# Enabling NAND Flash Memory Use Soft-Decision Error Correction Codes at Minimal Read Latency Overhead

Guiqiang Dong, *Student Member, IEEE*, Ningde Xie, and Tong Zhang, *Senior Member, IEEE*

***Abstract*—With the aggressive technology scaling and use of multi-bit per cell storage, NAND flash memory is subject to continuous degradation of raw storage reliability and demands more and more powerful error correction codes (ECC). This inevitable trend makes conventional BCH code increasingly inadequate, and iterative coding solutions such as LDPC codes become very natural alternative options. However, these powerful coding solutions demand soft-decision memory sensing, which results in longer on-chip memory sensing latency and memory-to-controller data transfer latency. Leveraging well-established lossless data compression theories, this paper presents several simple design techniques that can reduce such latency penalty caused by soft-decision ECCs. Their effectiveness have been well demonstrated through extensive simulations, and the results suggest that the latency can be reduced by up to 85.3%.**

***Index Terms*—LDPC, lossless compression, NAND flash memory, soft sensing.**

## I. Introduction

NAND flash memory must use error correction code (ECC) to ensure system-level data storage integrity, where BCH codes with classical hard-decision decoding algorithms [1] are being widely used in current design practice. As the industry continues to push the technology scaling envelope and pursue aggressive use of multi-level per cell (MLC) storage, raw storage reliability of NAND flash memory continues to degrade, which makes BCH code inadequate and hence naturally demands more powerful ECCs. Because of their well-proven superior error correction capability with reasonably low decoding complexity, advanced ECCs, such as low-density parity-check (LDPC) code and Reed-Solomon (RS) codes with soft-decision decoding algorithms, appear to be promising candidates. In particular, LDPC receives the most attentions from the industry (e.g., see [2]–[4]). These advanced ECCs are soft-decision in nature and hence demand NAND flash memory perform finer-grained soft-decision sensing, and their error correction performance depends on the precision of memory sensing. As a result, straightforward use of these advanced ECCs can significantly degrade the overall data storage system read response latency for two reasons: (i) NAND flash memory on-chip sensing latency is linearly proportional to the required sensing quantization granularity and (ii) the flash-to-controller data transfer latency will accordingly increase. Since the read response latency is one of the most important metrics in most applications, how to practically apply advanced ECCs in future flash-based data storage systems is a nontrivial issue.

This paper presents a set of design techniques that apply lossless data compression to reduce the latency overhead of both on-chip flash memory sensing and memory-to-controller data transfer. First, we propose to combine lossless data compression with multi-rate ECC processing to reduce the latency overhead. The rationale is intuitive and can be described as follows. In NAND flash memory, data storage is realized in the unit of page, and each page stores a certain amount of user data and the associated ECC coding redundancy. If one page of user data can be losslessly compressed, more storage space will become available to store ECC coding redundancy, which makes it possible to use a stronger-than-normal code. For ECC demanding soft-decision memory sensing, a stronger ECC on average requires less soft-decision precision than a weaker ECC. Therefore, intra-page lossless data compression can enable an opportunistic use of stronger-than-normal ECC, which can reduce memory sensing precision. Since the lossless data compression is carried out for each individual page, it can be completely transparent to the file system and users, hence is called transparent lossless data compression.

We further propose several techniques that can reduce memory-to-controller data transfer latency overhead induced by soft-decision memory sensing. Recent work [5], [6] well demonstrated the effectiveness of using nonuniform memory sensing quantization to reduce the required sensing precision. In addition, the latency can be reduced by using a progressive sensing/decoding strategy that progressively increases the precision of memory sensing in a trial-and-error manner. In this work, we propose three design techniques that complement with existing nonuniform quantization and progressive sensing design solutions to further reduce memory-to-controller data transfer latency. First, we propose to apply entropy coding [7], a family of classical data lossless compression schemes, to encode the nonuniform sensing results before they are transferred

Manuscript received August 20, 2012; revised December 01, 2012; accepted December 17, 2012. Date of publication June 11, 2013; date of current version August 26, 2013. This work was supported in part by the National Science Foundation under Grant 1162152. This paper was recommended by Associate Editor F. Lustenberger.

G. Dong is with Skyera Inc, San Jose, CA 95131 USA (e-mail: dongguiqiang@gmail.com).

N. Xie is with Intel, Hillsboro, OR 97124 USA.

T. Zhang is with Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute (RPI), Troy, NY 12180 USA (e-mail: tzhang@ecse.rpi.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2013.2244361

from flash memory to controller. Second, we propose a zoned entropy coding scheme in the context of progressive memory sensing, which is motivated by the following observation. In progressive sensing, previous sensing results already determines the possible region of memory cell threshold voltage, hence we only need to uniquely encode different sensing results within the same possible region, instead of uniquely encoding different sensing results within the entire threshold voltage window. This leads to a so-called zoned entropy coding scheme that can reduce the memory-to-controller data transfer latency. Moreover, we propose to apply run-length coding to the bitstream of entire page soft-decision sensing result in order to further reduce the memory-to-controller data transfer latency. The key is to exploit the run-length characteristics of the entire bitstream when entropy coding is applied to each memory cell sensing result.

Using a hypothetical 2 bits/cell NAND flash memory as a test vehicle, we carry out simulations to evaluate the effectiveness of these design techniques. For the purpose of simplicity, we only consider two-step progressive sensing with the 1st-step 4-level uniform hard-decision sensing and 2nd-step 7-level nonuniform soft-decision sensing. Assuming the use of the well-known LZSS lossless compression algorithm [8] and LDPC codes, simulation results show that the intra-page transparent lossless compression can significantly reduce the average read latency. Compared with the conventional fixed-length coding, the entropy coding can reduce the 2nd-step soft-decision sensing result transfer latency by 20.4%. Compared with the conventional fixed-length coding, the proposed zoned entropy coding strategy can reduce the data transfer latency by 64.8% for the 2nd-step soft-decision sensing. The additional use of run-length coding can further reduce the corresponding data transfer latency to 14.7%.

## II. Background

Each NAND flash memory cell is a floating gate transistor whose threshold voltage can be programmed by injecting certain amount of charges into the floating gate. Before memory cells can be re-programmed, they must be erased (i.e., evicting all the charges from the floating gate). In order to ensure sufficient noise margin, NAND flash memory programming must achieve a tight control on threshold voltage distribution, which is realized by using incremental step pulse programming (ISPP) [9]. Major memory cell storage distortion and noise sources are cell-to-cell interference [10], [11] and program/erase (P/E) cycling effects that include random telegraph noise (RTN) [12], [13] and interface state trap recovery and electron detrapping [14], [15].

