Luo, T., & Peleato, B. (2016). Spreading Modulation for Multilevel Nonvolatile Memories. IEEE Transactions on Communications, 64(3), 1110–1119.

DOI: 10.1109/tcomm.2016.2518682

Spreading Modulation for Multi-Level Non-Volatile Memories

Tianqiong Luo and Borja Peleato, *Member, IEEE*

***Abstract*—The aggressive scaling of NAND flash memories has caused significant degradation in their reliability and endurance. One of the dominant factors in this degradation is the inter-cell-interference, by which the programming of a cell can affect nearby neighboring cells corrupting the information that they store. This paper proposes a new data representation scheme which increases endurance and significantly reduces the probability of error caused by inter-cell-interference. The method is based on using an orthogonal code to spread each bit across multiple cells, resulting in lower variance for the voltages being programmed in the cells. This new data representation method is also shown to present many of the advantages that spreading sequences bring to wireless communications. For example, multiple information sequences can be written on the same cells at different times without interfering with each other. It also allows storing additional information on an already programmed memory in such a way that the new information is hidden by the noise.**

***Index Terms*—Memory signal processing, NAND flash memory, modulation, spreading sequence, CDMA**

I. Introduction

NAND Flash is a non-volatile memory technology which offers significantly higher speeds and power efficiency than hard drives, but its higher cost is still an obstacle for its widespread use. The cost of flash memories is dominated by the area of silicon that they require per Gigabyte of stored information. During the last decade, manufacturers have aggressively scaled the technology to pack more cells in the same silicon real estate, while they also increased the number of bits stored in each cell. These techniques succeeded in reducing the cost of flash memories to the same order of magnitude as that of hard drives, but they brought other problems. One of the main problems that flash memories are facing today is the reliability of the stored information [1].

A flash cell is a floating gate transistor whose threshold voltage can be adjusted by injecting charges into its floating gate. Information is stored by setting this voltage threshold to specific values. In its simplest form, one bit is stored in each cell, depending on whether it is charged or discharged. Memories of this type are known as SLC. In order to increase the capacity (and reduce their cost accordingly) most applications now use MLC memories, which can be programmed to four different voltage levels and store two bits in each cell. Some manufacturers have gone even further, producing memories which store three (TLC) or even four bits in each cell [2][3].

As Flash memory technology scales and more bits are stored in each cell, the signal to noise ratio observed in the programmed voltages decreases. One of the main sources of noise, which is becoming increasingly important as the technology scales and is expected to get even worse for the forthcoming 3D flash structures, is inter-cell interference (ICI) [4] [5]. The shift in threshold voltage of one cell can change the threshold voltage of its neighbors due to the parasitic capacitance-coupling effect [6]. Extensive measurements have shown that the ICI noise created by a cell is proportional to the voltage to which it is being programmed [7]. Other sources of noise include Gaussian noise, caused by overprogramming and charge leakage, and impulse noise, caused by defective or broken cells [8].

Additionally, flash cells have a limited lifetime. Before data can be written to a page[1], the block must have been erased (i.e., all the cells need to be discharged). The tunneling of charges into and out of the floating gate causes damage to the dielectric barrier that holds the charges, limiting the range of programmed voltages and the number of times that each cell can be written. The amount of damage that a cell suffers in a single write operation increases super-linearly with the programmed voltage [9]. Hence, writing data patterns that are represented by a lower threshold voltage could prolong the lifetime of the flash [10], [11], [12], [13].

This paper extends the work done in [14]. We study spreading modulation, a new data representation scheme which, among other features, reduces ICI and extends the lifetime of the memory by reducing the frequency with which the largest voltage levels are programmed. We analyze its performance under different noise conditions and show some of its applications in soft decoding and information security. The proposed modulation is based on using an orthogonal code to spread each information symbol across multiple cells, similar to how DS-CDMA is used in wireless communications [15] [16], reducing the variance in the programmed voltages. Instead of sacrificing capacity by using a modulation code as proposed by [17] and [18], this scheme increases the number of possible voltages to be programmed, disregarding the fact that they might overlap, and imposes a code over the extended space of voltage levels. It can therefore be used to store M symbols into

T. Luo and B. Peleato are with the Department of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907. (e-mail: luo133@purdue.edu; bpeleato@purdue.edu)

DOI: 10.1109/TCOMM.2016.2518682

[1]Cells in a NAND flash are grouped into pages, which is the smallest unit for write and read operations. Pages are grouped into blocks, which is the elementary unit for erase operations

N cells, just like the regular scheme, but with better endurance and SNR.

The proposed scheme increases the number of voltage levels, so reading and programming a page might take longer. In the read operation, the voltage of the cells in a flash memory cannot be measured directly. Reading the cells in a page is done by comparing their stored voltage with an adjustable reference voltage t. The read operation returns a binary vector with one bit for each cell: 1 if the cell has voltage lower than t and 0 otherwise. However, many memory manufacturers now incorporate dedicated hardware to perform multiple reads of the same page with different reference voltages to obtain a finer quantization of the voltages. These commands are generally used to perform soft reads for LDPC decoding, but they can also be used to accelerate the read of additional levels that our scheme proposes [19]. The penalty in terms of read speed might be alleviated.

The programming is done by sending high voltage pulses into one wordline. After each pulse, a verify read is performed and cells which have reached the desired level of charge are inhibited from further programming. This programming method is called ISPP [20]. Since the programming already involves multiple program-verify iterations, the penalty due to additional levels might not be as severe as it might seem at first.

The rest of the paper is organized as follows. Section II introduces the system model used in the rest of the paper. Section III explains the spreading data representation approach, analyzing its performance under different types of noise, and Section IV provides guidelines on how to adjust the spreading parameter. Sections V and VI respectively show how the spreading approach can be used to generate soft information and to hide data in the memory. Finally, Section VII presents simulation results to validate the method and Section VIII summarizes and concludes the paper.

II. System model

In order to better illustrate the features of the proposed scheme, this paper will consider multiple scenarios with different noise distributions and memory types. From a high level perspective, it will be assumed that in a write operation the host provides a vector of (possibly encoded) information symbols $\mathbf{b}$ $\mathcal{X}^M$ from an alphabet $\mathcal{X}$, which are then mapped to a vector of voltages $\mathbf{v}^0$ to be programmed on the cells. By the time that the cells are read, the voltages $\mathbf{v}^0$ will have suffered some amount of white Gaussian noise [21], [4], denoted $\mathbf{n}^w$, as well as inter-cell interference (ICI), denoted $\mathbf{n}^{\mathrm{ICI}}$. Therefore, the voltage actually stored in the cells at read time is

$$\mathbf{v} = \mathbf{v}^0 + \mathbf{n} \qquad \mathbf{n} = \mathbf{n}^w + \mathbf{n}^{\mathrm{ICI}} \tag{1}$$

The noise due to leakage is also assumed to be Gaussian and is therefore absorbed into the $\mathbf{n}^w$ term.

