On the Use of Hard-Decision LDPC Decoders on MLC NAND Flash Memory

Khoa Le and Fakhreddine Ghaffari
ETIS, UMR-8051, Université Paris Seine, Université de Cergy-Pontoise, ENSEA, CNRS, France.
{khoa.letrung, fakhreddine.ghaffari}@ensea.fr

***Abstract*—The soft-decision Low-Density Parity-Check (LDPC) decoders are widely used in the multi-level per cell (MLC) NAND flash memory to ensure the correctness of stored data. However, to obtain the good error correction capability, these soft-decision decoders require the soft-information, which results a long on-chip memory sensing and dramatically extends the memory read latency. In this paper, we explore the possibility of using some recent introduced Bit Flipping (BF) hard decision decoders, in replacing the soft-decision decoders, to correct the retention errors appeared in the MLC NAND flash memory. The applied BF decoders require only the hard information read from that memory to decode and thus, by avoiding soft-information sensing, the memory read latency is improved. Furthermore, by using the characteristic of the retention error where some information bits can be determined as error-free, we propose the adaptation of the BF decoders to be better adapted to MLC NAND flash memory. We show that, the adapted BF decoders can correct more erroneous bits thanks to these error-free values. The simulation results show that, the adapted BF decoders running on the flash memory provide a good error correction performance, being close to that of the soft-decision decoders, with only the hard information of the memory. In addition, it is shown that, the BF decoders provide a higher decoding speed compared to the soft-decision decoder.**

I. Introduction

NAND flash memory have been used in practical applications thanks to its favorable features such as the high data throughput, low-power consumption, compactness [1]. In order to increase the storage capacity of NAND flash memory, multi-level cell technology have been developed to store more than 1 bit per cell, *e.g.* 2 bits per cell in [2], 3 bits per cell in [3] or 4 bits per cell in [4]. This increasing storage density results to the fact that, the stored data is more prone to error due to various underlying physical factors [5], appearing at the type of retention error, cell-to-cell interference (CCI) error *etc*. This MLC technology significantly increases the bit error rate and a powerful error correction code with soft-information has to be employed to yield a good error correction performance. There are several works on the use of the soft-decision LDPC codes for error correction on MLC NAND flash memory [2][6][7]. There are also several works on efficient hardware implementation of LDPC decoding algorithm on that type of memory [8]. All of these researches have demonstrated that, the soft-decision LDPC decoders can achieve a noticeable error correction gain over the conventional bose-chaudhuri-hocquenghem (BCH) code. However, since the MLC NAND flash memory read latency is proportional to the memory sensing precision, the use of soft-information memory sensing causes significantly increase of memory read latency.

Low-Density Parity-Check (LDPC) are the well-known error correction code (ECC) applied in several communication standards such as IEEE 802.11n, 802.3an, 802.16a *etc.* [9][10][11] and storage applications as in [12][13][14]. LDPC can be effectively decoded by the iterative decoding algorithms which are the iterative process where messages are iteratively passed between the computing units: Variable Nodes (VNs) and Check Nodes (CNs). An LDPC decoder is either the *soft-information* message passing decoder, *i.e.* Min Sum (MS), Sum Product (SP) [15], or *hard-decision* decoder, *i.e.* Bit Flipping (BF), Gallager-A,B [16]. The soft-decision decoders require the soft-information from channel and during the decoding process, the soft-information are passed between the computing units. These soft-decision decoders provide the outstanding error correction, approaching the capacity. However, the decoder complexity is dramatically increased while the decoding speed is relatively low, due to the intensive computation. On the contrary, the hard decision decoders have been recently attracted much attention due to the high decoding throughput, low-complexity implementation and the progressive improvement in error correction, approaching or even surpassing the soft-decision decoders [17].

Recently, several BF decoders such as Gradient Descent Bit Flipping (GDBF) [18], Probabilistic Gradient Descent Bit Flipping (PGDBF) [19], Probabilistic Parallel Bit Flipping (PPBF) [20] decoders, have been introduced, in which the error correction performance is significantly improved, being very close or even comparable to the performance of soft-decision decoders. These decoders are shown to be very low-complexity, which comes from the fact that, only binary messages are iteratively passed between the computing units simplifying the connection networks. Furthermore, the computing units which process these binary messages are very simple and therefore, enhance the decoding throughput. More interestingly, these decoders require only the hard information from the channel to start the decoding process.

