Using Lossless Data Compression in Data Storage Systems: Not for Saving Space

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

Abstract—Lossless data compression for data storage has become less popular as mass data storage systems are becoming increasingly cheap. This leaves many files stored on mass data storage media uncompressed although they are losslessly compressible. This paper proposes to exploit the lossless compressibility of those files to improve the underlying storage system performance metrics such as energy efficiency and access speed, other than saving storage space as in conventional practice. The key idea is to apply runtime lossless data compression to enable an opportunistic use of a stronger error correction code (ECC) with more coding redundancy in data storage systems, and trade such opportunistic extra error correction capability to improve other system performance metrics in the runtime. Since data storage is typically realized in the unit of equal-sized sectors (e.g., 512 B or 4 KB user data per sector), we only apply this strategy to each individual sector independently in order to be completely transparent to the firmware, operating systems, and users. Using low-density parity check (LDPC) code as ECC in storage systems, this paper quantitatively studies the effectiveness of this design strategy in both hard disk drives and NAND flash memories. For hard disk drives, we use this design strategy to reduce average hard disk drive read channel signal processing energy consumption, and results show that up to 38 percent read channel energy saving can be achieved. For NAND flash memories, we use this design strategy to improve average NAND flash memory write speed, and results show that up to 36 percent write speed improvement can be achieved for 2 bits/cell NAND flash memories.

Index Terms—Data storage, lossless compression, hard disk drive, NAND flash memory, error correction code.

1 Introduction

One of the most prevailing real-life applications of data compression [1] is to increase the effective storage capacity of mass data storage media such as conventional hard disk drives and emerging solid-state drives based on NAND flash memory. However, compared with lossy data compression being universally used in multimedia file storage, lossless data compression is much less popular, which may arguably attribute to two main reasons: 1) a compressed file may not be easily modified and managed on the storage media and 2) 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 interest and incentive in using lossless data compression to save storage space, particularly as mass data storage systems are becoming increasingly cheap nowadays. Therefore, although lossless data compression was used to compress hard disk drives in early 1990s when hard disk drives were very small (e.g., less than 100 MB), it has become much less popular today. As a result, except multimedia data files, many files stored on mass data storage media today are not compressed at all although they are losslessly compressible, such as Web pages, text files, software and system files, etc. This work aims to exploit the lossless compressibility of those files to improve the underlying storage system performance metrics such as energy efficiency and access speed, other than saving storage space as in the conventional practice.

The rationale of this work is briefly described as follows: In hard disk drives and NAND flash memory, data storage is realized in the unit of sectors,[2] where the sector size ranges from 512-byte to 4-Kbyte user data. At present, the hard disk drive industry is migrating from the 30-year old 512-byte sector size to a new 4-Kbyte sector size, and most NAND flash memories use 4-Kbyte sector size. As the storage density continues to grow, both hard disk drives and NAND flash memory use increasingly powerful error correction code (ECC) on each individual data sector to ensure storage reliability [2], [3]. A more powerful ECC with stronger error correction capability tends to demand more redundant bits, and hence, occupy more storage space. Therefore, designers always select an ECC that provides *just enough* error correction capability to satisfy the given reliability specifications. It is clear that if one sector user data can be losslessly compressed to a certain extent, more storage space will become available to store ECC coding redundancy, which makes it possible to use a stronger ECC with more-than-enough error correction capability for the present sector. Intuitively, such opportunistic extra error correction capability may be traded to improve other system performance metrics in the runtime.

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.

2. In fact, *sector* is the terminology being used in hard disk drives. In NAND flash memory, it is called *page*. For the sake of simplicity, we use the terminology *sector* to represent the basic storage unit in both hard disk drives and NAND flash memory.

• *The authors are with the Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180. E-mail: ningdexie@gmail.com, dongg2@rpi.edu, tzhang@ecse.rpi.edu.*

Manuscript received 1 Sept. 2009; revised 23 Dec. 2009; accepted 19 Feb. 2010; published online 11 June 2010.
Recommended for acceptance by D. Gizopoulos.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-2009-09-0447.
Digital Object Identifier no. 10.1109/TC.2010.150.

Motivated by the above intuitive discussion, this paper aims to explore the practical feasibility and potential of such design strategy in real-life data storage systems including both hard disk drives and NAND flash memories. In particular, this work focuses on using low-density parity check (LDPC) code [4], [5], [6] as ECC for two main reasons: 1) As a topic of great current interest, LDPC codes have excellent error correction capability and are being widely used in real-life communication and storage systems; and 2) in sharp contrast to classical linear blocks codes such as BCH and Reed-Solomon (RS) codes [7], the computational complexity of LDPC code in each decoding iteration is largely independent with the code rate (and hence, error correction capability), which will be further elaborated in Section 2. To sustain the continuous growth of areal storage density of hard disk drives, besides the use of more and more powerful ECCs, increasingly sophisticated read channel with complex digital signal processing must be used [8], [9]. As a result, entire hard disk drive read channels tend to become increasingly power-hungry. Hence, in the context of hard disk drives, the more-than-enough error correction capability enabled by runtime opportunistic intrasector lossless data compression can be exploited to enable the use of less complex and less power-hungry read channel signal processing. Based upon a representative perpendicular recording read channel architecture, we demonstrate that this design strategy can reduce the entire read channel power consumption by up to 38 percent. It should be pointed out that although reducing energy consumption of read channel may not noticeably impact the total hard disk drive energy consumption, it is still valuable to reduce read channel energy consumption mainly because it can reduce the on-chip power delivery system design complexity and reduce heat dissipation, and hence, reduce packaging cost/complexity.

NAND flash memory has a relatively slow write speed with the write latency of a few hundred microseconds (e.g., see [10], [11]), which may become a system performance bottleneck, particularly for applications with heavy data traffics. As explained in Section 4, there is an inherent trade-off between the memory write speed and raw memory storage reliability in NAND flash memories. Therefore, the more-than-enough error correction capability enabled by runtime opportunistic intrasector lossless data compression can allow the raw memory storage reliability to degrade to a certain extent, which can be directly leveraged to improve the NAND flash memory write speed. Our simulation results show that the write speed can be improved by up to 36 percent for 2bits/cell NAND flash memory. Finally, we note that since this design strategy is applied to each sector independently at the physical layer, it is completely transparent to the firmware, operating systems, and users.

The remainder of this paper is organized as follows: Section 2 elaborates on the proposed design strategy and presents the basics of LDPC codes. Sections 3 and 4 present case studies on hard disk drives and NAND flash memory, respectively, to show the effectiveness of this proposed design strategy. The conclusions are drawn in Section 5.

2 Proposed Design Strategy

The key idea of this work is to apply runtime intrasector lossless data compression to opportunistically enable the use of a stronger ECC whose stronger-than-necessary error correction capability can be traded for improving certain data storage system performance metrics. The practical feasibility of this very intuitive idea nevertheless is subject to the following two issues:

1. Compression ratio of lossless data compression tends to heavily depend on the data length; hence, it can be very difficult to achieve a large compression ratio such as 2:1 within each individual sector.
2. Generally speaking, a stronger ECC tends to incur a higher decoding implementation cost and energy consumption, which may possibly offset the potential gain if we want to trade error correction capability for storage system energy consumption.

Moreover, the implementation and energy consumption cost of on-the-fly lossless data compression should also be carefully taken into account in practice.

As demonstrated in the case studies presented in Sections 3 and 4, the first issue above may not necessarily be a concern in practice. Different from the conventional use of lossless data compression that is used to improve effective storage capacity, and hence, should achieve a compression ratio of close or even larger than 2:1, a very small compression ratio such as 1.05:1 can provide enough extra storage space for a sufficiently stronger ECC to enable this design strategy. For example, a compression ratio of 1.03:1 can reduce the ECC code rate from 15/16 to 10/11, which represents a large coding gain (i.e., large error correction capability improvement). Hence, this can drastically simplify the implementation of intrasector lossless data compression. Moreover, modern data storage systems have reasonably large sector size, e.g., hard disk drives are migrating from 512-byte user data per sector to 4-Kbyte user data per sector, and most NAND flash memories use 4-Kbyte user data per sector (and some latest NAND flash memories even use 8-Kbyte user data per sector [12], [13]). Such relatively large sector sizes make it more feasible to achieve a small compression ratio within each sector at reasonable implementation cost.

