Error-Correcting WOM Constructions through Concatenation and Joint Design

Amit Solomon and **Yuval Cassuto**
Viterbi Department of Electrical Engineering, Technion – Israel Institute of Technology
samitsolomon@gmail.com,ycassuto@ee.technion.ac.il

***Abstract*— We construct error-correcting WOM (write-once memory) codes that can correct any specified number of errors in q-level memories. The constructions use suitably designed short q-ary WOM codes and concatenate them with outer error-correcting codes over different alphabets, using suitably designed mappings. With a new storage-efficiency measure we call EC-rate, we show that for common error types the codes save redundancy and implementation complexity over straightforward concatenation.**

I. Introduction

Flash-based non-volatile memories (NVM) are the storage media of choice in most modern information applications, thanks to their fast access and growing densities. However, the Flash technology suffers from the major impediment of not being able to update data in-place. Because removing charges from memory cells cannot be done at a fine granularity, it is not possible to update written data without first erasing a very large data unit. As a result, write performance is degraded and the wear of cells is accelerated. It has been demonstrated and recognized [10], [13] that *WOM (write-once memory) codes* [11] hold promise to mitigate this access limitation by allowing to update the logical data multiple times without need to physically remove charges from the cells. While WOM codes can support a more flexible write access to Flash, a concern is raised about their effect on data reliability. With WOM codes, cells are written multiple times between erases, and are accessed in a less predictable order than when written only once without WOM. These two effects may increase the severity of inter-cell interference, and degrade reliability if not properly addressed.

Our objective in this paper is to facilitate the reliability of WOM codes by making them resilient to errors from noise and interference. Our method is to combine WOM codes with error-correcting (EC) codes to get guaranteed error correction with flexible parameters. In general combining WOM codes with EC codes is a non-trivial task because the EC encoder may not respect the WOM constraints, and the WOM decoder may cause error propagation affecting a large number of EC symbols. While there have been some works addressing EC+WOM combination [15], [12], [8], the problem is yet to be adequately solved for practical deployment in Flash. In particular, prior to our work no scheme provides guaranteed correction of τ errors, for an arbitrary τ. One exception is a construction in [12] that lifts triple-error correcting WOM codes to general τ, but requires replicating the same WOM codeword a number of times linear in τ

Our contribution in this paper is a new scheme to combine WOM with EC codes. We use short ($n = 2$) WOM codes over the memory's q-ary alphabet (q is the number of levels supported by the physical device), and concatenate them with outer EC codes of any length N. We propose multiple code constructions for guaranteed τ error correction (for any τ), which improve storage rates over straightforward concatenation for errors common in Flash (e.g. magnitude-1 or asymmetric magnitude-1). Our main construction method is to jointly design the EC and WOM codes such that the same amount of WOM redundancy is exploited for stronger error correction. By that, our work develops concrete formal constructions building on the notion that WOM codes can have intrinsic error resilience, previously observed in [9] and further developed in [6]. To evaluate and compare our constructions, we define the *EC-rate* as a new rate measure that captures the residual redundancy added to a WOM code for error correction. Code families with concrete parameters are obtained by evaluating the EC-rate on parameters of the ubiquitously used BCH codes. In addition to concrete advantage in EC-rates, our constructions use lower-alphabet EC codes, which reduces the implementation complexity over straightforward concatenation. Our focus in the constructions is on $q = 8$-ary WOM codes corresponding to the ubiquitous TLC Flash technology, and offering $t = 4$ guaranteed writes. These WOM codes are attractive to use because their rate (without error correction) is within about 10% from the fixed-rate binary WOM capacity [5].

II. Preliminaries

We start by including definitions regarding WOM codes and error-correcting WOM codes. A q-ary fixed-rate WOM code (where q is the device's number of levels) is our basic object of study in this paper[1], and is defined next.

Definition 1. *A (n, q, t, M) (fixed-rate) **WOM code** is a code applied to a size n block of q-ary cells, and guaranteeing t writes of input size M each.*

A WOM code is specified through a pair of functions: the *decoding* and *update* functions.

