JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 25 No. 1, pp. 29-36, January 2020
https://doi.org/10.9708/jksci.2020.25.01.029

An Aging Measurement Scheme for Flash Memory Using LDPC Decoding Information

Taegeun Kang*, Hyunbean Yi*

*Student, Dept. of Computer Engineering, Hanbat National University, Daejeon, Korea
*Professor, Dept. of Computer Engineering, Hanbat National University, Daejeon, Korea

[Abstract]

Wear-leveling techniques and Error Correction Codes (ECCs) are essential for the improvement of the reliability and durability of flash memories. Low-Density Parity-Check (LDPC) codes have higher error correction capabilities than conventional ECCs and have been applied to various flash memory-based storage devices. Conventional wear-leveling schemes using only the number of Program/Erase (P/E) cycles are not enough to reflect the actual aging differences of flash memory components. This paper introduces an actual aging measurement scheme for flash memory wear-leveling using LDPC decoding information. Our analysis, using error-rates obtained from an flash memory module, shows that LDPC decoding information can represent the aging degree of each block. We also show the effectiveness of the wear-leveling based on the proposed scheme through wear-leveling simulation experiments.

▸ **Key words**: Flash Memory, Wear-Leveling, ECC, LDPC Codes, Aging Measurement

[요 약]

웨어-레벨링과 오류정정코드는 플래시 메모리의 신뢰성과 내구성을 위한 필수적인 기술이다. 플래시 메모리를 구성하는 요소들은 사용횟수에 따른 노화도가 서로 다를 수 있다. 따라서 기존의 쓰기/지우기 횟수를 바탕으로 하는 웨어-레벨링 기술은 요소들의 실제 노화도 차이를 반영하기에 충분하지 않다. 본 논문에서는 높은 오류정정율이 증명된 Low-Dencity Parity-Check (LDPC) 코드를 적용하고 복호 과정에서 나오는 정보를 이용하여 플래시 메모리의 실제 노화도를 측정하는 방법을 소개한다. 실험에서는 실제 플래시 메모리를 대상으로 측정한 오류율 데이터를 기반으로 LDPC 코드 복호 정보가 플래시 메모리 각 블록의 노화도를 나타낼 수 있음을 보인다. 또한, 웨어-레벨링 시뮬레이션을 통하여 제안하는 노화도 측정 방법 기반의 웨어-레벨링의 효과를 입증한다.

▸ **주제어**: 플래시 메모리, 웨어-레벨링, 오류정정코드, LDPC 코드, 노화 측정

• First Author: Taegeun Kang, Corresponding Author: Hyunbean Yi
*Taegeun Kang (taegnism@hanbat.ac.kr), Dept. of Computer Engineering, Hanbat National University
*Hyunbean Yi (bean@hanbat.ac.kr), Dept. of Computer Engineering, Hanbat National University
• Received: 2019. 11. 11, Revised: 2020. 01. 14, Accepted: 2020. 01. 14.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

Despite numerous advantages such as fast I/O, low power consumption, and high storage capacity, the limited lifespan of flash memories remains a challenge. As the P/E operation repeats, durability degrades, and each component eventually reaches a state in which it cannot operate normally. In order to keep the life of flash memories stable for a long time wear-leveling techniques and Error Correction Codes (ECCs) are essential.

An ECC, which is used in a flash memory device, encodes data to be stored into memory by adding redundancy information during a write operation, and decodes the stored data through a decoding process during a read operation [1, 2]. The lifespan of the flash memory can be ensured to a certain extent by the error recovery performance of the ECC algorithm used. Low-Density Parity-Check (LDPC) codes have recently been in the spotlight because of their high recovery performance [2-5]. A series of LDPC decoding process are repeated until all errors are corrected. Typically, the number of decoding iterations increases according to the number of errors. It means that the number of decoding iterations can be used for predicting an aging degree of each component.

