支持现代闪存页分离的高效约束码

Ahmed Hareedy，*IEEE*会员，Simeng Zheng，*IEEE*学生会员，
Paul Siegel，*IEEE*终身会士，以及Robert Calderbank，*IEEE*终身会士

arXiv:2302.01920v1 [cs.IT] 3 Feb 2023

摘要

2015年，固态器件在存储密度方面取得了对磁性器件的关键性优势，这得益于物理学、架构和信号处理领域的多项创新。其中一项最重要的创新，是在闪存器件中实现每个单元存储多于一个比特，即每个单元具有多于两个电荷电平。闪存器件采用约束编码，通过减轻源于单元间电荷传播的单元间干扰来提高可靠性。近期，人们提出了容量可达约束码，以用于每个单元具有多于两个电平的现代闪存器件。尽管这些码利用底层物理特性，实现了极低的冗余，但也导致复杂度显著增加，并限制了访问速度，因为页面无法单独读取。本文提出新的约束编码方案，既具有低复杂度，又能保留现代闪存器件所需的高速访问能力。其思路是通过仅对最左侧页上的数据进行编码（二进制编码），或仅对最左侧两页上的数据进行编码（四元编码），来消除易出错模式，同时保持所有其余页面上的数据不编码。我们的编码方案适用于每个单元具有任意数量 $q \geq 4$ 个电平的情况，支持系统编码与解码，并且码率接近容量。由于所提出的方案能够分离页面（四元编码情况下最左侧两页除外），我们将其称为读后即运行（*RR*）约束编码方案，以区别于对其他页面采用读后等待 的方案。提出四元RR编码方案，旨在限制二进制RR编码方案造成的码率损失；我们还表明，在复杂度和错误传播方面，四元RR编码方案同样具有竞争力。我们分析了这些新的RR编码方案，并讨论其对不同电荷电平出现概率的影响。

本研究部分经费由TÜB˙ITAK 2232-B青年研究人员国际奖学金资助。本文部分内容曾发表于2022年IEEE国际通信会议（ICC） [1]。Ahmed Hareedy任职于土耳其安卡拉06800中东技术大学（METU）电气与电子工程系（电子邮箱：ahareedy@metu.edu.tr）。Simeng Zheng和Paul Siegel任职于美国加利福尼亚州拉霍亚92093加利福尼亚大学圣迭戈分校（UCSD）电气与计算机工程系（电子邮箱：sizheng@ucsd.edu；psiegel@ucsd.edu）。Robert Calderbank任职于美国北卡罗来纳州达勒姆27708杜克大学电气与计算机工程系（电子邮箱：robert.calderbank@duke.edu）。

我们还展示了在实际的三级单元闪存器件上采用RR编码所取得的性能提升。

索引词

受限码、字典序、LOCO码、可重构码、数据存储、闪存、多级技术、可靠性、访问速度、读与运行。

I. 引言

受限编码的历史可以追溯到1948年。当时，香农用有限状态转移图（FSTD）表示受约束序列，并推导出约束条件下的容量 [2]。1970年，Tang 和 Bahl 提出了游程长度受限（RLL）码，以支持当时磁记录技术的发展 [3], ，这些码基于字典序索引。1973年，Cover 提出了一项关于枚举编码的结果 [4] ，数十年后，这项结果将被证明对基于字典序索引的受限码设计至关重要。Franaszek 等研究人员则开发了基于有限状态机（FSM）的受限码，这些有限状态机由有限状态转移图导出 [5]。1983年，Adler、Coppersmith 和 Hassner 提出了一种基于有限状态机系统地开发受限码的方法 [6]。截至1998年的受限编码发展史详见 [7]。

受限码能够通过消除易出错数据模式和不良序列来提升性能，因此有着广泛的应用。它们应用于一维（1D）磁记录设备，包括基于峰值检测的老式设备和基于序列检测的现代设备 [8], [9]。它们还可以与利用机器学习的稳健信号检测相结合 [10]。它们也应用于新兴的二维（2D）磁记录设备 [11], [12]。此外，受限码还用于在光记录设备中实现直流平衡和自校准 [13] ，以及用于许多数据传输相关的计算机标准 [14]。

在闪存器件中，编程为高电荷电平的单元向编程为低电荷电平的单元传播电荷，是单元间干扰（ICI） [15]的主要原因。对于每个单元具有任意数量 q 的电荷电平，这一结论都成立。如 [16] 所示，缓解单元间干扰可显著延长闪存的寿命；该研究针对多电平单元（MLC）闪存（$q = 4$）。有些数据模式被认为是造成单元间干扰的主要嫌疑模式。 [17] 研究了通过编码消除导致连续电平（$q - 10q - 1$）的数据模式

以及 [18]。 [16], [18], 和 [19]介绍了针对所有 $\mu < q-1$，通过编码消除导致连续电平（$q-1$）μ（$q-1$）的数据模式，这些模式也称为电平模式。

近期的一些研究重新探讨了 [3] 和 [4] ，以生成基于字典序索引的高效约束码，其中一个例子是 [20]。另一个例子是 [8], ，我们在其中提出了二进制对称字典序受限码（S-LOCO码），并展示了其在现代一维磁记录系统中的存储密度提升。我们将研究成果拓展至单层单元（SLC）闪存（$q=2$）， [21] 随后又拓展至每个单元具有任意数量 q 个电平的闪存 [19]。此外，我们设计了一种一般方法，可为禁止任意有限模式集合 [22], 设计 LOCO 码，这一方法将在本文中发挥作用。我们在 [23]中研究了二进制 LOCO 码的功率谱。LOCO 码具有达到容量、简单且易于重新配置的特点 [19], [22]。

尽管 [18] 和 [19] 中的受限码在码率方面效率颇高，但它们要求同时处理所有闪存页，这会降低访问速度。本文提出了二进制 读后即运行（*RR*）受限编码方案，使现代闪存器件中的页面能够分别访问，从而保持较高的访问速度。我们的二进制 RR 编码方案造成的码率损失很小，且适用于每个单元具有 $q \geq 4$ 个电平的任何闪存器件。其核心思想是只在一页（最左侧页）上应用受限码，而在其他 $\log_2 q-1$ 页上不进行编码。我们提出了一种二维RR编码方案，以及一种基于LOCO码的一维 RR 编码方案，并将后者命名为二进制 RR-LOCO编码。此外，为进一步降低码率损失且不影响设备可靠性，我们还提出了一种基于LOCO码的一维四元RR编码方案，称为四元RR-LOCO编码。具体而言，我们在两页（最左侧两页）上应用受限编码，而在其他 $\log_2 q-2$ 页上不进行编码。因此，除最左侧两页外，所有页面均彼此分离。我们的四元RR编码方案适用于每个单元具有$q \geq 8$ 个电平的任何闪存器件。[1] 我们表明，在逼近容量的码率下，我们的四元RR编码方案在复杂度和错误传播方面甚至优于二进制 RR 编码方案。文献中已有一些实现页分离的技术；然而，这些技术要么会造成显著的码率损失 [16] ，要么是针对特定闪存配置 [17]设计的。我们研究了所提出的RR编码方案的各个方面，包括电荷电平概率。我们

[1]这种 4 元 RR 编码 方案也适用于 $q=4$，但其禁止的模式更为温和，且不具备页分离 .

我们在一款实际的三层单元（TLC）闪存器件（$q = 8$）中介绍了实验结果，证明我们的编码方案能够显著提升寿命。

本文其余部分安排如下。第二节讨论有害模式、闪存映射以及我们提出的二维二进制 RR 编码方案。第三节介绍我们提出的一维二元 RR-LOCO 编码方案。第四节提出我们的一维四元 RR-LOCO 编码方案。第五节研究新方案的码率、复杂度和错误传播，并进行比较。第六节展示在 TLC 闪存上的实验结果。第七节总结全文。

II. 模式、映射与二维 RR 编码

如引言所述，文献中的研究对于所采用的禁用模式集合并没有严格统一的看法。此外，随着闪存器件老化，易出错模式的集合预计会扩大 [19]。根据我们最近的实验测试以及对 TLC 闪存进行的基于机器学习的 ICI 表征 [24]，我们决定重点研究以下特征所界定的集合。令

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

其中，q 是每个闪存单元的电平数（2 的正整数次幂），且 $\mathcal{V}_1 = \{0, 1, \ldots, q-1\} \setminus \mathcal{V}_0$。因此，我们关注的集合是在 $\mathcal{L}_q$[2] 中产生高-低-高电平模式的集合

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

该集合已涵盖文献中针对闪存采用的所有三元组禁用模式。正如我们将在第四节中看到的，可以通过移除少数对性能影响极小的模式来放宽该集合。闪存器件中的一个块可以视为由字线和位线构成的二维网格，每个交点处放置一个单元 [16]。无论电平模式出现在同一字线上的 3 个相邻单元中，还是出现在同一位线上的 3 个相邻单元中，$\mathcal{L}_q$ 中的电平模式都会造成不利影响。

示例 1. 考虑一个多层单元闪存器件，即 $q = 4$。在这种情况下，我们有 $\beta_1, \overline{\beta}_1 \in \{2, 3\}$。因此，我们关注的集合是产生以下结果的集合：

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

其中最后三个元素 $\mathcal{L}_4$ 是众所周知的 *[16], [17], [19]*。

[2]为简便起见，电平通过其索引 $\{0, 1, \ldots, q-1\}$ 来定义。

算法1递归交替格雷映射

输入： 单元的电平数 q，以及 $p = \log_2 q$。

定义映射 map，一个维度为 $q \times p$ 的二进制数组。

设置 $\mathrm{map}(0,:) = \mathbf{1}^p$。 （一个由 p 1 个 ’ 组成的序列）

对 $i \in \{0, 1, \ldots, p-1\}$ **执行**

  对 $j \in \{0, 1, \ldots, 2^i - 1\}$ **执行**

    映射$(2^i + j,\ :) = $映射$(2^i - 1 - j,\ :)$。

    翻转比特 映射$(2^i + j, i)$。 *(*映射中的每个序列都从右向左编索引，$0, 1, \ldots, p-1$*)*

  结束循环

结束循环

输出： 数组映射，将每个索引映射到二进制数据。

接下来，我们讨论如何在闪存中将数据映射为电荷电平，以及如何进行反向映射。由于本文始终关注页分离，因此这里的映射是将 q 个可能电荷电平之一映射为 $\log_2 q$ 个二进制位，每页对应一位，反之亦然。格雷映射的优点是任意两个相邻电荷电平之间只有一位不同，这有利于错误性能。我们采用递归交替格雷映射（RAGM），算法1展示了如何为任意 $q \geq 4$生成该映射。值得指出的是，文献中已将 RAGM 用于多级单元闪存 [16] 和 TLC闪存 [17]。因此，RAGM 严格来说并非一项全新的贡献。

示例 2. 考虑一个 *TLC* 闪存器件，即 $q = 8$。在这种情况下，算法*1*的输出（即 *RAGM*）为：

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

现在，我们可以讨论二进制编码方案了。首先，按照数组映射中每个序列里的比特索引方式，为闪存页编索引（参见算法1）。这意味着最左侧页的索引为 $p-1$。根据式（2）和算法1，电平模式

