FPGA-Based Failure Mode Testing and Analysis for MLC NAND Flash Memory

Meng Zhang, Fei Wu*, He Huang, Qian Xia, Jian Zhou and Changsheng Xie
Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan, China
*Corresponding author:{Fei Wu, wufei@hust.edu.cn}
{zgmeng, wufei}@hust.edu.cn, hhmagic@126.com.
CoffeeIris@163.com, i@janzhou.org, cs_xie@mail.hust.edu.cn

***Abstract*—With the improvement of flash memory storage density, data reliability and flash lifetime are decreased. Error correction codes (ECC) and error management schemes can boost both reliability and lifetime. However, in order to develop effective fault tolerance algorithms and management solutions, it is very necessary to have a more profound understanding of failure modes of flash memory. To enable such understanding, we design an experimental platform and scheme to clearly investigate flash failure modes. This paper examines various failure modes occurring at 2x-nm MLC NAND flash technologies, such as page allocation scheme-based program interference (PASBPI) errors (i.e., different page allocation schemes mean data can be programmed into flash pages in different ways, which can lead to different program interference errors), write errors of the least significant bit (LSB) and the most significant bit (MSB) and different data pattern-based read interference errors (i.e., different data values programmed into flash pages can cause differential read interference errors). We analyze these observed failure modes and explain why they exist. We hope it is helpful to understand these discovered failure modes to propose effective fault tolerance and error management algorithms.**

I. Introduction

Solid-State Disks (SSDs) with the advantages of high performance, large storage capacity, shock resistance, and non-volatility are widely used in various computer storage systems [17]. SSDs, using flash memory as the storage medium [9], are gaining the capacity as flash cell storage density increases. However, data reliability and flash lifetime are lowered [15]. Error correction codes and error management schemes are most promising technologies to improve both reliability and lifetime [2] [3] [16], but proposing effective fault tolerance algorithms and error management schemes requires a deep understanding of failure patterns of multi-level cell (MLC) NAND flash.

There are some existing works regarding to flash failure patterns. [4] studied how NAND flash errors are affected by the location and order of programmed cells, and data values to be programmed into flash cells. Specially, they characterized the impacts of the order of programmed cells on program interference errors from the threshold voltage shift point. In this paper, we explore the influences of different page allocation schemes (i.e., data can be programmed into flash pages in different ways when using different page allocation schemes) on program interference errors by counting error bits and error pages gained according to the built FPGA-based testing platform. From the large number of collected data, we analyze the sensitivity of program interference errors to page allocation schemes when data is programmed into flash pages. [1] characterized and analyzed the effects of page locations on program interference errors. It is concluded that bit error rates caused by the program interference for even pages and bottom pages are higher and program interference errors have the characteristic of location-dependency. In this paper, we investigate write errors of the least significant bits (LSB) and the most significant bits (MSB). The LSB and MSB write operations are finished in two distinct rounds. In the first round, the LSB is written and the MSB is written in the second round. Moreover, the MSB write process is dependent on the programmed LSB state. Once the LSB is experiencing write errors, it is more likely to cause write errors in the MSB. [11] proposed an read disturb error recovery mechanism by dynamically adjusting the pass-through voltage to mitigate read disturb errors. In this paper, we explore read interference errors by using different data patterns (i.e., different data values programmed into flash pages). We aim to observe the changes of read interference errors as different data patterns are utilized. Different data values programmed into flash pages have differential effects on read interference errors.

In order to fully understand failure modes of MLC NAND flash, we develop a FPGA-based testing platform to investigate flash error characteristics. This paper examines various failure modes occurring at 2x-nm MLC NAND flash technologies by using the built FPGA-based testing platform, such as page allocation scheme-based program interference errors, write errors of the LSB and the MSB, and read interference errors based upon different data patterns. We collect the large number of failure data based on the testing platform and analyze the experiment results, then discuss the causes of these failure modes. The page allocation scheme-based program interference (PASBPI) errors show strong allocation scheme-dependency. Besides, the write errors of the LSB and the MSB indicate asymmetric error characteristics. Finally, the read interference errors show differences when using different data patterns. These discovered failure modes are conducive to developing effective error tolerance algorithms and error management schemes.

