Read-and-Run Constrained Coding for Modern Flash Memories

Ahmed Hareedy*, Simeng Zheng†, Paul Siegel†, and Robert Calderbank‡
*Electrical and Electronics Engineering Department, Middle East Technical University (METU), 06800 Ankara, Turkey
†Electrical and Computer Engineering Department, University of California, San Diego, La Jolla, CA 92093 USA
‡Electrical and Computer Engineering Department, Duke University, Durham, NC 27708 USA
ahareedy@metu.edu.tr, sizheng@ucsd.edu, psiegel@ucsd.edu, and robert.calderbank@duke.edu

***Abstract*—Constrained coding is used in Flash devices to increase reliability via mitigating inter-cell interference that stems from charge propagation among cells. In this project, we suggest new constrained coding schemes that have low-complexity and preserve the desirable high access speed in modern Flash devices. The idea is to eliminate error-prone patterns by coding data either only on the left-most page (binary coding) or only on the two left-most pages (4-ary coding) while leaving data on all the remaining pages uncoded. Since the proposed schemes enable the separation of pages, except the two left-most pages in the case of 4-ary coding, we refer to them as *read-and-run (RR)* constrained coding schemes. We analyze the new RR coding schemes and discuss their impact on the probability of occurrence of different charge levels. We also demonstrate the performance improvement achieved via RR coding on a 1X-nm triple-level cell (TLC) Flash memory.**

I. Introduction

In Flash devices, charge propagation from cells programmed to high charge levels into cells programmed to lower charge levels is the main reason behind inter-cell interference (ICI). This remains true for any number q of charge levels per cell. Specific data patterns are expected to contribute most to ICI. Mitigating ICI results in remarkable lifetime gains in Flash as demonstrated in [1] for multi-level cell (MLC) Flash ($q = 4$).

In this paper, we propose *read-and-run (RR)* constrained coding schemes that allow pages to be accessed separately in modern Flash devices, thus preserving high access speed. The key idea is that the constrained code is applied only on one or two pages, the left-most page(s), while no coding is applied on the other $\log_2 q - 1$ or $\log_2 q - 2$ pages. This work has been published in [2].

II. Patterns, Mapping, and RR Coding

According to our recent experimental tests and a machine learning-based ICI characterization [3] of TLC Flash memories, we identified a set of ICI-prone patterns. Let

$$\beta_1, \overline{\beta}_1 \in \mathcal{V}_0 \triangleq \left\{ \frac{q}{2}, \frac{q}{2} + 1, \ldots, q - 1 \right\}, \tag{1}$$

where q is the number of levels per Flash cell (a positive power of 2) and $\mathcal{V}_1 = \{0, 1, \ldots, q-1\} \setminus \mathcal{V}_0$[1]. Then, the ICI-prone patterns of interest are the high-low-high level patterns in $\mathcal{L}_q$:

$$\mathcal{L}_q \triangleq \{\beta_1 \mu \overline{\beta}_1, \forall \beta_1, \overline{\beta}_1 \mid 0 \leq \mu < \min(\beta_1, \overline{\beta}_1)\}. \tag{2}$$

A block inside the Flash device can be seen as a 2D grid of wordlines and bitlines, with a cell being placed at each intersection [1]. Level patterns in $\mathcal{L}_q$ are detrimental whether they occur on 3 adjacent cells along the same wordline or along the same bitline. We adopt a recursive alternate Gray mapping (RAGM) [2] to map between data and charge levels.

Now, we are ready to discuss binary coding schemes. From (2) and RAGM, the level patterns in $\mathcal{L}_q$ correspond to binary patterns where the left-most page (pages) always has (have) two 0's separated by some bit, i.e., $0x0$. Based on that, forbidding $\{000, 010\}$ on the left-most page (pages) guarantees that no level pattern in $\mathcal{L}_q$ would appear while writing to a Flash device, with any $q \geq 4$, at least in the wordline (bitline) direction. This corresponds to an interleaved RLL $(d, k) = (0, 1)$ constraint [4]. We emphasize that no coding on any other page is needed. Data will therefore be read from each page independently, and processing can start immediately. RR coding allows low-density parity-check (LDPC) codes on uncoded pages. This idea is the key idea of our RR constrained coding.

RR coding can be performed in the wordline direction only (1D), the bitline direction only (1D), or both directions (2D). For 2D binary RR constrained coding, we want to prevent the patterns in $\{000, 010\}$ from appearing at the left-most pages in both wordline and bitline directions in the Flash device through simple encoding and decoding. One notable remark is that applying 1D RR coding in the wordline direction indirectly reduces the probability of detrimental patterns in the bitline direction, and vice versa.

