IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 5 October 2022, accepted 28 October 2022, date of publication 7 November 2022, date of current version 11 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3220337

RESEARCH ARTICLE

Dynamic Error Recovery Flow Prediction Based on Reusable Machine Learning for Low Latency NAND Flash Memory Under Process Variation

MINYOUNG HWANG[1], JEONGJU JEE[1], JOONHYUK KANG[1], (Member, IEEE), HYUNCHEOL PARK[1], (Senior Member, IEEE), SEONMIN LEE[2], AND JINYOUNG KIM[2]

[1]School of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 34141, South Korea
[2]Samsung Electronics, Hwaseong 18448, South Korea

Corresponding author: Hyuncheol Park (hcpark@kaist.ac.kr)

This work was supported by Samsung Electronics Co., Ltd. (1214-08152-01) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2020R1A2C1101158).

ABSTRACT NAND flash memory is becoming smaller and denser to have a larger storage capacity as technologies related to fine processes are developed. As a side effect of high-density integration, the memory can be vulnerable to circuit-level noise such as random telegraph noise, decreasing the reliability of the memory. Therefore, low-density parity-check code that provides multiple decoding modes is adopted in the NAND flash memory systems to have a strong error correcting capability. Conventional static error recovery flow (ERF) applies decoding modes sequentially, and read latency can increase when preceding decoding modes fail. In this paper, we consider a dynamic ERF using machine learning (ML) that predicts an optimal decoding mode guaranteeing successful decoding and minimum read latency and applies it directly to reduce read latency. Due to process variation incurred in the manufacturing of memory, memory characteristics are different by chips and it becomes difficult to apply a trained prediction model to different chips. Training the customized prediction model at each memory chip is impractical because the computational burden of training is heavy, and a large number of training data is required. Therefore, we consider ERF prediction based on reusable ML to deal with varying input and output relationships by chips due to process variation. Reusable ML methods reuse pre-trained model architecture or knowledge learned from source tasks to adapt the model to perform its task without any loss of performance in different chips. We adopt two reusable ML approaches for ERF prediction based on transfer learning and meta learning. Transfer learning method reuses the pre-trained model by reducing domain shift between a source chip and a target chip using a domain adaptation algorithm. On the other hand, meta learning method learns shared features from multiple source chips during the meta training procedure. Next, the meta-trained model reuses previously learned knowledge to fastly adapt to the different chips. Numerical results validate the advantages of the proposed methods with high prediction accuracy in multiple chips. In addition, the proposed ERF prediction based on transfer and meta learning can yield a noticeable reduction in average read latency as compared to conventional schemes.

INDEX TERMS NAND flash memory system, process variation, error management, reusable machine learning, transfer learning, meta learning.

I. INTRODUCTION

NAND flash memory is a type of nonvolatile memory that can read and write data electrically on flash cells. The flash cell is a metal–oxide–semiconductor field-effect transistor (MOSFET) with a floating gate (FG) completely surrounded by dielectrics [1]. Electron charges are trapped in FGs, and the corresponding voltage levels indicate stored data. Due to high data reliability and swift read/write operation, the NAND flash memory is widely used in many applications including

The associate editor coordinating the review of this manuscript and approving it for publication was Oussama Habachi.

smart devices, solid-state drives, and universal serial bus flash drives [2], [3], [4].

Recently, downscaling and multi-leveling technologies are applied to the NAND flash memory as demands for greater storage capacity increase [5], [6]. In addition, 3D NAND flash memory where cell arrays are vertically stacked is developed to cope with lithography problems and cell-to-cell interference of conventional 2D NAND flash memory [7], [8]. Besides capacity improvements from high-density integration, the reliability of the NAND flash memory can be significantly degraded.

The logical symbol data stored in a NAND flash memory is determined by the threshold voltages. If the number of levels of threshold voltage increases, the bit of symbol that can be stored in a single cell increases. In order to improve data capacity, the number of bits stored in a single cell increases while the flash cell size is reduced to densely arrange the memory [9]. Therefore, the distance between two threshold voltage distributions of adjacent symbols gets smaller becoming vulnerable to noise [10]. The increment on data retention time and program/erase (P/E) cycles generates additive noise that distorts or shifts the distribution of threshold voltages [11]. In addition, the threshold voltage distribution is different from memories due to process variations, even if the programmed symbol is the same. As a result, the error rate increases since the stored data becomes vulnerable to error on the highly degraded memory channel.

To enhance the reliability of the memory, error correction schemes such as read reference voltage readjustment [12] and error correcting codes (ECC) [13], [14] are applied to NAND flash memory. The memory controller detects the stored data symbol by sensing the voltage level of the cell using read reference voltage. Since the threshold voltage distribution is distorted as P/E cycles and retention time increases, sensing with fixed read reference voltage may result in high error rate. Therefore, the read reference voltage should be adjusted depending on the memory channel state. The optimal read reference voltage with minimal bit error can be set as an intersection of threshold voltage distributions of two adjacent symbol states. However, it is impossible to find such intersections since the knowledge of the threshold voltage distribution affected by process variations and additive noise cannot be perfectly obtained. Instead, the optimal read reference voltages are estimated by updating from default value using sentinel-cells-enabled method [15], read-retry method [16], valley tracking method [17], or machine learning (ML)-based method [18]. Although the read reference voltage estimation can reduce raw bit error rate (RBER), the inevitable bit error may occur. To improve error tolerance of flash memory, an ECC has been adopted in the NAND flash memory systems. The ECC can enhance data reliability through a method of encoding by adding parity bits to the message data. The study in [14] has reported that the conventional Bose-Chaudhuri-Hocquenghem (BCH) code is inadequate to handle continued technology scaling. Hence, an ECC such as low-density parity-check (LDPC) code has been adopted [14], [19] due to its strong error correction capability. The LDPC code decoder performs decoding through a process where variable nodes and check nodes iteratively change messages with each other [20]. The LDPC code supports hard-decision (HD) decoding mode and soft-decision (SD) decoding mode with strong error correcting capability.

The error correcting capability of the SD decoding mode is powerful. However, it requires additional sensing to obtain higher-precision log-likelihood ratio (LLR) information, thereby the read latency is increased. There have been intensive studies with various approaches to improve read performance by reducing read latency. The study in [21] has proposed algorithms to reduce read latency by estimating optimal read reference voltages with a limited number of re-reads. Similarly, [22] has proposed an adaptive soft sensing scheme that adjusts the number of sensing voltages by monitoring RBER in flash pages to decrease read latency. In [23], [24], nonuniform soft sensing read reference voltage has been demonstrated to decrease latency by achieving high error correcting capability with low precision decoding mode. The authors in [25] have introduced read-retry mechanism with consecutive retry steps in a pipelined manner to reduce the read latency. Moreover, they reduced page-sensing latency with an adaptive reference voltage precharging timing based on offline profiling. In [26], transparent lossless data compression has been adopted to reduce HD decoding failure probability. In addition, memory-to-controller data transfer latency has been reduced by applying entropy coding to memory sensing results. Next, [27] addressed an issue that the upper pages of the flash memory are subject to higher bit errors. They adopted unequal error correcting code where different codes with different redundancy by pages and proposed a partially concatenated coding design strategy to reduce latency in upper pages. In addition to the proposed methods introduced above, an efficient solution to reduce read latency is to skip unnecessary decoding processes.

Most of the existing literature has considered conventional static error recovery flow (ERF) which applies HD, 2-bit SD, and 3-bit SD decoding modes sequentially for error correction. However, if the memory channel is highly degraded, decoding mode with weak error correcting capability may fail and result in increased read latency. To reduce the unnecessary latency, dynamic ERF has been proposed that applies an optimal decoding mode guaranteeing successful decoding and minimum latency based on the memory channel states. The study in [28] has proposed a scheme to dynamically select decoding mode by monitoring RBER to minimize decoding iterations so that energy consumption and delay can be reduced. In [29], binary classification models, e.g., linear support vector classifier (SVC), random forest classifier, and naive Gaussian Bayes classifier, are applied to predict whether HD decoding mode may fail. The proposed schemes in [28] and [29] utilize RBER and 31 features observed in reading operations, respectively, to predict the proper decoding mode. In practice, a memory controller can only obtain limited information on the memory channel states. Therefore,

accurate predictions of the optimal decoding mode based on them are difficult. However, a classifier model based on a neural network (NN) can be utilized effectively in the prediction since it can be trained to infer complicated relations between memory channel states and corresponding optimal decoding mode from the limited evidence [30], [31]. Meanwhile, the ERF prediction based on ML has the problem that it is hard to apply a trained prediction model to different memory chips due to process variations.

