Error-Correcting WOM Codes for Worst-Case and Random Errors

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 guarantee correction of 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. In addition to constructions for guaranteed error correction, we develop an error-correcting WOM scheme for random errors using the concept of multi-level coding.**

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, yet the Flash technology suffers from the major impediment of not being able to update data in-place. As 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. Thus, write performance is degraded and the wear of cells is accelerated. It has been demonstrated and recognized [3], [7] that *WOM (write-once memory) codes* [4] 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.

In this work we aim to improve the WOM codes' reliability by making them resilient to errors from noise and interference. Our method is to combine WOM codes with error-correcting (EC) codes to get error correction capabilities 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.

We first give the general definition of a q-ary WOM code with *fixed-rate*, that is, the same number of bits is written in each of its writes.

Definition 1. *A (ℓ, q, t, B) (fixed-rate)* ***WOM code*** *is a code applied to a size ℓ block of q-ary cells, and guaranteeing t writes of input size B each, while the cell levels do not decrease.*

II. EC-WOM Concatenated Constructions

The route to error-correcting WOM codes we pursue in this work is through *concatenation*. We call such codes *EC-WOM* codes. Let $\mathcal{W}$ be a (n, q, t, M) WOM code. Denote an EC code $\mathcal{C}$ by $\mathcal{C}_A[N, N-r]$ if it is defined over an alphabet of size A (power of a prime), it has length N, and r redundant A-ary symbols. A pictorial illustration of concatenating the

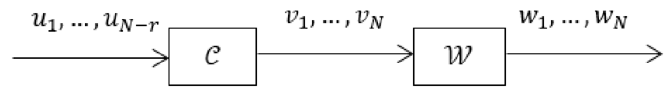

Figure 1: EC-WOM concatenation. $u_i, v_j \in GF(A)$, whereas $w_i \in \{0, ..., q-1\}^n$.

EC code $\mathcal{C}$ with the WOM code $\mathcal{W}$ is given in Figure 1. If we want to compare different EC-WOM codes with the same parameters of the inner WOM code (n, q, t, M), we can use the following measure we call EC-rate.

Definition 2. *Let $\mathcal{CW}$ be a $(nN, q, t, M^{N-r_{\text{eff}}})$ 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 $\mathcal{CW}$ to be $1 - \frac{r_{\text{eff}}}{N}$. r_{eff} is the effective redundancy of the EC-code in units of M-ary symbols.*

The EC-rate does not take into account the redundancy of the inner WOM code, and it only measures the storage efficiency of adding error-correction properties. In a baseline EC-WOM concatenation [2] the inner (n, q, t, M) WOM code is used with an outer M-ary τ error correcting EC-code $\mathcal{C}_M[N, N-r]$, and the resulting EC-rate is $1 - r/N$. In the full paper [5] we provide four new constructions of EC-WOM codes that offer higher EC-rates than the baseline option (for the same τ), for error types that are common in flash memories: magnitude-1 (mag-1) or asymmetric magnitude-1 (Amag-1). Table I shows a comparison of the EC-rates of these constructions for WOM parameters $(n = 2, q = 8, t = 4, M = 8)$, using BCH codes [1] over different alphabets. The first row shows the baseline and the other four rows show the new constructions. "single mag-1" in the first column means that the code corrects error patterns where none of the inner WOM codewords (of length $n = 2$) suffer mag-1 errors in both cells; "single mag-1, any" allows up to τ_2 inner WOM codewords to have mag-1 errors in both cells. In addition to the positive gaps in EC-rate, the new codes enjoy complexity savings thanks to their use of simpler BCH decoders (some of which are erasure decoders), over smaller alphabets.

TABLE I: Comparison of EC-WOM constructions

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

III. EC-WOM Codes for Random Errors

In addition to EC-WOM codes for guaranteed worst-case errors, we propose EC-WOM codes for random errors over realistic memory channel models, e.g., AWGN channels. To keep implementation complexity low, we only use binary EC-codes throughout the section, and use multiple codes through the concept of *multi-level coding* [6]. To not confuse the levels in the multi-level coding hierarchy with cell levels, we call the former "bit-levels" instead of just "levels". For our multi-level coding scheme it is sufficient to have the normal hard-decision memory readout (of discrete cell levels $\{0,\dots,q-1\}$), while still allowing soft decoding by extracting reliability information from the decoder of the inner WOM code.

