Trading Partially Stuck Cells with Errors

Haider Al Kim[1,2], Sven Puchinger[3], Ludo Tolhuizen[4], Antonia Wachter-Zeh[1]
[1]Institute for Communications Engineering, Technical University of Munich (TUM), Germany
[2]Department Electronic and Communications Engineering, University of Kufa (UoK), Iraq
[3]Hensoldt Sensors GmbH, 89077 Ulm, Germany
[4]Philips Research, High Tech Campus 34, Netherlands
Email: haider.alkim@tum.de, Sven Puchinger <mail@svenpuchinger.de>, ludo.tolhuizen@philips.com, antonia.wachter-zeh@tum.de

***Abstract*—This work considers coding for *partially stuck* memory cells. Such memory cells can only store partial information as some of their levels cannot be used due to, e.g., wearout. First, we present 2^μ-ary partially stuck cell code construction (over the finite field $\mathbb{F}_{2^\mu}$, where integer $\mu > 1$) for *masking* partially stuck cells while correcting substitution errors. "Masking" finds a word whose entries coincide with writable levels at the (partially) stuck cells. Then we investigate a technique where the encoder, after a first masking step, introduces errors at some partially stuck positions of a codeword in order to satisfy the stuck-at constraints. It turns that treating some of the partially stuck cells as erroneous cells can decrease the required redundancy for some parameters, e.g., by Lemma 2.**

***Index Terms*—flash memories, phase change memories, non-volatile memories, defective memory, (partially) stuck cells, error-correcting codes, Gilbert-Varshamov bound**

I. Introduction

The demand for reliable storage solutions, particularly for *non-volatile memory* NVM, such as *flash memory* and *phase change memories* (PCMs) for different applications, is steadily increasing. Non-volatile memory is a memory that stores information even when powered off. These multilevel devices provide permanent storage, a rapidly extendable capacity, faster data access, lower power consumption, and enhanced physical resilience [1]. A primary issue with NVMs technology is that its read channel degrades significantly over time, resulting in *unacceptable reliability*. Capturing and releasing charges in flash memory during programming and erasing causes damage, e.g., *wear-out* in the form of charge *trapping* in the oxide and interface states [2] and [3]. The *trap* (the *stuck-at*; see [4] and [5]) prohibits a cell from switching its level, although new charges are injected or removed from this cell.

On the other hand, the key characteristic of PCM cells is that they can switch between an *amorphous* state and a *crystalline* state. PCM cells may become *unreliable* (also called *defective* or *stuck* [6] and [7]) if they fail to switch their states. This occasionally happens due to the cooling and heating processes of the cells. Thus, they can only hold a single phase. In multi-level PCM cells, failure may occur at a position in either extreme states or in the *partially programmable states of crystalline*.

This deterioration can be handled at the system-level *channel codes*. Employing error correction schemes for unreliable memories dates back to the 1970s [8]. For NVMs devices, the case when cells are partially stuck at level 1 is particularly important. For instance, traps at level 1 mean no demand for mandatory erases that result in faster degradation in flash memories. For multi-level PCMs, it means that a cell can reach all crystalline sub-states but cannot reach the amorphous state. The suggested mechanism to deal with these defective memory cells is called *masking*. The "masking" process determines suitable codewords that regard the writable levels in the partially stuck positions. Due to the linearity, more than one codeword can represent the same message to recover from (partially) stuck-at errors. That is, for any position indexed as partially stuck-at, multiple possible values (equal or higher than) the corresponding stored values can be applied. Despite successfully masking the partially stuck positions [4], the storing process might fail to present a word that can be appropriately written to that memory due to substitution errors (i.e., caused by inter-cell interference noise, or other noise disturbance [1]), or the reading process might be unsuccessful [9].

II. Our Contribution

In this paper, we combine the method of the reduced redundancy necessary for masking from [4] with the capability to correct substitution errors from [5]. Compared to the conventional stuck-cell case in [5], we reduce the redundancy necessary for masking, similar to the results in [4], and even reduce further compared to [4, Construction 5]. We also show that treating some partially stuck cells as erroneous cells can decrease the required redundancy for some parameters. Because of space limitations we skip all proofs and redirect the interested reader to see them in [11]. Our focus is on long codes over small alphabets, i.e., the code length n is larger than the field size q. Otherwise, one could instead mask by a code of length $n < q$ (by using, e.g., [10]).

