# Error Scrubbing Codes for Flash Memories

Anxiao (Andrew) Jiang, Hao Li and Yue Wang
Computer Science and Engineering Department
Texas A&M University, College Station, TX 77843, USA
Email: {ajiang,hao,yuewang}@cse.tamu.edu

***Abstract*—Flash memories are the most widely used type of non-volatile electronic memories. Every flash memory cell has $q$ discrete levels for storing information. A prominent property of flash memories is that although it is easy to increase a cell level, to decrease any cell level, a whole block of cells have to be erased and reprogrammed. To minimize the number of expensive block erasure operations and to maintain the data integrity, the data needs to be stored with a strong error-correcting code that can correct enough errors between two erasure operations.**

**In this paper, we introduce the concept of *error scrubbing codes*. With this new type of error-correcting codes, the cell levels are actively increased when errors appear, even if the errors increase cell levels as well. We show that error-scrubbing codes can outperform conventional error-correcting codes for multi-level flash memories. We present two families of codes based on the $L_1$ metric and a modular construction.**

## I. Introduction

Flash memories have become the most widely used type of non-volatile memories (NVMs). Every flash memory cell has $q \geq 2$ levels – level $0, 1, \cdots, q-1$ – which can be increased or decreased by charge injection or removal based on the Fowler-Nordheim tunnelling mechanism or the hot-electron injection mechanism [1]. To increase data density, multi-level cells (MLCs) with greater values of $q$ are actively developed. Flash cells are organized as blocks, each containing about $10^5$ cells. Although it is comparatively simple to increase a cell level, it is very difficult to decrease one. To decrease any cell level, the whole block of cells must be erased and then reprogrammed. Block erasures significantly reduce the longevity, reliability and speed of flash memories [1]. To minimize the number of block erasure operations and to maintain the data integrity, the data needs to be stored with a strong error-correcting code that can correct enough errors between two erasure operations. While errors may accumulate in the codewords, with the next block erasure, the codewords can be decoded and the original error-free codewords can be written back into the block. This is called *memory scrubbing*, a commonly used operation in storage systems [4].

In this paper, we introduce the concept of *error scrubbing codes*. With this new type of error-correcting codes, the cell levels are actively increased when errors appear, even if the errors increase cell levels as well. The implementation is simple: the memory constantly reads the cells of a codeword; if a new error is detected, the memory increases the cell levels to a new state. No block erasure is needed unless the cell levels have reached the top. The key idea of error-scrubbing codes is that through the active adjustment of cell levels, which we call *scrubbing*, we can reduce the number of states that a given number of errors can turn the cells into, thus allowing the packing of more codewords for a higher rate. In this paper, we show that the performance of error-scrubbing codes can exceed that of conventional codes by presenting two families of code constructions based on the $L_1$ metric and a modular technique, respectively, and show their asymptotic optimality.

## II. Notations

Let $c_1, c_2, \cdots, c_n$ denote the levels of $n$ cells of $q$ levels. Here $c_i \in \{0, 1, \cdots, q-1\}$ denotes the $i$-th cell's level, for $i = 1, 2, \cdots, n$. The vector $\vec{c} = (c_1, c_2, \cdots, c_n)$ is called the *cell state*. Let $\mathcal{S}_{n,q}$ denote the set of all $q^n$ cell states. Given two cell states $\vec{c}_A = (c_1, c_2, \cdots, c_n)$ and $\vec{c}_B = (c'_1, c'_2, \cdots, c'_n)$, if $\forall\ i$ we have $c_i \geq c'_i$, we denote it by $\vec{c}_A \geq \vec{c}_B$ and say that $\vec{c}_A$ is above $\vec{c}_B$. If $\vec{c}_A \geq \vec{c}_B$ and there exists some $i$ such that $c_i > c'_i$, we denote it by $\vec{c}_A > \vec{c}_B$ and say that $\vec{c}_A$ is strictly above $\vec{c}_B$. Here we consider codes that correct errors between two block erasures, so the memory controller can only increase cell levels, not decrease them [3]. However, the errors (i.e., noise) may both increase and decrease cell levels, unless they are asymmetric errors.

An *error set* $\varepsilon$ is a set of integral vectors of length $n$. An *error* is a vector in the set $\varepsilon$. For convenience, we always assume that $(0, 0, \cdots, 0) \in \varepsilon$. Give an error $\vec{e} = (e_1, e_2, \cdots, e_n) \in \varepsilon$, it can change a cell state $\vec{c} = (c_1, c_2, \cdots, c_n)$ to $\vec{c} + \vec{e} = (c_1 + e_1, c_2 + e_2, \cdots, c_n + e_n)$. For example, if $\varepsilon$ consists of those errors that can increase some cell level by one, then $\varepsilon = \{(e_1, e_2, \cdots, e_n) \mid \sum_{i=1}^{n} e_i \leq 1,\ e_i = 0 \text{ or } 1 \text{ for all } i\}$.