In this paper, we investigate on the use of BF decoders, in replacing the soft-decision decoders, to correct the retention errors in the MLC NAND flash memory. We focus on the GDBF and PGDBF on MLC NAND flash memory in terms of error correction as well as the hardware efficiency such as the decoding throughput and decoder complexity, by comparison

with the soft-decision decoders. We take benefit from the fact that, the BF decoders require only hard information from channel to avoid the long memory sensing, and from that, the memory read latency is obviously reduced. Moreover, by using the characterization of the retention errors in MLC NAND flash memory, we can determine certain bits which are reliable. We then adapt the BF decoders such that the reliable VNs are kept unchanged in the VN processing units (VNUs) while decoding only on other unreliable VNs. We show that, with the additional information in determining the reliable VNs in MLC NAND flash memory, the modified BF decoders can correct more retention errors than the conventional counterparts.

The rest of the paper is organized as following. In Section II, we recall the characteristic of the retention noise in MLC NAND flash memory and characterize the principle to determine the reliable bits when the retention errors occur. The LDPC decoding principle and the deployed BF decoders (the GDBF and PGDBF decoders) are also introduced. In Section III, we present the adapted BF decoders (denoted as A-GDBF and A-PGDBF) dedicated for MLC NAND flash memory. In Section IV, we show for the test case that, the BF decoders provide a comparable error correction to that of the soft-decision decoders (the MS decoder) while the average number of decoding clock cycles is only $1/4$. Although the advantages of avoiding soft-information sensing is not quantified, the fact that the adapted BF decoders use only the hard-decision information reduces obviously the read latency of MLC NAND flash memory. Section V concludes the paper.

II. The MLC NAND flash memory retention error characterization and BF LDPC decoders recall

A. The characterization of retention errors in MLC NAND flash memory

MLC NAND flash memory allows storing more information in a single cell which is realized by the number of charges trapped in the floating gate and the numerous voltage thresholds to identify the storage states [2]. In each MLC NAND flash memory cell, the voltage levels between two adjacent storage states become closer, making it being more prone to errors when being read. Fig. 1 shows the relation between the stored data and its corresponding threshold voltage distribution for a 2-bits per cell NAND flash memory. There are 4 states of the stored data which are usually represented by the Gray code to reduce the raw bit error probability. While the data is stored in the memory cell, the programmed cell gradually loses its charge in the floating gate (Fig. 1b). This leads eventually to a change in logic states, causing the error known as the retention error. The appearance of retention error depends on several factors *e.g.* the number of Program/Erase (P/E), the storing time. There are also other types of errors in MLC NAND flash memory such as the cell-to-cell interference (CCI) or programming error (which appears when the data is being written). However, several works in literature showed that, the retention error is the main source of error appearing in MLC NAND flash memory [5]. We assume in this work that only

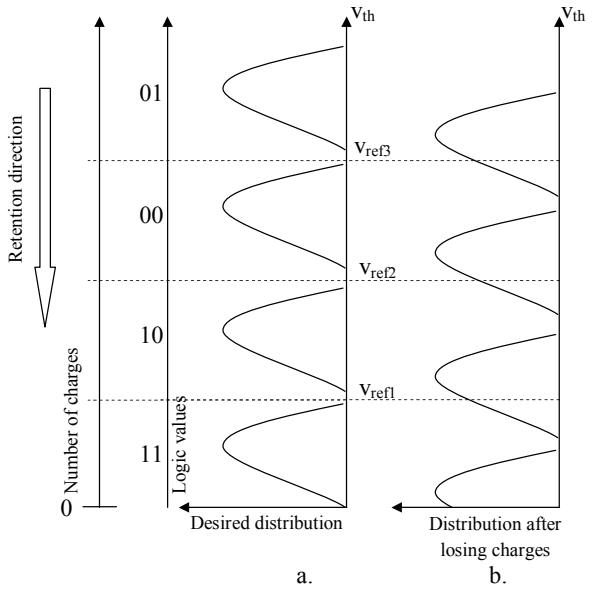

Figure 1. The retention behavior in MLC NAND flash memory.

retention errors appear in the read data from MLC NAND flash memory. Also, we limit the illustrations in this work to only 2-bits per cell MLC NAND flash memory. When reading data from an MLC NAND flash memory cell, the return data is in the form as (most-significant-bit, least-significant-bit) or (MSB,LSB). Further information on the MLC NAND flash memory such as the organization, program/read process can be found in [2][6][7].

