Spatio-Temporal Modeling for Flash Memory Channels Using Conditional Generative Nets

Simeng Zheng, Chih-Hui Ho, Wenyu Peng, and Paul H. Siegel
Electrical and Computer Engineering Dept., University of California, San Diego, La Jolla, CA 92093 U.S.A
{*sizheng,chh279,w6peng,psiegel*}*@ucsd.edu*

***Abstract*—Modeling spatio-temporal read voltages with complex distortions arising from the write and read mechanisms in flash memory devices is essential for the design of signal processing and coding algorithms. In this work, we propose a data-driven approach to modeling NAND flash memory read voltages in both space and time using conditional generative networks. This generative flash modeling (GFM) method reconstructs read voltages from an individual memory cell based on the program levels of the cell and its surrounding cells, as well as the time stamp. We evaluate the model over a range of time stamps using the cell read voltage distributions, the cell level error rates, and the relative frequency of errors for patterns most susceptible to inter-cell interference (ICI) effects. Experimental results demonstrate that the model accurately captures the complex spatial and temporal features of the flash memory channel.**

***Index Terms*—Machine learning, Flash memory channel, Generative modeling, Spatio-temporal analysis.**

I. Introduction

The steady reduction in technology feature size and the increase in cell bit-density have been accompanied by diminished memory reliability and reduced device endurance. Sources of errors are manifold, including programming errors, inter-cell interference (ICI), cell wear during program/erase (P/E) cycling, cell charge loss due to data retention, and program/read disturb effects. Wear leveling [21] can equalize the wear conditions across blocks by managing data placement to enhance endurance. Error correction codes (ECC) [7] are able to correct errors in read operation by incorporating redundant correction bits. Constrained codes [5] avoid patterns prone to errors caused by inter-cell interference (ICI) effects. The design of these reliability-enhancing algorithmic techniques relies upon a comprehensive understanding of the complex spatio-temporal behavior of the flash memory channel. An accurate mathematical model of the channel is therefore an indispensable tool. Moreover, such a model can potentially obviate the need for hardware- and time-intensive data collection for use in evaluating and optimizing the algorithms.

Traditional statistical modeling and physics-based modeling, as well as some recently proposed machine-learning models, have limitations in their ability to fully capture and fuse the spatial and temporal characteristics of the flash memory channel.

Our goal in this paper is to use advanced machine-learning techniques to develop an accurate generative model for flash memory read voltages that accounts for both spatial ICI effects and temporal distortions arising from P/E cycling and retention. This data-driven approach can be flexibly applied to flash memories of any technology generation and chip feature size.

Recently, generative modeling techniques such as the Generative Adversarial Network (GAN) [3] and the Variational Adversarial Encoder (VAE) [9] have been successfully applied to image processing [8]. In view of the demonstrated power of neural networks in learning complex multidimensional distributions, we propose the use of conditional generative nets as an approach to modeling flash memory read voltages in space and time. The learned simulator is trained to replace the real chip measurements, where the read voltage levels can be regenerated from program levels.

To our knowledge, this is the first modeling framework that combines both temporal features (P/E cycles and retention) and spatial characteristics (ICI effects). We summarize our contributions as follows,

1) We propose a flexible, data-driven, generative flash modeling (GFM) framework to accurately reconstruct soft read voltages at the cell level.
2) We formulate a temporally controllable, conditional VAE-GAN network to regenerate cell read voltages from the program levels of an array of flash memory cells and a time stamp representing the P/E cycle count and the retention time.
3) We validate the proposed GFM method on a 1X-nm, 3-bit per cell (TLC) NAND flash chip. The evaluation metrics used to compare the model outputs to measurements include the read voltage distributions at different time stamps, the error counts of different program levels, and the relative frequencies of spatially-dependent and pattern-dependent ICI-induced hard read errors.

II. Background and Related Work

A. Flash Memory Basics

The basic unit of data storage in NAND flash memory is a floating-gate transistor, referred to as a cell. Today's flash memories are capable of storing single or multiple bits (e.g. 2 to 5 bits) per cell, where the n-bit strings correspond to 2^n program levels. The cells are organized into an interconnected two-dimensional (2D) array, called a block, via horizontal wordlines (WLs) and vertical bitlines (BLs). The flash memory chip is composed of a collection of such blocks. In 3D NAND flash, these 2D arrays are stacked vertically to achieve larger volumetric density [13], [19]. Fig. 1 depicts a schematic diagram of a planar TLC flash memory block and an example of a Gray mapping from program levels to binary digits.

