Correcting Errors in MLCs with Bit-fixing Coding

Anxiao (Andrew) Jiang[†], Yue Li[†], and Jehoshua Bruck[*]
[†]*Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843*
[*]*Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125*
[†]{*ajiang, yli*}*@cse.tamu.edu* [*]*bruck@caltech.edu*

I. Introduction

Multi-level cell (MLC) nonvolatile memories have become increasing important for storage in recent years. A well-known example is flash memory, where cells with $q = 4$ and 8 levels are already widely used. It is important to design error-correcting codes (ECCs) that consider the many properties of MLC memories. Errors often have limited magnitudes and non-symmetric distributions, due to the memories' unique disturb and noise mechanisms. Also, current MLCs are restricted by q, the number of levels, being a power of 2. Coding schemes that can map cells to binary codes conveniently for an arbitrary number of levels are worth studying. All these will be addressed in this paper.

There are different approaches to map cell levels to binary codes when q is a power of 2, including binary representation and Gray codes. Consider n cells; and for $i = 1, \cdots, n$, let $\ell_i \in \{0, 1, \cdots, q-1\}$ be the level of the ith cell. Let $m = \log_2 q$. And let $\mathcal{B}_m(\ell_i) \triangleq (b_{i,m-1}, \cdots, b_{i,1}, b_{i,0}) \in \{0,1\}^m$ be the binary representation of ℓ_i, namely, $\ell_i = \sum_{j=0}^{m-1} b_{i,j} \cdot 2^j$. Since the m bits in a cell have different error probabilities, in a basic binary-representation approach, m ECCs of different rates are used. Specifically, for $j = 0, 1, \cdots, m-1$, we let $(b_{1,j}, \cdots, b_{n,j})$ be a separate ECC. To further reduce error probabilities, a more common approach is to represent the bits in a cell using Gray codes, and then apply m ECCs.

In this paper, we propose an alternative coding scheme named *bit-fixing code*. Its main idea is to sequentially correct the bits in the binary representation of errors. And it can be generalized to more numeral systems. When $q = 2^m$, let $\varepsilon_i \in \{-\ell_i, \cdots, 0, \cdots, q-1-\ell_i\}$ denote the additive error in the ith cell's level ℓ_i, and let $\mathcal{B}_m(\varepsilon_i \mod q) \triangleq (e_{i,m-1}, \cdots, e_{i,0}) \in \{0,1\}^m$ be the binary representation of $\varepsilon_i \mod q$. For $j = 0, \cdots, m-1$, let $(b_{1,j}, \cdots, b_{n,j})$ be a binary ECC $\mathcal{C}j$. The scheme has the nice property that the error bits $(e_{1,j}, \cdots, e_{n,j})$ only affect the code $\mathcal{C}_j$. (Note that this property does not hold for the binary-representation scheme introduced above.) That enables us to allocate redundancy appropriately and decode $\mathcal{C}_0$, $\cdots$, $\mathcal{C}_{m-1}$ sequentially.

The bit-fixing coding scheme can be applied to arbitrary error distributions, including both asymmetric and bidirectional errors. It can be generalized from the binary representation to many more numeral representations, including k-ary numbers (for any integer $k \geq 2$) and mixed-radix numeral systems such as factoradic systems. It can also be extended to an arbitrary number of cell levels, which means q can be any integer instead of a power of 2 and binary codes can still be used. The coding scheme in fact contains the ECC for asymmetric errors of limited magnitude in [1] as a special case. It is also related to the codes in [2], but is more specific in its construction and more general in various ways. It can be applied not only to storage but also to amplitude-modulation communication systems. Due to space limitation, readers are referred to the full version [3] of this paper for more details.

This work was supported in part by the NSF CAREER Award CCF-0747415, an NSF-NRI award, and NSF grant CCF-1217944.

II. Bit-fixing Coding Scheme

We present the bit-fixing coding scheme for the case where q is a power of 2 and binary representations are used. The extension to general cases is presented in [3].

