Error Generation for 3D NAND Flash Memory

Weihua Liu, Fei Wu*, Songmiao Meng, Xiang Chen, Changsheng Xie
Wuhan National Laboratory for Optoelectronics, Key Laboratory of Information Storage System,
Engineering Research Center of Data Storage Systems and Technology, Ministry of Education of China.
*Corresponding author: {Fei Wu, wufei@hust.edu.cn}

***Abstract*—Three-dimension (3D) NAND flash memory is the preferred storage component of solid-state drive (SSD) for its high ratio of capacity and cost. Optimizing the reliability of modern SSD needs to test and collect a large amount of real-world error data from 3D NAND flash memory. However, the test costs have surged dozens of times as its capacity increases. It's imperative to reduce the costs of testing denser and high-capacity flash memory. To facilitate it, in this paper, we aim to enable reproducing error data efficiently for 3D NAND flash memory. We use a conditional generative adversarial network (cGAN) to learn the error distribution with multiple interferences and generate diverse error data comparable to the real-world. Evaluation results demonstrate it is feasible and efficient for error generation with cGAN.**

***Index Terms*—Test, Generation, 3D NAND flash memory, Reliability**

I. INTRODUCTION

As three-dimension (3D) NAND flash memory attracts more eyes for its high storage capacity, solid-state drives (SSDs) are becoming the dominant storage choice [1]. The increasing storage capacity of 3D NAND flash memory mainly results from two aspects: (1) more vertically stacked layers. It changes the trend of increasing storage capacity merely depending on lithography evolution. (2) denser storage unit, such as the multi-level cell (MLC), the triple-level cell (TLC) [2], and the quad-level cell (QLC) [3, 4]. However, the growth of storage capacity yields more complicated interferences [5] and the fragile reliability of SSD [6].

3D NAND flash memory introduces the exclusive layer-to-layer interference and layer variation than planar structure, resulting in some new reliability characteristics and problems [5, 7]. Undoubtedly, Optimizing the reliability needs a large amount of real-world data to validate the proposed scheme [8, 9]. It requires comprehensive test experiments to collect error data from various interference scenarios. On the other hand, the costs of testing 3D NAND flash memory raise promptly with the two above-mentioned aspects. More stacked layers and denser storage units increase the operation time dramatically. Programming and reading a flash block need several to dozens of times than before referring to Samsung's V-NAND flash memories [2, 4]. As a result, a prior one-month test will cost several or even dozens of months for a new larger-capacity 3D NAND flash memory, and the test costs will be unacceptably expensive as the capacity continuously increases in the future [3].

Motivated by the gap between the rarity of real-world error data and the getting higher test costs for 3D NAND flash memory, we probe reproducing data to increase the population of real-world data. Two methods can approach this goal: (1) replicating real-world data by several times; (2) modeling the real-world data [6] and giving some noises to generate a large amount of data. However, the former limits the **diversity** of reproduced data, and it may result in unbalanced error correction ability, much like the over-fitting in machine learning. The latter is challenging to guarantee the **authenticity** of the generated data since the modeling usually eliminates the personality of samples. Besides, to exhaust all dimensions to evaluate the authenticity of data is theoretically impractical since some hidden features are still unknown for real-world data.

To reproduce realistic and diverse data comparable to the real-world data, in this paper, we delve into generating realistic enough error data for 3D NAND flash memory. Before that, we make comprehensive preliminary experiments to collect data and characterize the error distribution. We then employ a conditional generative adversarial network (cGAN) to learn and simulate the hidden features of real-world error data, shielding the authenticity and diversity for error generation. Evaluation results show the generated data have expected authenticity and diversity, which are realistic enough to increase the population of real-world error data.

The key contributions of this paper are as follows. First, we propose increasing the population of error data for 3D NAND flash memory by reproducing error data, to combat part of the overhead of traditional high-cost tests. We demonstrate it is feasible and efficient. Second, we give a methodology to generate error data using cGAN for 3D NAND flash memory. Evaluation results show the generated data are diverse and authentic enough. It will dramatically reduce the test costs.

