Non-Binary LDPC code with Multiple Memory Reads for Multi-Level-Cell (MLC) Flash

Chaudhry Adnan Aslam*, Yong Liang Guan† and Kui Cai‡
*School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore.
E-mail: ad0001ry@e.ntu.edu.sg
†School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore.
E-mail: EYLGuan@ntu.edu.sg
‡Data Storage Institute (DSI), Singapore.
E-mail: CAI_Kui@dsi.a-star.edu.sg

***Abstract*—NAND flash memory has been dominantly used in consumer electronic products ranging from hand-held phones to personal computers. However, the stored data in NAND flash memory is subject to several impairments such as Random Telegraph Noise (RTN), Cell-to-Cell Interference (CCI) and Data Retention Effect over time. In this paper, we focus on the RTN effect over flash memory cells which becomes even more serious as the memory approaches its lifetime. When the flash cells withstand increasingly large number of Program/Erase (P/E) operation, multiple interface traps are generated at tunnel oxide layer which results into large fluctuations in cell threshold voltage. These voltage fluctuations, in turn, degrade the system error performance. To tackle with this problem, we propose a simple yet effective system-level decoding scheme in which the memory cells are read multiple times to obtain threshold voltage fluctuations caused by RTN. Since each memory read operation produces a new realization of threshold voltage, we combine the read signal with LDPC extrinsic information. The performance improvements of our scheme are validated by computer simulation which shows that the lifetime of flash memory can be extended by more than $10K$ P/E cycles while maintaining bit-error-rate at 10^{-6} using NB-LDPC code over $GF(4)$ with frame size $N = 2272$. This paper also presents the trade-off between performance improvement and extra memory sensing latency.**

I. Introduction

The NAND flash memory being the fastest growing product in consumer electronics, has significantly overwhelmed the semiconductor non-volatile storage market with its matured technological performance and rapid growth in the storage capacity. This tremendous growth and extensive usage of flash memory has become possible owing to the advancements in circuit designing and chip manufacturing processes which have significantly scaled down the physical size of the memory chip. Now we can produce memory products with CMOS technology on the scale of sub 20nm [1], [2]. Simultaneously, the flash memory capacity has also been increased mainly due to multi-bit/cell storage technique where a *2-bit*, *3-bit* and even *4-bit* per cell can be stored [3], [4]. However, as the technology scales down the chip area and multi-bit/cell technique increases the storage capacity, it has also brought several challenges to maintain data reliability [5]. Integrating memory cells closer to each other for area optimization, introduces circuit level impairment such as cell-to-cell interference [6] where a victim cell's threshold voltage is shifted in positive direction due to the proximity of neighboring interfering cells [7]. Moreover, the program and erase operation on flash memory cells over large number of times causes damage to the tunnel oxide which lies between transistor's channel and floating gate [8]. This results into generation of charged trap sites where electrons are trapped during cell programming, which directly effect on the threshold voltage stability and compromises the data integrity.

To ensure data reliability, conventional error correction schemes such as BCH codes [9], [10] are used with flash memory system. Yet, these coding techniques are not powerful enough to significantly improve the system performance and call for stronger error correction schemes. To further improve the data integrity, soft-decision error correction codes are also used with flash memory such as LDPC codes where they have shown to outperform BCH coding scheme [11]–[13].

The main objective of this work is to capture the variations associated with random telegraph noise (RTN) by reading the flash memory cells multiple times. For this purpose, we use NB-LDPC code with flash memory and utilize the extrinsic information from LDPC decoder for further processing when decoding is unsuccessful. This extrinsic information is combined along with subsequent memory read operations which essentially provide new read signal due to threshold voltage fluctuations caused by RTN. As presented in [14], the NB-LDPC have shown better performance compared to their binary counterparts. Though NB-LDPC codes have been previously proposed for MLC NAND flash [15], we have demonstrated further improvements in system bit-error-rate performance using multiple memory read operations with minimal increase in overall memory sensing/decoding latency.

The rest of this paper is organized as follows. Section II reviews the basics of NAND flash memory and briefly explains the memory read and write operations. In Section III, we present the probabilistic model of cell threshold voltage which is subject to RTN distortion. In Section IV, we deliberate in detail, on the proposed decoding algorithm discussing each step individually. Computer simulations are presented in Section V and trade-off between latency and performance improvement is given in Section VI. In Section VII, conclusion

978-616-361-823-8 © 2014 APSIPA

are drawn.

II. Basics of MLC NAND Flash

The NAND flash memory is organized into an array of rows (word-line) and columns (bit-line) where each array element represents a memory cell as shown in Fig. 1. The memory array is logically divided into blocks and each block is further subdivided into pages. As an example, 2Gb Micron NAND (SLC) *(1-bit/cell)* flash device [16] is organized into 2048 blocks, with 64 pages per block and each page consists of 2048-bytes. Memory cells within the same page share a common word-line whereas cells within a block share a common bit-line and an on-chip page buffer to hold the data being programmed and read. The basic memory cell is made up of floating-gate MOS transistor which comprises of control gate (CG),floating gate (FG) and transistor channel. The channel and floating gate (FG) are isolated by means of dielectric material known as "tunnel oxide". The control gate is electrically connected with the word-line to read and write (program) the data on the memory cell.

The NAND flash memory is programmed page-wise according to FowlerNordheim (FN) tunneling [17] where charged particles are injected though the tunnel oxide and are captured at the floating gate. The actual binary data being stored on the cell is represented by the amount of charged particles accumulated at the floating gate which in turns varies the threshold voltage (voltage required to turn-on the transistor) of that particular cell. For example, the $2-bit/cell$ flash memory can be programmed with $2^2 = 4$ distinct threshold voltage levels, each representing a particular binary value. To tightly limit the threshold voltage, the memory cells are often programmed through an iterative process knows as Incremental Step Pulse Program (ISPP) [18] which consists of series of programming and verify pulses as shown in Fig. 2. In this technique, each programming pulse with an incremental step size of ΔV_{pp}, is followed by a verify operation to check whether the desired threshold voltage V_p has been achieved. Once the required threshold voltage is attained, the programming operation is terminated.

The next section describes the memory read operation and presents the probabilistic model of RTN and threshold voltage distribution.

III. Random Telegraph Noise and Cell Threshold Voltage Distribution

The read operation is performed on flash memory page by successively increasing the voltage across the word-line and monitoring the current flow through the transistor channel until its turned-on. Once the sensed current reaches a predefined level (around $10nA$), the amount of voltage applied is recorded as threshold voltage V_{th} for that particular cell. All the mathematical formulation presented in this paper are based on $2-bit/cell$ MLC flash memory, however, it can be generalized for any other flash device configuration. According to [19], the threshold voltage of erased cell resembles to

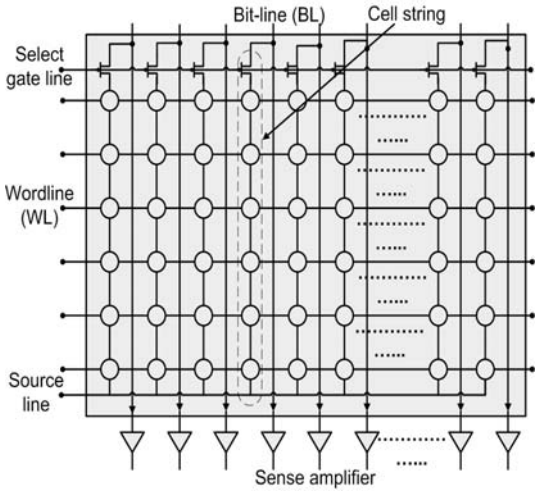

Fig. 1: Representation of NAND flash memory architecture: memory cells (circles) are organized into array of rows (word-line) and columns (bit-line).