Definition 2. *The decoding function is defined as $\psi : \{0,\ldots,q-1\}^n \to \{0,\ldots,M-1\}$, which maps the current levels of the n cells to the data input in the most recent write. The update function is defined as $\mu : \{0,\ldots,q-1\}^n \times \{0,\ldots,M-1\} \to \{0,\ldots,q-1\}^n$, which specifies the new cell levels as a function of the current cell levels and the new data value at the input. By the WOM requirement, the i-th cell level output by μ cannot be lower than the i-th cell level in the input, for each i.*

Definition 3. *The code's **physical state** is defined as the n q-ary levels to which the cells are currently programmed. The code's **logical state** is the data element from $\{0,\ldots,M-1\}$ returned by ψ on the current physical state.*

[1]Please refer to [4], [1] for more details on q-ary fixed-rate WOM codes.

978-1-5386-4780-6/18/$31.00©2018 IEEE

A WOM code with $n = 2$ is represented in the sequel by a 2-dimensional matrix showing its decoding function (for each physical state (c_1, c_2) the matrix holds the logical state in $\{0, \ldots, M-1\}$ corresponding to the state).

In the remainder of the paper we construct (ν, q, t, m) error-correcting WOM codes based on shorter WOM codes. A (ν, q, t, m) WOM code is called *τ-error correcting* if it can correct any error combination in up to τ (out of the ν) q-ary code symbols.

III. EC-WOM Concatenation

The route to error-correcting WOM codes we pursue in this paper is through *concatenation*. We start in this section with the most straightforward concatenated construction using an inner WOM code and an outer error-correcting (EC) code. Let $\mathcal{W}$ be a (n, q, t, M) WOM code. Denote an EC code C by $C_A[N, N-r]$ if it is defined over an alphabet of size A, it has length N, and r redundant A-ary symbols. A pictorial illustration of concatenating the EC code C with the WOM code $\mathcal{W}$ is given in Figure 1.

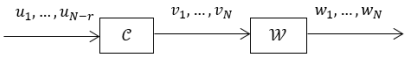

Figure 1. EC-WOM concatenation. $u_i, v_j \in GF(A)$, $w_i \in \{0, \ldots, q-1\}^n$.

We start with the simplest construction using an inner WOM code and an outer EC code.

Construction 1. *Given a (n, q, t, M) WOM code $\mathcal{W}$ and a $C_M[N, N-r]$ EC code C, the concatenated code $C\mathcal{W}_0$ is the $\left(nN, q, t, M^{N-r}\right)$ WOM code obtained by taking N copies of $\mathcal{W}$ and inputing to each a symbol of C. If C can correct τ M-ary errors, then $C\mathcal{W}_0$ is a τ-error correcting WOM code.*

The concatenation of Construction 1 is extremely simple, but it requires an EC code over the large alphabet of the WOM code, which implies high redundancy and high implementation complexity. Our next constructions show how better codes can be obtained by this concatenation method.

A. EC-rate and BCH redundancy

To compare between different EC-WOM concatenated constructions, we define the *EC-rate* of the code.

Definition 4. *Let $C\mathcal{W}$ be a $\left(nN, q, t, M^{N-r_{\text{eff}}}\right)$ WOM code with some error-correction capability, which is constructed with a (n, q, t, M) inner WOM code. We define the **EC-rate** of $C\mathcal{W}$ to be*

$$1 - \frac{r_{\text{eff}}}{N}. \tag{1}$$

The EC-rate of the code reflects its efficiency to correct errors, and ignoring its redundancy as a WOM code used for re-writing. The EC-rate definition is convenient for comparing between concatenation-based constructions that add certain EC capabilities to WOM codes with a given set of parameters (n, q, t, M). To get concrete EC-rate expressions, we assume the EC-codes used in the concatenation are taken from the family of BCH codes [2] [7], for which we (approximately) know the redundancy for all code parameters and alphabet sizes.

Definition 5. *For alphabet size A, code length N, and designed minimum distance d, define the **BCH (approximate) redundancy** as*

$$\frac{A-1}{A} \cdot (d-2) \cdot \log_A N \ \ [A\text{−ary symbols}]. \tag{2}$$

The BCH redundancy is the approximate number of parity symbols in the codeword, given in units of A-ary symbols. This definition of redundancy comes from a known approximation of the redundancy of primitive A-ary BCH codes for large N, when A is a power of a prime [14].