The charge trapped in the floating gate is gradually diminished forming the fact that, the retention error is always in the type of one-direction state transitions. Indeed, it can be seen in Fig. 1 that, the logic state 01 is realized at the highest charge trapped. The loss of charge changes only that logic state to the lower charge level states such as states 00 or 10 and there is no logic state that can form the state 01 when the retention errors happen. This implies the fact that, the 01 read from MLC NAND flash memory cell is the error-free (reliable) information. This observation have also introduced in [21]. The work of [21] also showed more cases where we can determine the reliability of the read data. Indeed, the MSB when reading 00 from the cell is reliable because if the retention error occurs, it changes from the state 01 to 00 and only LSB is changed. With the same principle, the LSB of 10 and MSB of 11 are also the reliable data. Note that, we assume in this work that, only single step retention error (where the logic state is only changed to its closest adjacent state) appears in MLC NAND flash memory. It is an acceptable assumption since the multiple state jump is extremely rare thanks to the observation in [5][22]. By using this characterization of retention error, we propose to have a pre-processing on the read data from the MLC NAND flash memory to determine the reliable information before launching the LDPC decoding process. This extra information will help the BF correct more

erroneous bits in the decoding process.

B. GDBF and PGDBF decoders

An LDPC code is defined by a sparse parity-check matrix H (M, N), $N > M$. Each row represents a parity check function, computed by a CN, on the VNs represented by the columns. The VN v_n $(1 \leq n \leq N)$ is checked by the CN c_m $(1 \leq m \leq M)$ if the entry $H(m, n) = 1$ and they are called neighbors. LDPC code can also be represented by a bipartite graph called Tanner graph composing two groups of nodes, the CNs c_m, $(1 \leq m \leq M)$ and the VNs v_n, $(1 \leq n \leq N)$. The VN v_n $(1 \leq n \leq N)$ connects to the CN c_m $(0 \leq m \leq M)$ by an edge in the Tanner graph if the entry $H(m, n) = 1$. We denote the set of CNs connected to the VN v_n as $\mathcal{N}(v_n)$, $(1 \leq n \leq N)$, and $|\mathcal{N}(v_n)|$ is called VN degree. Similarly, $\mathcal{N}(c_m)$, $(1 \leq m \leq M)$ denotes all VNs connected CN m and $|\mathcal{N}(c_m)|$ is the CN degree. This paper works on the regular LDPC code, *i.e.* $|\mathcal{N}(v_n)| = d_v$ and $|\mathcal{N}(c_m)| = d_c$ $\forall n, m$. When LDPC ECC is applied in the MLC NAND flash memory, the user data is encoded and stored in form of a codeword. A vector $\mathbf{x} = \{x_n\} = \{0, 1\}^N$, $(1 \leq n \leq N)$ is called a codeword if and only if $H\mathbf{x}^T = 0$. We denote $\mathbf{y} = \{y_n\}$, $(1 \leq n \leq N)$ as the read version of $\mathbf{x}$ from the memory. The retention error in this work is modeled as a crossover behavior such that each bit of $\mathbf{x}$ is flipped with a probability α, called raw Bit Error Rate (BER), *i.e.*, $\Pr(y_n = x_n) = 1 - \alpha$ and $\Pr(y_n = 1 - x_n) = \alpha$, $1 \leq n \leq N$. We use the superscript (k) along with the node (VN and CN) notations to indicate that node value at the k iteration.