$\mathcal{L}_q$ 中的电平模式对应于这样的二进制模式：最左侧页（页面）上总是有两个 0，中间隔着一些比特，即 $0x0$。据此，禁止最左侧页（页面）上出现 $\{000, 010\}$，即可保证在向闪存器件写入数据时，至少沿字线（位线）方向，不会出现 $\mathcal{L}_q$ 中的任何电平模式，无论 $q \geq 4$ 为何。这对应于交织式 RLL（d, k）=(0, 1) 约束 [25]。值得注意的是，其他任何页都无需编码。因此，每个页的数据都将独立读取，并立即传送至低密度奇偶校验（LDPC）译码器进行处理。这一理念正是我们二进制 RR 约束编码方案的核心。[3]

RR编码可以仅沿字线方向（1D）、仅沿位线方向（1D）或沿两个方向（2D）进行。请注意，二元RR编码也会禁止某些无害的电平模式，例如TLC闪存中的474、555和676，从而不可避免地产生码率损失。不过，正如我们将在第五节中看到的，这种码率损失很小。此外，从二元编码转为四元编码时，将允许出现其中一些无害的电平模式，从而降低码率损失，正如我们将在第四节中看到的。RR-LOCO码是逼近容量码。

我们首先介绍二维二进制RR约束编码方案。顾名思义，我们希望通过简单的编码与解码，防止模式 $\mathcal{R}^2 = \{000, 010\}$ 出现在闪存器件中最左侧页的字线方向和位线方向上。编码遵循以下规则：

1）对于索引模4余数为0或1的字线，允许在最左侧页中索引模4余数为0或1的比特位置自由写入0和1。2）对于索引模4余数为2或3的字线，允许在最左侧页中索引模4余数为2或3的比特位置自由写入0和1。3）在其他比特位置，只能在最左侧页的字线上写入1。

图 1 展示了这种二维二元 RR 受限编码方案。从图中可以清楚地看出，最左侧页面中消除了 $\mathcal{R}^2 = \{000, 010\}$ 处的模式，从而在两个方向上都禁止出现 $\mathcal{L}_q$中的所有电平模式。编码时，输入数据位可自由放置在最左侧页面中标记为 x 的位置，而在其他页中则直接放置（不编码）。译码时，省略标记为 1 的位置处的信息，并读取

[3]在 [25]中，为多级单元闪存提出了一种等效方案，即 $q = 4$。

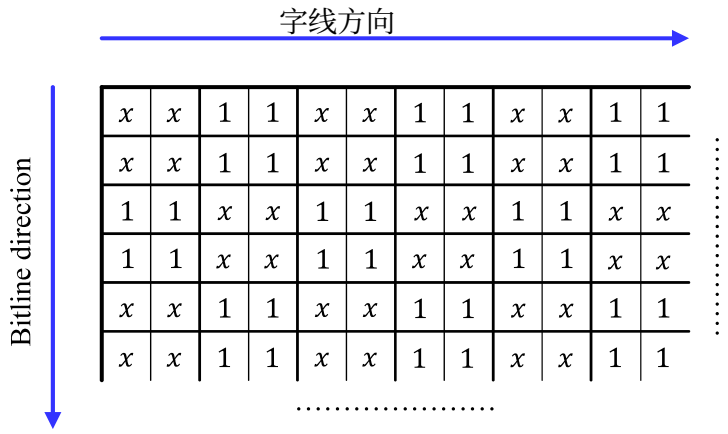


图 1. 通过所提出的二维二进制 RR 编码方案对数据进行编码后，二维闪存网格中最左侧的页面。符号 x 表示比特可以自由取 0 或 1。

剩余位置上的比特，无需额外处理，且不同闪存页之间不存在相关性。[4]

这种二维二进制方案在复杂度、访问速度和错误传播方面都很理想（见第五节）。从性能来看，它似乎也明显优于任何一维方案（二进制或四元）。然而，一维方案通过采用更高的码率，几乎可以达到相同的性能，我们稍后将对此进行更详细的讨论。

III. GF(2) 上的 RR-LOCO 编码

本节介绍一种二进制 RR 编码方案：该方案禁止在字线方向或位线方向上的最左侧页面中出现 $\{000, 010\}$，而其他页面不进行编码，因此会禁止 $\mathcal{L}_q$ 中的电平模式并实现页分离。该方案即二进制 RR-LOCO 编码方案。我们采用的受限码是一种根据 [22] 中的一般方法设计的二进制 LOCO 码。我们首先定义所提出的 LOCO 码。

定义 1. 禁止出现 $\mathcal{R}^2 = \{000, 010\}$ 中的模式的二进制 *LOCO* 码 $\mathcal{RC}_m^2$（其中 $m \geq 1$）由以下性质定义：

1） $\mathcal{RC}_m^2$ 中的码字定义在 $GF(2) = \{0, 1\}$ 上，长度为 m 比特。2） $\mathcal{RC}_m^2$ 中的码字按字典序排列。3） $\mathcal{RC}_m^2$ 中的码字不含 $\mathcal{R}^2$ 中的模式。

[4]文献 [25] 提出了一种等效的二维方案，可在多级单元闪存中沿字线和位线方向禁止最右侧页面上的模式 $\{101, 111\}$。

4) 包含所有满足 *1)–3)* 的码字。

字典序是按照规则“$0 < 1$”对码字进行升序排列，其中比特的重要性从左向右递减 [3], [19]。设计这种二进制 LOCO 码的第一步是确定分组结构。 $\mathcal{RC}_m^2$、 $m \geq 2$中的码字可划分为以下几组：

- 第 1 组：从左侧以 0011 开头的码字。
- 第2组：从左侧以 011 开头的码字。
- 第3组：从左侧以 1 开头的码字。

第二步是枚举 $\mathcal{RC}_m^2$ 中的码字，这可由定理 1 完成。令$N_2(m) \triangleq |\mathcal{RC}_m^2|$。

定理 1. 二进制 *LOCO* 码 $\mathcal{RC}_m^2$ 的基数由以下递归公式给出：

$$N_2(m) = N_2(m-1) + N_2(m-3) + N_2(m-4),\ m \geq 2, \tag{5}$$

其中定义的基数为：

$$N_2(-3) \triangleq 0,\ N_2(-2) = N_2(-1) = N_2(0) \triangleq 1,\ \textit{and}\ N_2(1) = 2. \tag{6}$$

证明： 我们分别计算每组的基数，然后将它们相加。设第 i 组的基数为 $N_{2,\ i}$。对于 $\mathcal{RC}_m^2$ 中的第 3 组，其码字与 $\mathcal{RC}_{m-1}^2$ 中的码字之间存在双射（从左侧附加 1 ）。因此，

$$N_{2,3}(m) = N_2(m-1). \tag{7}$$

对于$\mathcal{RC}_m^2$中的第2组，其码字与 $\mathcal{RC}_{m-2}^2$ 中从左侧以 1 开头的码字之间存在双射（从左侧附加 01）。因此，利用（7），

$$N_{2,2}(m) = N_{2,3}(m-2) = N_2(m-3). \tag{8}$$

对于 $\mathcal{RC}_m^2$中的第 1 组，其码字与 $\mathcal{RC}_{m-3}^2$ 中从左侧以 1 开头的码字之间存在双射（从左侧附加 001）。因此，利用（7），

$$N_{2,1}(m) = N_{2,3}(m-3) = N_2(m-4). \tag{9}$$

将 (7)、(8) 和 (9) 相加得到 (5)。除了 N_2(1)之外，定义的基数可以通过观察 N_2(1) $= 2$、 N_2(2) $= 4$、 N_2(3) $= 6$和 N_2(4) $= 9$来计算，由此得到四个方程。鉴于这些禁用模式，这一观察显而易见。 ■

将 $\mathcal{RC}_m^2$ 中的码字**c** 定义为**c** $\triangleq c_{m-1}c_{m-2}\ldots c_0$，其中 $c_i \triangleq \zeta$ 对应于 $i \geq m$，而 ζ 表示“超出码字边界”。LOCO码字比特 c_i, $0 \leq i \leq m-1$的整数等价形式为 a_i，即当 c_i 为 0 (1) 时， a_i 为 0 (1)。将 LOCO码 $\mathcal{RC}_m^2$ 中所有码字里的码字**c** 的字典序索引记为 g_2(**c**)，简称为 g(**c**)。一般而言，g(**c**) 属于 $\{0, 1, \ldots, N_2(m) - 1\}$。

第三步是在 $\mathcal{RC}_m^2$ 中指定码字内 1 出现的特殊情况。这些情况如下：

- 情况 2： $c_{i+2}c_{i+1}c_i = 001$。
- 情况 3： $c_{i+2}c_{i+1}c_i = 011$。
- 情况 4： $c_{i+2}c_{i+1}c_i = 101$ 或 $c_{i+2}c_{i+1}c_i = \zeta 01$。

典型情况或默认情况，即情况 1，简单来说就是“其他情况”。具体而言，就是 $c_{i+2}c_{i+1}c_i = 111$、 $c_{i+2}c_{i+1}c_i = \zeta 11$ 或 $c_{i+1}c_i = \zeta 1$ 的情况。

第四步和第五步是确定编码-解码规则，该规则规定了索引到码字以及码字到索引的映射。 $\mathcal{RC}_m^2$ 的这一规则见定理2。

定理2. 字典序索引 $g(\mathbf{c})$、$\mathbf{c} \in \mathcal{RC}_m^2$ 与二进制码字 **c** 本身之间的关系为：

$$g(\mathbf{c}) = \sum_{i=0}^{m-1} a_i \Big[(1 - y_{i,1})N_2(i-2) + (1 - y_{i,1} - y_{i,2})N_2(i-3)\Big], \tag{10}$$

其中$y_{i,\ 1}$ 和 $y_{i,\ 2}$ 定义如下：

$$\begin{aligned} &y_{i,1} = 1 \textit{ if } c_{i+2}c_{i+1}c_i \in \{001, 011\}, \textit{ and } y_{i,1} = 0 \textit{ otherwise}, \\ &y_{i,2} = 1 \textit{ if } c_{i+1}c_i = 01 \text{ s.t. } y_{i,1} = 0, \textit{ and } y_{i,2} = 0 \textit{ otherwise}. \end{aligned} \tag{11}$$

证明： 我们计算二进制 LOCO 码字中，情况 j 下比特 c_i 对所有 $j \in \{1, 2, 3, 4\}$ 的贡献 $g_{i,\ j}(c_i)$，然后将它们合并。对于典型情况（我们将其索引为 1），该贡献是 $\mathcal{RC}_{i+1}^2$ 中从左侧开始、以 0 开头的码字数量。因此，根据 (8) 和 (9)，

$$\begin{aligned} g_{i,1}(c_i) &= N_{2,2}(i+1) + N_{2,1}(i+1) \\ &= N_2(i-2) + N_2(i-3). \end{aligned} \tag{12}$$

