Codes for High Performance Write and Read Processes in Multi-Level NVMs

Evyatar Hemo and **Yuval Cassuto**
Department of Electrical Engineering, Technion – Israel Institute of Technology
evyatar@tx.technion.ac.il, ycassuto@ee.technion.ac.il

***Abstract*— Multi-level memory cells are used in non-volatile memories in order to increase the storage density. Using multi-level cells, however, imposes higher read and write latencies limiting high speed applications. In this work we study the tradeoff between storage density and write/read performance using codes. The contributions are codes that give high-performance write and read processes with minimal reduction in storage density. We describe the codes, give an analytical treatment of their information rate and speed, and compare them with more basic access schemes and upper bounds.**

I. Introduction

Multi-level non-volatile memories (NVMs) hold great promise to scale the storage capacity of mass-storage devices of the present and future. The main impediment of multi-level memories stems from the steep degradation of the read/write speed as the number of levels grows. This is because more levels mean longer processes to accurately set the cell levels and to measure them with precision upon read.

In order to achieve a viable storage device, the correct trade-off between storage density and read/write performance needs to be found. Toward this objective, we here propose and analyze coding schemes that efficiently balance between the rate of information storage and the time required to write and read information onto and from the media. With these codes, a storage device with a given number of physical levels q will be able to flexibly choose its encoding schemes to reach desired write and read speeds, with minimal resulting compromise of the information rate.

In this paper we propose two codes that optimize write and read times, while keeping the expended redundancy to a minimum. In order to do so, the codes constrain the stored levels within a memory block in a way that is most beneficial to the write or read processes. Previous schemes like [1],[2] considered access-optimizing mappings for multi-level cells, but were more limited as they did not use redundancy. In Section II we introduce the first code which is designed for high-performance write. The second code for high-performance read is presented in III. In order to analyze the codes, we also compare them to simpler natural alternatives and to upper bounds on read performance that are also included in our contributions. Our results on read-boosting codes build on our previous work [3] on algorithms for efficient multi-level threshold read (without coding). The best results of the codes are achieved when the number of memory cells that can be written or read in parallel is not much higher than the number of memory levels. This is, for example, the typical case for NOR flash memory and for phase-change memory (PCM) [4], when commonly 2 to 16 cells are programmed in parallel. Beyond the applicability to practical storage, the results lay a basis for studying this important fundamental trade-off between density and speed in non-volatile storage devices.

II. Codes for high-performance write

In the first part of the paper we look into ways to encode information when we wish to speed the *write* process up from the worst-case write time offered without coding.

A. Multi-level parallel write model

In the commonly used *breadth-first write* process (see description in [5]), a number of cells are programmed in parallel to the same target level. For NAND technology, the memory level is reflected by the amount of electrical charge in each cell. Alternatively, for PCM cells the levels are represented by the electrical resistance values of the cells. The formal model we define next abstracts the specific physical mechanism used for writing, and thus can apply to any technology that uses breadth-first write.

Let the state of the storage cell be represented as a discrete **cell level** c, taken from the integer set $\{0,\ldots,q-1\}$. Let $\mathcal{N}$ be a memory block of size n, and let $\boldsymbol{v}=(v_1,\ldots,v_n)$ be a vector of target levels we want to write into $\mathcal{N}$.

Definition 1. *Given a vector of cell levels $\boldsymbol{c}=(c_1,\ldots,c_n)$, with $c_i\in\{0,\ldots,q-1\}$, define the* **incidence set** *as the set $\mathcal{I}(\boldsymbol{c})=\{s\in\{1,\ldots,q-1\}|\exists i,\ c_i=s\}$. The elements of $\mathcal{I}(\boldsymbol{c})$ are assumed to be ordered in increasing order.*

The breadth-first write scheme of the values $\boldsymbol{v}$ into the memory block $\mathcal{N}$ of size n is given by:

Algorithm 1.