The basic unit of write (i.e., program) and read operations in flash memory is a page, corresponding to a logical bit position in a wordline of a block. We refer to the program level as PL

978-3-9819263-7-8/DATE23/© 2023 EDAA

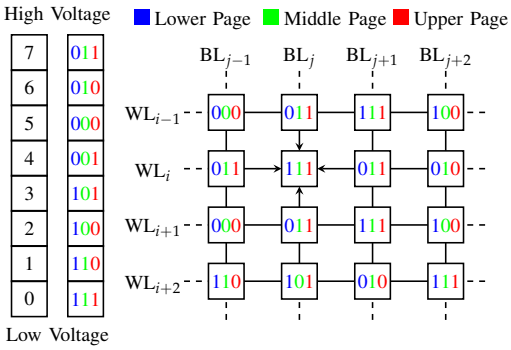


Fig. 1: (Left) Example mapping of cell program levels to binary representations in TLC flash. (Right) Schematic diagram of a TLC flash memory block showing the 2-D array of cells connected in the horizontal direction by wordlines (WLs) and in the vertical direction by bitlines (BLs).

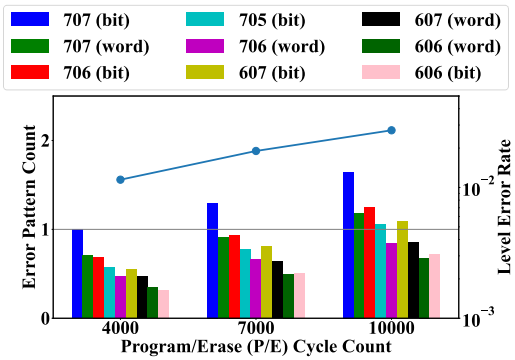


Fig. 2: Count of top error-prone patterns and level error rate at selected P/E cycles without retention. The error pattern counts are normalized by the count of pattern 707 in BL direction at 4000 P/E cycles.

and the soft read voltage level as VL. On the other hand, the basic unit of an erase operation is an entire block.

In the program operation, we refer to the PLs of three consecutive cells along WLs or BLs as a pattern. As an example, in Fig. 1, the programmed levels $\text{PL}_{i-1,j}\text{PL}_{i,j}\text{PL}_{i+1,j}$ in WLs $(i-1), i, (i+1)$ of BL j, correspond to bit strings "011", "111", "011", which we associate with the pattern 707 in the vertical (BL) direction.

B. Spatio-temporal Characteristics

Flash memory channels suffer from distortions of a spatio-temporal nature.

Spatial effects: ICI refers to the phenomenon where programming of a cell induces changes in the voltage levels of neighboring cells within its block. In particular, the read voltage level of a cell programmed to a low level may be inadvertently increased if its adjacent cells are programmed to high levels, i.e., when the programming pattern is high-low-high. As an example in Fig. 1, if we program a 707 pattern along WL or BL in a TLC chip, the read voltage $\text{VL}_{i,j}$ may be increased by its high adjacent cells. During data detection, the recovered program level of the central "victim" cell may therefore be erroneously interpreted as an incorrect level. ICI typically differs in the WL and BL directions.

Temporal effects: P/E cycling gradually wears out the oxide layer of a cell. Data retention causes the programmed cell to diffuse electrons over time. As a result of these effects, the information stored in a cell can be misread.

The integrated distortions of spatial ICI and temporal P/E cycling errors can be observed in Fig. 2. As the P/E cycle count increases, the error rate is increasing. For an individual P/E cycle count, the cell errors are clearly affected by neighboring program levels. Pattern 707 in the BL direction is the most severely affected by ICI. Moreover, patterns 707, 706, and 607 in the BL direction are more error-prone than those on the WL direction.

C. Related Work

Several mathematical and machine learning models of voltage levels supported by empirical measurements or simulated voltages in flash memory have appeared in the literature. Cai et al. [1] models the voltage distribution in 2-bit per cell MLC flash devices as a Gaussian distribution. Parnell et al. [20] proposed a parameterized Normal-Laplace mixture model that more accurately describes MLC flash read voltage distributions. Luo et al. [17] proposed another accurate and computationally more efficient model for MLC flash, based on a modified version of the Student's t-distribution and a temporal power law. Statistical analysis of hard bit errors in [19], [24] and characterization of dominant error patterns in [2], [11] offer additional empirical understanding of flash memories.