对于情况 2（情况 3），该贡献是 $\mathcal{RC}^2_{i+3}$ 中从左侧开始、以 000（010）开头的码字数量。请注意，000 和 010 是禁用模式。因此，

$$g_{i,2}(c_i) = 0 \text{ and}$$
$$g_{i,3}(c_i) = 0. \tag{13}$$

对于情况 4，该项表示从左侧开始以 00 开头的码字数量，即 $\mathcal{RC}^2_{i+2}$ 中的数量。因此，利用 (9)，

$$g_{i,4}(c_i) = N_{2,1}(i+2) = N_2(i-2). \tag{14}$$

利用 (11) 中的 $y_{i,\ 1}$（对于情况 2 和 3）以及 $y_{i,\ 2}$（对于情况 4），再结合 a_i 合并 (12)、(13) 和 (14)，得到：

$$g_i(c_i) = a_i\Big[(1-y_{i,1})N_2(i-2) + (1-y_{i,1}-y_{i,2})N_2(i-3)\Big]. \tag{15}$$

将 (15) 代入 $g(\mathbf{c}) = \sum_{i=0}^{m-1} g_i(c_i)$，得到 (10)。 ■

为简洁起见，我们略过第六步，即组装编码与译码算法。这些算法可直接由 (10) 中的规则得出，详情请参阅 [3], [19], [22], 和 [26] 。请注意，我们有时将 $\mathcal{RC}^2_m$ 称为 一维二进制*RR-LOCO*码。LOCO码的编码-解码规则正是其算法复杂度低的原因，因此重配置只需重新编程加法器即可轻松完成 [8], [22]。

备注 1. 如果在将编码比特写入页面之前对其取反，则最左侧页面上的禁用模式集合会变为 $\{101, 111\}$，这在 *[16]* 中也有出现。在这种情况下，二进制 *LOCO* 码的基数仍与式 *(5)* 中相同，而编码-解码规则则恰好变为 *[21]* 中针对 $x = 1$ 的二进制非对称 *LOCO* 码规则：

$$g(\mathbf{c}) = \sum_{i=0}^{m-1} a_i N_2(i - a_{i+1}). \tag{16}$$

最左侧页面上的编码与译码仅涉及减法和加法。至于其余页面，则直接写入和读取数据（不进行编码）。我们的 一维二进制RR-LOCO编码方案既保证了简单性，又保持了较高的访问速度。

IV. 基于 GF(4) 的 RR-LOCO 编码

在本节中，我们提出一种基于 GF(4) 上的 LOCO 码的一维 RR 编码方案。该方案也是我们的四元 RR-LOCO 编码方案。其目标是限制由二进制 RR 编码方案导致的码率

损失，具体方法是对最左侧两页进行编码。通过使用两个比特而非一个比特来表征易出错模式，可以对这些模式进行更细致的分类，从而允许一些无害或危害较小的模式，进而在对性能影响可以忽略不计的情况下提高码率。

我们首先修改易出错模式集合。设

$$\begin{aligned}\theta_1, \overline{\theta}_1 \in \mathcal{W}_0 &\triangleq \left\{\frac{3q}{4}, \frac{3q}{4}+1, \ldots, q-1\right\},\\ \theta_2, \overline{\theta}_2 \in \mathcal{W}_1 &\triangleq \left\{\frac{q}{2}, \frac{q}{2}+1, \ldots, \frac{3q}{4}-1\right\},\\ \theta_3, \in \mathcal{W}_2 \cup \mathcal{W}_3,\ \mathcal{W}_2 &\triangleq \left\{\frac{q}{4}, \frac{q}{4}+1, \ldots, \frac{q}{2}-1\right\},\ \mathcal{W}_3 \triangleq \left\{0, 1, \ldots, \frac{q}{4}-1\right\},\end{aligned} \tag{17}$$

其中 q 表示每个闪存单元的电平数（为 2 的正整数次幂）。尽管在数学上$q \geq 4$，这里我们关注 $q \geq 8$的情形。于是，所关注的集合是由 $\mathcal{L}'_q \subset \mathcal{L}_q$中的高-低-高电平模式构成的集合：

$$\begin{aligned}\mathcal{L}'_q \triangleq \{\theta_1 \eta \overline{\theta}_1, \forall \theta_1, \overline{\theta}_1 \mid 0 \leq \eta < \min(\theta_1, \overline{\theta}_1)\} \ \cup\ \{\theta_1\theta_3\theta_2, \forall \theta_1, \theta_2, \theta_3\} \cup\\ \{\theta_2\theta_3\theta_1, \forall \theta_1, \theta_2, \theta_3\} \ \cup\ \{\theta_2\theta_3\overline{\theta}_2, \forall \theta_2, \overline{\theta}_2, \theta_3\}.\end{aligned} \tag{18}$$

该集合也包含文献中针对闪存采用的所有三元组禁用模式。集合 $\mathcal{L}'_q$ 与集合 $\mathcal{L}_q$ 之间的唯一区别是：在前一个集合中，只要左侧电平或右侧电平，或二者都属于 $\mathcal{W}_1$，中间电平就始终属于 $\mathcal{W}_2 \cup \mathcal{W}_3$。我们的实验结果表明， $\mathcal{L}_q \setminus \mathcal{L}'_q$ 中的电平模式对闪存器件读取时产生的错误贡献极小。

例 3. 考虑一个 *TLC* 闪存器件，即 $q = 8$。在这种情况下，我们有 $\theta_1, \theta_1 \in \{6, \overline{7}\}$,$\theta_2$, $\theta_2 \in \{4, 5\}$和 $\theta_3 \in \{0, 1, 2, 3\}$。于是，两个相关集合之间的差异仅在于一个电平模式：

$$\mathcal{L}_8 \setminus \mathcal{L}'_8 = \{545\}. \tag{19}$$

对于从电荷电平到二进制位的映射，我们采用算法1中的 RAGM。此外，我们按照算法1中数组映射内各序列中比特的索引方式，对闪存页进行索引。因此，我们在此关注索引为 $p-1$ 和 $p-2$ 的最左侧两页中的数据。针对这两个特定的闪存页，我们采用以下二进制到四元映射-解映射，其中 GF(4) $= \{0, 1, \alpha, \alpha^2\}$：

$$11 \longleftrightarrow 0\ (\mathcal{W}_3), \qquad 10 \longleftrightarrow 1\ (\mathcal{W}_2),$$
$$00 \longleftrightarrow \alpha\ (\mathcal{W}_1), \qquad 01 \longleftrightarrow \alpha^2\ (\mathcal{W}_0). \tag{20}$$

每个 GF(4) 符号对应的电平模式集合列于括号内。

由 (17)、(18) 和 (20) 可知，通过禁止索引为 $p-1$ 和 $p-2$ 的最左侧两页写入以下集合 $\mathcal{R}^4$ 中的四元模式，可以禁止 $\mathcal{L}'_q$ 中的电平模式沿字线方向或位线方向出现：

$$\mathcal{R}^4 = \{\alpha 0\alpha, \alpha 1\alpha, \alpha 0\alpha^2, \alpha 1\alpha^2, \alpha^2 0\alpha, \alpha^2 1\alpha, \alpha^2 0\alpha^2, \alpha^2 1\alpha^2, \alpha^2\alpha\alpha^2, \alpha^2\alpha^2\alpha^2\}. \tag{21}$$

再次强调，无需对其他任何页进行编码。因此，除最左侧两页外，数据将从各页独立读取，并立即传送给低密度奇偶校验（LDPC）译码器进行处理。这一思路正是我们四元RR约束编码方案的关键。

再次考虑一个TLC闪存器件（$q=8$）。与在最左侧页上禁止 $\mathcal{R}^2$ 中的模式不同，在最左侧两页上禁止 $\mathcal{R}^4$ 中的模式，可以允许许多无害模式；若采用二元RR编码，这些模式将被禁止，例如 444、474 和 555。

现在，我们介绍四元 RR 编码方案，该方案禁止最左侧两页在字线方向或位线方向上出现 $\mathcal{R}^4$ 中的模式，同时其他页面不进行编码。我们采用的受限码是根据 [22]中的一般方法设计的四元 LOCO 码。首先，我们定义所提出的 LOCO 码。

定义2. 一种 4元 *LOCO* 码 $\mathcal{RC}_m^4$，其中 $m \geq 1$，禁止出现 $\mathcal{R}^4$ 中的模式，其定义如下：

1） $\mathcal{RC}_m^4$ 中的码字定义在 $GF(4) = \{0, 1, \alpha, \alpha^2\}$ 上，且由长度为 m 个符号组成。2） $\mathcal{RC}_m^4$ 中的码字按字典序排列。3） $\mathcal{RC}_m^4$ 中的码字不包含 $\mathcal{R}^4$中的模式。4）所有满足 *1*）–*3*）的码字均包含在内。

此处的字典序是指按照“$0 \prec 1 \prec \alpha \prec \alpha^2$”规则对码字进行升序排列，其中符号的重要性从左向右递减 [3], [19]。构造此四元 LOCO 码的第一步是指定分组结构。设 γ_1 和 γ_2 属于 $\{0, 1\}$。 $\mathcal{RC}_m^4$, $m \geq 3$中的码字可划分为以下各组：

- 第 1 组：从左侧开始以 γ_1, $\forall\gamma_1$开头的码字。

- 第2组：从左侧开始以 $\alpha\gamma_1\gamma_2$, $\forall\gamma_1,\gamma_2$开头的码字。
- 第3组：从左侧开始的码字为 $\alpha\alpha$ 或 $\alpha\alpha^2$ 。
- 第4组：从左侧开始的码字为 $\alpha^2\gamma_1\gamma_2$、 $\forall\gamma_1$， γ_2。
- 第5组：从左侧开始的码字为 $\alpha^2\alpha\gamma_1\gamma_2$、 $\forall\gamma_1$， γ_2。 。
- 第6组：从左侧开始的码字为 $\alpha^2\alpha\alpha$ 。
- 第7组：从左侧开始的码字为 $\alpha^2\alpha^2\gamma_1\gamma_2$、 $\forall\gamma_1$， γ_2。 。
- 第8组：从左侧开始的码字为 $\alpha^2\alpha^2\alpha$ 。

第二步是枚举 $\mathcal{RC}_m^4$ 中的码字，这可由定理3完成。令$N_4(m) \triangleq |\mathcal{RC}_m^4|$。

定理3。*Th e* 4元 *LOCO*码 $\mathcal{RC}_m^4$ 的基数由递归公式给出 *:*

$$\begin{aligned} N_4(m) &= 3N_4(m-1) - 2N_4(m-2) + 9N_4(m-3) \\ &\quad + 7N_4(m-4) + 6N_4(m-5) + 4N_4(m-6),\ m \geq 3, \end{aligned} \tag{22}$$

其中定义的基数如下:

$$N_4(-5) \triangleq \frac{1}{32},\ N_4(-4) \triangleq -\frac{1}{16},\ N_4(-3) \triangleq 0,\ N_4(-2) \triangleq \frac{1}{4},\ N_4(-1) \triangleq \frac{1}{2},\ N_4(0) \triangleq 1,$$
$$\textit{and}\ N_4(1) = 4,\ N_4(2) = 16. \tag{23}$$