$\boldsymbol{l}=\mathcal{I}(\boldsymbol{v})$
$\forall i, c_i=l_1$ // *program all cells to min($\boldsymbol{v}$)*
for k=2 to $|\boldsymbol{l}|$
    For all $c_i\in\mathcal{N}$ *with* $c_i<v_i$: $c_i=l_k$
    // *raising the level by* l_k-l_{k-1}
end

which means that first all cells are programmed (in parallel) to $min(\boldsymbol{v})$. Then the next higher level in $\boldsymbol{v}$ is found, and programmed into all cells that have target levels at least that value. The process continues until this is done with $max(\boldsymbol{v})$. Therefore, it is clear that the number of programming steps (which to first order is equivalent to write latency) is proportional to $|\mathcal{I}(\boldsymbol{v})|$ and not to n.

B. The CNL code

From analyzing the write model it is obvious that reducing the number of levels used to store information in a memory block will also reduce the time needed to write this information (number of write steps in the "for" loop at Algorithm 1). However, reducing the number of occupied memory levels will also reduce the information storage rate of the memory block, creating an interesting tradeoff between storage efficiency and write-speed. Therefore, write optimization is achieved by constraining the number of different levels occupied in each vector of target levels $\boldsymbol{v}$. The basic most naive write scheme supporting this principle is to use only up to ω **fixed** levels. For example, in a memory cell that includes $\{0,\ldots,q-1\}$ possible levels, the basic scheme will be limited to the values $\{0,\ldots,\omega-1\}$ only. However, by coding it is possible to improve the density/write speed tradeoff.

Definition 2. *The* CNL(ω) *code (Constrained Number of Levels) is a code in which every legal codeword* $\boldsymbol{u}$ *must fulfill* $|\mathcal{I}(\boldsymbol{u})| \leqslant \omega$, *where* $0 < \omega \leqslant q$. *In other words, the level subset is not fixed a priori, but any legal CNL codeword must have up to only* ω *different occupied levels within the same block.*

It is clear that according to the model, in a worst case scenario in which all of the possible levels are used, the write speed of the CNL code is the same as that of the fixed scheme for the same ω.

Example 1. *Suppose* $n = 8, q = 8$ *and we set* $\omega = q/2 = 4$. *A legitimate codeword in the fixed model is for example* $(3,0,2,1,0,3,1,1)$ *because it contains only levels in the set* $\{0,1,2,3\}$. *In the CNL code, in addition to the vector above we may use the codeword* $(7,0,4,1,0,0,7,1)$, *because it similarly includes* 4 *levels* $\{0,1,4,7\}$.

C. Information rate analysis

In order to asses the performance of a code the information rate must be analyzed.

Definition 3. *The* Information Rate $\mathcal{R}$ *of* n*-cell,* q*-level memory array is defined as*

$$\mathcal{R} = log_q(M)/n,$$

where M *is the number of legal combinations within the memory array. For example, when all possible combinations are legal* $M = q^n$, *hence* $\mathcal{R} = 1$.

Therefore, the information rate for the fixed write scheme is given by

$$\mathcal{R}_{Fixed}(n,q,\omega) = log_q(\omega^n)/n = log_q(\omega).$$

The CNL code, however, has a better information rate, which by definition is the optimal information rate that can be achieved by writing only ω different levels. To calculate this improved information rate we use the following proposition.