We review the background and related work in the next section. Section III is the preliminaries of error characterization and Section IV gives the methodology of using cGAN to learn and generate the error data. Section V evaluates the authenticity, diversity, and efficiency of the proposed method. We make a conclusion of this paper in Section VI.

II. BACKGROUND AND RELATED WORK

A. Basics of NAND Flash Memory

NAND flash memory distinguishes the stored data by quantifying the number of electrons stashed in a cell (a MOSFET transistor). For example, a TLC cell stores three bits from "000" to "111". A **word-line (WL)** consists of several series-connected cells in a bit-line (BL). All the bits in the specific position (e.g. the leftmost, the middle, or the rightmost) in a WL are structured to a **page**. Hence, a TLC WL has three pages. A flash **block** contains several WLs stacked at different layers, and it can be coarsely considered as a two-dimensional page array.

NAND flash memory has three operations: Erase, Program (Write), and Read. The erase operation sweeps out the electrons from all cells in a block, and the program operation injects electrons to all cells of a WL (or a page).

978-3-9819263-6-1/DATE22/©2022 EDAA

Any operation will disturb (or wear) the target WL and its adjacent WLs, leading to electrons injection or loss. The main considering interferences in system application include Program/Erase (P/E) cycle, data retention (DR) time, and read disturbance (RD).

Besides, 3D NAND flash memory introduces the exclusive layer variation and layer-to-layer interference due to process variation. As a result, pages at different layers exhibit conspicuous error variation. Thus, layer variation gives different pressures to error correction for different layers, and it will be further aggravated as more layers are stacked in a block, such as 96-layer, 144-layer, and even more. The continuously increasing number of layers also exacerbates the DR ability and RD [5].

B. Related Work

The study of errors in NAND flash memory usually contains characterization [5], modeling [10], and error correction [8] based on comprehensive test experiments. Mielke et al. [6] investigated the reliability of NAND flash memory and proposed a bit error rate (BER) model with P/E cycle, DR time, and RD. Cai and Luo et al. systematically characterized the errors with various interferences, and they summarized the reliability problems of NAND flash memory [7]. Liu et al. characterized the reliability of 3D CT NAND flash memory and gave several non-trivial observations [5]. Also, Wu et al. proposed several error-aware error correction schemes for 3D NAND flash memory [8, 9]. Papandreou et al. characterized and analyzed the bit errors in 3D TLC NAND flash memory [11].

At the same time, we observed that the test costs of P/E and read cycling have increased several times as the block capacity grows. Hsieh et al. developed a Monte Carlo simulation method to predict the memory product window and generate the threshold voltage distribution of large-density 3D NAND flash memory, based on the device parameters collected in a small-array test element group [12]. However, we have not seen that generating error data directly to reduce the surge of data acquisition costs for 3D NAND flash memory. Hence, we delve into error generation in this paper.

C. Motivation

The necessity of reproducing errors data is to reduce the extremely high costs of collecting a large amount of error data from 3D NAND flash memory. The areal density has increased by more than 8X since the introduction of 3D NAND technology in 2014 [3]. Compared with an early 24-layer MLC flash, a 128-layer QLC flash block increases a block size by more than 10X, and meanwhile, the average latencies of programming and read a page will also increase by more than 2X, theoretically.

Taking M1 [2] and M4 [4] as examples, a 92-layer QLC (M4) will spend 2944 or 648 ms to program or read a block, which are nearly 11X and 9X than M1, a 32-layer TLC, consuming 268 or 69 ms to program or read a block. As a result, the P/E cycling and RD test will be delayed directly, and the time costs of the data retention test will also increase. The future scaling target would achieve another 50X areal density scaling in the next decade [3], and the test costs thus will further grow by dozens of times.

III. Preliminary

A. Test Experiment

We take a 3D TLC NAND flash memory from Intel as an example to characterize the error distribution and illustrate the proposed error generation methodology. A flash block of this model has 2304 physical 16KB-pages. We count the number of bit errors in each page.