ICI occurs when a shift in the threshold voltage of one cell changes the threshold voltage of its neighbors due to the parasitic capacitance between cells, known as "floating-gate interference" [6]. Extensive measurements have shown that the change in threshold voltage suffered by the victim cell is proportional to the threshold voltage of the aggressor cell, with a proportionality factor that depends on the parasitic capacitance between the aggressor cell and the victim cell. This factor is commonly known as coupling ratio and will be denoted by . Hence,

$$\mathbf{n}^{\mathrm{ICI}} = \mathbf{v}_{\mathrm{aggressor}} \tag{2}$$

With the usual data representation scheme, each symbol b is mapped to a fixed nominal voltage v^0. So, for the sake of simplicity, it will be assumed that they both share the same alphabet $\mathcal{X}$ and $b = v^0$. In SLC memories these symbols are binary, in MLC they can take four values (representing two bits of information), in TLC they take 8 values (3 bits), etc. In general, the number of levels is chosen to be as large as possible while still avoiding potential overlap between the levels and excessive damage when programming the largest voltage, denoted $V_{\max}$.

Damage to flash memory cells is caused by program/erase (P/E) cycling. According to [9] and the experimental results presented in Section VII, the damage suffered by a cell when programmed to a voltage V_{th} is approximately proportional to V_{th}^2. Most of the damage happens when cells are programmed to the largest voltage $V_{\max}$, so writing data patterns that are represented by a lower threshold voltage could prolong the lifetime of the flash [10], [9], [12].

The proposed data representation scheme will use a linear mapping between the symbols $\mathbf{b}$ and the nominal voltages $\mathbf{v}^0$, to be described in the next section. This mapping will extend the number of possible voltages to be programmed, but also reduce the number of cells programmed to $V_{\max}$, attenuate the ICI, and increase robustness to impulse noise. This will result in increased capacity and extended lifetime for the memory.

In practice, the discharged state in which the cells are left after being erased sets a lower limit for the range of programmable voltages and the write procedure can only push the cells towards higher voltages. Thus it is not possible to program NAND flash cells with a negative voltage. However, for our derivations it will be useful to assume that the range of programmable voltages is symmetric and the voltages $\mathbf{v}^0$ and symbols $\mathbf{b}$ can take both positive and negative values. SLC cells will therefore have their voltage levels relabeled as 0 5 and +0 5, while MLC cells will be assumed to take voltage levels 1 5, 0 5, 0 5, and 1 5. In general, if there are $2S$ symbols in the alphabet $\mathcal{X}$, they will be labeled as $\mathcal{X} =$ 0 5 1 5 (S 0 5) . The largest symbol in the alphabet will be denoted by $V_{\max} = S$ 0 5. This represents a simple shift of the physical reference system. The rest of the paper assumes that the symbols b_i and voltages v_i^0 have zero mean.

III. The spreading approach

This section introduces the spreading modulation and then analyzes its performance against three types of noise: Gaussian, ICI, and impulse noise. Subsection III-A studies the trade-off between damage and Gaussian noise. The SNR can be increased by widening the range of programmed voltages,

but doing so increases the damage suffered by the cells. Subsections III-B and III-C study how the proposed modulation can attenuate ICI and impulse noise, respectively.

In a traditional flash memory, each cell stores a fixed number of bits. There is usually some redundant bits introduced by the ECC or RAID schemes but, ultimately, each bit is stored in a specific cell. Cells have a fixed number of voltage levels to which they can be programmed and all the levels are written with the same frequency. This section proposes a new data representation scheme which uses orthogonal codes to spread each bit across multiple cells, similar to DS-CDMA transmission in wireless communications. This data representation scheme reduces the variability of the voltages being programmed in the cells, resulting in improved endurance and additional robustness towards impulse noise and ICI.

Instead of mapping each symbol b_i to a fixed voltage v_i^0, the proposed scheme uses a matrix with orthogonal columns $\mathbf{C}$ (e.g., a Walsh matrix) to map the symbols $\mathbf{b}$ into voltages $\mathbf{v}^0$ to be programmed. For example, when mapping four symbols $\mathbf{b}$ $\mathcal{X}^4$ into four cells, the voltages to be programmed are:

$$\begin{matrix} v_1^0 \\ v_2^0 \\ v_3^0 \\ v_4^0 \end{matrix} = \frac{k}{4} \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{matrix} \begin{matrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{matrix}$$

where k is an adjustable parameter that controls the range of voltages being programmed. By scaling k, we can introduce more separation between the programmed voltage levels, but the damage suffered by the cells and the power consumed would also increase. In general, when M symbols are to be programmed into N M cells,

$$\mathbf{v}^0 = \frac{k}{M}\mathbf{C}\mathbf{b} \tag{3}$$

where $\mathbf{C}$ is a $1\ 1^{N\times M}$ matrix with orthogonal columns and kV_{max} is the maximum voltage to be programmed (remember that $\mathbf{v}^0$ are not the programmed voltages, but shifted versions of them). By scaling k, we can introduce more separation between the programmed voltage levels, but the noise is not affected by this scaling. We will refer to this operation as spreading.

By spreading each information symbol across multiple cells, we increase the number of possible programmed voltages in each cell, so symbols and nominal voltages no longer share the same alphabet. For example, in the SLC case where b_i $0\ 5\ 0\ 5$ and $M = N = 4$, our scheme would have five possible levels for each cell: v_i^0 $0\ 5k$ $0\ 25k\ 0\ 0\ 25k\ 0\ 5k$. In general, if $V_{\max}$ is the largest symbol in the alphabet $\mathcal{X}$, the voltage levels after spreading are in the range $[\ kV_{\max}\ kV_{\max}]$.

When the read operation is performed, the voltages are multiplied by a de-spreading matrix $\mathbf{C}^T$, which is the left inverse of the spreading matrix. Because of the properties of Walsh sequences, the de-spreading matrix is the transpose of the spreading matrix. Continuing with the previous example,

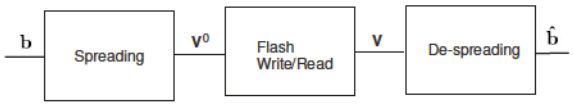

Fig. 1. Illustration of the spreading approach.

when $N = M = 4$

$$\begin{matrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{matrix} = \frac{1}{k} \begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{matrix} \begin{matrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{matrix}$$

where b_i $i = 1\ 2\ 3\ 4$ represent the information estimates after reading. In general,

$$\mathbf{b} = \frac{M}{Nk}\mathbf{C}^T\mathbf{v} \tag{4}$$

The processes of spreading and de-spreading discussed above are shown in Fig. 1.

Combining Eqs. (1), (3), and (4), the estimated information symbols can be represented as:

$$b_i = b_i + \frac{M}{Nk}\sum_{j=1}^{N} n_j \quad i = 1\ 2 \quad M \tag{5}$$

The noise can be arbitrarily attenuated by decreasing $\frac{M}{Nk}$, but that involves sacrificing capacity by decreasing $\frac{M}{N}$ or using a wider range of programmed voltages by increasing k. Since most practical applications are not willing to compromise capacity, the rest of the paper assumes $M = N$, which means that the storage space is the same as in the regular scheme.