证明: 我们计算每一组的基数，再将它们相加。设第 i 组的基数为 $N_{4,i}$。至于 $\mathcal{RC}_m^4$ 中的第 1 组，其码字与 $\mathcal{RC}_{m-1}^4$ 中的码字之间存在一个满射（从左侧附加 0 或 1）。因此,

$$N_{4,1}(m) = 2N_4(m-1). \tag{24}$$

至于 $\mathcal{RC}_m^4$ 中的第2组，其码字与$\mathcal{RC}_{m-3}^4$中的码字之间存在一个满射。因此,

$$N_{4,2}(m) = (2)(2)N_4(m-3) = 4N_4(m-3). \tag{25}$$

对于 $\mathcal{RC}_m^4$中的第3组，其码字与 $\mathcal{RC}_{m-1}^4$中从左侧以 α 或 α^2 开头的码字之间存在双射。因此，利用（24），

$$N_{4,3}(m) = N_4(m-1) - N_{4,1}(m-1) = N_4(m-1) - 2N_4(m-2). \tag{26}$$

至于 $\mathcal{RC}_m^4$中的第4组，其基数与第2组相同。Thu s,

$$N_{4,4}(m) = (2)(2)N_4(m-3) = 4N_4(m-3). \tag{27}$$

对于第 5在 $\mathcal{RC}_m^4$中，它的处理方式与组2和组4类似。因此，

$$N_{4,5}(m) = (2)(2)N_4(m-4) = 4N_4(m-4). \tag{28}$$

对于 $\mathcal{RC}_m^4$中的第6组，其码字与 $\mathcal{RC}_{m-2}^4$中从左侧以 α 开头的码字之间存在双射。因此，利用式（25）和（26），

$$N_{4,6}(m) = N_{4,2}(m-2) + N_{4,3}(m-2) = N_4(m-3) - 2N_4(m-4) + 4N_4(m-5). \tag{29}$$

对于 $\mathcal{RC}_m^4$中的第7组，其基数与第5组相同。因此，

$$N_{4,7}(m) = (2)(2)N_4(m-4) = 4N_4(m-4). \tag{30}$$

对于 $\mathcal{RC}_m^4$中的第8组，其码字与 $\mathcal{RC}_{m-1}^4$中从左侧以 $\alpha^2\alpha$ 开头的码字之间存在双射。因此，利用式（28）和（29），

$$N_{4,8}(m) = N_{4,5}(m-1) + N_{4,6}(m-1) = N_4(m-4) + 2N_4(m-5) + 4N_4(m-6). \tag{31}$$

将式（24）、（25）、（26）、（27）、（28）、（29）、（30）和（31）相加，得到式（22）。除 N_4(1)和 N_4(2)之外，定义的基数可根据 m取较小值时的基数计算，由此建立六个方程。 ■

将$\mathcal{RC}_m^4$ 中的码字**c** 定义为**c** $\triangleq c_{m-1}c_{m-2}\dots c_0$，其中 $i \geq m$对应 $c_i \triangleq \zeta$，而 ζ表示“超出码字边界”。LOCO码字符号 c_i,$0 \leq i \leq m-1$的整数等价形式为 a_i，即当 c_i 分别为0、1、 α或 α^2时， a_i 为0、1、2或3。将LOCO码$\mathcal{RC}_m^4$ 中所有码字里的码字**c** 的字典序索引记为 g_4(**c**,m)，简写为 g(**c**)。一般而言， g(**c**)属于 $\{0,1,\dots,N_4(m)-1\}$。

第三步是指定码字 $\mathcal{RC}_m^4$ 中 GF(4)$\setminus\{0\}$ 符号出现的典型/特殊情况。令 γ 属于 $\{\zeta,0,1\}$，且 χ 属于 $\{\alpha,\alpha^2\}$。这些情况如下：

- 情况 1.a： $c_{i+1}c_i = \gamma 1$ 或 $c_{i+1}c_i = \gamma\alpha$，对所有 γ 均成立。
- 情况 1.b： $c_{i+1}c_i = \gamma\alpha^2$，对所有 γ 均成立。
- 情况 2： $c_{i+1}c_i = \chi 1$ 或 $c_{i+1}c_i = \chi\alpha$，对所有 χ 均成立。
- 情况 3： $c_{i+1}c_i = \alpha\alpha^2$。
- 情况 4： $c_{i+1}c_i = \alpha^2\alpha^2$。

典型情况或默认情况是情况 1（情况 1.a 和情况 1.b 合并）。

第四步和第五步是找出编码-解码规则，该规则规定了索引与码字之间的双向映射。$\mathcal{RC}_m^4$ 的这一规则见定理 4。

定理 4。字典序索引 $g(\mathbf{c})$、$\mathbf{c} \in \mathcal{RC}_m^4$与 4元码字 $\mathbf{c}$ 本身之间的关系如下：

$$g(\mathbf{c}) = \sum_{i=0}^{m-1} \Big[\big[(y_{i,1} + y'_{i,1})a_i + y_{i,3}\big] N_4(i) + \big[2(y_{i,2}a_i + y_{i,3} - y'_{i,1}) + 5y_{i,\mathrm{d}}\big] N_4(i-1) \\ \big[4(y'_{i,1} + y_{i,3}) + 2y_{i,\mathrm{d}}\big] N_4(i-2) + 4y_{i,\mathrm{d}} N_4(i-3) \Big], \tag{32}$$

其中$y_{i,\ 1}$，$y'_{i,\ 1}$，$y_{i,\ 2}$，$y_{i,\ 3}$，以及 $y_{i,\ \mathrm{d}}$ 定义如下：

$$\begin{aligned}
&y_{i,1} = 1 \textit{ if } c_{i+1}c_i \in \{\gamma 1, \gamma\alpha \mid \forall\gamma\}, \textit{ and } y_{i,1} = 0 \textit{ otherwise},\\
&y'_{i,1} = 1 \textit{ if } c_{i+1}c_i \in \{\gamma\alpha^2 \mid \forall\gamma\}, \textit{ and } y'_{i,1} = 0 \textit{ otherwise},\\
&y_{i,2} = 1 \textit{ if } c_{i+1}c_i \in \{\chi 1, \chi\alpha \mid \forall\chi\}, \textit{ and } y_{i,2} = 0 \textit{ otherwise},\\
&y_{i,3} = 1 \textit{ if } c_{i+1}c_i = \alpha\alpha^2, \textit{ and } y_{i,3} = 0 \textit{ otherwise},\\
&y_{i,\mathrm{d}} = 1 \textit{ if } c_{i+1}c_i = \alpha^2\alpha^2, \textit{ and } y_{i,\mathrm{d}} = 0 \textit{ otherwise}.
\end{aligned} \tag{33}$$

证明：我们计算在情况 j 下，符号 c_i 对 4 元 LOCO 码字中所有 $j \in \{1,2,3,4\}$ 所产生的贡献 $g_{i,\ j}(c_i)$，然后将它们合并。对于典型情况 a（我们将其索引记为 $1.a$），该贡献是从 $\mathcal{RC}_{i+1}^4$ 左侧开始、以 $c'_i < c_i$ 开头的码字数量，其中 $c_i \in \{1, \alpha\}$。因此，利用 (24)，

$$g_{i,1.a}(c_i) = a_i N_4(i+1-1) = a_i N_4(i). \tag{34}$$

对于典型情况 b（我们将其索引记为 $1.b$），该贡献是从 $\mathcal{RC}_{i+1}^4$ 左侧开始、以 $c'_i < c_i$ 开头的码字数量，其中 $c_i = \alpha^2$。因此，利用 (24)、(25) 和 (26)，

$$\begin{aligned}
g_{i,1.b}(c_i) &= N_{4,1}(i+1) + N_{4,2}(i+1) + N_{4,3}(i+1)\\
&= 3N_4(i) - 2N_4(i-1) + 4N_4(i-2).
\end{aligned} \tag{35}$$

对于情况 2，该贡献是从 $\mathcal{RC}_{i+1}^4$ 左侧开始、以 $c'_i\gamma_1$, $c'_i < c_i$ 开头的码字数量，其中 $c_i \in \{1, \alpha\}$ 和 $\gamma_1 \in \{0, 1\}$。因此，利用 (24)，

$$g_{i,2}(c_i) = a_i N_{4,1}(i) = 2a_i N_4(i-1). \tag{36}$$

对于情况 3，这一项是从左侧开始、以 $\alpha c'_i$, $c'_i < c_i$ 开头的码字数量，其中$c_i = \alpha^2$，位于 $\mathcal{RC}_{i+2}^4$ 中。这些就是所有以 $\gamma_1\gamma_2$ 开头的码字，适用于所有 γ_1 和

γ_2，从左侧开始，位于 $\mathcal{RC}^4_{i+1}$ 中，再加上从左侧开始、以 α 开头且位于 $\mathcal{RC}^4_{i+1}$ 中的所有码字。因此，利用 (24)、(25)和(26)，

$$\begin{aligned} g_{i,3}(c_i) &= 2N_{4,1}(i) + N_{4,2}(i+1) + N_{4,3}(i+1) \\ &= N_4(i) + 2N_4(i-1) + 4N_4(i-2). \end{aligned} \tag{37}$$

对于情况 4，这一部分是从左侧开始以 $\alpha^2 c_i'$, $c_i' < c_i$ 开头的码字数量，其中 $c_i = \alpha^2$ 位于 $\mathcal{RC}^4_{i+2}$ 中。这些码字包括：对于所有 γ_1 和 γ_2，从左侧开始以 $\gamma_1\gamma_2$ 开头的所有码字，以及从左侧开始以 $\alpha^2\alpha$ 开头的所有码字，其中 $\alpha^2\alpha$ 位于 $\mathcal{RC}^4_{i+2}$ 中。因此，利用 (24)、(28) 和 (29)，

$$\begin{aligned} g_{i,4}(c_i) &= 2N_{4,1}(i) + N_{4,5}(i+2) + N_{4,6}(i+2) \\ &= 5N_4(i-1) + 2N_4(i-2) + 4N_4(i-3). \end{aligned} \tag{38}$$

我们使用 (33) 中的 $y_{i,\,1}$、$y'_{i,\,1}$（对于情况 1）、$y_{i,\,2}$（对于情况 2）、$y_{i,\,3}$（对于情况 3），以及 $y_{i,\,\mathrm{d}}$（对于情况 4），并结合 a_i，来合并 (34)、(35)、(36)、(37) 和 (38)。我们采用以下合并函数，其中 $f^{\mathrm{mer}}_\ell(\cdot)$ 与 $N_4(i+1-\ell)$ 相关联：

$$\begin{aligned} f^{\mathrm{mer}}_1(\cdot) &= (y_{i,1} + y'_{i,1})a_i + y_{i,3}, \\ f^{\mathrm{mer}}_2(\cdot) &= 2(y_{i,2}a_i + y_{i,3} - y'_{i,1}) + 5y_{i,\mathrm{d}}, \\ f^{\mathrm{mer}}_3(\cdot) &= 4(y'_{i,1} + y_{i,3}) + 2y_{i,\mathrm{d}}, \\ f^{\mathrm{mer}}_4(\cdot) &= 4y_{i,\mathrm{d}}. \end{aligned} \tag{39}$$