The test experiments contain two major experiments: (1) cycling test; (2) synthetic test with multiple interferences; In the cycling test, a batch of randomly selected fresh blocks will be executed P/E operations continuously and counting the errors every 100-cycle until reporting a erase error. In the synthetic test, a batch of randomly selected fresh blocks will be first divided into several groups, each group is set to different P/E cycle target, such as $\{1, 1000, 2000, \ldots, 10000\}$. The cycling process then is issued towards the target P/E cycles for each group. After that, these blocks will be launched the DR experiments ((high-temperature accelerated 7-, 30-, ..., 180- day) and RD experiments (1-, 1000-, ..., 5000- read count) and collect the error data. We take reading a full-block once as one read count.

We spend over three and four months to test and collect the error data from more than 3500 and 600 blocks for the cycling and synthetic tests, respectively. We simulate four representative scenarios of SSD through different interference permutations of test experiments: (1) freshly used; (2) frequent write (hot-write); (3) frequent read (hot-read); (4) long-term storage (cold- write and read).

B. Error Characterization

Fig. 1 is the gray image transformed from the error distribution in four scenarios. The interferences of (P/E cycle, RD, DR time) is entitled in each subfigure. Each subfigure exhibits the error distributions of 100 blocks, and each block contains 2304 pages. The total errors of 230400 pages are shaped to a (2304×100) array. The gray pixel value, $P(i)$, equals $E(i)/Max(E)$, where $E(i)$ is the number of bit errors of the i-th page in a block, and $Max(E)$ is the maximum $E(i)$ in this block. Note that, Fig. 1 only records the relative error distribution of a block. The gray value to bit error value cannot be compared directly crossing subfigures.

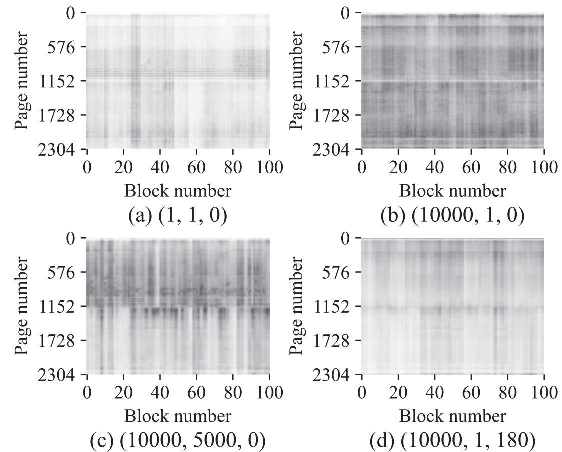


Fig. 1: The relative error distribution.

We can observe that the errors in every block show a similar inter-page distribution in each subfigure. This is because

every block has a similar layer variation. The similarity is an obvious feature and the basis of learning and generation. Besides, they have different error distributions in various scenarios. Fig. 1 (d) has the maximum $E(i)$ and $Max(E)$ in four subfigures. However, the $P(i)$ in Fig. 1 (d) is dominated by the catastrophic $Max(E)$ from the top several pages (the top black line in Fig. 1 (d)).

At the same time, we can also observe that every block has different error values ($P(i)$) than others in the same scenario due to block variation. Block variation is the hidden features and the premise of diversity. To exhibit the diversity more clearly, we reshape the error distribution in a block from (2304×1) to (48×48) and plot the first four blocks of Fig. 1 (b) and (c) in the top and bottom of Fig. 5a, respectively. Except for the similarity of error distribution, we are impressed by the diversity Fig. 5a indicated. The error values of the same pages in various blocks differ from others, which is the most challenging factor to be modeled with traditional statistical models, and it is the difficulty of generating more realistic and diverse errors.