Recently, machine learning has been exploited to model flash memory channels. Liu et al. [16] used a neural network (NN) to model simulated read voltages as a function of P/E cycles for one individual program level in isolated MLC flash cells. Liu et al. [14] provided flash memory conditions to a NN to model voltage distributions for 3D NAND flash. Liu et al. [15] used a time-dependent NN to predict page error counts in TLC flash. Liu et al. [12] generated errors in 3D NAND flash using a conditional GAN architecture.

However, as effective as these models have been in the scenarios to which they have been applied, none has provided an accurate model of both spatial and temporal characteristics of flash memory read voltages.

III. Generative Flash Modeling

In this section, we propose generative flash modeling (GFM). We adopt a conditional VAE-GAN (cVAE-GAN) architecture [10] for our pipeline, depicted in Fig. 3, where the fusion of the VAE [9] and GAN [3] can leverage the information from the latent space to produce high-quality, accurate reconstruction with the help of the discriminator. Our goal is to learn a mapping between program levels and soft read voltage levels at various time stamps, where the reconstructed voltage levels accurately reflect the spatial and temporal nature of the channel.

A. Pipeline Formulation

Given program level PL, read voltage level VL, time stamp P/E cycles P/E, and retention time γ, we aim to learn the

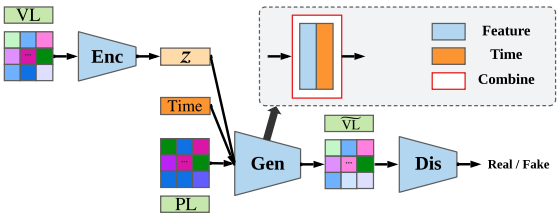


Fig. 3: Generative modeling pipeline: encoder, generator, and discriminator constitute the generative modeling workflow. Here, z is the latent vector; P/E is the corresponding P/E cycle count; PL is the array of program levels, VL is the array of measured read voltage levels, and $\widetilde{\text{VL}}$ is the reconstructed array of read voltage levels. In our implementation, PL, VL, and $\widetilde{\text{VL}}$ are all 64×64 arrays.

analytically intractable likelihood $P(\text{VL}|\text{PL}, \text{P/E}, \gamma)$, with the goal of capturing the spatio-temporal nature of flash memory channel.

Fig. 3 summarizes the architecture of the generative modeling pipeline. The conditional VAE-GAN architecture consists of three components: an encoder (Enc), a generator (Gen), and a discriminator (Dis). The encoder maps the read voltages to the latent vector z at a specific P/E cycle count and retention time γ and replaces the prior distribution $P(z)$ in the GAN with the learned posterior distribution $P(z|\text{VL}, \text{P/E}, \gamma)$. The decoder in the VAE shares its weights with the GAN generator [3]. In the conditional setting, the variational lower bound of $P(\text{VL}|\text{P/E})$ can be derived as

$$\begin{aligned}\log P(\text{VL}|\text{P/E}, \gamma) \geq & - D_{KL}(Q(z|\text{VL}, \text{P/E}, \gamma)||P(z|\text{P/E}, \gamma)) \\ & + \mathbb{E}_{Q(z|\text{VL},\text{P/E},\gamma)}[\log(P(\text{VL}|z, \text{P/E}, \gamma))]\end{aligned}$$

where D_{KL} represents the Kullback-Leibler (KL) divergence. The distribution $Q(z|\text{VL}, \text{P/E}, \gamma)$ of the latent vector z is trained to approach $P(z|\text{P/E}, \gamma)$ via the KL loss $\mathcal{L}_{KL}$, where $P(z|\text{P/E}, \gamma)$ is assumed to be a Gaussian distribution.