The decoding process of BF decoders is an iterative process where the binary messages are iteratively passed between the VNs and CNs. The CN operation is the parity check operation which is formulated in Equ. 1 where $\oplus$ is the Exclusive-OR (XOR) operation. The decoding process will be terminated if all the CN are satisfied *i.e.* $c_m^{(k)} = 0, \forall m$, otherwise the decoding process continues with the VN operations.

$$c_m^{(k)} = \bigoplus_{v_n \in \mathcal{N}(c_m)} v_n^{(k)} \tag{1}$$

$$E_n^{(k)} = v_n^{(k)} \oplus y_n + \sum_{c_m \in \mathcal{N}(v_n)} c_m^{(k)} \tag{2}$$

In each VN operation, first of all, a so-called energy value is computed using Equ. 2 basing on the all neighbor CN values, the VN current value ($v_n^{(k)}$) and its value received from the channel (y_n). The maximum energy value, $E_{max}^{(k)}$, is computed by a Maximum Finder (MF). In GDBF decoder, the value of a VN will be flipped if and only if its energy value is equal to the maximum energy value. In PGDBF, a binary random generator is implemented generating for each VN a random signal $R_n^{(k)}$ where $\Pr(R_n^{(k)} = 1) = p$ and $\Pr(R_n^{(k)} = 0) = 1 - p$ and p is a pre-defined signal. The value of each VN in PGDBF is flipped if and only if its energy equals to the maximum energy and $R_n^{(k)} = 1$. The GDBF and PGDBF provide the good error correction performance while requiring only the hard-information as shown in [19][23]. We present in the next section the adaptation of these decoders to work on the MLC NAND flash memory.

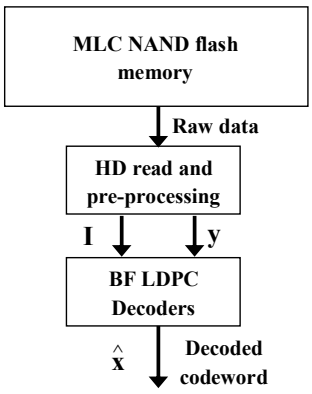

Figure 2. The utilization of BF decoders on the MLC NAND flash memory system.

III. Adapted BF (A-BF) decoders for MLC NAND flash memory

By understanding the retention error characteristic, we present the adaptation of the BF decoders to MLC NAND flash memory system. The MLC NAND flash memory system is described in the Fig. 2. The adaptation includes the additional pre-processing unit which provides the reliability signal for each data bit read from memory. The second part of the adaption is the modification in the decoding process basing on the reliability information from the pre-processing module.

A. The pre-processing module

The target of designing the pre-processing block is to identify the reliability of each bit in the data read from MLC NAND flash memory and to use this information improving the performance of the BF decoders. As presented above, 2 bits (MSB and LSB) are stored in the same cell and basing on their values, the reliability can be determined. We denote the reliability of the MSB as I^{MSB} and that of the LSB as I^{LSB}. The value of these variables represents the reliability of the bits (reliable bit is denoted by 1, otherwise, it is marked by 0). We show the values of I^{MSB} and I^{LSB} as the functions of MSB and LSB in table I. It can be seen that, when the read data is 01, both of MSB and LSB are reliable. In the other cases, only one bit (either MSB or LSB) is reliable.

The pre-processing module is designed to identify the reliability of a stored codeword and to produce a binary vector $\mathbf{I} = (I_1, I_2, \ldots, I_N)$. Each element of $\mathbf{I}$ represents the reliability of the corresponding bits (or VNs) in the read data. In the MLC NAND flash memory layout, two bits (MSB and LSB) in a cell may belong to the same codeword [22] or they are in two different codewords. In the latter case, pre-processing module needs the read of two consecutive codewords (in N cells) in order to compute $\mathbf{I}$ for a codeword, while in the former case, it needs the information in $N/2$ cells. A computing block is designed to produce the values I^{MSB}

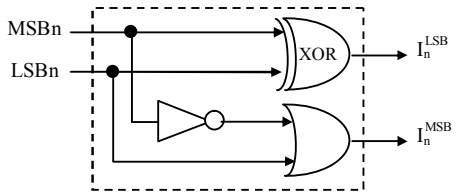

Figure 3. The computing unit to identify the reliability of the read data from one MLC NAND flash memory.

and I^{LSB} on the value read (MSB and LSB). The detailed circuit of this computation block is described in Fig. 3. In general, the pre-processing module may be implemented with only 1 computing block and used serially or many blocks are implemented operating in parallel for accelerating the reading speed. It can be seen that, the pre-processing unit is a simple combinatory circuit and it would require a negligible hardware overhead.