As technology continues to scale down, these memory cell storage distortion and noise sources become increasingly significant, leading to continuous degradation of NAND flash memory raw storage reliability. As a result, the industry has been actively pursuing the transition of ECC from conventional BCH codes to more powerful soft-decision iterative coding solutions, in particular LDPC codes. Nevertheless, since NAND flash memory sensing latency is proportional to the number of sensing quantization levels and the sensing results must be transferred to the memory controller through standard chip-to-chip links, a straightforward use of soft-decision ECC can result in significant read latency overhead. We may reduce the latency overhead from two aspects:

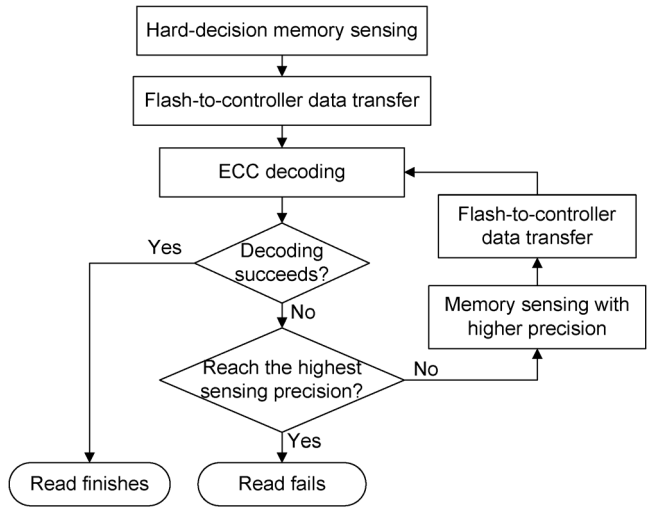



Fig. 1. Illustration of operational flow of progressive soft-decision sensing.

*1) Progressive Soft-Decision Sensing:* Fig. 1 illustrates the intuitive progressive soft-decision sensing strategy, which aims to use just-enough sensing precision for ECC decoding through a trial-and-error manner. This method can reduce the average read latency overhead for two reasons: (i) NAND flash memory raw storage reliability gradually degrades with the P/E cycling, hence fine-grained sensing may only be necessary as flash memory approaches its end of lifetime, and low-overhead coarse-grained sensing could be sufficient during memory early lifetime. (ii) Since ECC must ensure an extremely low page error rate (e.g., $10^{-12}$), low-overhead coarse-grained sensing may achieve a reasonably low error rate (e.g., $10^{-2}$), which makes the progressive sensing strategy justifiable, especially for applications that are not sensitive to read latency variation.

*2) Nonuniform Quantization Sensing:* As demonstrated in recent work [5], [6], compared with uniform quantization, nonuniform sensing quantization can noticeably reduce the required sensing precision without loss of error correction capability. Equivalently, given the same sensing precision, nonuniform quantization can lead to better ECC decoding performance than uniform quantization. As illustrated in Fig. 2, the key is to concentrate the sensing quantization around the dominating overlap regions of adjacent storage levels.

Moreover, Fig. 3 quantitatively demonstrates the advantages of soft-decision ECC (in particular LDPC code) over existing BCH code. All the codes protect 4 kB user data and have 8/9 code rate. For the LDPC code, we carried out simulations for both hard-decision sensing precision and soft seven-level sensing precision. As shown in the figure, although hard-decision decoding of LDPC code can slightly outperform the BCH code, soft-decision decoding can significantly improve the performance and advantage over BCH code.

This work concerns how to further reduce the read latency overhead caused by the use of soft-decision ECC. Very intuitively, this can only be achieved from two aspects: (i) go reduce the hard-decision ECC decoding failure probability and (ii) to

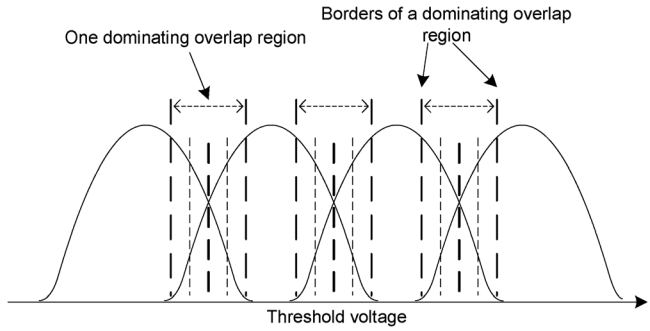



Fig. 2. Illustration of nonuniform sensing quantization for NAND flash memory.

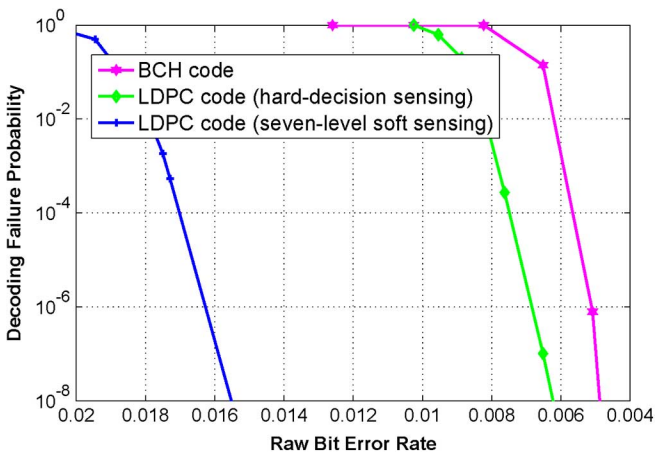



Fig. 3. Simulation results of rate-8/9 BCH and LDPC codes.

reduce the flash-to-controller data transfer latency in the case of soft-decision memory sensing. In the following two sections, we will present techniques that apply lossless compression schemes to reduce the read latency overhead from the above two aspects respectively.

## III. Reducing Hard-Decision ECC Decoding Failure Probability

One of the most prevailing real-life applications of data compression is to increase the effective storage capacity of mass data storage media. However, compared with lossy data compression being used in multimedia file storage, lossless data compression is much less popular for two main reasons: (i) a compressed file may not be easily modified and managed on the storage media and (ii) since the lossless data compression ratio[1] can rarely be much better than 2:1 in contrast to lossy data compression ratio of over hundreds, there is less incentive in using lossless data compression to save storage space. As a result, many files stored on mass data storage media today are not compressed at all although they are losslessly compressible. This observation directly motivates the basic idea of this work (as illustrated in Fig. 4): We can apply run-time transparent lossless data compression to opportunistically enable the use of a stronger ECC. When using the progressive memory sensing strategy, the stronger-than-normal error correction capability can reduce the hard-decision ECC decoding failure probability, leading to a reduced read latency. We note that lossless decompression only involves simple table lookup and can be done on-the-fly, hence it does not induce any noticeable read latency overhead.