Construction 1. Encoding of Bit-fixing Scheme

For $j = 0, 1, \cdots, m-1$, let $\mathcal{C}_j$ be an (n, k_j) binary ECC that can correct t_j errors. We store $k_0 + k_1 + \cdots + k_{m-1}$ information bits in n cells of $q = 2^m$ levels as follows. First, we partition the information bits into m chunks, where for $j = 0, \cdots, m-1$, the jth chunk has k_j information bits: $\mathbf{d}_j = (d_{j,1}, d_{j,2}, \cdots, d_{j,k_j})$. Next, for $j = 0, \cdots, m-1$, we use $\mathcal{C}_j$ to encode $\mathbf{d}_j$ into a codeword $\mathbf{b}_j = (b_{1,j}, b_{2,j}, \cdots, b_{n,j})$. Then, for $i = 1, \cdots, n$, let $\ell_i = \sum_{j=0}^{m-1} b_{i,j} \cdot 2^j$, and we write the ith cell's level as ℓ_i.

After cells are written, additive errors $\varepsilon_1, \cdots, \varepsilon_n$ will appear and change cell levels to $\ell'_1 = \ell_1 + \varepsilon_1, \cdots, \ell'_n = \ell_n + \varepsilon_n$.

Construction 2. Decoding of Bit-fixing Scheme

Let $\ell'_1, \cdots, \ell'_n$ be the noisy cell levels we read. As the initialization step, for $i = 1, \cdots, n$, let $\hat{\ell}_i = \ell'_i$.

For $j = 0, 1, \cdots, m-1$, carry out these three steps:

1) *For $i = 1, \cdots, n$, let $(\hat{b}_{i,m-1}, \cdots, \hat{b}_{i,1}, \hat{b}_{i,0}) = \mathcal{B}_m(\hat{\ell}_i)$ be the binary representation of the estimated cell level $\hat{\ell}_i$.*
2) *Use code $\mathcal{C}_j$ to decode the codeword $(\hat{b}_{1,j}, \cdots, \hat{b}_{n,j})$, and let $(\hat{e}_{1,j}, \cdots, \hat{e}_{n,j})$ be the discovered error vector. (That is, the recovered codeword is $(\hat{b}_{1,j} \oplus \hat{e}_{1,j}, \cdots, \hat{b}_{n,j} \oplus \hat{e}_{n,j})$, where "$\oplus$" is the exclusive-OR operation.)*
3) *For $i = 1, \cdots, n$, update the estimated cell level $\hat{\ell}_i$ as follows: $\hat{\ell}_i \leftarrow \left(\hat{\ell}_i - \hat{e}_{i,j} \cdot 2^j \mod q\right)$.*

Now $\hat{\ell}_1, \cdots, \hat{\ell}_n$ are our recovered cell levels. From them, the information bits can be readily obtained.

Example 3. *Consider n cells of $q = 8$ levels. Then $m = \log_2 q = 3$. Assume $\mathcal{C}_0, \mathcal{C}_1, \mathcal{C}_2$ can correct no less than 3, 1, and 2 errors, respectively. Without loss of generality, suppose that after cells are written, errors appear in cells 1, 2 and 3, respectively. Let $\ell_1 = 3, \ell_2 = 1, \ell_3 = 2$ be their original levels, and let $\varepsilon_1 = 1, \varepsilon_2 = 5, \varepsilon_3 = -1$ be their errors. Then their noisy levels are $\ell'_1 = 4, \ell'_2 = 6, \ell'_3 = 1$, respectively.*

	Cell 1	Cell 2	Cell 3
Original level	3: (0,1,1)	1: (0,0,1)	2: (0,1,0)
Error	1: (0,0,1)	5: (1,0,1)	-1: (1,1,1)
Noisy level	4: (1,0,0)	6: (1,1,0)	1: (0,0,1)
Level after decoding $\mathcal{C}_0$	3: (0,1,1)	5: (1,0,1)	0: (0,0,0)
Level after decoding $\mathcal{C}_1$	3: (0,1,1)	5: (1,0,1)	6: (1,1,0)
Level after decoding $\mathcal{C}_2$	3: (0,1,1)	1: (0,0,1)	2: (0,1,0)