Table I
THE RELIABILITY DETERMINATION BASED ON THE DATA READ FROM A MLC NAND FLASH CELL.

MSB	LSB	I^{MSB}	I^{LSB}
0	1	1	1
0	0	1	0
1	0	0	1
1	1	1	0

B. The adapted BF decoders for MLC NAND flash memory

The pre-processing unit provides the reliability information of each bit read from the MLC NAND flash memory. We adapt the BF decoders to be used in this type of memory. We have applied our strategy for the GDBF and PGDBF which are named as A-GDBF and A-PGDBF, respectively. The A-PGDBF decoding algorithm is described in Algorithm 1. It can be seen that, the A-PGDBF follows similarly the decoding step of the original PGDBF and the modification is on the VN operation where a verification step is added. In the VN processing of A-PGDBF, the flipping step is proceeded only if that VN value is indicated as unreliable ($I_n = 0$). This principle is also applied for the A-GDBF.

It can be seen that, the reliability information of each VN is used in the adapted decoders, changing the behaviors of VNs, compared to the original decoders. Furthermore, this information helps the adapted BF decoders correct more errors in the read data than their original counterparts. We illustrate an example where the original decoder (GDBF) fails to correct the errors while the adapted decoder (the A-GDBF) can correct in Fig. 4. It is shown that, the GDBF fails with only 3 error VNs located on a trapping set (5,3) (in Fig. 4a) [23] (the white circles (squares) represent the correct VNs (satisfied CNs) while the black circles (squares) represent the erroneous VNs (unsatisfied CNs), with the assumption that all zero codeword is sent). GDBF fails to find the correct codeword because when flipping the erroneous VNs (v_1 and v_3), it flips also the correct VNs (v_2 and v_4) and oscillates between the two states as in Fig. 4a and Fig. 4b. On the contrary, in A-GDBF decoder, the erroneous VNs will be certainly flipped while the correct VNs will be flipped only when they are unreliable. A-GDBF can correct all the errors in Fig. 4a after 2 iterations (the error configuration as in Fig. 4a $\rightarrow$ Fig. 4c $\rightarrow$ Fig. 4d) if the correct VNs v_2 and v_4 are reliable. It can also correct all error VNs in 3 iterations by going Fig. 4a $\rightarrow$ Fig. 4e $\rightarrow$ Fig. 4c $\rightarrow$ Fig. 4d if v_2 is reliable while v_4 is unreliable. A-GDBF fails to correct the errors when both v_2 and v_4 are unreliable. Therefore, while GDBF always fails, A-GDBF can correct 75% when this error pattern appears. The advantage in error correction of the adapted BF decoders is shown further in the simulation results in the next section.

Algorithm 1 The Adapted-PGDBF decoding algorithm for MLC NAND Flash memory

Initialization $k = 0, \mathbf{v}^{(0)} = \mathbf{y}$
while $k \leq It_{max}$ **do**
  for $1 \leq m \leq M$ **do** ▷ CN processing
    Compute $c_m^{(k)}$, using Equ. (1).
  end for
  if $\mathbf{c}^{(k)} = \mathbf{0}$ **then** ▷ Check syndrome
    exit while
  end if
  for $1 \leq n \leq N$ **do** ▷ VN processing
    Compute $E_n^{(k)}$, using Equ. (2).
  end for
  Sort $E_{max}^{(k)} = max_n(E_n^{(k)})$
  for $1 \leq n \leq N$ **do**
    if $I_n = 0$ **then** ▷ Added verification
      Generate $R_n^{(k)}$ from $\mathcal{B}(p)$.
      if $E_n^{(k)} = E_{max}^{(k)}$ **and** $R_n^{(k)} = 1$ **then**
        $v_n^{(k+1)} = v_n^{(k)} \oplus 1$
      end if
    end if
  end for
  $k = k + 1$
end while
Output: $\mathbf{v}^{(k)}$