The major contributions and new observations of this paper beyond previous works are as follows:

- We elaborately analyze flash failure behavior modes including the PASBPI errors, the write errors of the LSB and the MSB and the different data pattern-based read

978-3-9815370-8-6/17/$31.00 ©2017 IEEE

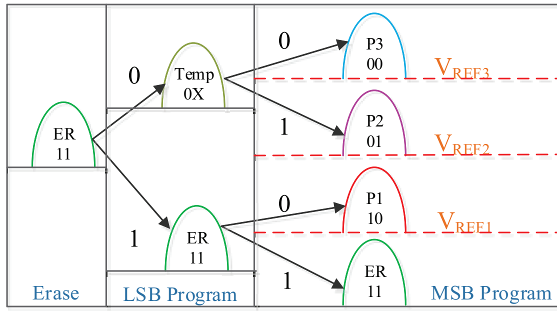

Fig. 1. Read and program processes of MLC NAND flash cell. The program processes of MLC are divided into two rounds. In the first round, the LSB is programmed and the MSB is programmed in the second round. The MSB program relies on the state of the LSB.

interference errors in MLC NAND flash, which have not been fully understood. To promote our research, we design a FPGA-based testing platform to further investigate the error characteristics.

- In order to quantitatively study the PASBPI errors, we collect the large number of data to count error bits and error pages. From the gathered data, the error bits and error pages show larger differences when the data is programmed into flash pages by using different page allocation schemes. It indicates that program interference errors have high correlation to page allocation schemes.
- Motivated by the write mechanisms of the LSB and the MSB, we investigate the write errors of the LSB and the MSB. Experimental results show that once the LSB write occurs to errors, it is able to result in errors in the MSB write process. Besides, we also explore the different data pattern-based read interference errors and the read interference errors show some differences for different data values programmed into flash pages. Finally, we analyze these discovered failure modes and explain why they exist.

The remainder of this paper is organized as follows. Section II introduces related background about NAND flash and related work. Section III describes NAND flash failure behavior modes including the PASBPI errors, the write errors of the LSB and the MSB, and the different data pattern-based read interference errors. The implementation methodology is gived in section IV. Experiment results are characterized in section V. Finally, in Section VI, the concluding remarks are made.

II. Background and Related Work

In this section, we introduce the background knowledge about the read and program mechanisms of MLC NAND flash and then present the related work.

A. Read and Program Processes

Each MLC cell stores 2 bits belonging to different pages, the left bit and the right bit from same cell are called the least significant bit (LSB) and the most significant bit (MSB) [1], respectively. Read and program processes are briefly introduced in Fig. 1.

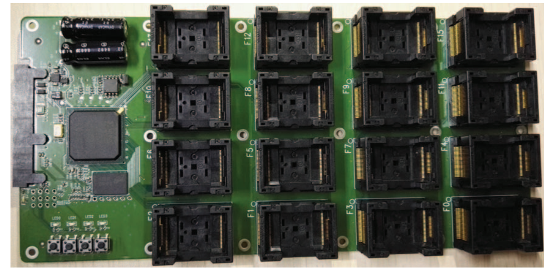

Fig. 2. FPGA-based testing platform.

