# LDPC with no Error Floors for Errors and Erasures Recovery

David Declercq
Shiva Planjery
Codelucida, Inc.
{declercq,planjery}@codelucida.com

Bane Vasić
Codelucida, Inc. & Department of ECE
University of Arizona
vasic@codelucida.com

Ben Reynwar
Vamsi Krishna Yella
Codelucida, Inc.
{reynwar,vyella1}@codelucida.com

***Abstract*—In this presentation, we propose an LDPC solution for emerging non volatile memories, such as 3D Xpoint or RRAMs. We optimize the quasi-cyclic LDPC so that the erasures coming from die read failures can be easily recovered before decoding. We then introduce a new Erasure-FAID decoder, with the feature of using different update rules for the erased and non-erased bits. Together with using the concept of decoder diversity, we show that our new decoder gives impressive gains compared to the legacy Reed-Solomon solution.**

## I. Introduction

In the growing industry of storage using non-volatile memories, the need for stronger LDPC-based error-correction coding (ECC) solutions is mandatory in order to enable the throughputs and reliability that are required. In particular the new emerging memories (3D Xpoint, MRAMs, etc) pose new challenges to the ECC, compared to the solutions that have been developed in the past few years for 3D NAND Flash memories. First, the codeword lengths used in these memories are smaller (typically around 200-300 bytes) than the lengths used in NAND (typically 1,000  4,000 bytes). At these smaller lengths, the error floor of LDPC decoders occurs at uncorrectable frame error-rates (FER) which are too high to ensure the reliability of the device. In addition, because of the read/write accesses, parts of the codeword could be erased. These erasures come together with binary errors in the non-erased part of the codeword, leading to a channel model which mixes both hard errors and erasures. One of the well knwon coding scheme that is able to correct jointly erasures and errors is based on Reed-Solomon (RS) codes. Using LDPC codes to replace the RS solutions is not trivial, and we propose in this paper an LDPC based solution to this ECC challenge.

First, we put constraints in the LDPC code design so that the erasures can be recovered using simple procedure, by a 1-step recovery (1-SR) algorithm. After this step, the LDPC decoder sees an equivalent binary symmetric channel (BSC) with more errors, but no erasures. We then capitalize on the Finite Alphabet Iterative Decoder (FAID) framework for LDPC decoding [1], in order to lower the error floor. First, new FAID decoders are proposed to specifically deal with the erased bits, such that different decoding rules are used at the erased and non-erased locations. Next, we design and optimize several FAID decoders, that we propose to use collectively, in a sequential diversity fashion, with the objective of lowering the error floor. Our new decoder, called Erasure-FAID (eFAID) shows an impressive improvement in raw bit error rate (RBER) compared to RS, without showing an error floor down to FER=$10^{-12}$. At this FER, our eFAID solution is *250 %* better in RBER than the RS decoder.

## II. Channel Model Assumptions

Because of the specific procedure that is used read the memories, composed of multiple stacked dies, the corrupted codewords experience both binary errors and block-erasures corresponding to an entire die read failure. Let us assume that a codeword of an LDPC code of length $N$ bits is split among $D$ dies, such that $N$ is a multiple of $D$. If any of the $D$ die reads fails, the consequence is that $n = N/D$ bits of the codeword are erased, with no measurement. Depending on the way the LDPC codewords are split among the memory dies, the positions of the erased bits could be either consecutive, forming a block of erasures, or randomly interleaved inside the codeword. As we will explain in section III, it is more favorable for an LDPC decoder to have the erasures appearing in consecutive bits than in random locations. The considered channel model is therefore a binary symmetric error and erasure (BSEE) channel. In this paper, we will assume that exactly 1 memory die is lost.

## III. LDPC Code Design for Erasure Recovery

Let $\mathcal{C}(M_b, N_b, L)$ be a QC-LDPC code. Its parity-check matrix (PCM) is composed of $L \times L$ blocks, in which each block $\mathbf{H}_{i,j}$ is either *(i)* an all-zero $L \times L$ block, or *(ii)* a circulant permutation matrix (CPM). For a $(d_v, d_c)$ regular LDPC code, only $d_c$ out of the $N_b$ blocks in each block-row are CPMs, and only $d_v$ out of the $M_b$ blocks in each block-column are CPMs. In this section, we present an ad-hoc design procedure for building a QC-LDPC code suited for recovering blocks of erasures with a simple procedure, based on the 1-step recovery (1-SR) algorithm [2]. Let us assume that the $D$ erased bits occur in consecutive positions, located on $d$ consecutive block-columns of the PCM, such that $D = d\,L$.