Multi-level coding with multi-stage decoder: In multi-level coding, one maps each high-order symbol (an M-ary WOM logical symbol in our case) to multiple bits, each coded by a separate binary EC-code. Each individual EC-code is designed for the channel induced on its bit-level by the high-order channel. Multi-level coding orders the bit-levels in a hierarchy, and applies *multi-stage decoding*, whereby the decoder outputs of a bit-level are supplied to the decoders of the bit-levels above it in the hierarchy. The inputs to the multi-stage decoder are m bit-levels ordered $1,\dots,m$, each having N inputs from the alphabet $\{0,1,\perp\}$. The erasure symbol $\perp$ is used for indices where the inner WOM codeword has an invalid state, or when a lower-level decoder corrected an error in this index. Each bit-level code with some minimum distance d_i is decoded by an error+erasure (hard-decision) bounded-distance decoder. To support this scheme, we use a refined definition for the WOM decoding function:

Definition 3. *The WOM decoding function is defined as* $\psi : \{0,\dots,q-1\}^n \times \{1,\dots,t\} \to \{0,\dots,M-1\} \cup \perp$, *where the second argument* $k \in \{1,\dots,t\}$ *is the write number, and the output* $\perp$ *represents decoding failure. In addition, let* $m = \log_2 M$ *and a* M*-ary to binary mapping be given. Define the WOM* binary *decoding function as* $\psi_b : \{0,\dots,q-1\}^n \times \{1,\dots,t\} \to \{0,1\}^m \cup \perp^m$.

The decoding function outputs $\perp$ when the WOM codeword corresponds to an invalid state or a state that cannot be reached in write number k.

WOM constructions for multi-level coding: We proposed three additional WOM codes (and matching M-ary to binary mappings) designed to minimize the propagation of errors from the q-ary memory alphabet to the binary alphabet of the bit-level codes. In particular, to maximize the error detection capabilities of the WOM decoder (use of $\perp$ output), we sought to introduce many invalid states in the decoding functions, as well as to minimize the number of states with overlapping write numbers. The decoding function of one of these constructions (for $(n=2, q=8, t=4, M=8)$) is given in Figure 2: the (c_1, c_2) positions of the matrix represent the cell levels at the WOM codeword's coordinates, the content of each position is the binary encoding of the M-ary symbol, and its color is the write number(s) in which it can be reached (an empty position is an invalid state).

Decoding error probability evaluation: We evaluate the multi-level coding WOM codes based on the decoding error probability of the (multi-level) concatenated code that uses them. In multi-level coding, a decoding-error event happens when at least one of the bit-level decoders either returns the wrong codeword or declares failure. Based on this we derived an analytic upper bound on the decoding error probability by analyzing the probabilities p_i,q_i, which denote the probability that a bit input to the i-th bit-level decoder is an error, erasure, respectively. These analytic tools were then used to optimize the redundancy allocation for the m bit-levels such that the error probability bound is minimized for the total redundancy budget. Sample results from this process are given in Figure 3 comparing the error-probability bound of four constructions (for the $k=3$-rd write), in addition to an EC-only code (top curve) that uses multi-level coding with no WOM code. It can be observed that in this write the code of Figure 2 (named MLCode3) is superior, and that all WOM constructions significantly outperform the no-WOM option. The explanation of the last fact is that the redundancy invested in the WOM codes toward the re-writing features is also (significantly) helpful in improving the error tolerance of the EC-WOM code.

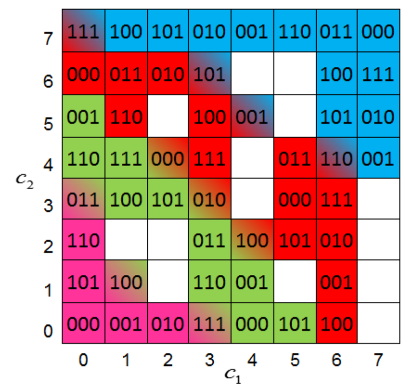

Figure 2: A WOM code for EC-WOM with multi-level coding.

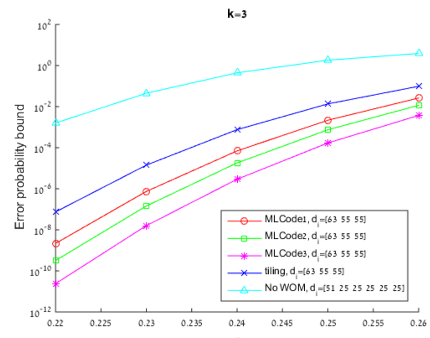

Figure 3: Error probability bound, write $k=3$.

REFERENCES

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

[2] Q. Huang, S. Lin, K. Abdel-Ghaffar, "Error-correcting codes for flash coding". IEEE Trans. Inf. Theory, 57(9), pp. 6097-6108, 2011.

[3] S. Odeh, Y. Cassuto, "NAND flash architectures reducing write amplification through multi-write codes". In IEEE conference on Mass Storage Systems and Technologies, MSST 2014.

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

[5] A. Solomon and Y. Cassuto, "Error-correcting WOM codes: concatenation and joint design". IEEE Trans. Inf. Theory, 65(9), pp. 5529-5546, 2019.

[6] U. Wachsmann, R.F. Fischer, J.B. Huber, "Multilevel codes: Theoretical concepts and practical design rules". IEEE Trans. Inf. Theory, 45(5), pp. 1361-1391, 1999.

[7] G. Yadgar, E. Yaakobi, A. Schuster, "Write once, get 50% free: saving SSD erase costs using WOM codes". In USENIX conference on File and Storage Technologies, FAST 2015.