Proposition 1. *The number of combinations in which* ***exactly*** $k \leqslant n$ *levels are occupied in an* n*-cell* q*-level array, is given by*

$$C(n,q,k) = k! \cdot S(n,k) \cdot \binom{q}{k}, \tag{1}$$

where $S(n,k)$ *is the Stirling number of the second kind [6]. For* $k > n$ *the number of combinations is clearly* 0 *by definition.*

Proof: The number of combinations to choose only k levels out of the existing q levels is given by $\binom{q}{k}$. By multiplying it by the number of surjections from n-cell set to q-level set we obtain the desired combinations count. The number of surjections from an n-set to a k-set equals $k! \cdot S(n,k)$ [7]. ■

Theorem 2. *The* Information Rate *of the CNL code is given by*

$$\mathcal{R}_{CNL}(n,q,\omega) = log_q\left[\sum_{k=1}^{min(n,\omega)} k! \cdot S(n,k) \cdot \binom{q}{k}\right]/n. \tag{2}$$

Proof: By Definition 2 and Definition 3 it is clear that the information rate of the code is given by

$$\mathcal{R}_{CNL}(n,q,\omega) = log_q\left[\sum_{k=1}^{min(n,\omega)} C(n,q,k)\right]/n.$$

Using the expression for counting combinations from Eq. (1) gives us claim (2). ■

When analyzing and comparing the information rates of the fixed model and the CNL code as a function of $log_2(q)$ it is possible to notice that the rate of the CNL code is higher than the rate of the fixed model at all q values for a chosen n. In addition, the rate of the CNL code increases as n becomes smaller and as q becomes larger, while the rate of the fixed model also increases as q becomes larger but does not depend on n.

D. Worst case write speed analysis

In order to assess the overall performance of the code, the write time, measured as the number of write steps, is now analyzed.

Definition 4. *The time* T *needed to write codeword* $\boldsymbol{u}$ *to* n *cells memory block is measured by the number of write steps given by* $|\mathcal{I}(\boldsymbol{u})|$, *(Conversion to write time given in [seconds] is achieved by multiplying* T *by the individual-write time in [seconds]). The subscript of* T *indicates the specific code used in the write process.*

In the worst-case write scenario exactly ω levels are occupied. Note that given a value of ω both the CNL code and the fixed scheme have the same worst-case write time. To compare between the CNL code and the fixed scheme, we will require identical information rates for given n and q, obtaining an equivalent number of occupied levels $\hat{\omega}$ for the fixed scheme. Equating $\mathcal{R}_{CNL}(n,q,\omega) = \mathcal{R}_{Fix}(n,q,\hat{\omega})$ yields

$$\hat{\omega} = \left[\sum_{k=1}^{min(n,\omega)} k! \cdot S(n,k) \cdot \binom{q}{k}\right]^{\frac{1}{n}}.$$

In other words, a CNL codeword with given n, q and ω has the same information rate as n cells with $\hat{\omega}$ fixed memory levels (out of q). After equating the information rates it is now possible to compare the worst-case write times. Note that the value of $\hat{\omega}$ may be a non-integer. In that case, an equivalent non-integer ω can be achieved by using the space-sharing concept in which a $(\lceil\omega\rceil - \omega)$ fraction of the total cell blocks uses $\lfloor\omega\rfloor$ levels while the rest use $\lceil\omega\rceil$ levels.

The worst-case *write-time ratio* (WTR) between the fixed scheme and the CNL code is given by

$$WTR = \frac{T_{Fixed}}{T_{CNL}} = \frac{\hat{\omega}}{\omega}.$$

Since legal codewords of Fixed are always legal codewords of CNL (but not vice-versa), we have $\hat{\omega} > \omega$ and the write-time ratio for the worst-case scenario is always higher than 1.

E. Average write speed analysis

We now examine the performance of the *average* write time when the codewords are distributed uniformly.

Theorem 3. *The average time for writing a CNL codeword of length n to a q-level memory block where only up to ω levels are occupied is given by*

$$T_{CNL}(n,q,\omega) = \frac{\sum_{k=1}^{min(n,\omega)} k \cdot k! \cdot S(n,k) \cdot \binom{q}{k}}{\sum_{k=1}^{min(n,\omega)} k! \cdot S(n,k) \cdot \binom{q}{k}}. \quad (3)$$