Process variations including inter-wordline variation, inter-block variation, and inter-chip variation affect the memory channel states by distorting threshold voltage distributions. Since the effects of process variation on each individual memory are generally unpredictable, it is difficult to develop memory management schemes utilizing knowledge of the process variation in advance. Exceptionally, the authors in [9] have proposed a read reference voltage estimation scheme considering layer-to-layer (L2L) process variation in 3D NAND flash, which is known to be consistent among different blocks. The scheme utilizes a look-up table on pre-defined read reference voltage that can be generated by considering the consistency of L2L variation. However, not all types of process variations are always consistent, and the characteristics of memory may vary depending on manufacturing process. Next, conventional wear leveling schemes classify flash blocks based on the number of P/E cycles. P/E cycle is a direct measurement of the weariness of flash memory, but it cannot reflect the reliability difference caused by process variations. Therefore, [32], [33] have proposed PV-aware wear leveling that classifies and manages blocks based on different reliability measures, i.e., RBER.

The prediction of optimal decoding mode using the ML model can be inaccurate since the optimal output can be different even if the input feature is the same due to process variations. The prediction model trained on a memory chip would have poor prediction performance in another chip environment due to inter-chip variation. Training the customized model for each memory chip is infeasible in the large-scale manufacturing process.

Motivated by these observations, we propose dynamic ERF prediction based on reusable ML for low-latency NAND flash memory systems to adapt the model to multiple chips with different characteristics. To develop an adaptive method for flash memory systems, the computational burden required for data acquisition and model fine-tuning should be considered. Reusable ML methods reuse previously obtained knowledge or learned model parameters to adapt quickly to different chips without loss of prediction performance. Instead of training the prediction model from the bottom for a new chip, we consider two reusable ML approaches based on transfer learning and meta learning which allow the prediction model to be applied to a new chip. Transfer learning aims to improve the learning in a target task by reusing the pre-trained model in a source task [34]. In [35], an unsupervised transfer learning called Adversarial Discriminative Domain Adaptation (ADDA) has been proposed. ADDA utilizes Generative Adversarial Network (GAN)-based framework to adapt a pre-trained model to a new task by reducing domain shift between different tasks. Meta learning is often called ''learning to learn'' in that it can adapt to new data quickly from only a few examples integrating its prior experiences from old data while avoiding overfitting [36], [37]. Meta learning aims to obtain meta-learned parameters that are shared and reused among different tasks. A meta-trained model learns context parameters to adapt to a target task based on the shared parameters. Fast Context Adaptation VIA meta-learning (CAVIA) that can adapt very fast using a small number of data has been proposed in [38]. The adaptation of CAVIA to a new task was faster than conventional meta-learning methods, showing better performance as well.

In order to propose ERF based on reusable ML, we first model the memory channel considering process variations and additive noises. Next, the fundamental framework for the ERF prediction model using ML is formulated. Moreover, we suggest input features that are practically obtained in the memory controller and are effective to describe the memory channel state. Finally, ERF predictions methods based on transfer and meta learning are proposed and their performances are compared from experiments. The contributions of this paper are summarized as follows.

- We synthetically model channel states of NAND flash memory systems considering process variations and additive noise in order to generate data for experiments. The considered process variations include variations among wordlines, blocks, and chips which occur by physical differences. Furthermore, the additive noise which depends on P/E cycles and data retention is considered. Data generated with the formulated memory channel model can be effectively used in experiments for training and verifying the performance of proposed ERF prediction methods instead of real observation data that cannot be obtained easily.
- We propose a dynamic ERF prediction method based on conventional ML. The ML-based ERF prediction model learns the complex relationships between input features illustrating memory channel state and output class. However, the memory controller cannot obtain direct information on several memory degrading factors such as data retention or unpredictable process variations. Instead, on-cell ratio is introduced as auxiliary input features and their effectiveness was analyzed. Next, the output class is labeled as the optimal decoding mode guaranteeing successful decoding and minimum average read latency among HD, 2-bit SD, and 3-bit SD modes. We derive equations to calculate the average read latency for labeling the output class.
- We propose reusable ML methods to adapt the ERF prediction model to different NAND flash memory chips under process variations. The conventional ERF prediction model based on ML should be re-trained using a large number of data in the target memory chip. As reusable ML methods, we consider two approaches

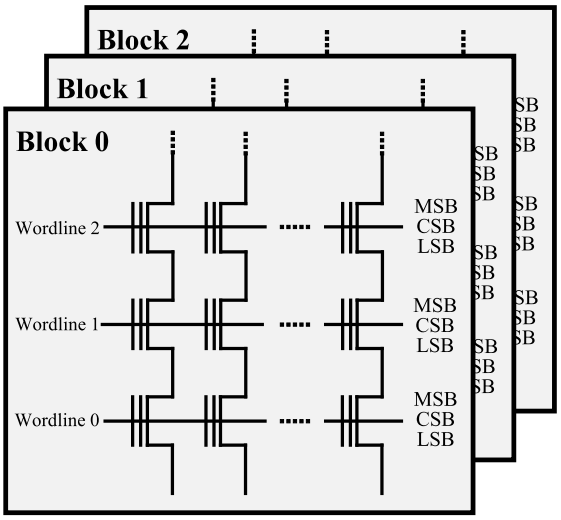


FIGURE 1. 3D NAND flash memory chip structure.

based on transfer and meta learning. First, we apply ADDA [35] which is an unsupervised transfer learning method to the ERF prediction. In ADDA, the model pre-trained on the source memory chip can be adapted to the target chip using a smaller number of data without labeling. Next, a fast meta learning method, CAVIA [38], is examined to be applied for the ERF prediction. CAVIA model learns its NN parameters from multiple source chip data. The meta-trained model obtains context variables in a target chip using a few data. The context variables contain information about inter-chip variation and are used as additional input features for the model.

The rest of this paper is organized as follows. In Section II, the system model of NAND flash memory system is described in detail. In Section III, we introduce a dynamic ERF prediction framework based on basic ML. Next, reusable ML methods are proposed. The experiment results and discussions on the proposed methods are presented in Section IV, followed by conclusions in Section V.

II. NAND FLASH MEMORY SYSTEM

A NAND flash memory chip is composed of thousands of two-dimensional arrays of flash cells, called blocks. Each block contains dozens of wordlines which are rows of flash cells where each of the wordlines contains 64K to 128K cells [9], [39]. Fig. 1 illustrates the basic structure of 3D NAND flash memory. The flash cell stores data in form of voltage levels. A voltage level corresponding to a data symbol is the threshold voltage. A single-level cell (SLC) NAND flash memory can store a 1-bit data symbol per cell with two levels of threshold voltages. A multi-level cell (MLC) and triple-level cell (TLC) NAND flash memory stores a 2 and 3-bit data symbol per cell respectively. In this paper, we consider a TLC NAND flash memory. In TLC memory, the threshold voltage of each cell belongs to one of 2^3 non-overlapping threshold voltage windows to represent a 3-bit data symbol. The three bits of each symbol state are divided into 3 separate pages depending on their logical locations, i.e., most significant bit (MSB), central significant bit (CSB), and least significant bit (LSB) pages.

A. BASIC OPERATIONS OF NAND FLASH MEMORY

There are three basic operations for a typical NAND flash memory system which are as follows [26], [39], [40], [41], [42].

1) Program: Programming is a process of storing data in memory cells. Electron charges are injected into a FG in a cell up to the target threshold voltage when sufficient positive voltage is applied to the control gate. As shown in Fig. 2, the programming is performed by a page unit in a wordline. A page usually stores multiple data frames.
2) Erase: The stored data can be erased by removing charges from a cell. After the erase operation, all cells are configured to erasure state, and the cells are ready to be reprogrammed. The erase operation is performed by a block unit.
3) Read: Data stored in cells is read by applying read reference voltages. To read n-bit data symbols, $2^n - 1$ read reference voltages are required. Fig. 3 shows the seven read reference voltages for TLC flash memory system. The sensing output is decoded by LDPC decoder. The read operation is performed by a wordline unit.

Fig. 4 describes the operation flow of the NAND flash memory system. Message data bits are encoded into codewords, and the codewords are mapped to 3-bit data symbols. Next, the codeword symbols are programmed into threshold voltages in memory cells. If distributions of threshold voltage for each symbol state are well separable, the data will be recovered intact by the read operation. However, the distributions of threshold voltage are distorted by the noise due to impairments of the memory. The read voltages are applied to sense stored data, and the sensed noisy codewords are corrected through a decoder. If the memory channel is highly degraded, which results in a high RBER, and hence decoding may fail.