A *scrubbing function* is a mapping $f : \mathcal{S}_{n,q} \to \mathcal{S}_{n,q}$ such that $\forall\ \vec{c} \in \mathcal{S}_{n,q},\ f(\vec{c}) \geq \vec{c}$. The idea of the error-scrubbing code is that when the cell state is $\vec{c}$, the memory will update it to $f(\vec{c})$ by charge injection.

Let $t \geq 1$ be an integer. Let $\vec{c} \in S_{n,q}$ be a cell state, and let $\vec{e}_1, \vec{e}_2, \cdots, \vec{e}_t \in \varepsilon$ be $t$ errors. When the first error $\vec{e}_1$ changes the cell state from $\vec{c}$ to $\vec{c} + \vec{e}_1$, the memory will update the cell state to $f(\vec{c} + \vec{e}_1)$. When the second error $\vec{e}_2$ changes the cell state to $f(\vec{c} + \vec{e}_1) + \vec{e}_2$, the memory will update the cell state to $f(f(\vec{c} + \vec{e}_1) + \vec{e}_2)$. And so on. In general, for $i = 1, 2, \cdots, t$, define a function $g(\vec{c}; \vec{e}_1, \cdots, \vec{e}_i)$ as follows: $g(\vec{c}; \vec{e}_1) = \vec{c} + \vec{e}_1$; for $i = 2, \cdots, t$, $g(\vec{c}; \vec{e}_1, \cdots, \vec{e}_i) = f(g(\vec{c}; \vec{e}_1, \cdots, \vec{e}_{i-1})) + \vec{e}_i$. Then, when the initial cell state is $\vec{c}$ and $t$ errors $\vec{e}_1, \vec{e}_2, \cdots, \vec{e}_t$ sequentially appear, the memory will sequentially change the cell state to $f(g(\vec{c}; \vec{e}_1)), f(g(\vec{c}; \vec{e}_1, \vec{e}_2)), \cdots, f(g(\vec{c}; \vec{e}_1, \cdots, \vec{e}_t))$. (By default, we assume that $g(\vec{c}; \vec{e}_1)$, $g(\vec{c}; \vec{e}_1, \vec{e}_2)$, $\cdots$,

$g(\vec{c};\vec{e}_1,\cdots,\vec{e}_t)$ all belong to $\mathcal{S}_{n,q}$.) The set of cell states $trace(\vec{c};\vec{e}_1,\cdots,\vec{e}_t) = \{\vec{c}\} \cup \{g(\vec{c};\vec{e}_1,\cdots,\vec{e}_i) \mid i = 1,2,\cdots,t\} \cup \{f(g(\vec{c};\vec{e}_1,\cdots,\vec{e}_i)) \mid i = 1,2,\cdots,t\}$ are called the *trace* caused by the initial cell state $\vec{c}$ and the $t$ errors $\vec{e}_1,\vec{e}_2,\cdots,\vec{e}_t$.

We are now in a position to define error-scrubbing codes.

**Definition 1** *(*ERROR-SCRUBBING CODE*) Let $\mathcal{C} \subseteq \mathcal{S}_{n,q}$ be a subset of cell states. Let $t \geq 1$ be an integer. Every vector in $\mathcal{C}$ is called a codeword. For every codeword $\vec{c} \in \mathcal{C}$, the set of cell states $B_{\vec{c}} = \bigcup_{\vec{e}_1,\cdots,\vec{e}_t \in \varepsilon} trace(\vec{c};\vec{e}_1,\cdots,\vec{e}_t)$ is called the "decoding sphere" of $\vec{c}$. Every vector in $B_{\vec{c}}$ is decoded as the data represented by the codeword $\vec{c}$. Then, $\mathcal{C}$ is called a $t$-error-scrubbing code if $\forall$ $\vec{c}_1$ and $\vec{c}_2$ in $\mathcal{C}$, $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$.*

It is simple to see that when data are stored as codewords of a $t$-error-scrubbing code, any sequence of $t$ or fewer errors can be corrected. For a $t$-error-scrubbing code $\mathcal{C}$, we define its *density* as $\frac{|\mathcal{C}|}{q^n}$. Naturally, the higher the density is, the higher the rate of the code is.

## III. LINEAR ERROR SCRUBBING CODES

In this section, we consider symmetric errors, and assume that the memory scrubs the codeword as soon as a new error appears. That is, the error set $\varepsilon = \{(e_1,e_2,\cdots,e_n) \mid \sum_{i=1}^{n}|e_i| \leq 1,\ e_i \in \mathbb{Z} \text{ for all } i\}$. For simplicity, we also assume that $q \to \infty$. We will discuss the case of finite $q$ later in the section.