Even though a flash memory is manufactured in a sophisticated process, aging rates can be different among internal components because of its fabrication variations. A conventional wear-leveling scheme based on only the number of P/E cycles is not enough to reflect the actual aging of flash memories. It is necessary to continue to monitor how much flash memory components have been worn out. In order to measure the actual aging degree and apply it to wear-leveling with a low overhead, this paper presents an aging measurement scheme that exploits the number of decoding iterations of LDPC codes.

This paper is organized as follows: Chapter II gives background of our work. Chapter III presents the proposed aging measurement scheme, including how to derive the aging degree from LDPC decoding information, and how to apply the information to a wear-leveling. Analysis results and simulation results, based on an actual flash memory model, are shown in Chapter IV. In Chapter V, we conclude the paper.

II. Background

1. Limited Lifetime of Flash Memories

A flash memory chip consists of thousands of blocks and each block is composed of hundreds of pages. Erase and write/read operations are respectively performed in blocks and pages. Data are stored in flash memory cells, which are made from floating-gate transistors that can hold charges. Electrons are ejected or injected during P/E operations, which can cause physical damage to transistors and thus create errors [6-8].

The goal of wear-leveling is to have pages and blocks used evenly so that the available memory space can be maximized. Recently research has been focused on static wear-leveling schemes that balance usage between hot data, which is frequently updated, and cold data, which is seldom modified, by changing the physical mapping of each kind of data. In [9], a static wear-leveling algorithm is proposed for determining priorities in a cold data group and a hot data group and calculating the exchange conditions. In [10, 11], efficient methods are introduced to implement static wear-leveling algorithms with low system resources. However, although flash memories are manufactured through sophisticated processes, aging rates among memory components can be different due to fabrication variations. Conventional wear-leveling schemes based only on the number of P/E cycles are not enough to reflect the actual aging degree of flash memories.

In order to calculate the aging degree of flash memory components, bit-error-rates (BERs) or

program-error-rates (PERs) have been used [12-14]. These techniques analyze BERs or PERs during read or write operations. However, additional circuits and procedures to obtain and manage the error-rates can be needed.

2. LDPC Codes Decoding Iterations

LDPC codes can be represented as an $(n - k) \times n$ parity check matrix, shown in fig. 1 (a), where n = codeword length, k = message length, and $n - k$ = redundancy length. A decoding process is easily described using a tanner graph, shown in fig. 1 (b), with check nodes, variable nodes (or bit nodes), and edges between the nodes. Checks nodes and variable nodes denote, respectively, the rows and the columns of the matrix and edges. A check node performs a parity check operation with messages from the variable nodes connected to it and sends the result back to the variable nodes. Using the results from the check nodes, the variable nodes update the corresponding bits and send them back to the check nodes. This is called a message passing algorithm, which consists of the two functions described above, named Check Node Updating (CNU) and Variable Node Updating (VNU).

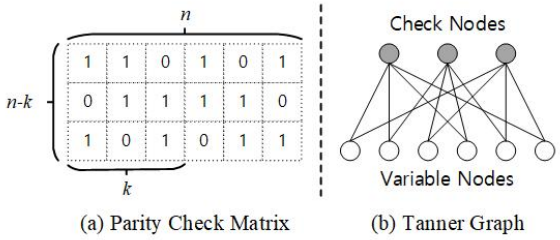


Fig. 1. A Parity Check Matrix and a Tanner Graph

A general LDPC decoding process is shown in fig. 2. A corrected codeword obtained by CNU and VNU is verified by an overall parity-check operation (Parity-Checking). This process is repeated until all errors are corrected or the number of decoding iterations reaches a pre-specified maximum count (Max Iteration). If the number of decoding iterations reaches Max Iteration, the corresponding memory area would be classified as an area that can no longer be used.