Example 1. *Take a $(n = 2, q = 8, t = 4, M = 8)$ WOM code $\mathcal{W}$ and concatenate it with an outer τ-error correcting (design minimum distance $d = 2\tau + 1$) BCH code $C = C_8[N, N-r]$ using Construction 1. Then we get the $\left(2N, 8, 4, 8^{N-r}\right)$ WOM code $C\mathcal{W}_0$ that is τ-error correcting.*

The outer code C used in the construction of Example 1 has BCH redundancy $\frac{7}{8} \cdot (2\tau - 1) \cdot \log_8 N$ [8−ary symbols]. When using Construction 1 with an M-ary BCH code we get the EC-rate $1 - \frac{\frac{M-1}{M}(2\tau-1)\log_M N}{N}$. Specifically for the parameters of Example 1 we get that Construction 1 attains the EC-rate of

$$1 - \frac{7}{8} \cdot \frac{(2\tau-1)\log_8 N}{N}. \tag{3}$$

IV. EC-WOM Concatenated Constructions

In this section we construct concatenated EC-WOM codes with better EC-rates than the basic Construction 1, and using lower-alphabet outer EC-codes that reduce complexity. The constructions are targeted for two common error types found in Flash, which are defined next. To define the error types, consider the alphabet of memory levels as $Q \triangleq \{0, 1, \ldots, q-1\}$.

Definition 6. *Let $c \in Q$ be the symbol written to a memory cell. The cell suffers a **mag-1** error if the read symbol is $c' \in Q$ such that $|c' - c| = 1$.*

mag-1 errors allow transitions of one level either upward or downward from the correct symbol. We similarly define the asymmetric version Amag-1.

Definition 7. *Let $c \in Q$ be the symbol written to a memory cell. The cell suffers an **Amag-1** error if the read symbol is $c' \in Q$ such that $c' - c = 1$.*

Amag-1 errors allow transitions of one level in the *upward* direction only. Amag-1 errors are common in non-volatile memories such as Flash due to inter-cell interference (ICI): adding charges to a cell due to writing to a neighboring cell. A WOM code with parameters (ν, q, t, m) is called *τ mag-1 error correcting* (resp. *τ Amag-1 error correcting*) if it can correct any combination of mag-1 (resp. Amag-1) errors in up to τ (out of the ν) q-ary code symbols. Note that in particular a τ mag-1 error correcting code is also τ Amag-1 error correcting.

To efficiently correct mag-1 and Amag-1 errors in WOM codes, in the sequel we specify the constructions in three steps: 1) specification of the inner WOM code, 2) mapping the M-ary logical alphabet of the WOM code $\{0, \ldots, M-1\}$ to a more structured alphabet, and 3) prescribing EC-codes over the structured alphabet with parameters that guarantee correcting τ errors. This construction method has the flavor of *generalized concatenation* (GC) and *coded modulation* (CM) [3] found

very useful in previous applications[2]. Unlike the simple Construction 1, in some of the improved constructions that follow we will need to go beyond specifying just the *parameters* of the inner WOM code, to designing new WOM codes with extra properties that are needed for the concatenation to work.

A. EC-WOM construction for mag-1 errors

1) Tiling WOM code: The inner WOM code we use in this construction is the known tiling-based $(n = 2, q = 8, t = 4, M = 8)$ code from [4], whose decoding function is depicted in Figure 2b (with the mapping specified in the next sub-subsection).

2) Mapping 8-ary to a product of 4-ary and binary: The concatenation requires introducing structure to the logical $M = 8$-ary alphabet as follows. We map each number $a \in \{0, \ldots, 7\}$ to its mixed radix representation (ah, al), where the upper symbol $ah \in \{0, 1, 2, 3\}$ and the lower symbol $al \in \{0, 1\}$ satisfy $2ah+al = a$, and are unique. In Figure 2 we apply this mapping to the tiling WOM code, where for clarity we map $\{0, 1, 2, 3\}$ to $\{a, b, c, d\}$. It can be seen in Figure 2b that tiling the space with the tile of Figure 2a gives the property that physical states adjacent horizontally or vertically have the opposite binary value in their lower symbol. This property will be used by the construction to combat mag-1 errors, which correspond horizontal and vertical transitions between adjacent physical states.

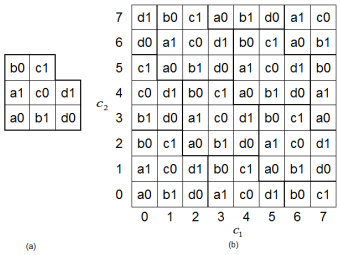

Figure 2. The tiling-based $\mathcal{W}(2, 8, 4, 8)$ code and mapping the 8-ary WOM logical symbols to 4-ary+binary symbols. (a) a single tile, and (b) the entire WOM code.