B. MEMORY CHANNEL MODEL

The threshold voltage distribution of the cells programmed in the ith symbol state can be modeled as a Gaussian distribution [26].

$$f_{V_{\text{th},i}}(x) = \frac{1}{\sigma_i\sqrt{2\pi}} e^{-\frac{(x-\mu_i)^2}{2\sigma_i^2}}, \; i \in \{0, \ldots, 7\}, \tag{1}$$

where $\mu_i(w, N)$ and $\sigma_i(w, N)$ are mean and standard deviation respectively which are dependent on wordline index w and the number of P/E cycles N. This distribution changes due to process variations and additive noises. Process variations including inter-chip variation and inter-wordline variation alter the standard deviation of the threshold voltages. In addition, the

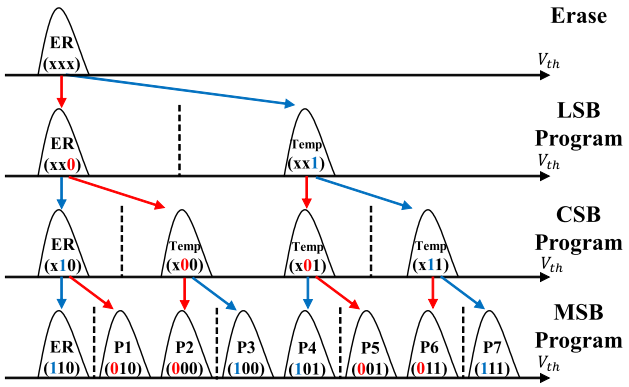


FIGURE 2. Programming of TLC NAND flash memory.

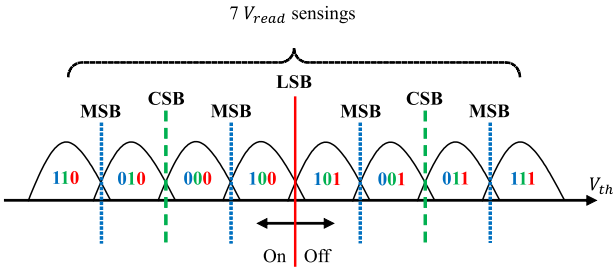


FIGURE 3. Hard-decision sensing of TLC NAND flash memory with 7 read reference voltages.

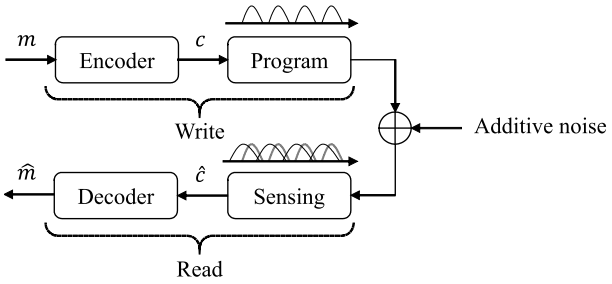


FIGURE 4. Basic operation flow of the NAND flash memory system.

additive noise comes from the increment in P/E cycles [43], [44] and data retention time [45], [46] distort the distribution.

1) PROCESS VARIATION

Process variation is a phenomenon in which manufacturing process error induces differences in threshold voltage distribution and reliability by chip/block/wordline. The main cause of process variation is random dopant fluctuation (RDF) and oxide trap fluctuation (OTF). The RDF is the random fluctuation of the location and density of impurities in the ion implantation process of the transistor [47], [48], [49]. The RDF is known as a major cause of the change in threshold voltage of MOSFET [47]. The RDF can be modeled as [48], [49]

$$\sigma_{\text{RDF}} = 3.19 \times 10^{-8} \frac{t_{\text{ox}} N_A^{0.4}}{\sqrt{WL}}, \tag{2}$$

where the parameters are defined on Table 1. Next, the OTF is a phenomenon caused by a defect in the structure of an oxide

TABLE 1. Parameters for RDF and OTF.

Parameter	Name	Range	Unit
N_A	Channel doping concentrations	$(1 \times 10^{18}, 5 \times 10^{18})$	$\text{cm}^{-}3$
Q_{ox}	Surface density of traps	10^{11}	cm^{-1}
t_{ox}	Oxide thickness	$(1, 6)$	nm
W	Channel width	$(50, 500)$	nm
L	Channel length	$(30, 100)$	nm

film [48], [49], [50]. The OTF can be modeled as follow [48].

$$\sigma_{\text{OTF}} = 10^{-6} \frac{t_{\text{ox}} \sqrt{Q_{\text{ox}}}}{\sqrt{WL}}, \tag{3}$$

where the parameters are given on Table 1 as well. As a result, the standard deviation of the threshold voltage affected by the RDF and the OTF is expressed by

$$\sigma^2_{\text{VTH}} = \sigma^2_{\text{RDF}}(t_{\text{ox}}, W, L) + \sigma^2_{\text{OTF}}(t_{\text{ox}}, W, L). \tag{4}$$

- Inter-chip variation: Parameters that affect σ_{VTH} such as oxide thickness, channel length, and channel width vary by different memory chips depending on which fragment of wafer it is made of. The oxide thickness variation across wafers is modeled by Gaussian distribution with a mean of 8 nm and a standard deviation of 0.0031 [51]. There are random and spatial variation for channel length and width. The random variations occur regardless of the chip location and are modeled as Gaussian distribution [52]. The random variation of channel width, ΔW_{rand} is modeled with truncated Gaussian distribution with a mean of 0 and a standard deviation of 0.25 where the range is $(-20, 20)$ [48]. Considering the random variation, the channel width is expressed by

$$W = \bar{W} + \Delta W_{\text{rand}}, \tag{5}$$

where $\bar{W} = 70$ nm. The spatial variation of channel length includes intra-wafer variation and intra-field variation. The intra-wafer variation is the difference in channel length by the location of the wafer that composes the chip. The intra-field variation is caused by differences in lithographic fields which are areas that can be patterned by a single device during the process. The intra-wafer and intra-field variations can be modeled as second-order polynomials as (6) and (7) respectively [52], [53].

$$\begin{aligned} \Delta L_w(x_w, y_w) = &-0.00055x_w^2 - 0.00021y_w^2 \\ &-0.0281x_w + 0.0137y_w \\ &-0.00009x_w y_w. \end{aligned} \tag{6}$$

$$\begin{aligned} \Delta L_f(x_f, y_f) = &-0.0019x_f^2 - 0.0264y_f^2 \\ &-0.0408x_f + 0.6225y_f, \end{aligned} \tag{7}$$

where (x_w, y_w) and (x_f, y_f) represent the coordinates on the wafer die and field respectively. As a result, the channel length is expressed by [52]

$$L = \bar{L} + \Delta L_w + \Delta L_f + \Delta L_{\text{rand}}, \tag{8}$$

where $\bar{L} = 50$ nm and ΔL_{rand} is truncated Gaussian random channel length variation with mean

of 0 and standard deviation of 0.0035 where the range is $(-20, 20)$ [48].

- Inter-block variation: There exists reliability difference among flash blocks in the same chip. The reliability difference that appeared in the rate of memory degradation is described by wearing degree [54]. If the wearing degree is high, the RBER increases faster as the number of P/E cycles or retention time increases. The probability density function of wearing degree s follows truncated Gaussian distribution as follows [32].

$$f_s(x) = \begin{cases} \dfrac{1}{\sigma_s\sqrt{2\pi}} e^{-\frac{(x-\mu_s)^2}{2\sigma_s^2}}, & \text{if } s_{\min} \le x \le s_{\max}, \\ 0, & \text{else}, \end{cases} \quad (9)$$

where $s_{\min} = 2.72 \times 10^{-4}$, $s_{\max} = 4.23 \times 10^{-4}$, $\mu_s = (s_{\min} + s_{\max})/2$, and $\sigma_s = 9 \times 10^{-5}$.

- Inter-wordline variation: The threshold voltage distribution is distorted due to interference between cells located in different wordlines. The RBER difference by the inter-wordline variation has been observed to be consistent across randomly-selected blocks [9]. Considering the inter-wordline variation, the standard deviation of threshold voltage distribution of the ith state, the wth wordline, and the kth chip is expressed by

$$\sigma_{k,w,i} = \sigma_{\text{std},w,i} \frac{\sigma_{\text{VTH},k}}{\sigma_{\text{std},w,0}}, \quad i \in \{0, \ldots, 7\}, \quad (10)$$

where $\sigma_{\text{std},w,i}$ is the standard deviation of the ith programmed state considering a multi-wordline threshold voltage distribution model and $\sigma_{\text{VTH},k}$ is the standard deviation of k-chip given by (4) considering inter-chip variation. The initial multi-wordline threshold voltage distributions for each state are decided by the NAND flash memory systems and programming schemes.