III. Preliminaries

A. Notations

For a prime power q, let $\mathbb{F}_q$ denote the finite field of order q and $\mathbb{F}_q[x]$ be the set of all univariate polynomials with coefficients in $\mathbb{F}_q$. For $g, f \in \mathbb{Z}_{>0}$, denote $[f] = \{0, 1, \dots, f-1\}$ and $[g, f] = \{g, g+1, \dots, f-1\}$. As usual, an $[n, k, d]_q$ code is a linear code over $\mathbb{F}_q$ of length n, dimension k and minimum (Hamming) distance d. The (Hamming) weight $\mathrm{wt}(\boldsymbol{x})$ of a vector $\boldsymbol{x} \in \mathbb{F}_q^n$ equals its number of non-zero entries. We fix throughout the paper a total ordering "$\geq$" of the elements of $\mathbb{F}_q$ such that $a \geq 1 \geq 0$ for all $a \in \mathbb{F}_q \setminus \{0\}$. So 0 is the smallest element in $\mathbb{F}_q$, and 1 is the next smallest element in $\mathbb{F}_q$. We extend the ordering on $\mathbb{F}_q$ to $\mathbb{F}_q^n$: for $\boldsymbol{x} = (x_0, \dots, x_{n-1}) \in \mathbb{F}_q^n$ and $\boldsymbol{y} = (y_0, \dots, y_{n-1}) \in \mathbb{F}_q^n$, we say that $\boldsymbol{x} \geq \boldsymbol{y}$ if and only if $x_i \geq y_i$ for all $i \in [n]$.

B. Definitions

1) Defective and Partially Defective Cells: A cell is called defective (*stuck-at level s*), if it can only store the value s. A cell is called partially defective (*partially-stuck-at level s*), if it can only store values which are at least s. Note that a cell that is partially defective at level 0 is a non-defective cell which can store any of the q levels and a cell that is partially defective at level $q-1$ is a (fully) defective cell.

2) $(n, M)_q(\Sigma, t)$ PSMC: Let $\Sigma \subset \mathbb{F}_q^n$ define the states of a memory. For non-negative integer t, a q-ary (Σ, t)-*partially-stuck-at-masking code* $\mathcal{C}$ of length n and size M is a linear coding scheme consisting of a message set $\mathcal{M}$ of size M, an encoder $\mathcal{E}$ and a decoder $\mathcal{D}$ satisfying:

1) The encoder $\mathcal{E}$ is a mapping from $\mathcal{M} \times \Sigma$ to $\mathbb{F}_q^n$ such that for each $(\boldsymbol{m}, \boldsymbol{s}) \in \mathcal{M} \times \Sigma, \quad \mathcal{E}(\boldsymbol{m}, \boldsymbol{s}) \geq \boldsymbol{s}$,
2) For each $(\boldsymbol{m}, \boldsymbol{s}) \in \mathcal{M} \times \Sigma$ and each $\boldsymbol{e} \in \mathbb{F}_q^n$ such that $\mathrm{wt}(\boldsymbol{e}) \leq t$ and $\mathcal{E}(\boldsymbol{m}, \boldsymbol{s}) + \boldsymbol{e} \geq \boldsymbol{s}$, it holds that $\mathcal{D}(\mathcal{E}(\boldsymbol{m}, \boldsymbol{s}) + \boldsymbol{e}) = \boldsymbol{m}$.

If $\Sigma = \{\mathbf{s} \in \{0,1\}^n \mid wt(\mathbf{s}) \leq u\}$, we say q-ary $(u, 1, t)$ PSMC. In this special case, the partially stuck-at condition means that the output of the encoder is non-zero at each position of the support $\boldsymbol{\phi}$ of $\boldsymbol{s}$.

IV. Codes for (Partially) Defective Memories

Construction 1. *Let $\mu > 1$. Suppose $\boldsymbol{G}$ is a $k \times n$ generator matrix of an $[n, k, d]_{2^\mu}$ code $\mathcal{C}$ of the form*

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{G}_1 \\ \boldsymbol{x} \end{bmatrix} \tag{1}$$

where

1) *$\boldsymbol{H}_0 \in \mathbb{F}_2^{l \times n}$ is a parity check matrix of an $[n, n-l, d_0]_2$ code $\mathcal{C}_0$,*
2) *$\boldsymbol{G}_1 \in \mathbb{F}_{2^\mu}^{(k-l-1) \times n}$,*
3) *$\boldsymbol{x} \in \mathbb{F}_{2^\mu}^{1 \times n}$ has Hamming weight n.*

From the code $\mathcal{C}$, a PSMC can be obtained, whose encoder and decoder are shown in [11, Algorithm 5] and [11, Algorithm 6].