To address the second issue above on the ECC decoder overhead, this work focuses on using LDPC code as the ECC in data storage systems. It is true that for classical linear block codes such as BCH and RS codes, their decoding computational complexities quadratically grow with their error correction capability. In sharp contrast, the decoding computational complexity of LDPC codes is largely independent on their code rate (and hence, error correction capability), i.e., once we fix the LDPC codeword length and the number of 1s in its parity check matrix, the decoding computational complexity is largely independent on the code rate. Meanwhile, as the storage densities are being pushed toward their limits for both hard disk drives and NAND flash memory, more powerful ECCs become almost indispensable, for which LDPC codes appear to be the most promising candidate and have been very actively studied for hard disk drives (e.g., see [14], [15], [16], [17], [18], [19]). In fact, LSI Corporation, one of leading hard disk drive chip vendors, recently announced that LDPC codes have been used in their latest hard disk drive read channel chips at the 40 nm technology node (see [20]). We note that LDPC codes

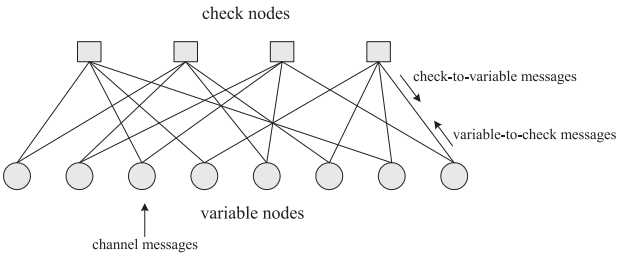


Fig. 1. Message-passing decoding based on LDPC code bipartite graph.

have been adopted by many recent communication standards such as DVB-S2 for video broadcasting, IEEE 802.11 for wireless LAN, and IEEE 802.3 for 10 Gigabit Ethernet.

In the following, we briefly explain why the LDPC code decoding computational complexity is largely independent on the code rate (and hence, the error correction capability). Readers are referred to [4], [5], [6] for detailed discussions on LDPC codes. An LDPC code is defined as the null space of an $M \times N$ sparse parity check matrix. It can be represented by a bipartite graph between M check (or constraint) nodes in one set and N variable (or message) nodes in the other set. For any "1" entry in the sparse parity check matrix at (i, j), there will be a connection between the ith check node and jth variable node in the bipartite graph. An LDPC code can be effectively decoded by the iterative message-passing decoding algorithm [5], [21] that directly matches the code bipartite graph, as illustrated in Fig. 1: After each variable node is initialized with the input channel message, the decoding messages are iteratively computed by all the variable nodes and check nodes and exchanged through the edges between the neighboring nodes.

Clearly, the LDPC code decoding computational complexity per decoding iteration only depends on the number of edges in the bipartite graph (i.e., the number of 1s in the LDPC code parity check matrix). Data storage systems always demand a relatively high ECC code rate, e.g., 8/9 and higher, for which the LDPC code parity check matrix should be regular (i.e., all the columns have the same number of 1s) and the column weight (i.e., the number of 1s per column) should be 3 or 4. Therefore, once we fix the ECC codeword length and the code parity check matrix column weight, the LDPC code decoding computational complexity and energy consumption per decoding iteration will remain almost the same, regardless to the code rate. Under the same condition, LDPC codes with different code rates may require different number of decoding iterations on average, i.e., a lower code rate LDPC code tends to require a (slightly) less number of decoding iterations than its higher code rate counterpart. Therefore, overall decoding energy consumption may (slightly) differ among LDPC codes with different code rates, and lower code rate codes tend to consume less decoding energy consumption.

Let R_n denote the code rate of the normal LDPC code used by the data storage system in current design practice without using any intrasector lossless data compression, i.e., the same rate-R_n LDPC code is used to protect all the sectors of uncompressed original user data as in current design practice. If one sector of user data can be compressed with a compression ratio of r_c, we could accordingly reduce the LDPC code rate from R_n to R_n/r_c, leading to a stronger error correction capability. This work aims to leverage such runtime opportunistically enhanced error correction capability to improve other performance metrics of data storage systems without sacrificing the overall storage system reliability. In particular, we consider both hard disk drives and NAND flash memory, as illustrated in Figs. 2a and 2b, which is further explained as follows:

- Modern hard disk drives employ very complex and power-intensive magnetic recording read channel signal processing [8], [9], which works together with the ECC to ensure a satisfied hard disk drive data storage reliability. In the presence of a stronger-than-necessary error correction capability enabled by the opportunistic intrasector lossless data compression, we could accordingly *relax* the signal processing performance requirement of the magnetic recording read channel signal processing. This allows the use of less complex, and hence, less power-intensive read channel signal processing. If the average read channel signal processing energy saving can offset the energy overhead induced by runtime intrasector lossless data compression, we can reduce the overall hard disk drive energy consumption.
- One of the critical drawbacks of NAND flash memory, especially multilevel NAND flash memory that largely dominates the entire flash memory market, is the relatively long write latency compared with other solid-state memory technologies. As explained in Section 4, there is an explicit trade-off between NAND flash memory write latency and raw storage reliability: As we reduce the NAND flash memory write latency by adjusting certain memory circuit operational parameters, the raw NAND flash memory storage reliability will degrade. The opportunistic stronger-than-necessary error correction capability enabled by the opportunistic intrasector lossless data compression can accommodate a worse

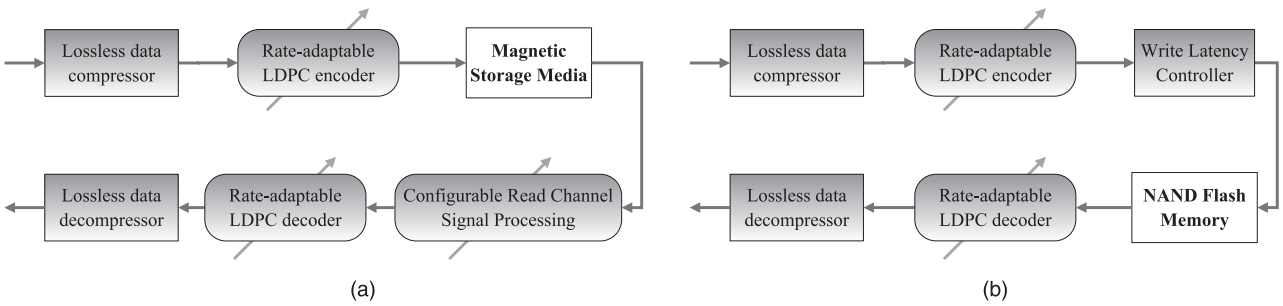


Fig. 2. Illustration of proposed design approaches for (a) hard disk drives and (b) NAND flash memory.

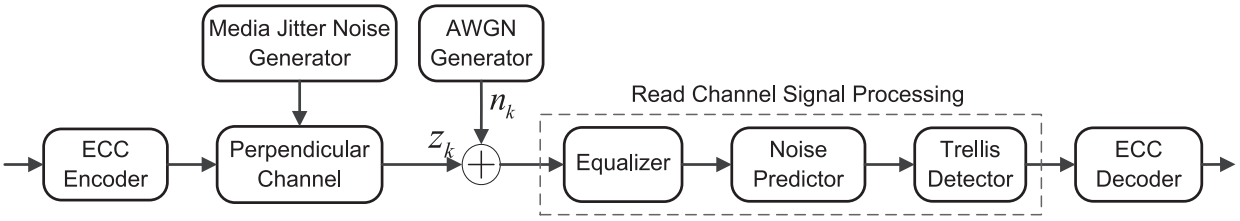


Fig. 3. Perpendicular read channel communications model.

raw NAND flash memory storage reliability, which can be directly leveraged to reduce the NAND flash memory write latency.