Generator Gen will take both the learned latent vector and PL as input and generate a "fake" $\widetilde{\text{VL}}$. The latent vectors are sampled from $Q(z|\text{VL}, \text{P/E}, \gamma)$ using the re-parameterization trick [9]. When sampling different latent vectors z from the same distribution, we can generate multiple arrays of plausible voltages levels. The variations in these output arrays for a given array of program levels reflect the stochasticity of the channel. The discriminator measures the difference between PL and $\widetilde{\text{VL}}$. The loss in the conditional GAN part is

$$\mathcal{L}_{GAN} = \log(1 - Dis(\text{PL}, Gen(\text{PL}, \text{P/E}, \gamma, z))) + \log(Dis(\text{PL}, \text{VL})).$$

Similar to VAE-GAN [10], we encourage the reconstructed voltage levels to match the authentic voltage levels, using the ℓ_2-norm to measure the reconstruction loss

$$\mathcal{L}_{recon} = ||\text{VL} - Gen(\text{PL}, \text{P/E}, z)||_2.$$

Combining these equations, we formulate the loss function of the cVAE-GAN architecture as

$$\min_{Gen,En} \max_{Dis} \mathcal{L}_{GAN} + \alpha\mathcal{L}_{recon} + \beta\mathcal{L}_{KL}. \quad (1)$$

B. Spatio-temporal Fusion

To capture the spatial ICI effects in the channel model, we implement the generator using a convolutional neural network (CNN) in Gen, where VL is reconstructed from the PL values in its cell and neighboring cells. To generate VL at an explicit time stamp, we control the generator with an additional temporal factor and incorporate the time stamp into Gen.

We use controllable P/E cycle count and fixed retention time as an example. We first encode the normalized P/E cycle count into a d-dimensional P/E vector, which contains expressive powers of the normalized P/E cycle, e.g., P/E^2, $\sqrt{\text{P/E}}$, etc. Then, we spatially replicate the d-dimensional P/E vector to the feature map with appropriate size $H \times W \times d$ and concatenate it with the $H \times W \times C$ feature from each layer in Gen, where $H \times W$ is the spatial dimension of the feature from each convolutional layer and C is the number of channels in the CNN. The channel-wise fusion produces the final feature with size $H \times W \times (C + d)$ of each layer. The fusion of the features from the program levels and the P/E feature maps guarantees the spatial-temporal characteristics of the reconstructed voltage levels.

C. Implementation Details

Datasets: We verify the GFM method using datasets collected from one commercial 1X-nm TLC flash chip, where the selected P/E cycles are 4000, 7000, and 10000 and read operation is implemented with fixed retention time. The P/E cycling experiment is conducted by erasing a block, programming pseudo-random data, and reading voltage level at selected P/E cycle counts.

We crop the $\{\text{PL}, \text{VL}\}$ pairs of the recorded blocks into non-overlapping 64×64 2-D arrays. The number of 2-D arrays in the training set is 1.5×10^5 (5×10^4 for each P/E cycle) and the size of the evaluation dataset is 2.1×10^4 (7×10^3 for each P/E cycle).

Network: Three network modules in Fig. 3 refer to: ResNet [6] (Enc), U-net [22] (Gen), and PatchGAN [8] (Dis). The dimensions of latent vector z and P/E cycle vector are both set to 6. The following descriptions of the modules exploit the terminologies in the corresponding references.

1) Encoder: We use the two residual blocks, each of which contains two 3×3 convolutional layers with stride 1 and padding 1. We then add two linear layers, which map output features to mean and variance for the latent vector.
2) Generator: Ck denotes a Convolution-BatchNorm-ReLU layer with k output channels. All convolutions are 4×4 kernels applied with stride 2 and padding 1. The network architecture before spatio-temporal fusion can be described as

   (Down Part) $C64, C128, C256, C512, C512, C512$

   (Up Part) $C512, C512, C256, C128, C64, C1$

   where we inject latent vector z by replication and concatenation into every layer in the "Down" part [25], and each layer in the "Up" part receives skip connections from the corresponding layer in the "Down" part [22].

3) Discriminator Dis: The input to the discriminator is the concatenation of fake voltage levels and program levels. With the same naming convention as in the generator, we express the discriminator as $C64, C128, C1$.

We compared the cVAE-GAN model to other popular generative modeling architectures: conditional GAN [8], conditional VAE [23], and Bicycle GAN [25]. The cVAE-GAN model reconstructs the read voltage levels with the highest quality.

Learning: We settled upon the training parameters after several experiments. Adam optimizer is used with learning rate 2×10^{-4}. Parameters in the loss function (1) are set to $\alpha = 10$ and $\beta = 0.01$. We train cVAE-GAN for 7 epochs with batch size 2.