2) NOISE

The additive noise distorts the programmed threshold voltage distribution. The major cause of the additive noise is impairments of the memory due to increments in the number of P/E cycles and data retention time. The threshold voltage distribution distorted by additive noise is represented by

$$\tilde{f}_{V_{\text{th},i}}(x) = f_{V_{\text{th},i}}(x) * f_r(x) * f_t(x), \quad (11)$$

where $f_r(x)$ and $f_t(x)$ are the probability density function of random telegraph noise (RTN) and data retention respectively.

- RTN: As the number of programming and erasing operations increases, the memory is electrically worn so that the threshold voltage distribution is distorted. If the wearing degree of the memory is high, the distortion by the P/E cycles gets severer. The probability distribution of RTN can be modeled as a Laplacian distribution as follow [55].

$$f_r(x) = \frac{1}{2\lambda_r} e^{-\frac{|x-\mu_r|}{\lambda_r}}, \quad (12)$$

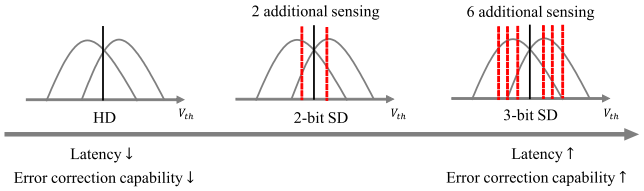


FIGURE 5. Decoding modes of LDPC.

where $\mu_r = C_r + A_r(N \cdot s)^{0.62}$, $\lambda_r = K_r(N \cdot s)^{0.5}$, $K_r = 4 \times 10^{-4}$, $C_r = 1.26 \times 10^{-3}$, $A_r = 1.8 \times 10^{-4}$, N is the number of P/E cycles.

- Data retention: Since the electron charges trapped on the FG of the cell leak out as time goes on, the data retention affects the threshold voltage. The noise due to data retention can be modeled as a Gaussian distribution as below [55].

$$f_t(x) = \frac{1}{\sigma_d\sqrt{2\pi}} e^{-\frac{(x-\mu_d)^2}{2\sigma_d^2}}, \quad (13)$$

where μ_d and σ_d are modeled as follows.

$$\mu_d = K_s(x - x_0)K_d(N \cdot s)^{0.5} \ln(1 + t), \quad (14)$$
$$\sigma_d^2 = K_s(x - x_0)K_m(N \cdot s)^{0.6} \ln(1 + t), \quad (15)$$

where $K_d = 4 \times 10^{-4}$, $K_m = 4 \times 10^{-4}$, $K_s = 0.333$, and $x_0 = 1.4$. Note that the degradation due to RTN and data retention is accelerated with large wearing degree s.

C. ERROR RECOVERY SCHEMES

The process variations and additive noise introduced in Section II-B distort the distribution of threshold voltage distribution by shifting or widening it. The recovered data is highly vulnerable to error when the memory channel is degraded by the impairments. To enhance data reliability from distortions, strong error recovery schemes should be applied to NAND flash memory systems [11].

1) READ REFERENCE VOLTAGE ESTIMATION

If the threshold voltage distributions are shifted, the controller should adjust the read reference voltage to improve the accuracy of the sensing operation. The read reference voltage is ideally set as the intersections between two adjacent symbol states' threshold voltage distributions. Since it is hard to obtain information on the intersection point, read reference voltages are estimated utilizing the methods introduced in [15], [16], [17], and [18]. In this paper, we assume that the optimal read reference voltages are obtained using one of the pre-mentioned methods.

2) ERROR CORRECTING CODES

To enhance data reliability by correcting bit errors, an ECC has been adopted. LDPC code is widely used in NAND flash memory systems due to its strong error correction capability to enhance data reliability. The LDPC code supports multiple decoding modes including HD decoding, 2-bit SD decoding, and 3-bit SD decoding. The HD decoding is performed with

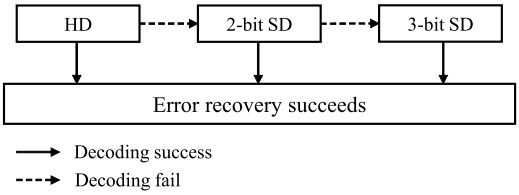


FIGURE 6. Static error recovery flow.

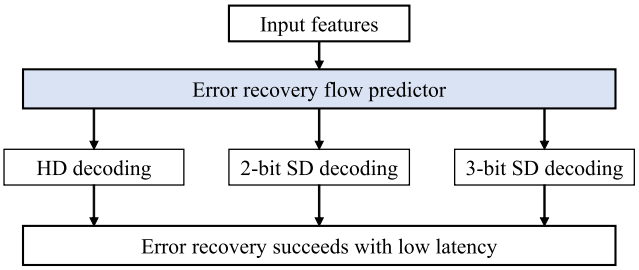
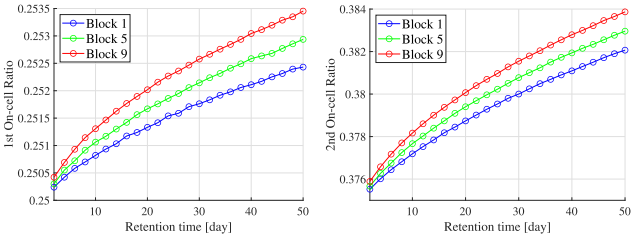


FIGURE 7. Dynamic error recovery flow.

one-bit log-likelihood ratio (LLR) information obtained by single sensing. On the other hand, The SD decoding manipulates soft information through additional sensing for higher error correction capability. n-bit SD decoding mode requires n bits LLR information through $2^n - 2$ additional sensings after HD sensing as shown in Fig. 5. The error correcting capability of the SD decoding is stronger but the latency increases due to the additional sensing operations.

3) ERROR RECOVERY FLOW

In a conventional NAND flash memory system with LDPC code, the ERF is static where the HD and SD decodings are sequentially applied without consideration of the memory channel state as shown in Fig. 6. If the process variation is weak and the additive noise is small, the HD decoding will succeed to decode. However, if the memory channel is heavily degraded by noise, the HD and 2-bit SD decoding may fail while 3-bit SD decoding succeeds. In this case, the HD decoding and 2-bit SD decoding induce unnecessary latency. To prevent this unnecessary latency, a dynamic ERF prediction scheme based on ML has been proposed in [30] and [31]. In dynamic ERF, the decoding mode that guarantees successful decoding and the smallest latency is predicted and applied immediately as shown in Fig. 7. Since the memory channel is affected by various physical factors, accurate inference based on a mathematical model is very difficult. Therefore, the ERF prediction model estimates the optimal decoding mode by utilizing limitedly obtainable evidence on the memory channel state. In addition, the relationships between such evidence and the corresponding optimal decoding mode are complex and non-linear. The ML prediction model can learn such complex input/output relationships through a training procedure so that the prediction can be accurate.



(a) The 1st on-cell ratio. (b) The 2nd on-cell ratio.

FIGURE 8. On-cell ratio by different memory blocks.

III. DYNAMIC ERROR RECOVERY FLOW PREDICTION BASED ON REUSABLE MACHINE LEARNING

A. DYNAMIC ERROR RECOVERY FLOW PREDICTION BASED ON MACHINE LEARNING

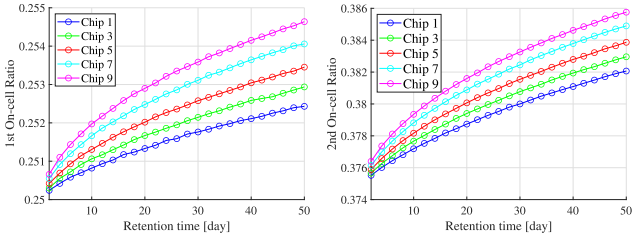
ERF prediction model based on ML can predict an optimal decoding mode that guarantees successful decoding with minimum average read latency by utilizing evidence on the memory channel state. The prediction model is basically an NN-based classifier. It observes input features that are evidence of the memory channel state and classify it to a corresponding class which indicates the optimal decoding mode.