- ***Program:*** The MLC program experiences two rounds [10]. If logic data "1" is programmed into the LSB page finished in the first round, the cell remains in the erased state ER. Otherwise, the cell is charged to a temporary $temp$ state that is finally determined when the MSB program is finished. In the second round, the MSB is programmed according to the completed LSB state. When the LSB lies in the ER state, if logic data "1" is to be programmed to the MSB, the ER state is unchanged. Otherwise, the ER state is charged to the $P1$ state. When the LSB locates into the $temp$ state, if bit "1" is written, the $temp$ state is charged to the $P2$ state. Otherwise, the $temp$ state is charged to the $P3$ state.
- ***Read:*** The data value stored in flash cells is determined by the preseted threshold voltage ranges [14]. The read operations of MLC NAND flash have two processes. Reference voltage V_{REF2} is firstly applied to read the LSB in the first process. If the cell situates in a conduction state, it means that the threshold voltage of the read cell is less than the reference voltage V_{REF2}. It indicates that the stored bit value of the LSB is "1", otherwise "0". The process of the MSB read in the second process is considered to be more complex than that of the LSB. Firstly, V_{REF2} is used to judge the current LSB state. And then V_{REF1} or V_{REF3} is applied according to the judged LSB state. If $V_{REF1} < V_{TH} < V_{REF2}$ or $V_{TH} > V_{REF3}$, where V_{TH} represents the threshold voltage read from the cell. It demonstrates that the read data is "0". If $V_{TH} < V_{REF1}$ or $V_{REF2} < V_{TH} < V_{REF3}$, it implies that the read data is "1".

B. Related Work

There are some previous works to study error patterns of NAND flash memory, and various techniques and schemes are proposed to ensure data reliability and improve flash lifetime according to the investigated error patterns. Tseng *et al.* [9] designed a hardware platform to investigate bit error characteristics of flash when the flash power is cut off during operations. Cai *et al.* [1] constructed a hardware platform based upon FPGA to study error patterns of MLC NAND flash that are divided into retention errors, program interference errors, read interference errors and erase errors. Meza *et al.* [5] collected most of data located on SSDs in the FaceBook data center and failure modes of SSDs are analyzed through the collected data. Besides, Cai *et al.* [4] constructed a program interference prediction model and

proposed a scheme to improve flash lifetime according to the built model. Yaakobi *et al* [3] proposed the corresponding error correction technique by observing the error behavior of TLC NAND flash and compared the advantages of the technology with the LDPC and BCH codes. Zhao *et al*. [2] proposed three techniques, such as pre-detection, step detection and data crossing, according to the characteristics of flash memory chips and LDPC encoding/decoding. Zhao *et al*. [10] developed two techniques (i.e., intra-cell data placement interleaving and intra-cell data-dependency aware min-sum decoding) to improve decoding throughput depended on the characteristics of unbalance bit errors and data-dependency in flash cells. Cai *et al*. [11] proposed some error recovery techniques to minimize read disturb errors.

III. NAND Flash Failure Behavior Patterns

When data is written into flash pages, different page allocation schemes can result in differential program interference errors. Different page allocation schemes mean that data can be programmed into flash pages in various ways. In this paper, we explore the relationship between different allocation schemes of physical pages (i.e., in-random-order allocation, in-page-order allocation and in-wordline-order allocation) and program interference errors. We design three different "IO Patterns" to write pseudo-random data into flash blocks. Concretely, three blocks are randomly selected as $B1$, $B2$ and $B3$. The in-random-order allocation scheme is adopted to randomly program data into flash pages in $B1$, each page is programmed only once and the LSB page is preferred to be programmed to the corresponding MSB page. In $B2$, the in-page-order allocation scheme is used and each page in the block is programmed in turn according to page number sequences. The LSB page and the MSB page in B_3 are programmed in turn according to wordline number sequences.

Besides, the write errors of the LSB and the MSB are studied by using two different program schemes. We randomly select four blocks as $C1$, $C2$, $C3$ and $C4$, respectively, where $C1$ and $C2$ belong to the same group $G1$, $C3$ and $C4$ are classified as the another group $G2$. In $G1$, LSB pages are written in turn according to page number sequences. At the same time, bit errors are detected when programming the LSB pages. Similarly, MSB pages in $G2$ are written in turn according to the page number sequences and bit errors are also counted. It needs to demonstrate that the wrote data are all randomly generated.