IV. Methodology

A. cGAN for Error Generation

GAN has been used in image reconstruction, image style transformation, etc. It can learn hidden features by mapping planar images into high dimensions. It consists of two networks, a generator and a discriminator. The generator generates candidate data and feeds them to the discriminator. The discriminator learns features from real data to discriminate if the fed data (real and generated data) are real. The generator tries to deceive the discriminator, and the discriminator adversarially struggles to detect the generated data are fake. The confrontation and competition between the generator and discriminator continue to intensify until reaching Nash equilibrium. However, the original GAN cannot generate results quantitatively. Hence, we select and modify the cGAN as the model to generate the specific result.

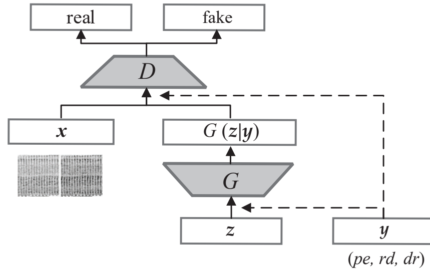


Fig. 2: The diagram of cGAN model for error generation.

As Fig. 2 explains, we consider the interferences as the conditions since a strong relationship between the error distribution and the conditions can be observed from Fig. 1. $\boldsymbol{x}$ is the real error data of a block with the interference condition $\boldsymbol{y}$, where P/E cycle = pe, read count = rd, and DR time = dr, denoted as $\boldsymbol{y} = (pe, rd, dr)$. D and G are the discriminator and generator respectively. $\boldsymbol{z}$ is a random-number vector initiating the diversity of a generation as the noise. cGAN introduces the condition $\boldsymbol{y}$ into training the generator and discriminator to resolve the unconstrained generation of GAN.

The implementations of D and G have many choices, such as multilayer perceptron, convolutional neural networks, and etc. Considering the uncomplicated error distribution and the distinct visual feature indicated in Fig. 1 and Fig. 5a, we implement D and G with multilayer perceptron.

B. Training and Generation

Before the training, we first normalize the error data $\boldsymbol{x}$ with the maximum error value in the training dataset, and reshape the error distribution of a block $P_{data(\boldsymbol{x})}$ to (48×48). If it cannot be reshaped into a square in other flash models, we can fill the vacant position with zero.

We launch the training of D with the data combining $\boldsymbol{x}$ and the corresponding condition $\boldsymbol{y}$. After that, we feed a random-number vector $\boldsymbol{z}$ and $\boldsymbol{y}$ to G to train and get the generated data $G(\boldsymbol{z}|\boldsymbol{y})$, which are then sent to D. D will discriminate the fed data are real or fake, and feedback the result to G. D and G are both simultaneously trained until G cannot discriminate the fake data correctly. In the training, cGAN adjusts the parameters of G to minimize $log(1-D(G(\boldsymbol{z}|\boldsymbol{y})))$ and adjusts parameters of D to minimize $log(\boldsymbol{x}|\boldsymbol{y})$, as if they are following the two-player min-max game with the value function $V(G;D)$:

$$\min_{G} \max_{D} V(G, D) = \mathbb{E}_{\boldsymbol{x} \sim P_{data(\boldsymbol{x})}} [logD(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z} \sim P_{z(\boldsymbol{z})}} [log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))] \quad (1)$$

Repeating the process of feeding new noise vector $\boldsymbol{z}$ and condition $\boldsymbol{y}$ to G, and denormalizing $G(\boldsymbol{z}|\boldsymbol{y})$ to get the final generated error data. A note for training is keeping the population balance of data from different conditions.

V. Evaluation

In this section, we evaluate the authenticity, diversity, and efficiency of the proposed methodology. We use half of the collected data to train and generate the same amount of data, then compare them compare with the left half collected data.