Ideally, we may expect that the LDPC code rate (and hence, error correction capability), together with the read channel signal processing (for hard disk drives) and write latency configuration (for NAND flash memories), can be gracefully adjusted with a very fine granularity in order to maximize the potential benefit. However, such ideal scenarios with fine-grained configurability and adaptability may largely complicate the overall system design and incur non-negligible implementation overhead. Therefore, in this work, we consider the simplest and most practical scenario where only two different LDPC code rates are allowed. Let R_n and R_s (where $R_n > R_s$) denote these two code rates, where R_n is the rate of the normal LDPC code when storage systems do not use any intrasector data compression and R_s is the rate of a stronger LDPC code enabled by opportunistic runtime lossless data compression. Accordingly, in case of hard disk drives, the read channel signal processing only needs to support two different signal performance versus power consumption trade-off configurations, and in case of NAND flash memory, the memory circuits only need to support two different memory write speed modes. Define compression ratio threshold $r_t = R_n/R_s$, given the runtime intrasector lossless compression ratio r, we have

- If $r \leq r_t$, then we use the normal rate-R_n LDPC code to protect the uncompressed user data, and accordingly, we set the read channel signal processing in its normal power mode (for hard disk drives) and set the memory write circuits operate with the normal write speed (for NAND flash memory).
- If $r > r_t$, then we use the stronger rate-R_s LDPC code to protect the compressed user data, and accordingly, we set the read channel signal processing in its low-power mode (for hard disk drives) and set the memory write circuits operate with the fast write speed (for NAND flash memory).

In the remainder of this paper, we present two case studies to quantitatively demonstrate the effectiveness of the above presented design strategy with its simplest two-mode realization to reduce read channel signal processing power consumption in hard disk drives and improve write speed in NAND flash memory.

3 Case Study I: Reducing Read Channel Energy Consumption in Hard Disk Drives

To sustain the continuous storage density growth of hard disk drives, magnetic recording read channel plays an increasingly important role and employs more and more powerful and complex signal processing [22], [23], which nevertheless incurs significant energy consumption overhead. In this section, we show that the above presented design strategy can be used to reduce the read channel signal processing power consumption.

3.1 Magnetic Recording Read Channel Basics

The magnetic recording media magnetic anisotropy can be either oriented in the recording medium plane, referred to as longitudinal recording, or aligned perpendicular to the recording medium plane, referred to as perpendicular recording [24]. Although it has been well recognized that perpendicular recording could result in higher storage density than its longitudinal counterpart, practical realization of suitable perpendicular recording media is much more complicated than its longitudinal counterpart [25]. As a result, longitudinal recording has been dominating the commercial hard disk drives until recently when the industry began to ship perpendicular recording hard disk drives in 2005. Since then, perpendicular recording has gained an ever increasing momentum, and the entire industry is quickly switching from longitudinal recording to perpendicular recording. Therefore, this work only considers the use of perpendicular recording. Fig. 3 shows a digital communication model for magnetic read channel [9]. It models the perpendicular recording channel as

$$z_k = \sum_m a_m h(kT - mT + \delta_m),$$

where

$$h(t) = erf\big(2t\sqrt{\ln 2}/(PW50)\big),$$

and the media jitter noise δ_m is modeled as a Gaussian variable $\mathcal{N}(0, \sigma_j^2)$. The parameter $PW50$ is defined as the pulsewidth of the derivative of $h(t)$ at half of its peak amplitude. Following [26], we define the overall SNR as

$$\mathrm{SNR} = \frac{\int_{-\infty}^{+\infty} \big(h(t) - h(t-T)\big)^2 \,\mathrm{d}t}{2\sigma_n^2 + 2\sigma_j^2 \int_{-\infty}^{+\infty} \big(h'(t)\big)^2 \,\mathrm{d}t},$$

where the signal energy is in the "dibit" response and the noise reflects the first-order jitter model. The read channel uses an equalizer, noise predictor, and detector to recover the recorded bit sequence, which is further passed to the ECC decoder.

As the storage density increases, the readback signal from hard disks is subject to severe intersymbol interference (ISI), i.e., readback signals from adjacent bits on the disk tend to interfere with each other. The equalizer is used to reshape the readback signals so that the equalized signal only depends on a small number of adjacent bits on the

disk, which allows a detector with reasonable hardware complexity. The equalizer in read channel typically is implemented in the form of an finite impulse response (FIR) filter that carries out a convolution:

$$r_k = \sum_{i=0}^{L_e-1} c_i \cdot x_{k-i},$$

where x_k are the input samples, c_i are the FIR equalizer coefficients, r_k are the output of equalizer, and L_e is the number of equalizer taps. Since the equalizer tends to introduce colored noise, which will degrade the subsequent signal detection, a noise predictor (also called noise whitening filter) is always used to further filter the output of equalizer so that the input to the subsequent trellis detector becomes

$$\sum_{i=0}^{L_p-1} a_i \cdot r_{k-i},$$

where a_i are the noise predictor coefficients and L_p is the number of filter taps. Assume that the equalizer has an M-order equalization target, i.e., the ISI of the equalized signal has the form $H(D) = h_0 + h_1 \cdot D + \cdots + h_M \cdot D^M$. The L_p-tap noise predictor will increase the order of ISI to $M + L_p - 1$, and hence, the state number of the subsequent trellis detector is 2^{M+L_p-1}. The design parameters M, L_e, and L_p directly affect the read channel signal processing performance versus power consumption trade-off.

3.2 A Baseline Read Channel Design

Following the perpendicular recording read channel parameter configurations presented in [27] where the ratio of the transition noise to the total noise power is 0.5 and the normalized channel bit density is 2.5 (we use these channel parameters for all the work), we design a baseline perpendicular recording read channel by choosing $H(D) = 1 + 0.75D$ with $M = 1$, $L_e = 10$, and $L_p = 3$ which has the reasonable complexity and performance. Therefore, the state number of the trellis detector is $2^{M+L_p-1} = 8$. Since the LDPC code decoder demands soft input, the trellis detector must be able to generate soft detection output. In this work, we implement the soft-output trellis detector using the soft-output Viterbi algorithm (SOVA) with a two-step detector structure [28]. Targeting at 4 KB user data per sector, we construct a regular rate-15/16 (34,976, 32,790) quasi-cyclic LDPC (QC-LDPC) code with the parity check matrix column weight of 4. The code parity check matrix contains an array of 2×32 circulant matrices, where all the circulant matrices have a column weight of 2 and are constructed randomly subject to the four-cycle free constraint. The QC-LDPC encoder is designed using the approach presented in [29], and the QC-LDPC decoder employs the min-sum decoding algorithm [30] and is implemented using the decoder architecture presented in [31]. To determine finite word length configurations, we carried out simulations and choose the configurations so that there is almost no performance gain if we further increase the finite word length. The chosen finite word length configuration is listed as follows:

- The coefficients and inputs/outputs of the equalizer and noise predictor use 6 bits.
- The path metric and soft output of the SOVA detector use 9 and 3 bits, respectively.
- The internal LDPC decoding messages use 3 bits.

With the above design parameters and assuming a 2 Gbps magnetic recording read channel operational throughput, we designed the entire read channel data path application-specific integrated circuit (ASIC) using Synopsys tools and TSMC 65 nm CMOS standard cell and SRAM libraries with 1.2 V supply voltage. The number of the maximum allowable internal LDPC decoding iterations is set to 16. Table 1 lists the power estimation results, which show that the read channel signal processing, including equalization, noise prediction, and SOVA detection, consume more than 58 percent of the total read channel power consumption.

