Read-and-Run Constrained Coding for Modern Flash Devices

Ahmed Hareedy[*], Simeng Zheng[†], Paul Siegel[†], and Robert Calderbank[*]

[*]Electrical and Computer Engineering Department, Duke University, Durham, NC 27708 USA

[†]Electrical and Computer Engineering Department, University of California, San Diego, La Jolla, CA 92093 USA

ahmed.hareedy@duke.edu, sizheng@ucsd.edu, psiegel@ucsd.edu, and robert.calderbank@duke.edu

Abstract—The pivotal storage density win achieved by solid-state devices over magnetic devices in 2015 is a result of multiple innovations in physics, architecture, and signal processing. One of the most important innovations in that regard is enabling the storage of more than one bit per cell in the Flash device, i.e., having more than two charge levels per cell. Constrained coding is used in Flash devices to increase reliability via mitigating inter-cell interference that stems from charge propagation among cells. Recently, capacity-achieving constrained codes were introduced to serve that purpose in modern Flash devices, which have more than two levels per cell. While these codes result in minimal redundancy via exploiting the underlying physics, they result in non-negligible complexity increase and access speed limitation since pages cannot be read separately. In this paper, 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 only on the left-most page while leaving data on all the remaining pages uncoded. Our coding schemes work for any number of levels per cell, offer systematic encoding and decoding, and are capacity-approaching. Since the proposed schemes enable the separation of pages, we refer to them as *read-and-run (RR)* constrained coding schemes as opposed to schemes adopting *read-and-wait* for other pages. 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 practical triple-level cell Flash device.

I. INTRODUCTION

The history of constrained coding dates back to 1948, when Shannon represented a constrained sequence via a finite-state transition diagram (FSTD) and derived the capacity under a constraint [1]. Run-length-limited (RLL) codes were introduced by Tang and Bahl in 1970 to support the evolution of magnetic recording at that time [2], and these codes were based on lexicographic indexing. In 1973, Cover presented a result about enumerative coding [3] that will prove fundamental for the design of constrained codes based on lexicographic indexing decades later. Among other researchers, Franaszek developed constrained codes based on finite-state machines (FSMs) derived from FSTDs [4]. In 1983, Adler, Coppersmith, and Hassner introduced a systematic method to develop constrained codes based on FSMs [5]. Details about the history of constrained coding until 1998 are in [6].

Because of their ability to improve performance via eliminating error-prone data patterns and undesirable sequences, constrained codes have a plethora of applications. They find application in one-dimensional (1D) magnetic recording devices, both the old ones, which are based on peak detection, and the modern ones, which are based on sequence detection

[7], [8]. They can also be combined with robust signal detection using machine learning [9]. They find application in the emerging two-dimensional (2D) magnetic recording devices as well [10], [11]. Moreover, constrained codes are used to achieve DC balance and self-calibration in optical recording devices [12] in addition to many computer standards for data transmission [13].

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) [14]. This is correct for any number q of charge levels per cell. Mitigating ICI results in remarkable lifetime gains in Flash as demonstrated in [15] for multi-level cell (MLC) Flash ($q = 4$). There are data patterns that are considered usual suspects for contributing most to ICI. Coding to eliminate data patterns resulting in consecutive levels $(q-1)0(q-1)$ was considered in [16] and [17]. Coding to eliminate data patterns resulting in consecutive levels or level patterns $(q-1)\mu(q-1)$, for all $\mu < q-1$, was presented in [15], [17], and [18].

A number of recent results revisited [2] and [3] in order to produce efficient constrained codes based on lexicographic indexing, and one example is [19]. Another example is [7], in which we introduced binary symmetric lexicographically-ordered constrained (S-LOCO) codes and demonstrated density gains in a modern magnetic recording system. We extended our result to single-level cell (SLC) Flash ($q = 2$) [20] then to Flash with any number q of levels per cell [18]. Moreover, we devised a general method to design LOCO codes for any finite set of patterns to forbid [21], which will be useful in this paper. We studied the power spectra of binary LOCO codes in [22]. LOCO codes are capacity-achieving, simple, and easily reconfigurable [18], [21].

While the constrained codes in [17] and [18] are quite efficient in terms of rate, they require all Flash pages to be processed together, which negatively affects the access speed. 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. There are techniques in the literature that allow page separation; however, they are either incurring notable rate loss [15] or designed for a specific Flash setup [16]. Our RR coding schemes incur minor rate loss and work for any Flash device. The key idea is that the constrained code is applied only on one page, while no coding is applied on the other $\log_2 q - 1$ pages. We present a 2D RR coding scheme as well as a 1D RR coding scheme that is based on LOCO codes. We study various aspects about these

schemes, including the charge-level probabilities. We introduce experimental results in a practical triple-level cell (TLC) Flash device ($q = 8$) that demonstrate notable lifetime gains achieved by our coding schemes.

The rest of the paper is organized as follows. In Section II, we discuss the detrimental patterns, the Flash mapping, and our 2D RR coding scheme. In Section III, we introduce our 1D RR-LOCO coding scheme. In Section IV, we study the rate, complexity, and error propagation of the new schemes. In Section V, we present the experimental results on TLC Flash. In Section VI, we conclude the paper.