In the decoding process, we first decode $\mathcal{C}_0$, where the noisy codeword is $(0,0,1,\cdots)$. (It is because the least-significant bits (LSB) of $(\mathcal{B}_m(\ell'_1), \mathcal{B}_m(\ell'_2), \mathcal{B}_m(\ell'_3), \cdots)$ are $(0,0,1,\cdots)$.) By decoding it, we find its error vector $(e_{1,0}, e_{2,0}, e_{3,0}, \cdots) = (1,1,1,\cdots)$. So we change the cell levels to $(4 - e_{1,0} \mod 8, 6 - e_{2,0} \mod 8, 1 - e_{3,0} \mod 8) = (3,5,0)$. Next, we decode $\mathcal{C}_1$, where the noisy codeword is $(1,0,0,\cdots)$. (It is because the middle bits of $(\mathcal{B}_m(3), \mathcal{B}_m(5), \mathcal{B}_m(0), \cdots)$ are $(1,0,0,\cdots)$.) By decoding it, we find its error vector $(e_{1,1}, e_{2,1}, e_{3,1}, \cdots) = (0,0,1,\cdots)$. So we change the cell levels to $(3 - e_{1,1} \cdot 2 \mod 8, 5 - e_{2,1} \cdot 2 \mod 8, 0 - e_{3,1} \cdot 2 \mod 8) = (3,5,6)$. We then decode $\mathcal{C}_2$, where the noisy codeword is $(0,1,1,\cdots)$. (It is because the most-significant bits (MSB) of $(\mathcal{B}_m(3), \mathcal{B}_m(5), \mathcal{B}_m(6), \cdots)$ are $(0,1,1,\cdots)$.) By decoding it, we find its error vector $(e_{1,2}, e_{2,2}, e_{3,2}, \cdots) = (0,1,1,\cdots)$. So we change the cell levels to $(3 - e_{1,2} \cdot 2^2 \mod 8, 5 - e_{2,2} \cdot 2^2 \mod 8, 6 - e_{3,2} \cdot 2^2 \mod 8) = (3,1,2)$. They are the original cell levels, from which we can recover information bits. □

Given a vector $\mathbf{v} = (v_{k-1}, \cdots, v_1, v_0) \in \{0,1\}^k$, define its *support* as $support(\mathbf{v}) \triangleq \{i | i \in \{0,1,\cdots,k-1\}, v_i = 1\}$. Given $i \in \{0,1,\cdots,m-1\}$, we define the *cross of i* as $cross_m(i) \triangleq \{j | j \in \{0,1,\cdots,2^m-1\}, i \in support(\mathcal{B}_m(j))\}$. Namely, $cross_m(i)$ is the set of integers in $\{0,1,\cdots,2^m-1\}$ whose binary representations have 1 in the ith position. For $i = 0,1,\cdots,q-1$, define $\gamma_i \triangleq \big|\{j | j \in \{1,\cdots,n\}, \varepsilon_j \equiv i \mod q\}\big|$. That is, there are γ_i cells with errors of magnitude exactly $i \pmod q$.

Theorem 4. *The bit-fixing coding scheme can recover all information bits if for $j = 0,1,\cdots,m-1$, the binary error-correcting code $\mathcal{C}_j$ can correct $\sum_{k \in cross_m(j)} \gamma_k$ binary errors.*

III. Optimal Labeling of Cell-Levels

We present a new technique, *labeling of cell levels*, for better performance. Let $\pi : \{0,1,\cdots,q-1\} \to \{0,1,\cdots,q-1\}$ be a permutation function that maps every physical state s to its corresponding level $\pi(s)$. Let $s \in \{0,1,\cdots,q-1\}$ denote the original physical state of a cell, let $\delta \in \{-s,\cdots,0,\cdots,q-1-s\}$ denote the physical error in it, and let $s' = s + \delta$ denote its noisy physical state. Correspondingly, let $\ell = \pi(s)$ denote its original level, let $\ell' = \pi(s')$ denote its noisy level, and let $\varepsilon = \ell' - \ell$ denote the *error in the cell level*. The objective of a good labeling is to decrease the number of bit-errors in $\mathcal{C}_0, \cdots, \mathcal{C}_{m-1}$ caused by physical errors, and maximize the overall code rate. In the following, for simplicity, assume $q = 2^m$ and binary representations are used.