We first present a new linear code construction of its general form, for $n \geq 4$. We will show later how to set its parameters to achieve optimal performance. Note that given two vectors $\vec{a} = (a_1,a_2,\cdots,a_n)$ and $\vec{b} = (b_1,b_2,\cdots,b_n)$, $\vec{a}\cdot\vec{b} = \sum_{i=1}^{n} a_i b_i$. Given an integer $i$, $i\cdot\vec{a} = (ia_1,ia_2,\cdots,ia_n)$.

**Construction 2** *(*LINEAR ERROR-SCRUBBING CODE*) Build a $t$-error-scrubbing code $\mathcal{C}$ for $n \geq 4$ cells as follows. Let $t \geq 1$ be an integer. Let $\vec{a} = (a_1,a_2,\cdots,a_n)$ and $\vec{b} = (b_1,b_2,\cdots,b_n)$ be two vectors, where $a_i, b_i$ are positive integers for all $i$. Let*

$$V = \max_{\vec{e}\in\varepsilon} \vec{e}\cdot\vec{b} - \min_{\vec{e}\in\varepsilon} \vec{e}\cdot\vec{b} + 1 = 1 + 2\max_i b_i.$$

*For $i = 0,1,\cdots,\lfloor\frac{\vec{a}\cdot\vec{b}}{V}\rfloor - 1$, let $\mathcal{C}_i = \{(c_1,c_2,\cdots,c_n) \mid \sum_{j=1}^{n} b_j c_j \equiv iV \bmod (t\cdot\vec{a}\cdot\vec{b})\}$. Let $\mathcal{C} = \bigcup_{i=0}^{\lfloor\frac{\vec{a}\cdot\vec{b}}{V}\rfloor-1} \mathcal{C}_i$.*

*The scrubbing function $f : \mathcal{S}_{n,\infty} \to \mathcal{S}_{n,\infty}$ is defined as follows. Let $\vec{s} \in \mathcal{S}_{n,\infty}$ be any cell state. If there exists a codeword $\vec{c} \in \mathcal{C}$, an integer $i \in \{0,1,\cdots,t-1\}$ and an error $\vec{e} \in \varepsilon$ such that $\vec{s} = \vec{c} + i\vec{a} + \vec{e}$ and $\vec{c} + i\vec{a} > \vec{s}$, then $f(\vec{s}) = \vec{c} + i\vec{a}$. If there exists a codeword $\vec{c} \in \mathcal{C}$, an integer $i \in \{0,1,\cdots,t-2\}$ and an error $\vec{e} \in \varepsilon$ such that $\vec{s} = \vec{c} + i\vec{a} + \vec{e}$ and $\vec{s} > \vec{c} + i\vec{a}$, then $f(\vec{s}) = \vec{c} + (i+1)\vec{a}$. If $\vec{s}$ belongs to neither of the above two cases, then $f(\vec{s}) = \vec{s}$.*

**Lemma 3.** *Let $\mathcal{C}$ be the code of Construction 2. Let $\vec{c} \in \mathcal{C}$ be a codeword. Then, the decoding sphere of $\vec{c}$ is $B_{\vec{c}} = \{\vec{c} + i\cdot\vec{a} + \vec{e} \mid i \in \{0,1,\cdots,t-1\};\ \vec{e} \in \varepsilon\}$.*

The following theorem shows that the code $\mathcal{C}$ of Construction 2 is a $t$-error-scrubbing code.

**Theorem 4** *Let $\mathcal{C}$ be the code of Construction 2. Then, for any two codewords $\vec{c}_1$ and $\vec{c}_2$ of $\mathcal{C}$, we have $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$.*

*Proof:* For any cell state $\vec{s} \in \mathcal{S}_{n,\infty}$, let us define its *signature* as $sig(\vec{s}) = (\vec{b}\cdot\vec{s} \bmod t\cdot(\vec{a}\cdot\vec{b}))$. For any codeword $\vec{c} \in \mathcal{C}$, let $G(\vec{c})$ denote the set of signatures of the cell states in the decoding sphere of $\vec{c}$. That is, $G(\vec{c}) = \{sig(\vec{s}) \mid \vec{s} \in B_{\vec{c}}\}$. Let $b_{max} = \max_i b_i$. Construction 2 shows that $\mathcal{C} = \bigcup_{i=0,1,\cdots,\lfloor\frac{\vec{a}\cdot\vec{b}}{V}\rfloor-1} \mathcal{C}_i$, where each $\mathcal{C}_i$ contains a set of codewords. Then, it is simple to verify that if $\vec{c} \in \mathcal{C}_i$, then $G(\vec{c}) \subseteq \{iV + j\cdot\vec{a}\cdot\vec{b} + k \mod t\cdot\vec{a}\cdot\vec{b} \mid j \in \{0,1,\cdots,t-1\};\ k \in \{-b_{max},-b_{max}+1,\cdots,b_{max}\}\}$.