During evaluation, we use program levels and latent vector z sampled from a standard Gaussian distribution. For each program level array, we sample 10 different latent vectors to evaluate the learned model.

IV. Experimental Results

We first discuss the measurement time from the real chip and the inference time of GFM. When we collect one block of data at selected P/E cycles, it requires approximately 4 hours on our FPGA-based platform and the block becomes unavailable for further experiments. However, in the GFM pipeline, the data generation of one block can be completed within 400 seconds under CPU mode (Intel i7-9700K, 3.60GHz×8) and no flash block is wasted.

To evaluate the quality of the reconstructed voltage levels and analyze the spatio-temporal nature of flash memory channel, we analyze the regenerated VL using the following metrics.

1) Distribution: The frequency of occurrence of each voltage level given the program level and P/E cycle count is used to estimate the conditional probability of that level and time. We visualize the PDFs for measured data and reconstructed data.
2) Error count: We compare the generated read voltages with thresholds to determine the read levels and then compute the error count for each program level. For quantitative comparison, we compare the read voltages generated by our data-driven method with three statistical fitting methods [17], using the metric of level error count.
3) Inter-cell interference (ICI): For cells programmed to 0 level that suffer an error according to their voltage level, we compute the relative frequencies of the patterns of program levels in adjacent cells in the WL and BL directions. We visualize these relative error frequencies using pie charts.

A. Distribution Analysis

As we evaluate our learned model using input arrays of program levels, we collect regenerated voltage levels and count the frequency of occurrence of voltage levels over the voltage range. We then estimate the conditional PDFs of voltages associated with each program level and given P/E cycle.

Fig. 4 shows the conditional PDFs for measured data and regenerated data in the evaluation dataset at three different time stamps. The x-axis represents the soft read voltages spanning a certain voltage level range. The y-axis represents the conditional PDF. We only show conditional PDFs from program level 1 to 7 due to normalization problems of program 0; the detailed analysis of level 0 will be discussed in Section IV-C. We make two observations from Fig. 4. First, as P/E cycle count increases, the peak of the distribution in each program level drops and the distribution becomes wider. This indicates that more voltage levels exceed the read thresholds and more errors will be detected, which is consistent with the increased error rate in Fig. 2. Second, our modeled data (solid curves) generally fit the measured data (triangle markers) and capture the dependence on P/E cycles. The occasional discrepancies in the peak do not affect the error analysis in the tails.

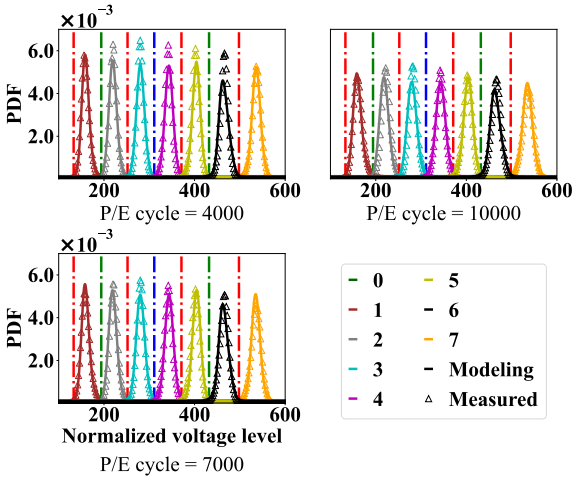


Fig. 4: PDF visualizations for measured and cVAE-GAN generated voltage levels at 4000, 7000, and 10000 P/E cycles. In each subfigure, solid curves represent the cVAE-GAN modeled distribution and triangles represent the measured distribution. The plots are in linear scale. Vertical dash-dotted lines are fixed default voltage thresholds.

B. Error Count

Accurate estimation of error count is essential in the design of wear leveling and ECC algorithms for NAND flash. For a more quantitative assessment of error count, we compare our generative model with three state-of-the-art statistical models using the metric of error count: Gaussian model [1], Normal-Laplace model [20], and Student's t-distribution model [17]. As far as we know, our proposed machine learning method is the first approach to provide accurate estimation of two-dimensional pattern-dependent and time-dependent errors. Following the optimization process used in [17] for an MLC flash device, we fit those statistical distributions to our TLC measured distributions. We minimize the KL divergence between real distribution P_{real} and fake distribution P_{fake} by using the Nelder-Mead simplex method [18], where the KL divergence is denoted as $D_{KL}(P_{real}, P_{fake})$. We obtain the best-fit parameters for all program levels, except PL $= 0$, with each of the statistical distributions.