A. Gaussian noise and damage

For fixed voltage range ($k = 1$) and signal-independent Gaussian noise, spreading actually decreases the signal-to-noise ratio (SNR) at read time. Assuming independent and identically distributed noise components n_i $\mathcal{N}(0\ \ ^2)$ [21], the SNR of the regular and spreading schemes are:

$$SNR_{\text{regular}} = \frac{P_s}{^2} \qquad SNR_{\text{spread}} = \frac{P_s}{\frac{N}{k^2}\ ^2} \tag{6}$$

where $P_s = E[b_i^2]$ represents the power of the stored symbols. It is easy to increase SNR_{spread} when needed by increasing the scaling constant k, but doing so widens the range of programmed voltages and thus causes more damage and consumes more power. This subsection studies such tradeoff.

One of the advantages of the spreading scheme is that it reduces the probability of programming the maximum voltage as shown in Fig. 2, thus reducing the damage to the flash memory. The amount of damage suffered by the memory is approximately proportional to the square of the voltage programmed. As mentioned in Section II, cell voltages must be non-negative so in practice they are shifted to $b_i + V_{\max}$ in the regular scheme and to $v_i^0 + kV_{\max}$ in the spreading scheme when $E[b_i] = 0$ and $E[v_i^0] = 0$. Denote T_{spread} and

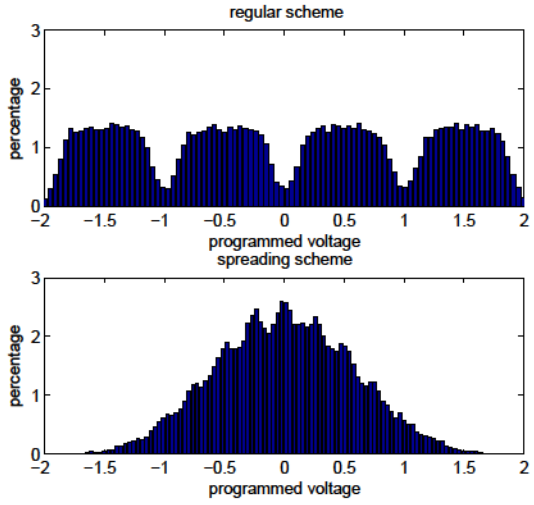

Fig. 2. Distribution of cell voltages for both modulation schemes when M=N=4, k=1, = 0 1, = 0 2. Spreading leads to a distribution with less variance.

T_{regular} the damage with the spreading and the regular scheme, respectively. Then,

$$\begin{aligned} T_{\text{regular}} &= a \quad E \quad (b_i + V_{\max})^2 \\ &= a(E[b_i^2] + V_{\max}^2) \\ T_{\text{spread}} &= a \quad E \quad (v_i^0 + kV_{\max})^2 \\ &= a \quad \frac{k^2}{N} E[b_i^2] + k^2 V_{\max}^2 \end{aligned} \tag{7}$$

for some constant a. For $k = 1$ (i.e., both schemes have identical programming range), spreading causes less damage than the regular scheme but it lowers the SNR. For $k = \overline{N}$ both schemes have the same SNR, but spreading causes more damage. Section IV will elaborate how to choose an optimal k in between.

B. Inter-cell interference

The previous section showed that when the noise is independent from the voltages being programmed, our spreading scheme does not provide any improvement in terms of BER unless $k \quad \overline{N}$. However, the main source of noise in new memory generations is ICI, which is proportional to the voltages being programmed in the cells.

In most memories, f ash cells are organized in an array structure as shown in Fig. III-B, where all the cells in a wordline are programmed simultaneously and wordlines are programmed in increasing order. The ISPP [20] algorithm used to program wordlines can compensate for the inter-cell interference caused by previously programmed wordlines, but not for the interference of subsequent program operations. Hence, most of the ICI suffered by a specif c cell is caused by the direct-neighbor. This will be the only ICI component considered in our analysis, but the simulations in Section VII will include 3 neighbors.

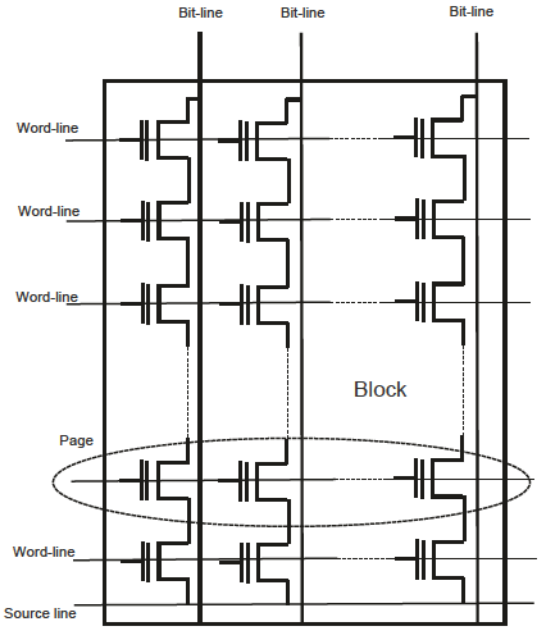

Fig. 3. Bitline-Wordline structure of f ash memory.

Assuming $n^{ICI} \quad n^w$ and $M = N$, Eq. (5) becomes

$$b_i = b_i + \frac{1}{k} \sum_{j=1}^{N} n_j^{\text{ICI}} \quad i = 1 \ 2 \quad M$$

where n^{ICI} is proportional to the programmed voltages. According to Eq. (2), n^{ICI} can be represented as:

$$n^{\text{ICI}} = \frac{k}{N} \sum_{j=1}^{N} b_j$$

So the estimated symbol can be represented as $b_i = b_i + b_{\text{spread}}$, where

$$b_{\text{spread}} = \overline{N} \sum_{j=1}^{N^2} b_j$$

If the distance between the symbols is d, errors happen only when $b_{\text{spread}} \quad \frac{d}{2}$. The distribution of b_{spread} is approximately $\mathcal{N}(0 \quad {}^2E[b_i^2])$ when N is large according to the Central Limit Theorem. Then the probability of error for non-extreme symbols[2] is approximately

$$P_e^{\text{spread}} \quad 2 \quad \frac{d}{2 \quad \overline{E[b_i^2]}} \tag{8}$$

where $(u) = \int_{-\infty}^{u} \frac{1}{\sqrt{2}} e^{-\frac{y^2}{2}} dy$. Note that in Eq. (8), the scaling parameter k has no effect on ICI.

In the regular scheme, the estimated symbol is:

$$b_i = b_i + \quad b_j$$

[2]Non-extreme symbols refer to the symbols which are not programmed to the highest or lowest voltage levels.

with probability of error for non-extreme symbols

$$P_e^{\text{regular}} = P \quad b_j > \frac{d}{2} \tag{9}$$

The main advantages of the proposed spreading scheme comes from the fact that it leads to less variance in the programmed voltages than the regular scheme, as shown in Fig. 2. As increases, the regular scheme introduces much more probability of error than our spreading scheme. For example, if = 0 35 and $\mathcal{X} =$ 0 5 1 5 then $d = 1$ and for MLC memories

$$P_{e(MLC)}^{\text{regular}} \quad 0\ 25 \qquad P_{e(MLC)}^{\text{spread}} \quad 0\ 2 \tag{10}$$

for intermediate (non-extreme) symbols when Gaussian noise is negligible according to Eq. (9) and Eq. (8). The lowest and highest symbol would suffer half as much probability of error in both cases.