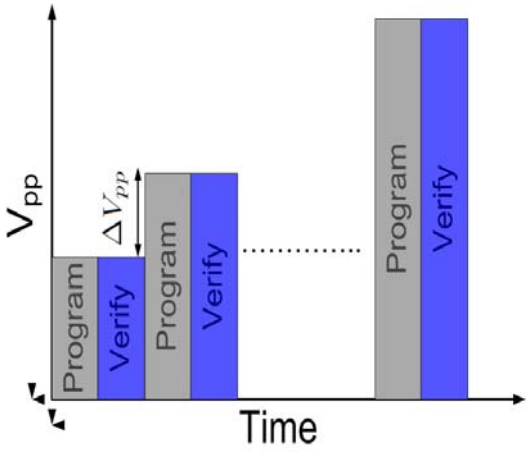

Fig. 2: Illustration of Incremental Step Pulse Programming (ISPP) scheme: each program pulse is followed by verify pulse. Voltage step size between two program pulses is ΔV_{pp}.

Gaussian distribution with mean V_e and variance σ_e as given by (1)

$$P_e\left(V_{th}\right) = \frac{1}{\sigma_e\sqrt{2\pi}} exp^{-\frac{\left(V_{th}-V_e\right)^2}{2{\sigma_e}^2}} \tag{1}$$

Without the presence of any noise, the threshold voltage of a programmed cell tends to follow uniform distribution. Let us denote the program voltage level by V_p and program pulse width by ΔV_{pp}, then we can model the threshold voltage distribution $P_u\left(V_{th}\right)$ after ideal programming as

$$P_u\left(V_{th}\right) = \begin{cases} \frac{1}{\Delta V_{pp}}, & \text{for } V_p \leqslant V_{th} \leqslant V_p + \Delta V_{pp} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

However, the ideal distribution is affected by random telegraph noise (RTN) which is generated due to electrons

trapping at the tunnel oxide. There are several studies on the behavior of RTN and its effect on cell threshold voltage. According to [20], the RTN can be modeled as symmetric exponential random process whose distribution is given by (3)

$$P_n(n) = \frac{1}{2\lambda} exp^{-\frac{\|n\|}{\lambda}} \tag{3}$$

Moreover, RTN distribution is non-stationary and varies with Program/Erase (P/E) cycling. If PE denotes the P/E cycling count, then by [21], λ scales with PE according to power law fashion and approximately proportional to $\lambda \propto PE^{\alpha}$ for some constant α . The RTN causes the instability of cell threshold voltage which can be represented mathematically as

$$V_{th} = u + n \tag{4}$$

where u is uniform random variable, n is RTN random sample and V_{th} is the final threshold voltage.

Denoting $P_{pr}(V_{th})$ as the distribution of programmed cells after incorporating RTN, then it can be written as

$$P_{pr}(V_{th}) = (P_u * P_n)(V_{th})$$

where $*$ is convolution integral. This integral can be written as

$$P_{pr}(V_{th}) = \begin{cases} \frac{c}{\Delta V_{pp}} \left[1 - exp^{-\frac{\Delta V_{pp}}{\lambda}}\right] exp^{-\frac{(V_p - V_{th})}{\lambda}}, \\ \qquad\qquad \text{for } V_{th} < V_p \\ \frac{c}{\Delta V_{pp}}, \quad \text{for } V_p \leqslant V_{th} \leqslant V_p + \Delta V_{pp} \\ \frac{c}{\Delta V_{pp}} \left[exp^{\frac{\Delta V_{pp}}{\lambda}} - 1\right] exp^{-\frac{(V_{th} - V_p)}{\lambda}}, \\ \qquad\qquad \text{for } V_{th} > V_p + \Delta V_{pp} \end{cases} \tag{5}$$

where c is normalizing factor given by

$$c = \frac{1}{1 + \frac{2\lambda}{\Delta V_{pp}} \left(1 - exp^{-\frac{\Delta V_{pp}}{\lambda}}\right)}$$

This is an approximation of convolution integral as given in [20]. We notice that the effect of RTN on threshold voltage distribution tends to introduce exponential tails below V_P and above $V_p + \Delta V_{pp}$ as shown in Fig. 3. In must be noted that, RTN also effect erased cells but they are still modeled with Gaussian distribution. In this figure, the mean and standard deviation of erased cell are set to 1.4 and 0.35 respectively. The mean of programmed states are set to 2.6, 3.2 and 3.8 respectively and programming step size ΔV_{PP} is set to 0.2. The RTN parameter λ is set to $0.00025\,(PE)^{0.5}$. It is evident from this figure that, the effect of P/E cycling widens the voltage distribution and results into overlap between adjacent distribution curves.

IV. Proposed Decoding Scheme

The proposed decoding algorithm is shown in Fig. 4. This decoding scheme is based on iterative detection and decoding of read signal using NB-LDPC code where each LDPC symbol is given by $GF(q)(q > 2)$. Let K be the number of input

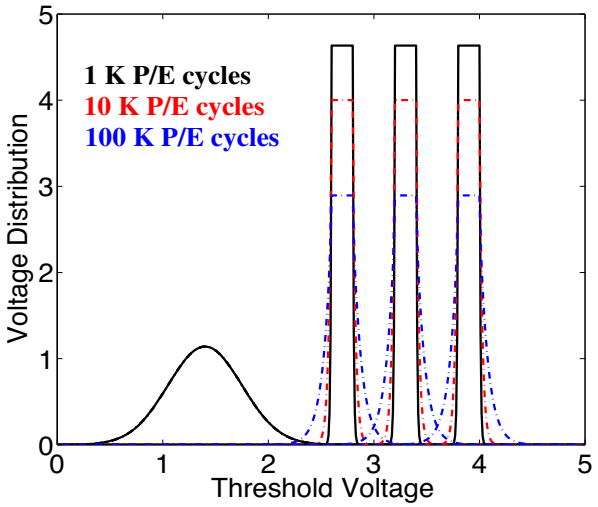

Fig. 3: Simulation results of threshold voltage distribution at different P/E cycles. With the increase in P/E cycles, adjacent distribution curves tend to overlap.

symbols to LDPC encoder and N be the number of LDPC coded symbols (LDPC frame size). Then, each LPDC frame contains mN bits where m is the number of bits per LDPC symbol($m = \log_2[q]$). In this scheme, each LDPC coded symbol is stored over memory cells such that bits of one encoded LDPC symbol span over m adjacent memory cells as shown in Fig. 5. It must be mentioned that we can also store one NB-LDPC symbol on single flash cell, provided the compatibility between flash technology (bits/cell) and NB-LDPC symbol size, however with proposed arrangement, we can easily use any NB-LDPC symbol size (e.g. $m = 2, 3, 4, ...$) to store over MLC flash memory. We should notice that, since no interference is assumed from neighboring cells, there will be no correlation effect by storing LDPC symbol across adjacent cells. Fig. 4 shows different steps involved in our proposed scheme and their detailed description is given in the following sections.