A. Authenticity

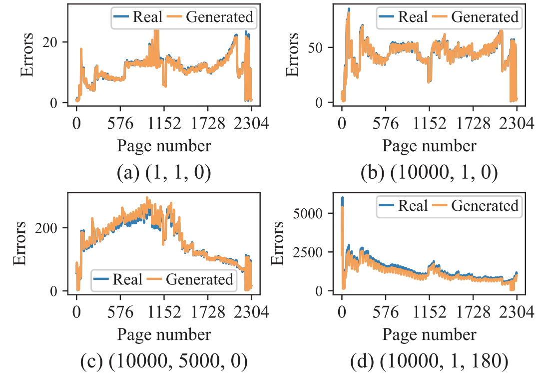


Fig. 3: Evaluation of error distribution.

The authenticity of error data is multi-dimensional. The first dimension is the error distribution comparison between the real and generated error data. We train the error data of 300 blocks covering all interference conditions and then generate 300-block new error data. The comparisons of average error distribution in four classic scenarios are plotted in Fig. 3. The generated error distribution has a high coincidence with the real distribution in all scenarios. The results validate

cGAN generates the expected error distribution of the group through the learning of individuals.

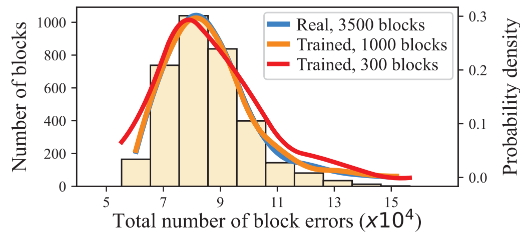
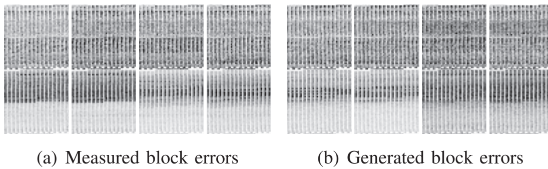


Fig. 4: Evaluation of the total number of block errors.

To further validate the authenticity, we select the total number of block errors as the metric since it has not participated in the training at all. Given the data amount of the synthetic test is less than 1000, we use the data collected from the cycling test in this evaluation. We trained the error data from 300 and 1000 blocks respectively, and generate 3500 blocks. We delineate the histogram (the left y-axis) of the total number of block errors from 3500 real blocks in Fig. 4 and fit the probability density (the right y-axis) of it. It has a Poisson-quasi distribution. When trained more data to 1000 blocks, the distribution of the total number of block errors gets more approximating to the real. The distribution curve of trained 1000 blocks almost coincides with the real distribution, which demonstrates cGAN's ability of learning hidden features and the authenticity of generated data.

B. Diversity

We expect to generate diverse data to increase the population of real-world error data. However, there is no metric or model to characterize the diversity for individuals. Hence, we directly observe the diversity of generated results shielded by cGAN. Fig. 5b gives some examples of generated blocks. The top four blocks are generated with the interference condition (10000, 1, 0), and the below four blocks are with (10000, 5000, 0). Compared with Fig. 1 and Fig. 5a, we observe each generated block has unique error values and error distribution characteristics for itself. The proposed methodology shows an excellent ability to generate diverse error data while keeping the expected group features.



(a) Measured block errors (b) Generated block errors

Fig. 5: The diversity evaluation of generated block errors.

C. Efficiency

We train cGAN and evaluate it on a Ubuntu 18.04 server equipping an Intel i9-10920X CPU, 192 GB DDR4 memory, and an NVIDIA RTX 2080Ti GPU. We implement cGAN with Pytorch 1.3.0 and Python 3.7.7. The time of training 1000 epochs to obtain the above-evaluated models varies from a few minutes to a few hours, and it depends on the batch size, the learning rate, and other parameters. It generates 1 million blocks of error data efficiently taking only 410 seconds in CPU mode. Compared with several months to collect a limited amount of data, the overhead of training and generation is negligible.