Input features suffice for perfect prediction if it contains information about inter-chip variation, inter-block variation, inter-wordline variation, page index, P/E cycle, and data retention time. The P/E cycle and the page index are obtainable features in memory controller. The inter-wordline variation tends to be consistent in different blocks and it can be inferred from wordline index which is known to the controller. On the other hand, the memory controller is inaccessible to information on the inter-chip variation, inter-block variation, and retention time. As auxiliary information, we can use on-cell ratio which is the ratio of the cells turned on among all cells on the page when sensing with the predefined reference voltage [31]. In the TLC NAND flash memory, the on-cell ratio is defined as

$$r = \frac{\text{number of on cells at } V_{\text{fix}}}{\text{number of total cells}}, \tag{16}$$

where V_{fix} is a predefined reference voltage. The on-cell ratios are obtained from the sensing operation without any storage overhead by applying read reference voltages to a specific wordline for each block because variation between wordlines for each block tends to be consistent [9]. We configured two predefined reference voltages that corresponding on-cell ratios are well distinguished by different blocks and chips with respect to data retention time as shown in Fig. 8 and 9. Therefore, we choose the 1st and the 2nd on-cell ratios as alternative input features replacing inter-chip variation, inter-block variation, and data retention time. To summarize it, the input features for the prediction model are the number of P/E cycles N, the wordline index w, the page index p, the 1st on-cell ratio r_1, and the 2nd on-cell ratio r_2.

The objective of the ERF prediction is to find the optimal decoding mode. The optimal decoding mode is a decoding mode that minimizes average read latency with successful



(a) The 1st on-cell ratio. (b) The 2nd on-cell ratio.

FIGURE 9. On-cell ratio by different memory chips.

decoding among HD, 2-bit, and 3-bit modes. In order to train the ML-based ERF prediction model, we have to generate a training dataset. In a specific memory channel realization, we decode the data multiple times in three decoding modes and calculate the average read latency for each decoding mode. $\mathbb{E}[lat_{d,p}]$ denotes the average read latency for d-bit decoding mode of page p and it is derived in (17)–(28), as shown at the bottom of the next page. $d \in \{1, 2, 3\}$ represents the HD, 2-bit SD, and 3-bit SD modes and $p \in \{1, 2, 3\}$ denotes the LSB, CSB, and MSB respectively. Notations used in the equations are defined as follows.

- $\text{PER}_{d,p}$ is page error rates when the page p is decoded in the mode d and it can be derived as

$$\text{PER}_{d,p} = 1 - (1 - \text{FER}_{d,p})^{N_\text{f}}, \tag{29}$$

  where $\text{FER}_{d,p}$ is frame error rates and N_f is the number of data frames in a page.
- $\mathbb{E}[lat_{S_{\text{flow},p}}]$ is an average read latency of decoding page p in the flow S_flow, and example flows starting from HD is as follows.
  - S_1: The flow that error recovery succeeds in HD mode at once.
  - $S_{1\to2}$: The flow that error recovery fails in HD mode and succeeds in 2-bit SD mode.
  - $S_{1\to2\to3}$: The flow that error recovery fails up to 2-bit SD modes and succeeds in 3-bit SD mode.
  - $S_{1\to2\to3\to\text{fail}}$: The flow that error recovery fails in all modes.
- $\mathbb{E}[iter_d]$ is an average number of belief propagation iterations of successful decoding with mode d.
- MaxIter_HD and MaxIter_SD are the maximum number of iterations of HD decoding and SD decoding.
- t_sen is a sensing latency for sensing 3 pages of TLC NAND.
- t_dec is a latency per one decoding iteration.

From the derived average read latency, we can define the optimal decoding mode as follows.

$$d^\star = \arg\min_d \mathbb{E}[lat_{d,p}]. \tag{30}$$

Each set of input features is labeled with corresponding $d^\star$ to compose a single training data sample. A training data sample pair with input features and output class is given by $(x, d^\star)$ where $x = (N, w, p, r_1, r_2)$.

The ERF prediction model based on ML is designed as a multi-layer NN classifier model with L layers. This can be expressed as follows [56].

$$p(d|x, \theta) = \frac{\exp\left([f_{\theta^{(L-1)}}(\cdots f_{\theta^{(1)}}(x))]_d\right)}{\sum_{d'\in\{1,2,3\}} \exp\left([f_{\theta^{(L-1)}}(\cdots f_{\theta^{(1)}}(x))]_{d'}\right)}, \tag{31}$$

where $f_{\theta^{(l)}}(x) = \sigma(W^{(l)}x + b^{(l)})$ is the lth layer of NN with non-linear activation function $\sigma(\cdot)$, e.g., a Rectified Linear Unit (ReLU) or a hyperbolic tangent function. $\theta = \{\theta^{(l)}\}_{l=1,\ldots,L-1}$ is the parameters of the model where $\theta^{(l)} = \{W^{(l)}, b^{(l)}\}$ with weight matrix $W^{(l)}$ and bias vector $b^{(l)}$. The notation $[\cdot]_d$ stands for the element regarding d. The last layer of the network is a softmax function. The parameters of the model are trained using stochastic gradient descent (SGD) method as follows.

$$\theta \leftarrow \theta + \alpha\nabla_\theta\mathcal{L}_\mathcal{D}(\theta), \tag{32}$$

where α is a learning rate, $\mathcal{L}$ is a loss function, and $\mathcal{D}$ is a dataset composed of data pairs $(x, d^\star)$. In this paper, we use cross-entropy loss function which is defined by

$$\begin{aligned}\mathcal{L}_\mathcal{D}(\theta) &= \mathbb{E}_{x\sim\mathcal{D}}\left[-\log p(d^\star|x,\theta)\right] \\ &= -\sum_{(x,d^\star)\in\mathcal{D}} \log p(d^\star|x,\theta).\end{aligned} \tag{33}$$

The prediction model based on ML showed prediction accuracy of about 94 % in a single chip environment in [31] so that the read latency was reduced considerably. On the other hand, the prediction accuracy across different chips is expected to achieve poor accuracy due to process variation. This is because input and output relationships are different by chips, thereby a model trained on a source chip cannot predict the correct output in a target chip. To resolve this problem and to develop a prediction model that can be generally used in multiple chips via fast adaptation, we consider reusable ML methods that include transfer and meta learning. Transfer and meta learning perform adaptation procedures based on reused features to counteract process variation differ by memory chips. Since the adaptation is performed on-chip in general, methods with low complexity for adaptation should be selected. In that sense, we propose ADDA as a transfer learning method and CAVIA as a meta learning method. ADDA has an advantage in that it does not require labeling on target chip data. Moreover, CAVIA is favorable because it requires a small number of data in the adaptation.

B. PROPOSED DYNAMIC ERROR RECOVERY FLOW PREDICTION BASED ON TRANSFER LEARNING

Conventional ML-based ERF prediction model trained on a source chip may not be applicable to a target chip due to inter-chip variation. In order to apply the model in another chip, say a target chip, it must be re-trained using a large number of data to adapt parameters for accurate prediction. On the other hand, transfer learning allows knowledge transfer from a source chip and applies to the prediction in a target chip [34]. Especially, transductive transfer learning is widely used when the tasks in source and target are the same but the domains of data are different. Domain adaptation, which is

one method of transductive transfer learning, can be usefully adopted in our system. By applying domain adaptation, the domain shift between source chip data and target chip data is reduced so that the performance of the pre-trained model for the target chip is improved [57]. In [35], ADDA has been proposed unsupervised domain adaptation using GAN-based structure. The labeling data is time-consuming since the output class for the input is calculated from (17)-(19) using multiple decoding results. ADDA is very advantageous for our task since domain adaptation of target chip data can be done without class labeling. Therefore, we propose an ERF prediction model based on ADDA.