A. Step 1 (Read Memory)

We assume that one memory page (word-line) contains M cells which can store bits equal to LDPC frame length mN, then sensing memory array page-wise will provide threshold voltage of cells ${V_{th}}^{k(r)}$ where $k \in \{1, 2, ..., M\}$ and r represents the number of times memory is sensed. For example, if we read memory page for two-times, then we denote threshold voltage realization associated with first and second memory read as ${V_{th}}^{k(1)}$ and ${V_{th}}^{k(2)}$ respectively.

B. Step 2 (Compute Cell State Probability)

Using the threshold voltage obtained in the previous step, we compute the cell state probability ${P_{i,j}}^{k(r)}$ where $k \in \{1, 2, ..., M\}$ and (i, j) represents the MSB and LSB bit position within a memory cell. Considering $2 - bit$ per cell flash memory and assuming 00 being the erased state and

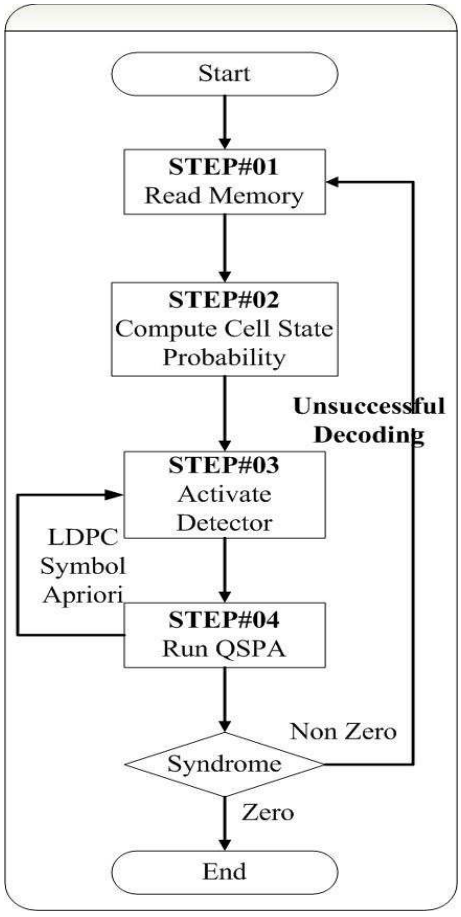

Fig. 4: Illustration of steps involved with iterative decoding and detection using multiple memory reads.

$01, 10$ and 11 being the programmed states respectively, we shall compute ${P_{00}}^{k^{(r)}}, {P_{01}}^{k^{(r)}}, {P_{10}}^{k^{(r)}}$ and ${P_{11}}^{k^{(r)}}$ by using equations (1) and (5) respectively.

C. Step 3 (Activate Detector)

The detector unit only combines the extrinsic information of LDPC decoder with the cell state probability. The detection process should not be assumed similar to turbo detection/equalization. In the detection process, we perform two operations; converting LDPC symbol into MLC cell probability and then converting MLC cell to LDPC symbol probability as explained in next section.

1) LDPC Symbol to MLC Cell Probability - Intrinsic Information to Detector: In this process, the detector unit uses intrinsic information coming from LDPC decoder ${I^l}_S{}^{(r)}$ where $S \in GF(q)$ and $l \in \{1, 2, ..., N\}$. For our convenience, we write S into vector representation by replacing S with its equivalent binary notation as $\overline{S} = (s_0, s_1, ..., s_{m-1})$ and $s_i \in \{0, 1\}$. Hence by using ${I^l}_{\overline{S}}{}^{(r)}$, the detection unit produces MLC cell probability ${g_{x,y}}^{k^{(r)}}$ given as

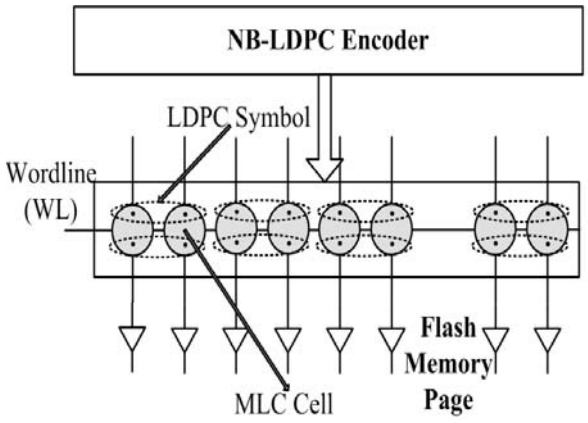

Fig. 5: Representation of NB-LDPC ($q = 4$) symbol storage over MLC NAND flash page ($2 - bit$ per cell): NB-LDPC symbols (dotted ellipse) span over adjacent memory cells (circles) where period "." represents one-bit storage.

$$ {g_{x,y}}^{k^{(r)}} = \left(\sum_{\overline{S}:s_i=x} {I_{\overline{S}}}^{l^{(r)}} \right) \left(\sum_{\overline{S}:s_i=y} {I_{\overline{S}}}^{l+1^{(r)}} \right) \quad (6) $$

where $k \in \{1, 2, ..., M\}, (x, y) \in \{00, 01, 10, 11\}, i = [(k-1) \mod m], l = \lceil \frac{k}{m} \rceil$.

Example 1: For NB-LDPC code defined over $GF(4)$, ${g_{x,y}}^{k^{(r)}}$ for first memory read and $k = 1, 2$ can be written using equation (6) as

$$
\begin{aligned}
{g_{00}}^{1^{(1)}} &= \left({I_{00}}^{1^{(1)}} + {I_{01}}^{1^{(1)}} \right) \left({I_{00}}^{2^{(1)}} + {I_{01}}^{2^{(1)}} \right) \\
{g_{01}}^{1^{(1)}} &= \left({I_{00}}^{1^{(1)}} + {I_{01}}^{1^{(1)}} \right) \left({I_{10}}^{2^{(1)}} + {I_{11}}^{2^{(1)}} \right) \\
{g_{10}}^{1^{(1)}} &= \left({I_{10}}^{1^{(1)}} + {I_{11}}^{1^{(1)}} \right) \left({I_{00}}^{2^{(1)}} + {I_{01}}^{2^{(1)}} \right) \\
{g_{11}}^{1^{(1)}} &= \left({I_{10}}^{1^{(1)}} + {I_{11}}^{1^{(1)}} \right) \left({I_{10}}^{2^{(1)}} + {I_{11}}^{2^{(1)}} \right) \\
{g_{00}}^{2^{(1)}} &= \left({I_{00}}^{1^{(1)}} + {I_{10}}^{1^{(1)}} \right) \left({I_{00}}^{2^{(1)}} + {I_{10}}^{2^{(1)}} \right) \\
{g_{01}}^{2^{(1)}} &= \left({I_{00}}^{1^{(1)}} + {I_{10}}^{1^{(1)}} \right) \left({I_{01}}^{2^{(1)}} + {I_{11}}^{2^{(1)}} \right) \\
{g_{10}}^{2^{(1)}} &= \left({I_{01}}^{1^{(1)}} + {I_{11}}^{1^{(1)}} \right) \left({I_{00}}^{2^{(1)}} + {I_{10}}^{2^{(1)}} \right) \\
{g_{11}}^{2^{(1)}} &= \left({I_{01}}^{1^{(1)}} + {I_{11}}^{1^{(1)}} \right) \left({I_{01}}^{2^{(1)}} + {I_{11}}^{2^{(1)}} \right)
\end{aligned}
$$

