EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

A concatenation scheme of LDPC codes and source codes for flash memories

Qin Huang[1], Song Pan[2], Mu Zhang[1] and Zulin Wang[1*]

Abstract

Recently, low-density parity-check (LDPC) codes have been applied in flash memories to correct errors. However, as verified in this article, their performance degrades rapidly as the number of stuck cells increases. Thus, this paper presents a concatenation reliability scheme of LDPC codes and source codes, which aims to improve the performance of LDPC codes for flash memories with stuck cells. In this scheme, the locations of stuck cells is recorded by source codes in the write process such that erasures rather than wrong log-likelihood ratios on these cells are given in the read process. Then, LDPC codes correct these erasures and soft errors caused by cell-to-cell interferences. The analyses of channel capacity and compression rates of source codes with side information show that the memory cost of the proposed scheme is moderately low. Simulation results verify that the proposed scheme outperforms the traditional scheme with only LDPC codes.

Introduction

In flash memories, as *programming and erasure* (P/E) cycles increase, some cells may be stuck at a voltage level not related to the data being written in, resulting in stuck cells or defective memory cells [1,2]. At present, a class of hard-decision decoding error correction codes (ECC), Bose Chaudhuri Hocquenghem (BCH) codes, are employed to correct such stuck errors caused by stuck cells as well as soft errors [3,4], induced by cell-to-cell interferences. Moreover, if the number of stuck cells in one page exceeds a certain limit, the whole page will be discarded and masked.

As scaling is becoming more and more aggressive, reliability is also becoming one of the most important challenges in *multi-level cell* (MLC) flash memories now and future. Therefore, a class of more powerful soft-decision decoding ECCs, *low-density parity-check* (LDPC) codes, [5-7], are introduced to improve the reliability of flash memories [3,4,8]. Different from hard-decision decoders, stuck errors are very harmful to soft-decision decoders, since their log-likelihood ratios (LLRs) could have a large magnitude such that soft-decision decoders may mistakenly consider them correct with very high reliability. From our simulation results, even a small number of these stuck errors may trap soft-decision message-passing decoders of LDPC codes. It causes severe performance degradation, such as worse waterfall thresholds and slower slope of bit error rate (BER) curves.

In this article, we present a concatenation reliability scheme of LDPC codes and source codes to alleviate the performance degradation. The key idea is to erase the wrong LLRs of stuck cells for message-passing decoders according to their locations provided by source codes. The scheme consists of two parts, namely the write and the read parts. When the controller is writing data into flash memories, it is also designed to detect stuck cells. The locations of these stuck cells are compressed by source codes with side information to efficiently reduce the memory cost. In the read part, the compressed location information is recovered and the wrong LLRs related to these locations are erased, i.e., set their LLRs to zero. Then, LDPC decoders correct both these erasures and soft errors.

We also compare the proposed concatenation scheme with the traditional one from the angle of error performance and memory cost. On one hand, it can significantly outperforms the traditional one, since it removes the wrong LLRs for LDPC decoders. On the other hand, capacity and compression rate analyses indicate that it moderately requires a small percentage of memory cost.

*Correspondence: qhuang.smash@gmail.com
[1]School of Electronic and Information Engineering, Beihang University, Beijing 100191, China
Full list of author information is available at the end of the article

Simulation results show that it achieves significantly better error performance than the traditional one, even if the traditional one uses LDPC codes with lower rate. Because the proposed scheme can tolerate more stuck errors, it is also able to prolong the life of flash memories, besides reliability.

The rest of the article consists of four sections. In “Introduction” section, we introduce some backgrounds on channel models of flash memories and LDPC codes. The proposed concatenation scheme is presented in “Concatenation scheme of LDPC codes and source codes” section. Analyses and simulation are given in “Analyses and simulation” section. This article is concluded in the last section.

Background

Channel models for different types errors

In flash memories, each cell uses 2^s voltage levels $L_1, L_2, \ldots, L_{2^s}$ to represent s stored bits. If $s = 1$, they are single-level cell flash memories; if $s > 1$, they are 2^s MLC flash memories. The writing processing, i.e., voltage level programming, is realized by incremental step pulse program with an incremental program step voltage ΔV_{pp} [4]. Under the ideal condition, the distributions of voltage levels except the first one are uniform distributions with the width of ΔV_{pp}. The distribution of the first level is a wide Gaussian-like distribution. Since these distributions do not overlap each other, no errors occur. However, because of inevitable cell-to-cell interferences in real systems, all the voltage distributions change to Gaussian-like distributions and may overlap (Figure 1). Thus, soft errors may occur. Cell-to-cell interferences [3,4] can be modeled as an additive white Gaussian noise (AWGN) channel with pulse-amplitude modulation (PAM). In other words, the voltage level L_i has the output

$$R_j = L_i + N_i \tag{1}$$

for the jth flash cell through the AWGN channel, where N_i is the additive Gaussian noise with zero mean and variance σ_i^2.