II. PATTERNS, MAPPING, AND 2D RR CODING

As implied in the introduction, literature works do not strictly agree on the set of forbidden patterns to operate on. Additionally, as the Flash device ages, the set of error-prone patterns is expected to expand [18]. Based on our recent experimental tests on a practical TLC Flash device, we decided to focus on the set characterized as follows. Let

$$\beta_1, \beta_2 \in \mathcal{V}_0 \triangleq \left\{ \frac{q}{2}, \frac{q}{2} + 1, \ldots, q - 1 \right\}, \tag{1}$$

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

$$\mathcal{L}_q \triangleq \{\beta_1 \mu \beta_2, \forall \beta_1, \beta_2 \mid 0 \leq \mu < \min(\beta_1, \beta_2)\}. \tag{2}$$

This set already subsumes all 3-tuple forbidden patterns adopted in the literature for Flash. 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 [15]. Level patterns in $\mathcal{L}_q$ are detrimental whether they occur on 3 adjacent cells along the same wordline or along the same bitline.

Example 1. *Consider an MLC Flash device, i.e., $q = 4$. In this case, we have $\beta_1, \beta_2 \in \{2, 3\}$. Then, the set of interest is the set resulting in:*

$$\mathcal{L}_4 = \{202, 212, 203, 213, 302, 312, 303, 313, 323\}. \tag{3}$$

The last three elements in $\mathcal{L}_4$ are quite known [15], [16], [18].

Algorithm 1 Recursive Alternate Gray Mapping

1: **Input:** Number of levels per cell q, and $p = \log_2 q$.
2: Define map, a binary array of dimensions $q \times p$.
3: Set $\text{map}(0, :) = \mathbf{1}^p$. *(a sequence of p 1's)*
4: **for** $i \in \{0, 1, \ldots, p - 1\}$ **do**
5:     **for** $j \in \{0, 1, \ldots, 2^i - 1\}$ **do**
6:         $\text{map}(2^i + j, :) = \text{map}(2^i - 1 - j, :)$.
7:         Flip the bit $\text{map}(2^i + j, i)$. *(each sequence in map is indexed from right to left by $0, 1, \ldots, p - 1$)*
8:     **end for**
9: **end for**
10: **Output:** Array map that maps each index to binary data.

Next, we discuss how to map from data to charge levels in Flash and vice versa. Since we are interested in page separation throughout this work, the mapping here is from a charge level

out of q possible ones to $\log_2 q$ binary bits, one for each page, and vice versa. Gray mapping offers the advantage that there is only one-bit difference between any two adjacent levels, which is valuable for error performance. We adopt a recursive alternate Gray mapping (RAGM), and Algorithm 1 shows how to produce it for any q. We highlight that RAGM has already been used in the literature in MLC Flash [15] and TLC Flash [16]. Thus, RAGM is not strictly a new contribution.

Example 2. *Consider a TLC Flash device, i.e., $q = 8$. In this case, the output of Algorithm 1, which is RAGM, becomes:*

$$\begin{array}{ll} 0 \longleftrightarrow 111, & 1 \longleftrightarrow 110, \\ 2 \longleftrightarrow 100, & 3 \longleftrightarrow 101, \\ 4 \longleftrightarrow 001, & 5 \longleftrightarrow 000, \\ 6 \longleftrightarrow 010, & 7 \longleftrightarrow 011. \end{array} \tag{4}$$

Now, we are ready to discuss coding schemes. Let us first index the Flash pages the same way the bits in each sequence in the array map are indexed (see Algorithm 1). This means that the left-most page is the one indexed by $p - 1$. From (2) and Algorithm 1, the level patterns in $\mathcal{L}_q$ correspond to binary patterns where the left-most page(s) always has (have) two 0's separated by some bit, i.e., $0x0$. Based on that, forbidding $\{000, 010\}$ on the left-most page(s) guarantees that no level pattern in $\mathcal{L}_q$ would appear while writing to a Flash device, with any $q > 2$, at least in one direction. This corresponds to an interleaved RLL $(d, k) = (0, 1)$ constraint [23]. Notably, no coding on any other page is needed. Data will therefore be read from each page independently, and immediately passed to the low-density parity-check (LDPC) decoder. This idea is the key idea of our RR constrained coding schemes.[2]

RR coding can be performed in the wordline direction only (1D), the bitline direction only (1D), or both directions (2D). Observe that such coding will also prevent benign level patterns, e.g., 555 and 676 in TLC Flash, resulting in inevitable rate loss. However, as we shall see in Section IV, this rate loss is small, and RR-LOCO codes are capacity-approaching.

We start here with our scheme for 2D RR constrained coding. As the name suggests, we want to prevent the patterns in $\{000, 010\}$ from appearing at the left-most pages in both wordline and bitline directions through simple encoding and decoding. The encoding follows the rules:

1) On wordlines with indices congruent to 0 or 1 (mod 4), you are allowed to write 0's and 1's freely in bit positions congruent to 0 or 1 (mod 4) at the left-most pages.
2) On wordlines with indices congruent to 2 or 3 (mod 4), you are allowed to write 0's and 1's freely in bit positions congruent to 2 or 3 (mod 4) at the left-most pages.
3) In the other bit positions, you can only write 1's on wordlines at the left-most pages.