2) MLC Cell to LDPC Symbol Probability - Extrinsic Information from Detector: After computing the MLC cell probability ${g_{x,y}}^{k^{(r)}}$ in the previous step, we will combine this information with cell state probability computed in *Step 2* and produce aposteriori information ${E_{\overline{S}}}^{l^{(r)}}$ from detector as

$$
{E^l}_{\overline{S}=(s_0,s_1,...,s_{m-1})}{}^{(r)} = \begin{cases} \prod_{p=0}^{m-1} \left({g^k}_{s_p,0}{}^{(r)} . {P^k}_{s_p,0}{}^{(r)} + {g^k}_{s_p,1}{}^{(r)} {P^k}_{s_p,1}{}^{(r)} \right), \text{for } l \text{ is odd} \\ \prod_{p=0}^{m-1} \left({g^k}_{0,s_p}{}^{(r)} {P^k}_{0,s_p}{}^{(r)} + {g^k}_{1,s_p}{}^{(r)} {P^k}_{1,s_p}{}^{(r)} \right), \text{for } l \text{ is even} \end{cases} \quad (7)
$$

where $l \in \{1, 2, ..., N\}$, $s_i \in \{0, 1\}$, $k = \left(\lceil \frac{l}{2} \rceil - 1\right) + (p + 1)$.

Example 2: For NB-LDPC code defined over $GF(4)$, ${E^l}_{\overline{S}=(s_0,s_1)}^{(r)}$ for first memory read and $l = 1, 2$ can be written using equation (7) as

$$
\begin{aligned}
{E_{00}}^{1^{(1)}} &= \left({g_{00}}^{1^{(1)}} {P_{00}}^{1^{(1)}} + {g_{01}}^{1^{(1)}} {P_{01}}^{1^{(1)}}\right) \\
&\quad \left({g_{00}}^{2^{(1)}} {P_{00}}^{2^{(1)}} + {g_{01}}^{2^{(1)}} {P_{01}}^{2^{(1)}}\right) \\
{E_{01}}^{1^{(1)}} &= \left({g_{00}}^{1^{(1)}} {P_{00}}^{1^{(1)}} + {g_{01}}^{1^{(1)}} {P_{01}}^{1^{(1)}}\right) \\
&\quad \left({g_{10}}^{2^{(1)}} {P_{10}}^{2^{(1)}} + {g_{11}}^{2^{(1)}} {P_{11}}^{2^{(1)}}\right) \\
{E_{10}}^{1^{(1)}} &= \left({g_{10}}^{1^{(1)}} {P_{10}}^{1^{(1)}} + {g_{11}}^{1^{(1)}} {P_{11}}^{1^{(1)}}\right) \\
&\quad \left({g_{00}}^{2^{(1)}} {P_{00}}^{2^{(1)}} + {g_{01}}^{2^{(1)}} {P_{01}}^{2^{(1)}}\right) \\
{E_{11}}^{1^{(1)}} &= \left({g_{10}}^{1^{(1)}} {P_{10}}^{1^{(1)}} + {g_{11}}^{1^{(1)}} {P_{11}}^{1^{(1)}}\right) \\
&\quad \left({g_{10}}^{2^{(1)}} {P_{10}}^{2^{(1)}} + {g_{11}}^{2^{(1)}} {P_{11}}^{2^{(1)}}\right) \\
{E_{00}}^{2^{(1)}} &= \left({g_{00}}^{1^{(1)}} {P_{00}}^{1^{(1)}} + {g_{10}}^{1^{(1)}} {P_{10}}^{1^{(1)}}\right) \\
&\quad \left({g_{00}}^{2^{(1)}} {P_{00}}^{2^{(1)}} + {g_{10}}^{2^{(1)}} {P_{10}}^{2^{(1)}}\right) \\
{E_{01}}^{2^{(1)}} &= \left({g_{00}}^{1^{(1)}} {P_{00}}^{1^{(1)}} + {g_{10}}^{1^{(1)}} {P_{10}}^{1^{(1)}}\right) \\
&\quad \left({g_{01}}^{2^{(1)}} {P_{01}}^{2^{(1)}} + {g_{11}}^{2^{(1)}} {P_{11}}^{2^{(1)}}\right) \\
{E_{10}}^{2^{(1)}} &= \left({g_{01}}^{1^{(1)}} {P_{01}}^{1^{(1)}} + {g_{11}}^{1^{(1)}} {P_{11}}^{1^{(1)}}\right) \\
&\quad \left({g_{00}}^{2^{(1)}} {P_{00}}^{2^{(1)}} + {g_{10}}^{2^{(1)}} {P_{10}}^{2^{(1)}}\right) \\
{E_{11}}^{2^{(1)}} &= \left({g_{01}}^{1^{(1)}} {P_{01}}^{1^{(1)}} + {g_{11}}^{1^{(1)}} {P_{11}}^{1^{(1)}}\right) \\
&\quad \left({g_{01}}^{2^{(1)}} {P_{01}}^{2^{(1)}} + {g_{11}}^{2^{(1)}} {P_{11}}^{2^{(1)}}\right)
\end{aligned}
$$

D. Step 4 (Run QSPA)

In this paper, we use *Fast Fourier Transform* (FFT) based *q-ary Sum Product Algorithm* (QSPA) [22] for decoding NB-LDPC code, however, we can use any other reduced-complexity algorithm for this purpose. Once the detector produces extrinsic information ${E^l}_{\overline{S}}$, it is fed to FFT-QSPA LDPC decoder. In the decoding process, the decoder computes *syndrome* vector after every iteration as shown in Fig. 4. The decoding is terminated if either *syndrome* vector is zero or decoder has reached maximum allowed iterations $Iter_{max}$. The output from the decoder unit is referred as extrinsic information which is further utilized and sent back to detector unit in case the *syndrome* check is not satisfied.

E. Step 5 (Unsuccessful Decoding - Read Memory)

Based on the *syndrome* check, we will read the memory page multiple times. The next read will yield a new realization of threshold voltage which when combined with detection and decoding process will improve the system error performance. In [23], it has been shown that reading a memory page multiple times, results into threshold voltage fluctuation as shown in Fig. 6. Traps generated at Tunnel Oxide by excessive P/E cycling can easily gain/loose electrons which cause variations in the read voltage around the mean value. Thus higher P/E operations will result into more threshold voltage instability.