ERF prediction model based on ADDA is composed of two parts, i.e., an encoder and a classifier. The encoder and the classifier cascaded to compose a full prediction model, and they are designed as multi-layer NNs as in (34) and (35).

$$f_{\theta_E}(x) = f_{\theta_E^{(L_E)}}(\cdots f_{\theta_E^{(1)}}(x)) = x_E, \tag{34}$$

$$p(d|x_E, \theta_C) = \frac{\exp\left([f_{\theta_C^{(L_C-1)}}(\cdots f_{\theta_C^{(1)}}(x_E))]_d\right)}{\sum_{d'\in\{1,2,3\}} \exp\left([f_{\theta_C^{(L_C-1)}}(\cdots f_{\theta_C^{(1)}}(x_E))]_{d'}\right)}, \tag{35}$$

where θ_E and θ_C are parameters of the encoder and the classifier respectively. The encoder process the input feature data and delivers it to the classifier. The classifier generates the probability of output classes with the softmax function based on the encoded data x_E. The classifier is pre-trained on the source chip and reused on the target chip without an adaptation process. On the other hand, the encoder is adapted to the target chip by adversarial learning against the source encoder. In addition, the adaptation process utilizes a discriminator that is designed as a classifier with two classes as follows.

$$p(e|x_E, \theta_D) = \frac{\exp\left([f_{\theta_D^{(L_D-1)}}(\cdots f_{\theta_D^{(1)}}(x_E))]_e\right)}{\sum_{e'\in\{0,1\}} \exp\left([f_{\theta_D^{(L_D-1)}}(\cdots f_{\theta_D^{(1)}}(x_E))]_{e'}\right)}. \tag{36}$$

The discriminator distinguishes which encoder has generated the input encoded data. Therefore, the output class label $e = \{0, 1\}$ indicates the target encoder, i.e., 0, and the source encoder, i.e., 1, respectively. ERF prediction based on ADDA has three phases as shown in Fig. 10.

In the pre-training phase, a cascaded network model with a source encoder and a classifier is trained using a labeled source chip dataset $\mathcal{D}_\mathcal{S}$. The loss function for the training is cross entropy loss as described in (33). The parameters of the network are learned with SGD method in (32).

In domain adaptation phase, a target encoder and a discriminator are trained adversarially using unlabeled data. Source chip data is encoded using the pre-trained source encoder and target chip data is encoded by the target encoder. The discriminator receives the encoded data and distinguishes whether the data is from the source or the target encoder. The learning is adversarial since the target encoder learns to deceive the discriminator while the discriminator learns

$$\begin{aligned}\mathbb{E}[lat_{1,p}] = {} & (1-\mathrm{PER}_{1,p})\mathbb{E}[lat_{S_{1,p}}] + \mathrm{PER}_{1,p}(1-\mathrm{PER}_{2,p})\mathbb{E}[lat_{S_{1\to2},p}] \\ & + \mathrm{PER}_{1,p}\mathrm{PER}_{2,p}(1-\mathrm{PER}_{3,p})\mathbb{E}[lat_{S_{1\to2\to3},p}] + \mathrm{PER}_{1,p}\mathrm{PER}_{2,p}\mathrm{PER}_{3,p}\mathbb{E}[lat_{S_{1\to2\to2\to3\to\mathrm{fail}},p}],\end{aligned} \tag{17}$$

$$\begin{aligned}\mathbb{E}[lat_{2,p}] = {} & (1-\mathrm{PER}_{2,p})\mathbb{E}[lat_{S_{2,p}}] + \mathrm{PER}_{2,p}(1-\mathrm{PER}_{3,p})\mathbb{E}[lat_{S_{2\to3},p}] \\ & + \mathrm{PER}_{2,p}\mathrm{PER}_{3,p}\mathbb{E}[lat_{S_{2\to3\to\mathrm{fail}},p}],\end{aligned} \tag{18}$$

$$\mathbb{E}[lat_{3,p}] = (1-\mathrm{PER}_{3,p})\mathbb{E}[lat_{S_{3,p}}] + \mathrm{PER}_{3,p}\mathbb{E}[lat_{S_{3\to\mathrm{fail}},p}], \tag{19}$$

$$\mathbb{E}[lat_{S_{1,p}}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) + (t_{\mathrm{dec}} \times \mathbb{E}[iter_1] \times N_{\mathrm{f}}), \tag{20}$$

$$\mathbb{E}[lat_{S_{1\to2,p}}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) \times 3 + (t_{\mathrm{dec}} \times \mathrm{MaxIter}_{\mathrm{HD}} \times N_{\mathrm{f}}) + (t_{\mathrm{dec}} \times \mathbb{E}[iter_2] \times N_{\mathrm{f}}), \tag{21}$$

$$\mathbb{E}[lat_{S_{1\to2\to3,p}}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) \times 7 + (t_{\mathrm{dec}} \times (\mathrm{MaxIter}_{\mathrm{HD}} + \mathrm{MaxIter}_{\mathrm{SD}}) \times N_{\mathrm{f}}) + (t_{\mathrm{dec}} \times \mathbb{E}[iter_3] \times N_{\mathrm{f}}), \tag{22}$$

$$\mathbb{E}[lat_{S_{1\to2\to3\to\mathrm{fail}},p}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) \times 7 + (t_{\mathrm{dec}} \times (\mathrm{MaxIter}_{\mathrm{HD}} + 2 \cdot \mathrm{MaxIter}_{\mathrm{SD}}) \times N_{\mathrm{f}}), \tag{23}$$

$$\mathbb{E}[lat_{S_{2,p}}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) \times 3 + (t_{\mathrm{dec}} \times \mathbb{E}[iter_2] \times N_{\mathrm{f}}), \tag{24}$$

$$\mathbb{E}[lat_{S_{2\to3,p}}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) \times 7 + (t_{\mathrm{dec}} \times \mathrm{MaxIter}_{\mathrm{SD}} \times N_{\mathrm{f}}) + (t_{\mathrm{dec}} \times \mathbb{E}[iter_3] \times N_{\mathrm{f}}), \tag{25}$$

$$\mathbb{E}[lat_{S_{2\to3\to\mathrm{fail}},p}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) \times 7 + (t_{\mathrm{dec}} \times 2 \cdot \mathrm{MaxIter}_{\mathrm{SD}} \times N_{\mathrm{f}}), \tag{26}$$

$$\mathbb{E}[lat_{S_{3,p}}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) \times 7 + (t_{\mathrm{dec}} \times \mathbb{E}[iter_3] \times N_{\mathrm{f}}), \tag{27}$$

$$\mathbb{E}[lat_{S_{3\to\mathrm{fail}},p}] = (\frac{t_{\mathrm{sen}}}{7} \times 2^{p-1}) \times 7 + (t_{\mathrm{dec}} \times \mathrm{MaxIter}_{\mathrm{SD}} \times N_{\mathrm{f}}). \tag{28}$$

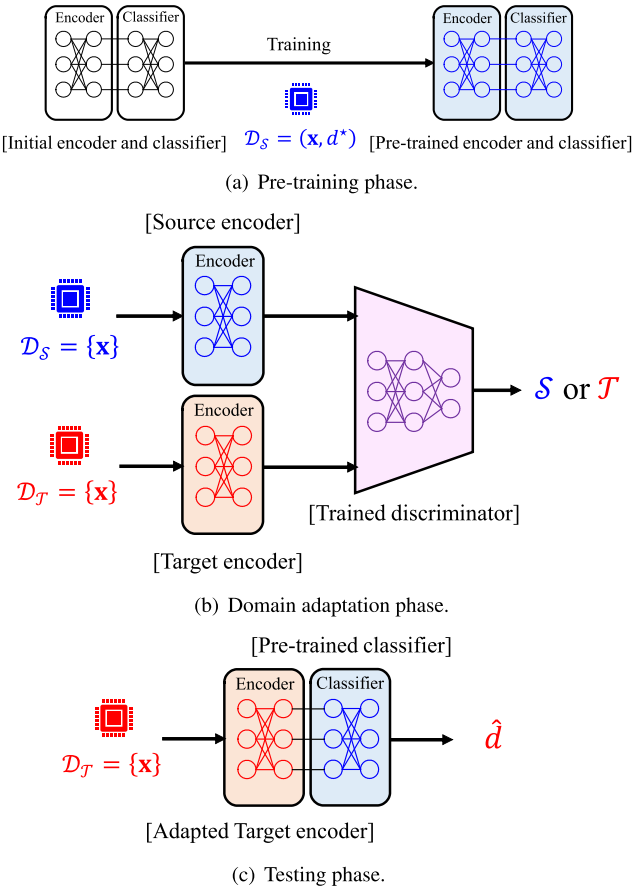


FIGURE 10. ERF prediction based on ADDA.

to distinguish better. If the target encoder is well-trained from adversarial learning, the output of the target encoder is indistinguishable from the output of the source encoder. As a result, the output of the target encoder can be well classified with the pre-trained classifier. In order to overcome the vanishing gradients problem, the discriminator employs a least-squares loss function [58]. The least square loss for the discriminator and the target encoder is described in (37) and (38) where $E_\mathcal{S}$ and $E_\mathcal{T}$ stands for the source encoder and target encoder respectively.

$$\mathcal{L}_{\text{adv}}(\theta_D) = \frac{1}{2}\mathbb{E}_{x\sim\mathcal{D}_\mathcal{S}}\left[\left\{\arg\max_e p(e|f_{\theta_{E_\mathcal{S}}}(x), \theta_D) - 1\right\}^2\right], + \frac{1}{2}\mathbb{E}_{x\sim\mathcal{D}_\mathcal{T}}\left[\left\{\arg\max_e p(e|f_{\theta_{E_\mathcal{T}}}(x), \theta_D)\right\}^2\right] \tag{37}$$

$$\mathcal{L}_{\text{adv}}(\theta_{E_\mathcal{T}}) = \mathbb{E}_{x\sim\mathcal{D}_\mathcal{T}}\left[\left\{\arg\max_e p(e|f_{\theta_{E_\mathcal{T}}}(x), \theta_D) - 1\right\}^2\right]. \tag{38}$$

The parameters of the discriminator, i.e., θ_D and the parameters of the target encoder, i.e., $\theta_{E_\mathcal{T}}$ are learned through the SGD method as described in (32).