Let $\vec{c}_1$ and $\vec{c}_2$ be two different codewords in $\mathcal{C}$. If $\vec{c}_1 \in \mathcal{C}_i$ and $\vec{c}_2 \in \mathcal{C}_j$ for some $i \neq j$, then it is simple to see that $G(\vec{c}_1) \cap G(\vec{c}_2) = \emptyset$, so $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$. Now suppose that $\vec{c}_1 \in \mathcal{C}_i$ and $\vec{c}_2 \in \mathcal{C}_i$ for some $i$. We will prove $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$ by contradiction. Assume there exists a cell state $\vec{s} \in B_{\vec{c}_1} \cap B_{\vec{c}_2}$. It is simple to verify that for any codeword $\vec{c} \in \mathcal{C}$ and any cell state $\vec{s}_0 \in B_{\vec{c}}$, $sig(\vec{s}_0) - sig(\vec{c}) \bmod t\cdot\vec{a}\cdot\vec{b}$ is a function of $\vec{s}_0 - \vec{c}$, and no two cell states in $B_{\vec{c}}$ have the same signature. Since $sig(\vec{c}_1) = iV = sig(\vec{c}_2)$, we get $\vec{s} - \vec{c}_1 = \vec{s} - \vec{c}_2$. So $\vec{c}_1 = \vec{c}_2$, which is not true. So $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$. ■

We now present a specific code construction.

**Construction 5** *Let $\vec{a} = (1,1,\cdots,1)$ and $\vec{b} = (1,2,\cdots,n)$. Then, use Construction 2 to build a $t$-error-scrubbing code $\mathcal{C}$.*

**Theorem 6** *Let $\mathcal{C}$ be the $t$-error-scrubbing code of Construction 5. Its density is*

$$\frac{2\lfloor\frac{n(n+1)}{2(2n+1)}\rfloor}{tn(n+1)} = \Theta(\frac{1}{tn}),$$

*which is asymptotically optimal.*

*Proof:* Let $V$ and $C_i$ be as defined in Construction 2. Then for the code $\mathcal{C}$ of Construction 5, $V = 2n+1$. A cell state $(c_1,c_2,\cdots,c_n)$ is in $C_i$ if and only if $\sum_{j=1}^{n} jc_j \equiv i(2n+1) \bmod \frac{tn(n+1)}{2}$. Whatever values $c_2,c_3,\cdots,c_n$ take, the above equation produces $c_1 \equiv i(2n+1) - \sum_{j=2}^{n} jc_j \bmod \frac{tn(n+1)}{2}$. So $\lim_{q\to\infty}\frac{|\mathcal{C}_i|}{q^n} = \frac{2}{tn(n+1)}$. So the density of $\mathcal{C}$ is $\lim_{q\to\infty}\frac{|\mathcal{C}|}{q^n} = \lim_{q\to\infty}\frac{|\bigcup_{i=0}^{\lfloor\frac{n(n+1)}{2}/(2n+1)\rfloor-1}\mathcal{C}_i|}{q^n} = \lfloor\frac{n(n+1)}{2(2n+1)}\rfloor\cdot\frac{2}{tn(n+1)} = \Theta(\frac{1}{tn})$.

To prove that the density of $\mathcal{C}$ is asymptotically optimal (up to a constant ratio), it is sufficient to show that for any $t$-error-scrubbing code $\mathcal{C}'$, $|B_{\vec{c}}| = \Omega(tn)$ for any codeword $\vec{c} \in \mathcal{C}'$. For $i = 1,2,\cdots,n$, let $\vec{e}_i \in \varepsilon$ be the vector where the $i$-th element is 1 and all other elements are 0. Let $\vec{s}_0 = \vec{c}$; for $i =$

$1, 2, \cdots, t-1$, let $\vec{s}_i = f(\vec{s}_{i-1} + \vec{e}_1)$. For $i = 0, 1, \cdots, t-1$, let $S_i = \{\vec{s}_i + \vec{e}_j \mid 1 \le j \le n\}$. It is simple to see that $\forall$ $i$, $S_i \subseteq B_{\vec{c}}$; also, $\forall$ $i \ne j$, $S_i \cap S_j = \emptyset$ (because the cell states in $S_i$ and $S_j$ have different values in terms of the summation of cell levels). So $|B_{\vec{c}}| \ge \sum_{i=0}^{t-1} |S_i| = tn = \Omega(tn)$. ■

We can choose different values for the vector $\vec{a} = (a_1, a_2, \cdots, a_n)$ to further increase the code's density. The density shown in the above theorem is upper bounded by $1/t(2n+1)$. The following theorem shows that the code density can reach this value through a different set of parameters. The tradeoff is to increase some cell levels by more than one in a scrubbing operation. We skip the proof of the following theorem due to its similarity to the previous analysis.