3) Outer 4-ary and binary EC codes: For the 4-ary upper symbols we take the $C_4[N, N - r_1]$ BCH code with design distance $\tau + 1$; for the binary lower symbols we take the $C_2[N, N - r_2]$ BCH code with design distance $2\tau + 1$. These choices will allow us to correct up to τ errors in the binary lower symbols and up to e' errors and e erasures in the 4-ary upper symbols, for $2e' + e \leqslant \tau$.

4) The construction:

Construction 2. *Given the* $(2, 8, 4, 8)$ *WOM code* $\mathcal{W}$ *specified with its logical mapping in Figure* 2*b and EC codes* $C_4[N, N - r_1], C_2[N, N - r_2]$ *specified in Section* IV-A.3*, the concatenated code* $\mathcal{CW}_1$ *is the* $\left(2N, 8, 4, 4^{N-r_1} \cdot 2^{N-r_2}\right)$ *WOM code obtained by taking* N *copies of* $\mathcal{W}$ *and inputing to each a symbol of* C_4 *to its upper symbol and a symbol of* C_2 *to its lower symbol.*

The essential properties of Construction 2 are given in the following proposition. The proof includes the decoding algorithm.

Proposition 1. *For any* N *and* τ*, a* $\left(2N, 8, 4, 4^{N-r_1} \cdot 2^{N-r_2}\right)$ *code* $\mathcal{CW}_1$ *obtained by Construction* 2 *is* τ *mag-1 error correcting.*

[2]GC fits when seeing the WOM as a code, and CM is appropriate when thinking about WOM as modulation.

Moreover, $\mathcal{CW}_1$ *has EC-rate*

$$1 - \frac{\left[\frac{3}{4}(\tau - 1) + \frac{1}{2}(2\tau - 1)\right] \log_8 N}{N}. \tag{4}$$

Proof: To see that $\mathcal{CW}_1$ is τ mag-1 error correcting, define by τ_1 the number of copies of $\mathcal{W}$ that suffered a mag-1 error in exactly one of their two symbols. Similarly, define by τ_2 the number of copies of $\mathcal{W}$ that suffered mag-1 errors in both their symbols. We show that any combination with $\tau_1 + 2\tau_2 \leqslant \tau$ is correctable. We first decode the binary code C_2. From the mapping seen in Figure 2, there will be bit errors in all positions corresponding to the τ_1 mag-1 errors affecting one symbol out of a $\mathcal{W}$ pair. Since $\tau_1 \leqslant \tau$, the decoder of C_2 will locate and correct all these errors. Next we decode the 4-ary code C_4, where every position with error in C_2 is erased in the input of C_4's decoder. The decoder sees τ_1 erasures and τ_2 errors, and can correct them because of the restriction $\tau_1+2\tau_2 \leqslant \tau$ (recall that C_4 has design minimum distance $\tau+1$). At this point all up to τ symbols in error are recovered by C_2 and C_4 jointly. To prove the EC-rate, we observe that

$$4^{N-r_1} \cdot 2^{N-r_2} = 8^{N-\frac{2}{3}r_1-\frac{1}{3}r_2}. \tag{5}$$

Hence in Definition 4 we have $r_{\text{eff}} = \frac{2}{3}r_1 + \frac{1}{3}r_2$ 8-ary symbols. Substituting in r_{eff} the BCH redundancies from (2)

$$r_1 = \frac{3}{4}(\tau - 1)\log_4 N, \;\; r_2 = \frac{1}{2}(2\tau - 1)\log_2 N,$$

we get (4) for the EC-rate. ■

Comparing to Construction 1, we see that Construction 2 gives EC-rate higher by $\frac{3}{8}\frac{\log_8 N}{N}$. Consequently, if errors are predominantly mag-1 errors, with a more clever concatenation we can correct the same number of errors with less redundancy compared to the straightforward concatenation. Another important advantage of Construction 2 is in its much simpler decoding. While Construction 1 requires a BCH decoder for τ errors over the finite field GF(8), Construction 2 only needs to correct τ binary errors in C_2, and in the worst case only $\tau/2$ errors over GF(4) in C_4. Moreover, in typical decoding instances, the code C_4 will mostly need to deal with erasures, because in most error patterns very few copies of $\mathcal{W}$ suffer errors in both symbols.