TABLE 1
Baseline Read Channel Power Estimation Results

	Power Consumption (mW)
LDPC Decoder @ 11.1dB	101.76
8-state SOVA Detector	83.81
Equalizer + Noise Predictor	61.05
Total	246.62

3.3 Low-Power Read Channel Design

Based upon the above baseline read channel, this section quantitatively investigates the power saving potential when using the design strategy presented in Section 2. To reduce the read channel signal processing power consumption at the cost of degraded signal processing performance, we could reduce the equalization target order M, equalizer tap number L_e, and/or noise predictor tap number L_p. In the above baseline read channel, the equalization target order M is only 1. As a result, further decreasing M will drastically degrade the read channel signal processing performance, which can hardly be compensated by using a stronger LDPC code. Therefore, in this work, we only consider to decrease the other two parameters L_e and L_p.

As pointed out in Section 2, to most simplify its practical implementation, we only use two different LDPC codes (i.e., a normal high-rate LDPC code and a stronger low-rate LDPC code) and the read channel only needs to operate under two modes (i.e., a normal mode and a low-power mode). Following the above baseline read channel design, the normal LDPC code rate is 15/16 and the normal mode read channel signal processing has $L_e = 10$ and $L_p = 3$. Let L_{se} and L_{sp} denote the reduced tap numbers of the equalizer and noise predictor when read channel signal processing operates in the low-power mode. We set $L_{se} = 3$ and $L_{sp} = 2$ while keeping the same equalization target $H(D) = 1 + 0.75D$. As a result, the trellis state number reduces from 8 to 4 in the low-power mode. Clearly, when operating in the low-power mode, the read channel signal processing is subject to performance degradation. As shown in Fig. 4, if we keep the same rate-15/16 LDPC code, such read channel power reduction will come with about 0.5 dB performance loss. If we could leverage the intrasector lossless compression to enable the use of a rate-10/11 LDPC code, the overall read channel sector error rate (SER) performance can be fully recovered, as shown in Fig. 4. Hence, we have the two

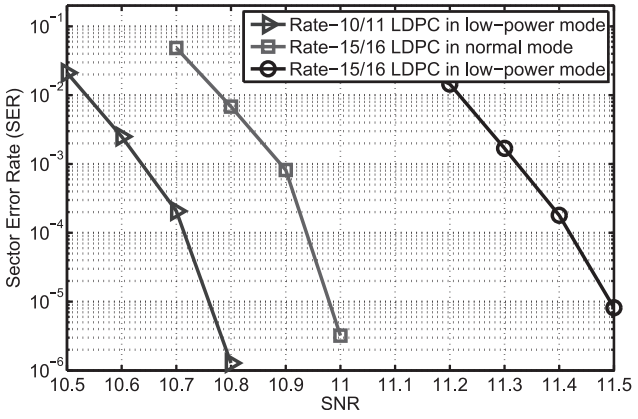


Fig. 4. Sector error rate comparison for different modes.

LDPC code rates $R_n = 15/16$ and $R_s = 10/11$, and the compression ratio threshold $r_t = R_n/R_s = 1.03$. Accordingly, given the runtime lossless compression ratio r, we have

- If $r \leq 1.03$, then we use the normal rate-15/16 LDPC code to protect the uncompressed user data, and accordingly, the read channel signal processing data path operates in the normal mode and contains a 10-tap equalizer, 3-tap noise predictor, and an 8-state SOVA detector.
- If $r > 1.03$, then we use the stronger rate-10/11 LDPC code to protect the compressed user data, and accordingly, the read channel signal processing data path operates in the low-power mode and contains a 3-tap equalizer, 2-tap noise predictor, and a 4-state SOVA detector.

Let P_{eq}, P_{prd}, P_{det}, and P_{LDPC} denote the power consumption of the 10-tap equalizer, 3-tap noise predictor, 8-state SOVA detector, and rate-15/16 LDPC code decoder in the normal mode read channel, respectively; and let P^l_{eq}, P^l_{prd}, P^l_{det}, and P^l_{LDPC} denote the power consumption of the 3-tap equalizer, 2-tap noise predictor, 4-state SOVA detector, and rate-10/11 LDPC code decoder in the low-power mode read channel, respectively. Moreover, let P_{com} denote the power consumption of the lossless data compressor. We note that the power consumptions of LDPC encoder and lossless data decompressor are relatively very small; hence, they are not considered in the following power saving analysis. Assume that the hard disk drive carries out read and write operations with the probabilities of α and $1-\alpha$, and let β represents the probability that the runtime lossless data compression ratio $r > 1.03$. Then, the total read channel power saving can be estimated as

$$P_{sav} = \alpha \cdot \beta(P_t - P^l_t) - (1-\alpha)P_{com}, \quad (1)$$

where

$$P_t = P_{eq} + P_{prd} + P_{det} + P_{LDPC},$$
$$P^l_t = P^l_{eq} + P^l_{prd} + P^l_{det} + P^l_{LDPC}.$$

To quantitatively evaluate the power saving and extra silicon area cost, we carried out further ASIC design using Synopsys tools and TSMC 65 nm CMOS standard cell and SRAM libraries with 1.2 V supply voltage. All the finite word length configurations remain exactly the same as those presented in the above baseline read channel design. Table 2 lists design results of silicon area. To support such a dual-mode operation, we need to redesign and/or add four components: lossless compressor, LDPC decoder, SOVA detector, and equalizer/noise predictor. Among them, the equalizer and noise predictor only need to store seven extra 6-bit coefficients, leading to negligible extra silicon cost. Hence, the silicon area of the equalizer and noise predictor remains the same, as shown in Table 2.

TABLE 2
Silicon Area Cost Results

	Silicon Area (mm^2)	
	Baseline	Dual-code-rate architecture
LDPC Decoder	1.32	1.36
SOVA Detector	0.27	0.28
Equalizer and Noise Predictor	0.12	0.12
LZ77 Compressor	N/A	0.05
Total	1.71	1.81

Targeting at relatively low lossless compression ratios for common and representative files including Web pages, e-mails, text files, executable files, and system files, etc., the lossless data compressor is designed based on the well-known LZ77 algorithm [32]. The LZ77 compressor mainly uses a 1 Kbyte content addressable memory (CAM) with 128 address entries and 8-byte word length for each address for high-speed comparison, and contains a 128-byte buffer. The estimated silicon area for the designed LZ77 compressor based on 65 nm technology is 0.05 mm^2.

In the context of LDPC code decoder, we use the decoder architecture presented in [31] that employs the min-sum decoding algorithm as pointed out in the above. To support such dual-code rate decoding with the same target decoding throughput, we only need to increase the storage capacity of the decoding message storage module and add configurability to the decoding computation module, which tends to incur small silicon overhead. Therefore, the overall silicon area of LDPC code decoder only increases from 1.32 to 1.36 mm^2, as shown in Table 2. In the context of SOVA detector, following the architecture presented in [28], in order to support both 4-state and 8-state SOVA, two extra MUXs are needed in the add, compare, and select (ACS) module. The survivor memory unit (SMU) and the path equivalence detector (PED) also need two extra MUXs for each step. Hence, we need to add $2 \cdot (10 + 10) = 40$ binary MUXs and two 9-bit MUXs in total, where 10 is the traceback length for the SMU and PED in the case of 4-state mode. As a result, the silicon area of SOVA detector increases from 0.27 to 0.28 mm^2, as shown in Table 2.

Table 3 summarizes the estimated power consumption when the read channel signal processing operates in the low-power mode. We notice that the rate-10/11 LDPC code decoder consumes slightly less power compared with the rate-15/16 LDPC code, which is mainly because the average LDPC internal decoding iteration number slightly reduces under the same channel SNR.