Finally, it is known that when a page is read constantly, it can cause a disturbance to other pages [1] [11]. However, different data patterns programmed into flash pages have differential effects on read interference errors, which have not fully understood. The relationship between read interference errors and data patterns are explored for further understanding the characteristics of read interference errors.

IV. Implementation Methodology

In this section, we introduce experimental environment and the relevant implementation scheme.

Algorithm 1 In the testing process, two data structures of "Map" and "ECCTable" are constructed to store address sequences and bit error information involving bit error numbers and locations. Address sequences stored in the "ECCTable" are same with the addresses in the "Map". The "IO Pattern" provides the specified page address sequences for the programmed data, stored in the "Map" and the "ECCTable". The "Data Pattern" generates data to be written into flash pages. The process involves two stages of data programmed and read to check bit errors.

Input: Program patterns (i.e., the program order and mechanism)

Output: Bit error information of each page.

1: Generate "Data Pattern" for the current program pattern and initialize the data pool used to store the generated data.
2: Produce "IO Pattern" for the current program pattern, each page address is stored as an element in the "Map", the total number of physical page addresses recorded in the "Map" is denoted as "TotalMapEntries"
3: Trigger the program operation and set the traversal index $i = 0$
4: According to the index i, take out the physical page address Map$[i]$ from the "Map"
5: The data corresponding to the physical page is taken out from the data pool to program the generated data
6: Implement the "PageWrite" operation on flash
7: **for** i from 1 to "TotalMapEntries" **do**
8:     **if** $i <$"TotalMapEntries" **then**
9:         Repeat the step 4 to the step 8
10:     **else**
11:         Terminate the program operation and implement the next step operation
12:     **end if**
13: **end for**
14: Start the bit error check operation and set the traversal index $j = 0$
15: According to the index j, take out the physical page address Map$[j]$ from the "Map"
16: Implement the "PageReadBypass" operation on physical pages and read the interfered raw data
17: Take out the corresponding data from the data pool
18: Compare the read raw data with the data taken from the data pool, record the bit error information and create the corresponding "ECCEntry", the recorded information is stored in the "ECCTable"
19: **for** j from 1 to "TotalMapEntries" **do**
20:     **if** $j <$"TotalMapEntries" **then**
21:         Repeat the step 15 to the step 20
22:     **else**
23:         Terminate the operation on the error detection and implement the next step operation
24:     **end if**
25: **end for**
26: Traverse the "ECCTable" to obtain the bit error information of each page and write them into the log file

Fig. 3. The number of bit errors in the error pages under different page allocation schemes.

A. Experimental Environment

We build a flash testing platform (shown in Fig. 2) with FPGA-based PCBA and window-based applications. The PCBA equipped with 16 TSSOP48 sockets connects to the Xilinx Spartan3E FPGA, so we can mount 16 NAND flash chips to the PCBA and test them simultaneously. The applications connect with the hardware platform via a SATA cable. We design a standard SATA device which complys with the ATA command set in the FPGA. The designed FPGA testing platform consists of an embedded CPU MicroBlaze, a DDR2 controller, a ATA device IP, a 32KB Dual-Port RAM buffer, and 4 flash controllers of supporting 4-way interleave. The embedded code running in MicroBlaze acts as an agent of host applications, implementing the ATA protocol, managing the flash read/write and data transfer and reporting status to the host applications. In the experiment, we choose 2x-nm 2 bits MLC NAND flash as the testing object.

B. Experimental Scheme

The experimental process mainly includes two stages, which are concretely described in Algorithm 1. The “PagereadBypass” operation means that we bypass the error correction code module to read the raw data stored in flash pages and check bit errors.

V. Experimental Results

In this section, we characterize and analyze the measured PASBPI errors, the write errors of the LSB and MSB pages and the read interference errors based upon different data patterns.