Since compression ratio of lossless data compression heavily depends on the data length, it can be very difficult to achieve a large compression ratio such as 2:1 within each individual page. Fortunately, as demonstrated in the following case study, the compression ratio may not necessarily be a concern in practice. Different from the conventional use of lossless data compression that aims to improve effective storage capacity, a very small compression ratio can provide enough extra storage space for a sufficiently stronger ECC. For example, a compression ratio of 1.11:1 can reduce the ECC code rate from 19/20 to 6/7, which represents a very large coding gain and hence can enable a much coarser-grained memory sensing. Such a weak demand on compression ratio can drastically simplify the implementation of intra-page transparent lossless data compression. Moreover, current NAND flash memories have reasonably large page size, e.g., most NAND flash memories store 4 kbyte or 8 kbyte user data per page. Such relatively large page sizes make it more feasible to achieve a small compression ratio within each page at reasonable implementation cost.

[1]In this work, compression ratio is defined as the ratio of the data length before compression over the length after compression, hence it should be greater than one.

## IV. Reducing Soft-Decision Sensing Data Transfer Latency

As pointed out above, soft-decision sensing increases the latency of both on-chip memory sensing and memory-to-controller data transfer. The existing progressive sensing and nonuniform quantization sensing schemes described in Section II can reduce the latency overhead from both aspects. In this section, we present techniques that can further reduce the memory-to-controller soft-decision sensing data transfer latency by applying lossless data compression concepts.

### A. Applying Entropy Coding to Memory Sensing Results

Most straightforwardly, we can use a fixed-length binary representation to encode the soft-decision sensing result of each memory cell, i.e., when we quantize the threshold voltage of each memory cell into $n$ regions, we use an unique $\lceil \log_2 n \rceil$-bit number to represent each region. From the source coding point of view, such a straightforward fixed-length coding is optimal (i.e., the maximum entropy can be achieved) only if all the threshold voltage quantization regions have (almost) the same probability. As described in Section II, prior work [5], [6] demonstrated the effectiveness of using nonuniform sensing quantization to reduce the number of sensing quantization levels and hence reduce the on-chip memory sensing latency. In the presence of nonuniform sensing quantization, the fixed-length binary representation of sensing results is sub-optimal, leading to a low-entropy data stream.

The above discussion suggests that we should employ entropy coding to better match the nonuniform sensing quantization in order to improve the entropy of the sensing results. For example, let us consider 2 bits/cell NAND flash memory with the memory cell threshold voltage distribution as shown in

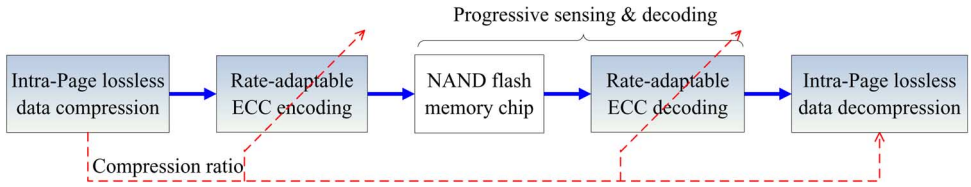



Fig. 4. Illustration of the proposed design strategy to apply transparent lossless data compression to reduce hard-decision ECC decoding failure probability.

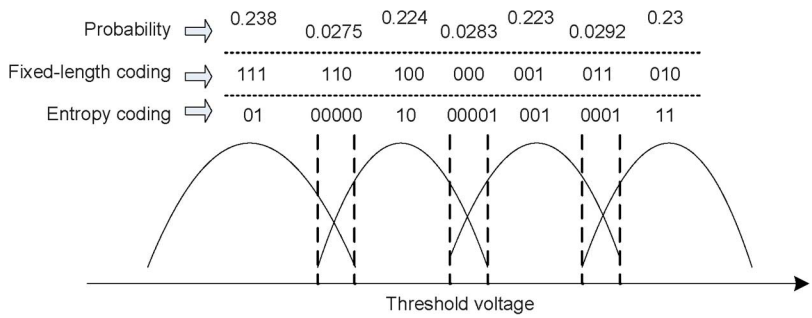



Fig. 5. Simple example to illustrate the use of fixed-length coding and entropy coding to represent memory sensing results.

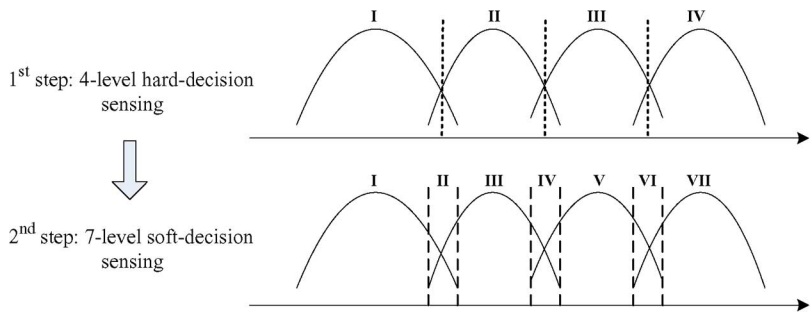



Fig. 6. Illustration of progressive memory sensing with the 1st step of 4-level hard-decision sensing and 2nd step of 7-level soft-decision sensing.

Fig. 5. Suppose we apply a 7-level nonuniform sensing quantization with the corresponding probability of each region as shown in Fig. 5, the use of an entropy coding can reduce the number of bits for sensing results representation from 3 to 2.45, leading to about 20.4% reduction of memory-to-controller data transfer latency.

### *B. Progressive Sensing With Zoned Entropy Coding*

As pointed out in Section II, we can exploit flash memory wear-out dynamics and graceful sensing precision vs. error correction capability to employ progressive memory sensing (as illustrated in Fig. 1) to reduce the read latency overhead. As illustrated in Fig. 6, memory sensing is first carried out with a hard-decision quantization, and if the hard-decision ECC decoding fails, a higher-precision soft-decision sensing is carried out in order to improve ECC decoding performance.

Intuitively, even in the case of hard-decision ECC decoding failure, the memory controller already has certain knowledge (i.e., the hard-decision sensing result) regarding the threshold voltage of each memory cell, and it is not necessary to transfer the complete soft-decision sensing results. For example, let us consider 2 bits/cell NAND flash memory with two steps of progressive sensing as illustrated in Fig. 6. Suppose the 1st-step hard-decision sensing result of one memory cell is the region II, the possible 2nd-step soft-decision sensing result can only be regions II, III, or IV. Therefore, we only need at most a 2-bit representation to transfer such extra memory sensing information to the controller, which is further combined with the existing hard-decision sensing results to obtain the complete soft-decision sensing results [16]. The same concept can be straightforwardly extended to all the following steps throughout the entire progressive sensing process.