Construction 5. A Method for Cell-Level Labeling *Let $\pi(0) = 0$. For $i = 1,2,\cdots,m$ and $j = 2^{i-1}, 2^{i-1}+1, \cdots, 2^i - 1$, let $\pi(j) = \pi(j - 2^{i-1}) + 2^{m-i}$.*

Theorem 6. *Construction 5 minimizes the total number of bit-errors introduced by magnitude-one (including +1 and -1) physical errors.*

IV. Code Rate Comparison

We have evaluated extensively the code rates of the bit-fixing scheme, and compared them to the commonly used basic binary-representation scheme and the Gray-code based scheme. The following example is an illustration of their performance. Let errors be in the range $[-L^-, L^+]$, which is modeled as follows. Let there be $n \to \infty$ cells of $q = 2^m$ levels, whose errors are i.i.d. Let $p \in [0,1]$ be a parameter, let L^+ and L^- be non-negative integers (with $L^+ + L^- > 0$), and let $\tilde{\delta} \in \{-L^-, \cdots, 0, \cdots, L^+\}$ be a random variable with this distribution: $Pr\{\tilde{\delta} = 0\} = 1 - p$; and $\forall\ i \in \{-L^-, \cdots, -1, 1, \cdots, L^+\}$, $Pr\{\tilde{\delta} = i\} = p/(L^- + L^+)$. For a cell of original physical state $s \in \{0,1,\cdots,q-1\}$, the noise $\tilde{\delta}$ is added to it. If $\tilde{\delta} > 0$, the noisy physical level s' becomes $\min\{s + \tilde{\delta}, q-1\}$; otherwise, s' becomes $\max\{s + \tilde{\delta}, 0\}$. (It is modeled this way because a cell's state must be in $\{0,1,\cdots,q-1\}$. And given a labeling π, the error changes the level from $\ell = \pi(s)$ to $\ell' = \pi(s')$.) We consider the practical case where $Pr\{b_i = 0\} = 1/2$ for $i = 0,1,\cdots,m-1$. Some results on achievable rates are shown in Fig. 1, with $q = 16$, $p = 0.01$, and L^+ changing from 1 to 6. It can be seen that the bit-fixing coding scheme compares favorably with the basic binary-representation scheme, and is comparable to the Gray-code based scheme.

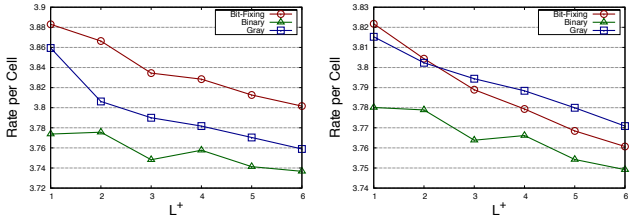

Fig. 1. Comparison of achievable rates (number of stored bits per cell). Here $q = 16$, $p = 0.01$, and L^+ increases from 1 to 6. (a) Asymmetric errors, where $L^- = 0$. (b) Bidirectional errors, where $L^- = 3$.

References

[1] Y. Cassuto, M. Schwartz, V. Bohossian and J. Bruck, "Codes for asymmetric limited-magnitude errors with application to multilevel flash memories," in *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1582–95, 2010.

[2] E. Yaakobi, P. H. Siegel, A. Vardy, and J. K. Wolf, "On codes that correct asymmetric errors with graded magnitude distribution," in *Proc. ISIT*. pp. 1021–1025, St. Petersburg, Russia, August 2011.

[3] A. Jiang, Y. Li, and J. Bruck, "Bit-fixing Codes for Multi-level Cells," in *Proc. IEEE Information Theory Workshop (ITW)*, 2012.