B. Improved EC-WOM construction for mag-1 errors

While Construction 2 offered an improvement of the EC-rate over the basic concatenation, the improvement is quite small, and in particular the difference $\frac{3}{8}\frac{\log_8 N}{N}$ does not grow with τ. In this sub-section we aim at improving the EC-rate further, and more significantly. To get the desired improvement in EC-rate we need two ingredients: 1) a new suitably designed $(2, 8, 4, 8)$ WOM code, and 2) a slight refinement of the error model to distinguish between mag-1 errors not in the same $\mathcal{W}$ pair (more common) and those falling in the same pair (less common). Also, from now on we will rely on the fact that the decoder knows the current write count, that is, how many times (out of the t) the WOM has been written so far.

1) 3-Manhattan WOM: In order to improve the EC-rate, we now show a more suitably designed $(n = 2, q = 8, t = 4, M = 8)$ WOM code. The decoding function of the WOM code is shown in Figure 3a (with the mapping explained in the next sub-subsection). The physical states are colored according to which write(s) reach them:

magenta, green, red, blue for writes 1-4, respectively. The physical states with mixed colors are shared between two adjacent writes; for example physical state $(c_1, c_2) = (0, 7)$ is shared between writes 3 and 4. With the colors of the states provided, the update function can be specified in a straightforward manner. This property of restricting the physical state to belong to the corresponding write's color will be useful for later proving the error-correction properties of the construction.

Definition 8. *Given two physical states α, β of a (n, q, t, M) WOM code, the* ***Manhattan distance*** *between the physical states is defined as $d_m(\alpha, \beta) = \sum_{i=1}^{n} |\alpha_i - \beta_i|$, where α_i, β_i are the value written to the i-th cell of α, β respectively.*

2) Mapping 8-ary to a product of 4-ary and binary: We use an 8-ary to a product of 4-ary and binary mapping, like in Construction 2 (Section IV-A.2) to map the 8-ary alphabet to a product of 4-ary and binary alphabets. This mapping gives the WOM the following property: the Manhattan distance between two physical states *in the same write* that have the same 4-ary symbol and opposite binary bit is at least 3. This property can be seen in Figure 3a showing the WOM code with the product alphabet. For example, in write 2 physical state $(3, 1)$ with value $b0$ and physical state $(5, 0)$ with value $b1$ are at Manhatten distance 3, and no $b1$ state is closer to $(3, 1)$ in write 2 (there is a closer $b1$ state in $(1, 1)$, but it does not belong to write 2.)

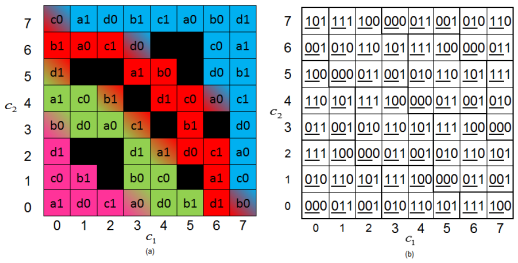

Figure 3. WOM codes and mappings. (a) Construction 3 and (b) Construction 4.

3) Outer 4-ary and binary EC codes: For the 4-ary symbols we take the $C_4[N, N-r]$ BCH code with design distance $2\tau+1$; the binary bits are left uncoded (i.e., we use the trivial code $C_2[N, N]$). These choices will allow us to correct up to τ errors in the 4-ary symbols and recover the binary bit.

4) The construction:

Construction 3. *Given the $(2, 8, 4, 8)$ WOM code $\mathcal{W}_2$ specified with its logical mapping in Figure 3a and EC codes $C_4[N, N-r]$, $C_2[N, N]$ specified in Section IV-B.3, the concatenated code $\mathcal{CW}_2$ is the $\left(2N, 8, 4, 4^{N-r} \cdot 2^N\right)$ WOM code obtained by taking N copies of $\mathcal{W}_2$ and inputing to each a symbol of C_4 to its upper symbol and a symbol of C_2 to its lower symbol.*

The concatenated EC-WOM code specified in Construction 3 has the following properties.