As increases, P_e^{spread} increases slower than P_e^{regular}. So, when is large enough, the spreading scheme has a better performance. It is also important to take into account that the grouping of cells into spreading blocks must be done carefully. If the same cells, say 1 4, were taken as a spreading block in two consecutive wordlines, the ICI would have the form of a scaled codeword, and would therefore not be attenuated by the de-spreading.

C. Impulse noise

Another important advantage of the spreading approach lies on its increased robustness to impulse noise. Flash memories are currently being used in a wide variety of environments. In most of them they compete with HDD and DRAM but there are some cases in which f ash is the only viable option. One of those cases are satellite applications. Hard drives have moving parts, and need a certain air pressure for the head to f y appropriately. DRAM memories are volatile and require frequent refreshing to avoid losing the information. Flash memories, however, are perfect for satellite applications. Their lack of moving parts makes them very compact and shock resistant, and they can be powered off for extended periods of time without losing information.

Satellites suffer a signif cant amount of radiation, constituting one of the leading causes of electrical component failures [22]. A high energy particle impacting on a NAND f ash cell usually causes what is known as a stuck-at defect [8]. The cell effectively breaks and will henceforth be read as storing the same voltage value, regardless of what it was meant to be programmed to. In the regular scheme, any bit written to that cell will most likely be lost. The scheme proposed in this paper, on the other hand, spreads each bit across multiple cells, and has a chance to recover the bit even if one of the cells is stuck at a given value.

Broken cells can usually be identif ed before they are read. The ISPP programming mechanism checks the cell voltages after sending each pulse and, when the controller detects that the cell voltage has not changed after having sent multiple pulses, the cell is marked as broken. If this were known *before* the programming started, we could just ignore that cell altogether and not store anything in it. Unfortunately, if the broken cell is detected *during* programming, it is too late to stop the programming of the other cells in the page.

Let p denote the probability of a cell breaking. We assume that the controller knows which cells are broken, and can therefore assign them an arbitrary voltage at read time, independently from the actual state they are in. In order to minimize the resulting noise, broken cells will be read as having a voltage of 0, the average voltage stored by a healthy cell.

Equation (3) shows that the nominal voltage programmed in the i-th cell is $v_i^0 = c_i^T b$, where c_i^T represents the i-th row of the spreading matrix $\mathbf{C}$. If the i-th cell is broken, the controller interprets $v_i = 0$, which is equivalent to replacing i-th column of the de-spreading matrix by zeros when the read operation is performed. Denote by $\mathbf{C}$ the matrix $\mathbf{C}$ with the i-th row replaced by zeros, the estimated data symbols can then be represented as:

$$\begin{aligned}\mathbf{b} &= \frac{1}{N}\ \mathbf{C}^{\mathbf{T}}\ \mathbf{C}\ \mathbf{b} \\ &= \frac{1}{N}\begin{matrix} N & 1 & 1 & 1 & b_1 \\ 1 & N & 1 & 1 & b_2 \\ \vdots & & & \vdots & \vdots \\ 1 & 1 & N & 1 & b_N \end{matrix}\end{aligned}$$

where all the noise except impulse noise has been neglected. In other words, the estimated information symbol b_i can be represented as:

$$b_i = \frac{N \quad 1}{N} b_i + \frac{1}{N} \quad b_j \quad i = 1\ 2 \quad N$$

where the signs of the error terms depend on the spreading matrix and the signs of the different bits.

In SLC f ash memories, an error will occur if the sign of b_i is different from the sign of b_i. This can only happen if the signs of the other bits align just right so that the N 1 error terms cancel out the correct $\frac{N-1}{N}$ contribution. It happens with probability $\frac{1}{2^{N-1}}$. Moreover, even when the signs align just right to give $b_i = 0$, we still have a 50% chance of guessing the sign correctly. So the probability of error due to broken cells is $\frac{N}{2^N} p(1 \quad p)^{N-1} + O(p^2)$.

However, for the regular scheme, whatever bits were stored in the broken cells are completely lost. The ECC will have to recover them if possible. If a cell is broken, it has a $\frac{1}{2}$ chance of storing the correct value, so the probability of error is $\frac{p}{2}$ for the regular scheme, which is much larger than with the spreading scheme.

In MLC f ash memories, however, our scheme no longer offers advantages towards impulse noise. Since there are more programming voltage levels, the error terms may play a more important role because it may contain some large voltage levels. But space applications generally use SLC memories because they are more reliable than MLC.

IV. Choice of k

Increasing k can improve SNR through noise attenuation, but the range of programmed voltages becomes wider. It was

shown in Fig. 2 that the probability of programming a very large or small voltage with the spreading scheme is very low, so it could be helpful to increase k and then crop those extremes. If the gains in terms of noise attenuation obtained by increasing k make up for the cropping noise, the overall SNR will increase.

Instead of increasing k and then cropping the largest voltages, our scheme crops both high and low voltages symmetrically, so as to minimize the cropping noise. Assuming that the desired range of programmed voltages is $[\ V_{\max}\ V_{\max}]$, the quantization noise introduced by cropping is

$$q_i = \begin{array}{ll} 0 & \text{if } v_i^0 \quad [\ V_{\max}\ V_{\max}] \\ v_i^0 + V_{\max} & \text{if } v_i^0 > V_{\max} \\ v_i^0 \quad V_{\max} & \text{if } v_i^0 < \quad V_{\max}, \end{array}$$

where $i = 1\ 2 \quad N$ and v_i^0 is the i-th component of the programmed voltage $\mathbf{v}^0$ defined in Eq.(3). The information symbols read can be represented as:

$$\mathbf{b} = \mathbf{b} + \frac{1}{k}\mathbf{C}^T\mathbf{n} + \frac{1}{k}\mathbf{C}^T\mathbf{q}$$

where $\mathbf{n}$ is write noise and ICI noise as defined in Eq.(1) and $\mathbf{q} = [q_1\ q_2 \quad q_N]^T$ is the quantization noise.

In other words, for each estimated information value b_i:

$$b_i = b_i + \frac{1}{k}\sum_{j=1}^{N}(n^{\mathrm{ICI}} + n^w) + \frac{1}{k}\sum_{j=1}^{N} q_j$$

To minimize the overload distortion introduced by cropping, we hope to crop only the largest and smallest voltages in our scheme, kV_{max}. These levels are programmed with probability $\frac{2}{L^N}$, where L is the number of possible voltage levels, hence q_j can be represented as:

$$q_j = \begin{array}{ll} (k \quad 1)V_{\max} & \text{with probability } \frac{2}{L^N} \\ 0 & \text{with probability } \frac{L^N - 2}{L^N} \end{array}$$

The Gaussian noise, ICI, and quantization noise are uncorrelated, so the total noise power P_N can be found by a simple sum of the components $P_N = P_w + P_{\mathrm{ICI}} + P_q$, where:

$$\begin{aligned} P_w &= \frac{N}{k^2}\ ^2 \\ P_{\mathrm{ICI}} &= \ ^2E[b_i^2] \\ P_q &= \frac{2N(k \quad 1)^2}{k^2L^N}V_{\max}^2 \end{aligned} \tag{11}$$

As k increases the write noise decreases but the quantization noise increases. There is a trade-off between quantization noise and write noise. As shown in Fig. 4, the optimal k which minimizes the total noise power and consequently maximizes the SNR is:

$$k \ = \arg\min_k \ \frac{2N(k \quad 1)^2}{k^2L^N}V_{max}^2 + \frac{N}{k^2}\ ^2$$