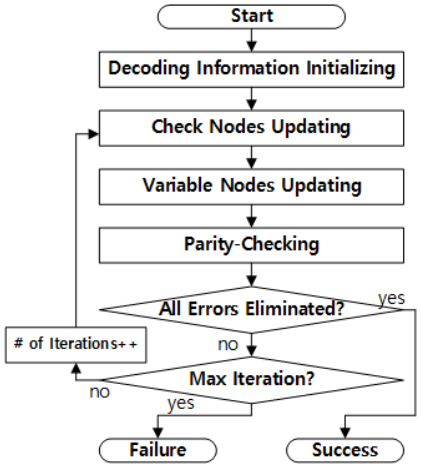


Fig. 2. A General Decoding Process of LDPC Codes.

In general, the more errors occur, the more iterations of the decoding process are required. The number of decoding iterations can be representative information used to measure the aging degree. However, due to the characteristics of LDPC codes, which are related to structure of parity-check matrices, the recovery rates vary depending on the locations of error bits [15, 16]. As a result, the number of decoding iterations can temporarily be greatly increased compared to the actual aging. Therefore, we use normalized values instead of directly using the number of decoding iterations. The proposed scheme to obtain the normalized number of decoding iterations and the aging degree are described in the following chapter.

III. Proposed Scheme

1. Aging Degree Measurement

The read operation in flash memories is performed in a page unit. Decoding is also conducted in a page unit during the read operation. If the error correction fails, the page or the block with the page may be determined to be worn out and difficult to use normally. As a result,

when the number of decoding iterations is *ITR*, the highest *ITR* out of *ITR*s of pages in a block becomes the representative *ITR* of the block. The representative *ITR* can be used for wear-leveling and bad block management.

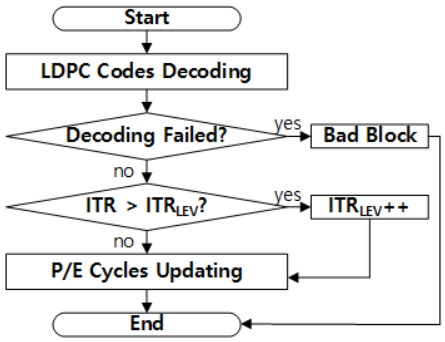


Fig. 3. Updating the Level of Decoding Iterations (*ITR*$_{LEV}$)

As mentioned previously, an *ITR* can be temporarily increased, unlike the actual aging degree. In addition, since the error probability of each bit is calculated by combining the results of many different parity checks, *ITR* can instantaneously increase with even small changes in error-rates. In particular, from timings when the error-rates become much higher, the probabilities of unexpected variation occurrences of *ITR* also become much higher than the probabilities of earlier periods. Recording and applying the number of decoding iterations every time error correction is performed can make aging monitoring complicated and degrade measurement accuracy. Therefore, in this paper we define a normalized variable, level of decoding iterations (*ITR*$_{LEV}$), to eliminate the momentary noise of *ITR* and reflect the gradual progress of actual aging. Fig. 3 shows the ITR normalization procedure that occurs after a decoding process is finished. If the error recovery fails, the corresponding block is treated as a bad block. Otherwise, *ITR*$_{LEV}$ goes up only one step even though a new *ITR* is much bigger than the previous *ITR*$_{LEV}$.

Although an *ITR*$_{LEV}$ can reflect the aging degree better than the number of P/E cycles (*PE*), if the flash memory manufacturing process is highly stable and the operating environment is not harsh, depending on only *PE* cycles may be sufficient to some extent. Therefore, by combining *PE* and *ITR*$_{LEV}$, an even more precise aging degree can be obtained. The aging degree (*AGING*$_{DEG}$) is as follows.