Theorem 1. *The coding scheme in Construction 1 is a 2^μ-ary $(2^{\mu-1}d_0 - 1, 1, \lfloor \frac{d-1}{2} \rfloor)$ PSMC of length n and cardinality $2^{\mu(k-l-1)}2^{l(\mu-1)}$.*

V. Partial Masking of Partially Stuck Memory Cells

Proposition 1. *If there is an $(n, M)_q(u, 1, t)$ PSMC, then for any j with $0 \leq j \leq t$, there is an $(n, M)_q(u + j, 1, t - j)$ PSMC.*

We generalize the above proposition to general Σ (Theorem 2).

Theorem 2. *Let $\Sigma \subset \mathbb{F}_q^n$, and assume that there exists an $(n, M)_q(\Sigma, t)$ PSMC $\mathcal{C}$. For any $j \in [t]$, there exists an $(n, M)_q(\Sigma^{(j)}, t - j)$ PSMC $\mathcal{C}_j$, where*

$$\Sigma^{(j)} = \left\{ \boldsymbol{s'} \in \mathbb{F}_q^n \mid \exists \boldsymbol{s} \in \Sigma \left[d(\boldsymbol{s}, \boldsymbol{s'}) \leq j \text{ and } \boldsymbol{s'} \geq \boldsymbol{s} \right] \right\}.$$

Let the encoder $\mathcal{E}_j$ and the decoder $\mathcal{D}_j$ for $\mathcal{C}_j$ be Algorithm 1 and Algorithm 2, respectively.

Algorithm 1: Encoding

Input: $(\boldsymbol{m}, \boldsymbol{s'}) \in \mathcal{M} \times \Sigma^{(j)}$.
Determine $\boldsymbol{s} \in \Sigma$ such that $d(\boldsymbol{s}, \boldsymbol{s'}) \leq j$ and $\boldsymbol{s'} \geq \boldsymbol{s}$.
Let $\boldsymbol{c} = \mathcal{E}(\boldsymbol{m}, \boldsymbol{s})$.
Define $\boldsymbol{c'} = \mathcal{E}'_j(\boldsymbol{m}, \boldsymbol{s'})$ as $c'_i = \max(c_i, s'_i)$ for $i \in [n]$.
Output: Codeword $\boldsymbol{c'}$.

Algorithm 2: Decoding

Input: Received $\boldsymbol{y} = \boldsymbol{c'} + \boldsymbol{e}$ where $\mathrm{wt}(\boldsymbol{e}) \leq t - j$ and $\boldsymbol{y} \geq \boldsymbol{s'}$
Message $\boldsymbol{m} = \mathcal{D}(\boldsymbol{y})$
Output: Message vector $\boldsymbol{m}$

We can improve on Theorem 2 for Construction 1 giving Lemma 1.

Lemma 1. *Given an $[n, k, d]_q$ code as defined in Construction 1, then for any j such that $0 \leq j \leq \lfloor \frac{d-1}{2} \rfloor$, there is a 2^μ-ary $(2^{\mu-1}(d_0 + j) - 1, 1, \lfloor \frac{d-1}{2} \rfloor - j)$ PSMC of length n and size q^{k-l-1}.*

In the following lemma, we use another approach for introducing errors in order to satisfy the stuck-at conditions.

Lemma 2. *Given an $[n, k, d]_q$ code containing a word of weight n, for any j with $0 \leq j \leq \lfloor \frac{d-1}{2} \rfloor$, there is a q-ary $(q - 1 + qj, 1, \lfloor \frac{d-1}{2} \rfloor - j)$ PSMC of length n and size q^{k-1}.*

VI. Comparisons Gilbert–Varshamov-type Bounds

For Construction 1, we have derived Gilbert–Varshamov-type bound in [11, Theorem 11]. We compare the direct application of [11, Theorem 11] with the exchange of a one error correction ability with a single masking capability of a partially stuck cell by Theorem 2 that further compared to Lemma 1 and Lemma 2 as shown in Figure 1.