In addition, since P/E cycles gradually degrade flash memory medium, some cells may permanently be stuck with one voltage level [1,2], named *stuck errors* or *stuck-at-faults*. In this article, we mainly consider cells stuck at the highest level which is a common type of stuck errors after many P/E cycles. Suppose that there are u% stuck cells in the flash memory. Such errors can be described by a stuck channel with a transition probability $p = 0.01u$ as shown in Figure 2. Each level L_i has the same probability p to be stuck at the highest level L_{2^s}, noted $\ominus_p\{L_i\}$.

LDPC codes

A binary regular *(n, k)* LDPC code [6] is defined as the null space of an *m* × *n sparse parity-check matrix* **H**, $m \geq n-k$, over GF(2) with the following structural properties: (1) each row has constant weight ρ; (2) each column has constant weight γ; and (3) no two rows (or two columns) can have more than one position where they both have 1-components. **H** is said to be (γ, ρ)-regular and the code given by the null space of **H** is called a (γ, ρ)-regular LDPC code. Property (3) is referred to as the *row-column (RC) constraint*. Most LDPC codes investigated in flash memories are regular due to high code rate and low error-floor requirements.

Consider a binary message vector **m** of length k and an *(n, k)* binary LDPC code. Its corresponding codeword **c** of length n is $\mathbf{m} \cdot \mathbf{G}$ and satisfies $\mathbf{c} \cdot \mathbf{H}^{\mathrm{T}} = \mathbf{0}$, where **G** is the generator matrix of the LDPC code.

Soft-decision decoding algorithms for LDPC codes are mostly *iterative* in nature and devised based on *message-passing*. The most well known ones [6] are the *sum-product algorithm* and its simplified version, the *min-sum* (MS) algorithm [9].

Concatenation scheme of LDPC codes and source codes

Channel model for both soft errors and stuck errors

Two types of errors in flash memories are introduced in “Channel models for different types errors” section. Soft errors caused by cell-to-cell interferences are described by an AWGN channel. Stuck errors caused by stuck cells are described by a stuck sub-channel. In existing researches, the two types of errors from the two channels are separately considered for correction [3,4,10]. However, both of them exist in real systems at the same time, especially at middle-late stage of memory life. Thus, we consider a channel model consisting of two sub-channels, the AWGN sub-channel for soft errors and the stuck channel for stuck errors. Furthermore, it can be time-varying according to P/E cycles thus more realistic. As a result, the reliability scheme based on the proposed channel model will be more efficient in applications and make it possible to extend the life of memory cells.

The block diagram of the channel model is shown in Figure 3. Consider 2^s level flash memories and a binary data vector **c** of length $n = sn'$ as the input. By Gray mapping, it is stored into a voltage level vector **V** of length of n'. After the first stuck sub-channel, the vector of **V** becomes

$$\mathbf{S} = \ominus_p\{\mathbf{V}\}, \tag{2}$$

where $S_j = \ominus_p\{V_j\}$ for the jth cell, $j = 0, 1, \ldots, n' - 1$. If S_j is not stuck, it will pass the second AWGN sub-channel and the voltage level S_j will change to $R_j = L_i + N_i$; otherwise, $R_j = L_i$.

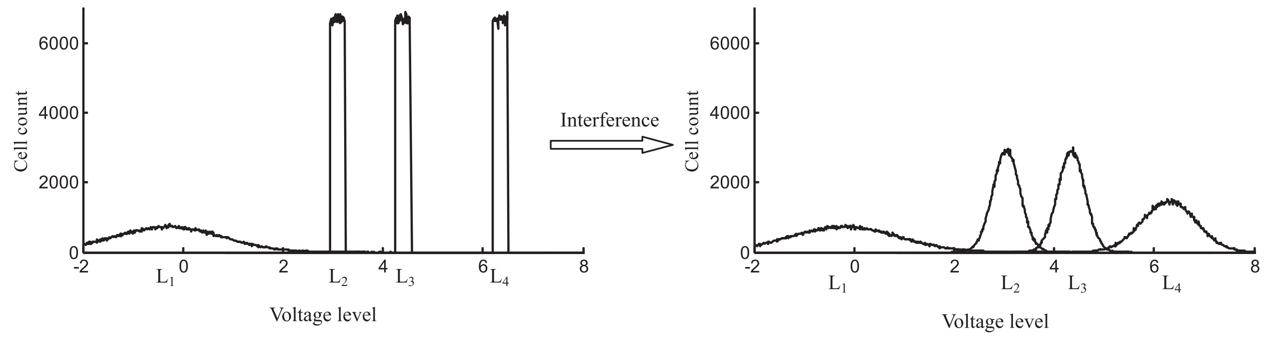

Figure 1 Voltage distributions with cell-to-cell interferences. Each level counts 22k cells.