$$AGING_{DEG} = a\frac{ITR_{LEV}}{ITR_{MAX}} + (1-a)\frac{PE}{PE_{MAX}}$$

Where *ITR*$_{MAX}$, *PE*$_{MAX}$ and '*a*' are the maximum *ITR* (Max Iteration in fig. 2), the maximum *PE*, and the weight control parameter, which is a real number, respectively. *PE*$_{MAX}$, *ITR*$_{MAX}$, and '*a*' are set by manufacturers or test engineers through the analysis of the durability of flash memories and the error correction capability of the LDPC codes. The parameter '*a*' can be adjusted and programmed during operations according to the characteristics of the flash memory and the usage environment. For example, a wear-leveling can, in the early stages when errors rarely occur, set a = 0 and count only on *PE*, and from the stage when *PE* or *ITR* reaches certain values, can increase '*a*' and begin to calculate the aging degree. In this way, it is possible to reduce the operation complexity of wear-leveling by applying *ITR*$_{LEV}$ after a necessary timing.

2. View of Application Architecture

For the aging degree measurement, *PE* and the representative *ITR*$_{LEV}$ are recorded and updated when they are needed. A wear-leveling refers to the information at regular intervals or at certain events to perform bad block management, garbage collection, and address mapping. Devices including flash memories usually have a Flash Translation Layer (FTL) for interface between flash memories and the operation system. Fig. 4 shows an FTL including the aging monitoring function that collects *PE* and *ITR*$_{LEV}$ obtained from modules of LDPC codes. *PE* and *ITR*$_{LEV}$ are managed in table form organized in blocks. Based on the aging monitoring information, *AGING*$_{DEG}$ is calculated and transmitted when wear-leveling is performed.

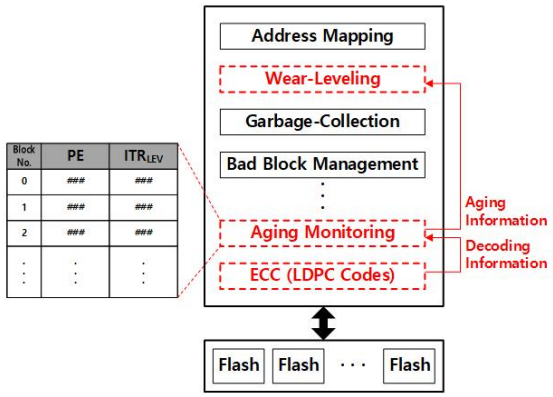


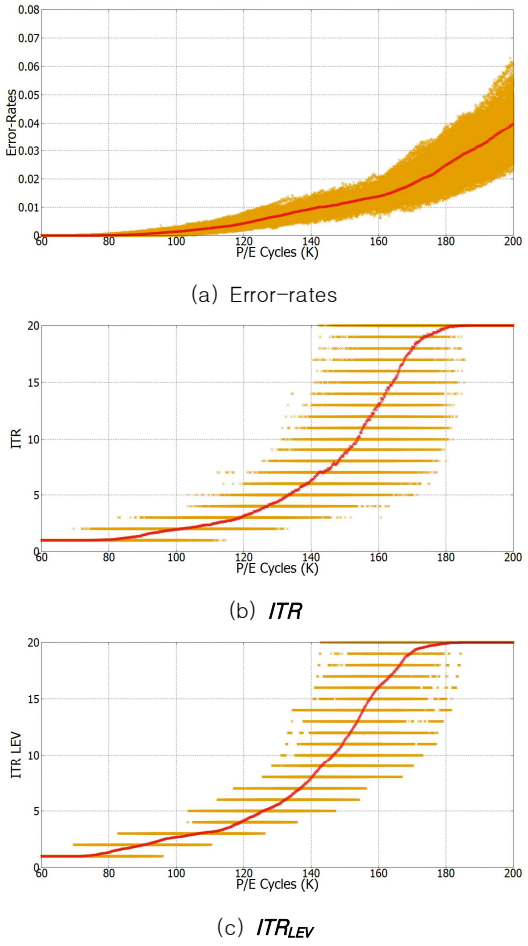
Fig. 4. An FTL Structure Including the Aging Monitoring

In the next chapter, we show that ITR_{LEV} can reflect the aging degree by analyzing error-rates, ITR, and ITR_{LEV}, based on an actual flash memory module. The effectiveness of the proposed aging measurement scheme is shown by performing wear-leveling simulations.