A. The Measured PASBPI Errors

Three different “IO Patterns” are used to program data into flash pages to explore the relationship between allocation schemes of physical pages and program interference errors. Experimental results are listed in Table I. Fig. 3 shows the distribution of bit errors in all error pages under different page allocation schemes.

Table I shows that a number of error pages and error bits are much fewer when the in-page-order allocation scheme is used, compared with the in-random-order and the in-wordline-order

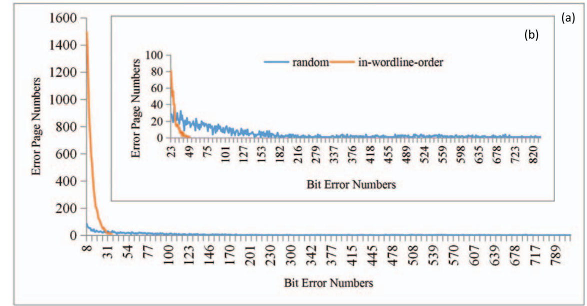

Fig. 4. Distribution of bit errors when the in-random-order and the in-wordline-order schemes are adopted.

TABLE I
THE RELATIONSHIP BETWEEN ALLOCATION SCHEMES OF PHYSICAL PAGES AND PROGRAM INTERFERENCE ERRORS

Allocation Schemes	random	in-page-order	in-wordline-order
Written Data Amount	100MB	100MB	100MB
Written Page Amount	25600	25600	25600
Data Patterns	random	random	random
Raw Bit Error Numbers	430310	1244	152748
Error Page Numbers	7702	1166	23120

schemes (i.e., same random data is written into flash blocks). Experimental results (see Fig. 3) show that a number of error bits in the error pages are the maximum of 3 when the in-page-order allocation scheme is adopted. The maximum number of error bits in the error pages is 50 when using the in-wordline-order allocation scheme, while the maximum number of error bits in the error pages is as high as 920 and most of error bit numbers in error pages are 1 and more than or equal 8 when making use of the in-random-order allocation scheme. When a number of error bits are greater than or equal to 8, the distribution of bit errors in the error pages is shown in Fig. 4 (a) under cases of the in-random-order allocation and the in-wordline-order allocation, while the distribution of bit errors in the error pages is shown in Fig. 4 (b) when a number of error bits are greater than or equal to 23.

The reasons for the above phenomena are mainly due to different degrees of the program interference on flash cells under different physical page allocation schemes, here only to consider the program interference between adjacent wordlines. In this experiment, we select 2x-nm 2 bits MLC NAND flash with

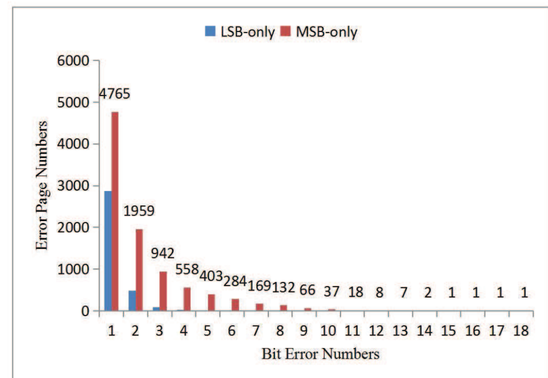

Fig. 5. Write modes and corresponding bit errors.