Proposition 2. *For any N and τ, a $\left(2N, 8, 4, 4^{N-r} \cdot 2^N\right)$ code $\mathcal{CW}_2$ obtained by Construction 3 is τ mag-1 error correcting, assuming no copy of $\mathcal{W}_2$ suffers two mag-1 errors. Moreover, $\mathcal{CW}_2$ has EC-rate*

$$1 - \frac{3}{4} \cdot \frac{(2\tau - 1) \cdot \log_8 N}{N}. \tag{6}$$

Proof: Suppose τ copies of $\mathcal{W}_2$ suffered a mag-1 error in exactly one of their two cells. Assuming the decoder knows the current write number, a mag-1 error resulting in a physical state outside this write's color can be detected by the decoder, and mapped to a 4-ary *erasure* in C_4. Other mag-1 errors map to 4-ary errors in C_4, because adjacent physical states within the same write's color have different 4-ary symbols (check in Figure 3a). Thus all τ 4-ary symbols in the erroneous cells can be recovered since C_4 is designed with distance $2\tau+1$. To recover the lower bits of the cells with mag-1 errors we use the Manhattan-distance-3 property shown in Section IV-B.2 that guarantees that not both $x0$ and $x1$ states are adjacent to the read cell state, where x is the 4-ary symbol recovered by C_4. The EC-rate follows a similar calculation to Proposition 1 with $r_{\text{eff}} = \frac{2}{3}r$ and $r = \frac{3}{4}(2\tau - 1)\log_4 N$. ■

Examining the EC-rate attained for the improved Construction 3, we see a significant improvement over both Construction 1 and Construction 2. The new EC-rate is higher than Construction 1 by $\frac{1}{8}\frac{(2\tau-1)\log_8 N}{N}$, now an improvement that *grows with* τ. This advantage comes at the cost of excluding error patterns with both cells in error at the same copy of $\mathcal{W}_2$ (most errors of this type are still correctable, but not all; see Figure 3a $c1$ in state $(3, 3)$ changing to $d1$ in state $(4, 4)$: the decoder will miscorrect to $c0$ in state $(5, 4)$). While the uncorrectable error combinations are unlikely in a random error pattern, we next refine Construction 3 to also guarantee a specified number of double mag-1 errors in the same $\mathcal{W}_2$ copy.

3') Refined outer 4-ary and binary EC codes: For correcting τ_1 single mag-1 errors (one in a $\mathcal{W}_2$ copy) and τ_2 double mag-1 errors (both in the $\mathcal{W}_2$ copy), use for the 4-ary upper symbols the code $C_4[N, N - r_1]$ with design distance $2(\tau_1 + \tau_2) + 1$. For the lower binary symbols use the code $C_2[N, N - r_2]$ designed with distance $2\tau_2 + 1$.

The refined Construction 3 has the following correction capabilities.

Proposition 3. *For any N and τ_1, τ_2, a $\left(2N, 8, 4, 4^{N-r_1} \cdot 2^{N-r_2}\right)$ code $\mathcal{CW}'_2$ obtained by the refined Construction 3 corrects any error combination where at most τ_2 $\mathcal{W}_2$ copies suffered double mag-1 errors and at most $\tau_1 + \tau_2$ $\mathcal{W}_2$ copies suffered mag-1 errors (double or single). Moreover, $\mathcal{CW}'_2$ has EC-rate*

$$1 - \frac{3}{4} \cdot \frac{\left(2\tau_1 + \frac{10}{3}\tau_2 - \frac{5}{3}\right) \cdot \log_8 N}{N}. \tag{7}$$

Proof: As in Proposition 2, C_4 will recover all the 4-ary symbols in $\mathcal{W}_2$ copies that suffered mag-1 errors, single or double. After setting the bits in the binary symbols according to the closest physical states to the read outputs, the decoder of C_2 will be able to correct the at most τ_2 bit errors from double-error $\mathcal{W}_2$ copies. The EC-rate follows a similar calculation to Proposition 1 with $r_{\text{eff}} = \frac{2}{3}r_1 + \frac{1}{3}r_2$, $r_1 = \frac{3}{4}(2\tau_1 + 2\tau_2 - 1)\log_4 N$, and $r_2 = \frac{1}{2}(2\tau_2 - 1)\log_2 N$. ■

It can be shown that the refined Construction 3 can correct τ_2 arbitrary errors in $\mathcal{W}_2$ copies, not just double mag-1 errors.

In an effort to further increase the EC-rate and reduce complexity, we next construct EC-WOM codes that correct the weaker but similarly motivated Amag-1 errors.

C. EC-WOM construction for Amag-1 errors

In the following construction we endow a WOM code with guaranteed error correction while *exclusively using binary*

codes in the concatenation. This is a major implementation advantage in practice, as binary codes (in particular BCH codes) are much easier to implement.

1) Tiling WOM code: The inner WOM code we use in this construction is the known tiling-based $(n = 2, q = 8, t = 4, M = 8)$ code from [4], same as in Section IV-A. We use a different mapping as specified next.