Proof: When using the CNL code, the probability that in a codeword only k levels are occupied, $1 \leqslant k \leqslant \omega$, is given by

$$Prob(\boldsymbol{u} \in CNL, |\mathcal{I}(\boldsymbol{u})| = k) = \frac{C(n,q,k)}{\sum_{k'=1}^{min(n,\omega)} C(n,q,k')}.$$

Calculating the expectation of the number of occupied levels using Eq. (1) gives us claim (3). ■

It is clear from Theorem 3 that for the fixed write scheme the average write time is

$$T_{Fixed}(n,q,\omega) = \frac{1}{\omega^n} \sum_{k=1}^{min(n,\omega)} k \cdot k! \cdot S(n,k) \cdot \binom{\omega}{k}.$$

It is important to notice that by the way we define it, T_{CNL} and T_{Fixed} are actually the average number of the different levels occupied in the CNL code and fixed write methods, respectively.

To compare between the CNL code and the fixed scheme in terms of average write time, we will again require identical information rates for given n and q. Fig. 1 presents the average write-time ratio between the fixed model and the CNL code as a function of n and q, with $\omega = q/2$. The entire plot reflects parameters where the CNL code outperforms the fixed scheme (time ratios greater than 1). In addition, it is observed that it is possible to achieve up to $20-30\%$ speed improvement by using the CNL code.

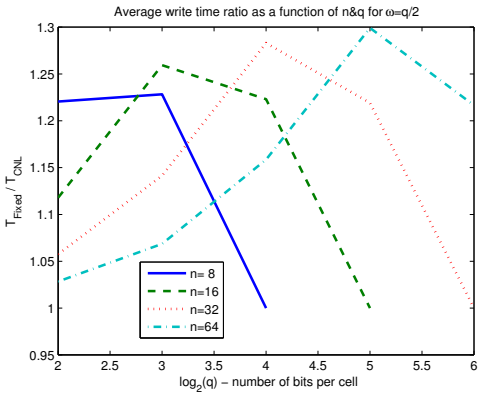

Figure 1. Average write time ratio for different values of n and q with $\omega = q/2$: n=8 (solid), n=16 (dashed), n=32 (dotted), n=64 (semi-dotted)

For lower values of ω, for example $\omega = q/4$, the superiority of the average write time of the CNL code was improved from $20-30\%$ to $40-50\%$ (for maximal improvement values).

III. Codes for high-performance read

In a similar way, by using a code it is possible to achieve high-performance read. We use the parallel threshold model (described in [3]) which means the cells are read by applying a sequence of threshold measurements, each applied to the n cells in parallel and returns n binary values of whether the cell levels are above or below the threshold.

Definition 5. *A* **threshold** *τ is an integer from the set $\{1,\ldots,q-1\}$. Given a threshold τ, a cell is said to be* **active** *with respect to τ if its cell level satisfies $c \geqslant \tau$. In the complementary case of $c < \tau$, the cell is said to be* **inactive** *with respect to τ.*

Definition 6. *A* **measurement** *is an operator acting on a set of cells $\mathcal{S}$ by applying a threshold of τ, and obtaining $|\mathcal{S}|$ binary values reflecting the activity of each cell in $\mathcal{S}$ with respect to τ. We denote the measurement as a vector $M_\tau(\mathcal{S}) = (m_1,\ldots,m_{|\mathcal{S}|})$, where $m_i \in \{0,1\}$. $m_i = 1$ represents an active cell with respect to τ, and $m_i = 0$ represents an inactive cell.*

The number of measurements M, needed to fully read a memory block, represents the effort (or time) required to complete the reading process. Therefore, in order to achieve high-performance read, M must be minimized while maintaining the information rate as high as possible.

A. The FNCL code

We now introduce a code that has higher read-performance than basic more natural read schemes. We first give some formal definitions.

Definition 7. *For a q-level memory cell, let us define the ω-***range** *$r_i(\omega)$ as the ω consecutive levels starting from $i-1$. Therefore, $\{r_i(\omega) = [i-1, i-2+\omega] \mid 1 \leqslant i \leqslant q-\omega+1\}$.*

From Definition 7 it is clear that there are $q-\omega+1$ different ω-ranges in a q-level memory cell.