The scaling parameter k should not be too large, so that the range of the programmed voltage of the spreading approach is close to that of the regular scheme. However, if k is small, the distance between any two adjacent levels will be reduced or

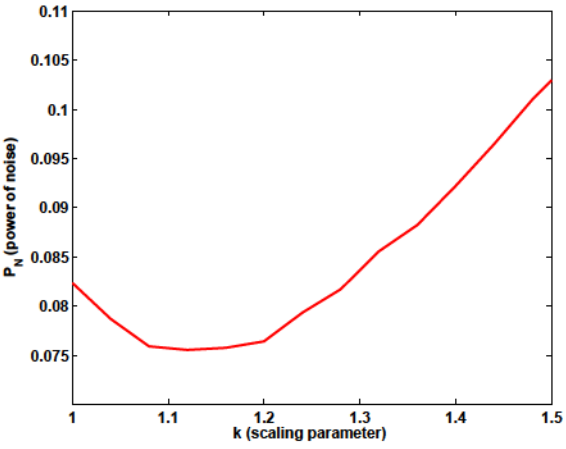

Fig. 4. Quantization noise power as a function of for a SLC flash memory with M=N=4, $= 0\ 1$, and $= 0\ 2$

compressed. For some memories, it may be hard to control the small voltage increments between the levels in the spreading scheme, specially if k is small. The over-programming could introduce Gaussian noise, but the total power of this noise would still be lower than that in the regular scheme, since the programming pulses would also be smaller.

In addition to cropping, there are other ways to increase SNR and at the same time maintain the same programming range: we can reassign the programmed voltages to reduce the probability of error. That is, we can increase the distance between the voltage levels with higher probability and decrease the distance between the voltage levels with lower probability. This scheme is more complex than cropping and is suitable for flash memories with high computational capability. We will not discuss it in detail in this paper.

V. Obtaining soft input

There are two types of decoders in flash: hard-decoders and soft-decoders. The difference between them lies in the input and output dictionary: Hard-decoders usually have the same input and output dictionary which is a fixed set of deterministic symbols; Soft-decoders operate on log-likelihood ratios (LLR), specifying the probability of each input being a noisy version of each symbol. Soft-decoders can correct more errors, but they require more reads and a more complex decoding algorithm. Some flash controllers use a hard-decoder when BER is low and switch to a soft one when the former one fails [23].

In flash memories, cells are read by comparing their voltage with a number of reference thresholds. If a total of l reads have been performed on a page, each cell can be classified as falling into one of the $l+1$ intervals between the read thresholds. The problem of reliably storing information on the flash is therefore equivalent to the problem of error-free transmission over a Pulse amplitude modulation (PAM) channel [24]. The channel inputs represent the levels to which the cells are written, the outputs represent read intervals, and the channel transition probabilities specify how likely it is for cells programmed to

P_{ij}	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$i=1$	0.0555	0.2048	0.1784	0.0578	0.0041
$i=2$	0.0040	0.0580	0.1786	0.2034	0.0555
LLR_j	2.6301	1.2616	-0.0011	-1.2582	-2.6054

TABLE I
TRANSITION PROBABILITIES AND LLR VALUES FOR SLC CELLS.

a specific level to be found in each interval at read time [25] [12].

When we perform the minimum required number of reads on a page, cells can only be classified into the nominal symbols. However, if we perform additional reads, we can achieve a finer quantization of the cell voltages. It is then possible to assign an LLR value to each of these voltage intervals. The LLR value associated with a read interval r between level i and level j is defined as $LLR_r = \log(P_{ir}/P_{jr})$, where P_{ab} denotes the transition probability from a to b. A hard decoder takes a greedy approach mapping each interval to the most possible nominal symbol and returning the closest codeword. A soft decoder operates on the LLR values and uses those probabilities to perform a maximum likelihood estimation of the codeword.

As mentioned in section III, the spreading approach brings more possible programming voltage levels and requires more reads to distinguish them. This results in a finer quantization of the cell voltages and provides soft inputs to the decoder. This holds even if we reduce the number of reads to be the same as in the regular scheme, since the de-spreading step will combine the read voltages increasing the total number of possible values. For example, conventional SLC memories use a single read to classify the cells into two states. The channel with the regular scheme is then equivalent to a Binary Symmetric Channel (BSC). In the spreading scheme, however, a single read will still classify each cell into one of two states, but after de-spreading with $N = 4$, each component can take five possible values. The channel observed by each symbol is then equivalent to the PAM channel with five outputs illustrated in Fig. 5. As an example, Table I shows the transition probabilities for Fig. 5 when write noise is Gaussian with variance $\sigma = 0.3$ and ICI parameter $\gamma = 0.5$. Soft information plays an important role when noise is large and helps to minimize the probability of error. In our numerical simulation with strong noise in SLC mentioned above(i.e., write noise with $\sigma = 0.3$ and ICI noise with $\gamma = 0.5$), the regular scheme causes probability of error 0.1043 and the spreading scheme causes probability of error 0.0893.

Fig. 6 shows the symmetric capacity (i.e., capacity under uniform distribution of inputs) of the channel in the MLC case with write noise $\mathcal{N}(0, 0.1^2)$, as a function of the ICI parameter γ. When the ICI is weak, the regular scheme (using 3 reads) has higher capacity than the spreading one, even when the latter uses 12 reads. However, the capacity of the regular scheme decreases rapidly when the ICI increases, falling below the capacity of the spreading scheme, even when the latter uses 3 reads.

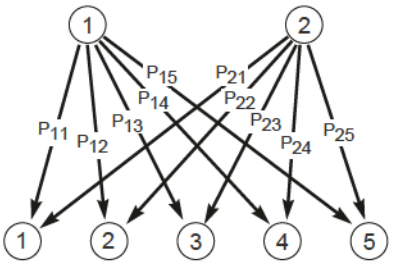

Fig. 5. PAM channel equivalent to SLC flash read channel in spreading scheme with $M = N = 4$.

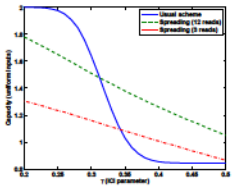

Fig. 6. Comparison of the channel capacity of the spreading scheme and regular scheme for MLC flash.

VI. SECURITY

Section III has shown how spreading can be beneficial to reduce the probability of error when ICI is large. This section will show that it can also be used to hide information using a technique known as superposition coding [26]. This technique has been widely used in Direct-sequence spread spectrum (DSSS) communications to make spread-spectrum signals appear wide-band and noise-like, thus making them hard to detect [27].

The key idea of hiding information using superposition coding is making the modulated hidden information look like additional noise. In this paper, we are going to use a long Pseudo noise (PN) sequence [28] to spread a single symbol of hidden information over many cells. This will create a very long sequence of voltage components, which will be added on top of the original information stored in flash in plain view. The spreading and de-spreading process are described as follows: Denote the spreading sequence (PN sequence) by $\mathbf{d} \in \{+1, -1\}^L$ and a hidden information symbol by h. The voltage components for the hidden information can be represented as

$$\widetilde{\mathbf{v}} = \varepsilon \mathbf{d} h,$$

where ε is a (small) scaling parameter that controls the range of programmed voltages.