IV. ERROR CORRECTION PERFORMANCE AND ANALYSIS

In this section, we present the simulated error correction performance of adapted BF decoders on the MLC NAND flash memory with the retention errors. Although the practical LDPC code in the MLC NAND flash memory is usually with very high code rate and the long codeword [8], we simulated our BF decoders on a moderate code rate ($R = 0.75$), and relative short ($N = 1296$) code to save the simulation time. We assume that, the MSB and LSB in a MLC NAND flash memory cell belong to 2 different codewords and we decode only the codeword formed by the LSB bits. As shown above,

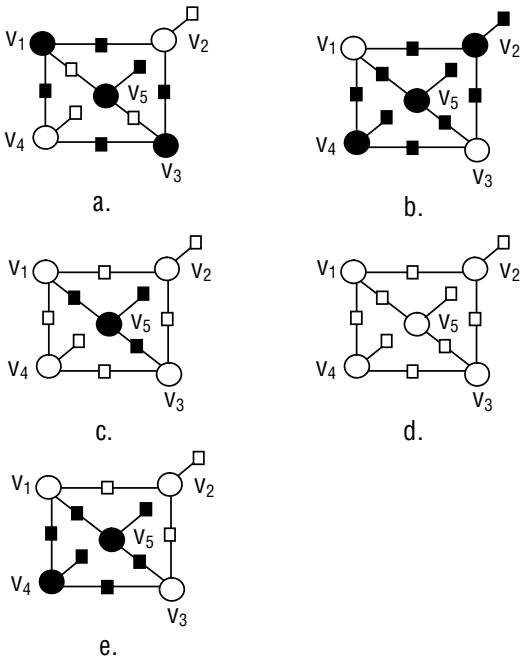

Figure 4. The error pattern with which the GDBF fails to correct the error while the A-GDBF can.

the LSB codeword has higher probability to be erroneous than the MSB codeword. We produce, therefore, the worst-case decoding performance of our BF decoders. We also show the decoding performance of the quantized MS decoder with 4 bits for LLR and 6 bits for the AP-LLR information [24]. The MS decoder is implemented with the layered scheduling in which a decoding iteration is composed by 4 layers.

It can be seen in the Fig. 5 in general that, the BF decoders provide the good error protection ability for MLC NAND flash memory. Indeed, the A-PGDBF provides an BER at 8.10^{-10} and A-GDBF is with 4.10^{-8} when the raw BER at 2.10^{-3}. The A-PGDBF is much better than the A-GDBF decoder in term of error correction at the cost of hardware overhead when being implemented, as observed in many works [23][25]. The reliability information of each bit helps the BF decoders correct more retention errors. In fact, we introduce also in the Fig. 5 the decoding performance of the BF decoders when the bit reliability information are not used and denote them as baseline. It is shown that, the A-GDBF loses around 5dB in BER while A-PGDBF loses 2dB at raw BER 2.10^{-3}. Also, the A-GDBF seems to be affected on all values of α while the A-PGDBF is affected on the error floor region only.

In general, the MS is better than all BF decoders in error correction at the cost of hardware complexity [23] and the decoding throughput. We show in Fig. 6 the comparison on the average number of clock cycles needed to decode a codeword, as a function of α. Due to the lack of space, we do not show the detail of the BF and MS hardware architectures (which can be found in [23] and [24]). The MS decoders requires 2 clock cycles to decode 1 layer (it does 4 layers in an iteration) while

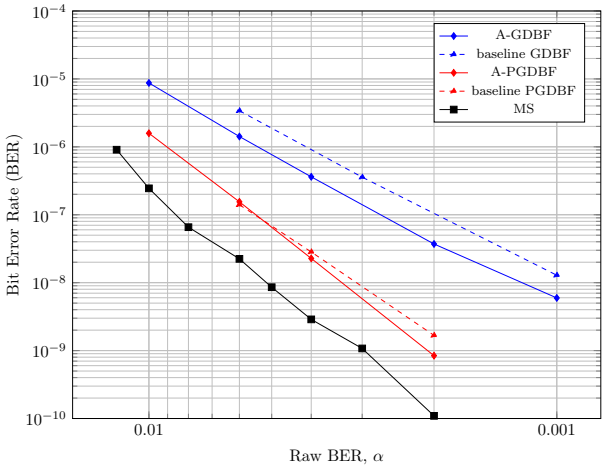

Figure 5. The decoding performance of the BF decoders on MLC NAND flash memory.