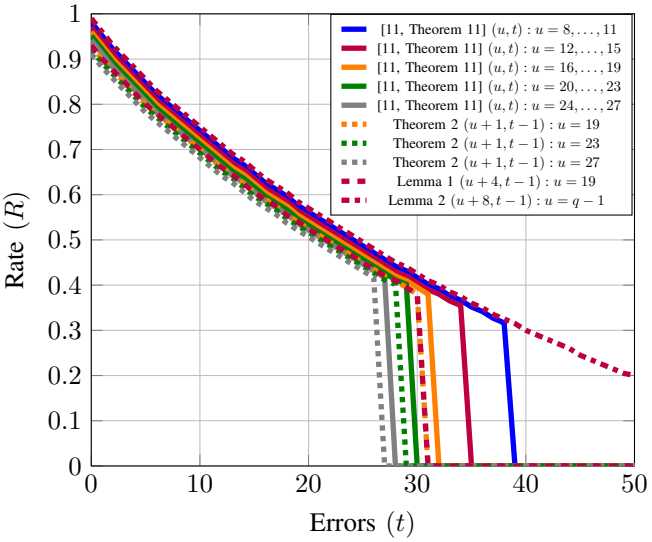

Figure 1. The achievable rates $R = \frac{1}{n} \log_{2^3} \mathcal{M}$ of GV bounds for different u, and t for $n = 200$ and $q = 2^3$ in [11, Theorem 11]. The solid plots are the rates from the derived GV like bound and the dotted lines are the rates after trading $u + 1, t - 1$ by Theorem 2. Trading one correctable error by Lemma 1 and Lemma 2 increases u by $2^{\mu-1}$ and 2^μ, respectively. Lemma 2 gives slightly higher rates for *all* $t \leq 50$ while treating the same number of u cells compared to the corresponding curves from [11, Theorem 11], Theorem 2, and Lemma 1.

References

[1] L. Dolecek and F. Sala. Channel Coding Methods for Non-Volatile Memories. 2016.
[2] P. OLIVO, B. RICCO, and E. SANGIORGI. "High-field-induced voltage-dependent oxide charge". English. In: Applied physics letters (1986). issn: 0003-6951.
[3] C. Monzio Compagnoni, M. Ghidotti, A. L. Lacaita, A. S. Spinelli, and A. Visconti. "Random Telegraph Noise Effect on the Programmed Threshold-Voltage Distribution of Flash Memories". In: IEEE Electron Device Letters 30.9 (2009), pp. 984–986. doi: 10.1109/LED. 2009.2026658.
[4] A. Wachter-Zeh and E. Yaakobi, "Codes for Partially Stuck-at Memory Cells," *IEEE Trans. Inf. Theory*, vol 62, no. 2, pp.639-654, 2016.
[5] C. Heegard, "Partitioned Linear Block Codes for Computer Memory with 'Stuck-at' Defects," *IEEE Trans. Inf. Theory*, vol. 29, no. 6, pp. 831–842, 1983.
[6] B. Gleixner, F. Pellizzer, and R. Bez, "Reliability Characterization of Phase Change Memory," in *2009 10th Annual NVMTS*. IEEE, 2009, pp. 7–11.
[7] A. Pirovano, A. Redaelli, F. Pellizzer, F. Ottogalli, M. Tosi, D. Ielmini, A. L. Lacaita, and R. Bez, "Reliability Study of Phase-Change Nonvolatile Memories," *IEEE Trans Device Mater Reliab*, vol. 4, no. 3, pp. 422–427, 2004.
[8] A. Kuznetsov and B. Tsybakov. "Coding for memories with defective cells," in: (in Russian) Problems Inf. Transmiss. Vol. 10. 2. 1974, pp. 52–60.
[9] A. Solomon and Y. Cassuto. "Error-correcting WOM codes: Concatenation and joint design". In: IEEE Trans. Inf. Theory 65.9 (Sept. 2019), pp. 5529–5546.
[10] G. Solomon. "A Note on Alphabet Codes and Fields of Computation". In: Inf. Control. 25 (1974), pp. 395–398.
[11] H. Al Kim, S. Puchinger, L. Tolhuizen, and A. Wachter-Zeh, "Coding and Bounds for Partially Defective Memory Cells," (submitted to) Design, Code and Cryptography, 2022. Arxiv version is here: https://arxiv.org/pdf/2202.07541.pdf