Entropy coding can be used to further reduce the data transfer latency in the case of progressive sensing. The key is to exploit the unequal probability of different sensing regions. For example, let us consider the progressive sensing scenario as illustrated in Fig. 6. Suppose the possible 2nd-step soft-decision sensing result can only be regions II, III, or IV, there is a much higher probability that the sensing result is the region III than the other two regions. Therefore, we can apply entropy coding as illustrated in Fig. 7 to reduce the amount of data to be transferred from the flash memory to the controller. Given the 1st-step hard-decision sensing results, we must ensure the two or three consecutive sensing regions within each entropy coding zone, as illustrated in Fig. 7, must be uniquely coded, while sensing regions within different entropy coding zones can be represented with the same codeword. This method is referred to as *zoned entropy coding*.

Fig. 8 illustrates the process of final sensing result construction in the case of 2-step progressive sensing with zoned en-

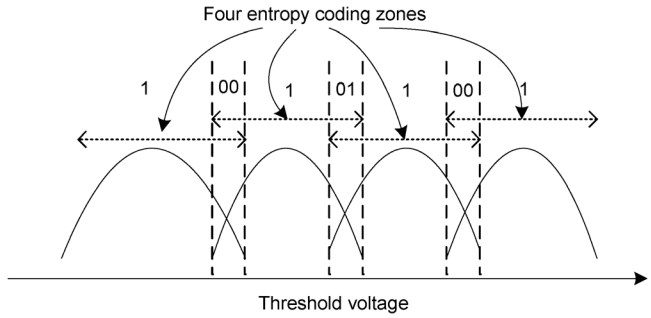



Fig. 7. Illustration of zoned entropy coding.

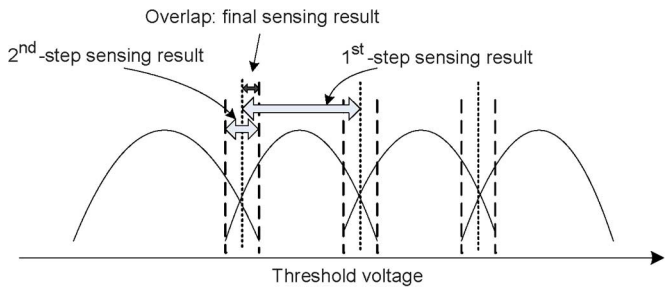



Fig. 8. Illustration of constructing real soft-decision sensing result in progressive sensing.

tropy coding. The overlap between the 1st-step hard-decision and the 2nd-step zoned entropy coded soft-decision sensing result can uniquely determine the final soft-decision sensing result. This process can be extended straightforwardly to progressive sensing with more than two steps.

### C. Run-Length Coded Data Transfer

When using the progressive sensing with zoned entropy coding as illustrated in Fig. 7, the sensing results of most memory cells will be a single '1'. Hence, sensing result bit stream tends to contain many strings of consecutive 1's. This motivates us to consider the use of run-length coding on the sensing result bit stream. As a very simple form of data lossless compression being widely used in fax machines, run-length coding represents each string of consecutive identical elements with a single element and length of the string. Very naively, leveraging the higher probability of occurrence of 1 in the sensing result bit stream, we can apply the run-length coding to further reduce the amount of sensing data to be transferred from the NAND flash memory to the controller.

For the 2nd-step nonuniform memory sensing quantization with zoned entropy coding as shown in Fig. 7, let $p_{00}$, $p_1$, and $p_{01}$ denote the probability that the zoned entropy coded sensing result is '00', '1', and '01', respectively. Since any string (or run) of consecutive 1's can have either a '00' or '01' on both sides, we can estimate the probability of the occurrence of 1-string with the length of $n$ as

$$P^{(1,n)} = \left(p_{00} \cdot p_1^n + p_{01} \cdot p_1^{n-1}\right) \cdot (p_{00} + p_{01}). \tag{1}$$

To simplify the run-length codec implementation, we apply run-length coding to both 1-string and 0-string, and use fixed-length coding to encode the run length. Regarding the occurrence of 0-string, if its length is an even number, the 0-string can only contain multiple '00' and have '1' on both sides, if its length is an odd number, the 0-string can follow either a '1' or '01' and only be followed by a '01'. Hence, we can estimate the probability of the occurrence of 0-string with the length of $2n$ and $2n+1$ as

$$P^{(0,2n)} = (p_1 + p_{01}) \cdot p_{00}^n \cdot p_1 \tag{2}$$

$$P^{(0,2n+1)} = (p_1 + p_{01}) \cdot p_{00}^n \cdot p_{01}. \tag{3}$$

Assume we use $r_1$-bit and $r_0$-bit word to encode the length of each 1-string and 0-string, we can estimate the average length of encoded 1-strings and 0-strings as

$$L_{en}^{(1)} = \sum_{n=1}^{\infty} \left(P^{(1,n)} \cdot \left\lceil \frac{n}{2^{r_1}} \right\rceil \cdot r_1\right) \tag{4}$$

$$L_{en}^{(0)} = \sum_{n=0}^{\infty} \left(P^{(0,2n)} \cdot \left\lceil \frac{2n}{2^{r_0}} \right\rceil + P^{(0,2n+1)} \cdot \left\lceil \frac{2n+1}{2^{r_0}} \right\rceil\right) \cdot r_0. \tag{5}$$

Meanwhile, we can estimate the average length of uncoded 1-strings and 0-strings as

$$L_{un}^{(1)} = \sum_{n=1}^{\infty} \left(P^{(1,n)} \cdot n\right) \tag{6}$$

$$L_{un}^{(0)} = \sum_{n=0}^{\infty} \left(P^{(0,2n)} \cdot 2n + P^{(0,2n+1)} \cdot (2n+1)\right). \tag{7}$$

Clearly, given the probabilities $p_{00}$, $p_1$, and $p_{01}$, we should carefully choose the value of $r_1$ and $r_0$ in order to minimize the compression ration and hence maximize the reduction of flash-to-controller data transfer latency.

Based upon the above discussions, we can naturally combine the progressive sensing with entropy coding, which is applied to the sensing result of each individual memory cell, and run-length coding, which is applied to the entire bit stream of sensing results of each page. Fig. 9 further illustrates such combination in the case of 2-step progressive memory sensing.

## V. Case Studies

For the purpose of quantitative evaluation, we develop a quantitative NAND flash memory device model that can capture the major threshold voltage distortion sources. Based upon this model, we carry out simulations to evaluate the above presented design techniques.