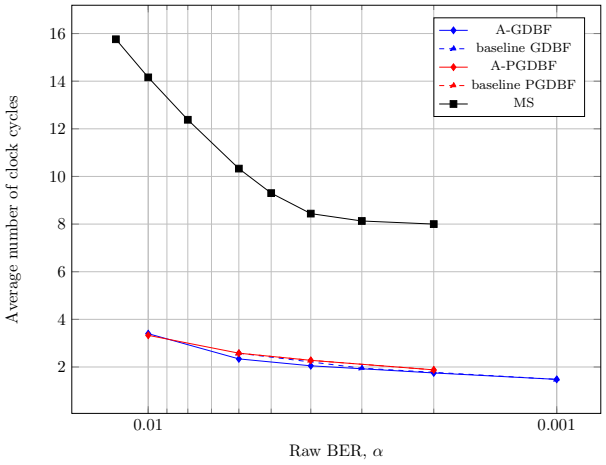

Figure 6. The average clock cycles required to decode 1 codeword.

the BF decoders requires 1 clock cycle to finish 1 iteration. The difference in the average number of clock cycles can be interpreted to the difference in decoding throughput if the decoders operate at the same frequency. It can be seen in the Fig. 6 that, the BF decoders provide a much higher decoder speed than that of the MS decoder. At raw BER 2.10^{-3}, the BF decoders is 4 times faster than the MS decoder. Note further that, the BF decoders require only the hard information from MLC NAND flash memory while the soft-decision decoders need the soft-information. Although we do not quantify in detail this advantage, it will obviously improve the reading latency when the BF decoders are used in MLC NAND flash memory.

V. Conclusion

We propose in this paper an investigation in using the recent introduced BF hard-decision LDPC decoders (the GDBF and PGDBF decoders), in replacing the soft-decision, to correct the retention error in MLC NAND flash memory. The motivation

comes from the fact that, BF decoders require only the hard-information from the channel to proceed the decoding process, and from that, the long soft-information sensing can be omitted. Furthermore, by using the characteristic of retention error, we have adapted these BF decoders in such a way that they can correct more errors in MLC NAND flash memory. The adapted BF decoders show an admirable error correction improvement, being very close or even comparable to soft-decision decoders. The advantage of using BF decoders are also confirmed by the very small number of clock cycles needed for decoding a codeword which is interpreted to the high decoding throughput. The promising results in terms of error correction capability, decoding throughput and complexity lead the BF decoders to become the good ECC candidates in the new generation MLC NAND flash memory.

ACKNOWLEDGMENT

This work was funded by the french ANR under grant number ANR-15-CE25-0006-01 (NAND project).

REFERENCES

[1] R. S. Liu, M. Y. Chuang, C. L. Yang, C. H. Li, K. C. Ho, and H. P. Li, "Improving read performance of nand flash ssds by exploiting error locality," *IEEE Transactions on Computers*, vol. 65, no. 4, pp. 1090–1102, April 2016.

[2] H. Sun, W. Zhao, M. Lv, G. Dong, N. Zheng, and T. Zhang, "Exploiting intracell bit-error characteristics to improve min-sum ldpc decoding for mlc nand flash-based storage in mobile device," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 8, pp. 2654–2664, Aug 2016.

[3] Y. L. et al., "A 16 gb 3-bit per cell (x3) nand flash memory on 56 nm technology with 8 mb/s write rate," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 1, pp. 195–207, Jan 2009.

[4] N. S. et al., "A 70 nm 16 gb 16-level-cell nand flash memory," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 4, pp. 929–937, April 2008.

[5] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Error patterns in mlc nand flash memory: Measurement, characterization, and analysis," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2012, pp. 521–526.

[6] G. Dong, N. Xie, and T. Zhang, "Enabling nand flash memory use soft-decision error correction codes at minimal read latency overhead," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 9, pp. 2412–2421, Sept 2013.

[7] D. h. Lee and W. Sung, "Estimation of nand flash memory threshold voltage distribution for optimum soft-decision error correction," *IEEE Transactions on Signal Processing*, vol. 61, no. 2, pp. 440–449, Jan 2013.