**Theorem 7** *Let $W$ be the smallest integer such that $W \ge \frac{n(n+1)}{2}$ and $W$ is a multiple of $2n+1$. There exists an integer vector $\vec{a} = (a_1, a_2, \cdots, a_n)$ such that $\sum_{i=1}^{n} i a_i = W$ and $\forall$ $i$, $a_i = 1$ or $2$. Let $\vec{b} = (1, 2, \cdots, n)$. With the above parameter vectors $\vec{a}$ and $\vec{b}$, we can use Construction 2 to build a $t$-error-scrubbing code $\mathcal{C}$. The density of $\mathcal{C}$ is $\frac{1}{t(2n+1)}$.*

The above codes are for $n \ge 4$. When $n = 1, 2, 3$, we can build $t$-error-scrubbing codes of density $\frac{1}{t+2}$, $\frac{1}{3t+2}$ and $\frac{1}{7t}$, respectively. We summarize them with the following theorem.

**Theorem 8** *When $n = 1, 2, 3$, there exist $t$-error-scrubbing codes of density $\frac{1}{t+2}$, $\frac{1}{3t+2}$ and $\frac{1}{7t}$, respectively.*

*Proof:* The proof is constructive. We skip the analysis for $n = 1$ due to its simplicity. (Its codewords are cell levels that are multiples of $t+2$.) When $n = 2$, let the code $\mathcal{C} = \{(c_1, c_2) \mid c_1 + 2c_2 \equiv 0 \mod (3t+2)\}$. When $n = 3$, let the code $\mathcal{C} = \{(c_1, c_2, c_3) \mid c_1 + 2c_2 + 4c_3 \equiv 0 \mod 7t\}$. For both codes, given a codeword $(c_1, c_2)$ or $(c_1, c_2, c_3)$ in $\mathcal{C}$, for $i = 0, 1, \cdots, t-1$, call the cell state $(c_1 + i, c_2 + i)$ or $(c_1 + i, c_2 + i, c_3 + i)$ the "$i$-shift" of the codeword. The decoding sphere of every codeword consists of the cell states within $L_1$ distance one from one of the $t$ shifts of the codeword. The scrubbing function $f : \mathcal{S}_{n,\infty} \to \mathcal{S}_{n,\infty}$ is defined as follows. If a cell state $\vec{s}$ is at $L_1$ distance one from the $i$-shift of a codeword for some $i \in \{0, 1, \cdots, t-1\}$ and that $i$-shift is above $\vec{s}$, then $f(\vec{s})$ equals the $i$-shift of that codeword. If $\vec{s}$ is at $L_1$ distance one from the $i$-shift of a codeword for some $i \in \{0, 1, \cdots, t-2\}$ and $\vec{s}$ is above that $i$-shift, then $f(\vec{s})$ equals the $(i+1)$-shift of that codeword. If neither of the above two cases is true, then $f(\vec{s}) = \vec{s}$. It is not difficult to verify that the decoding spheres of codewords do not intersect. In fact, for both codes, the decoding spheres form a perfect packing in the $L_1$-metric space of dimension $n$. It is not difficult to see that the densities of the two codes are $1/(3t+2)$ and $1/7t$, respectively. ■

Construction 2 can be generalized to correct other types of errors, including asymmetric errors, errors of large $L_1$-metric sizes, etc., by adjusting the parameters. Due to the space limit, we skip the details.

The $t$-error-scrubbing codes can correct errors whose total size (in the $L_1$ metric) is up to $t$. Let us compare them to conventional error-correcting codes that can correct errors of the same size. For a conventional code, the decoding sphere for a codeword $(c_1, c_2, \cdots, c_n)$ is $B = \{(s_1, s_2, \cdots, s_n) \mid \sum_{i=1}^{n} |s_i - c_i| \le t\}$. Since $|B| = \Omega(n^t)$, by the sphere-packing bound, the density of a conventional error-correcting code is $O(\frac{1}{n^t})$. The density of a $t$-error-scrubbing code, $\Theta(\frac{1}{tn})$, can be substantially better.

When $q$ is finite, some codewords may not be able to scrub $t$ errors because their cell levels are too close to the maximum value $q-1$. In practice, the memory can read cells to see how close they are to the limit of decodability, and adaptively schedule block erasures for refreshing codewords.