Definition 8. *For a q-level n-cell memory block, we define $Z_i(\omega)$ to be all possible combinations of length-n words where all n levels are taken to be in the range $r_i(\omega)$.*

Definition 9. *The FNCL(ω) code (Fixed Number of Consecutive Levels) is a code in which for every q-level, n-cell memory block $\boldsymbol{c}$ there is an i such that $\boldsymbol{c} \in Z_i(\omega)$.*

In the sequel we omit the argument ω from Z_i, r_i when its value is clear from the context.

B. The information rate

Theorem 4. *The* Information Rate *of the FNCL(ω) code is given by*

$$\mathcal{R}_{FNCL}(n,q,\omega) = log_q\left[(q-\omega)\cdot(\omega^n - (\omega-1)^n) + \omega^n\right]/n. \quad (4)$$

Proof: For the proof, we make the following definition.

Definition 10. *Define the sets S_i as $S_1 = Z_1$ and $S_i = Z_i \setminus (Z_i \cap Z_{i-1})$, $i > 1$.*

To simplify the counting of codewords, we replace the sets $\{Z_i\}_{i=1}^{q-\omega+1}$ used to define the code with the *disjoint* sets $\{S_i\}_{i=1}^{q-\omega+1}$. There is no codeword that belongs to more than one set S_i, because belonging to S_i, $i > 1$ implies that the maximal level $i-2+\omega$ is occupied in at least one cell, and this level is not part of any set S_j with $j < i$.
Every set S_i, $i > 1$ includes the ω^n combinations of Z_i,

excluding the combinations of $Z_i \cap Z_{i-1}$. By definition the joint combinations of Z_i and Z_{i-1} can only include the lower $\omega - 1$ levels of Z_i. Therefore, the number of combinations for each S_i, $i > 1$ is given by $\omega^n - (\omega - 1)^n$. There are $(q - \omega)$ such sets in the code plus the extra set S_1 which includes ω^n combinations. Applying the total number of combinations to Definition 3 yields (4). ■

An interesting special case is $\omega = q/2$, where the redundancy is less than one bit per q-ary cell, while the performance savings can be significant. For $\omega = q/2$ we get

$$\mathcal{R}_{FNCL}\left(n, q, \frac{q}{2}\right) = log_q\left[\frac{q}{2} \cdot \left(\left(\frac{q}{2}\right)^n - \left(\frac{q}{2} - 1\right)^n\right) + \left(\frac{q}{2}\right)^n\right]/n.$$

As in Section II, we compare the FNCL code with a natural more basic read scheme in which for every memory block a *fixed* interval of ω consecutive levels (e.g. $\{0, \ldots, \omega - 1\}$) are used. We refer to this read scheme as "FixCons", and calculate its information rate to be

$$\mathcal{R}_{FixCons}(n, q, \omega) = log_q\left[\omega^n\right]/n.$$

For the case $\omega = q/2$ we get

$$\mathcal{R}_{FixCons}\left(n, q, \frac{q}{2}\right) = log_q\left[\left(\frac{q}{2}\right)^n\right]/n = 1 - log_q 2.$$

Due to the fact that $(q/2)^n - (q/2 - 1)^n > 0$ for all relevant n and q, it is easy to see that $\mathcal{R}_{FNCL}(n, q, q/2) > \mathcal{R}_{FixCons}(n, q, q/2)$, which means that the rate of the FNCL code exceeds that of FixCons for all n and q.

C. Number of measurements in the read process

We now examine the benefits in read performance offered by the coding schemes defined in the previous sub-section. We assume that upon reading a block the reader does *not* know if the levels are taken according to the code or unconstrained from the entire set of q levels. This assumption is necessary if one wants to mix fast and regular reads without storing side information for the reader.