因此，符号贡献 $g_i(c_i)$的一般形式为：

$$g_i(c_i) = \sum_{\ell=1}^{4} f^{\mathrm{mer}}_\ell(\cdot) N_4(i+1-\ell). \tag{40}$$

将 (39) 和 (40) 代入 $g(\mathbf{c}) = \sum_{i=0}^{m-1} g_i(c_i)$，即可得到 (32)。 ■

注 2. 请注意，线性无关的合并变量的数量始终小于最终情况 *[22]* 的数量。此处，$y_{i,\,\mathrm{d}}$ 依赖于其他合并变量，因为它可以写成 $y_{i,\,\mathrm{d}} = \mathbb{1}(a_i)(1 - y_{i,\,1} - y'_{i,\,1} - y_{i,\,2} - y_{i,\,3})$，其中 $\mathbb{1}(a_i) = 1$ 若 $a_i > 0$ 且 $\mathbb{1}(a_i) = 0$ 若 $a_i = 0$。

为简洁起见，我们再次略过第六步，即编写编码和解码算法。这些算法可由式 (32) 中的规则直接推导得出，详情请参阅

读者可参阅 [3], [19], [22], 和 [26] 。注意，我们有时将 $\mathcal{RC}_m^4$ 称为 *1D*4元 *RR-LOCO*码。

V. 码率、复杂度和错误传播

我们首先计算渐近码率。遗憾的是，众所周知，推导二维受限码的容量极其困难。因此，我们将仅在一维受限编码设置下推导容量 $C_{\mathcal{L}_q}^{\text{1D}}$，而该容量已经高于二维设置下的容量。因此，对于至少在一个方向上禁止出现 $\mathcal{L}_q$ 中的模式的器件而言， $C_{\mathcal{L}_q}^{\text{1D}}$ 是可实现的最高码率上限。我们很快将说明，就性能而言，一维受限编码已足够。

图 2 展示了禁止出现 $\mathcal{L}_q$ 中的电平模式的序列的 FSTD。根据该 FSTD，通用邻接矩阵为（向量均为行向量）：

$$\mathbf{A}_1 = \left[\begin{array}{c|c|c|c} \frac{q}{2} & \mathbf{1}_{\frac{q}{2}} & 0 & \mathbf{0}_{\frac{q}{2}-1} \\ \hline \mathbf{0}_{\frac{q}{2}}^{\mathrm{T}} & \mathbf{U}_{\frac{q}{2}}^{1} & \frac{q}{2}\mathbf{1}_{\frac{q}{2}}^{\mathrm{T}} & \begin{array}{c}\mathbf{0}_{\frac{q}{2}-1} \\ \hline \mathbf{L}_{\frac{q}{2}-1}^{1}\end{array} \\ \hline \frac{q}{2} & \mathbf{0}_{\frac{q}{2}} & 0 & \mathbf{0}_{\frac{q}{2}-1} \\ \hline \mathbf{0}_{\frac{q}{2}-1}^{\mathrm{T}} & \mathbf{I}_{\frac{q}{2}-1} \;\big|\; \mathbf{0}_{\frac{q}{2}-1}^{\mathrm{T}} & \frac{q}{2}\mathbf{1}_{\frac{q}{2}-1}^{\mathrm{T}} & \begin{array}{c}\mathbf{0}_{\frac{q}{2}-1} \\ \hline \mathbf{L}_{\frac{q}{2}-2}^{1} \;\big|\; \mathbf{0}_{\frac{q}{2}-2}^{\mathrm{T}}\end{array} \end{array}\right], \tag{41}$$

其中，$\mathbf{U}_\delta^1$（$\mathbf{L}_\delta^1$）是大小为 $\delta \times \delta$ 的仅含 **1** 的上（三角）矩阵。因此，根据 [2],，禁止出现 $\mathcal{L}_q$ 中的电平模式的一维受限码的归一化容量为：

$$C_{\mathcal{L}_q}^{\text{1D}} = \frac{\log_2(\lambda_{\max}(\mathbf{A}_1))}{\log_2 q}, \tag{42}$$

其中 $\lambda_{\max}$**(A)** 是矩阵**A**的最大正实特征值。[5]

防止 $\{000, 010\}$ 的二维二进制码的容量等于二维 **(0，1)** 游程受限码的容量，即 ≈ 0.5879 [28]。因此，我们的二维 RR 编码方案的归一化容量为：

$$C_{\text{RR2}}^{\text{2D}} \approx \frac{0.5879 + \log_2 q - 1}{\log_2 q} = \frac{\log_2 q - 0.4121}{\log_2 q}. \tag{43}$$

[5]对于正整数 $a + b \le q$、 $\{0, 1, \ldots, q-1\}$中最大的 a 个电平构成的集合 H，以及最小的 b 个电平构成的集合 L，文献 [27] 推导出了禁止 $\{\beta_1\beta_2\beta_1 \mid \beta_1, \beta_1 \in H,\ \beta_2 \in L\}$ 中所有电平模式的受限系统的（计数受限）容量公式。 −

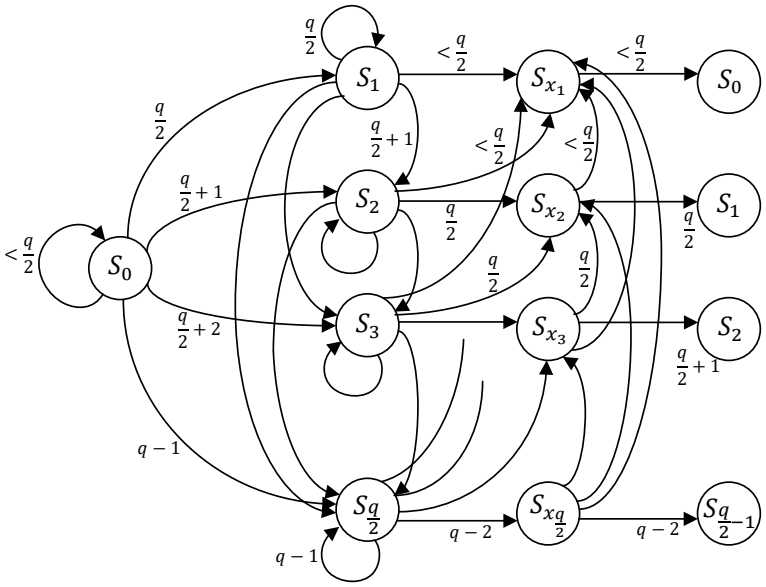


图 2. 禁止出现 $\mathcal{L}_q$ 中电平模式的一维受限序列的 FSTD，适用于任意 q。为简化起见，此处直接对电平模式进行操作。

表 I $C^{\mathrm{1D}}_{\mathcal{L}_q}$、一维二进制 RR 容量 $C^{\mathrm{1D}}_{\mathrm{RR2}}$ 与一维 4 元 RR 容量 $C^{\mathrm{1D}}_{\mathrm{RR4}}$ 的容量比较

q	$C^{\mathrm{1D}}_{\mathcal{L}_q}$	$C^{\mathrm{1D}}_{\mathrm{RR2}}$	容量差 %	$C^{\mathrm{1D}}_{\mathrm{RR4}}$
4	0.8941	0.8471	5.257%	0.8859
8	0.9235	0.8981	2.750%	0.9239
16	0.9401	0.9235	1。766%	0.9429
32	0.9509	0.9388	1.272%	0.9544

如上所述，禁止 $\mathcal{R}^2 = \{000, 010\}$ 中模式的一维受限系统可解释为交织 RLL（d, k）=(0，1) 受限系统，其容量已知为 $\log_2((1+\sqrt{5})/2) \approx 0.6942$。因此，我们的一维 RR-LOCO 编码方案的归一化容量为:

$$C^{\mathrm{1D}}_{\mathrm{RR2}} = \frac{\log_2((1+\sqrt{5})/2) + \log_2 q - 1}{\log_2 q} \approx \frac{\log_2 q - 0.3058}{\log_2 q}. \tag{44}$$

不同 q 取值下，$C^{\mathrm{1D}}_{\mathcal{L}_q}$ 与 $C^{\mathrm{1D}}_{\mathrm{RR2}}$ 之间的容量差见表 I。表中显示，容量差很小，并且随着 q 增大，容量差还会进一步缩小。

当禁止出现 $\mathcal{L}'_q$ 中的电平模式时，一维受限系统的容量 $C^{\mathrm{1D}}_{\mathcal{L}'_q}$ 略高于 $C^{\mathrm{1D}}_{\mathcal{L}_q}$，因为 $\mathcal{L}'_q \subset \mathcal{L}_q$。为简洁起见，我们省略 $C^{\mathrm{1D}}_{\mathcal{L}'_q}$ 的推导。

图 3 给出了一个一维四元受限系统的 FSTD，其中禁止出现 $\mathcal{R}^4$ 中的模式。邻接矩阵为：

$$\mathbf{A}_2 = \begin{bmatrix} 2 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 1 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 1 & 0 \end{bmatrix}.$$

特征多项式为：

$$\det(x\mathbf{I} - \mathbf{A}_2) = x^6 - 3x^5 + 2x^4 - 9x^3 - 7x^2 - 6x - 4. \tag{45}$$

可以看出，若将 x 替换为 $\lambda_{\mathrm{c}} = \lambda_{\max}(\mathbf{A}_2)$，则得到：

$$\lambda_{\mathrm{c}}^m = 3\lambda_{\mathrm{c}}^{m-1} - 2\lambda_{\mathrm{c}}^{m-2} + 9\lambda_{\mathrm{c}}^{m-3} + 7\lambda_{\mathrm{c}}^{m-4} + 6\lambda_{\mathrm{c}}^{m-5} + 4\lambda_{\mathrm{c}}^{m-6}, \tag{46}$$

这与式 (22) 中的基数递推一致。该四元受限系统的容量为 $\log_2(\lambda_{\max}(\mathbf{A}_2)) = \log_2(3.4147) = 1.7718$ 比特/符号。因此，我们的一维四元 RR-LOCO 编码方案的归一化容量为：

$$C_{\mathrm{RR4}}^{\mathrm{1D}} = \frac{1.7718 + \log_2 q - 2}{\log_2 q} \approx \frac{\log_2 q - 0.2282}{\log_2 q}. \tag{47}$$

表 I 展示了一维四元 RR 方案相较于一维二进制 RR 方案所实现的容量增益；我们将说明，其性能（即闪存器件保护能力）几乎相同。一个有趣的观察是，对于 $q \in \{8, 16, 32\}$，我们的一维四元 RR 方案的容量 $C_{\mathrm{RR4}}^{\mathrm{1D}}$ 略高于 $C_{\mathcal{L}_q}^{\mathrm{1D}}$。