## IV. Modular Codes for Limited-Magnitude Errors

In flash memories, errors in cell levels can be caused by several mechanisms, including read disturbs, write disturbs, charge leakage, etc. [1] They often change the cell levels in one direction more significantly than the other. In this section, we consider a more general form of errors, where at most $x \le n$ cell levels can have errors between two scrubbing operations, and the errors make every cell level decrease by at most $d_{min}$ and increase by at most $d_{max}$. That is, the error set is $\varepsilon = \{(e_1, e_2, \cdots, e_n) \mid e_i \in \{-d_{min}, -d_{min}+1, \cdots, d_{max}\}$ for all $i$; $|\{i | e_i \ne 0\}| \le x\}$. The error set studied in the previous section is a special case with $x = 1$ and $d_{min} = d_{max} = 1$. For simplicity, we assume $q \to \infty$. The case of finite $q$ can be processed the same way as before.

We first present a $t$-error-scrubbing code construction of its general form based on the modular technique. The modular technique has been proposed in [2], where strong codes for correcting asymmetric limited-magnitude errors have been presented. In this section, we use the modular technique for error-scrubbing coding.

**Construction 9** *(Modular Error-Scrubbing Code)*

*Build a $t$-error-scrubbing code $\mathcal{C}$ as follows. Let $\ell = d_{min} + d_{max} + 1$. Let $\mathcal{D} \subseteq \{0, 1, \cdots, \ell-1\}^n$ be an error-correcting code of alphabet $\{0, 1, \cdots, \ell-1\}$ and length $n$, which can correct $x$ errors. (For code $\mathcal{D}$, an error is the distortion of one of its $n$ symbols.) Then, a cell state $\vec{c}$ is a codeword in $\mathcal{C}$ if and only if there exists $\vec{s} \in \mathcal{D}$ and non-negative integers $a_1, a_2, \cdots, a_n$ and $b$ such that $\min\{a_1, a_2, \cdots, a_n\} = 0$ and $\vec{c} = \vec{s} + \ell \cdot (a_1, a_2, \cdots, a_n) + b \cdot (t\ell, t\ell, \cdots, t\ell)$.*

*The scrubbing function $f : \mathcal{S}_{n,\infty} \to \mathcal{S}_{n,\infty}$ is defined as follows. Let $\vec{s} = (s_1, s_2, \cdots, s_n) \in \mathcal{S}_{n,\infty}$ be any cell state. If there exists a codeword $\vec{c} \in \mathcal{C}$, an integer $i \in \{0, 1, \cdots, t-1\}$ and an error $\vec{e} \in \varepsilon$ such that $\vec{s} = \vec{c} + i \cdot (\ell, \ell, \cdots, \ell) + \vec{e}$ and $\vec{c} + i \cdot (\ell, \ell, \cdots, \ell) > \vec{s}$, then $f(\vec{s}) = \vec{c} + i \cdot (\ell, \ell, \cdots, \ell)$. If there exists a codeword $\vec{c} = (c_1, c_2, \cdots, c_n) \in \mathcal{C}$, an integer $i \in \{0, 1, \cdots, t-2\}$ and an error $\vec{e} \in \varepsilon$ such that $\vec{s} = \vec{c} + i \cdot (\ell, \ell, \cdots, \ell) + \vec{e}$ and $s_j > c_j + i\ell$ for some $j \in \{1, 2, \cdots, n\}$, then $f(\vec{s}) = \vec{c} + (i+1) \cdot (\ell, \ell, \cdots, \ell)$. If $\vec{s}$ belongs to neither of the above two case, then $f(\vec{s}) = \vec{s}$.*

We now show that the code built by Construction 9 is a $t$-error-scrubbing code.

**Theorem 10** *Let $\mathcal{C}$ be the code of Construction 9. Then, for any two codewords $\vec{c}_1$ and $\vec{c}_2$ of $\mathcal{C}$, we have $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$.*

*Proof:* Let $\vec{c}_1 = \vec{s}_1 + \ell \cdot (a_1, a_2, \cdots, a_n) + b \cdot (t\ell, t\ell, \cdots, t\ell)$ and $\vec{c}_2 = \vec{s}_2 + \ell \cdot (a'_1, a'_2, \cdots, a'_n) + b' \cdot (t\ell, t\ell, \cdots, t\ell)$. Here $\vec{s}_1, \vec{s}_2, a_i, a'_i, b, b'$ are defined the way as in Construction 9. Without loss of generality (WLOG), assume that $a_{j_1} = a'_{j_2} = 0$. For every cell state $\vec{v} = (v_1, v_2, \cdots, v_n)$, define its "*signature*" as $sig(\vec{v}) = (v_1 \mod \ell, v_2 \mod \ell, \cdots, v_n \mod \ell)$. Then $sig(\vec{c}_1) = \vec{s}_1$ and $sig(\vec{c}_2) = \vec{s}_2$. To prove $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$, consider three cases.