### A. NAND Flash Device Model

*1) Erase and Programming Operation Modeling:* The threshold voltage of erased memory cells tends to have a wide Gaussian-like distribution [17], and we approximately model the threshold voltage distribution of erased state as

$$p_e(x) = \frac{1}{\sigma_e\sqrt{2\pi}} e^{-\frac{(x-\mu_e)^2}{2\sigma_e^2}} \tag{8}$$

where $\mu_e$ and $\sigma_e$ are the mean and standard deviation of the erased state. The electron injection statistical spread [18] tends

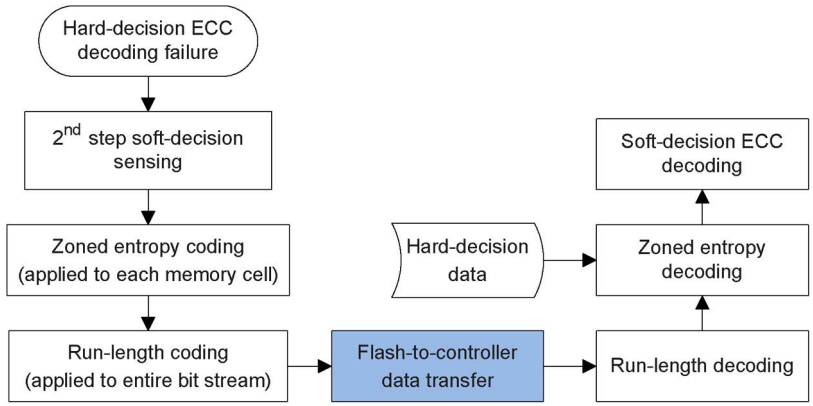



Fig. 9. Combination of zoned entropy coding and run-length coding to reduce read latency overhead.

to make the threshold voltage of programmed states more like Gaussian distribution. Hence, in this work we approximately model the distribution of programmed state as

$$p_p(x) = \frac{1}{\sigma_p\sqrt{2\pi}} e^{-\frac{(x-\mu_p)^2}{2\sigma_p^2}} \tag{9}$$

where $\mu_p$ and $\sigma_p$ are the mean and standard deviation of the programmed state right after programming.

*2) Random Telegraph Noise (RTN):* The fluctuation magnitude of RTN is subject to exponential decay. The probability density function $p_r(x)$ of RTN-induced threshold voltage fluctuation is modeled as a symmetric exponential function [13]:

$$p_r(x) = \frac{1}{2\lambda_r} e^{-\frac{|x|}{\lambda_r}}. \tag{10}$$

Since the significance of RTN is proportional to the interface trap density, we model the mean of RTN, i.e., $\mu_{RTN} = 1/\lambda_r$, approximately follows

$$\mu_{RTN} = A_{RTN} \cdot N^{a_{IT}} \tag{11}$$

where $N$ is the number of P/E cycles that flash memory cells have endured, and $A_{RTN}$ and $a_{IT}$ are constant parameters that are extracted from experiments for estimating the dependence of telegraph noise on $N$.

*3) Retention Process:* Since interface trap recovery and electron detrapping processes tend to follow Poisson statistics [19], we approximately model the induced threshold voltage reduction as a Gaussian distribution, i.e., $p_t(x) = \mathcal{N}(\mu_d, \sigma_d^2)$. As demonstrated in relevant device studies [19], [20], mean value of threshold voltage shift scales approximately with $\ln(1+t)$ over the time, where $t$ is the data retention time. The mean value of retention shift is set to follow the mean of sum of interface traps and oxide traps:

$$\mu_d = (A_t \cdot N^{a_{IT}} + B_t \cdot N^{a_{OT}}) \cdot \ln(1+t) \tag{12}$$

where $N$ is the number of P/E cycles, and the other four constant parameters (i.e., $A_t$, $a_{IT}$, $B_t$, and $a_{OT}$) are constant for estimating the densities of interface traps and oxide traps.

Moreover, the significance of threshold voltage reduction induced by interface trap recovery and electron detrapping is also proportional to the initial threshold voltage magnitude [21]. Hence, we set the generated retention noise approximately scale $K_s(x - x_0)$, where $x$ is the initial threshold voltage before retention, and $x_0$ and $K_s$ are constants.

*4) Cell-to-Cell Interference:* To capture inevitable process variability in cell-to-cell interference model, we set both the vertical coupling ratio $\gamma_y$ and diagonal coupling ratio $\gamma_{xy}$ as random variables with truncated Gaussian distribution

$$p_c(x) = \begin{cases} \frac{c_c}{\sigma_c\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu_c)^2}{2\sigma_c^2}}, & \text{if } |x-\mu_c| \le w_c \\ 0, & \text{else} \end{cases} \tag{13}$$

where $\mu_c$ and $\sigma_c$ are the mean and standard deviation, and $c_c$ is chosen to ensure the integration of this bounded Gaussian distribution equals to 1. According to [22], we set the ratio between the means of $\gamma_x$, $\gamma_y$ and $\gamma_{xy}$ as 0.1:0.08:0.006.

*5) Overall Device Model:* Based upon (8) and (9), we can obtain the threshold voltage distribution function $p_p(x)$ right after programming operation. Then we have the threshold voltage distribution after incorporating RTN $p_{ar}(x)$ as

$$p_{ar}(x) = p_p(x) \bigotimes p_r(x) \tag{14}$$

where $\bigotimes$ denotes the convolution operation. After we incorporate the cell-to-cell interference, we have the threshold voltage distribution $p_{ac}$. Let $p_t(x)$ denote the distribution of retention noise caused by interface state trap recovery and electron detrapping. The final threshold voltage distribution $p_f$ is obtained as

$$p_f(x) = p_{ac}(x) \bigotimes p_t(x). \tag{15}$$

In this work, we use 2 bits/cell NAND flash memory with the following device parameters as a test vehicle. We set the normalized $\sigma_e$ and $\mu_e$ of the erased state as 0.35 and 1.4, respectively. For programmed state, we set the normalized program step voltage $\Delta V_{pp}$ as 0.2, and its deviation as 0.05. According to [15], the exponents for interface and oxide traps generation are estimated as $a_{IT} = 0.62$ and $a_{OT} = 0.3$, respectively. For RTN, we set $A_{RTN} = 2.72 \times 10^{-4}$. The coupling strength factor is set as 1. As for retention shift, we set $\sigma_d = 0.3|\mu_d|$, and $A_t = 3.5 \times 10^{-5}$, $B_t = 2.35 \times 10^{-4}$, which are chosen to match the 70%:30% ratio of interface trap recovery and electron

TABLE I
SIMULATED CODE RATE REDUCTION FOR DIFFERENT LOSSLESS COMPRESSIBILITY, CORRESPONDING POSSIBILITY AND SENSING BIT WIDTH, AND THE OVERALL DATA TRANSFER LATENCY REDUCTION

| Coding rate | Compression ratio | Possibility | Sensing bits | Transfer latency reduction |
|---|---|---|---|---|
| 19/20 | 1 | 100% | 6 | 0 |
| 15/16 | 1.01 | 99.12% | 5 | 16.65% |
| 12/13 | 1.03 | 97.87% | 4 | 32.62% |
| 8/9 | 1.07 | 92.01% | 3 | 46.01% |
| 7/8 | 1.09 | 88.17% | 2 | 58.78% |

detrapping presented in [15]. Regarding the influence of the initial threshold voltage, we set $x_0 = 1.4$ and $K_s = 0.333$. We set $w_c = 0.1\ \mu_c$ and $\sigma_c = 0.4\ \mu_c$.