When the dual-mode data path works under the normal mode, the power consumption of LDPC decoder, SOVA detector, and equalizer/noise predictor is the same as the baseline power consumption in Table 1. Therefore, according

TABLE 3
Low-Power Mode Power Estimation Results

	Power Consumption (mW)
LDPC Decoder @ 11.1dB	P^l_{LDPC} = 93.26
4-state SOVA Detector	P^l_{det} = 29.08
Equalizer + Noise Predictor	$P^l_{eq} + P^l_{prd}$ = 26.94
LZ77 Compressor @ $r > 1.03$	P_{com} = 14.12
P^l_t	149.28

to (1) and Tables 1 and 3, we can calculate the power saving as $P_{sav} = 97.34 \cdot \alpha \cdot \beta - 14.12 \cdot (1 - \alpha)$, where α is the probability that one hard disk drive access is a read operation and β is the probability that the runtime lossless data compression ratio $r > 1.03$. Clearly, different environments and workloads will have different α and/or β, leading to different power saving potentials.

Fig. 5 shows the power saving for $\alpha \in [0.5, 1]$ and $\beta \in [0, 1]$. At the extreme scenario (i.e., each user data sector can be compressed with the ratio over 1.03 and all the system requests are read), about 39 percent power saving can be achieved. We can see that in order to achieve net power saving when half of the system requests are read, at least 18 percent user sectors stored in the hard disk should be compressible with the ratio over 1.03. Therefore, this proposed method is not effective when the disk drive is used for cameras, MP3/MP4 players, and media servers where most files in these devices are lossless incompressible. On the other hand, when the system requests are dominated by read and many frequently accessed files are losslessly compressible with the ratio over 1.03, there will be noticeable power saving. We note that for large-capacity hard disk drives in data centers and high-performance computing systems where high reliability is extremely important, people use the "scrubbing" technique [33] to detect latent errors by always trying to read hard disk drives during idle time. Meanwhile, the stored files in high-performance computing systems may be losslessly compressible with higher possibilities. Hence, this proposed design strategy can be particularly useful for those high-performance computing systems.

In order to further verify the power saving under real applications, we carried out compression for various user-stored data, executable and system files on one server with

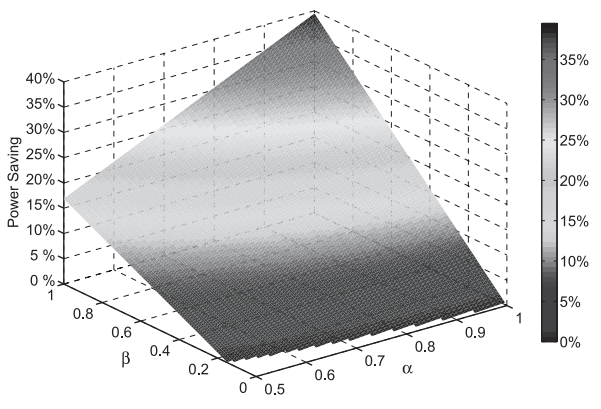


Fig. 5. Read channel power saving for different α and β.

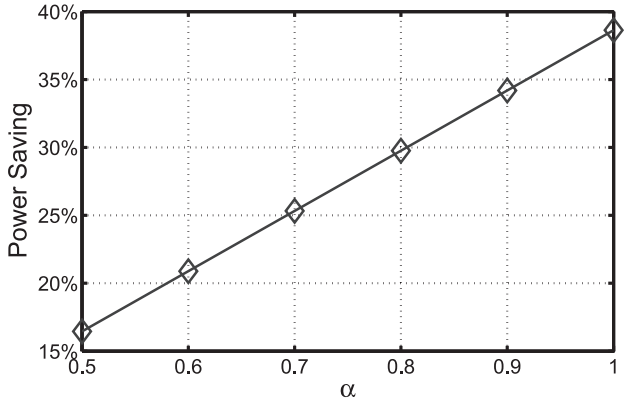


Fig. 6. Read channel power saving for different α under the environment of a Linux server.

red hat enterprise edition Linux, and several EDA software installed. The results show $\beta = 0.9787$. Fig. 6 shows the overall read channel power saving for $\alpha \in [0.5, 1]$ with β being fixed as 0.9787. Under this case, when the hard disk drive predominantly carries out read operations, the overall power saving can be as high as 38 percent.

4 Case Study II: Improving Write Speed of NAND Flash Memory

As the fastest growing segment in global semiconductor industry, NAND flash memory is being widely used in consumer electronics and quickly entering various high-end and personal computing system. As the technology continues to scale down and multibit per cell storage is being aggressively pursued to push the envelope of storage density, NAND flash memory faces increasingly serious storage reliability and endurance problems, which makes the use of powerful ECCs indispensable. Meanwhile, NAND flash memory has a relatively slow write speed with the typical write latency of a few hundred microseconds. In this section, we discuss and demonstrate the use of the above presented design strategy to improve NAND flash memory write speed.

4.1 Basics of NAND Flash Memory

Each NAND flash memory cell is a floating gate transistor whose threshold voltage can be configured (or programmed) by injecting certain amount of charges into the floating gate. The continuous growth of NAND flash memory storage density has been mainly driven by aggressive technology scaling, e.g., an NAND flash memory with 34 nm CMOS technology has been recently reported in [34]. In fact, NAND flash memory has surpass microprocessors as the leading edge technology scaling driver. Besides technology scaling, multilevel per cell (MLC) technique, i.e., to store more than 1 bit in each memory cell (or floating gate MOS transistor) by programming the cell threshold voltage into one of $l > 2$ voltage windows, has been widely used to further improve the NAND flash memory storage density. Because of its obvious storage density advantage, MLC NAND flash memory has been increasingly dominating flash memory market. In current design practice, most MLC NAND flash memories store 2 bits per cell, while 3 and even 4 bits per cell NAND flash memories have been recently reported [11], [12], [13], [35]

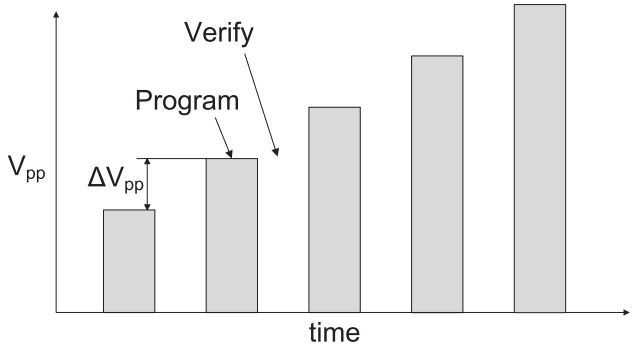


Fig. 7. Control gate voltage pulses in program-and-verify operation.

and are being manufactured by several NAND flash memory vendors such as Micron and Samsung.

Before one flash memory cell can be programmed, it must be erased (i.e., its threshold voltage is set to the lowest voltage window). A tight threshold voltage control is typically realized by using program-and-verify approach with a staircase program voltage V_{pp} [3], as illustrated in Fig. 7. Because NAND flash memory should sweep the entire operational voltage window using this program-and-verify procedure, the NAND flash memory write latency tends to be inversely proportional to the program step voltage ΔV_{pp}.