the all bitline architecture in each block and two flash pages (i.e., the LSB and MSB pages) locate on the same wordline. For the wordline N, it corresponds to page numbers $(2N-1)$ and $(2N+2)$, while its adjacent wordline $(N-1)$ corresponds to page numbers $(2N-3)$ and $2N$, another adjacent wordline $(N+1)$ corresponds to page numbers $(2N+1)$ and $(2N+4)$. Different program sequences of these six pages have differential effects on pages on the wordline N. When utilizing the in-page-order allocation scheme, the program sequence of these six pages mentioned above is $(2N-3) \rightarrow (2N-1) \rightarrow (2N) \rightarrow (2N+1) \rightarrow (2N+2) \rightarrow (2N+4)$, the corresponding wordline program sequence is $(N-1) \rightarrow N \rightarrow (N-1) \rightarrow (N+1) \rightarrow N \rightarrow (N+1)$. After two pages of the wordline N are programmed, page program operations of adjacent wordlines produce the impacts of the program interference on pages on the wordline N. For the in-page-order allocation scheme, only the MSB page program operation of the wordline $(N+1)$ has an impact on pages on the wordline N. It means that the pages on the wordline only undergo the program interference from one page when the in-page-order allocation scheme is utilized. When adopting the in-wordline-order allocation scheme, the program order of these six pages mentioned above is $(2N-3) \rightarrow 2N \rightarrow (2N-1) \rightarrow (2N+2) \rightarrow (2N+1) \rightarrow (2N+4)$, and the corresponding wordline program order is $(N-1) \rightarrow (N-1) \rightarrow N \rightarrow N \rightarrow (N+1) \rightarrow (N+1)$. It indicates that program operations of the LSB and MSB pages have the impacts of the program interference on pages on the wordline N. In other words, when we exploit the in-wordline-order allocation scheme, pages on the wordline can be affected by two pages. Compared with the in-page-order allocation scheme, pages on the wordline are subject to the program interference from more pages so that it is easier to occur to page errors and bit errors. For the in-random-order allocation scheme, the page program order is completely uncertain, so pages on the wordline N may not be affected by the program interference (i.e., the LSB and MSB pages locating on the wordline N are programmed after the other four pages are programmed), may also be interfered by four pages (i.e., the LSB and MSB pages locating on the wordline N are programmed before in the other four pages are programmed), may also be influenced by two or three pages. There is a large probability of being error pages since pages on the wordline N experience the program interference incurred by more pages. When the program interference is more serious, the threshold voltage of multiple cells may shift at the same time, leading to a large number of bit errors and page errors.

B. *The Write Errors of The LSB and The MSB*

Two kinds of "IO Patterns" are exploited to explore failure modes of NAND flash when the LSB and MSB pages are written respectively. Experimental results are listed in Table II. There are more error pages and bit errors when the MSB page is written. Fig. 5 shows that there are fewer errors in the LSB page when the page is written, and bit errors occurring in each error page are at most 6. The maximum number of bit errors appearing in the error pages is 18 when the MSB page is written. A number of errors in the MSB page are higher than that of the LSB page. According to the program processes of MLC NAND flash cells (shown in Fig. 1), we can know that specific operations on the MSB program depend on the current state of the LSB. If errors occur in the LSB program process, the cell situates in a non-expected state, then according to the LSB state to determine the value of the MSB, there is a large probability of being errors. On the other hand, the cell threshold voltage varies with the change of charge quantities. Once the amount of charges changes to a certain extent, the cell threshold voltage is transferred to other states. As a result, according to the threshold voltage states to judge data stored in flash cells, it is able to bring about errors. Furthermore, MLC NAND flash scaled down to smaller process technology nodes makes spacing area between adjacent cells become much narrower so that the cell threshold voltage is more prone to migrating. The cell has only two states (i.e., ER and $temp$ states) when the LSB is written, so the distance between such two states is relatively large and the probability of threshold voltage shift is much smaller. Conversely, when programming the MSB page, the cell has four states, thus the distance between these states is relatively small, the possibility of threshold voltage shift is much larger so that there is easier to occur to bit errors. It requires to illustrate that bit errors can not be completely counted when exploiting the in-random-order allocation scheme. Therefore, we only consider the in-page-order and the in-wordline-order allocation schemes to test a flash block selected randomly. Under these two allocation schemes, a number of the error pages appearing in the LSB page and the MSB page are counted, as shown in Table III. When the in-page-order allocation scheme is adopted, there is a total of 10 error pages, including 1 LSB page and 9 MSB pages. When utilizing the in-wordline-order allocation scheme, there is a total of 47 error pages, involving 7 pages occurring to 2 bit errors, 11 pages and 29 pages happening to 1 bit error. Based on the collected data and above analysis, it is concluded that MSB pages become vulnerable to bit errors when writing data into 2 bits MLC NAND flash pages. Need to specify that the conclusion is drawn only on the basis of the selected 2x-nm 2 bits MLC NAND flash in our test.