This 2D RR constrained coding scheme is depicted in Fig. 1. It is clear from the figure that the patterns in $\{000, 010\}$ are eliminated from the left-most pages, which forbids all level patterns in $\mathcal{L}_q$, in both directions. Upon encoding, input data bits are freely placed at the positions marked by x and directly at the other pages. Upon decoding, information at the positions

---

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

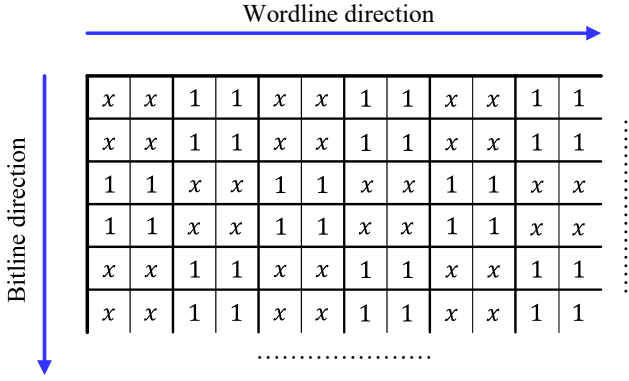[2]An equivalent scheme was proposed for MLC Flash, i.e., $q = 4$, in [23].

Fig. 1. The left-most pages of a 2D Flash grid with data encoded via the proposed 2D RR coding scheme. Symbol x means bit can be 0 or 1 freely.

marked by 1 is omitted, and data bits at the remaining positions are read with no additional processing and with no correlation between different Flash pages.[3]

This 2D scheme is ideal in terms of complexity, access speed, and error propagation (see Section IV). It might also seem notably better than any 1D scheme in terms of performance. However, as we shall justify later, 1D schemes can achieve almost the same performance with higher rates.

III. RR-LOCO CODING OVER GF(2)

In this section, we introduce an RR coding scheme that forbids $\{000, 010\}$ on the left-most pages in either the wordline or the bitline direction, while leaving all other pages with no coding, which forbids the level patterns in $\mathcal{L}_q$ and achieves page separation. The code we apply is a binary LOCO code devised according to the general method in [21]. We start by defining the LOCO code.

Definition 1. *A LOCO code $\mathcal{RC}_m$, where $m \geq 1$, that forbids $\{000, 010\}$ is defined by the following properties:*

1) *Codewords in $\mathcal{RC}_m$ are defined over GF(2) = $\{0, 1\}$ and are of length m bits.*
2) *Codewords in $\mathcal{RC}_m$ are ordered lexicographically.*
3) *Codewords in $\mathcal{RC}_m$ do not have patterns in $\{000, 010\}$.*
4) *All codewords satisfying 1)–3) are included.*

Lexicographic ordering is ordering codewords ascendingly according to the rule "$0 < 1$", where bit significance reduces from left to right [2], [18]. The first step to devise the LOCO code is to specify the group structure. Codewords in $\mathcal{RC}_m$, $m \geq 2$, can be partitioned into the following groups:

- Group 1: Codewords starting with 0011 from the left.
- Group 2: Codewords starting with 011 from the left.
- Group 3: Codewords starting with 1 from the left.

The second step is to enumerate the codewords, which is done by Theorem 1. Let $N(m) \triangleq |\mathcal{RC}_m|$.

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

$$N(m) = N(m-1) + N(m-3) + N(m-4), \ m \geq 2, \ (5)$$

[3]An equivalent 2D scheme forbidding patterns $\{101, 111\}$ on the right-most pages in both worline and bitline directions in MLC Flash was proposed in [23].

where the defined cardinalities and $N(1)$ are:

$$N(-2) = N(-1) = N(0) \triangleq 1 \ and \ N(1) = 2. \quad (6)$$

Proof: We compute the cardinalities of each group then add them all. Let the cardinality of Group i be N_i. As for Group 3 in $\mathcal{RC}_m$, there is a bijection between its codewords and the codewords in $\mathcal{RC}_{m-1}$ (attach 1). Thus,

$$N_3(m) = N(m-1). \quad (7)$$

As for Group 2 in $\mathcal{RC}_m$, there is a bijection between its codewords and the codewords starting with 1 from the left in $\mathcal{RC}_{m-2}$ (attach 01). Thus using (7),

$$N_2(m) = N_3(m-2) = N(m-3). \quad (8)$$

As for Group 1 in $\mathcal{RC}_m$, there is a bijection between its codewords and the codewords starting with 1 from the left in $\mathcal{RC}_{m-3}$ (attach 001). Thus using (7),

$$N_1(m) = N_3(m-3) = N(m-4). \quad (9)$$

Adding (7), (8), and (9) gives (5). The defined cardinalities can be computed by observing that $N(1) = 2$, $N(2) = 4$, and $N(3) = 6$, which sets up three equations. This observation is immediate given the forbidden patterns. ∎

Define a codeword $\mathbf{c}$ in $\mathcal{RC}_m$ as $\mathbf{c} \triangleq c_{m-1}c_{m-2}\ldots c_0$, with $c_i \triangleq \zeta$ for $i \geq m$, where ζ represents out of codeword bounds. The integer equivalent of a LOCO codeword bit c_i, $0 \leq i \leq m-1$, 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 $\mathcal{RC}_m$ by $g(\mathbf{c})$. In general, $g(\mathbf{c})$ is in $\{0, 1, \ldots, N(m) - 1\}$.

The third step is to specify the special cases of occurence for a 1 inside a codeword in $\mathcal{RC}_m$. These cases are:

- Case 1: $c_{i+2}c_{i+1}c_i = 001$.
- Case 2: $c_{i+2}c_{i+1}c_i = 011$.
- Case 3: $c_{i+2}c_{i+1}c_i = 101$ or $c_{i+2}c_{i+1}c_i = \zeta 01$.