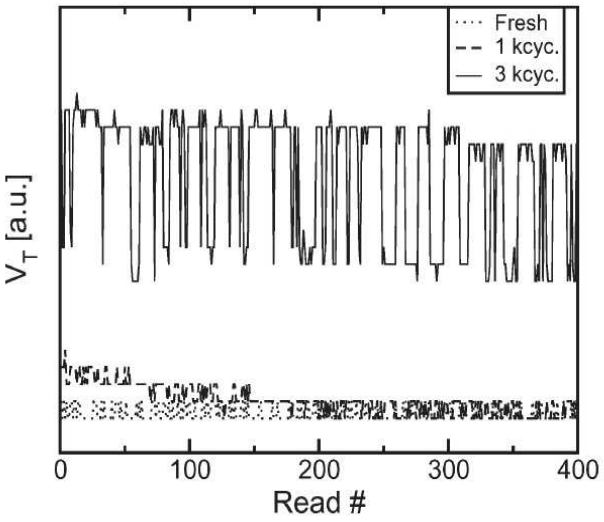

Fig. 6: Illustration of threshold voltage instability between multiple memory read operations as presented in [23].

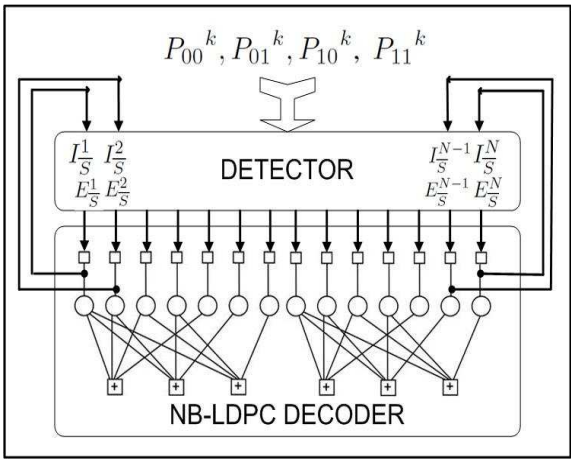

Fig. 7: Visualization of iterative detection and decoding mechanism: extrinsic information ${I^l}_S^{(r)}$ produced by NB-LDPC decoder is used with cell state probability ${P_{i,j}}^{k^{(r)}}$ at detector to produce updated extrinsic information ${E_{\overline{S}}}^{l^{(r)}}$.

The author in [24], [25] has presented mathematical description for the fluctuation of sensed voltage between multiple read operations. According to the author, multiple read operations of same flash cell may provide different realization of threshold voltage due to the possibility of traps to change their state in subsequent memory reads. Considering independent RTN effect between two read operations, we compute the cell state probability according to *Step 2* using new realization of threshold voltage ${V_{th}}^{k^{(2)}}$. The detector unit receives this new cell state probability along with extrinsic information from LDPC decoder and produces updated aposteriori information which is further sent to QSPA decoder as shown in Fig. 7.

V. Computer Simulations

We have performed monte-carlo simulations to demonstrate the effectiveness of proposed decoding scheme using the following flash parameters:

$V_{00} = 1.4, V_{01} = 2.6, V_{10} = 3.8, V_{11} = 3.2$ (mean threshold voltage of erased and programmed states respectively). $\sigma_e = 0.35$ (standard deviation of erased state). $\Delta V_{PP} = 0.2$ (ISPP step voltage). $\lambda = 0.00025\,(PE)^{0.5}$ (λ is RTN distribution parameter and *PE* is program/erase cycle count). NB-LDPC over $GF\,(4)$ with $N = 2272$ (LDPC frame length), $Iter_{max} = 15$ and $R = 0.9014$ (LDPC code rate).

Fig. 8. shows the BER performance of simulated NAND flash memory using *unquantized* threshold voltage for 1, 2 and 3 memory read operation. We can observe that, with more memory read operations, the endurance of flash memory can be improved. We can interpret this as with multiple read operations, the flash memory can handle more P/E cycles while maintaining the same BER performance. Since the proposed decoding technique caters RTN problem, it yields more coding gain at high RTN region as also evident from the simulation results. In Fig. 9, we have plotted the frame error rate (FER) corresponding to 1, 2 and 3 read operations with unquantized information.

From the practical implementation perspective, the acquisition of *unquantized information* (infinite-level quantization) is not possible since it involves extensively large number of quantization levels which incur unbearable memory sensing latency. Therefore, we have also plotted BER and FER performance for 15-level and 30-level quantization as shown in Fig. 10 and Fig. 11 respectively. The quantization boundaries are obtained by uniformly dividing the overlapping region between two adjacent cell distributions as explained in [26]. In Fig. 12 and Fig. 13, we have shown the average memory read operations and average LDPC decoder iterations consumed per LDPC frame for our proposed iterative detection and decoding scheme. Since we only read memory multiple times when previous decoding round has failed, we have very few subsequent memory reads. As the flash memory undergoes more P/E cycles, it will also increase both average read operations (sensing latency) and LDPC decoder iterations.

VI. Trade-off between Performance and Latency

In this section we present the trade-off between the performance gain and extra latency corresponding to multiple memory read operations. This discussion is based on fixed BER level at 10^{-6}. In Fig. 14, we can observe that with multiple read operations, the flash memory can endure more P/E cycles while maintaining the same error performance. In Fig. 15, we have plotted the average increase in memory lifetime. This lifetime increase is defined as more number of P/E operations the device can withstand(P/E gain) for the same error performance. As an example, with second memory read operation, the *15-level* quantization scheme can extend the operational lifetime of NAND flash upto $11K$ P/E cycles.

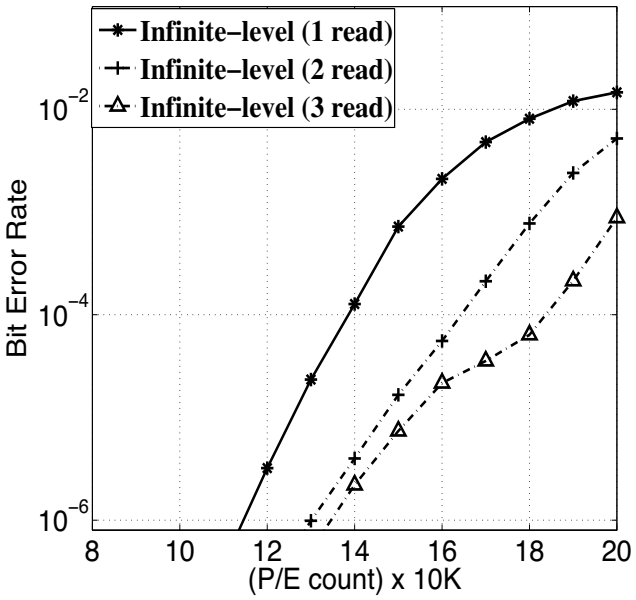

Fig. 8: Simulation result of bit-error-rate performance using 1, 2 and 3 read operations with **infinite-level** quantization.