We start with describing the reader for FixCons. In the FixCons scheme (assumed to use the lowest consecutive levels as its fixed interval), the reader will start a sequential threshold-measurements scan from level 1 and terminate when it reaches the first level that is higher than all stored levels, which is ω in the worst case. Thus the worst-case number of measurements needed to fully read a memory block in the FixCons scheme is $M_{FixCons} = \omega$, for any $\omega \leqslant q - 1$. We now show a read algorithm for FNCL.

Algorithm 2.

ReadFNCL($\mathcal{N}$)
- $\tau_0 = q/2$
- $\boldsymbol{m_0} = M_{\tau_0}(\mathcal{N})$
- $\tau = \tau_0$, $\boldsymbol{m} = \boldsymbol{m_0}$
- *while* $'1' \in \boldsymbol{m}$
  - $\tau = \tau + 1$ // *go upwards*
  - $\boldsymbol{m} = M_\tau(\mathcal{N})$
- *end*
- $\tau = \tau_0$, $\boldsymbol{m} = \boldsymbol{m_0}$
- *while* $'0' \in \boldsymbol{m}$
  - $\tau = \tau - 1$ // *go downwards*
  - $\boldsymbol{m} = M_\tau(\mathcal{N})$
- *end*

Theorem 5. *The number of threshold measurements needed to fully read an FNCL codeword is $M_{FNCL} = \omega + 1$ when $q/2 \leqslant \omega \leqslant q - 2$.*

Proof: When reading a memory block using Algorithm 2 with $\omega \geqslant q/2$, there is at most one measurement that falls outside the set of levels allowed for codewords in S_i. To see this, observe that the center level $\tau_0 = q/2$ is allowed in all sets S_i, except possibly S_1. Starting at $\tau_0 = q/2$, Algorithm 2 will go upward until reaching τ such that all levels are $< \tau$. Then it will go downward until reaching τ such that all levels are $\geqslant \tau$. The total number of visited levels is thus at most $\omega + 1$. When $i = 1$, the level $\tau_0 = q/2$ may be above the interval of levels in S_1, but the total number of measurements is still at most $\omega + 1$ (in fact at most ω in this case). ■

D. Comparing FNCL vs. FixCons

To compare between the FNCL code and the FixCons scheme, we will require identical information rates for given n and q, obtaining an equivalent number of allowed levels $\hat{\omega}$ for the FixCons scheme. Equating $\mathcal{R}_{FNCL}(n, q, \omega) = \mathcal{R}_{FixCons}(n, q, \hat{\omega})$ for $\omega = q/2$ yields

$$\hat{\omega} = \left[\frac{q}{2} \cdot \left(\left(\frac{q}{2}\right)^n - \left(\frac{q}{2} - 1\right)^n\right) + \left(\frac{q}{2}\right)^n\right]^{1/n}. \tag{5}$$

Definition 11. *For given n and q and $\omega = q/2$, we define the **equal-information measurement ratio (EMR)** as the ratio between the number of measurements of FixCons and that of FNCL. That is*

$$EMR = \frac{M_{FixCons}}{M_{FNCL}} = \frac{\hat{\omega}}{\omega + 1} = \frac{\hat{\omega}}{q/2 + 1},$$

where $\hat{\omega}$ is given in (5).

The maximal value of the EMR is attained when $n = 1$, yielding $\hat{\omega} = q$ and $EMR = q/(q/2 + 1)$, which approaches 2 for large q. At the other extreme of $n >> q$, we get EMR close to 1 (no advantage for FNCL). In intermediate cases of constant $n > 1$

Theorem 6. *When $n > 1$ and $q >> n$ the asymptotic expression for the EMR is given by*

$$EMR = \sqrt[n]{n+1} \cdot \left(1 - \frac{n-1}{n+1} \cdot \frac{1}{q}\right). \tag{6}$$