The proposed concatenation scheme

Until now only ECCs, e.g., BCH codes or LDPC codes, are used to correct errors in practice, since the AWGN sub-channel dominants the performance in the early stage (i.e., few P/E cycles) of flash memories. However, as the number of P/E cycles increases, the stuck sub-channel will bring significant effects, especially harmful to probability decoders like message-passing decoders of LDPC codes, and will finally kill the memories.

If we only consider the reading process in ECC schemes, the controller can only receive voltage levels **R** which may be mixed with stuck errors. From these voltage levels, LLRs are calculated based on the AWGN sub-channel with PAM modulation. Since these LLRs are mixed with wrong magnitudes and signs, even the maximum likelihood codeword based on these LLRs may not be correct.

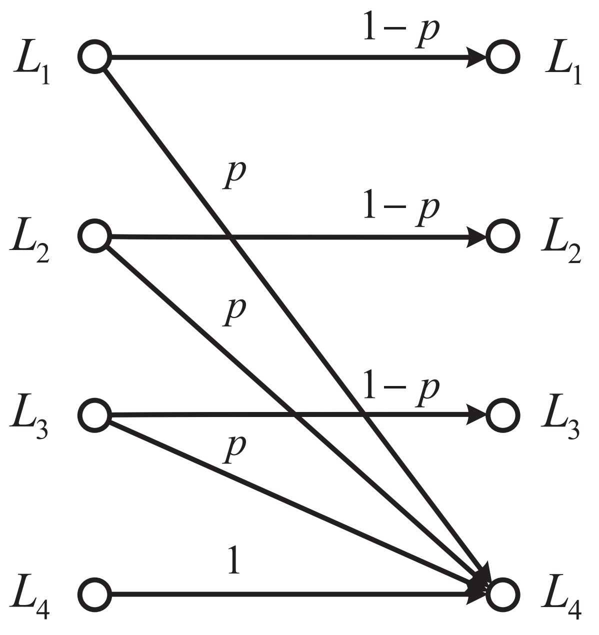

Figure 2 Model of stuck channel in flash memories.

In "Analyses and simulation" section, we will show that a small number of stuck cells will cause significant performance degradation.

However, if we also consider both the writing process, the decoder will be able to distinguish normal cells and stuck cells by the information provided by the controller.

As described in [4], MLC flash memories are written by a program and verify scheme. If some cells are stuck, their voltages keep unchanged such that they can be detected by a well-designed controller. As shown in Figure 4, we propose that the controller detects stuck cells and then records their locations into a vector **l** of length n', in the writing process, where $l_j = 1$ if jth cell is stuck; otherwise $l_j = 0$. To reduce the memory cost, the location vector **l** is compressed into a vector **z**. In the next section, we will present the compression approach and its memory cost.

In the reading process, as shown in Figure 5, the controller receives voltage levels **R** from the current page and the compressed vector from the compressed location page. The vector **r** of LLRs [11] is calculated based on the AWGN sub-channel with PAM modulation from **R**. After the location information **l** is recovered from the compressed vector **z** and the side information **v** [12] (see "Memory cost of source codes" section), the LLRs on the stuck locations are set to zero, i.e., $r_{sj+t} = 0$ if $l_t = 0$, where $t = 0, 1, \ldots, s-1$. Then, the LLRs are sent to LDPC decoders to output the decoded data.

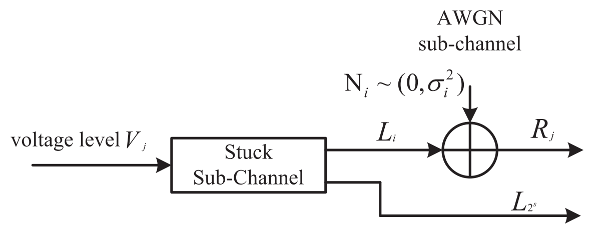

Figure 3 Channel model composed of AWGN sub-channel and stuck sub-channel.

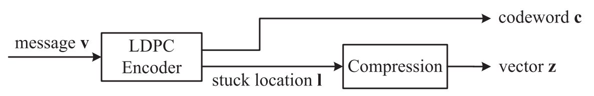

Figure 4 The write part of the reliability concatenation scheme of LDPC codes and sources codes.