The typical or default case is simply the case of "otherwise". In particular, it is the case that $c_{i+2}c_{i+1}c_i = 111$, $c_{i+2}c_{i+1}c_i = \zeta 11$, or $c_{i+1}c_i = \zeta 1$.

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 $\mathcal{RC}_m$ is given in Theorem 2.

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

$$g(\mathbf{c}) = \sum_{i=0}^{m-1} a_i \Big[(1 - y_{i,1})N(i-2) + (1 - y_{i,1} - y_{i,2})N(i-3) \Big], \quad (10)$$

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

$y_{i,1} = 1$ *if* $c_{i+2}c_{i+1}c_i \in \{001, 011\}$, *and* $y_{i,1} = 0$ *otherwise,*
$y_{i,2} = 1$ *if* $c_{i+2}c_{i+1}c_i \in \{101, \zeta 01\}$, *and* $y_{i,2} = 0$ *otherwise.* (11)

Proof: We compute the contributions $g_{i,j}(c_i)$ of a bit c_i under Case j, for all $j \in \{0, 1, 2, 3\}$, in a LOCO codeword then merge them all. As for the typical case, which we index

by 0, this contribution is the number of codewords starting with 0 from the left in $\mathcal{RC}_{i+1}$. Thus using (8) and (9),

$$g_{i,0}(c_i) = N_2(i+1) + N_1(i+1)$$
$$= N(i-2) + N(i-3). \quad (12)$$

As for Case 1 (Case 2), this contribution is the number of codewords starting with 000 (010) from the left in $\mathcal{RC}_{i+3}$. Note that 000 and 010 are forbidden patterns. Thus,

$$g_{i,1}(c_i) = 0 \text{ and}$$
$$g_{i,2}(c_i) = 0. \quad (13)$$

As for Case 3, this contribution is the number of codewords starting with 00 from the left in $\mathcal{RC}_{i+2}$. Thus using (9),

$$g_{i,3}(c_i) = N_1(i+2) = N(i-2). \quad (14)$$

Using $y_{i,1}$ and $y_{i,2}$ from (11) along with a_i to merge (12), (13), and (14) gives:

$$g_i(c_i) = a_i\Big[(1-y_{i,1})N(i-2)+(1-y_{i,1}-y_{i,2})N(i-3)\Big]. \quad (15)$$

Substituting (15) in $g(\mathbf{c}) = \sum_{i=0}^{m-1} g_i(c_i)$ gives (10). ∎

For brevity, we skip the sixth step, which is to assemble the encoding and decoding algorithms. These algorithms are a direct consequence of the rule in (10), and we refer the reader to [2], [18], [21], and [24] for details. Note that we sometimes refer to $\mathcal{RC}_m$ as a *1D RR-LOCO code*. The encoding-decoding rule of a LOCO code is the reason behind its low complexity algorithms, where reconfiguration becomes as easy as reprogramming an adder [7], [21].

Remark 1. *If the coded bits are complemented before writing to pages, the set of forbidden patterns on the left-most pages becomes $\{101, 111\}$ instead, which appears in [15] as well. In this case, the cardinality of the LOCO code remains as in (5), while the encoding-decoding rule becomes exactly that of an asymmetric LOCO code in [20] for $x = 1$:*

$$g(\mathbf{c}) = \sum_{i=0}^{m-1} a_i N(i - a_{i+1}). \quad (16)$$

Encoding and decoding on the left-most pages are just subtractions and additions. As for the remaining pages, data is written and read directly. This guarantees simplicity and maintains high access speed via our 1D RR-LOCO coding scheme.

IV. RATE, COMPLEXITY, AND ERROR PROPAGATION

We start by calculating asymptotic rates. Unfortunately, deriving the capacity for 2D constrained codes is known to be notoriously hard. Therefore, we will derive the capacity $C_{\mathcal{L}_q}^{1D}$ only under the 1D constrained coding setup, which is already higher than the capacity under the 2D setup. Thus, $C_{\mathcal{L}_q}^{1D}$ serves as a ceiling for the highest achievable rate in a device where patterns in $\mathcal{L}_q$ are forbidden at least in one direction. We will shortly show that 1D constrained coding suffices.

An FSTD of a sequence where level patterns in $\mathcal{L}_q$ are forbidden is shown in Fig. 2. Based on this FSTD, the general adjacency matrix is:
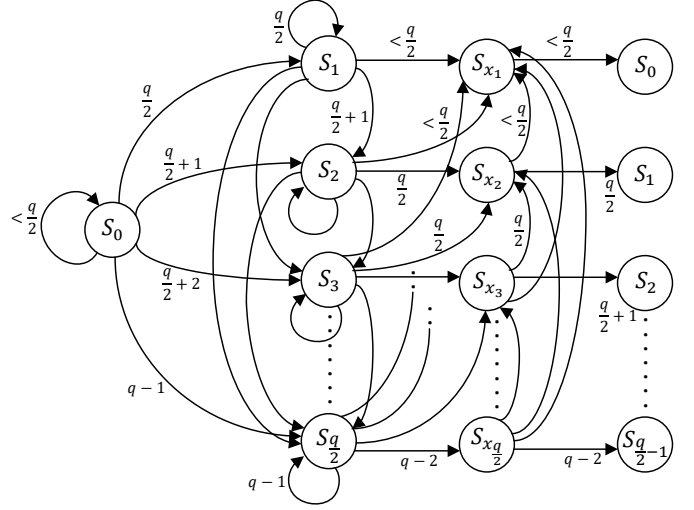

Fig. 2. An FSTD of a constrained sequence forbidding level patterns in $\mathcal{L}_q$, for any q. We operate directly on level patterns for simplicity.