VI. Conclusion

In this paper, we made a retrospect and a future outlook of 3D NAND flash memory. Motivated by the gap between the increasing test costs and the need for a large amount of real-world data, we proposed an error generation method to reproduce realistic data efficiently. We demonstrated the feasibility, authenticity, and efficiency of the proposed methodology, based on comprehensive characterizations and evaluations. We hope this method can enrich the research of test generation and facilitate the test and reliability optimization for 3D NAND flash memory. We especially expect to design a soft emulator of 3D NAND flash memory with more characteristics to further reduce the hardware validation costs in the future.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grants No. U2001203, No. 61872413, No. 62102156, and No. 61902137, in part by the Fundamental Research Funds for the Central Universities, HUST: No. 2021JYCXJJ033, in part by the Key Project of Shandong Wisdom Joint Fund under Grant No. ZR2019LZH009.

References

[1] W. Liu, F. Wu, M. Zhang, C. Yang, Z. Lu, J. Wan, and C. Xie, "Deps: Exploiting a dynamic error prechecking scheme to improve the read performance of ssd," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 1, pp. 66–77, 2020.

[2] D. Kang, W. Jeong, C. Kim, D.-H. Kim, Y. S. Cho, K.-T. Kang, J. Ryu, K.-M. Kang, S. Lee, W. Kim, *et al.*, "256 gb 3 b/cell v-nand flash memory with 48 stacked wl layers," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 210–217, 2016.

[3] A. Goda, "3-d nand technology achievements and future scaling perspectives," *IEEE Transactions on Electron Devices*, vol. 67, no. 4, pp. 1373–1381, 2020.

[4] D.-H. Kim, "A 1 tb 4bcell 5th-generation 3d-nand flash memory with 2ms tprog, 110us tr and 1.2 gbspin interface," in *2021 IEEE International Memory Workshop (IMW)*, pp. 1–4, IEEE, 2021.

[5] W. Liu, F. Wu, M. Zhang, Y. Wang, Z. Lu, X. Lu, and C. Xie, "Characterizing the reliability and threshold voltage shifting of 3d charge trap nand flash," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 312–315, IEEE, 2019.

[6] N. R. Mielke, R. E. Frickey, I. Kalastirsky, M. Quan, D. Ustinov, and V. J. Vasudevan, "Reliability of solid-state drives based on nand flash memory," *Proceedings of the IEEE*, vol. 105, no. 9, pp. 1725–1750, 2017.

[7] Y. Cai, S. Ghose, E. F. Haratsch, Y. Luo, and O. Mutlu, "Error characterization, mitigation, and recovery in flash-memory-based solid-state drives," *Proceedings of the IEEE*, vol. 105, no. 9, pp. 1666–1704, 2017.

[8] M. Zhang, F. Wu, Q. Yu, W. Liu, L. Cui, Y. Zhao, and C. Xie, "Beldpc: bit errors aware adaptive rate ldpc codes for 3d tlc nand flash memory," in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 302–305, IEEE, 2020.

[9] M. Zhang, F. Wu, Q. Yu, W. Liu, Y. Wang, and C. Xie, "Exploiting error characteristic to optimize read voltage for 3-d nand flash memory," *IEEE Transactions on Electron Devices*, vol. 67, no. 12, pp. 5490–5496, 2020.

[10] W. Liu, F. Wu, J. Zhou, M. Zhang, C. Yang, Z. Lu, Y. Wang, and C. Xie, "Modeling of threshold voltage distribution in 3d nand flash memory," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1729–1732, IEEE, 2021.

[11] N. Papandreou, H. Pozidis, T. Parnell, N. Ioannou, R. Pletka, S. Tomic, P. Breen, G. Tressler, A. Fry, and T. Fisher, "Characterization and analysis of bit errors in 3d tlc nand flash memory," in *2019 IEEE International Reliability Physics Symposium (IRPS)*, pp. 1–6, IEEE, 2019.

[12] C.-C. Hsieh, H.-T. Lue, T.-H. Hsu, P.-Y. Du, K.-H. Chiang, and C.-Y. Lu, "A monte carlo simulation method to predict large-density nand product memory window from small-array test element group (teg) verified on a 3d nand flash test chip," in *2016 IEEE Symposium on VLSI Technology*, pp. 1–2, IEEE, 2016.