Analyses and simulation

In this section, we first would like to analyze the capacity of the flash channel models. Then, based on the analyses of channel capacity and compression rates of source codes, the memory cost of source codes and the overall code rate of the proposed scheme are presented. Last, simulation results verify that the proposed scheme achieves better performance than the traditional one, even if the traditional one deploys lower rate LDPC codes.

Capacity analysis

It is interesting to know the capacity C of the proposed channel model. First, it is clear that the channel consisting of two sub-channels is a Markov Chain. Thus, its capacity is upper bounded by the capacity of the AWGN sub-channel C_a and the capacity of the stuck sub-channel C_s [13],

$$C \leq \min\{C_a, C_s\}. \quad (3)$$

Consider 4-level flash memories, i.e., $2^s = 4$. The capacity of the stuck sub-channel C_s is

$$C_s = (1-p)\log_2(p^{-p} + 3(1-p)) + p\log_2 \frac{p^{(1-p)} + 3p(1-p)}{p^{(1-p)} + (1-p)p^{-p}}. \quad (4)$$

The derivation of C_s is given in the Appendix.

By the proposed scheme, stuck errors are transferred to erasures. Thus, the channel can be viewed as the concatenation of the AWGN sub-channel and the 2^s-ary erasure sub-channel (Figure 6). It is well known that the capacity of the erasure sub-channel C_e with erasure probability p is $s(1-p)$ [13]. If $s = 2$, then the capacity of the erasure sub-channel is

$$C_e = 2(1-p). \quad (5)$$

In the next section, the capacity of the stuck sub-channel (4) and the erasure sub-channel (5) is used to derive the memory cost of source codes.

Memory cost of source codes

In this article, we propose to deploy lossless source codes to compressed the location vector **l** into the compressed vector **z**. Moreover, as shown in Figure 5, side information [12,14] from voltage levels is used to further reduce the memory cost of compression.

It is well known in information theory that compression rates of lossless source codes can efficiently approach [15] the entropy of the original data. For the stuck location vector, its entropy is

$$n'H(\mathbf{l}) = n'H(l_i) = n'(-p\log_2 p - (1-p)\log_2 p). \quad (6)$$

In addition, according to Slepian-Wolf theorem [12], the theoretical limit of lossless compression of vector **l** is improved to the conditional entropy $H(\mathbf{l}|\mathbf{v})$, with the help of *side information* **v**, where **l** and **v** are correlated with $Pr(l_i \neq v_i) < 0.5$. Since stuck cells stay in the highest level as shown in Figure 2, the voltage level vector **R** from data pages provides side information to improve the efficiency of source codes. In other words, if the $\hat{R}_j \neq L_{2^s}$, the jth cell is not stuck, i.e., $l_j = 0$, where $\hat{R}_j$ is the hard-decision of R_j [11]. Let us describe the side information by the vector $\mathbf{v} = [v_0, v_1, \ldots, v_{n'-1}]$, where $v_i = 0$ if $\hat{R}_j \neq L_{2^s}$; else $v_i = 1$. Then, the cell is stuck, i.e., $l_j = 1$ only if $v_i = 1$. Suppose that the data bit is discrete uniform distribution in '0' and '1'. Recall that the transition probability is p. The probability of stuck errors is $p(2^s-1)/2^s$, since there is a probability $1/2^s$ that stuck cells are coincidentally written into L_{2^s} such that no errors occur in these cells. Then, the probability that $v_i = 1$ is $p + (1-p)/2^s = \frac{1+p(2^s-1)}{2^s}$ and the conditional probabilities are $Pr(l_i = 0|v_i = 0) = 1$, $Pr(l_i = 1|v_i = 0) = 0$, $Pr(l_i = 0|v_i = 1) = \frac{1}{1+p(2^s-1)}$, and $Pr(l_i = 1|v_i = 1) = \frac{p(2^s-1)}{1+p(2^s-1)}$. Thus, the conditional

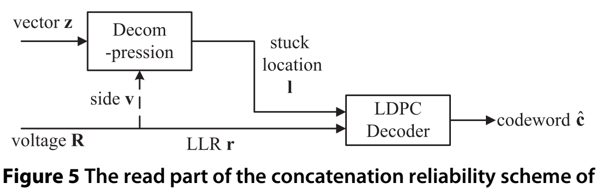

Figure 5 The read part of the concatenation reliability scheme of LDPC codes and source codes.