IV. Experimental Results

1. Evaluation of the Effectiveness of ITR_{LEV}

For our simulation, Quasi Cyclic (QC) LDPC code was designed and applied [17, 18]. In QC-LDPC codes a parity-check matrix is designed with cyclic shifted unit matrices of a certain size. The min-sum algorithm is adopted for check node update modules [19]. All simulations are conducted with ITR_{MAX} = 20. The flash memory module we used to observe error-rates consisted of 1024 blocks in one package and 64 pages in a block. The size of the page was 2112 bytes, including 64 bytes of extra space. The number of errors per page was measured for every 100 P/E cycles with respect to 150 blocks. The data retention time was not considered, and the same operation clock and P/E period were applied for every measured block.



(a) Error-rates

(b) ITR

(c) ITR_{LEV}

Fig. 5. Comparison of Error-Rates, ITR, and ITR_{LEV}

Fig. 5(a) shows the error-rates and averages (red line) of 150 blocks according to the P/E cycles. In this simulation the error-rates signify the ratio of the number of errors in the size of a page, and represent the highest value among the 64 pages of each block. It was observed that the error-rates deviation and sudden changes in error-rates flow increase as the number of P/E cycles increases. Fig. 5(b) shows ITRs of each block and the average when the data of the measured error-rates are decoded through the QC-LDPC codes we applied. The flow of the ITRs is similar to the flow of the error-rates. This indicates that the aging degree of blocks can be measured by ITRs without calculating the error-rates. However, there are cases where ITRs measured are excessively different when compared with its averages. Fig. 5(c) is obtained by applying ITR_{LEV} in order to reduce sudden

variations of *ITR*s. Compared with fig. 5(b), fig. 5(c) shows that ITR_{LEV} can prevent the instantaneous changes of values, which means that a more accurate tracking of the aging degree can be performed by using ITR_{LEV}.

2. Wear-Leveling Simulations

Our wear-leveling simulations were conducted for three cases: 1) P/E cycles only, 2) the combination of Error-Rates and P/E cycles (50:50), and 3) $AGING_{DEG}$ (a = 0.5). The simulation procedure for wear-leveling is shown in fig. 6. Based on P/E cycles, the combination of Error-Rates and P/E cycles, or $AGING_{DEG}$, the youngest block (Youngest Block) is selected. If there are one or more blocks with the same aging degree the lower indexed block is selected. For the selected block and its error-rate, the decoding operation is performed and *ITR* is obtained. The new ITR_{LEV} is updated by comparing the *ITR* obtained in the decoding process with the previous ITR_{LEV} of the corresponding block. If the decoding fails, the block is managed in a bad state and is excluded from subsequent wear-leveling processes. When a bad block is recorded in the bad block list, the Accumulated P/E Cycles, which is the sum of the number of P/E cycles of all blocks up to the current P/E cycle, is updated. The wear-leveling process of each simulation is repeated until all blocks are in bad state, that is, until decoding failures occur for all blocks.

To compare the results of applying the cases to the wear-leveling, fig. 7 depicts the Error-Rate of the block selected in each accumulated P/E cycle as a graph until all blocks are in bad state, and fig. 8 shows the decoding fail timings of each block. Since a block is selected per P/E cycle, the number of accumulated P/E cycles can be interpreted as a period of usage of a flash device made up of 150 blocks. Through the deviation of error-rates and distribution of decoding failure time timings, it can be confirmed that wear-leveling using the proposed scheme can more uniformly equalize the aging degree of all blocks and maximize the entire lifespan of the blocks. In particular, in the later portion of accumulated P/E cycles it can be seen that the deviation of error-rates in the proposed scheme is relatively low. This indicates that the proposed scheme is clearly effective for performing overall wear-leveling, even in high aging deviation environments.