TABLE II
THE WRITE ERROR OF LSB AND MSB PAGES

Write Object	LSB page	MSB page
The Written Data Amount	100MB	100MB
The Written Page Amount	25600	25600
Data Pattern	random	random
Raw Bit Errors	4265	21142
Error Page Amount	3487	9354

TABLE III
THE WRITE-IN ERROR OF LSB AND MSB PAGES

	in-page-order		in-wordline-order	
	LSB page	MSB page	LSB page	MSB page
2 bit errors	0	0	0	7
1 bit errors	1	9	11	29

C. *Different Data Pattern-Based Read Interference Analysis*

The implemented experiment utilizes three different "Data Patterns" (i.e., all "0" (0x00), all "1" (0xFF) and pseudo-

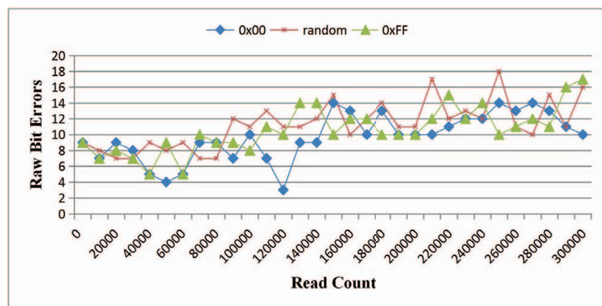

Fig. 6. The relationship between bit error numbers and read times under different data patterns.

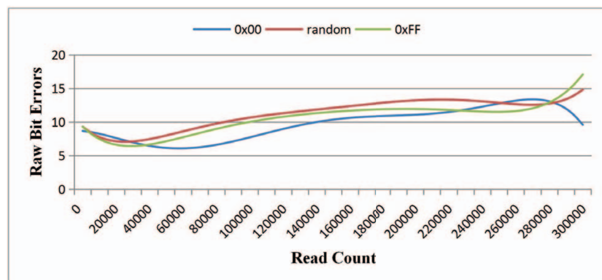

Fig. 7. The change trend of bit errors with the number of read operations under different data patterns.

random data) to investigate read interference error patterns. The relationship between the number of error bits in the block and read operation times is shown in Fig. 6. From the experimental results, it can be seen that a number of bit errors are not strictly increased with the improvement of read operation times, but the overall variation has an upward trend. Overall, when read operation times are no more than "100000", a number of bit errors are substantially lower than "10", and when a number of read operations are more than "200000", the number of bit errors basically exceeds "10". In order to clearly observe the change tendency of bit errors with the number of read operations, the experimental results shown in Fig. 6 are implemented with the higher order curve fitting operation (See the Fig. 7). It can be seen that bit errors are generally improved as read operations increase. However, the corresponding curve with the "Data Pattern" 0x00 is slightly different, when the number of read operations is close to "300000", there is a downward trend due to the effect of retention errors. When the "Data Pattern" 0x00 is written, it is more prone to charge leakage due to the fact that more electrons are injected into the floating gate. During the experiment, read operations are executed continuously, the more the number of execution is, the longer the time is, the cumulative loss of the amount of charges is also more. As a result, the read interference caused by a small amount of electron injections just compensates for the loss of charges, and thus the total number of bit errors is reduced. It is concluded that the read interference errors have the characteristic of data pattern-dependency.