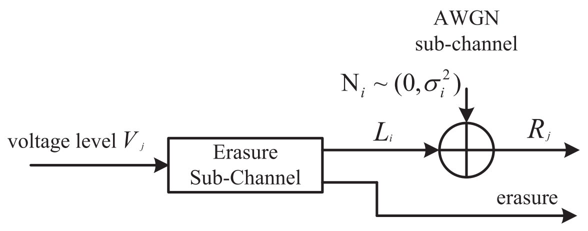

Figure 6 Channel model with source codes with side information.

entropy $H(\mathbf{l}|\mathbf{v})$

$$\begin{aligned} lllH(\mathbf{l}|\mathbf{v}) &= n'H(l_i|v_i) \\ &= n'\left\{\frac{1+p(2^s-1)}{2^s}\left[Pr(l_i=1|v_i=1)\log_2\frac{1}{Pr(l_i=1|v_i=1)}\right.\right. \\ &\qquad \left.\left.+Pr(l_i=0|v_i=1)\log_2\frac{1}{Pr(l_i=0|v_i=1)}\right]\right\}, \\ &= -\frac{n'(1+p(2^s-1))}{2^s}\left[\frac{p(2^s-1)}{1+p(2^s-1)}\log_2\frac{p(2^s-1)}{1+p(2^s-1)}\right. \\ &\qquad \left.+\frac{1}{1+p(2^s-1)}\log_2\frac{1}{1+p(2^s-1)}\right]. \end{aligned} \tag{7}$$

Thus, the compression rate R_{sc} of source codes is about

$$R_{sc} \geq \frac{H(\mathbf{l}|\mathbf{v})}{sn'}. \tag{8}$$

Suppose that the code rate of the LDPC code is R_{cc}, then the overall code rate of the concatenation scheme is

$$R_{or} = R_{cc}/(1+R_{sc}). \tag{9}$$

The stuck location changes much slower, since stuck cells occur only after many P/E cycles on data pages. In other words, P/E cycles happen much less often in the compressed location pages than data pages. Thus, we could consider that compressed location pages contain few errors before the flash memories die. A high rate and short ECC is strong enough to guarantee the reliability, which has a small impact on the overall code rate. From brevity, we does not involve the error correcting issue of compressed location pages in this article.

It is clear that the proposed scheme takes about $H(\mathbf{l}|\mathbf{v})$ bits to record the stuck locations. On the other hand, the stuck sub-channel is replaced by the erasure sub-channel, which has larger channel capacity. Recall the capacity of the erasure sub-channel C_s (4) and the capacity of the erasure sub-channel (5). We define the memory cost of source codes as

$$C_{mc} = 1 - \frac{C_e/(1+R_{sc})}{C_s} = 1 - \frac{C_e}{C_s(1+R_{sc})}. \tag{10}$$

In Figure 7, we show that the memory cost is moderately low and grows almost linearly with the transition probability p when p is less than 0.02. It is also worth to mention that the combination of the BEC sub-channel and the AWGN channel is inherent better the combination of the stuck sub-channel and the AWGN channel for the soft-decision decoders, since the harmful wrong LLRs have been removed with the help of source codes. As we will show in the next section, the proposed scheme outperforms the traditional ECC only scheme.

Simulation results

In this section, we will show the performance degradation of LDPC codes caused by stuck cells. Then, the proposed scheme is compared with the traditional scheme with only one ECC.

In our simulation, we consider 2-bit (4-level) flash memories, i.e., $s = 2$. Gray mapping is used to map 2 bits of data to a voltage level L_i, where L_1, L_2, L_3, and L_4 represent '00', '01', '11', and '10', respectively. Consider cell-to-cell interferences as the AWGN sub-channel, their means are all zero and their variances σ_i^2 are $4\sigma^2$, σ^2, σ^2, and $2\sigma^2$, where σ^2 is measured by

$$V_{\sigma^2} \triangleq 10\log_{10}\frac{(L_4-L_1)^2}{\sigma^2}.$$

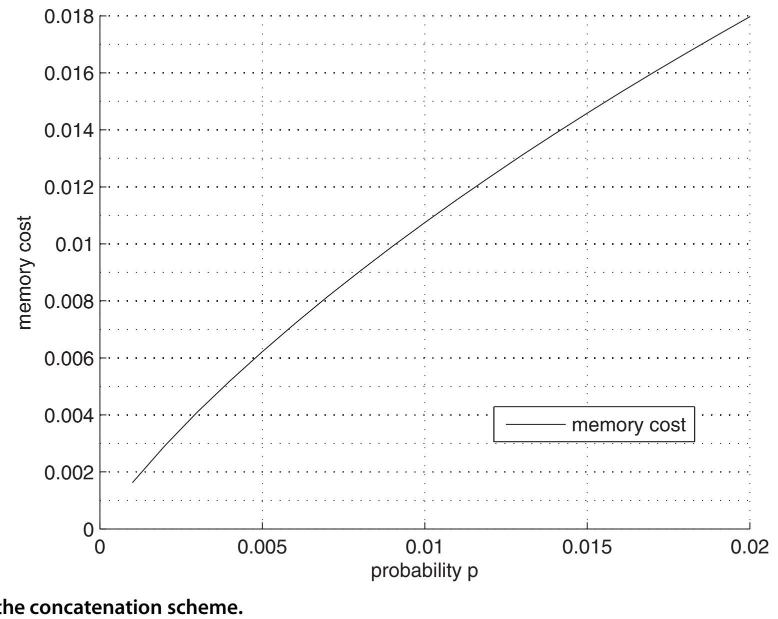

Figure 7 Memory cost of the concatenation scheme.