Ideally, threshold voltage distributions of different storage states should be sufficiently far away from each other to ensure a high raw storage reliability. In practice, due to various effects such as background pattern dependency, noises, and cell-to-cell interference [36], the threshold voltage distributions may be very close to each other or even overlap, leading to non-negligible raw bit error rates. In the following, we present an MLC cell threshold voltage distribution model that will be used for quantitative performance evaluation and comparison in this work. The erase state tends to have a wide Gaussian-like distribution [37], i.e., the probability density function (PDF) of the threshold voltage distribution can be approximated as

$$p_0(x) = \frac{1}{\sigma_0\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma_0^2}},$$

where σ_0 is the standard deviation and μ is the mean threshold voltage of the erase state. All the other states tend to have the same threshold voltage distribution, as illustrated in Fig. 8. The model consists of two parts, including an uniform distribution in the middle and Gaussian distribution tail on both sides [37]. The width of the uniform distribution equals to the program step voltage ΔV_{pp}, and the standard deviation of the Gaussian distribution is denoted as σ. The Gaussian distribution on both sides models the overall effect of background pattern dependency, noises, and cell-to-cell interference. Let P_0 and P_1 denote the probabilities of the uniform distribution and the Gaussian distribution, respectively. We have the overall PDF $f_{pr}(x)$ as

$$f_{pr}(x) = \begin{cases} \frac{c}{\sigma\sqrt{2\pi}}, & b - 0.5\Delta V_{pp} \le x \le b + 0.5\Delta V_{pp}, \\ \frac{c}{\sigma\sqrt{2\pi}} e^{-\frac{(x-b-0.5\Delta V_{pp})^2}{2\sigma^2}}, & x > b + 0.5\Delta V_{pp}, \\ \frac{c}{\sigma\sqrt{2\pi}} e^{-\frac{(x-b+0.5\Delta V_{pp})^2}{2\sigma^2}}, & x < b - 0.5\Delta V_{pp}, \end{cases}$$

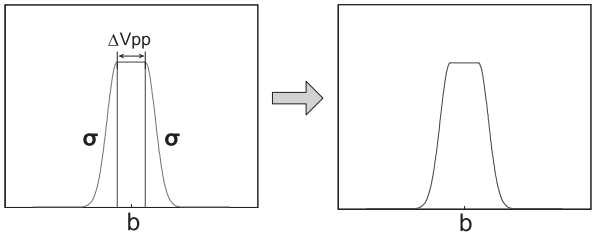


Fig. 8. Threshold voltage distribution model NAND flash memory (except the erase state).

where b is the mean of the threshold voltage (i.e., the center of the distribution, as shown in Fig. 8), and the constant c can be solved based on $P_0 + P_1 = \int_{-\infty}^{+\infty} f_{pr}(x)dx = 1$.

To push the storage density envelope, NAND flash memory cells are organized in an array→ block→ page hierarchy, as illustrated in Fig. 9, where an NAND flash memory array is partitioned into blocks, and each block contains a number of pages. Within each block, each memory cell string typically contains 16-64 memory cells, and all the memory cells driven by the same word line are programmed and sensed at the same time. All the memory cells within the same block must be erased at the same time. Data are programmed and fetched in the unit of page, where the page size ranges from 512 B to 8 KB user data.

4.2 Quantitative Evaluation

In this work, we consider the use of LDPC code as ECC for 2 bits/cell NAND flash memory with 4 KB page size. We use the same rate-15/16 QC-LDPC code as in Section 3 for hard disk drives. Since the LDPC code decoder requires soft input, we assume that each NAND flash memory cell is sensed with a 16-level uniform quantization scheme, as illustrated in Fig. 10, where the quantization thresholds are represented by the red dot lines.

In a baseline case when the user data are not compressed and the rate-15/16 LDPC code is being used, we set the program step voltage ΔV_{pp} as 0.16 while normalizing the distance between the mean of two adjacent threshold voltage windows as 1. Given the value of program step voltage ΔV_{pp}, the LDPC code decoding failure rate (i.e., the sector error rate) will depend on the standard deviations of the erased state (i.e., σ_0) and the other three programmed

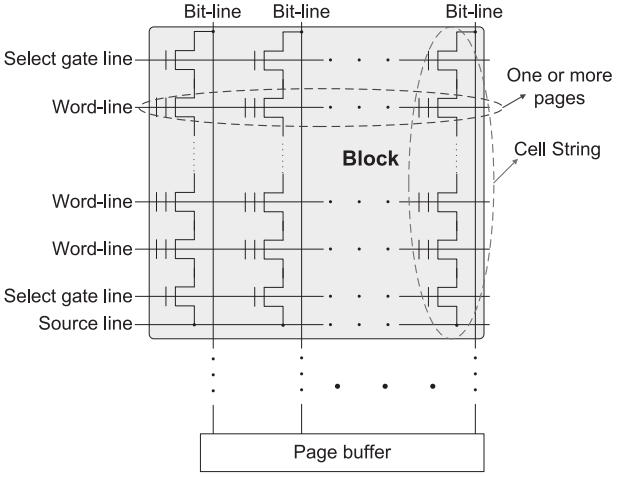


Fig. 9. NAND flash memory structure.

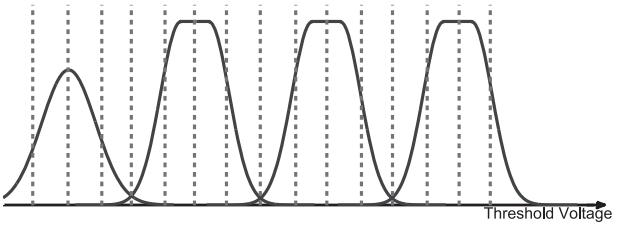


Fig. 10. The sensing quantization schemes for LDPC-coded system.

states (i.e., σ). In this work, we fix the normalized value of σ_0 as 0.2 and carry out simulations to evaluate the sector error rate versus normalized σ, as shown in Fig. 11. In order to improve the NAND flash memory write speed, we must increase the program step voltage ΔV_{pp}, which nevertheless will widen each programmed state threshold voltage window. As a result, there is a higher probability that adjacent states have overlaps, leading to a higher raw bit error rate. As shown in Fig. 11, if we increase ΔV_{pp} from 0.16 to 0.22 while keeping the same rate-15/16 LDPC code, a significant sector error rate degradation is incurred due to the degraded raw NAND flash memory storage reliability.

Using the design strategy presented in Section 2, we can exploit runtime intrasector lossless data compression to allow the use of a stronger LDPC code whose extra error correction capability can be directly traded to enable an increase of the program step voltage ΔV_{pp}. In this case study, we consider to use a stronger rate-9/10 LDPC code. Similarly, the rate-9/10 LDPC code is a regular QC-LDPC code with the parity check matrix column weight of 4. The code parity check matrix contains an array of 2×20 circulant matrices, where all the circulant matrices also have a column weight of 2 and are constructed randomly subject to the four-cycle free constraint. As shown in Fig. 12, when the rate-9/10 LDPC code is being used, we could increase the program step voltage ΔV_{pp} from 0.16 to 0.22 while maintaining almost the same overall NAND flash memory system storage reliability.

Hence, we have $R_n = 15/16$ and $R_s = 9/10$, and the compression ratio threshold $r_t = R_n/R_t = 1.04$. Accordingly, given the runtime lossless compression ratio r, we have

- If $r \leq 1.04$, then we use the normal rate-15/16 LDPC code to protect the uncompressed user data, and accordingly, the data are written to the NAND flash memory at a low speed with the normalized program step voltage ΔV_{pp} of 0.16.
- If $r > 1.04$, then we use the stronger rate-9/10 LDPC code to protect the compressed user data, and accordingly, the data are written to the NAND flash memory at a high speed with the normalized program step voltage ΔV_{pp} of 0.22.

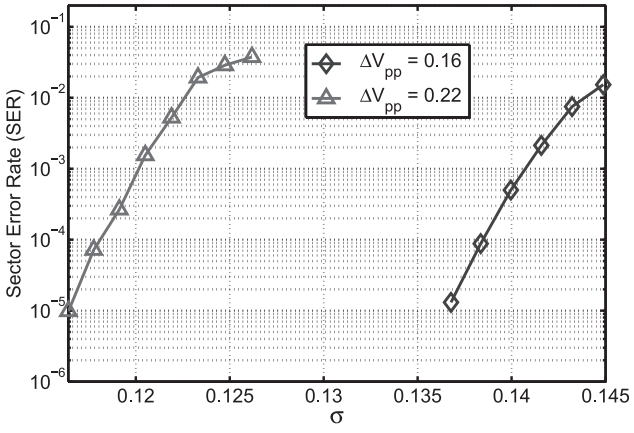


Fig. 11. Simulated NAND flash memory sector error rates when rate-15/16 LDPC code is being used.