TABLE I
CAPACITY GAP BETWEEN $C_{\mathcal{L}_q}^{1D}$ AND 1D RR-LOCO CAPACITY C_{RR}^{1D}

q	$C_{\mathcal{L}_q}^{1D}$	C_{RR}^{1D}	Capacity gap %
4	0.8941	0.8471	5.257%
8	0.9235	0.8981	2.750%
16	0.9401	0.9235	1.766%

$$\mathbf{A}_1 = \left[\begin{array}{c|c|c|c} \frac{q}{2} & \mathbf{1}_{\frac{q}{2}}^{\mathrm{T}} & 0 & \mathbf{0}_{\frac{q}{2}-1}^{\mathrm{T}} \\ \hline \mathbf{0}_{\frac{q}{2}} & \mathbf{U}_{\frac{q}{2}}^1 & \frac{q}{2}\mathbf{1}_{\frac{q}{2}} & \begin{array}{c} \mathbf{0}_{\frac{q}{2}-1}^{\mathrm{T}} \\ \hline \mathbf{L}_{\frac{q}{2}-1}^1 \end{array} \\ \hline \frac{q}{2} & \mathbf{0}_{\frac{q}{2}}^{\mathrm{T}} & 0 & \mathbf{0}_{\frac{q}{2}-1}^{\mathrm{T}} \\ \hline \mathbf{0}_{\frac{q}{2}-1} & \mathbf{I}_{\frac{q}{2}-1} \mid \mathbf{0}_{\frac{q}{2}-1} & \frac{q}{2}\mathbf{1}_{\frac{q}{2}-1} & \begin{array}{c|c} \mathbf{0}_{\frac{q}{2}-1}^{\mathrm{T}} \\ \hline \mathbf{L}_{\frac{q}{2}-2}^1 & \mathbf{0}_{\frac{q}{2}-2} \end{array} \end{array}\right]. \quad (17)$$

Thus and from [1], the normalized capacity of a 1D constrained code forbidding the level patterns in $\mathcal{L}_q$ is:

$$C_{\mathcal{L}_q}^{1D} = \frac{\log_2(\lambda_{\max}(\mathbf{A}_1))}{\log_2 q}, \quad (18)$$

where $\lambda_{\max}(\mathbf{A})$ is the maximum real positive eigenvalue of the matrix $\mathbf{A}$.[4]

The capacity of a 2D code preventing $\{000, 010\}$ is the capacity of a 2D $(0,1)$ RLL code, which is ≈ 0.5879 [26]. Thus, the normalized capacity of our 2D RR coding scheme is:

$$C_{RR}^{2D} \approx \frac{0.5879 + \log_2 q - 1}{\log_2 q} = \frac{\log_2 q - 0.4121}{\log_2 q}. \quad (19)$$

As mentioned above, the constrained system where patterns in $\{000, 010\}$ are forbidden can be interpreted as an interleaved RLL $(d,k) = (0,1)$ constraint, whose capacity is known to be $\log_2((1+\sqrt{5})/2) \approx 0.6942$. Thus, the normalized capacity of our 1D RR-LOCO coding scheme is

---

[4]For positive integers $a + b \leq q$, the set H of the a largest symbols and the set L of the b smallest symbols in $\{0, 1, \ldots, q-1\}$, a formula for the (count-constrained) capacity of the constrained system forbidding patterns $\{\beta_1\beta_2\beta_3 \mid \beta_1, \beta_3 \in H, \beta_2 \in L\}$ was derived in [25].

$$C_{\text{RR}}^{\text{1D}} = \frac{\log_2((1+\sqrt{5})/2) + \log_2 q - 1}{\log_2 q} \approx \frac{\log_2 q - 0.3058}{\log_2 q}. \tag{20}$$

The capacity gap between $C_{\mathcal{L}_q}^{\text{1D}}$ and $C_{\text{RR}}^{\text{1D}}$ for different values of q is given in Table I. The table shows that the capacity gap is small, and it gets even smaller as q increases.

Next, we discuss the finite-length rates. First, the normalized rate of our 2D RR constrained coding scheme is:

$$R_{\text{RR}}^{\text{2D}} = \frac{0.5 + \log_2 q - 1}{\log_2 q} = \frac{\log_2 - 0.5}{\log_2 q} \tag{21}$$

since the rate of our left-most page coding is 0.5.