The combined voltage $\mathbf{v}^c$ for the original and hidden information is:

$$\mathbf{v}^c = \mathbf{b} + \varepsilon \mathbf{d} h + \mathbf{n},$$

where $\mathbf{b}$ is the vector of L plain view information symbols and $\mathbf{n}$ is the noise as defined in Eq. (1).

If the distribution of the combined voltage $\mathbf{v}^c$ still looks similar to the original information to any unauthorized reader, hence making it difficult for them to notice the existence of the hidden information. For example, as shown in Fig. (7), the distribution of the combined voltage of the original symbols

and hidden information is still similar to the distribution of the original information when we choose the scaling factor appropriately. To make the combined voltage as random as possible, we may decrease so that the power of hidden information is reduced. However, small also brings higher probability of error when decoding the hidden information because write noise will play a comparatively larger inf uence.

Two steps are required to decode the hidden information. The f rst step is subtracting the original information. Assuming that we can recover the original information with low probability of error, the voltage left after subtracting the original information is approximately:

$$\mathbf{v}^{\mathrm{s}} \quad \mathrm{d}h + \mathbf{n}$$

The second step is de-spreading using $\mathbf{d}$, the decoded hidden information symbol is:

$$h = \frac{\mathbf{v}^{\mathrm{s}}\mathbf{d}^{T}}{L} = h + \sum_{i=1}^{L} \frac{1}{L} n_i$$

where L is the length of the spreading sequence.

Assume the write noise to be Gaussian with variance ^{2}, the SNR after the de-spreading process is:

$$SNR_{\mathrm{PN}} = \frac{P_s L^{\ 2}}{^2} \tag{12}$$

where P_s is the power of each hidden information symbol.

According to Eq. (12), another way to increase the accuracy of the recovered hidden information is to increase the length of the spreading sequence L. As shown in Fig. 8, P_e^{hidden} is lower for the same P_e^{original} (equivalently, for the same noise variance) when the length of the spreading sequence L is larger; and vice versa. However, as L increases, so does the number of cells required to store each hidden information symbol for hidden information, thereby reducing the effective capacity of the memory.

In order to both increase the accuracy of the recovered hidden information and save storage space, we may group several information symbols together. Grouping means that we can write several information symbols in one group of cells using orthogonal spreading sequences and decode them separately. For example, Fig. 8 shows that choosing the spreading sequence length to be $L = 32$ and using two overlapping orthogonal sequences has a better performance than the case with $L = 16$ and a single sequence, despite both schemes use the same storage space.

Additionally, the spreading approach supports multiple access: different hidden information sequences can be written at different times using orthogonal spreading sequences and without erasing the cells between writes. This can be achieved by shifting v up to be non-negative, so that each write only needs to introduce a small voltage increment on the cells, and shifting the read voltages down before despreading.

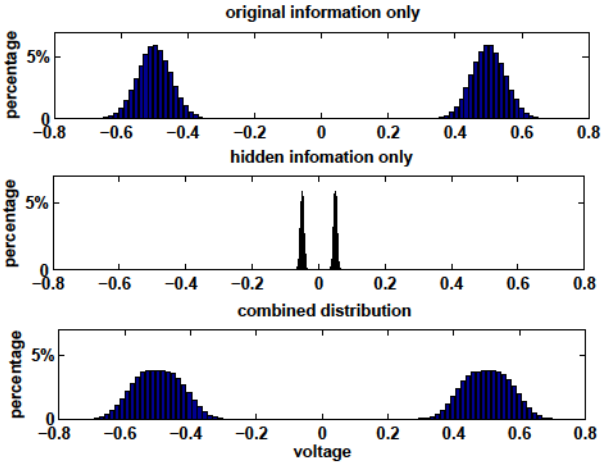

Fig. 7. voltage distribution for a SLC cell with $= 0\ 1$, spreading factor $= 0\ 1$ and length of spreading sequence $= 16$.

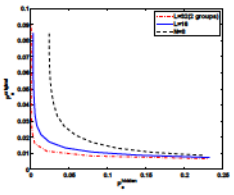

Fig. 8. SLC cell with $= 0\ 2$. e^{original} represents the probability of error of the decoded original information and e^{hidden} represents the probability of error of the decoded hidden information.

VII. Simulation Results

This section compares the proposed data representation scheme with the traditional one through simulations. It evaluates both of them in terms of BER and damage caused to the memory. We simulate 10 memory blocks with 128 pages per block and 8096 cells in each page. Each cell is assumed to suffer ICI from 3 neighbors in the next wordline, with ICI coeff cients $(\ _y \ \ _{xy}) = (0\ 08\ \ 0\ 006)$, where $_y$ is the ICI coeffcient for the direct neighbor (the one in the same bitline) and $_{xy}$ is the ICI coeff cient for the two diagonal ones [4] [29]. The write noise is assumed to be Gaussian with zero mean and $= 0\ 1$, so $n^w \quad \mathcal{N}(0\ \ 0\ 1^2)$.

First, we study how BER increases with ICI when $M = N$, so that the storage eff ciency is the same as that in the regular scheme. Assume $M = N = 4$, $k = 1\ 1$, and the voltages v^0 are cropped to be in the range $[\ \ 1\ 5\ \ 1\ 5]$ for a MLC memory and $[\ \ 3\ 5\ \ 3\ 5]$ for a TLC memory. The f rst two curves in Fig. 9 and Fig. 10 (no redundancy) show the results for MLC and TLC, respectively. When ICI is small the regular scheme performs better in both MLC and TLC cells, but when ICI increases, the spreading scheme provides lower BER.

We also study the case when there is redundancy. We assume $N = 4$ and $M = 3$, that is, spreading 3 symbols over 4 cells, so that the code rate is 75% in the spreading scheme. In the regular scheme, we use $(15\ \ 11)$ Hamming code to encode the input information so that the code rate is almost the same as that in the spreading scheme. The last two curves in Fig. 9

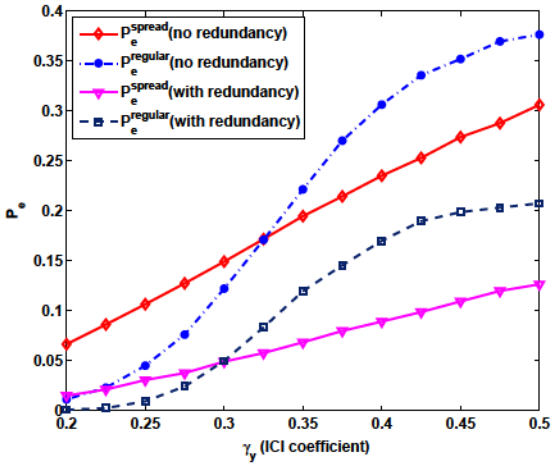

Fig. 9. Evolution of the probability of error as ICI increases for an MLC memory (i.e., b $\in \{-1.5, -0.5, 0.5, 1.5\}^4$), when k=1.1, $\sigma = 0.1$, $(\gamma_y, \gamma_{xy}) = (0.08, 0.006)$, and broken rate $p = 0.001$.