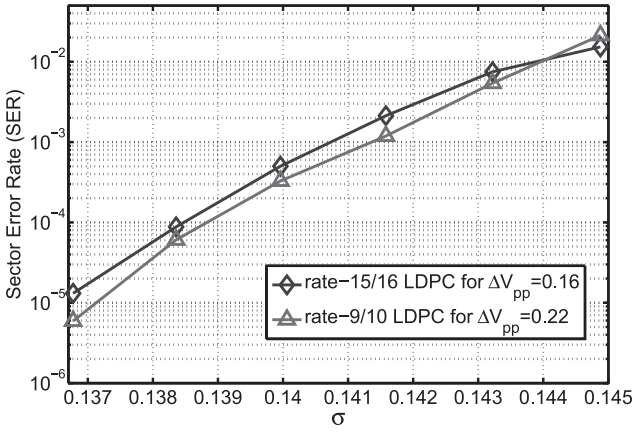


Fig. 12. Simulated NAND flash memory sector error rates.

It is clear that the rate-9/10 LDPC code is certainly not the only choice. Given the baseline LDPC code rate in the normal mode, as we reduce the rate of LDPC code in the high write speed mode, we could more aggressively increase the program step voltage ΔV_{pp} but demand a larger lossless data compression ratio threshold r_t, while a larger r_t may directly reduce the probability β. To demonstrate the involved trade-off, we further considered two other low-rate LDPC codes including 13/14 and 11/12. For each code rate, we constructed a regular QC-LDPC code with column weight of 4 and carried out extensive computer simulations to find a corresponding program step voltage ΔV_{pp} that can ensure almost the same sector error rate performance as the baseline case with rate-15/16 LDPC code and $\Delta V_{pp} = 0.16$. Simulations show that for rate-13/14 and rate-11/12 LDPC codes, choices of ΔV_{pp} as 0.18 and 0.20 can maximize the write speed while ensuring storage reliability. Meanwhile, according to each code rate, we are able to calculate the lossless compression ratio threshold r_t. Again, let β represent the probability that the runtime lossless data compression ratio $r > r_t$. The average NAND flash memory write speed improvement γ can be expressed as

$$\gamma = \left(\frac{\Delta V_{pp}}{0.16} - 1\right) \cdot \beta \times 100\%. \tag{2}$$

Since different environments and workloads have different β, leading to different write speed improvement potentials, we show in Fig. 13 the write speed improvement for different code rate LDPC when β changes from 0 to 1. It can be seen that up to 38 percent write speed improvement can be achieved when each sector of data stored in the flash memory can be compressed by a ratio over r_t. This proposed method can be useful for NAND-flash-based SSD, especially the enterprise-level SSD in data center,

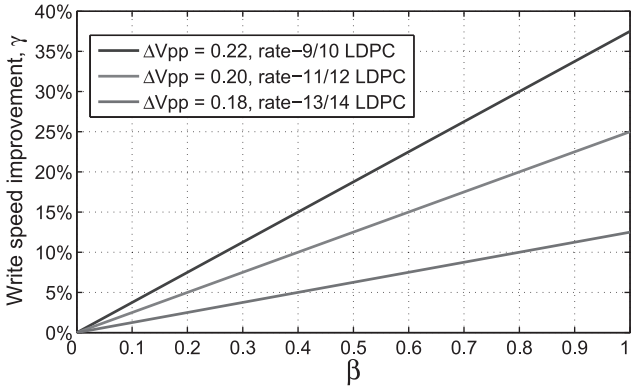


Fig. 13. Write speed improvement in different cases.

where the stored data may be more likely losslessly compressible and at the same time, high write speed is important. Of course, for applications such as MP3/MP4 player, cameras, and media servers, this proposed method will have little performance gain as the media files in those application are losslessly incompressible.

Using the same LZ77 compressor as described in Section 3.3, we carry out compression for various files on the same Linux server. Results show $\beta = 0.9613$, 0.9807, and 0.9991 when rate-9/10, 11/12, and 13/14 LDPC code is used, respectively. It translates to an average NAND flash memory write speed improvement of 36, 24.5, and 12.5 percent, respectively, according to (2). The results are further summarized in Table 4.

5 Conclusions

This paper demonstrates the potential of using lossless data compression to improve other performance metrics of data storage systems, including energy efficiency and access speed, instead of saving storage space. This work has a twofold motivation: 1) lossless data compression is not being widely used today in mass data storage systems such as hard disk drives and NAND flash memories, leaving many losslessly compressible files are stored uncompressed, and 2) data storage is typically realized in the unit of equal-sized sectors, and modern data storage systems employ increasingly powerful ECCs to protect each sector data to ensure the storage integrity, while a stronger ECC needs more storage space to store coding redundancy. The underlying idea of this work is to apply runtime intrasector lossless data compression to enable an opportunistic use of a stronger ECC without incurring any storage space overhead, and trade such more-than-enough error correction capability for other data storage system performance metrics.

To evaluate and demonstrate this design strategy, we carried out case studies for both hard disk drives and NAND flash memory, the two mainstream mass data storage media. In the case studies, we use LDPC codes as the ECC in data storage systems. For hard disk drives, this design strategy is used to reduce average hard disk drive read channel signal processing energy consumption. To sustain continuous hard disk drive storage density growth, ever increasingly powerful and complex read channel signal processing is being used, which nevertheless incur more energy consumption overhead. We carried out detailed read channel ASIC design and power estimation, and results show that up to 38 percent energy saving can be achieved. For NAND flash memory, we use this design strategy to improve the average NAND flash memory write speed in order to alleviate the slow write speed problem of NAND flash memory. Results show that up to 36 percent write speed improvement can be achieved for 2 bits/cell NAND flash memories.

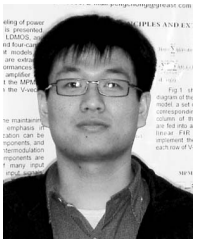
TABLE 4
Results When Different Low-Rate LDPC Codes Are Used

LDPC code rate	13/14	11/12	9/10
ΔV_{pp}	0.18	0.20	0.22
r_t	1.01	1.02	1.04
β	0.9991	0.9807	0.9613
γ	12.5%	24.5%	36%