III. RR-LOCO Coding

We introduce a binary RR coding scheme using lexicographically-ordered constrained (LOCO) codes that forbids $\{000, 010\}$ on the left-most pages in either the wordline direction or the bitline direction. The constrained code we apply is a binary LOCO code devised according to the general method in [5]. The formal definition of this LOCO code $\mathcal{RC}_m^2$ is in [2], and we start by specifying a group structure (partition) for it. The second step is to enumerate the codewords in $\mathcal{RC}_m^2$, which is done by Theorem 1. Let $N_2(m) \triangleq |\mathcal{RC}_m^2|$.

Theorem 1. *The cardinality of a LOCO code $\mathcal{RC}_m^2$ is given by the recursive formula:*

$$N_2(m) = N_2(m-1) + N_2(m-3) + N_2(m-4), \ m \geq 2, \tag{3}$$

where the defined cardinalities are:

$$N_2(-3) \triangleq 0, \ N_2(-2) = N_2(-1) = N_2(0) \triangleq 1, \ \textit{and} \ N_2(1) = 2. \tag{4}$$

Define a codeword $\mathbf{c}$ in $\mathcal{RC}_m^2$ as $\mathbf{c} \triangleq c_{m-1} c_{m-2} \ldots c_0$. The integer equivalent of a LOCO codeword bit c_i, $0 \leq i \leq m-1$,

[1]Levels are defined through their indices $\{0, 1, \ldots, q-1\}$ for simplicity.

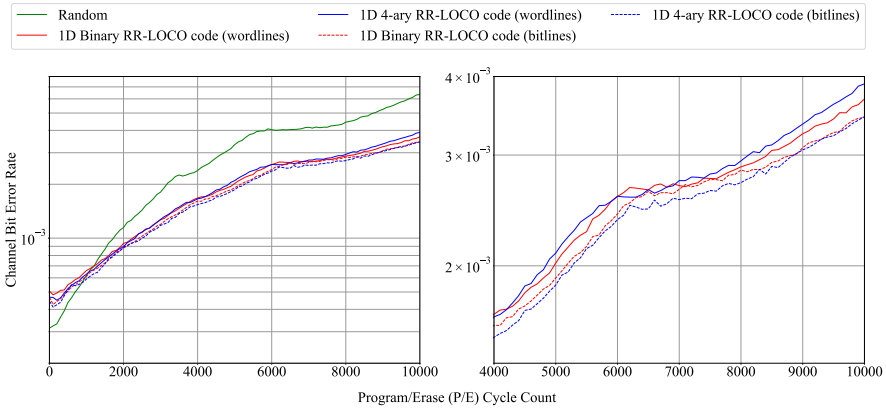

Fig. 1. (Left) Measured average channel bit error rate (BER) comparison when all pages are programmed with random data (green curve), a rate 24:36 1D binary RR-LOCO coded data (red curves) along wordlines (solid curve) or bitlines (dashed curve), and a rate 20:12 bits/symbol 1D 4-ary RR-LOCO coded data (blue curves) along wordlines (solid curve) or bitlines (dashed curve) from P/E cycle 0 to P/E cycle 10,000. (Right) Measured average channel BER excluding random data from P/E cycle 4,000 to P/E cycle 10,000. All coding schemes have overall rate $8/9$. Lifetime gain of RR coding is about 3,700 P/E cycles when BER is 3×10^{-3}.

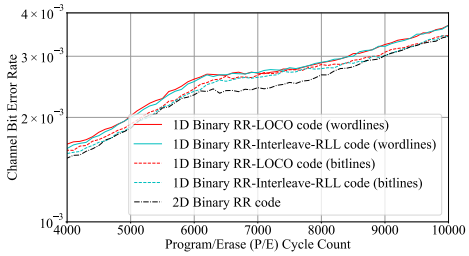

Fig. 2. Measured average channel BER comparison of 1D binary RR-LOCO coded data (red curves) along wordlines (solid curve) or bitlines (dashed curve), 1D binary interleaved rate 12:18 RLL-$(0,1)$ coded data (cyan curves) along wordlines (solid curve) or bitlines (dashed curve), and 2D binary rate 1:2 RR coded data (black curve) from P/E cycle 4,000 to P/E cycle 10,000. 2D RR coding scheme has overall rate $5/6$.

is a_i, i.e., a_i is 0 (1) when c_i is 0 (1). Denote the lexicographic index of a codeword $\mathbf{c}$ among all codewords in the LOCO code $\mathcal{RC}_m^2$ by $g_2(m, \mathbf{c})$, which is abbreviated to $g(\mathbf{c})$. In general, $g(\mathbf{c})$ is in $\{0, 1, \ldots, N_2(m) - 1\}$.