### B. Reducing Hard-Decision ECC Decoding Failure Probability

Based upon the above NAND flash memory device model, we first carried out simulations to evaluate the effectiveness of using transparent intra-page lossless compression. In this work, we set that each page contains 4 k-byte user data and is protected by an LDPC code. Ideally, when using transparent intra-page lossless compression, we may expect that the ECC code rate can be gracefully adjusted with a very fine granularity in response to the compression ratio. However, such ideal scenarios with fine-grained configurability may largely complicate the LDPC encoder and decoder silicon implementation. Therefore, in this case study, we consider the scenario that only five different code rates are allowed. In this work, we chose the popular LZSS lossless compression algorithm [8], and apply it to various user data, executable and system files on one of our department servers with Red Hat enterprise edition Linux and several EDA software installed. Table I lists the compressibility, the corresponding possibility.

We constructed a rate-19/20 LDPC code as the baseline code when transparent lossless data compression is not being used, i.e., this rate-19/20 baseline LDPC code protects 4 kbyte compressed user data. Corresponding to the code rate reduction factors listed in Table I, we further constructed four LDPC codes with the code rate of 15/16, 12/13, 8/9, and 7/8, respectively. All the LDPC codes are regular quasi-cyclic LDPC (QC-LDPC) code with the parity check matrix column weight of 4, which are constructed randomly subject to the 4-cycle free constraint. LDPC code decoding employs the min-sum decoding algorithm [23]. Based upon the NAND flash memory noise model presented above, we carried out simulations to evaluate the error correction performance under the baseline scenario. For the other QC-LDPC codes with lower code rates (i.e., rate 15/16, 12/13, 8/9, and 7/8), we carried out simulations based upon the same NAND flash memory noise model in order to search the corresponding reduced sensing precision that can ensure the error correction performance no-worse-than the baseline scenario. Fig. 10 and Table I show the simulated LDPC decoding failure probability under different P/E cycles. The results clearly show that the use of transparent lossless compression can noticeably reduce LDPC decoding failure probability under low-precision sensing, leading to reduced read latency when the progressive sensing strategy is being used.

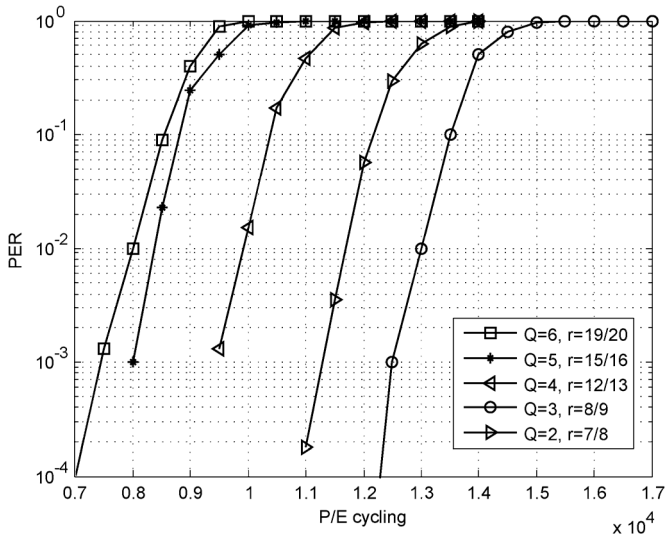



Fig. 10. PER of LDPC codes with various coding rate under different sensing precision in MLC NAND flash memory.

### C. Reducing Soft-Sensing Data Transfer Latency

In this case study, for the purpose of simplicity, we only consider the 1st-step 4-level hard-decision sensing and 2nd-step 7-level soft-decision nonuniform sensing as illustrated in Fig. 6. Based upon the NAND flash memory device model presented above, we can obtain the threshold voltage distribution under different P/E cycling. Using the nonuniform memory sensing quantization strategies presented in [5], [6], we estimated the probabilities of the 7 levels in the 2nd-step nonuniform sensing as listed in Table II assuming the present P/E cycle number is 4000. It is obvious that there is large difference among all the sensing levels in probabilities. This implies we can apply entropy coding to this, to reduce the average codeword length. If we use the straightforward entropy coding to uniquely encode all the 7 levels according to their probabilities, we have the entropy codewords as listed in Table II. Compared with the conventional 3-bit fixed-length coding, such straightforward entropy coding can reduce the 2nd-step soft-decision sensing result transfer latency by 18.4%.

In progressive sensing scheme, previous sensing results already determines the possible region of threshold voltage, hence we only need to uniquely encode different sensing results within the same possible region, instead of uniquely encoding different sensing results within the entire memory cell threshold voltage operational window. This leads to the zoned entropy coding that aims to apply entropy coding within each possible region determined by previous sensing results, as shown in Fig. 7. Table II lists the codeword of each level in the case of zoned entropy

TABLE II
PROBABILITIES AND CODEWORDS OF THE 2ND-STEP SOFT-DECISION SENSING RESULTS

| Level index | Probability | Fixed-length coding | Entropy coding | Zoned entropy coding |
|---|---|---|---|---|
| I | 23.97% | 111 | 01 | 1 |
| II | 2.55% | 110 | 00000 | 00 |
| III | 22.55% | 100 | 10 | 1 |
| IV | 2.63% | 000 | 00001 | 01 |
| V | 22.45% | 001 | 001 | 1 |
| VI | 2.70% | 011 | 0001 | 00 |
| VII | 23.15% | 010 | 11 | 1 |

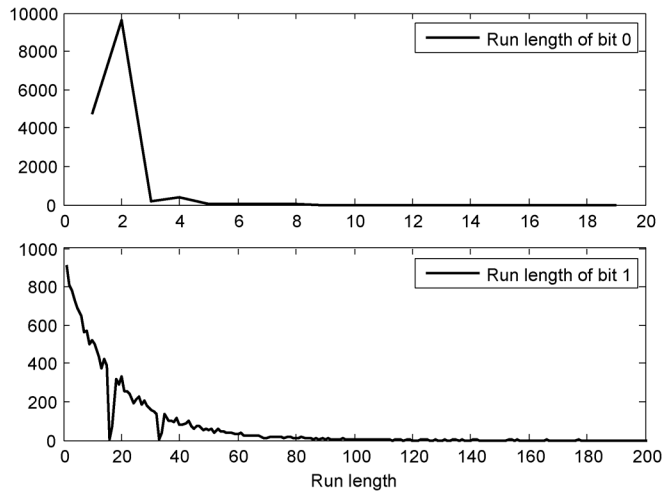



Fig. 11. Run-length of bit 1 and 0 in the 2nd-step progressive sensing bit stream.