接下来，我们讨论有限长度码率。首先，我们的二维二元 RR 受限编码方案的归一化码率为：

$$R_{\mathrm{RR2}}^{\mathrm{2D}} = \frac{0.5 + \log_2 q - 1}{\log_2 q} = \frac{\log_2 - 0.5}{\log_2 q} \tag{48}$$

由于我们的最左页编码的码率为 0.5。

对于我们的一维二进制RR-LOCO编码方案，我们在 $\mathcal{RC}_m^2$ 最左侧页上相邻码字之间插入模式 11 进行桥接，并移除码字 $\mathbf{1}^m$ 以实现自时钟 [19], [22]。因此，最左侧页上的码率为 $\lfloor \log_2(N_2(m) - 1) \rfloor / (m+2)$，我们的一维二进制RR-LOCO编码方案的归一化码率为：

$$R_{\mathrm{RR2}}^{\mathrm{1D}} = \frac{1}{\log_2 q} \left[\frac{\lfloor \log_2(N_2(m) - 1) \rfloor}{m+2} + \log_2 q - 1 \right]. \tag{49}$$

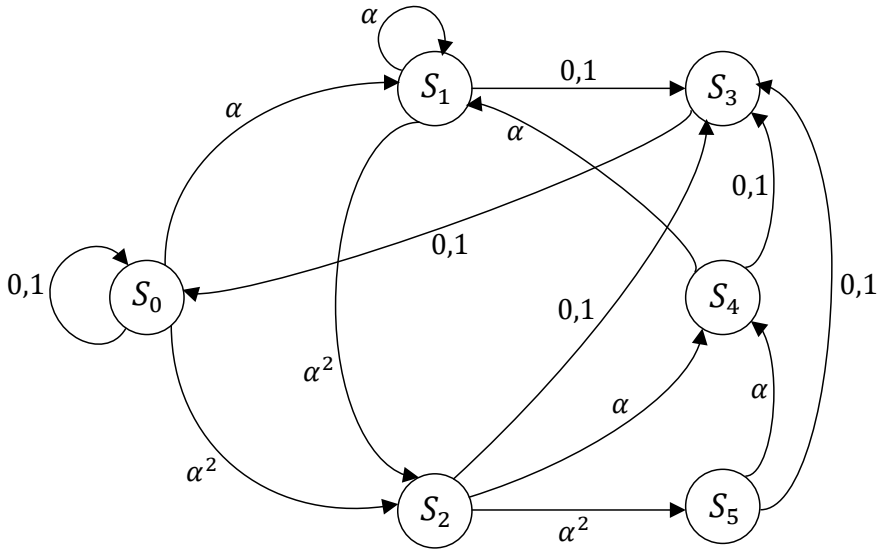


图 3. 禁用模式为 $\mathcal{R}^4$ 的一维四元受限序列的 FSTD。

一维二进制RR-LOCO编码方案是容量达成方案，即 $m \to \infty$ 趋于 $R_{\mathrm{RR2}}^{\mathrm{1D}}$ 时， $C_{\mathrm{RR2}}^{\mathrm{1D}}$ 的极限为 [19]（另见，）。另一种容量达成的一维 RR 受限编码方案，可通过对用于 RLL（d，k）=（0，1）约束 [29] 的最优分组码的码字进行交织来实现，无需使用桥接比特，并可应用于最左侧页面。不过，LOCO码具有简单且可重配置的优点，这一点在设备老化时尤为重要 [19]。

对于我们的一维四元 RR-LOCO 编码方案，由于任何用于分隔 α^2 和 α^2 的符号都会生成禁用模式，因此无法在 $\mathcal{RC}_m^4$ 的最左侧两页上的连续码字之间用单个 GF(4) 符号进行桥接。我们提出一种新颖的双符号桥接方案，可以在桥接区间内对输入信息比特进行编码，如下所示：

- 对于输入信息比特 $00 \in$ GF(2)，使用 $00 \in$ GF(4) 进行桥接。
- 对于输入信息比特 $01 \in$ GF(2)，使用 $01 \in$ GF(4) 进行桥接。
- 对于输入信息比特 $10 \in$ GF(2)，使用 $10 \in$ GF(4) 进行桥接。
- 对于输入信息比特 $11 \in$ GF(2)，使用 $11 \in$ GF(4) 进行桥接。

虽然该桥接方案不会影响渐近速率，但它显著缩短了达到特定速率所需的码长，从而大幅降低了复杂度和错误传播。

为实现自时钟，我们移除了两个码字 $\mathbf{0}^m$ 和 $\mathbf{1}^m$，鉴于上述桥接 [19], [22]，这是意料之中的。因此，最左侧两页上的码率为 ($\lfloor \log_2(N_4(m) - 2) \rfloor +$

2)/(m+2) 比特/s 符号，而我们的一维四元 RR-LOCO 编码方案的归一化码率为 :

$$R_{\mathrm{RR4}}^{\mathrm{1D}}=\frac{1}{\log_2 q}\left[\frac{\lfloor \log_2(N_4(m)-2)\rfloor+2}{m+2}+\log_2 q-2\right]. \tag{50}$$

一维四元 RR-LOCO 编码方案属于容量达成方案，也就是说，当 $m\to\infty$ 趋于$R_{\mathrm{RR4}}^{\mathrm{1D}}$ 时，$C_{\mathrm{RR4}}^{\mathrm{1D}}$。RR-LOCO码简单且具有可重配置性，这一点在器件老化 [19]时尤为重要。

我们提出的二维二进制 RR 约束编码方案无需增加编码与解码的复杂度，因为数据直接写入/读取最左侧页上的特定位置，并直接写入/读取其他页上的所有位置。至于一维二进制 RR-LOCO 编码方案，其复杂度取决于执行编码-解码规则的加法器大小，即:

$$s_2=\lfloor \log_2(N_2(m)-1)\rfloor \tag{51}$$

比特。同样，对于一维四元 RR-LOCO 编码方案，其复杂度取决于加法器大小，即:

$$s_4=\lfloor \log_2(N_4(m)-2)\rfloor \tag{52}$$

比特。为便于实现并避免影响访问速度，我们倾向于沿字线而非位线应用一维 RR-LOCO 编码方案，因为其性能非常接近，如第六节的实验结果所示。

错误传播是指一次写入错误在读取时导致多个错误的现象。二维二进制 RR 编码方案不会产生错误传播。因此，其错误传播因子为 $E_{\mathrm{RR2}}^{\mathrm{2D}}=1$。对于一维二进制 RR-LOCO 编码方案，不存在码字到码字的错误传播。不过，仅在最左侧页上，由码字到消息转换 [8], [19] 导致的有限错误传播仍然存在。这种错误传播平均会影响 $s_2/2$ 比特，其中 s_2 也是（ 51 ）中的消息长度。因此，在 $\log_2 q$ 个页面上取平均后的错误传播因子为:

$$E_{\mathrm{RR2}}^{\mathrm{1D}}=\frac{1}{\log_2 q}\left[\frac{s_2}{2}+\log_2 q-1\right]. \tag{53}$$

对于一维四元 RR-LOCO 编码方案，同样存在有限的错误传播，这完全是由最左侧两页上的 LOCO 码字到消息转换 [8], [19] 造成的。平均而言，这种错误传播会波及 $s_4/2$ 比特，其中 s_4 同样表示式（52）中的消息长度。注意，在每个桥接区间编码的两个额外比特不会产生错误传播，这些比特用于指定两个四元桥接符号。

表 II 相同长度下二维 RR 与一维二进制 RR 受限编码方案在码率、复杂度和错误传播方面的比较

q	m	$R_{\mathrm{RR2}}^{\mathrm{2D}}$	$R_{\mathrm{RR2}}^{\mathrm{1D}}$	s_2	$E_{\mathrm{RR2}}^{\mathrm{2D}}$	$E_{\mathrm{RR2}}^{\mathrm{1D}}$
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

中的两个四元桥接符号。因此，最左侧两页中任一页上的平均错误传播为：

$$\frac{s_4}{2}\cdot\frac{m}{m+2}+1\cdot\frac{2}{m+2}=\frac{s_4m+4}{2(m+2)}. \tag{54}$$

因此，对 $\log_2 q$ 个页面求平均的错误传播因子为：

$$E_{\mathrm{RR4}}^{\mathrm{1D}}=\frac{1}{\log_2 q}\left[2\cdot\frac{s_4m+4}{2(m+2)}+\log_2 q-2\right]=\frac{1}{\log_2 q}\left[\frac{s_4m+4}{m+2}+\log_2 q-2\right]. \tag{55}$$

比较一维二进制与一维四元 RR-LOCO 编码方案的另一个指标，是给定码率下编码数据的量。该数据量越小，码就越能以较短的长度 m达到所需码率，这是一个优势。由于对于我们的一维二进制和一维四元 RR-LOCO 编码方案，桥接分别使用两个比特和两个符号，因此这些编码数据量 $D_{\mathrm{RR2}}^{\mathrm{1D}}$ （二进制）和 $D_{\mathrm{RR4}}^{\mathrm{1D}}$ （四元）为：

$$D_{\mathrm{RR2}}^{\mathrm{1D}}=(m+2)\log_2 q,\ m \text{ is the length of } \mathcal{RC}_m^2, \tag{56}$$

$$D_{\mathrm{RR4}}^{\mathrm{1D}}=(m+2)\log_2 q,\ m \text{ is the length of } \mathcal{RC}_m^4. \tag{57}$$

表 II 给出了所提出的二进制 RR 方案在不同参数下的归一化码率、加法器位宽和错误传播因子。与二维二进制RR约束编码方案相比，一维二进制 RR-LOCO 编码方案具有显著的码率优势；对于 $q=4$、 $q=8$ 和 $q=16$，其优势分别达到 10.147%、6.096% 和 4.343%。二维二进制 RR 方案在复杂度和错误传播方面均具有明显优势，因为它无需进行任何编码和解码处理。

表 III 达到特定码率时，一维二进制 RR 与一维四元 RR 约束编码方案在最小编码数据量、复杂度和错误传播方面的比较

q	Rate	$D^{\text{1D}}_{\text{RR2}}$	$D^{\text{1D}}_{\text{RR4}}$	s_2	s_4	$E^{\text{1D}}_{\text{RR2}}$	$E^{\text{1D}}_{\text{RR4}}$
8	0.8500	27	21	5	9	1.500	2.667
8	0.8750	48	24	10	11	2.333	3.250
8	0.8900	138	48	31	25	5.833	7.708
8	0.9000	–	60	–	32	–	10.000
16	0.8900	48	28	7	9	1.625	2.250
16	0.9050	64	32	10	11	2.000	2.688
16	0.9150	144	48	24	18	3.750	4.333
16	0.9200	288	64	49	25	6.875	6.031
16	0.9300	–	100	–	41	–	9.970

尽管如此，随着 q 增大，一维二进制 RR 方案的错误传播因子会显著降低。例如，对于 $q = 16$ 和 $m = 21$， $E^{\text{1D}}_{\text{RR2}} = 2.625$；考虑到码长，这一数值相当小。