We then compute the level error counts under each of the distributions and quantitatively compare those fake distributions with real distributions, where the error count is a measure of the reliability and endurance of the flash memory. To calculate the level error counts from distributions, we fix 7 default read thresholds, as shown by the dash-dotted vertical lines in Fig. 4. The hard read voltages are determined by comparing soft voltages to these thresholds. For instance, if a voltage level of program level 1 lies below the first threshold or above the second threshold, the hard read level of the cell will not be designated as 1. The error probability of PL $= 1$ is denoted as

$$P(\text{VL} < V_{th(01)}|\text{PL} = 1) + P(\text{VL} > V_{th(12)}|\text{PL} = 1)$$

where $V_{th(01)}$ denotes the voltage threshold used to distinguish between level 0 and level 1, and $V_{th(12)}$ denotes the voltage threshold used to distinguish between level 1 and level 2.

We present the level error counts for five models in the form of bar charts Fig. 5. The x-axis represents the chosen P/E cycle count and the label directly under each bar represents the corresponding model name. The y-axis corresponds to the normalized error count. At each P/E cycle count, for each model, the errors from 7 program levels are stacked as one individual bar. The stacked bar represent the total error count. For the measured distributions, we see that level 1 has the highest error counts and the total error count at 10000 P/E cycles is around 2.5× that at 4000 P/E cycles.

For the statistical distributions, we observe that the Gaussian model does not estimate the error well; this is because the tails in the actual distribution are becoming heavier as the device is cycled to severe conditions. The Normal-Laplace model, on the other hand, takes the heavier tails into consideration and provides accurate estimations of error counts at each P/E cycle. Student's t-distribution over-estimates the errors at those P/E cycles. For the machine learning approach, cVAE-GAN slightly overestimates the wear conditions in 7000 and 10000 P/E cycles. At 4000 P/E cycles, cVAE-GAN produces more errors than the measured data. In conclusion, Normal-Laplace is the best statistical model to capture the distributions of flash devices. Our GFM works better than Normal-Laplace at 7000 and 10000 P/E cycles but overestimate the errors at 4000 P/E cycles. (However, as shown in the section IV-C, the generative model can not only generate realistic-looking voltage distributions but also accurately learn spatial characteristics of the read voltages.)

C. ICI Characterization

Generating voltage levels with ICI effects is complicated due to pattern-dependent and direction-dependent distortions. We remark that classical statistical models focus on regeneration of the PDFs of the measured data and, as such, are not expected to be effective in capturing ICI effects.

We evaluate how well the generative model learns spatial ICI properties by examining errors associated with program level patterns $\text{PL}_{i,j-1}$ $\text{PL}_{i,j}$ $\text{PL}_{i,j+1}$ and $\text{PL}_{i-1,j}$ $\text{PL}_{i,j}$ $\text{PL}_{i+1,j}$ in the WL and BL directions, respectively. As we observed in Fig. 2, the most error-prone patterns have central victim cell as 0, i.e., $\text{PL}_{i,j} = 0$. We consider pattern-dependent error probabilities in both directions. The error probability measures

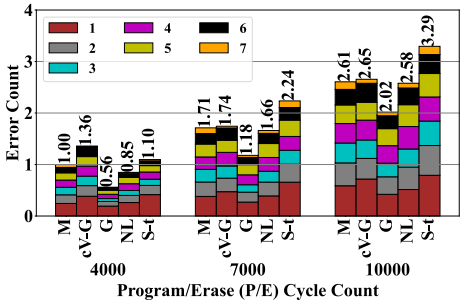


Fig. 5: Total error counts of measured ('M'), cVAE-GAN ('cV-G'), Gaussian ('G'), Normal-Laplace ('NL'), Student's t ('S-t') model.In each bar, we stack the errors from program level 1 to program level 7. We normalize the error counts of measured data at 4000 P/E cycle as 1.