Clearly, the larger V_{σ^2}, the less cell-to-cell interferences. A similar definition can be found in [16]. Moreover, two percentages, 0.5 and 1%, of stuck cells are investigated. From (8), the compression rates of source codes for 0.5 and 1% are about 0.0161 and 0.0245, respectively.

In the proposed scheme, a (16000,15000) LDPC code with constant column weight 4 and code rate 15/16 is used. The source code costs about $0.0141 \times 16000 \approx 226$ bits to record the location vector $\mathbf{l}$ if $p = 0.005$ and $0.0245 \times 16000 \approx 392$ bits if $p = 0.01$. Thus, their overall rates (9) are 0.924 if $p = 0.05$ and 0.915 if $p = 0.01$. To be fair, we add the same overhead into the traditional scheme such that a (16226,15000) LDPC code with constant column weight 4 and a (16392,15000) LDPC code with constant column weight 4 are used in the traditional scheme for $p = 0.005$ and $p = 0.01$, respectively. All the LDPC codes are decoded by the MS algorithm [9] with 50 iterations.

The simulation results are shown in Figure 8. First, we show the severe performance degradation caused by a small number of stuck errors. Consider the traditional scheme, if there are 0.5% of stuck cells in the flash memory, the performance degradation between no stuck cells and 0.5% stuck cells is about 1.8 dB at BER 10^{-6}. Moreover, the slope of the BER curve with stuck cells obviously reduces. If the percentage of stuck cells increases to 1%, the performance degradation is 2.5 dB at BER 10^{-3}. In addition, the BER curve with stuck cells becomes flat. Second, we compare the traditional scheme with the concatenation scheme. If there are 0.5% stuck cells, the concatenation scheme provides about 1.3 dB at BER 10^{-6} over the traditional scheme with the (16226,15000) LPDC code. Moreover, the proposed scheme outperforms the traditional scheme with the (16500,15000) LPDC code 0.4 dB at BER 10^{-6}, whose code rate 0.909 is lower than 0.924. If the percentage increases to 1%, the performance gain is much larger, about 1.7 dB at BER 10^{-3}. It is worth mentioning that though the concatenation scheme suffers from stuck cells, the slopes of its BER curves are as well as LDPC codes without stuck errors.

Conclusion

Cell-to-cell interferences and stuck cells are the two most important reliability issues in flash memories. In this article, we present a concatenation reliability scheme of LDPC codes and source codes with side information to enhance the reliability. The proposed scheme erases stuck cells according to their recorded stuck locations such that the performance of soft-decision decoding algorithms for LDPC codes is significantly improved. Simulation results indicate that the proposed scheme greatly outperforms the traditional scheme and does not have reduction in the slopes of BER curves. Moreover, the memory cost of recording stuck locations is moderately low and grows almost linearly of the transition probability p.