- CASE ONE: $sig(\vec{c}_1) \neq sig(\vec{c}_2)$. Let $\vec{v}$ be a cell state in $B_{\vec{c}_1}$. Then, $\vec{v} = \vec{c}_1 + i \cdot (\ell, \ell, \cdots, \ell) + \vec{e}$ for some $i \in \{0, 1, \cdots, t-1\}$ and $\vec{e} \in \varepsilon$. So $sig(\vec{v}) = sig(\vec{s}_1 + \vec{e})$ is a vector in $\{0, 1, \cdots, \ell - 1\}^n$ that is within Hamming distance $x$ from $\vec{s}_1$. Since $\mathcal{D}$ is an error-correcting code that corrects any $x$ errors, $sig(\vec{v})$ cannot be any vector within Hamming distance $x$ from $\vec{s}_2$. So $\vec{v} \notin B_{\vec{c}_2}$. So $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$.
- CASE TWO: $sig(\vec{c}_1) = sig(\vec{c}_2)$, and $b \neq b'$. WLOG, assume that $b < b'$. In this case, let $\vec{s}_1 = \vec{s}_2 = (s_1, s_2, \cdots, s_n)$. Then for any cell state in $B_{\vec{c}_1}$, its $j_1$-th element is at most $s_{j_1} + bt\ell + (t-1)\ell + d_{max} < s_{j_1} + b't\ell - d_{min}$, while the $j_1$-th element of a cell state in $B_{\vec{c}_2}$ is at least $s_{j_1} + a'_{j_1}\ell + b't\ell - d_{min}$. So $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$.
- CASE THREE: $sig(\vec{c}_1) = sig(\vec{c}_2)$, $b = b'$, and there exists some $j_3$ such that $a_{j_3} \neq a'_{j_3}$. WLOG, assume that $a_{j_3} < a'_{j_3}$. In this case, let $\vec{s}_1 = \vec{s}_2 = (s_1, s_2, \cdots, s_n)$. Let $(d_1, d_2, \cdots, d_n) = \vec{s}_1 + \ell \cdot (a_1, a_2, \cdots, a_n) + b \cdot (t\ell, t\ell, \cdots, t\ell) + \alpha \cdot (\ell, \ell, \cdots, \ell) + \vec{e}_1$ be a cell state in $B_{\vec{c}_1}$, and let $(d'_1, d'_2, \cdots, d'_n) = \vec{s}_1 + \ell \cdot (a'_1, a'_2, \cdots, a'_n) + b \cdot (t\ell, t\ell, \cdots, t\ell) + \alpha' \cdot (\ell, \ell, \cdots, \ell) + \vec{e}_2$ be a cell state in $B_{\vec{c}_2}$. Here $\alpha, \alpha' \in \{0, 1, \cdots, t-1\}$, and $\vec{e}_1 = (\beta_1, \beta_2, \cdots, \beta_n)$, $\vec{e}_2 = (\beta'_1, \beta'_2, \cdots, \beta'_n)$ are two errors in $\varepsilon$. Consider two subcases.
  - SUBCASE ONE: $a_{j_3} + \alpha \neq a'_{j_3} + \alpha'$. WLOG, assume that $a_{j_3} + \alpha < a'_{j_3} + \alpha'$. Then, $d_{j_3} = s_{j_3} + bt\ell + (a_{j_3} + \alpha)\ell + \beta_{j_3} \leq s_{j_3} + bt\ell + (a'_{j_3} + \alpha' - 1)\ell + d_{max} < s_{j_3} + bt\ell + (a'_{j_3} + \alpha')\ell - d_{min} \leq d'_{j_3}$.
  - SUBCASE TWO: $a_{j_3} + \alpha = a'_{j_3} + \alpha'$. Since $a_{j_3} < a'_{j_3}$, we get $\alpha > \alpha'$. Then, $d_{j_2} = s_{j_2} + bt\ell + (a_{j_2} + \alpha)\ell + \beta_{j_2} \geq s_{j_2} + bt\ell + (\alpha' + 1)\ell - d_{min} > s_{j_2} + bt\ell + \alpha'\ell + d_{max} \geq d'_{j_2}$.

So in both subcases, $(d_1, d_2, \cdots, d_n) \neq (d'_1, d'_2, \cdots, d'_n)$. Therefore, $B_{\vec{c}_1} \cap B_{\vec{c}_2} = \emptyset$. The theorem is proved. ■

**Theorem 11** *Let $\mathcal{C}$ be the $t$-error-scrubbing code of Construction 9. Let $\mathcal{D}$ be the $x$-error-correcting code used in Construction 9. Let $\ell = d_{min} + d_{max} + 1$. Then, the density of $\mathcal{C}$ is $\frac{|\mathcal{D}|}{t\ell^n}$.*