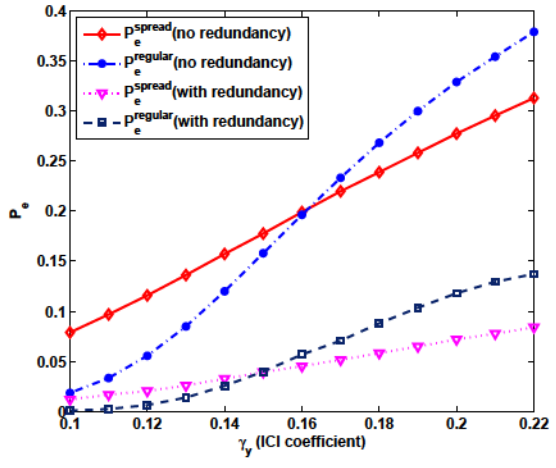

Fig. 10. Evolution of the probability of error as ICI increases for an TLC memory (i.e., b $\in \{\pm 3.5, \pm 2.5, \pm 1.5, \pm 0.5\}^4$), when k=1.1, $\sigma = 0.1$, $\gamma_y : \gamma_{xy} = 0.08 : 0.006$, and broken rate $p = 0.001$.

and Fig. 10 show that the spreading scheme provides lower BER as ICI increases.

Then, we study the case when the impulse noise dominates the BER in SLC. In order to focus on the impulse noise, both the Gaussian noise and ICI are assumed to be small. The results without parity are given by the first two curves in Fig.11 (no LDPC), showing that the BER with the traditional scheme is much larger than with the spreading scheme. Additionally, we analyzed the performance when the spreading modulation was combined with an LDPC encoding of the information. Specifically, we used the (64800,58320) LDPC code which is embedded in matlab R2015b. When the write noise and ICI noise is negligible, the problem of writing and reading information from a flash memory with the traditional modulation is equivalent to transmission over a binary erasure channel (BEC). The spreading scheme, on the other hand, spreads out the noise caused by broken cells, effectively transforming the BEC channel into a binary input AWGN channel, which is better for soft decoding. Fig.11 shows that the spreading

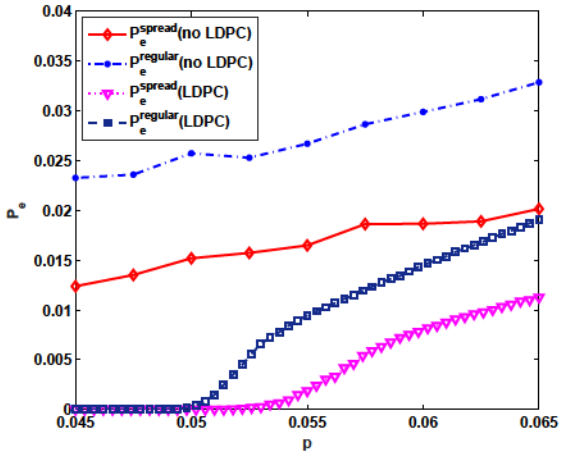

Fig. 11. Evolution of BER as cell broken rate p increases for an SLC memory (i.e., b $\in \{-0.5, 0.5\}^4$), when M=N=4, k=1.1, $\sigma = 0.1$, ICI coefficient $\gamma = 0.1$, the impulse noise dominates the BER.

scheme begins to fail at a larger p and has lower BER among the output bits.

Finally, we designed an experiment to evaluate how the voltage level to which a cell is programmed influences the damage that it suffers. Our preliminary results showed that when memories are programmed with highly structured data (e.g. 50% of the cells in a wordline written to the same level), they behave abnormally. Hence we tried to use random data in our experiment, while still imposing enough structure to observe different amount of damage in different cells. Four blocks in a 19nm MLC flash were repeatedly erased and programmed with random data, generated according to a different distribution for each cell. For example, some cells were programmed to the highest level 90% of the time, while others only reached that level on 10% of the PE cycles. After the wearing phase, each cell was programmed 100 more times with uniform random data and a dwell time of 1 hour at a temperature of 60C between writes. The information was read back before each new write, so as to obtain an average BER (proxy for damage) at the end of the 100 writes.

Once the cells had been worn (to a different number of PE cycles for each block) and the BER data had been collected, we performed a least squares fit to the model

$$BER_i = \alpha_1 P_i^{(1)} + \alpha_2 P_i^{(2)} + \alpha_3 P_i^{(3)} + \alpha_4 P_i^{(4)}, \quad i = 1, \ldots, 10^8, \quad (13)$$

where $P_i^{(j)}$ denotes the probability of programming the i-th cell to level j on each cycle of the wearing phase. The coefficients obtained for each of the four blocks, which should be proportional to the damage caused by programming each level, are shown in Fig. 12. Assuming that the voltage levels are equally spaced[3], a clear superlinear behavior can be observed. A quadratic model was adopted for simplicity, yielding the expression in Eq. (7) for the damage with each scheme (without loss of generality, we assumed $a = 1$).

[3]Unfortunately, we were not able to verify this fact from our memory manufacturer.

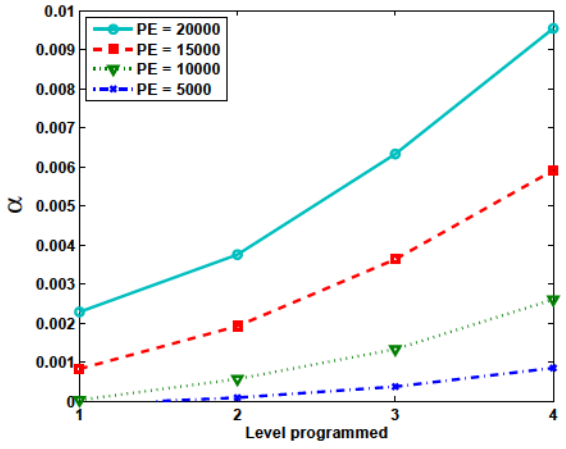

Fig. 12. Coeff cients modeling the damage suffered by a 19nm MLC cell when programmed to each voltage level, for different numbers of PE cycles.

VIII. Conclusion

This paper proposed a novel data representation scheme where Walsh codes are used to store the information in a NAND f ash memory, so that M symbols are spread out over N cells. We only discuss the case where $M = N$ so that the storage eff ciency is the same as that in the regular scheme in this paper. However, we could have better performance in the proposed scheme at the cost of more storage space when $N > M$.

By increasing the number of possible voltage levels in each cell, disregarding the fact that these levels could overlap, the proposed scheme can provide signif cant gains in terms of robustness towards inter-cell interference and impulse noise. Additionally, higher levels are used less frequently than in the regular scheme, reducing the damage suffered by the cells and thereby extending the endurance of the memory. We also show that this spreading technique can be used to overlap a hidden layer of information on top of the one stored in plain view. The paper provides analytical expressions for the SNR and BER of this spreading scheme under Gaussian noise, ICI, and impulse noise. Its performance is then studied through simulations.

In future work, we will study the best way to combine this spreading scheme with error correcting codes and will try to f nd ways to overcome the penalty in terms of read speed that the additional number of levels poses.

References

[1] R. Frickey, "Data integrity on 20nm SSDs," in Flash Memory Summit, 2012.