表 III 以不同方式比较了一维二进制编码方案与一维四元 RR-LOCO 编码方案。具体而言，我们固定归一化码率，并找出这两种编码方案达到该目标码率所需的最小编码数据量和最低复杂度（加法器位宽），以及与之相关的最小错误传播。[6] 每当二进制编码方案无法达到该码率时，表中便使用符号“–”。表 III 的主要结论如下：

- 对于 $q = 8$ 和 $q = 16$，在所有目标码率下，四元编码方案所需的编码数据量（长度更短）都少于二进制编码方案。随着码率提高，四元编码方案的优势也越来越明显。

- 在较低码率下，二进制编码方案的复杂度低于四元编码方案。然而，在码率为 ≥ 0.8900 （对应 $q = 8$ ）和 ≥ 0.9150 （对应 $q = 16$）时，四元编码方案在复杂度方面更具优势。

- 正如预期，通常二进制编码方案造成的错误传播较少，因为 LOCO 编码仅在一页上进行。然而，在较高码率和较高 q 下，

[6] Ach在此处达到期望码率，意味着归一化码率达到或超过该期望码率。

四元编码方案的竞争力相当强，令人惊讶的是，在码率为 0.9200 和 $q = 16$ 时，它造成的错误传播已经更少。

一维和二维 RR 编码方案可以在同一器件中使用，但适用于不同的寿命阶段。器件较新时，或在经历适度数量的编程/擦除（P/E）周期之前，可以使用一维二元或四元 RR-LOCO 编码方案；而器件老化后，则可以使用二维 RR 约束编码方案。此时，抑制两个方向上的易出错模式可能会产生明显影响，而相应的码率损失也可能可以接受。不过，第六节表明，至少对于我们使用的 TLC 闪存器件，这种性能差异很小。该节还表明，一维二元 RR-LOCO 编码方案与一维四元 RR-LOCO 编码方案之间的性能差异可以忽略不计。

备注 3。 *[16]* 中介绍了一种适用于 *MLC* 闪存、可实现页分离的思路。然而，其提供的码率仅为 0.7500，显著低于我们的一维二进制 *RR* 编码方案在 *MLC* 中提供的码率。 *[17]* 中介绍了另一种适用于 *TLC* 闪存、可实现页分离的思路。然而，该方法仅以启发式方式处理电平模式707。

VI. TLC闪存上的实验结果

为评估所提出的 RR 约束编码方案的性能，我们在一款商用 1X-nm TLC 闪存芯片的多个闪存块上进行了编程/擦除（P/E）循环实验，具体如下：

1) 擦除待测闪存块。2) 使用数据对待测闪存块的所有页面进行编程。对于未编码实验，在每个 P/E 周期编程伪随机数据。对于 RR 实验，在每个 P/E 周期编程满足 RR 约束的预先准备好的数据。3) 对于 RR 实验中的每个后续 P/E 周期，对数据进行“轮换”，使原先写入页面 i 的数据写入页面 $(i + 1)$，并将最后一页的数据绕回到第一页。4) 每经过 100 个 P/E 周期，记录比特错误并计算信道比特误码率（BER）。

P/E 循环实验在室温下连续进行，擦除、编程和读取操作之间没有等待时间。

闪存器件使用的格雷映射可能因制造商和产品代际而异。在我们的初步工作 [1],中，我们根据器件映射修改了禁用的二进制模式，

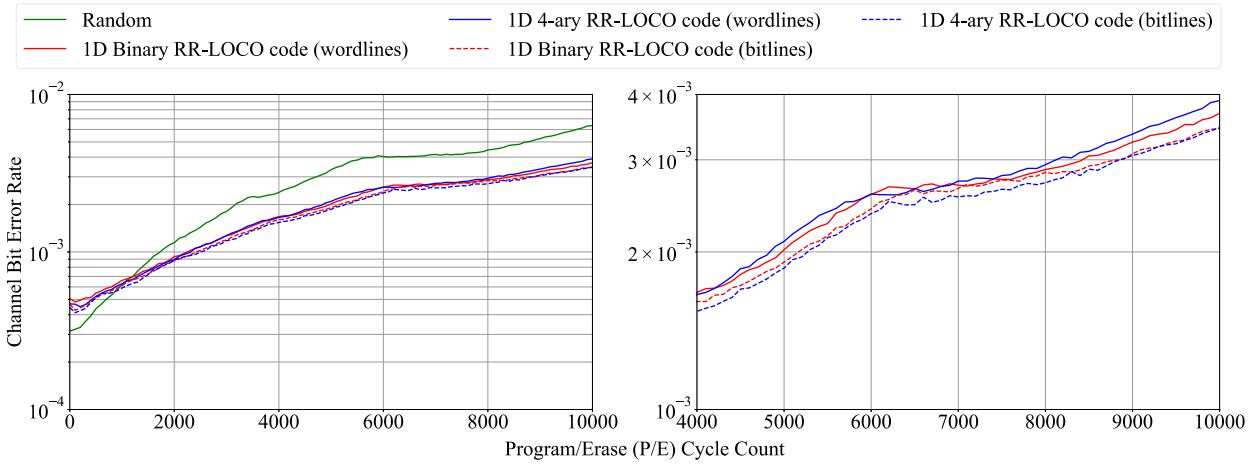


图4。（左）从编程/擦除周期0到编程/擦除周期10，000，所有页面分别采用随机数据、沿字线或位线排列的一维二进制 RR-LOCO编码数据，以及沿字线或位线排列的一维四元RR-LOCO码编码数据进行编程时，测得的平均信道误码率对比。（右）从编程/擦除周期4，000到编程/擦除周期10，000，排除随机数据后的测得平均信道误码率对比。

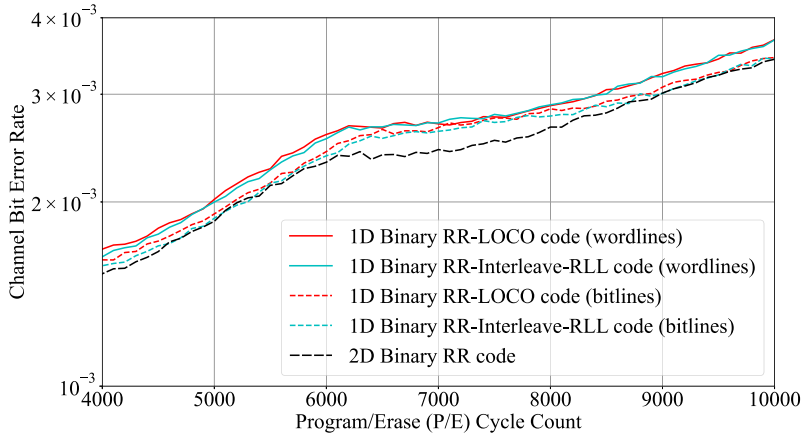


图5。从编程/擦除周期4，000到编程/擦除周期10，000，沿字线或位线排列的一维二进制RR-LOCO码、沿字线或位线排列的一维二进制交织RLL-(0，1)码，以及二维二进制 RR 码的测得平均信道误码率对比。

使得在每条字线上对一个页进行 RR 编码能够消除 $\mathcal{L}_q$ 中大多数会导致最严重单元间干扰的模式（见备注 1）。

本文中，采用RAGM映射，由此处所述RR编码器生成的8元编码电平序列，根据器件格雷映射转换为TLC闪存中下页、中页和上页对应的二进制序列。因此，存储在存储器中的8元电平序列恰好就是RR编码后的电平序列（每个单元都被编程到 $\{0, 1, \ldots, q-1\}$中的一个电平）。

图4左侧子图展示了采用伪随机数据、沿字线或位线使用码率为24:36的一维二进制RR-LOCO码，以及沿字线或位线使用码率为20:12比特/符号的一维四元RR-LOCO码时，从编程/擦除周期0到编程/擦除周期10，000的信道误码率。图4右侧子图更详细地展示了这些情况下从编程/擦除周期4，000到编程/擦除周期10，000的信道误码率。请注意，二进制RR码和四元RR码的总码率相同：$R_{\mathrm{RR2}}^{\mathrm{1D}} = 8/9 \approx 0.8889$ 根据（49）以及 $R_{\mathrm{RR4}}^{\mathrm{1D}} = 8/9 \approx 0.8889$ 根据（50）。因此，一维二进制编码方案达到约99%（96%）的容量$C_{\mathrm{RR2}}^{\mathrm{1D}}$ （$C_{\mathcal{L}_q}^{\mathrm{1D}}$），而一维四元编码方案达到约96%的容量$C_{\mathrm{RR4}}^{\mathrm{1D}}$。

如图4所示，在约1,200个编程/擦除周期之前，未编码性能优于二进制RR码和四元RR码；此后则明显变差。在编程/擦除循环的后期，单元间干扰变得严重，RR码的性能优于未编码设置。具体而言，在信道误码率为 2×10^{-3}时，沿字线使用一维二进制RR-LOCO码可使器件寿命增加约1,800个编程/擦除周期，相当于57%的寿命增益；在信道误码率为 3×10^{-3}时，可获得约3,700个编程/擦除周期的增益，相当于79%的寿命增益。如图4右侧子图所示，在2,000至8,000个编程/擦除周期之间，沿字线的一维二进制RR码的误码率与四元RR码几乎相同。当编程/擦除周期数大于8,000时，沿字线的一维二进制RR码的误码率略优于一维四元RR码。特别地，当信道误码率为 3×10^{-3}时，沿字线的一维二进制RR码所提供的寿命比沿字线的一维四元RR码长约300个编程/擦除周期。沿位线方向，一维四元RR-LOCO码的性能通常优于一维二进制RR-LOCO码，但二者差距较小。自第6,300个编程/擦除周期至第8,300个编程/擦除周期，一维四元RR-LOCO码的优势最为明显。

图 5 比较了高编程/擦除周期下不同二进制 RR 码实现的误码率性能：沿字线或位线方向使用的 24:36 一维二进制 RR-LOCO 码、沿字线或位线方向使用的一维二进制交织 12:18 RLL $(d,\ k) = (0,\ 1)$ 码（交织后总块长为 36），以及二维二进制 RR 码。利用 (48)，我们得到 $R_{\mathrm{RR2}}^{\mathrm{2D}} = 5/6 \approx 0.8333$。因此，二维编码方案达到容量 $C_{\mathrm{RR2}}^{\mathrm{2D}}$ ($C_{\mathcal{L}_q}^{\mathrm{1D}}$) 的约 93%（90%）。

从图 5 中可以观察到以下情况：沿位线方向的一维RR编码方案性能略优于沿字线方向的一维RR编码方案；二维 RR 受限码的性能优于沿任一方向的

一维RR码；沿相同方向的一维RR编码方案性能相近。例如，当信道误码率为 2×10^{-3} 时，二维二进制RR编码相比沿位线的一维二进制RR-LOCO编码，可将寿命延长100 个编程 / 擦除周期；相比沿字线的一维二进制 RR-LOCO 编码，可延长 300 个编程 / 擦除周期。当信道误码率为 3×10^{-3} 且闪存器件的磨损状况严重时，二维二进制RR编码相比沿位线的一维二进制RR-LOCO编码，可多承受约 200 个编程/擦除周期；相比沿字线的一维二进制RR-LOCO编码，可多承受约 600 个编程/擦除周期。