[8] J. Kim and W. Sung, "Rate-0.96 ldpc decoding vlsi for soft-decision error correction of nand flash memory," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 5, pp. 1004–1015, May 2014.

[9] "Wireless lan medium access control and physical layer specifications: Enhancements for higher throughput, p802.11n/d3.07," *IEEE-802.11n*, Mar. 2008.

[10] "Local and metropolitan area networks - part 16, ieee std 802.16a-2003," *IEEE-802.16*.

[11] "Ieee standard for information technology - corrigendum 2: Ieee std 802.3an-2006 10gbase-t correction," *IEEE Std 802.3-2005/Cor 2-2007 (Corrigendum to IEEE Std 802.3-2005)*, Aug. 2007.

[12] S. Jeon and B. V. K. V. Kumar, "Performance and complexity of 32 k-bit binary ldpc codes for magnetic recording channels," *IEEE Transactions on Magnetics*, vol. 46, no. 6, pp. 2244–2247, June 2010.

[13] K. C. Ho, C. L. Chen, Y. C. Liao, H. C. Chang, and C. Y. Lee, "A 3.46 gb/s (9141,8224) ldpc-based ecc scheme and on-line channel estimation for solid-state drive applications," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 1450–1453.

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

[15] D. Declercq, M. Fossorier, and E. Biglieri, "Channel coding: Theory, algorithms, and applications," *Academic Press Library in Mobile and Wireless Communications, Elsevier, ISBN: 978-0-12-396499-1*, 2014.

[16] R. G. Gallager, "Low density parity check codes," *MIT Press, Cambridge, 1963, Research Monograph series*, 1963.

[17] G. Sundararajan, C. Winstead, and E. Boutillon, "Noisy gradient descent bit-flip decoding for ldpc codes," *IEEE Transactions on Communications*, vol. 62, no. 10, pp. 3385–3400, Oct 2014.

[18] T. Wadayama, K. Nakamura, M. Yagita, Y. Funahashi, S. Usami, and I. Takumi, "Gradient descent bit flipping algorithms for decoding ldpc codes," *IEEE Transactions on Communications*, vol. 58, no. 6, pp. 1610–1614, June 2010.

[19] O. A. Rasheed, P. Ivanis, and B. Vasić, "Fault-tolerant probabilistic gradient-descent bit flipping decoder," *IEEE Communications Letters*, vol. 18, no. 9, pp. 1487–1490, Sept 2014.

[20] K. Le, F. Ghaffari, D. Declercq, B. Vasic, and C. Winstead, "A novel high-throughput, low-complexity bit-flipping decoder for ldpc codes," in *2017 International Conference on Advanced Technologies for Communications (ATC)*, Oct 2017, pp. 126–131.

[21] L. Qiao, H. Wu, D. Wei, and S. Wang, "A joint decoding strategy of non-binary ldpc codes based on retention error characteristics for mlc nand flash memories," in *2016 Sixth International Conference on Instrumentation Measurement, Computer, Communication and Control (IMCCC)*, July 2016, pp. 183–188.

[22] M. Zhang, F. Wu, X. He, P. Huang, S. Wang, and C. Xie, "Real: A retention error aware ldpc decoding scheme to improve nand flash read performance," in *2016 32nd Symposium on Mass Storage Systems and Technologies (MSST)*, May 2016, pp. 1–13.

[23] K. Le, F. Ghaffari, D. Declercq, and B. Vasić, "Efficient hardware implementation of probabilistic gradient descent bit-flipping," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 4, pp. 906–917, April 2017.

[24] T. Nguyen-Ly, K. Le, F. Ghaffari, A. Amaricai, O. Boncalo, V. Savin, and D. Declercq, "Fpga design of high throughput ldpc decoder based on imprecise offset min-sum decoding," in *2015 IEEE 13th International New Circuits and Systems Conference (NEWCAS)*, June 2015, pp. 1–4.

[25] K. Le, D. Declercq, F. Ghaffari, L. Kessal, O. Boncalo, and V. Savin, "Variable-node-shift based architecture for probabilistic gradient descent bit flipping on qc-ldpc codes," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. PP, no. 99, pp. 1–13, 2017.