Similarly, third memory read can further improve the memory endurance by $3K$ P/E cycles. We can observe that, the second memory read provides more significant gain as compared with subsequent read operations. In order not to offset the extra memory sensing latency, we have shown the increased read latency as we perform more read operations in Fig. 16. This figure presents average number of memory reads performed to obtain the given BER. For example, the blue bars show the average latency incurred when maximum of two read operations are set only because the first read operation results into unsuccessful LDPC decoding. We notice that when *15-level* quantization is used, the 2-read operation introduces around 0.5% of additional latency compared with 1-read case. If one read operation requires $25nsec$ of time, then our scheme would add extra memory sensing latency of $0.125nsec$ which is quite acceptable in comparison with improvement in system performance.

VII. Conclusion

We have presented iterative detection and decoding scheme for MLC NAND flash using multiple memory read operations. The presented algorithm caters RTN related problem especially when the flash device undergoes large number of P/E cycles. The proposed technique is applied to simulated NAND flash channel using NB-LDPC code over $GF\,(4)$. The iterative detection and decoding is implemented by using extrinsic information from NB-LDPC decoder along with new threshold voltage realization obtained from multiple read operations. The improvements are evident from simulation results which shows that the lifespan of flash memory can be extended by more than $10K$ P/E cycles with an additional memory read operation for $BER = 10^{-6}$. We have also presented the trade-off between

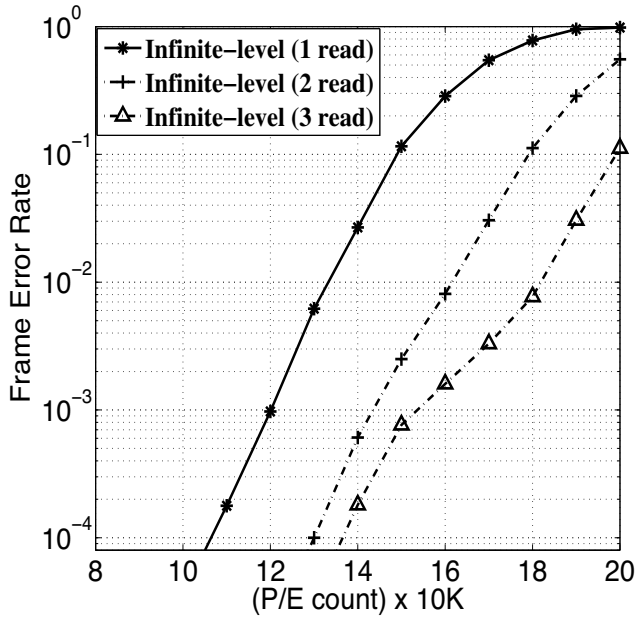

Fig. 9: Simulation result of frame-error-rate performance using 1, 2 and 3 read operations with **infinite-level** quantization.

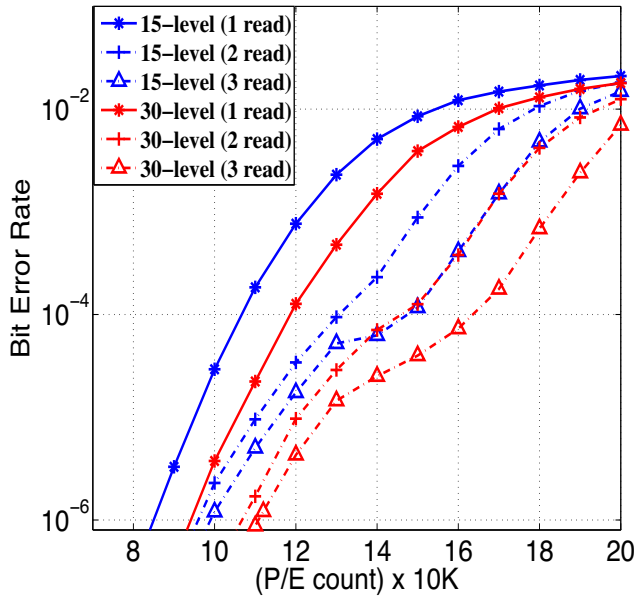

Fig. 10: Simulation result of bit-error-rate performance using 1, 2 and 3 read operations with **15-level** and **30-level** quantization.

performance and memory sensing latency. With a gain of $11K$ P/E cycles at $BER = 10^{-6}$, the extra memory sensing latency is only around 0.5%. We expect that the proposed system-level decoding technique can be used to handle large P/E cycles and can prolong the operational lifetime of MLC NAND flash memory.

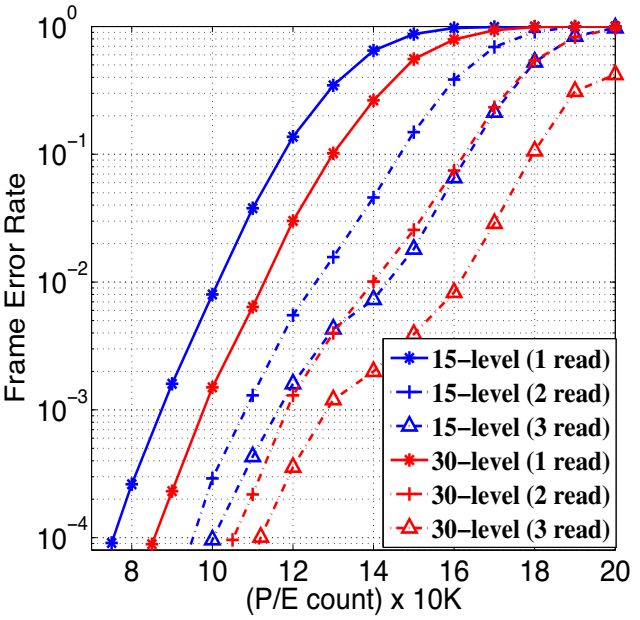

Fig. 11: Simulation result of frame-error-rate performance using 1, 2 and 3 read operations with **15-level** and **30-level** quantization.

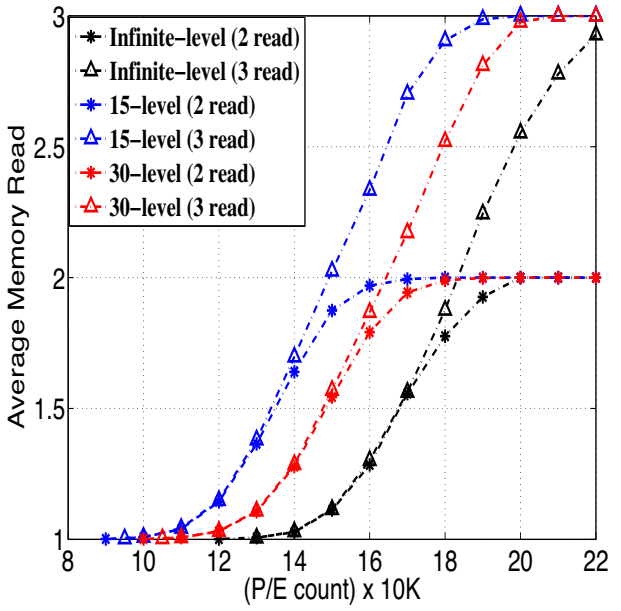

Fig. 12: Simulation result of average memory read operations performed while using **infinite-level**, **15-level** and **30-level** quantization.