这些测量结果证实了四元RR码所宣称的一些实际优势。一维二进制RR-LOCO码和一维四元RR-LOCO码沿字线和位线方向的性能结果非常相似，而且设计码的总码率相同（包括桥接符号）。一维四元RR-LOCO码的总块长较短，每个编码页对应12比特（10个符号加2个桥接符号）；相比之下，一维二进制RR-LOCO码的编码页总块长为36比特。此外，在码设计中，一维四元RR-LOCO编码器使用的加法器位宽为18比特，而一维二进制RR-LOCO编码器所需的加法器位宽为24比特。

考察一维二进制和一维四元RR约束所产生的电平概率，可以直观地理解图4和图5中的实验结果。在RLL（d，k）=(0，1)约束下，二进制符号0和1的概率分别约为0.2764和0.7236 [25]。渐近而言，这意味着对应于 $\mathcal{V}_0 = \{4,5,6,7\}$ 和 $\mathcal{V}_1 = \{0,1,2,3\}$ 中电平的各个符号的概率分别约为0.0691和0.1809。根据图3所示、禁止 $\mathcal{R}_4$中模式的四元约束的FSTD，我们发现对应于 $\mathcal{W}_0 = \{6,7\}$、$\mathcal{W}_1 = \{4,5\}$、 $\mathcal{W}_2 = \{2,3\}$和 $\mathcal{W}_3 = \{0,1\}$ 中电平的各个符号的概率分别约为0.0787、0.1030、0.1591和0.1591。桥接符号会使这些概率略有变化，进一步提高对应于 $\{0,1,2,3\}$ 中电平的符号相对于对应于 $\{4,5,6,7\}$中电平的符号的概率。这些概率与未编码随机数据的情况形成对比；在未编码随机数据中，每个符号/电平的概率均为 $1/8 = 0.125$。

修改后的符号概率有助于解释一维二进制 RR 码在字线方向和位线方向上的相对性能，以及二维 RR 码的相对性能。沿字线方向应用一维二进制 RR编码也会间接降低位线方向上有害模式的概率，反之亦然。当位线方向存在更严重的单元间干扰时，这会减小位线编码相对于字线编码的预期优势。

同样，与任一方向上的一维编码相比，二维编码的优势也没有预期那么明显（即使不考虑二维编码所带来的码率惩罚）。

我们指出，所设计的码效率较高，码率相当接近容量；此外，在闪存中写入的数据里观察到的符号概率和模式概率，也接近上文提到的理论值。

如果该码消除的电平模式（尤其是易受ICI影响的模式）并非器件寿命早期误差的重要来源，那么图4中观察到的交叉行为便可以得到解释。与未编码设置相比，二元RR编码显著改变了电平概率，可能会提高某些剩余电平模式出现的概率；这些模式会因其他效应导致错误，从而在较低的编程/擦除周期数下增加其对误码率的贡献。一种避免这种情况的建议是在交叉点前后采用不同的约束。可以利用LOCO码设计的可重配置性，并使用机器学习模块识别器件状态，根据该状态在适当时机引导从一种码切换到另一种码。此外，我们还指出，如 [30], 中所提出的机器学习建模，可用于表征闪存器件的时空ICI效应，并为优化RR-LOCO码设计提供工具。

VII. 结论

本文介绍了适用于现代闪存器件的读写（RR）约束编码方案。RR 编码方案能够消除易引发单元间干扰（ICI）错误的模式，同时支持系统编码器和解码器实现、高总码率，以及数据恢复时的页分离。我们分析了一维二进制读写LOCO码、一维四进制读写LOCO码和一种二维二进制 RR 码的特性。这三种 RR 编码方案各有优势；我们建议根据器件寿命不同阶段的系统需求，确定最适合采用的一种或多种方案。实验结果表明，在商用闪存器件中，这些方案显著提升了编程/擦除周期寿命。未来的工作包括将低密度奇偶校验码 [31] 与 RR 编码方案结合，并开发机器学习辅助的可重构读写编码方案，以最大限度延长闪存器件的寿命。

参考文献

[1] A. Hareedy、S. Zheng、P. Siegel 和 R. Calderbank，“面向现代闪存器件的读写约束编码”，载于*IEEE*国际通信会议（*ICC*）论文集，韩国首尔，2022年5月，第1–6页。

[2] C. E. Shannon，“通信的数学理论”，《贝尔系统技术期刊》，第27卷，1948年10月。[3] D. T. Tang 和 R. L. Bahl，“一类受限无噪信道的分组码”，《信息与控制》，第17卷，第5期，第436–461页，1970年。[4] T. Cover，“枚举式信源编码”，《IEEE信息论汇刊》，第19卷，第1期，第73–77页，1973年1月。[5] P. A. Franaszek，“游程受限编码的序列状态方法”，《IBM研究与开发期刊》，第14卷，第4期，第376–383页，1970年7月。[6] R. Adler、D. Coppersmith 和 M. Hassner，“滑动分组码算法——符号动力学在信息论中的应用”，《IEEE信息论汇刊》，第29卷，第1期，第5–22页，1983年1月。[7] K. A. S. Immink、P. H. Siegel 和 J. K. Wolf，“数字记录器的编码”，《IEEE信息论汇刊》，第44卷，第6期，第2260–2299页，1998年10月。[8] A. Hareedy 和 R. Calderbank，“LOCO码：字典序受限码”，《IEEE信息论汇刊》，第66卷，第6期，第3572–3589页，2020年6月。[9] B. Vasic 和 E. Kurtas，《磁记录系统的编码与信号处理》。CRC出版社，2005年。[10] S. Zheng、Y. Liu 和 P. H. Siegel，“PR-NN：面向编码部分响应信道的基于循环神经网络的检测”，《IEEE精选通信领域期刊》，第39卷，第7期，第1967–1982页，2021年7月。[11] B. Dabak、A. Hareedy 和 R. Calderbank，“二维磁记录的非二元受限码”，《IEEE磁学汇刊》，第56卷，第11期，第1–10页，2020年11月。[12] R. Wood、M. Williams、A. Kavcic 和 J. Miles，“在传统介质上实现每平方英寸10太比特磁记录的可行性”，《IEEE磁学汇刊》，第45卷，第2期，第917–923页，2009年2月。[13] K. A. S. Immink，“具有光学读取功能的数字音频光盘调制系统”，载于IEEE国际声学、语音与信号处理会议（ICASSP）论文集，佐治亚州亚特兰大，美国，1981年3月至4月，第587–589页。[14] J. Saadé、A. Goulahsen、A. Picco、J. Huloux 和 F. Pétrot，“低开销、直流平衡的游程受限线路编码”，载于IEEE第19届信号与电源完整性研讨会（SPI）论文集，德国柏林，2015年5月，第1–4页。[15] J.-D. Lee、S.-H. Hur 和 J.-D. Choi，“浮栅干扰对NAND闪存存储单元操作的影响”，《IEEE电子器件快报》，第23卷，第5期，第264–266页，2002年5月。[16] V. Taranalli、H. Uchikawa 和 P. H. Siegel，“多级单元闪存中的误差分析与单元间干扰抑制”，载于IEEE国际通信会议（ICC）论文集，英国伦敦，2015年6月，第271–276页。[17] R. Motwani，“用于抑制闪存中浮栅间耦合的分层受限编码”，载于IEEE全球通信会议（GLOBECOM）论文集，美国得克萨斯州休斯敦，2011年12月，第1–5页。[18] Y. M. Chee、J. Chrisnata、H. M. Kiah、S. Ling、T. T. Nguyen 和 V. K. Vu，“抑制闪存中单元间干扰和电荷泄漏的达容量码”，《IEEE信息论汇刊》，第65卷，第6期，第3702–3712页，2019年6月。[19] A. Hareedy、B. Dabak 和 R. Calderbank，“管理器件生命周期：适用于M/T/Q/P-LC闪存的可重构受限码”，《IEEE信息论汇刊》，第67卷，第1期，第282–295页，2021年1月。[20] V. Braun 和 K. A. S. Immink，“无直流游程受限序列的枚举编码技术”，《IEEE通信汇刊》，第48卷，第12期，第2024–2031页，2000年12月。[21] A. Hareedy 和 R. Calderbank，“非对称LOCO码：闪存的受限码”，载于第57届Allerton通信、控制与计算会议论文集，美国伊利诺伊州蒙蒂塞洛，2019年9月，第124–131页。[22] A. Hareedy、B. Dabak 和 R. Calderbank，“模式的秘密算术：基于字典序索引设计受限码的一般方法”，《IEEE信息论汇刊》，第68卷，第9期，第5747–5778页，2022年9月。[23] J. Centers、X. Tan、A. Hareedy 和 R. Calderbank，“基于电平信号传输的受限码功率谱：克服有限长度挑战”，《IEEE通信汇刊》，第69卷，第8期，第4971–4986页，2021年8月。

[24] S. Zheng、C. Ho、W. Peng 和 P. H. Siegel，“利用条件生成网络对闪存信道进行时空建模”，已被*2023*年欧洲设计、自动化与测试会议暨展览（*DATE*）论文集接收，安特卫普，比利时。[25] P. H. Siegel，“多级闪存的受限码”，发表于北美信息论学校（帕多瓦尼讲座），拉霍亚，加利福尼亚州，2015年8月12日。可访问：http://cmrr-star.ucsd.edu/static/presentations/Padovani_Lecture_NASIT_Website.pdf。视频：https://www.youtube.com/watch?v=FCv2PJryUr4。[26] R. Laroia、N. Farvardin 和 S. A. Tretter，“关于多维星座的最优整形”，*IEEE*信息论汇刊，第40卷，第4期，第1044–1056页，1994年7月。[27] N. Kashyap、R. M. Roth 和 P. H. Siegel，“计数受限无ICI系统的容量”，载于*IEEE*国际信息论研讨会，法国巴黎，2019年7月，第1592–1596页。[28] A. Kato 和 K. Zeger，“关于二维游程受限信道的容量”，*IEEE*信息论汇刊，第45卷，第5期，第1527–1540页，1999年7月。[29] B. H. Marcus、P. H. Siegel 和 J. K. Wolf，“用于数据存储的有限状态调制码”，载于*IEEE J. Sel. Areas Commun.*，第10卷，第1期，第5–37页，1992年1月。[30] S. Zheng 和 P. H. Siegel，“通过机器学习进行码感知存储信道建模”，载于*IEEE*信息论研讨会（*ITW*）论文集，印度孟买，2022年11月，第196–201页。[31] A. Hareedy、R. Kuditipudi 和 R. Calderbank，“最小化多维图码中有害对象的数量”，*IEEE*通信汇刊，第68卷，第9期，第5299–5312页，2020年9月。