[2] P. Pavan, R. Bez, P. Olivo, and E. Zanoni, "Flash memory cells-an overview," Proc. IEEE, vol. 85, no. 8, pp. 1248–1271, Aug. 1997.

[3] B. Shin, C. Seol, J.-S. Chung, and J. J. Kong, "Error control coding and signal processing for f ash memories," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), 2012, pp. 409–412.

[4] G. Dong, S. Li, and T. Zhang, "Using data postcompensation and predistortion to tolerate cell-to-cell interference in MLC NAND f ash memory," IEEE Trans. Circuits Syst. I: Reg. Papers, vol. 57, no. 10, pp. 2718–2728, Oct. 2010.

[5] M. Qin, E. Yaakobi, and P. H. Siegel, "Constrained codes that mitigate inter-cell interference in read/write cycles for f ash memories," IEEE J. Sel. Areas Commun., vol. 32, no. 5, pp. 836–846, May 2014.

[6] J.-D. Lee, S.-H. Hur, and J.-D. Choi, "Effects of f oating-gate interference on NAND f ash memory cell operation," IEEE Electron Device Lett., vol. 23, no. 5, pp. 264–266, May 2002.

[7] Y. Kim, B. Vijaya Kumar, K. L. Cho, H. Son, J. Kim, J. J. Kong, and J. Lee, "Modulation coding for f ash memories," in Proc. of Int. Conf. on Computing, Networking and Communications (ICNC). IEEE, 2013, pp. 961–967.

[8] Y. Kim and B. V. Kumar, "Coding for memory with stuck-at defects," in IEEE Int. Conf. on Communications (ICC). IEEE, 2013, pp. 4347 – 4352.

[9] W. Wang, T. Xie, and D. Zhou, "Understanding the impact of threshold voltage on MLC f ash memory performance and reliability," in Proc. 28th ACM Int. Conf. on Supercomputing (ICS). ACM, 2014, pp. 201–210.

[10] H.-W. Tseng, L. Grupp, and S. Swanson, "Understanding the impact of power loss on f ash memory," in Proc. 48th Design Automation Conf. (DAC). ACM, 2011, pp. 35–40.

[11] A. Jagmohan, M. Franceschini, L. Lastras-Montano, J. Karidis et al., "Adaptive endurance coding for NAND f ash," in Proc. IEEE GLOBECOM Workshops. IEEE, Dec. 2010, pp. 1841–1845.

[12] B. Peleato and R. Agarwal, "Maximizing MLC NAND lifetime and reliability in the presence of write noise," in IEEE Int. Conf. on Communications (ICC). IEEE, 2012, pp. 3752–3756.

[13] B. Peleato, R. Agarwal, and J. Cioff , "Probabilistic graphical model for f ash memory programming," in IEEE Statistical Signal Processing Workshop (SSP). IEEE, 2012, pp. 788–791.

[14] T. Luo and B. Peleato, "Spread programming for NAND f ash," in IEEE Int. Conf. on Communications (ICC). IEEE, 2015, pp. 277–282.

[15] S. Moshavi, "Multi-user detection for DS-CDMA communications," IEEE Commun. Mag., vol. 34, no. 10, pp. 124–136, Oct. 1996.

[16] F. Adachi, M. Sawahashi, and H. Suda, "Wideband DS-CDMA for next-generation mobile communications systems," IEEE Commun. Mag., vol. 36, no. 9, pp. 56–69, Sep. 1998.

[17] A. Berman and Y. Birk, "Constrained f ash memory programming," in IEEE Int. Symp. on Information Theory (ISIT), 2011, pp. 2128–2132.

[18] A. Jiang, J. Bruck, and H. Li, "Constrained codes for phase-change memories," in IEEE Information Theory Workshop (ITW). IEEE, 2010, pp. 1–5.

[19] G. Dong, N. Xie, and T. Zhang, "On the use of soft-decision error-correction codes in NAND f ash memory," IEEE Trans. Circuits Syst. I: Reg. Papers, vol. 58, no. 2, pp. 429–439, Nov. 2011.

[20] K.-D. Suh, B.-H. Suh, Y.-H. Lim, J.-K. Kim, Y.-J. Choi, Y.-N. Koh, S.-S. Lee, S.-C. Kwon, B.-S. Choi, J.-S. Yum et al., "A 3.3 v 32 mb NAND f ash memory with incremental step pulse programming scheme," IEEE J. Solid-State Circuits, vol. 30, no. 11, pp. 1149–1156, Nov. 1995.

[21] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Threshold voltage distribution in MLC NAND f ash memory: Characterization, analysis, and modeling," in Proc. Design, Autom., Test in Eur. Conf. (DATE), Mar. 2013, pp. 1285–1290.

[22] S. Gerardin, M. Bagatin, A. Paccagnella, K. Grurmann, F. Gliem, T. Oldham, F. Irom, and D. Nguyen, "Radiation effects in f ash memories," IEEE Trans. Nucl. Sci., vol. 60, no. 3, pp. 1953–1969, 2013.

[23] B. Peleato, R. Agarwal, J. Cioff , M. Qin, and P. H. Siegel, "Towards minimizing read time for NAND f ash," in IEEE Global Communications Conf. (GLOBECOM). IEEE, 2012, pp. 3219–3224.

[24] J. Wang, T. Courtade, H. Shankar, and R. Wesel, "Soft information for LDPC decoding in f ash: Mutual information optimized quantization," in IEEE Global Communications Conf. (GLOBECOM). IEEE, 2011, pp. 5–9.

[25] Y. Maeda and H. Kaneko, "Error control coding for multilevel cell f ash memories using nonbinary low-density parity-check codes," in Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. IEEE, 2009, pp. 367–375.

[26] L. Wang, E. Sasoglu, B. Bandemer, and Y.-H. Kim, "A comparison of superposition coding schemes," in IEEE Int. Symp. on Information Theory (ISIT). IEEE, 2013, pp. 2970–2974.

[27] B. Sklar, Digital communications. Prentice Hall NJ, 2001, vol. 2.

[28] T. S. Rappaport et al., Wireless communications: principles and practice. prentice hall PTR New Jersey, 1996, vol. 2.

[29] D.-h. Lee and W. Sung, "Direct and indirect measurement of inter-cell capacitance in NAND f ash memory," in Proc. of IEEE Workshop on Signal Processing Systems (SiPS). IEEE, 2014, pp. 1–6.

Tianqiong Luo received her B.S. from Fudan University, Shanghai, China, in 2013. She is currently pursuing the Ph.D. degree of Electrical and Computer Engineering at Purdue University. Her research interests involve signal processing and coding for non-volatile storage.

Borja Peleato (S'12-M'13) is a Visiting Assistant Professor in the Electrical and Computer Engineering department at Purdue University. He received his B.S. degrees in telecommunications and mathematics from Universitat Politecnica de Catalunya, Barcelona, Spain, in 2007, and his M.S. and Ph.D. degrees in electrical engineering from Stanford University in 2009 and 2013, respectively. He was a visiting student at the Massachusetts Institute of Technology in 2006, and a Senior Flash Channel Architect with Proton Digital Systems in 2013. His research interests include signal processing and coding for non-volatile storage, convex optimization, and communications. Dr. Peleato received a "La Caixa" Graduate Fellowship in 2006.