2) Mapping 8-ary to three bits: We map the 8-ary symbols to a 3-bit binary representation as specified in Figure 3b showing the decoding function. We order the bits' significance from right to left (the LSB is the right bit), and note the following properties of this mapping. 1) Between two physical states adjacent horizontally or vertically, *exactly* one of the two higher bits is flipped, and between two physical states adjacent on the main diagonal (bottom-left to top-right), both high bits are flipped. 2) The LSB flips along the secondary diagonal (top-left to bottom-right). These properties will be used later on for decoding.

3) Outer binary EC codes: We use the code $C_2\,[2N, 2N - r_1]$ with design distance $2\tau + 1$ for the two upper bits, and for the lower bit we use the code $C_2\,[N, N - r_2]$ designed with distance $\tau + 1$.

4) The construction:

Construction 4. *Given the* $(2, 8, 4, 8)$ *WOM code* $\mathcal{W}$ *specified with the logical mapping in Figure* 3*b and EC codes specified in Section* IV-C.3*, the concatenated code* $\mathcal{CW}_3$ *is the* $\left(2N, 8, 4, 2^{2N-r_1} \cdot 2^{N-r_2}\right)$ *WOM code obtained by taking* N *copies of* $\mathcal{W}$ *and inputing to each two bits of* $C_2\,[2N, 2N - r_1]$ *as the higher bits and a bit of* $C_2\,[N, N - r_2]$ *as the lower bit.*

The properties of Construction 4 result in the following proposition.

Proposition 4. *For any* N *and* τ*, a* $\left(2N, 8, 4, 2^{3N-r_1-r_2}\right)$ *code* $\mathcal{CW}_3$ *obtained by Construction* 4 *is* τ *Amag-1 error correcting. Moreover,* $\mathcal{CW}_3$ *has EC-rate*

$$1 - \frac{3}{4} \cdot \frac{\left(2\tau - \frac{4}{3}\right)\log_8 N + \frac{4}{9}\tau - \frac{2}{9}}{N}. \tag{8}$$

Proof: Let τ_1 denote the number of copies of $\mathcal{W}$ that suffered an Amag-1 error in exactly one of their two cells, and τ_2 denote the number of copies of $\mathcal{W}$ that suffered Amag-1 errors in both their cells. We show that any combination with $\tau_1 + 2\tau_2 \leqslant \tau$ is correctable. We first decode the binary code $C_2\,[2N, 2N - r_1]$ of the two higher bits. From the mapping seen in Figure 3b, each Amag-1 error results in an error in the two upper bits, thus the $2N$ upper bits suffer at most $\tau_1 + 2\tau_2$ errors. Since $\tau_1 + 2\tau_2 \leqslant \tau$, the decoder of $C_2\,[2N, 2N - r_1]$ can correct all these errors. Next we decode $C_2\,[N, N - r_2]$ of the lower bit. For each copy of $\mathcal{W}$ that suffered a single Amag-1 error we input an erasure to the lower bit's decoder. All other lower bits are correct. Since the decoder sees at most $\tau_1 \leqslant \tau$ erasures, it can recover the lower bits successfully. The EC-rate follows a similar calculation to Proposition 1 with $r_{\text{eff}} = \frac{1}{3}(r_1 + r_2)$, $r_1 = \frac{1}{2}(2\tau - 1)\log_2 2N$, and $r_2 = \frac{1}{2}(\tau - 1)\log_2 N$. ■

As N grows, r_{eff} of Construction 4 tends to $\left(\frac{3}{2}\tau - 1\right)\log_8 N$ (see (8)), which is the lowest redundancy among all the constructions given so far in the paper. The closest redundancy to this is achieved by Construction 3, but without guarantee to correct double errors in the same WOM copy. Another benefit of using Construction 4 in the presence of Amag-1 errors is that both codes are binary codes, which implies simpler implementation with BCH codes, and also greater flexibility to use alternative lower complexity binary codes like LDPC codes. A summary and comparison of the constructions in this section is given in Figure 4.