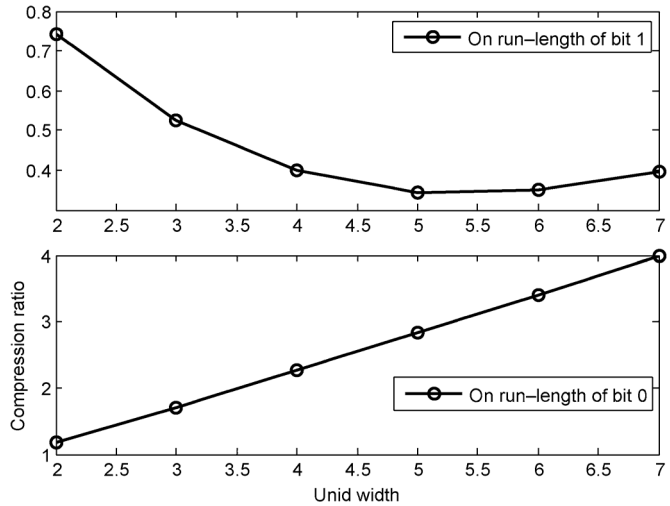



Fig. 12. Simulated compression ratios for 1-string and 0-string respectively under different values of $r_1$ and $r_0$.

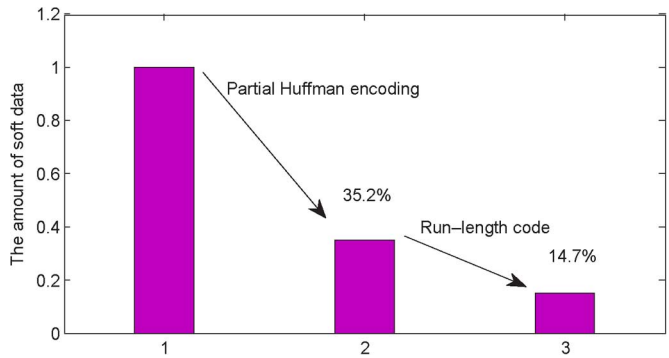



Fig. 13. Illustration of the effectiveness of zoned entropy coding and run-length coding.

coding. As illustrated in Fig. 7, we apply entropy coding to four zones individually, which are formed by all the 7 levels as {I, II}, {II, III, IV}, {IV, V, VI}, and {VI, VII}, respectively. These zoned entropy codewords will be decoded and combined with the hard-decision sensing results to construct the complete soft-decision sensing result. In this context, there is a relatively much higher probability that we only need to transfer one bit for each cell during the 2nd-step soft-decision sensing. Compared with the conventional fixed-length coding and complete entropy coding, this zoned entropy coding strategy can reduce the data transfer latency by 64.8% and 20.4%, respectively, for the 2nd-step soft-decision sensing.

Next, we investigate the effectiveness of using run-length coding to further reduce the read latency overhead. As discussed in Section IV-C, the key is to leverage the fact that, when using progressive sensing with zoned entropy coding as illustrated in Fig. 7, the probability of the occurrence of 1 s is much higher than that of 0. As a result, there is a relatively high probability that a string of consecutive 1's may occur in the sensing result bit stream. Under the P/E cycling of 4000, we estimate the statistics of run length of bit 1 and bit 0 as shown in Fig. 11.

As discussed in Section IV-C, the run-length coding codeword size (i.e., $r_1$ for 1-string and $r_0$ for 0-string) should be carefully chosen in order to maximize the compression efficiency. Fig. 12 further shows the simulated results on the compression ratio for 1-string and 0-string respectively under different values of $r_1$ and $r_0$. The results clearly suggest that we can set $r_1$ as 5 and set $r_0$ as 2, and the overall compression ratio is estimated as 41.8%.

Fig. 13 shows the effectiveness of both zoned entropy coding and run-length coding. Compared with the scenario without using any compression schemes on soft-decision memory sensing results, the zoned entropy coding can reduce the amount of sensing data (hence data transfer latency) to 35.2%, and the use of run-length coding can further reduce it to 14.7%.

### *D. Discussions on Implementation Overhead*

The proposed design techniques, including intra-page lossless compression, entropy coding, and run-length coding, inevitably result in latency and silicon overhead. Although the main focus of this work is to demonstrate the potential effectiveness of using these well-established compression schemes to reduce latency penalty caused by soft-decision ECC, it is still worthy to discuss the implementation overhead. First, it is reasonable to expect that the latency overhead is very small mainly because all these functions process the data in a streaming manner (i.e., continuous data-in and continuous data-out). As a result, the net extra latency is almost negligible (e.g., only tens of cycles in comparison to the data transfer latency of thousands of cycles). Regarding the silicon overhead, we implemented the LZSS compression core using a 1

kbyte content addressable memory (CAM) with 128 address entries and 8-byte word length for each address for high speed comparison, and it contains a 128-byte buffer. Using 65 nm technology library and Synopsys synthesis tool, we estimation its silicon cost is 0.05 $\mathrm{mm}^2$. The Huffman and run-length encoding/decoding are much simpler than LZSS compression, hence we can estimate that the total silicon cost is less than 0.06 $\mathrm{mm}^2$ at 65 nm node, hence we can safely conclude that the total implementation overhead is almost negligible as well.

## VI. Conclusion

This paper concerns the memory read latency overhead induced by the use of powerful soft-decision ECC in future NAND flash memory. Although ECCs such as LDPC codes can achieve excellent error correction capability, their soft-decision decoding nature directly results in significant latency overhead in terms of on-chip memory sensing and memory-to-controller data transfer. We propose several simple yet effective design techniques that can reduce such read latency penalty. We first proposed to apply intra-page lossless compression to opportunistically enable the use of stronger-than-normal ECC in order to reduce the probability of invoking soft-decision memory sensing. To reduce memory-to-controller data transfer latency, we propose three techniques. The first technique is to straightforwardly complement existing nonuniform memory sensing quantization scheme with entropy coding to reduce data transfer latency. The second technique is to apply zoned entropy coding in the context of progressive memory sensing to reduce the data transfer latency. The third technique applies run-length coding on the entire soft-decision sensing result stream in order to further reduce the data transfer latency. Based upon an approximate NAND flash memory device model, we carried out simulations and the results clearly demonstrate the effectiveness of the proposed design solutions.

## References


[1] R. E. Blahut, *Algebraic Codes For Data Transmission*. Cambridge, U.K.: Cambridge University Press, 2003.

[2] J. Yang, "Novel ECC architecture enhances storage system reliability," in *Proc. Flash Memory Summit*, Aug. 2012.