Regarding our 1D RR-LOCO coding scheme, we bridge with the pattern 11 between consecutive codewords in $\mathcal{RC}_m$ on the left-most page, and we remove the codeword $\mathbf{1}^m$ for self-clocking [18], [21]. Thus, the rate on the left-most page is $\lfloor \log_2(N(m) - 1) \rfloor / (m+2)$, and the normalized rate of our 1D RR-LOCO coding scheme is:

$$R_{\text{RR}}^{\text{1D}} = \frac{1}{\log_2 q} \left[\frac{\lfloor \log_2(N(m) - 1) \rfloor}{m+2} + \log_2 q - 1 \right]. \tag{22}$$

1D RR-LOCO coding schemes are capacity-achieving schemes in the sense that the limit as $m \to \infty$ of $R_{\text{RR}}^{\text{1D}}$ is $C_{\text{RR}}^{\text{1D}}$ (see also [18]). Another capacity-achieving 1D RR constrained coding scheme, implementable using enumerative coding without the need for bridging bits, can be obtained by interleaving codewords from an optimal block code for the RLL $(d, k) = (0, 1)$ constraint [27] on the left-most pages. LOCO codes, however, offer simplicity and reconfigurability, which is important as the device ages [18].

The 2D RR constrained coding scheme we propose requires no additional complexity for encoding and decoding since data is written/read directly to/from pages. As for the 1D RR-LOCO coding scheme, the complexity is governed by the size of the adder that executes the encoding-decoding rule, which is:

$$s = \lfloor \log_2(N(m) - 1) \rfloor \tag{23}$$

bits. For ease of implementation and to avoid affecting the access speed, we prefer to apply the 1D RR-LOCO coding scheme along wordlines instead of bitlines since the performance is almost the same, as demonstrated by the experimental results in Section V.

Similarly, the 2D RR coding scheme does not incur any error propagation. Thus, the error propagation factor of it is $E_{\text{RR}}^{\text{2D}} = 1$. As for the 1D RR-LOCO coding scheme, there is no codeword-to-codeword error propagation. However, there exists limited error propagation resulting from the codeword-to-message conversion [7], [18] on the left-most page only. This error propagation reaches $\frac{s}{2}$ bits on average, where s is the message length as well from (23). Consequently, the error propagation factor averaged over $\log_2 q$ pages is:

$$E_{\text{RR}}^{\text{1D}} = \frac{1}{\log_2 q} \left[\frac{s}{2} + \log_2 q - 1 \right]. \tag{24}$$

Table II gives the rates, adder sizes, and error propagation factors of the proposed RR schemes under various parameters. The 1D RR-LOCO coding scheme has a remarkable rate advantage that reaches 10.147%, 6.096%, and 4.343% for $q = 4$, $q = 8$, and $q = 16$, respectively over the 2D RR constrained

TABLE II
RATE, COMPLEXITY, AND ERROR PROPAGATION COMPARISONS BETWEEN
2D AND 1D RR CONSTRAINED CODING

q	m	$R_{\text{RR}}^{\text{2D}}$	$R_{\text{RR}}^{\text{1D}}$	s	$E_{\text{RR}}^{\text{2D}}$	$E_{\text{RR}}^{\text{1D}}$
4	7	0.7500	0.7778	5	1.000	1.750
4	11	0.7500	0.8077	8	1.000	2.500
4	21	0.7500	0.8261	15	1.000	4.250
8	7	0.8333	0.8519	5	1.000	1.500
8	11	0.8333	0.8718	8	1.000	2.000
8	21	0.8333	0.8841	15	1.000	3.167
16	7	0.8750	0.8889	5	1.000	1.375
16	11	0.8750	0.9038	8	1.000	1.750
16	21	0.8750	0.9130	15	1.000	2.625

coding scheme. The 2D RR scheme has a clear advantage in terms of both complexity and error propagation as it requires no processing to encode and decode. Having said that, the error propagation factor of the 1D RR scheme decreases notably as q increases. For example, $E_{\text{RR}}^{\text{1D}} = 2.625$ for $q = 16$ and $m = 21$, which is remarkably small given the code length.

The two coding schemes can be used in the same device, but at different lifetime stages. The 1D RR-LOCO coding scheme can be used when the device is still fresh or until a moderate number of program/erase (P/E) cycles, while the 2D RR constrained coding scheme can be used when the device ages, where preventing the error-prone patterns in both directions could make a difference and the rate loss could be acceptable. However, this performance difference is shown to be small in Section V, at least for the TLC Flash device we used.

Remark 2. *An idea that allows page separation for MLC Flash was introduced in [15]. However, the rate offered is only* 0.7500*, which is significantly below the rates offered via our 1D RR coding scheme for MLC. Another idea that allows page separation for TLC Flash was introduced in [16]. However, it only heuristically addresses the level pattern* 707.

V. EXPERIMENTAL RESULTS ON TLC FLASH

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

1) Erase Flash memory block under test.
2) Program all pages of block under test with data. For uncoded experiments, program pseudo-random data at each P/E cycle. For RR experiments, program prepared data satisfying RR constraints at each P/E cycle.
3) For each successive P/E cycle of RR experiments, "rotate" the data, so the data that was written on the page i is written on the page $(i+1)$, wrapping around the last page to the first page.
4) Record bit errors and compute channel bit error rate (BER) every 100 P/E cycles.

Remark 3. *Gray mappings used in Flash devices may vary between manufacturers and product generations. The forbidden patterns can usually be modified in accordance with the mapping so that RR coding on one page per wordline will eliminate all or most of the patterns in $\mathcal{L}_q$ that induce the most severe ICI.*

Fig. 3 presents the channel BER from P/E cycle 0 to P/E cycle 10,000 using pseudo-random data and a rate 24:36 1D RR-LOCO code either along wordlines or along bitlines. Using (22), $R_{RR}^{1D} = 0.8889$. Therefore, the 1D coding scheme achieves about 99% (96%) of the capacity C_{RR}^{1D} ($C_{\mathcal{L}_q}^{1D}$). The uncoded performance is better than that of the RR codes up to around 1,800 P/E cycles and is notably worse thereafter.