the relative frequency of occurrence of the WL and BL patterns when an error occurs in the victim cell. More precisely, we calculate the pattern-dependent error probabilities in WL and BL directions,

$$P(\text{PL}_{i,j-1}, \text{PL}_{i,j} = 0, \text{PL}_{i,j+1}|\text{VL}_{i,j} > V_{th(01)}, \text{PL}_{i,j} = 0)$$
$$P(\text{PL}_{i-1,j}, \text{PL}_{i,j} = 0, \text{PL}_{i+1,j}|\text{VL}_{i,j} > V_{th(01)}, \text{PL}_{i,j} = 0).$$

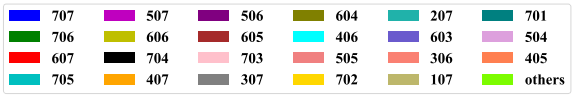
The probabilities for measured and regenerated data are visualized as pie charts in Fig. 6. When we program an interior cell to level 0, there are 64 such patterns of program levels for the pair of adjacent cells in both WL and BL directions. Without ICI effects, the errors would occur randomly for all possible patterns.

In the measured data, the 23 listed patterns account for 55% of the errors in the WL direction and around 70% of the errors in the BL direction. The dominant error pattern in both WL and BL directions is 707. Comparing the area of pattern 707 in WL and BL, we find that pattern 707 in the WL direction is less severe than that in the BL direction.

As shown in Fig. 6, for the prevalent error patterns at 7000 P/E cycles, probabilities observed in the data generated by cVAE-GAN with 10 sampled latent vector during evaluation are very similar to those seen in the measured data. The only substantial discrepancy we observed is that the generative model underestimates the fraction of the 707 pattern in the WL direction. At 4000 (resp., 10000) P/E cycles, the model underestimates (resp., overestimates) the fraction of the 707 pattern in both directions. However, at all P/E cycles, GFM generates the same rank ordering of pattern fractions as the measured data in both directions.

D. Discussion

An advantage of the proposed cVAE-GAN architecture for generative flash modeling is that it can provide an unlimited supply of read voltages under different programming conditions by repeated sampling of latent vectors. This is vital, for example, when optimizing read thresholds for ECC decoding via error rate simulation [7]. It also offers the ability to generate estimated error counts for determining the health of blocks when designing wear-leveling algorithms [21].



(a) Measured errors in WL (left) and BL (right)
97921 total errors observed at PL = 0

(b) cVAE-GAN errors in WL (left) and BL (right)
989565 total errors observed at PL = 0

Fig. 6: Pie charts showing pattern-dependent error probabilities for measured and GFM generated voltages at 7000 P/E cycles. The sector labeled **others** combines 41 less harmful patterns.

GFM also provides accurate characterization of dominant error patterns in the presence of ICI effects, which is an essential tool in the design of constrained codes [5]. Moreover, by including time-aware ICI modeling, GFM can assist the development of adaptive and reconfigurable constrained coding schemes that eliminate different error-prone code patterns as a function of time stamps [4].

Our framework can be used to model retention effects over extended periods of time when given an appropriate dataset of experimental measurements. Moreover, we believe the method can be generalized to any multi-level flash device including 3D NAND flash. In order to characterize the severe layer-to-layer interference in 3D flash [12], [19], we can crop the 3D arrays of $\{\text{PL}, \text{VL}\}$ pairs and modify the GFM network realization to include 3D convolutional networks in all of the modules.

V. Conclusion

In this paper, we explored the use of conditional generative networks to model the flash memory channel. Unlike traditional modeling and previous machine learning approaches, our model can generate "realistic" soft voltage levels from program level arrays at different time stamps, thereby reflecting both temporal and spatial characteristics of the flash memory channel.

References

[1] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling," in *Proc. Design, Automation & Test in Europe Conf. & Exhib. (DATE)*, Grenoble, France, Mar. 2013.

[2] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis," in *Proc. Design, Automation & Test in Europe Conf. & Exhib. (DATE)*, Dresden, Germany, Mar. 2012.

[3] I. J. Goodfellow, J. P.-Abadie, M. Mirza, B. Xu, D. W.-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Neural Inf. Process. Syst. (NIPS)*, Montréal, Canada, Dec. 2014, pp. 2672–2680.

[4] A. Hareedy, B. Dabak, and R. Calderbank, "Managing device lifecycle: reconfigurable constrained codes for M/T/Q/P-LC flash memories," *IEEE Trans. Inf. Theory*, vol. 67, no. 1, pp. 282–295, Oct. 2020.