The third step is to specify the special cases of occurrence for a 1 inside a codeword in $\mathcal{RC}_m^2$. The fourth and fifth steps are to find the encoding-decoding rule, which specifies the mapping from index to codeword and vice versa. This rule for $\mathcal{RC}_m^2$ is given in Theorem 2. The sixth step is to assemble the encoding and decoding algorithms.

Theorem 2. *The relation between the lexicographic index $g(\mathbf{c})$, $\mathbf{c} \in \mathcal{RC}_m^2$, and the codeword $\mathbf{c}$ itself is given by:*

$$g(\mathbf{c}) = \sum_{i=0}^{m-1} a_i \Big[(1 - y_{i,1}) N_2(i-2) + (1 - y_{i,1} - y_{i,2}) N_2(i-3) \Big], \quad (5)$$

where $y_{i,1}$ and $y_{i,2}$ are specified as follows:

$$y_{i,1} = 1 \text{ if } c_{i+2}c_{i+1}c_i \in \{001, 011\}, \text{ and } y_{i,1} = 0 \text{ otherwise},$$
$$y_{i,2} = 1 \text{ if } c_{i+1}c_i = 01 \text{ s.t. } y_{i,1} = 0, \text{ and } y_{i,2} = 0 \text{ otherwise}. \quad (6)$$

We next propose a 1D RR coding scheme over GF(4), which is also based on LOCO codes. The goal is to limit the rate loss and/or reduce the code complexity at higher rates compared to binary RR coding by coding on the two left-most pages. Finer classification of error-prone patterns, stemming from characterizing them via two bits instead of one, results in allowing some benign or less detrimental patterns, and therefore increasing the rate with negligible effect on performance. Shorter codewords and smaller encoder-decoder adder sizes can reduce implementation complexity.

The detailed construction, analysis of coding rate, complexity, and error propagation can be found in [2].

IV. Experimental Results on TLC Flash

To characterize the performance of the proposed RR constrained coding schemes, we conducted program/erase (P/E) cycling experiments on several blocks of a 1X-nm TLC Flash chip.

As shown in Fig. 1, the uncoded performance is better than that of both binary and 4-ary RR codes up to around 1,200 P/E cycles and is notably worse thereafter. At the later stages of P/E cycling, ICI becomes severe and RR codes outperform the uncoded setting. As shown in the right subfigure of Fig. 1, the BER of 1D binary RR code along wordlines is almost the same as that of the 4-ary RR code between 2,000 and 8,000 P/E cycles. When the P/E cycle count is larger than 8,000, the BER of 1D binary RR code along wordlines is slightly better than that of the 1D 4-ary RR code.

As shown in Fig. 2, each 1D RR coding scheme along the bitline direction achieves a slightly better channel BER performance than along the wordline direction; the 1D RR coding schemes along the same direction have similar performance; and the performance of the 2D RR constrained code is better than that of the 1D RR codes along any one direction.

References

[1] V. Taranalli *et al.,*"Error analysis and inter-cell interference mitigation in multi-level cell flash memories," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, UK, Jun. 2015, pp. 271–276.

[2] A. Hareedy *et al.,* "Efficient constrained codes that enable page separation in modern flash memories," *IEEE Trans. Commun.*, vol. 71, no. 12, pp. 6834–6848, Dec. 2023.

[3] S. Zheng *et al.,*"Spatio-temporal modeling for flash memory channels using conditional generative nets," *Proc. 2023 Design, Automation & Test in Europe Conf. & Exhib. (DATE)*, Antwerp, Belgium.

[4] P. H. Siegel, "Constrained Codes for Multilevel Flash Memory," presented at *North American School of Information Theory (Padovani Lecture)*, La Jolla, California, Aug. 12, 2015. Available: http://cmrr-star.ucsd.edu/static/presentations/Padovani_Lecture_NASIT_Website.pdf. Video: https://www.youtube.com/watch?v=FCv2PJryUr4.

[5] A. Hareedy *et al.,* "The secret arithmetic of patterns: A general method for designing constrained codes based on lexicographic indexing," *IEEE Trans. Inf. Theory*, vol. 68, no. 9, pp. 5747–5778, Sep. 2022.