*Proof:* For every cell state $\vec{v} = (v_1, v_2, \cdots, v_n)$, define its "*signature*" as $sig(\vec{v}) = (v_1 \mod \ell, v_2 \mod \ell, \cdots, v_n \mod \ell)$. For every codeword $\vec{c} \in \mathcal{C}$, we call $\vec{c} + i \cdot (\ell, \ell, \cdots, \ell)$ the $i$-shift of $\vec{c}$. Then the 0-shift, 1-shift, $\cdots$, $(t-1)$-shift of $\vec{c}$ are all in $B_{\vec{c}}$. Let $\vec{d} = (d_1, d_2, \cdots, d_n)$ be a codeword of $\mathcal{D}$. The density of cell states of signature $\vec{d}$ – defined as the number of such cell states divided by $q^n$ – is $1/\ell^n$. We now show that every cell state of signature $\vec{d}$ must be the $i$-shift of a codeword of $\mathcal{C}$ for some $i \in \{0, 1, \cdots, t-1\}$.

Let $\vec{s} = (d_1 + k_1\ell, d_2 + k_2\ell, \cdots, d_n + k_n\ell)$ be a general cell state of signature $\vec{d}$. Let $j$ be the integer in $\{1, 2, \cdots, n\}$ such that $k_j = \min\{k_1, k_2, \cdots, k_n\}$. So $\vec{s} = (d_1 + (k_1 - k_j)\ell + k_j\ell, d_2 + (k_2 - k_j)\ell + k_j\ell, \cdots, d_n + (k_n - k_j)\ell + k_j\ell) = (d_1 + (k_1 - k_j)\ell + \lfloor \frac{k_j}{t} \rfloor t\ell + (k_j \bmod t)\ell, d_2 + (k_2 - k_j)\ell + \lfloor \frac{k_j}{t} \rfloor t\ell + (k_j \bmod t)\ell, \cdots, d_n + (k_n - k_j)\ell + \lfloor \frac{k_j}{t} \rfloor t\ell + (k_j \bmod t)\ell)$. So $\vec{s}$ is the $(k_j \mod t)$-shift of the codeword $(d_1 + (k_1 - k_j)\ell + \lfloor \frac{k_j}{t} \rfloor t\ell, d_2 + (k_2 - k_j)\ell + \lfloor \frac{k_j}{t} \rfloor t\ell, \cdots, d_n + (k_n - k_j)\ell + \lfloor \frac{k_j}{t} \rfloor t\ell) \in \mathcal{C}$.

Since every codeword of $\mathcal{C}$ has exactly $t$ shifts in its decoding sphere, the density of $\mathcal{C}$ is $|\mathcal{D}|/t\ell^n$. ■

Let $\ell = d_{min} + d_{max} + 1$. By the sphere-packing bound, the density of a conventional error-correcting code that can correct $t$ errors in the error set $\varepsilon$ is $O(\frac{1}{n^{tx}(\ell-1)^{tx}})$. So when $\frac{|\mathcal{D}|}{\ell^n} = \Omega(\frac{1}{n^x(\ell-1)^x})$, the $t$-error-scrubbing code can significantly outperform the conventional code. For example, assume $n = 2^m - 1$, $d_{max} = 1$, $d_{min} = 0$, $x = 1$ and $\mathcal{D}$ is the $(2^m - 1, 2^m - m - 1)$ Hamming code. Then the density of the $t$-error-scrubbing code is $\frac{|\mathcal{D}|}{t\ell^n} = \frac{1}{t \cdot 2^m} = \frac{1}{t(n+1)}$, which is asymptotically optimal in $t$ and can be much higher than the density of a conventional code, $O(\frac{1}{n^t})$.

## V. CONCLUSIONS

In this paper, we present the concept of error-scrubbing codes for memory scrubbing in multi-level flash memories without using block erasures. We present two code constructions, and show that error-scrubbing codes can outperform conventional codes because through the actively adjustment of cell levels, the decoding sphere size can be minimized.

## ACKNOWLEDGMENT

This work was supported in part by the NSF CAREER Award CCF-0747415 and the NSF grant ECCS-0802107.

## REFERENCES

[1] P. Cappelletti, C. Golla, P. Olivo and E. Zanoni (*Ed.*), *Flash memories*, Kluwer Academic Publishers, 1st Edition, 1999.

[2] Y. Cassuto, M. Schwartz, V. Bohossian and J. Bruck, Codes for multilevel flash memories: Correcting asymmetric limited-magnitude errors, in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Nice, France, June 2007, pp. 1176-1180.

[3] A. Jiang, V. Bohossian and J. Bruck, Floating codes for joint information storage in write asymmetric memories, in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Nice, France, June 2007, pp. 1166-1170.

[4] A. M. Saleh, J. J. Serrano and J. H. Patel, Reliability of scrubbing recovery-techniques for memory systems, in *IEEE Trans. Reliability*, vol. 39, no. 1, pp. 114-122, 1990.