[5] A. Hareedy, S. Zheng, P. H. Siegel, and R. Calderbank, "Read-and-run constrained coding for modern flash devices," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Seoul, South Korea, May 2022.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, July 2016, pp. 770–778.

[7] P. Huang, Y. Liu, X. Zhang, P. H. Siegel, E. F. Haratsch, "Syndrome-coupled rate-compatible error-correcting codes: theory and application," *IEEE Trans. Inf. Theory*, vol. 66, no. 4, pp. 2311–2330, Jan. 2020.

[8] P. Isola, J-Y. Zhu, T. Zhou, A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, July 2017, pp. 1125–1134.

[9] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Represent. Learn. (ICLR)*, Banff, Canada, Apr. 2014.

[10] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, O. Winther, "Autoencoding beyond pixels using a learned similarity metric," in *Int. Conf. Mach. Learn. (ICML)*, New York, USA, June 2016.

[11] Q. Li, A. Jiang, and E. F. Haratsch, "Noise modeling and capacity analysis for NAND flash memories," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, USA, June. 2014, pp. 2262–2266.

[12] W. Liu, F. Wu, S. Meng, X. Chen, and C. Xie, "Error generation for 3D NAND flash memory," in *Proc. Design, Automation & Test in Europe Conf. & Exhib. (DATE)*, Antwerp, Belgium, Mar. 2022.

[13] W. Liu, F. Wu, M. Zhang, Y. Wang, Z. Lu, X. Lu, C. Xie, "Characterizing the reliability and threshold voltage shifting of 3D charge trap NAND flash," in *Proc. Design, Automation & Test in Europe Conf. & Exhib. (DATE)*, Florence, Italy, Mar. 2019.

[14] W. Liu, F. Wu, J. Zhou, M. Zhang, C. Yang, Z. Lu, Y. Yang, and C. Xie, "Modeling of threshold voltage distribution in 3D NAND flash memory," in *Proc. Design, Automation & Test in Europe Conf. & Exhib. (DATE)*, Grenoble, France, Feb. 2021.

[15] Y. Liu, S. Wu, and P. H. Siegel, "Bad page detector for NAND flash memory," in *Annual Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, USA, Mar. 2020.

[16] Z. Liu, Y. Liu, and P. H. Siegel, "Generative modeling of NAND flash memory voltage level," in *Annual Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, USA, Mar. 2021.

[17] Y. Luo, S. Ghose, Y. Cai, E. F. Haratsch and O. Mutlu, "Enabling accurate and practical online flash channel modeling for modern MLC NAND flash memory," *IEEE J. Select. Areas Commun.*, vol. 34, no. 9, pp. 2294–2311, Sept. 2016.

[18] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[19] N. Papandreou, H. Pozadis, T. Parnell, N. Loannoum, 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 *IEEE Int. Relilability Phys. Symp. (IRPS)*, Monterey, CA, USA, Apr. 2019.

[20] T. Parnell, N. Papandreou, T. Mittelholzer, and H. Pozidis, "Modelling of the threshold voltage distributions of sub-20nm NAND flash memory," in *Proc. IEEE Global Commun. Conf.*, Dec. 2014, pp. 2351–2356.

[21] R. Pletka, I. Koltsidas, N. Ioannou, S. Tomic, N. Papandreou, T. Parnell, H. Pozidis, A. Fry, and T. Fisher, "Management of next-generation NAND flash to achieve enterprise-level endurance and latency targets," *ACM Trans. Storage*, vol. 14, no. 4, pp. 1–25, Nov. 2018.

[22] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," in *Int. Conf. Med. Imag. Comput. Comput.-assisted Intervention (MICCAI 2015)*, Springer, Cham.

[23] K. Sohn, H. Lee, and X. Yan "Learning structured output representation using deep conditional generative models," in *Proc. Neural Inf. Process. Syst. (NIPS)*, Montréal, Canada, Dec. 2015.

[24] V. Taranalli, H. Uchikawa, and P. H. Siegel, "Channel models for multi-level cell flash memories based on empirical error analysis," *IEEE Trans. Commun.*, vol. 64, no. 8, pp. 3169–3181, Aug. 2016.

[25] J-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward multimodal image-to-image translation," in *Proc. Neural Inf. Process. Syst. (NIPS)*, Montréal, Canada, Dec. 2017, pp. 465–476.