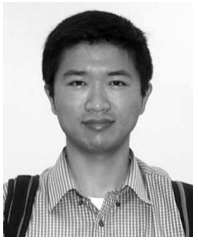
References

[1] K. Sayood, *Introduction to Data Compression,* second ed. Morgan Kaufmann, 2000.
[2] R. Wood, Y. Sonobe, Z. Jin, and B. Wilson, "Perpendicular Recording: The Promise and the Problems," *J. Magnetism and Magnetic Materials,* vol. 235, pp. 1-9, Oct. 2001.
[3] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, "Introduction to Flash Memory," *Proc. IEEE,* vol. 91, no. 4, pp. 489-502, Apr. 2003.
[4] R.G. Gallager, *Low-Density Parity-Check Codes.* MIT Press, 1963.
[5] D.J.C. MacKay, "Good Error-Correcting Codes Based on Very Sparse Matrices," *IEEE Trans. Information Theory,* vol. 45, no. 2, pp. 399-431, Mar. 1999.
[6] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of Capacity-Approaching Low-Density Parity-Check Codes," *IEEE Trans. Information Theory,* vol. 47, no. 2, pp. 619-637, Feb. 2001.
[7] S. Lin and D.J. Costello, *Error Control Coding: Fundamentals and Applications,* second ed. Prentice Hall, 2004.
[8] R.D. Cideciyan, E. Eleftheriou, and T. Mittelholzer, "Perpendicular and Longitudinal Recording: A Signal-Processing and Coding Perspective," *IEEE Trans. Magnetics,* vol. 38, no. 4, pp. 1698-1704, July 2002.
[9] E.F. Haratsch and Z.A. Keirn, "Digital Signal Processing in Read Channels," *Proc. IEEE Custom Integrated Circuits Conf.,* pp. 683-690, Sept. 2005.
[10] T. Hara, "A 146-mm^2 8-Gb Multi-Level NAND Flash Memory with 70-nm CMOS Technology," *IEEE J. Solid-State Circuits,* vol. 41, no. 1, pp. 161-169, Jan. 2006.
[11] Y. Li, "A 16Gb 3b/Cell NAND Flash Memory in 56nm with 8MB/s Write Rate," *Proc. IEEE Int'l Solid-State Circuits Conf. (ISSCC),* pp. 506-632, Feb. 2008.
[12] Y. Li, "A 16 Gb 3-Bit Per Cell (X3) NAND Flash Memory on 56 nm Technology with 8 MB/s Write Rate," *IEEE J. Solid-State Circuits,* vol. 44, no. 1, pp. 195-207, Jan. 2009.
[13] C. Trinh, "A 5.6MB/s 64Gb 4b/Cell NAND Flash Memory in 43nm CMOS," *Proc. IEEE Int'l Solid-State Circuits Conf. (ISSCC),* pp. 246-247, Feb. 2009.
[14] L. Sun, H. Song, and B.V.K.V. Kumar, "Error Floor Investigation and Girth Optimization for Certain Types of Low-Density Parity Check Codes," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing,* pp. 1101-1104, Mar. 2005.
[15] L. Sun, H. Song, B.V.K.V. Kumar, and Z. Keirn, "Field-Programmable Gate-Array-Based Investigation of the Error Floor of Low-Density Parity Check Codes for Magnetic Recording Channels," *IEEE Trans. Magnetics,* vol. 41, no. 10, pp. 2983-2985, Oct. 2005.
[16] X. Hu and B.V.K.V. Kumar, "Evaluation of Low-Density Parity-Check Codes on Perpendicular Magnetic Recording Model," *IEEE Trans. Magnetics,* vol. 43, no. 2, pp. 727-732, Feb. 2007.
[17] S. Jeon, X. Hu, L. Sun, and B.V.K.V. Kumar, "Performance Evaluation of Partial Response Targets for Perpendicular Recording Using Field Programmable Gate Arrays," *IEEE Trans. Magnetics,* vol. 43, no. 6, pp. 2259-2261, June 2007.

[18] X. Hu, B.V.K.V. Kumar, Z. Li, and R. Barndt, "Error Floor Estimation of Long LDPC Codes on Partial Response Channels," *Proc. Global Telecomm. Conf. (GLOBECOM),* pp. 259-264, Nov. 2007.
[19] N. Xie, W. Xu, T. Zhang, E.F. Haratsch, and J. Moon, "Concatenated LDPC and BCH Coding System for Magnetic Recording Read Channel with 4K-Byte Sector Format," *IEEE Trans. Magnetics,* vol. 44, no. 12, pp. 4784-4789, Dec. 2008.
[20] *LDPC Technology Comes to Disk-Read Channels,* http://www.edn.com/article/CA6670958.html, July 2009.
[21] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Trans. Information Theory,* vol. 47, no. 2, pp. 498-519, Feb. 2001.
[22] R. Wood, "The Feasibility of Magnetic Recording at 1 Terabit Per Square Inch," *IEEE Trans. Magnetics,* vol. 36, no. 1, pp. 36-42, Jan. 2000.
[23] E.M. Kurtas, M.F. Erden, and X. Yang, "Future Read Channel Technologies and Challenges for High Density Data Storage Applications," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP),* pp. v/737-v/740, Mar. 2005.
[24] S.I. Iwasaki and Y. Nakamura, "An Analysis for the Magnetization Mode for High Density Magnetic Recording," *IEEE Trans. Magnetics,* vol. MAG-13, no. 5, pp. 1272-1277, Sept. 1977.
[25] W. Cain, A. Payne, M. Baldwinson, and R. Hempstead, "Challenges in the Practical Implementation of Perpendicular Magnetic Recording," *IEEE Trans. Magnetics,* vol. 32, no. 1, pp. 97-102, Jan. 1996.
[26] J. Moon, "SNR Definition for Magnetic Recording Channels with Transition Noise," *IEEE Trans. Magnetics,* vol. 36, no. 5, pp. 3881-3883, Sept. 2000.
[27] R.D. Cideciyan, E. Eleftheriou, and T. Mittelholzer, "Perpendicular and Longitudinal Recording: A Signal-Processing and Coding Perspective," *IEEE Trans. Magnetics,* vol. 38, no. 4, pp. 1698-1704, July 2002.
[28] E. Yeo, "A 500 Mb/s Soft-Output Viterbi Decoder," *IEEE J. Solid-State Circuits,* vol. 38, no. 7, pp. 1234-1241, July 2003.
[29] Z. Li, L. Chen, S. Lin, W. Fong, and P.-S. Yeh, "Efficient Encoding of Quasi-Cyclic Low-Density Parity-Check Codes," *IEEE Trans. Comm.,* vol. 54, no. 1, pp. 71-81, Jan. 2006.
[30] J. Chen, A. Dholakia, E. Eleftheriou, M.P.C. Fossorier, and X.-Y. Hu, "Reduced-Complexity Decoding of LDPC Codes," *IEEE Trans. Comm.,* vol. 53, no. 8, pp. 1288-1299, Aug. 2005.
[31] H. Zhong, W. Xu, N. Xie, and T. Zhang, "Area-Efficient Min-Sum Decoder Design for High-Rate Quasi-Cyclic Low-Density Parity-Check Codes in Magnetic Recording," *IEEE Trans. Magnetics,* vol. 43, no. 12, pp. 4117-4122, Dec. 2007.
[32] J. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression," *IEEE Trans. Information Theory,* vol. IT-23, no. 3, pp. 337-343, May 1977.
[33] L.N. Bairavasundaram, G.R. Goodson, S. Pasupathy, and J. Schindler, "An Analysis of Latent Sector Errors in Disk Drives," *Proc. 2007 ACM SIGMETRICS,* pp. 289-300, 2007.
[34] R. Zeng, "A 172 mm^2 32 Gb MLC NAND Flash Memory in 34nm CMOS," *Proc. IEEE Int'l Solid-State Circuits Conf. (ISSCC),* pp. 236-237, Feb. 2009.
[35] N. Shibata, "A 70nm 16Gb 16-Level-Cell NAND Flash Memory," *IEEE J. Solid-State Circuits,* vol. 43, no. 4, pp. 929-937, Apr. 2008.
[36] K.-T. Park, "A Zeroing Cell-to-Cell Interference Page Architecture with Temporary LSB Storing and Parallel MSB Program Scheme for MLC NAND Flash Memories," *IEEE J. Solid-State Circuits,* vol. 43, no. 4, pp. 919-928, Apr. 2008.
[37] 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.



Ningde Xie received the BS and MS degrees in radio engineering from Southeast University, Nanjing, China, in 2004 and 2006, respectively. He has been working toward the PhD degree at the Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, New York, since 2006. His research interests include VLSI system and architecture design for communication and storage systems. Currently, he is working on high-performance read channels and NAND-based solid-state drives.



Guiqiang Dong received the BS and MS degrees from the University of Science and Technology of China, Hefei, China, in 2004 and 2008, respectively. He is currently working toward the PhD degree at the Department of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, Troy, New York. His research interests include coding theory, signal processing for data storage system, and fault-tolerant system design for digital memory. He is a student member of the IEEE.

Tong Zhang (M'02-SM'08) received the BS and MS degrees in electrical engineering from Xian Jiaotong University, China, in 1995 and 1998, respectively, and the PhD 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, New York. His research activities span over circuits and systems for various data storage and computing applications. He currently serves as an associate editor for the *IEEE Transactions on Circuits and Systems-II* and the *IEEE Transactions on Signal Processing*. He is a senior member of the IEEE.