In the testing phase, the domain adapted target encoder and pre-trained classifier are cascaded as an ERF prediction model. The target chip data is processed with the target encoder and the following classifier predicts the corresponding optimal decoding mode. Algorithm 1 provides a summary of the procedures.

Algorithm 1: ERF Prediction Based on ADDA

Input : Labeled source chip dataset $\mathcal{D}_\mathcal{S} = \{(x, d^\star)_n\}_{n=1,\ldots,N_\mathcal{S}}$ and unlabeled target chip dataset $\mathcal{D}_\mathcal{T} = \{(x)_n\}_{n=1,\ldots,N_\mathcal{T}}$

Output : Source encoder network parameters $\theta_{E_\mathcal{S}}$, Classifier network parameters θ_C, Target encoder network parameters $\theta_{E_\mathcal{T}}$, and Discriminator network parameters θ_D.

`*Pre-training phase*`

Initialize: Parameters $\theta_{E_\mathcal{S}}$ and θ_C

Define a ERF prediction network model by cascading encoder network with $\theta_{E_\mathcal{S}}$ and classifier network with θ_C

for *Pre-training iteration number* **do**
  Draw labeled data $(x, d^\star)$ from $\mathcal{D}_\mathcal{S}$
  Calculate cross entropy loss via (33)
  Update parameters $\theta_{E_\mathcal{S}}$ and θ_C via (32)
end

return $\theta_{E_\mathcal{S}}$ and θ_C with minimum training loss $\mathcal{L}_{\mathcal{D}_\mathcal{S}}(\theta_{E_\mathcal{S}}, \theta_C)$

`*Domain adaptation phase*`

Initialize: Parameters $\theta_{E_\mathcal{T}}$ and θ_D

for *Domain adaptation iteration number* **do**
  Draw unlabeled data x from $\mathcal{D}_\mathcal{S}$ and $\mathcal{D}_\mathcal{T}$
  Calculate least squares losses via (37)-(38)
  Update parameters $\theta_{E_\mathcal{T}}$ and θ_D via (32)
end

return $\theta_{E_\mathcal{T}}$ with minimum adversarial training loss $\mathcal{L}_{\text{adv}}(\theta_{E_\mathcal{T}})$

`*Testing phase*`

Predict optimal decoding mode of $x \in \mathcal{D}_\mathcal{T}$

C. PROPOSED DYNAMIC ERROR RECOVERY FLOW PREDICTION BASED ON META LEARNING

Meta learning, or learning to learn, refers to learning algorithms whose performance at each task improves with experience from the related tasks [36]. Meta learning aims to acquire an inductive bias that can be leveraged to learn a new task. Unlike transfer learning where the knowledge obtained on the source task is directly transferred to the target task, meta learning learns the bias in multiple source tasks. The inductive bias that is shared and reused among different tasks is called shared variable. With the baseline of shared variable, a meta trained model tries to adapt to a new task by acquiring context variable. Authors in [38] proposed CAVIA which provides a faster adaptation with a small number of data. The context variable in CAVIA is additional input of the model, while shared variables are characterized as the model parameters reused in multiple tasks. We applied CAVIA to revise the difference between prediction tasks caused by process

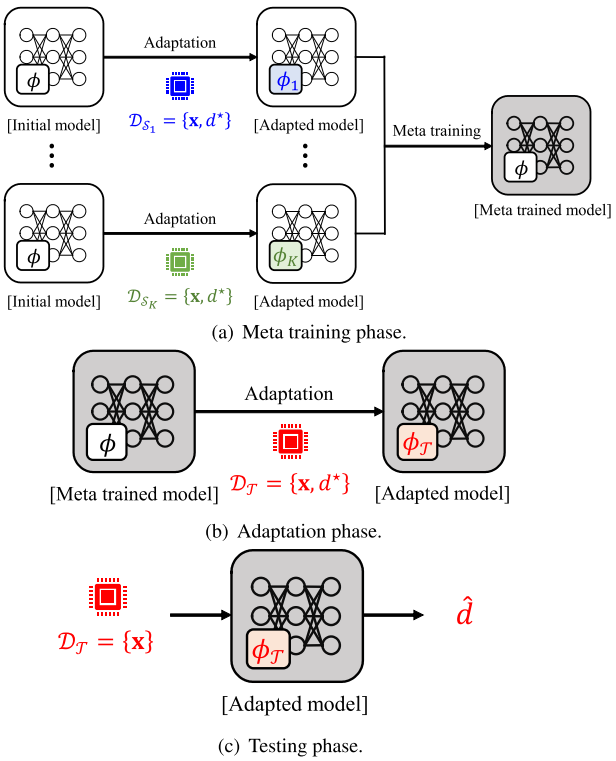


FIGURE 11. ERF prediction based on CAVIA.

variation by learning the context variable as an indicator of the variation.

ERF prediction model based on CAVIA is designed as a multi-layer NN as described in (31). CAVIA receives context variable ϕ as an addition input to the model, so that concatenated input vector becomes $\bar{x} = [x, \phi]$. Parameters of CAVIA network are learned in meta training procedure and are shared among different chips. Before applying the prediction model to the target chip, the context variables are updated to adapt the model. Three phases of ERF prediction based on CAVIA are shown in Fig. 11.

In the meta training phase, the model network parameter θ is trained using data from multiple chips. Given K of source chips and the dataset $\{\mathcal{D}_{\mathcal{S}_1}, \ldots, \mathcal{D}_{\mathcal{S}_K}\}$, context variables for each chip are adapted followed by a model parameter update. In every meta training iteration, the context variables for every chip are initialized, i.e., $\phi_k = 0, \forall k = \{1, \ldots, K\}$. For the kth chip, the context variable is updated with one or more SGD updating as described as follows.

$$\phi_k \leftarrow \phi_k - \eta \nabla_{\phi_k} \mathcal{L}_{\mathcal{D}_{\mathcal{S}_k}}(\theta), \tag{39}$$

where η is the learning rate of context variable updating and $\mathcal{L}_{\mathcal{D}_{\mathcal{S}_k}}$ is the loss function which can be obtained as in (33). After the context variable adaptation, the gradients on loss function with respect to θ are calculated for all chips. The model network parameter θ is updated with the average of the obtained gradients as follows.

$$\theta \leftarrow \theta - \alpha \nabla_{\theta} \sum_{k=1}^{K} \mathcal{L}_{\mathcal{D}_{\mathcal{S}_k}}(\theta). \tag{40}$$

In this way, the model learns common patterns in input/output relationships from multiple chips during the meta training phase.