At the later stages of P/E cycling, ICI increases, and the results in Fig. 3 reflect this phenomenon. Specifically, RR-LOCO codes along wordlines increase device lifetime by about 1,200 P/E cycles when channel BER is 2×10^{-3}, i.e., 37% lifetime gain, and achieve about 2,600 P/E cycles gain when channel BER is 3×10^{-3}, i.e., 58% lifetime gain. After 2,000 P/E cycles, the BER of 1D RR coding is almost the same on wordlines as it is on bitlines.

Fig. 4 compares the BER performance of the 24:36 1D RR-LOCO code with the interleaved 12:18 RLL $(d, k) = (0, 1)$ code (which has an overall block length 36 after interleaving) and the 2D RR code. Using (21), $R_{RR}^{2D} = 0.8333$. Therefore, the 2D coding scheme achieves about 93% (90%) of the capacity C_{RR}^{2D} ($C_{\mathcal{L}_q}^{1D}$). The two 1D RR coding schemes have similar performance, with both showing that bitline coding performs almost the same as wordline coding. The 2D RR coding scheme achieves a slightly better performance than the two 1D RR coding schemes.

An examination of level probabilities induced by 1D RR constraints builds intuition towards the experimental results in Figs. 3 and 4. The probabilities of binary symbols 0 and 1 under the RLL $(d, k) = (0, 1)$ constraint are approximately 0.2764 and 0.7236, respectively [23]. Asymptotically, this leads to probabilities of individual symbols in $\mathcal{V}_0$ and $\mathcal{V}_1$ of about 0.0691 and 0.1809, respectively.

The cross-over behavior observed in Fig. 3 can be explained if the level patterns eliminated by the code, especially ICI-prone patterns, are not the only significant contributors to error early in the device lifetime. The RR coding significantly changes level probabilities compared with the uncoded setting, increasing the probability of some of the remaining level patterns that cause errors due to other effects, and accordingly increasing their contribution to the BER at low P/E cycles. One suggestion to prevent this behavior is applying two different constraints before and after the cross-over point, making use of the LOCO reconfigurability feature that could be directed by a machine learning module.

Similarly, 1D RR coding in the wordline direction will reduce the probabilities of detrimental patterns in the bitline direction, and vice versa. This reduces the impact of the more severe ICI in the bitline direction on the overall error rate, while simultaneously reducing the expected advantage of the 2D RR coding (even without taking into account the rate penalty associated with the 2D coding).

These effects on level-pattern probabilities have been confirmed by examination of the data written to the Flash memory block and the observed error-inducing patterns.

VI. CONCLUSION

We introduced 2D and 1D RR coding schemes for modern Flash devices. RR coding schemes are systematic, and they
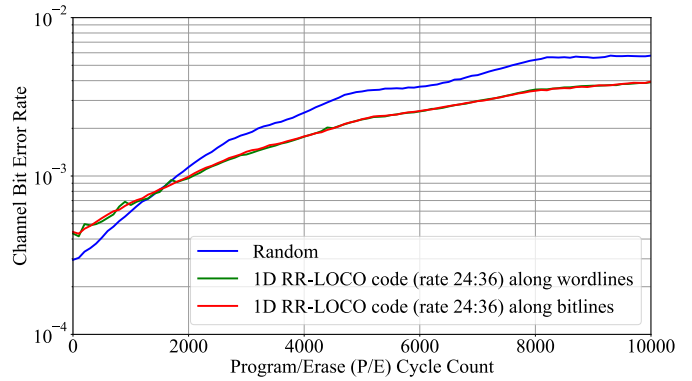

Fig. 3. Measured average channel BER comparison when all pages are programmed with random data, 1D RR-LOCO coded data (rate 24:36) along wordlines or bitlines.
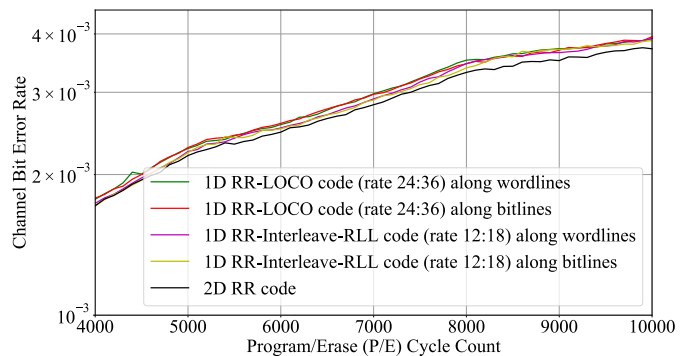

Fig. 4. Measured average channel BER comparison of 1D RR-LOCO codes (rate 24:36) along wordlines or bitlines, 1D interleaved RLL-(0, 1) codes (rate 12:18) along wordlines or bitlines, and 2D RR code.

incur limited redundancy to improve performance. Experimental results reveal significant P/E cycles gains in a commercial Flash device. In summary, RR codes offer an efficient and practical approach to mitigating ICI that can enhance Flash device lifetime. Future work includes combining RR constrained codes with effective LDPC codes [28].

REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication," Bell Sys. Tech. J., vol. 27, Oct. 1948.
[2] D. T. Tang and R. L. Bahl, "Block codes for a class of constrained noiseless channels," *Inf. and Control*, vol. 17, no. 5, pp. 436–461, 1970.
[3] T. Cover, "Enumerative source encoding," *IEEE Trans. Inf. Theory*, vol. 19, no. 1, pp. 73–77, Jan. 1973.
[4] P. A. Franaszek, "Sequence-state methods for run-length-limited coding," *IBM J. Res. Dev.*, vol. 14, no. 4, pp. 376–383, Jul. 1970.
[5] R. Adler, D. Coppersmith, and M. Hassner, "Algorithms for sliding block codes–An application of symbolic dynamics to information theory," *IEEE Trans. Inf. Theory*, vol. 29, no. 1, pp. 5–22, Jan. 1983.
[6] K. A. S. Immink, P. H. Siegel, and J. K. Wolf, "Codes for digital recorders," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2260–2299, Oct. 1998.
[7] A. Hareedy and R. Calderbank, "LOCO codes: Lexicographically-ordered constrained codes," *IEEE Trans. Inf. Theory*, vol. 66, no. 6, pp. 3572–3589, Jun. 2020.
[8] B. Vasic and E. Kurtas, *Coding and Signal Processing for Magnetic Recording Systems*. CRC Press, 2005.
[9] S. Zheng, Y. Liu, and P. H. Siegel, "PR-NN: RNN-based detection for coded partial-response channels," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 1967–1982, Jul. 2021.
[10] B. Dabak, A. Hareedy, and R. Calderbank, "Non-binary constrained codes for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 56, no. 11, pp. 1–10, Nov. 2020.

[11] R. Wood, M. Williams, A. Kavcic, and J. Miles, "The feasibility of magnetic recording at 10 terabits per square inch on conventional media," *IEEE Trans. Magn.*, vol. 45, no. 2, pp. 917–923, Feb. 2009.

[12] K. A. S. Immink, "Modulation systems for digital audio discs with optical readout," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Atlanta, Georgia, USA, Mar.–Apr. 1981, pp. 587–589.

[13] J. Saadé, A. Goulahsen, A. Picco, J. Huloux, and F. Pétrot, "Low overhead, DC-balanced and run length limited line coding," in *Proc. IEEE 19th Workshop on Signal and Power Integrity (SPI)*, Berlin, Germany, May 2015, pp. 1–4.

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

[15] V. Taranalli, H. Uchikawa, and P. H. Siegel, "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.

[16] R. Motwani, "Hierarchical constrained coding for floating-gate to floating-gate coupling mitigation in Flash memory," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Houston, TX, USA, Dec. 2011, pp. 1–5.

[17] Y. M. Chee, J. Chrisnata, H. M. Kiah, S. Ling, T. T. Nguyen, and V. K. Vu, "Capacity-achieving codes that mitigate intercell interference and charge leakage in Flash memories," *IEEE Trans. Inf. Theory*, vol. 65, no. 6, pp. 3702–3712, Jun. 2019.

[18] A. Hareedy, B. Dabak, and R. Calderbank, "Managing device lifecycle: Reconfigurable constrained codes for M/T/Q/P-LC Flash memories," *IEEE Trans. Inf. Theory*, vol. 67, no. 1, pp. 282–295, Jun. 2021.

[19] V. Braun and K. A. S. Immink, "An enumerative coding technique for DC-free runlength-limited sequences," *IEEE Trans. Commun.*, vol. 48, no. 12, pp. 2024–2031, Dec. 2000.

[20] A. Hareedy and R. Calderbank, "Asymmetric LOCO codes: Constrained codes for Flash memories," in *Proc. 57th Annual Allerton Conf. Commun., Control, and Computing*, Monticello, IL, USA, Sep. 2019, pp. 124–131.

[21] A. Hareedy, B. Dabak, and R. Calderbank, "The secret arithmetic of patterns: a general method for designing constrained codes based on lexicographic indexing," Oct. 2020. [Online]. Available: https://arxiv.org/abs/2010.10686

[22] J. Centers, X. Tan, A. Hareedy, and R. Calderbank, "Power spectra of constrained codes with level-based signaling: Overcoming finite-length challenges," *IEEE Trans. Commun.*, vol. 69, no. 8, pp. 4971–4986, Aug. 2021.

[23] 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.

[24] R. Laroia, N. Farvardin, and S. A. Tretter, "On optimal shaping of multidimensional constellations," *IEEE Trans. Inf. Theory*, vol. 40, no. 4, pp. 1044–1056, Jul. 1994.

[25] N. Kashyap, R. M. Roth and P. H. Siegel, "The capacity of count-constrained ICI-free systems," in *IEEE Int. Symp. Inf. Theory*, Paris, France, Jul. 2019, pp. 1592–1596.

[26] A. Kato and K. Zeger, "On the capacity of two-dimensional run-length constrained channels," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 1527–1540, Jul. 1999.

[27] B. H. Marcus, P. H. Siegel and J. K. Wolf, "Finite-state modulation codes for data storage," in *IEEE J. Sel. Areas Commun.*, vol. 10, no. 1, pp. 5–37, Jan. 1992.

[28] A. Hareedy, R. Kuditipudi, and R. Calderbank, "Minimizing the number of detrimental objects in multi-dimensional graph-based codes," *IEEE Trans. Commun.*, vol. 68, no. 9, pp. 5299–5312, Sep. 2020.