[3] X. Hu, "LDPC codes for flash channel," in *Proc. Flash Memory Summit*, Aug. 2012.

[4] E. Yeo, "An LDPC-enabled flash controller in 40 nm CMOS," in *Proc. Flash Memory Summit*, Aug. 2012.

[5] G. Dong, N. Xie, and T. Zhang, "On the use of soft-decision errorcorrection codes in nand flash memory," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 2, pp. 429–439, 2011.

[6] J. Wang, T. A. Courtade, H. Shankar, and R. D. Wesel, "Soft information for LDPC decoding in flash: Mutual-information optimized quantization," in *Proc. Globecom*, 2011.

[7] K. Sayood, *Lossless Compression Handbook*. New York: Academic Press, 2003.

[8] J. A. Storer and T. G. Szymanski, "Data compression vis textual substitution," *J. ACM*, vol. 29, no. 4, pp. 928–951, Oct. 1982.

[9] K.-D. Suh *et al.*, "A 3.3 V 32 Mb NAND flash memory with incremental step pulse programming scheme," *IEEE J. Solid-State Circuits*, vol. 30, no. 11, pp. 1149–1156, Nov. 1995.

[10] J.-D. Lee, S.-H. Hur, and J.-D. Choi, "Effects of floating-gate interference on NAND flash memory cell operation," *IEEE Electron. Device Letters*, vol. 23, no. 5, pp. 264–266, May 2002.

[11] H. Liu, S. Groothuis, C. Mouli, J. Li, K. Parat, and T. Krishnamohan, "3D simulation study of cell-cell interference in advanced NAND flash memory," in *Proc. IEEE Workshop on Microelectronics and Electron Devices*, Apr. 2009.

[12] K. Fukuda, Y. Shimizu, K. Amemiya, M. Kamoshida, and C. Hu, "Random telegraph noise in flash memories model and technology scaling," in *Proc. IEEE Int. Electron Devices Meeting*, 2007, pp. 169–172.

[13] C. M. Compagnoni, M. Ghidotti, A. L. Lacaita, A. S. Spinelli, and A. Visconti, "Random telegraph noise effect on the programmed threshold-voltage distribution of flash memories," *IEEE Electron Device Lett.*, vol. 30, no. 9, Sep. 2009.

[14] N. Mielke, H. P. Belgal, A. Fazio, Q. Meng, and N. Righos, "Recovery effects in the distributed cycling of flash memories," in *Proc. IEEE Int. Reliability Physics Symp.*, 2006, pp. 29–35.

[15] H. Yang *et al.*, "Reliability issues and models of sub-90 nm NAND flash memory cells," in *Proc. Int. Conf. Solid-State and Integrated Circuit Technology*, Oct. 2006, pp. 760–762.

[16] G. Dong, Y. Zou, and T. Zhang, "Reducing data transfer latency of NAND flash memory with soft-decision sensing," in *IEEE ICC Workshop on Emerging Data Storage Technologies*.

[17] K. Takeuchi, T. Tanaka, and H. Nakamura, "A double-level-Vth select gate array architecture for multilevel NAND flash memories," *IEEE J. Solid-State Circuits*, vol. 31, no. 4, pp. 602–609, Apr. 1996.

[18] C. M. Compagnoni, A. S. Spinelli, R. Gusmeroli, A. L. Lacaita, S. Beltrami, A. Ghetti, and A. Visconti, "First evidence for injection statistics accuracy limitations in NAND Flash constant-current Fowler-Nordheim programming," in *Proc. IEEE Int. Electron Devices Meeting*, 2007, pp. 165–168.

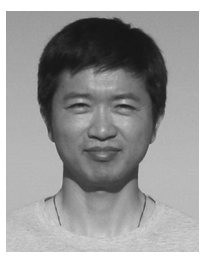
[19] N. Mielke, H. Belgal, I. Kalastirsky, P. Kalavade, A. Kurtz, Q. Meng, N. Righos, and J. Wu, "Flash EEPROM threshold instabilities due to charge trapping during program/erase cycling," *IEEE Trans. Device Mater. Rel.*, vol. 4, no. 3, pp. 335–344, 2004.

[20] C. M. Compagnoni, C. Miccoli, R. Mottadelli, S. Beltrami, M. Ghidotti, A. L. Lacaita, A. S. Spinelli, and A. Visconti, "Investigation of the threshold voltage instability after distributed cycling in nanoscale NAND flash memory arrays," in *Proc. IEEE Int. Reliability Physics Symp. (IRPS)*, 2010, pp. 604–610.

[21] J. D. Lee, J. H. Choi, D. Park, K. Kim, R. D. Center, S. E. Co, and S. K. Gyunggi-Do, "Effects of interface trap generation and annihilation on the data retention characteristics of flash memory cells," *IEEE Trans. Device Mater. Rel.*, vol. 4, no. 1, pp. 110–117, Jan. 2004.

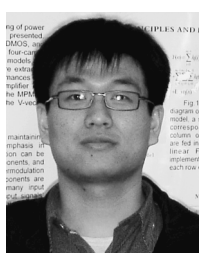
[22] N. Shibata *et al.*, "A 70 nm 16 Gb 16-level-cell NAND flash memory," in *IEEE Symp. VLSI Circuits*, 2007, pp. 190–191.

[23] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288–1299, Aug. 2005.

**Guiqiang Dong** (S'09) received the B.S. and M.S. degrees from the University of Science and Technology of China, Hefei, in 2004 and 2008, respectively, and the Ph.D. degree from the Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute (RPI), Troy, NY, in 2012.

He has been with Skyera Inc., San Jose, CA, since 2012. His research interests include coding theory, signal processing, and FTL design for solid-state storage. During his Ph.D. study at RPI, he found novel methods and designed software tools to estimate very fast error-floor of LDPC codes. Currently, he is working on ECC and FTL design in NAND based solid-state drive.



**Ningde Xie** received the B.S. and M.S. degrees in radio engineering from Southeast University, Nanjing, China, in 2004 and 2006, respectively, and the Ph.D. degree from the Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY, in 2010.

He has been working in the Storage Technology Group at Intel, Hillsboro, OR, since 2010. His research interests include system and architecture design for high performance, low power communication, and storage systems. Currently, he is working on error control coding and signal processing in NAND based solid-state drive, as well as system design in phase change memory.

**Tong Zhang** (M'02–SM'08) received the B.S. and M.S. degrees in electrical engineering from Xian Jiaotong University, Xian, China, in 1995 and 1998, respectively, and the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, in 2002.

He is currently an Associate Professor with the Department of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY. His research activities span over circuits and systems for various data storage and computing applications.

Dr. Zhang currently serves as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II and the IEEE TRANSACTIONS ON SIGNAL PROCESSING.