REFERENCES

[1] Y. Koh, "Nand flash scaling beyond 20nm," in *Memory Workshop, 2009. IMW '09. IEEE International*, 2009, pp. 1–3.

[2] Y. Lee, B. Park, D. Yun, Y. Jeong, P. H. Kim, J. Y. Park, H. C. Yang, M. K. Cho, K.-O. Ahn, and Y. Koh, "The challenges and limitations on triple level cell geometry and process beyond 20 nm nand flash

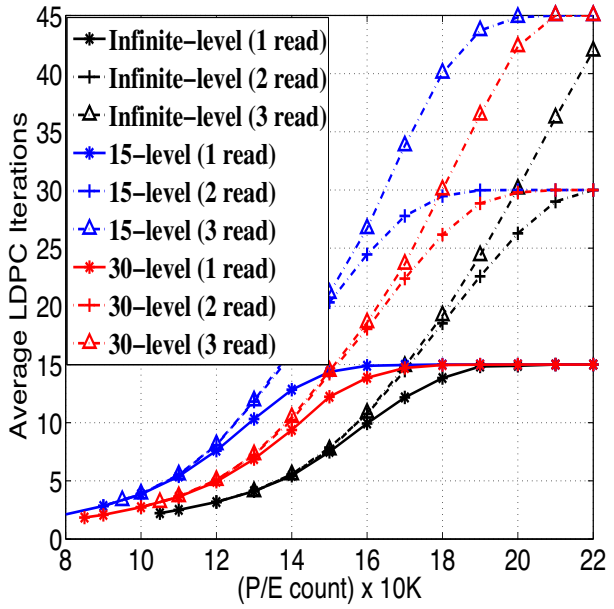

Fig. 13: Comparison of average LDPC decoder iterations between 1, 2 and 3 read operations using **infinite-level**, **15-level** and **30-level** quantization.

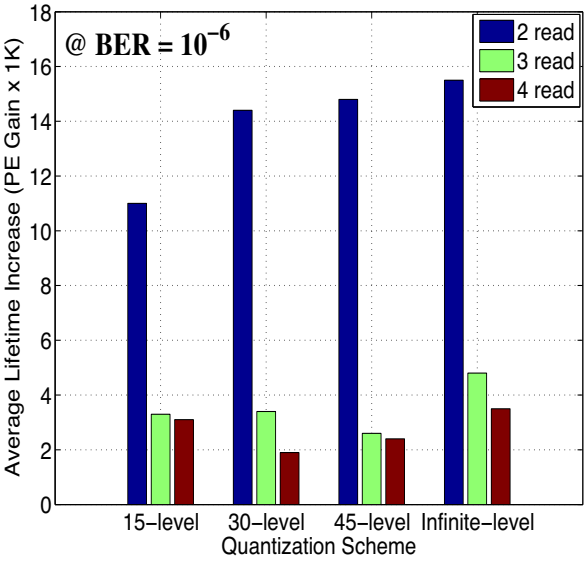

Fig. 15: Illustration of P/E gain (enduring more P/E cycles) for different quantization schemes at $BER = 10^{-6}$: lifetime increase is significant from 1-read to 2-read.

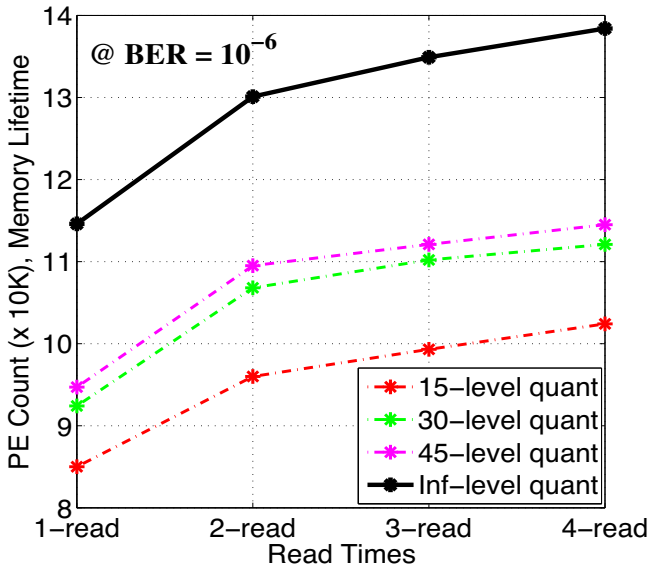

Fig. 14: Performance of different quantization schemes at $BER = 10^{-6}$: Increasing the read operations, improves the flash endurance. Black solid line is an upper bound for quantization levels.

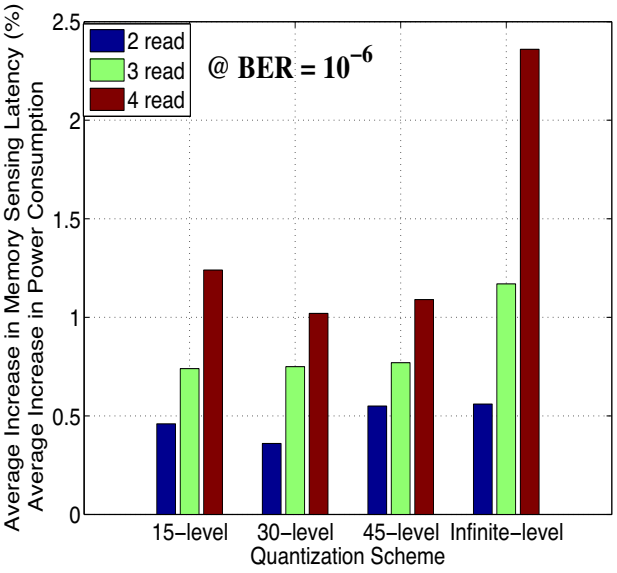

Fig. 16: Simulation of average increase in memory sensing latency due to multiple read operations as a function of read times and quantization levels at $BER = 10^{-6}$.

technology," in *Memory Workshop (IMW), 2010 IEEE International*, 2010, pp. 1–2.

[3] Y. Li, S. Lee, Y. Fong, F. Pan, T.-C. Kuo, J. Park, T. Samaddar, H. T. Nguyen, M. Mui, K. Htoo, T. Kamei, M. Higashitani, E. Yero, G. Kwon, P. Kliza, J. Wan, T. Kaneko, H. Maejima, H. Shiga, M. Hamada, N. Fujita, K. Kanebako, E. Tam, A. Koh, I. Lu, C.-H. Kuo, T. Pham, J. Huynh, Q. Nguyen, H. Chibvongodze, M. Watanabe, K. Oowada, G. Shah, B. Woo, R. Gao, J. Chan, J. Lan, P. Hong, L. Peng, D. Das, D. Ghosh, V. Kalluru, S. Kulkarni, R.-A. Cernea, S. Huynh, D. Pantelakis, C.-M. Wang, and K. Quader, "A 16 gb 3-bit per cell (x3) nand flash memory on 56 nm technology with 8 mb/s write rate," *Solid-State Circuits, IEEE Journal of*, vol. 44, no. 1, pp. 195–207, 2009.