*Property 1:* A sufficient condition to ensure 1-SR recovery in the $d$ consecutive block-columns $\{\mathbf{H}_{:,i+1}, \ldots, \mathbf{H}_{:,i+d}\}$, is that, for every block-column index $j \in \{i+1, \ldots, i+d\}$, there exists at least one block-row $m$ with exactly one CPM at location $\mathbf{H}_{m,j}$ and $(d-1)$ all-zeros blocks elsewhere.

The condition is sufficient since in any block-column $j$, all bits are connected to at least one check-node that has a single erased bit, the one in block column $j$. As a result, the erased bit can be recovered from its non-erased $2^{nd}$-order neighors in only $k = 1$ step. An example of QC-LDPC matrix with $L = 64$ following property 1 $d = 3$ is shown in equation (1).

$$\mathbf{H} = \begin{bmatrix} 19 & -1 & -1 & 20 & 62 & 1 & -1 & -1 & 33 & 6 & 60 & 58 \\ -1 & -1 & 51 & 22 & 41 & 43 & 59 & -1 & -1 & 14 & 49 & 62 \\ -1 & 29 & -1 & 50 & 61 & 51 & -1 & 60 & -1 & 32 & 28 & 63 \\ 29 & 26 & 40 & -1 & -1 & 59 & 7 & 52 & 9 & -1 & -1 & 6 \\ 1 & 17 & 1 & 9 & -1 & -1 & 62 & 1 & 57 & 50 & -1 & -1 \\ 15 & 51 & 8 & -1 & 47 & -1 & 39 & 31 & 22 & -1 & 49 & -1 \end{bmatrix} \quad (1)$$

The 1-SR operation is performed prior to the actual LDPC decoder, and its main drawback is that it introduces errors in the erased positions that are propagated from the channel errors occuring in the non-erased positions. The initial errors from the channel will be refered to as *channel errors*, while the errors at the erased location, created by the 1-SR, will be refered to as *generated errors*. As a result, after the 1-SR, the LDPC decoder needs to correct an equivalent BSC channel with a higher RBER than the original one: $\text{RBER}_2 > \text{RBER}_1$. This loss in RBER is associated with a loss in channel capacity which cannot be recovered by any ECC scheme.

## IV. FAID Decoding for Channel with Erasures (eFAID)

Assume that $\mathbf{x}$ is a codeword of a length $N$-bits QC-LDPC code. The noisy codeword obtained after reading the memory and 1-SR recovery is denoted $\mathbf{y} = \mathbf{x} \oplus \mathbf{e}$, where $\oplus$ denotes the bitwise XOR operation, and $\mathbf{e}$ represents the error pattern. In FAID decoders for regular $d_v = 4$ LDPC codes, the variable node update rule is implemented using a look-up table (LUT) function, that we denote $\Phi_v(m_1, m_2, m_3, y)$. In this update rule, $m_i$ represent incoming messages to the variable node belonging to a finite alphabet $\mathcal{A}$.

In this paper, we propose to use a specific FAID update rule for the erased variable nodes. Indeed, the generated errors are induced by parity-check computations involving channel errors, and therefore, it is likely that the combination of channel errors and generated errors are in the support of TSs. Our solution to avoid the attraction of these artificially created harmful error patterns is to use a specific FAID update rule, so that the generated errors contribute less to the iterative decoding process. The specific update rule is still a FAID type rule, but for which there is no channel information. We denote such rule $\Phi_\epsilon(m_1, m_2, m_3)$. In the decoding, $\Phi_\epsilon(m_1, m_2, m_3)$ will be used for the erased bits, and $\Phi_v(m_1, m_2, m_3, y)$ for the non-erased bits. For the design of $\Phi_\epsilon(m_1, m_2, m_3)$, we followed the same principle as for the design of the classical FAIDs, by selecting the rule which has the best density evolution (DE) threshold and best metrics on isolated TSs, under the symmetry constraint [1].

## V. eFAID Diversity for Improved ECC performance

The key unique feature of FAID is that many good update rules $\Phi_v$ exist, and that all of them can be considered as a potential update rule in an iterative decoder, with the same hardware cost [1]. Using the FAID framework opens the possibility to use not just one, but a multiplicity of FAID rules, with the objective of improving the ECC performance. We call this strategy *FAID diversity*. Let us assume that we have at hand a collection of $\omega$ eFAID decoders:

$$\left\{\mathcal{F}^{(k)}\right\}_{k=1,\ldots,\omega} = \left\{\Phi_\epsilon(m_1, m_2, m_3)^{(k)}, \Phi_v(m_1, m_2, m_3, y)^{(k)}\right\}$$

that we use collectively to correct the errors in $\mathbf{y}$. We start the decoding with $\mathcal{F}^{(1)}$. If $\mathcal{F}^{(1)}$ fails to correct the error event $\mathbf{e}$ in less than $N_{it}$ iterations, then $\mathbf{y}$ is used as the input of the next decoder $\mathcal{F}^{(2)}$ and a new tentative decoding is triggered for another $N_{it}$ iterations. The process is repeated until decoding is succesful, or all eFAIDs in the diversity set have been exhausted.