VI. Conclusions

In order to have a more profound understanding of flash failure modes, we construct a hardware testing platform based on FPGA. Various failure modes consisting in state-of-the-art 2x-nm MLC NAND flash are measured, characterized and analyzed by using the built testing environment, such the PASBPI errors, the write errors of the LSB and the MSB, and the different data pattern-based read interference errors. We obtain some valuable observations and analyze the causes of these failure modes mentioned above. Firstly, for the PASBPI errors, adopting different page allocation schemes can lead to differential program interference errors when data is programmed into flash pages. Secondly, the error of the LSB will influence the MSB, it means when there is an error occurring in the LSB, it will lead to the error of the MSB. Finally, different data patterns can cause differential read interference errors. These discovered failure modes help to develop effective error tolerance mechanisms and error management algorithms.

Acknowledgments

This research is sponsored by the National Natural Science Foundation of China under Grant No. 61300047 , No. 61572209 and National Natural Science Foundation of Hubei Province No. 2015CFB315 and the Fundamental Research Funds for the Central Universities No. 2016YXMS019. This work is also supported by Key Laboratory of Data Storage Systems, Ministry of Education.

References

[1] Y. Cai, et al,"Error patterns in MLC NAND flash memory: measurement, characterization, and analysis," in *IEEE/ACM DATE*, 2012.

[2] K. Zhao, et al, "LDPC-in-SSD: making advanced error correction codes work effectively in solid state drives," in *Proceedings of USENIX FAST*, 2013.

[3] E. Yaakobi, et al, "Characterization and error-correcting codes for TLC flash memories," in *IEEE ICNC*, 2012.

[4] Y. Cai, et al, "Program interference in MLC NAND flash memory: characterization, modeling, and mitigation," in *IEEE ICCD*, 2013.

[5] J. Meza, et al, "A large-scale study of flash memory failures in the field," in *ACM SIGMETRICS*, 2015.

[6] D. H. Lee, et al, "Least squares based cell-to-cell interference cancelation technique for multi-level cell NAND flash memory," in *IEEE ICASSP*, 2012.

[7] Y. Cai, et al, "Flash correct-and-refresh: retention-aware error management for increased flash memory lifetime", in *IEEE ICCD*, 2012.

[8] W. Wang et al, "Reducing MLC flash memory retention errors through programming initial step only," in *IEEE MSST*, 2015.

[9] H. W. Tseng, et al, "Understanding the impact of power loss on flash memory," in *ACM DAC*, 2011.

[10] W. Z. Zhao, et al, "Improving Min-sum LDPC decoding throughput by exploiting intra-cell bit error characteristic in MLC NAND flash memory", in *IEEE MSST*, 2014.

[11] Y. Cai, et al, "Read disturb errors in MLC NAND flash memory: characterization, mitigation, and recovery," in *IEEE/IFIP DSN*, 2015.

[12] Y. Cai, et al, "FPGA-based solid-state drive prototyping platform," in *IEEE FCCM*, 2011.

[13] P. Cappelletti, et al, "Failure mechanisms of flash cell in program/erase cycling," in *IEEE IEDM Tech. Dig,* 1994, pp. 291-294.

[14] L. M. Grupp, et al, "Characterizing flash memory: anomalies, observations, and applications," in *IEEE/ACM MICRO*, 2009.

[15] H. Q. Pon, et al, "Reliability issues studied in solid-state drives," *IEEE Memory Workshop (IMW)*, 2014.

[16] E. Yaakobi, et al, "Error characterization and coding schemes for flash memories," *IEEE GLOBECOM Workshops*, 2010.

[17] S. Tanakamaru, et al, "Over-10x-extended-lifetime 76%-reduced-error solid-state drives (SSDs) with error-prediction LDPC architecture and error-recovery scheme," in *IEEE ISSCC,* 2012.