[4] C. Trinh, N. Shibata, T. Nakano, M. Ogawa, J. Sato, Y. Takeyama, K. Isobe, B. Le, F. Moogat, N. Mokhlesi, K. Kozakai, P. Hong, T. Kamei, K. Iwasa, J. Nakai, T. Shimizu, M. Honma, S. Sakai, T. Kawaai, S. Hoshi, J. Yuh, C. Hsu, T. Tseng, J. Li, J. Hu, M. Liu, S. Khalid, J. Chen, M. Watanabe, H. Lin, J. Yang, K. McKay, K. Nguyen,

T. Pham, Y. Matsuda, K. Nakamura, K. Kanebako, S. Yoshikawa, W. Igarashi, A. Inoue, T. Takahashi, Y. Komatsu, C. Suzuki, K. Kanazawa, M. Higashitani, S. Lee, T. Murai, K. Nguyen, J. Lan, S. Huynh, M. Murin, M. Shlick, M. Lasser, R. Cernea, M. Mofidi, K. Schuegraf, and K. Quader, "A 5.6mb/s 64gb 4b/cell nand flash memory in 43nm cmos," in *Solid-State Circuits Conference - Digest of Technical Papers, 2009. ISSCC 2009. IEEE International*, 2009, pp. 246–247,247a.

[5] S. Aritome, R. Shirota, G. Hemink, T. Endoh, and F. Masuoka, "Reliability issues of flash memory cells," *Proceedings of the IEEE*, vol. 81, no. 5, pp. 776–788, 1993.

[6] K. Kim, "Future memory technology: challenges and opportunities," in *VLSI Technology, Systems and Applications, 2008. VLSI-TSA 2008. International Symposium on*, 2008, pp. 5–9.

[7] G. Dong, S. Li, and T. Zhang, "Using data postcompensation and predistortion to tolerate cell-to-cell interference in mlc nand flash memory," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 57, no. 10, pp. 2718–2728, 2010.

[8] N. Mielke, H. Belgal, I. Kalastirsky, P. Kalavade, A. Kurtz, Q. Meng, N. Righos, and J. Wu, "Flash eeprom threshold instabilities due to charge trapping during program/erase cycling," *Device and Materials Reliability, IEEE Transactions on*, vol. 4, no. 3, pp. 335–344, 2004.

[9] W. Liu, J. Rho, and W. Sung, "Low-power high-throughput bch error correction vlsi design for multi-level cell nand flash memories," in *Signal Processing Systems Design and Implementation, 2006. SIPS '06. IEEE Workshop on*, 2006, pp. 303–308.

[10] R. Micheloni, R. Ravasio, A. Marelli, E. Alice, V. Altieri, A. Bovino, L. Crippa, E. Di Martino, L. D'Onofrio, A. Gambardella, E. Grillea, G. Guerra, D. Kim, C. Missiroli, I. Motta, A. Prisco, G. Ragone, M. Romano, M. Sangalli, P. Sauro, M. Scotti, and S. Won, "A 4gb 2b/cell nand flash memory with embedded 5b bch ecc for 36mb/s system read throughput," in *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*, 2006, pp. 497–506.

[11] J. Kim, D. hwan Lee, and W. Sung, "Performance of rate 0.96 (68254, 65536) eg-ldpc code for nand flash memory error correction," in *Communications (ICC), 2012 IEEE International Conference on*, 2012, pp. 7029–7033.

[12] J. Wang, T. Courtade, H. Shankar, and R. Wesel, "Soft information for ldpc decoding in flash: Mutual-information optimized quantization," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, 2011, pp. 1–6.

[13] F. Zhang, H. Pfister, and A. Jiang, "Ldpc codes for rank modulation in flash memories," in *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, 2010, pp. 859–863.

[14] M. Davey and D. MacKay, "Low-density parity check codes over gf(q)," *Communications Letters, IEEE*, vol. 2, no. 6, pp. 165–167, 1998.

[15] Y. Maeda and H. Kaneko, "Error control coding for multilevel cell flash memories using nonbinary low-density parity-check codes," in *Defect and Fault Tolerance in VLSI Systems, 2009. DFT '09. 24th IEEE International Symposium on*, 2009, pp. 367–375.

[16] (2006) Nand flash 101: An introduction to nand flash and how to design it in to your next product. [Online]. Available: http://download.micron.com/pdf/technotes/nand/tn2919.pdf

[17] S. Aritome, S. Satoh, T. Maruyama, H. Watanabe, S. Shuto, G. Hemink, R. Shirota, S. Watanabe, and F. Masuoka, "A 0.67 /spl mu/m/sup 2/ self-aligned shallow trench isolation cell (sa-sti cell) for 3 v-only 256 mbit nand eeproms," in *Electron Devices Meeting, 1994. IEDM '94. Technical Digest., International*, 1994, pp. 61–64.

[18] 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, J.-H. Choi, J.-R. Kim, and H.-K. Lim, "A 3.3 v 32 mb nand flash memory with incremental step pulse programming scheme," *Solid-State Circuits, IEEE Journal of*, vol. 30, no. 11, pp. 1149–1156, 1995.

[19] K. Takeuchi, T. Tanaka, and H. Nakamura, "A double-level-vth select gate array architecture for multilevel nand flash memories," *Solid-State Circuits, IEEE Journal of*, vol. 31, no. 4, pp. 602–609, 1996.

[20] C. Monzio Compagnoni, M. Ghidotti, A. Lacaita, A. Spinelli, and A. Visconti, "Random telegraph noise effect on the programmed threshold-voltage distribution of flash memories," *Electron Device Letters, IEEE*, vol. 30, no. 9, pp. 984–986, 2009.

[21] G. Dong, Y. Pan, N. Xie, C. Varanasi, and T. Zhang, "Estimating information-theoretical nand flash memory storage capacity and its implication to memory system design space exploration," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 20, no. 9, pp. 1705–1714, 2012.

[22] L. Barnault and D. Declercq, "Fast decoding algorithm for ldpc over gf(2q)," in *Information Theory Workshop, 2003. Proceedings. 2003 IEEE*, 2003, pp. 70–73.

[23] C. Monzio Compagnoni, A. Spinelli, S. Beltrami, M. Bonanomi, and A. Visconti, "Cycling effect on the random telegraph noise instabilities of nor and nand flash arrays," *Electron Device Letters, IEEE*, vol. 29, no. 8, pp. 941–943, 2008.

[24] C. Monzio Compagnoni, R. Gusmeroli, A. Spinelli, A. Lacaita, M. Bonanomi, and A. Visconti, "Statistical model for random telegraph noise in flash memories," *Electron Devices, IEEE Transactions on*, vol. 55, no. 1, pp. 388–395, 2008.

[25] C. Monzio Compagnoni, R. Gusmeroli, A. Spinelli, and A. Visconti, "Rtn vt instability from the stationary trap-filling condition: An analytical spectroscopic investigation," *Electron Devices, IEEE Transactions on*, vol. 55, no. 2, pp. 655–661, 2008.

[26] G. Dong, N. Xie, and T. Zhang, "On the use of soft-decision error-correction codes in nand flash memory," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 58, no. 2, pp. 429–439, 2011.