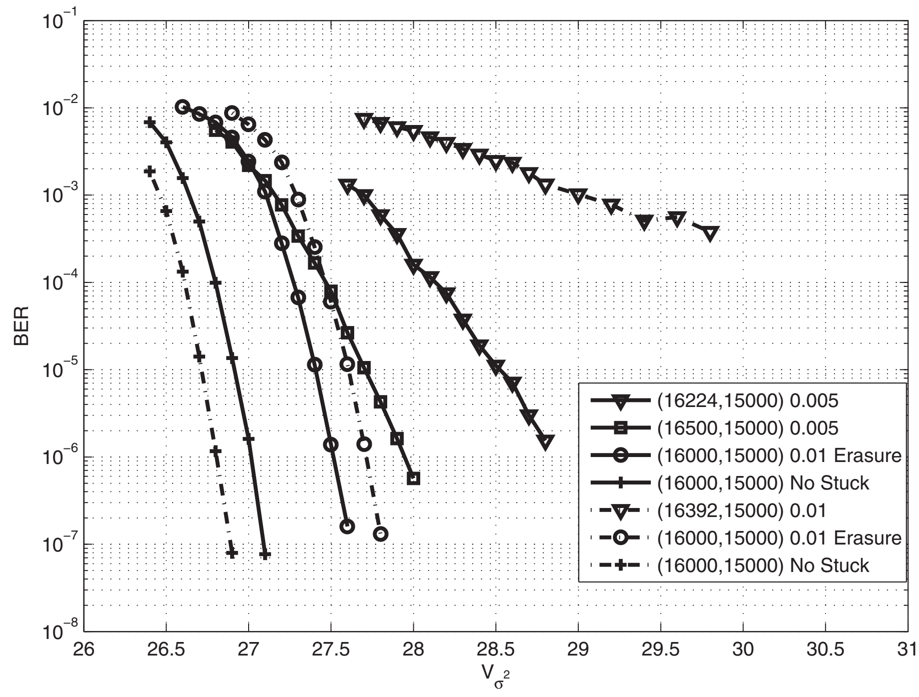

Figure 8 BER of different schemes with different percentages of stuck cells.

Last we would like to give some remarks on the concatenation scheme:

(1) Due to the asymmetric property of the stuck subchannel, Figure 2, it is not optimal to set LLRs to zero for erased cells. Thus, we set the LLR to $\mp\alpha \log 2$ instead of zero in the simulation, where α is a scaling factor. However, no obvious coding gain is observed, since the amount of stuck cells is relatively small and little bias on them will not have obvious impacts on LDPC decoders. Thus, this article sets zero for simplicity.

(2) In this article, side information from the asymmetric property, Figure 2, is used to reduce the memory cost of source codes. Other side informations are also amicable to improve the source codes' efficiency, as long as they are correlated to the stuck location vector $\mathbf{l}$ with $Pr(l_i \neq v_i) < 0.5$ [14].

Appendix

Capacity of 4-level stuck sub-channel

The channel capacity of the stuck sub-channel is defined as follows,

$$C_s = \max_{p(V_i)} \left[I(\mathbf{V};\mathbf{S}) \right].$$

By constructing a new function with the Lagrange multiplier λ,

$$F = I(\mathbf{V};\mathbf{S}) - \lambda \sum_{\mathbf{V}} p(V_i),$$

we obtain a function system

$$\begin{cases} \frac{\partial F}{\partial p(V_i)} = \frac{\partial \left[I(\mathbf{V};\mathbf{S}) - \lambda \sum_{\mathbf{V}} p(V_i) \right]}{\partial p(V_i)} = 0 \\ \sum_{\mathbf{V}} p(V_i) = 1 \end{cases}.$$

As is known, the mutual information of the channel is

$$I(\mathbf{V};\mathbf{S}) = \sum_{i=1}^{s} \sum_{j=1}^{s} p(V_i) p(S_j|V_i) \log_2 \frac{p(S_j|V_i)}{p(S_j)}, \qquad (11)$$

where $p(V_i)$'s represent the probability contribution of the input while the probability of the output is given by

$$p(S_j) = \sum_{i=1}^{s} p(V_i) p(S_j|V_i).$$

The functions are transformed as below,

$$\begin{cases} \sum_{j=1}^{s} p(S_j|V_i) \log_2 \frac{p(S_j|V_i)}{p(S_j)} = \log_2 e + \lambda \\ \sum_{\mathbf{V}} p(V_i) = 1 \end{cases},$$

where e represents the Euler number. For the 4-level stuck-sub-channel,

$$\begin{cases} p(S_1|V_1) \log_2 \frac{p(S_1|V_1)}{p(S_1)} + p(S_4|V_1) \log_2 \frac{p(S_4|V_1)}{p(S_4)} = \log_2 e + \lambda, \\ p(S_2|V_2) \log_2 \frac{p(S_2|V_2)}{p(S_2)} + p(S_4|V_2) \log_2 \frac{p(S_4|V_2)}{p(S_4)} = \log_2 e + \lambda, \\ p(S_3|V_3) \log_2 \frac{p(S_3|V_3)}{p(S_3)} + p(S_4|V_3) \log_2 \frac{p(S_4|V_3)}{p(S_4)} = \log_2 e + \lambda, \\ p(S_4|V_4) \log_2 \frac{p(S_4|V_4)}{p(S_4)} = \log_2 e + \lambda, \\ p(V_1) + p(V_2) + p(V_3) + p(V_4) = 1, \end{cases}$$

where $\log_2 e + \lambda$ equals to the channel capacity. Since $p(S_4|V_1) = p(S_4|V_2) = p(S_4|V_3) = p$, $p(S_1|V_1) = p(S_2|V_2) = p(S_3|V_3) = 1 - p = q$, and $p(S_4|V_4) = 1$,

$$\begin{cases} q \log_2 \frac{1}{p(V_1)} + p \log_2 \frac{p}{p + qp(V_4)} = \log_2 e + \lambda, & (12) \\ q \log_2 \frac{1}{p(V_2)} + p \log_2 \frac{p}{p + qp(V_4)} = \log_2 e + \lambda, & (13) \\ q \log_2 \frac{1}{p(V_3)} + p \log_2 \frac{p}{p + qp(V_4)} = \log_2 e + \lambda, & (14) \\ \log_2 \frac{1}{p + qp(V_4)} = \log_2 e + \lambda, & (15) \\ p(V_1) + p(V_2) + p(V_3) + p(V_4) = 1. & (16) \end{cases}$$