Proof: Due to the fact that $\hat{\omega} > q/2$ and $q >> n > 1$ the EMR can be approximated by

$$EMR \cong \frac{\hat{\omega}}{q/2} = \left[1 + \frac{q}{2} \cdot \left(1 - \left(1 - \frac{2}{q}\right)^n\right)\right]^{1/n}.$$

Approximating $(1 - 2/q)^n$ by $[1 - 2n/q + 2n(n-1)/q^2 + \ldots]$ gives

$$EMR \cong \left[1 + n - \frac{n(n-1)}{q}\right]^{1/n} = \sqrt[n]{n+1} \cdot \left[1 - \frac{n-1}{n+1} \cdot \frac{n}{q}\right]^{1/n},$$

considering $n/q << 1$, further approximation yields (6). ■

For very large n the asymptotic expression of the EMR reaches 1. However, for lower values of n, the EMR is larger than 1. Therefore, the number of measurements needed for full read in the FixCons scheme is 43% ($n = 4, q = 64$) to 10% ($n = 16, q = 32$) higher than in the FNCL code read process (achieving the same information rate). Lower number of

measurements enables faster and more efficient read process. As can be deduced from (6), for higher values of n the superiority of the FNCL code diminishes.

E. Upper bound on information rate given read effort

After presenting constructive methods to lower the measurement count using redundancy, we now seek a converse result on limits for such measurement reduction given the expended redundancy. Specifically, we look for upper bounds on the best information rate achievable when up to M threshold measurements are applied. The upper bounds enable us to asses the performance of the FixCons and FNCL read schemes. We start with a simple upper bound, followed by a tighter, more involved one.

Theorem 7. *The information rate in a q-level memory block read by up to M threshold measurements is bounded from above by*

$$\mathcal{R}(q,M) \leqslant min\left(M \cdot log_q 2, 1\right). \tag{7}$$

We refer to the upper bound presented in Theorem 7 as the *Naive Upper Bound*, since it does not take into account the structure of the read process obtaining the nM bits. We now derive a significantly tighter upper bound.

Theorem 8. *The information rate in a n-cell, q-level memory block read by up to M threshold measurements is bounded from above by*

$$\mathcal{R}(n,q,M) \leqslant \log_q \left[\sum_{m=1}^{M} \sum_{(k,L,j)\in\psi(m)} k! \cdot S(n,k) \cdot D_j(q,k,L) \right] / n, \tag{8}$$

where $\psi(m) = \{(k,L,j) \mid k+L-j=m\}$ and $1 \leqslant k \leqslant n$, $1 \leqslant L \leqslant k$, $j \in \{0,1,2\}$ and the expressions for $D_j(q,l,L)$ are [9]:

$$D_0(q,l,L) = \binom{l-1}{L-1}\binom{q-l-1}{L},$$

$$D_1(q,l,L) = 2\binom{l-1}{L-1}\binom{q-l-1}{L-1},$$

$$D_2(q,l,L) = \binom{l-1}{L-1}\binom{q-l-1}{L-2} + \Delta[l=q; L=1].$$

The function $\Delta[]$ is 1 when all of its arguments are true and 0 otherwise.

Proof: In order to calculate the upper bound we need to find a necessary condition for being able to read an information vector in up to M measurements. For a given codeword $\boldsymbol{c}$, the minimal number of measurements needed to fully read $\boldsymbol{c}$ is given by [3] $\hat{M}(\boldsymbol{c}) = |\mathcal{I}(\boldsymbol{c}) \cup \mathcal{I}^*(\boldsymbol{c})|$, where $\mathcal{I}^*(\boldsymbol{c})$ is the **shifted incidence set** defined as $\mathcal{I}^*(\boldsymbol{c}) = \{s \in \{1,\ldots,q-1\} | \exists i,\ c_i + 1 = s\}$. Therefore, in order to find all the combinations which can be read by at most M measurements we need to find all codewords $\boldsymbol{c}$ that fulfill $\hat{M}(\boldsymbol{c}) \leqslant M$. In order to do so, we recall from [3] that there are $k! \cdot S(n,k) \cdot D_j(q,k,L)$ level combinations with k used levels, falling into L consecutive runs, with j levels in the two extremes 0 and $q-1$. Each such combination requires at least $k+L-j$ measurements. Hence the number of combinations that can be measured by M measurements are not more than those that satisfy $k+L-j \leqslant M$. This is exactly the expression inside the log in (8). ■