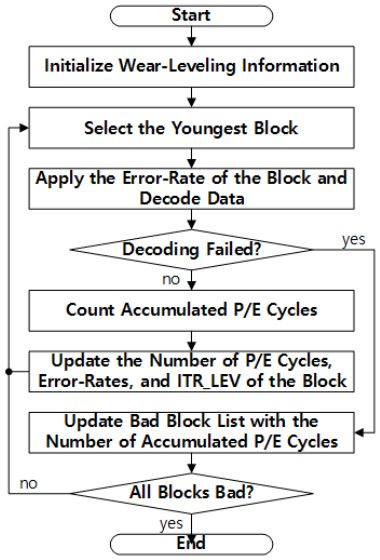
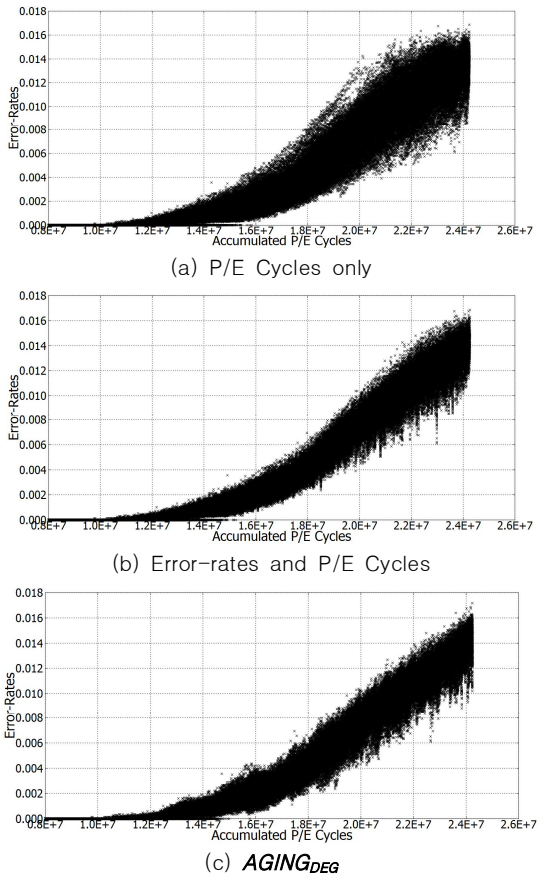


Fig. 6. The Wear-Leveling Simulation Procedure

Table 1 summarizes the averages (Average), minimum values (Min), and deviations (Standard Deviation) of Accumulated P/E Cycles for each case and provides estimates of the amount of additional information bits needed for aging monitoring. By using $AGING_{DEG}$, he average lifespan of blocks became longer and the time timings when blocks became bad blocks were delayed. This means that the proposed scheme-based wear-leveling allowed younger blocks to operate than in other cases, which is proven by the low Standard Deviation. The amount of information needed to store the error-rates is 9 bits per block but ITR_{LEV} requires 5 bits, theoretically. Since general flash memory devices have many more blocks, the difference in the amount of additional information bits will be larger.