From Equations (12)–(16), we obtain

$$\begin{aligned} &- q \log_2 p(V_1) + p \log_2 p - p \log_2 (p + qp(V_4)) \\ &= - \log_2 (p + qp(V_4)). \end{aligned}$$

Therefore,

$$p(V_1) = \frac{1}{p^{-p} + 3q} = p(V_2) = p(V_3) = \frac{q(V_4)}{3}. \qquad (17)$$

If $p(V_4) \geq 0$, (17) is the optimum distribution of the input. Thus,

$$\begin{aligned} C_s &= I\left(\mathbf{V};\mathbf{S}|p(V_1), p(V_2), p(V_3), p(V_4)\right) \\ &= q \log_2 (p^{-p} + 3q) + p \log_2 \frac{p^q + 3pq}{p^q + qp^{-p}}. \end{aligned}$$

However, if $p(V_4) < 0$, we set $p(V_4)$ to 0 to solve Equations (12)–(16) again and obtain

$$p(V_1) = p(V_2) = p(V_3) = \frac{1}{3}.$$

In this case,

$$C_s = q \log_2 3.$$

Competing interest

The authors declare that they have no competing interests.

Acknowledgements
The authors would like to thank all the anonymous reviewers for their valuable comments. This study was supported by the National Natural Science Foundation of China (61071070).

Received: 15 April 2012 Accepted: 17 August 2012
Published: 25 September 2012

References

1. P Pavan, R Bez, P Olivo, E Zanoni, Flash memory cells—an overview. Proc. IEEE. **85**, 1248–1271 (1997)
2. NAND Flash Design and Use Considerations Introduction, Micron TN-29-17. [http://download.micron.com/pdf/technotes/nand/tn2917.pdf]
3. J Wang, G Dong, T Zhang, R Wesel, *Mutual-information optimized quantization for LDPC decoding of accurately modeled flash data*, (2012). arXiv:1202.1325v1 [cs.IT]
4. G Dong, N Xie, T Zhang, On the use of soft-decision error-correction codes in NAND Flash memory. IEEE Trans. Circuits Syst. I. **58**(2), 429–439 (2010)
5. R Gallager, Low density parity check codes. IRE Trans. Inf. Theory. **8**, 21–28 (1962)
6. WE Ryan, S Lin, *Channel Codes: Classical and Modern*. (Cambridge, MA, Cambridge University Press, 2009)
7. M Franceschini, G Ferrari, R Raheli, Does the performance of LDPC codes depend on the channel. IEEE Trans. Commun. **54**(12), 2129–2132 (2006)
8. F Zhang, H Pfister, A Jiang, LDPC codes for rank modulation in flash memories. in *IEEE International Symposium on Information Theory*. (Austin, TX, USA, June 2010) pp. 859–863
9. M Fossorier, M Mihaljevic, H Imai, Reduced complexity iterative decoding of low-density parity check codes based on belief propagation. IEEE Trans. Commun. **58**(5), 673–680 (1999)
10. AV Kuznetsov, T Kasami, S Yamamura, An error correcting scheme for defective memory. IEEE Trans. Inf. Theory. **24**(6), 712–718 (1978)
11. J Proakis, *Digital Communications*. (New York: McGraw-Hill, 1989)
12. D Slepian, JK Wolf, Noiseless coding of correlated information sources. IEEE Trans. Inf. Theory. **19**, 471–480 (1973)
13. RJ McEliece, *The Theory of Information and Coding: A Mathematical Framework for Communication*. (MA, Addison-Wesley, Reading, 1977)
14. A Liveris, Z Xiong, C Georghiades, Compression of binary sources with side information at the decoder using LDPC codes. IEEE Commun. Lett. **6**(10), 440–442 (2002)
15. TM Cover, JA Thomas, *Elements of Information Theory*. (New York, Wiley, 1991)
16. X Zhang, J Zhu, Y Wu, *Efficient one-pass Chase soft-decision BCH decoder for multi-level cell NAND flash memory*. (Seoul, Korea, August 7–10, 2011)

doi:10.1186/1687-6180-2012-208
Cite this article as: Huang *et al.*: A concatenation scheme of LDPC codes and source codes for flash memories. *EURASIP Journal on Advances in Signal Processing* 2012 **2012**:208.