, Amin; Amiri, Kiarash; Khairy, Muhammed S.; Eltawil, Ahmed M.; Kurdahi, Fadi J.; , "A Unified Hardware and Channel Noise Model for Communication Systems," *GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference*, pp.1-5, 6-10 Dec. 2010
- [8] Hussien, A.M.A.; Khairy, M.S.; Khajeh, A.; Amiri, K.; Eltawil, A.M.; Kurdahi, F.J.; , "A combined channel and hardware noise resilient Viterbi decoder," *Signals, Systems and Computers (ASILOMAR), 2010 Conference Record of the Forty Fourth Asilomar Conference on*, pp.395-399, 7-10 Nov. 2010
- [9] Bahl, L.; Cocke, J.; Jelinek, F.; Raviv, J.; , "Optimal decoding of linear codes for minimizing symbol error rate (Corresp.)," *Information Theory, IEEE Transactions on*, vol.20, no.2, pp. 284- 287, Mar 1974
- [10] Hagenauer, J.; Hoehner, P.; , "A Viterbi algorithm with soft-decision outputs and its applications," *Global Telecommunications Conference, 1989, and Exhibition. Communications Technology for the 1990s and Beyond. GLOBECOM '89., IEEE*, pp.1680-1686 vol.3, 27-30 Nov 1989
- [11] Robertson, P.; Villebrun, E.; Hoehner, P.; , "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," *Communications, 1995. ICC '95 Seattle, 'Gateway to Globalization', 1995 IEEE International Conference on*, vol.2, pp.1009-1013 vol.2, 18-22 Jun 1995
- [12] Hagenauer, J.; Offer, E.; Papke, L.; , "Iterative decoding of binary block and convolutional codes," *Information Theory, IEEE Transactions on*, vol.42, no.2, pp.429-445, Mar 1996
- [13] Chen, J.; Dholakia, A.; Eleftheriou, E.; Fossorier, M.P.C.; Hu, X.-Y.; , "Reduced Complexity Decoding of LDPC Codes," *Communications, IEEE Transactions on*, vol.53, no.7, pp. 1232- 1232, July 2005
- [14] Jinghu Chen; Tanner, R.M.; Jones, C.; Yan Li; , "Improved min-sum decoding algorithms for irregular LDPC codes," *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, pp.449-453, 4-9 Sept. 2005
- [15] Mansour, M.M.; Shanbhag, N.R.; , "High-throughput LDPC decoders," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol.11, no.6, pp. 976- 996, Dec. 2003
- [16] Mansour, M.M.; , "A Turbo-Decoding Message-Passing Algorithm for Sparse Parity-Check Matrix Codes," *Signal Processing, IEEE Transactions on*, vol.54, no.11, pp.4376-4392, Nov. 2006
- [17] Chen-Hung Lin; Chun-Yu Chen; An-Yeu Wu; , "Area-Efficient Scalable MAP Processor Design for High-Throughput Multistandard Convolutional Turbo Decoding," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol.19, no.2, pp.305-318, Feb. 2011 doi: 10.1109/TVLSI.2009.2032553
- [18] Urard, P.; Paumier, L.; Heinrich, V.; Raina, N.; Chawla, N.; , "A 360mW 105Mb/s DVB-S2 Compliant Codec based on 64800b LDPC and BCH Codes enabling Satellite-Transmission Portable Devices," *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pp.310-311, 3-7 Feb. 2008