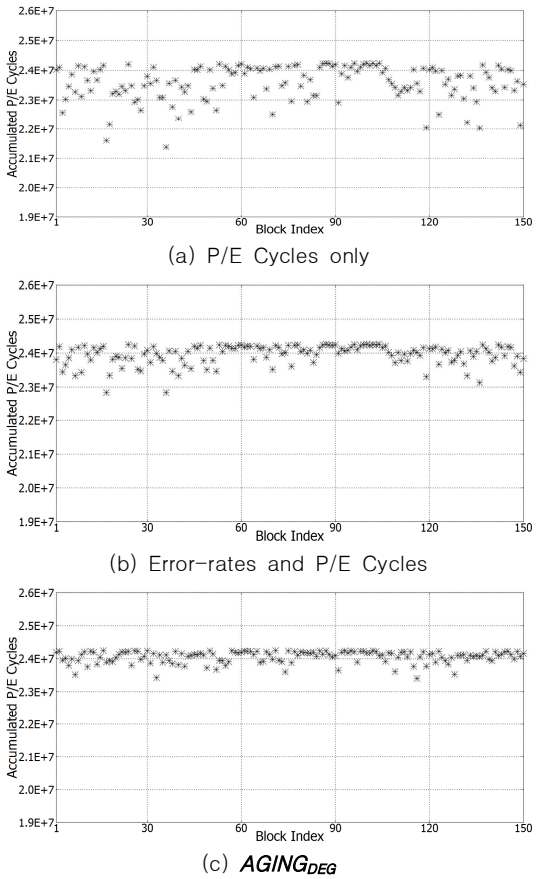


(a) P/E Cycles only

(b) Error-rates and P/E Cycles

(c) $AGING_{DEG}$

Fig. 7. Error-Rates of the Selected Blocks



(a) P/E Cycles only

(b) Error-rates and P/E Cycles

(c) $AGING_{DEG}$

Fig. 8. Decoding Fail Timings

Table 1. Comparison of Decoding Failure Timings and Amount of Additional Information

	Average	Min.	Standard Deviation	Additional Information
P/E only (criterion)	$\fallingdotseq 2.358 \cdot 10^7$	$\fallingdotseq 2.138 \cdot 10^7$	$\fallingdotseq 5.907 \cdot 10^5$	-
Error-Rates and P/E	$\fallingdotseq 2.396 \cdot 10^7$ (+1.61%)	$\fallingdotseq 2.284 \cdot 10^7$ (+6.82%)	$\fallingdotseq 2.907 \cdot 10^5$ (-50.79%)	9·150 (bits)
$AGING_{DEG}$	$\fallingdotseq 2.406 \cdot 10^7$ (+2.05%)	$\fallingdotseq 2.396 \cdot 10^7$ (+9.84%)	$\fallingdotseq 1.824 \cdot 10^5$ (-69.12%)	5·150 (bits)

V. Conclusions

In this paper, we proposed an aging measurement scheme for wear-leveling of flash memories by using the number of LDPC decoding iterations. By carrying out the wear-leveling simulations, the effectiveness was shown in terms of overall lifespan extension of memory blocks and the amount of information to be managed. There is no additional circuit overhead because the proposed scheme gathers aging information from the ECC decoding process which is essentially conducted in data read operation. We expect that our proposed scheme can be utilized for bad block management and over-provisioning to predict and replace bad blocks in advance. In future work, we will apply the proposed scheme to a page based wear-leveling. Even though a block is treated as a bad block, some pages may be still available. By measuring the aging of each page, finding out available pages in bad blocks, and creating logical blocks with the pages, we will be able to increase the storage space and the durability.

ACKNOWLEDGEMENT

This work was supported by University Accounting of Hanbat National University, South Korea.

REFERENCES

[1] R. Micheloni, A. Marelli, and R. Ravasio, "Error Correction Codes for Non-Volatile Memories," *Springer*, 2008.

[2] J. C. Moreira and P. G. Farrell, "Essentials of Error-Control Coding," *John Wiley and Sons*, 2006.

[3] R. G. Gallager, "Low Density Parity Check Codes," I*RE Transactions on Information Theory*, vol. 8, no. 1, pp. 21-28, Jan. 1962.

[4] 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*, vol. 58, no. 2, pp. 429-439, Feb. 2011.

[5] 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.

[6] Y. Cai, et al, "Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives," *Proceedings of the IEEE*, vol. 105, no. 9, pp. 1666-1704, Sep. 2017.

[7] R. Bez, et al, "Introduction to Flash Memory," *Proceedings of the IEEE*, vol. 91, no. 4, pp. 489-502, Apr. 2003.