Construction	Error type	EC-rate	EC-alphabets	Gap from construction 1
1	any	$1-\frac{7}{8}\cdot\frac{(2\tau-1)\log_8 N}{N}$	$GF(8)$	0
2	mag-1	$1-\frac{\left(\frac{3}{4}(\tau-1)+\frac{1}{2}(2\tau-1)\right)\log_8 N}{N}$	$GF(4), GF(2)$	$\frac{3}{8}\cdot\frac{\log_8 N}{N}$
3	single mag-1	$1-\frac{3}{4}\cdot\frac{(2\tau-1)\log_8 N}{N}$	$GF(4)$	$\frac{1}{8}\cdot\frac{(2\tau-1)\log_8 N}{N}$
3'	single mag-1, any	$1-\frac{3}{4}\cdot\frac{\left(2\tau_1+\frac{10}{3}\tau_2-\frac{5}{3}\right)\log_8 N}{N}$	$GF(4), GF(2)$	$\frac{1}{8}\cdot\frac{(14\tau-12\tau_1-20\tau_2+3)\log_8 N}{N}$
4	amag-1	$1-\frac{3}{4}\cdot\frac{\left(2\tau-\frac{4}{3}\right)\log_8 N+\frac{4}{9}\tau-\frac{2}{9}}{N}$	$GF(2), GF(2)$	$\frac{1}{24}\cdot\frac{(6\tau+3)\log_8 N-8\tau+4}{N}$

Figure 4. Comparison of EC-WOM constructions.

V. Conclusion

This paper contributes several WOM codes with guaranteed error correction of τ errors, for arbitrary τ. The advantage of concatenation with standard EC-codes is that any code can be used (not just BCH), including LDPC or Polar codes over different alphabets. Interesting future work is to analyze EC-WOM constructions in the case of random errors following common error models in Flash devices.

VI. Acknowledgement

This work was supported in part by the Israel Science Foundation (ISF), and by the US-Israel Binational Science Foundation (BSF).

References

[1] Bhatia, A., Qin, M., Iyengar, A., Kurkoski, B. and Siegel, P., 2014. "Lattice-based WOM codes for multilevel flash memories". IEEE Journal on Selected Areas in Communications, 32(5), pp.933-945.

[2] Bose, R. and Ray-Chaudhuri, D., 1960. "On a class of error correcting binary group codes". Information and Control, 3(1), pp.68-79

[3] Bossert, M., 1999. "Channel Coding for Telecommunications". John Wiley & Sons, Inc..

[4] Cassuto, Y. and Yaakobi, E., 2014. "Short Q-ary fixed-rate WOM codes for guaranteed rewrites and with hot/cold write differentiation". IEEE Transactions on Information Theory, 60(7), pp.3942-3958.

[5] Heegard, C., 1985. "On the capacity of permanent memory". IEEE Transactions on Information Theory, 31(1), pp.34-42.

[6] Hemo, E. and Cassuto, Y., 2016. "d-imbalance WOM codes for reduced inter-cell interference in multi-level NVMs". IEEE Journal on Selected Areas in Communications, 34(9), pp.2378-2390.

[7] Hocquenghem, A., 1959. "Codes correcteurs derreurs". Chiffres, 2(2), pp.147-56.

[8] Jiang, A., Li, Y., Gad, E., Langberg, M. and Bruck, J., 2013. "Joint rewriting and error correction in write-once memories". In Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on (pp. 1067-1071). IEEE.

[9] Kurkoski, B., 2010. "A note on using lattices for error-correction and rewriting in flash memories". In Proceedings of the 33rd Symposium on Information Theory and its Applications, (Nagano, Japan), pp. 99-102, SITA.

[10] Odeh, S. and Cassuto, Y., 2014, June. "NAND flash architectures reducing write amplification through multi-write codes". In Mass Storage Systems and Technologies (MSST), 2014 30th symposium on (pp. 1-10). IEEE.

[11] Rivest, R. and Shamir, A., 1982. "How to reuse a write-once memory". Information and Control, 55(1-3), pp.1-19.

[12] Yaakobi, E., Siegel, P., Vardy, A. and Wolf, J., 2012. "Multiple error-correcting WOM-codes". IEEE Transactions on Information Theory, 58(4), pp.2220-2230.

[13] Yadgar, G., Yaakobi, E. and Schuster, A., 2015. "Write once, get 50% free: saving SSD erase costs using WOM codes". In FAST (pp. 257-271).

[14] Yekhanin, S. and Dumer, I., 2004. "Long nonbinary codes exceeding the Gilbert-Varshamov bound for any fixed distance". IEEE Transactions on Information Theory, 50(10), pp.2357-2362.

[15] Zemor, G. and Cohen, G., 1991. "Error-correcting WOM-codes". IEEE Transactions on Information Theory, 37(3), pp.730-734.