The information rates of the upper bounds, FNCL code and the FixCons scheme are presented in Fig. 2. The FNCL

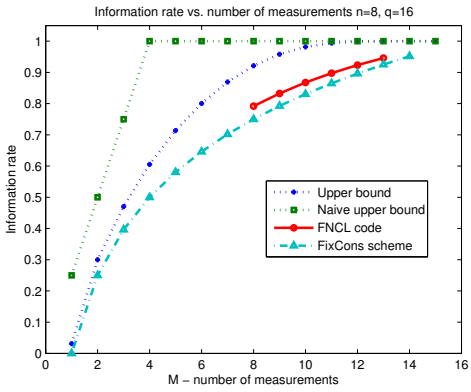

Figure 2. Information rates as a function of number of measurements: Upper bound (pluses), Naive upper bound (squares), FNCL code (circles), FixCons scheme (triangles)

is presented for the higher more interesting and practical information rates. The FNCL outperforms the FixCons scheme for all relevant values of M, however, the superiority of the FNCL (relative to the FixCons scheme) decreases with M. In addition, for middle-range values of M, the upper bound is much tighter than the naive upper bound.

IV. Conclusion

Using coding within multi-level NVMs can significantly improve read and write performance keeping the expended redundancy to a minimum. The CNL code for optimal write process was presented. In addition, the FNCL code for high-performance read was introduced and compared to a fixed scheme and upper bounds. It is an interesting question whether there exist better codes (and matching read algorithms) than the FNCL code presented here.

V. Acknowledgment

This work was supported in part by the Israel Ministry of Science and Technology, and by an Intel ICRI-CI grant.

References

[1] K. Takeuchi, et-al. "A multipage cell architecture for high-speed programming multilevel NAND flash memories," IEEE Journal of Solid-State Circuits, vol. 33.8, pp.1228-1238, Aug 1998.

[2] A. Berman, Y. Birk, Minimal Maximum-Level Programming: Faster Memory Access via Multi-Level Cell Sharing, IEEE Global Communications Conference (GLOBECOM), 2013.

[3] E. Hemo, and Y. Cassuto, "Adaptive threshold read algorithms in multi-level non-volatile memories," in Proc. of IEEE Int. Symp. on Information Theory, ISIT, July 2013, pp.714-718.

[4] S. Kang et al., "A 0.1-m 1.8-V 256-Mb phase-change random access memory (PRAM) with 66-MHz synchronous burst-read operation," IEEE Journal of Solid-State Circuits, vol. 42.1, pp.210-218, Jan 2007.

[5] A. Berman, Y. Birk, "Constrained flash memory programming" in Proc. of IEEE Int. Symp. on Information Theory, July 2011, pp.2128-2132.

[6] J. van Lint and R. Wilson, *A Course in Combinatorics, second edition*. Cambridge UK: Cambridge University Press, 2001.

[7] A. Mohr and T.D. Porter, "Applications of Chromatic Polynomials Involving Stirling Numbers", Department of Mathematics Southern Illinois University, 2008.

[8] C. Trinh et al., "A 5.6 MB/s 64 Gb 4 b/Cell NAND ash memory in 43 nm CMOS," ISSCC, vol. 52, pp.246-247, Feb. 2009.

[9] Y. Cassuto and M. Blaum, "Codes for symbol-pair read channels", IEEE Transactions on Information Theory, Vol 57, No. 12, December 2011.