[8] N. Mielke, et al, "Bit Error Rate in NAND Flash Memories," *2008 IEEE International Reliability Physics Symposium*, pp. 9-19, May. 2008.

[9] L.-P. Chang, "On Efficient Wear Leveling for Large-Scale Flash-Memory Storage Systems," *Proceedings of the 2007 ACM symposium on Applied computing*, pp 1126-1130, Mar. 2007.

[10] Y.-H. Chang, J.-W. Hsieh, and T.-W. Kuo, "Endurance Enhancement of Flash-Memory Storage, Systems: An Efficient Static Wear Leveling Design," *2007 44th ACM/IEEE Design Automation Conference*, pp. 212-217, Jun. 2007.

[11] M. Murugan and D.H.C. Du, "Rejuvenator: A Static Wear Leveling Algorithm for NAND Flash Memory with Minimized Overhead," *2011 IEEE 27th Symposium on Mass Storage Systems and Technologies* (MSST), May 2011.

[12] Y. Pan, G. Dong, and T. Zhang, "Error Rate-Based Wear-Leveling for Nand Flash Memory at Highly Scaled Technology Nodes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 7, pp. 1350-1354, Jul. 2013.

[13] B. Peleato, et al, "BER-Based Wear Leveling and Bad Block Management for NAND Flash," *2015 IEEE International Conference on Communications (ICC)*, pp. 295-300, Jun. 2015.

[14] X. Shi, et al, "Program Error Rate-based Wear Leveling for NAND Flash Memory," *2018 Design, Automation and Test in Europe Conference & Exhibition (DATE)*, pp. 1241-1246, Mar. 2018.

[15] M. Karimi and A. H. Banihashemi, "On Characterization of Elementary Trapping Sets of Variable-Regular LDPC codes," *IEEE Transactions on Information Theory*, vol. 60, no. 9, pp. 5188-5203, Sep. 2014.

[16] S. Laendner and O. Milenkovic, "LDPC Codes Based on Latin Squares: Cycle Structure, Stopping Set, and Trapping Set Analysis," *IEEE Transactions on Communications*, vol. 55, no. 2, pp. 303-312, Feb. 2007.

[17] M. P. C. Fossorier, "Quasi-Cyclic Low-Density Parity-Check Codes from Circulant Permutation Matrices," *IEEE Transactions on Information Theory*, vol. 50, no. 8, pp. 1788-1793, Aug. 2004.

[18] C. Yoon, et al, "Arbitrary Bit Generation and Correction Technique for Encoding QC-LDPC Codes with Dual-Daigonal Parity Structure," 2007 *IEEE Wireless Communications and Networking Conference*, pp. 663-667, Mar. 2007.

[19] J. Chen, et al, "Reduced-Complexity Decoding of LDPC Codes," *IEEE Transactions on Communications*, vol. 53, no. 8, pp. 1288-1299, Aug. 2005.

Authors

Taegeun Kang received the B.S. and M.S. degrees in the Department of Computer Engineering from Hanbat National University, Daejeon, Korea, in 2013 and 2015, respectively. He is currently working toward the Ph. D. degree at Hanbat National University. His research interests include error correction codes (ECC), design-for-security (DfS), and circuit aging monitoring.

Hyunbean Yi received the B.S., M.S. and Ph. D. degrees in Computer Science and Engineering from Hanyang University, Korea, in 2001, 2003, and 2007, respectively. He was with Korea Electronics Technology Institute (KETI) from 2002 to 2007. He had been a Research Scholar at the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, USA from 2007 to 2009. He had been a Postdoctoral Researcher at the Graduate School of Information Science, Nara Institute of Science and Technology (NAIST), Japan from 2009 to 2011. Currently, he is a Professor in the Department of Computer Engineering/Graduate School of Information & Communications, Hanbat National University, South Korea. His research interests include design-for-testability (DfT), design-for-security (DfS), and circuit aging monitoring.