The eFAID diversity requires a very small hardware overhead, but introduces extra decoding latency each time the first decoder $\mathcal{F}^{(1)}$ does not correct the errors. However, when $\mathcal{F}^{(1)}$ is a powerful enough decoder the rest of the decoders are not triggered very often. This means that the average decoding latency does not increase significantly, and only the worst case latency is impacted. The performance gain that can be obtained using eFAID diversity is tighly linked to the choice of the set of decoding rules $\mathcal{F}^{(k)}$. First, $\mathcal{F}^{(1)}$ is optimized using DE techniques to get the best possible performance in the waterfall region. Then, the next eFAIDs $\mathcal{F}^{(2)}$ to $\mathcal{F}^{(\omega)}$ must be chosen such as to collectively correct a maximum number of TSs Errors. In [3], we developed a expansion-contraction method to characterize efficiently the harmfulness of the TSs in a fixed LDPC code. We use this method to optimize the eFAID diversity set.

## VI. Performance of eFAID on the MSEE channel

We illustrate our eFAID performance on a QC-LDPC specifically designed with 1-SR constraints. The considered codeword length is small, equal to N=176 Bytes, and the code rate is R=0.77. The QC-LDPC is column regular, with $d_v = 4$. On figure 1 we show the performance of eFAID diversity, with $\omega \in \{1, 2, 8\}$ decoders.

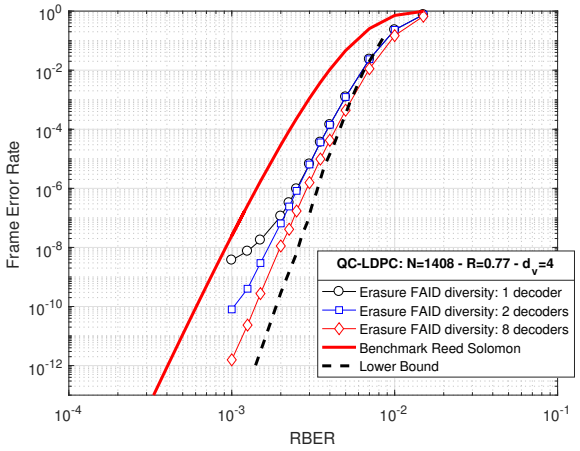


Fig. 1. Performance of Erasure-FAID diversity on the BSEE channel for a $R = 0.77$, $N = 176$ Bytes QC-LDPC code.

The Benchmark RS solution in GF($2^8$) with same rate and length as the LDPC is also shown. Our eFAID diversity decoder shows extremely good performance. With only $\omega = 2$ decoders, we are able to lower the error floor by 2 orders of magnitude. With a diversity of $\omega = 8$ decoders, the eFAID shows no error floor at all down to FER$= 1e-12$. We have an impressive gain in RBER of 250 % at FER$= 1e-12$ and more than 4 orders of magnitude improvement in terms of FER. We finally consider comparison with FAID decoding on an equivalent BSC channel after the 1-SR step. The FER obtained by FAID diversity is measured at $\text{RBER}_2$, which we shift to plot against $\text{RBER}_1$. This plot represents a lower bound as it assumes that the errors are uniformy distributed over the codeword, which is not the case after the 1-SR. Our eFAID performance is close to the lower bound, which is a good indication that our solution is very robust to the presence of erasures.

## VII. Conclusion

In this paper, we presented an LDPC-based ECC solution for emerging memories, in which a mixture of errors and erasures have to be corrected. Our approach is based on 1-SR recovery of the erased bits, followed by a novel erasure-FAID decoder, thoroughly optimized to cancel the error floor. Our eFAID decoder yields an improvement of 250 % in RBER compared to the RS legacy solution.


## Acknowledgments

This work was supported by the National Science Foundation under SBIR Phase II Grant 1534760.


## References


[1] S. K. Planjery, D. Declercq, L. Danjean, and B. Vasić, "Finite alphabet iterative decoders, Part I: Decoding beyond belief propagation on the binary symmetric channel," *IEEE Trans. Commun.*, vol. 61, no. 10, pp. 4033–4045, Nov. 2013.

[2] J. Ha, J. Kim, D. Klinc, and S. McLaughlin, "Rate-compatible punctured low-density parity-check codes with short block lengths," *IEEE Trans. Inform. Theory*, vol. 52, no. 2, pp. 728–738, Feb. 2006.

[3] N. Raveendran, D. Declercq, and B. Vasic, "A sub-graph expansion-contraction method for error floor computation," *submitted to IEEE Trans. Commun.*, Nov. 2019.