In the adaptation phase, the parameter θ is fixed, and the context variable $\phi_{\mathcal{T}}$ is updated using data on the target chip $\mathcal{D}_{\mathcal{T}}$ with SGD updates as

$$\phi_{\mathcal{T}} \leftarrow \phi_{\mathcal{T}} - \eta \nabla_{\phi_{\mathcal{T}}} \mathcal{L}_{\mathcal{D}_{\mathcal{T}}}(\theta). \tag{41}$$

Algorithm 2: ERF Prediction Based on CAVIA

Input : Multiple labeled source chip datasets $\{\mathcal{D}_{\mathcal{S}_1}, \ldots, \mathcal{D}_{\mathcal{S}_K}\}$ and labeled target chip dataset $\mathcal{D}_{\mathcal{T}} = \{(x, d^{\star})_n\}_{n=1,\ldots,N_{\mathcal{T}}}$

Output : Meta-trained network parameters θ and context variable for target chip $\phi_{\mathcal{T}}$.

```
*Meta training phase*
```

Initialize: Parameters θ and ϕ_k for $k = 1, \ldots, K$

for *Meta training iteration number* **do**

  for *K source chips* **do**

    Initialize: Context variable ϕ_k

    for *Adaptation iteration number* **do**

      Draw labeled data $(x, d^{\star})$ from $\mathcal{D}_{\mathcal{S}_k}$

      Calculate cross entropy loss with respect to ϕ_k via (33)

      Update context variable ϕ_k via (39)

    end

    Calculate cross entropy loss with respect to θ via (33)

  end

  Update parameter θ via (40)

end

return θ with minimum training loss $\sum_{k=1}^{K} \mathcal{L}_{\mathcal{D}_k}(\theta)$

```
*Adaptation phase*
```

Initialize: Context variable $\phi_{\mathcal{T}}$

for *Adaptation iteration number* **do**

  Draw data $(x, d^{\star})$ from $\mathcal{D}_{\mathcal{T}}$

  Calculate cross entropy loss with respect to $\phi_{\mathcal{T}}$ via (33)

  Update context variable $\phi_{\mathcal{T}}$ via (41)

end

return updated context variable $\phi_{\mathcal{T}}$

```
*Testing phase*
```

Predict optimal decoding mode of $x \in \mathcal{D}_{\mathcal{T}}$

In the testing phase, the adapted prediction model can predict the optimal decoding mode for the input features concatenated with the context variable. Algorithm 2 provides a summary of CAVIA learning procedures [56].

The major drawback of CAVIA compared to ADDA is that CAVIA requires labeled data for the adaptation phase. However, labeled data can be inaccessible in a target chip due to high labeling costs. As an alternative approach, we can label input features with decoding results by applying static ERF and use them for the adaptation phase. After several static read operations in the memory chips, the memory controller can obtain sample data pairs composed of input features and

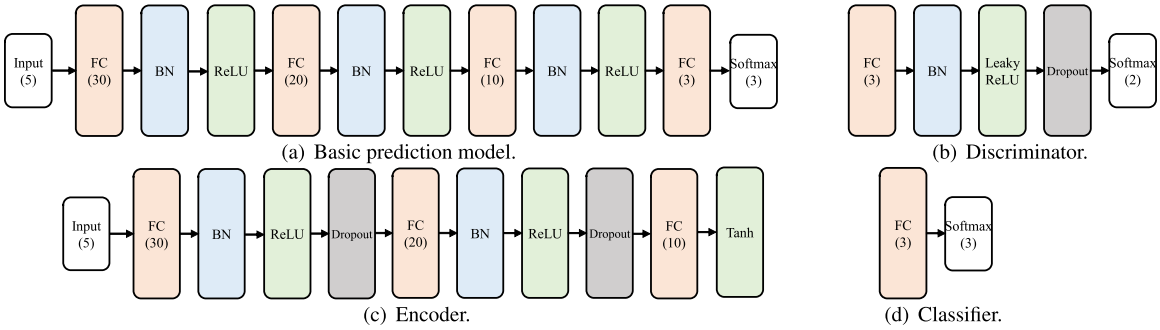


FIGURE 12. Neural network structures.

a decoding mode with the smallest read latency, i.e., $(x, \tilde{d}^{\star})$ where $\tilde{d}^{\star} = \arg\min_d lat_d$. CAVIA can use these data labeled with a decoding mode with the smallest read latency, not average, for the adaptation.

IV. EXPERIMENTS AND DISCUSSIONS

In this section, we provide experiment results to show the effectiveness of the proposed dynamic ERF prediction based on reusable ML methods. To compare the performance of the proposed methods, we consider the following baselines.

- Iteration count-based model (IC): IC selects decoding mode with minimal decoding iterations based on the RBER [28]. Decision boundaries on selecting decoding mode are optimized offline by collectively profiling the relation between RBER and successful decoding iteration number from data of multiple source chips.
- Support Vector Classifier-based model (SVC): SVC with polynomial kernels is applied for ERF prediction by referring to [29]. The task in [29] is a binary classification that determines whether HD decoding may fail or succeed, and information on the input features for the SVC model is not provided. For a fair comparison, we modified the SVC model to perform the same ERF prediction task introduced in Section II-C3. SVC is optimized on multiple source chips collectively.
- NN model Trained on Target (TT): The multi-layer NN classifier-based ERF prediction model described in Section III-A is trained on a target chip and applied to the same chip for testing. The performance of TT can be regarded as performance without process variations.
- NN model trained on Single Source (SS): The NN model is trained on a single source chip and applied to a target chip for testing. SS shows degraded prediction accuracy by process variations.
- NN model trained on Multi Source (MS): The NN model is trained on multiple source chips and applied to a target chip for testing. MS shows the performance when the model is trained with data acquired from multiple chips without adaptation.

We compose prediction models of each method as shown in Fig. 12. A basic NN-based ERF prediction model consists of four fully connected layers, of which the number of neurons is 30, 20, 10, and 3 respectively. The network adopts ReLU activation functions and batch normalization for regularization. The basic model can be trained by TT, SS, or MS methods. The network for CAVIA is the same as the basic network, except CAVIA network does not have batch normalization. Fig. 12(b)-(d) illustrates the network structure of the encoder, classifier, and discriminator for ADDA-based ERF prediction model, respectively. The network in Fig. 12(a) is divided into encoder and classifier, and dropout layers with a ratio of 0.25 are added to reduce overfitting problems. In the discriminator, the activation function of the hidden layer is set as a Leaky ReLU function with a negative slope of 0.2.

Other numerical details are as follows. The data mini-batch sizes for pre-training of ADDA, domain adaptation of ADDA, meta training of CAVIA, and training of baseline methods are set as 128. The number of data used in the adaptation phase of CAVIA is 100. For domain adaptation in ADDA, the learning rate of SGD updates for encoder and discriminator is 0.0002. All the other SGD updates adopt a learning rate of 0.01. For the experiments, we have generated a hundred standard deviations of threshold voltage distribution σ_{VTH} considering inter-chip variations as described in Section II. The generated σ_{VTH} values are sorted in ascending order and nine sample values have been selected by the steps of 10 % in the total range of values. A multi-chip dataset for experiments has been prepared from the selected σ_{VTH} values, i.e., $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_3, \mathcal{C}_5, \mathcal{C}_7, \mathcal{C}_9\}$. We extracted initial threshold voltage distributions for multi-wordline memory channel by emulating a 64-stack TLC vertical NAND (V-NAND), following the similar process in [59].[1] Next, nine sample values of wearing degrees s for blocks are generated by (9),and 1st, 5th, and 9th values are selected for experiments.

First, we evaluate the prediction accuracy by different methods for ERF prediction. Table 2 shows the prediction accuracy by different methods for ERF-prediction. IC is designed to achieve energy efficiency but without consideration of process variation. Therefore, the prediction accuracy of IC is the worst since it cannot predict optimal decoding

[1] Due to the security issue of the NAND flash memory manufacturer, unfortunately, we are not able to disclose information for specific performance and parameters of realization. In this paper, nevertheless, we validate the performance of the proposed framework over the emulated memory channel model shows high accuracy in the conformity assessment with the real 64-stack TLC V-NAND.

TABLE 2. Prediction accuracy (%) comparison by different methods for ERF-prediction.

Method	Source Chip (Training)	Target Chip (Test)					
		1	3	5	7	9	Avg
IC [28]	1, 3, 5 ,7 ,9	81.8	77.8	74.1	71.0	71.1	75.2
SVC [29]	1, 3, 5 ,7 ,9	88.2	91.2	93.2	88.3	77.2	87.6
TT	On target	95.6	93.3	93.4	91.5	92.0	93.2
SS	5	76.7	82.6	-	89.8	77.7	81.7
MS	1, 3, 5, 7, 9	82.5	86.6	91.2	90.7	80.9	86.4
ADDA	5	95.3	91.9	-	89.3	87.2	90.9
CAVIA	1, 3, 5, 7, 9	90.2	92.3	93.6	92.7	92.0	92.2

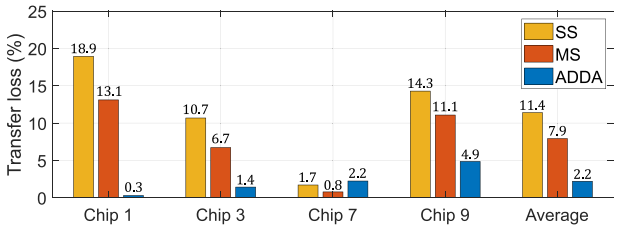


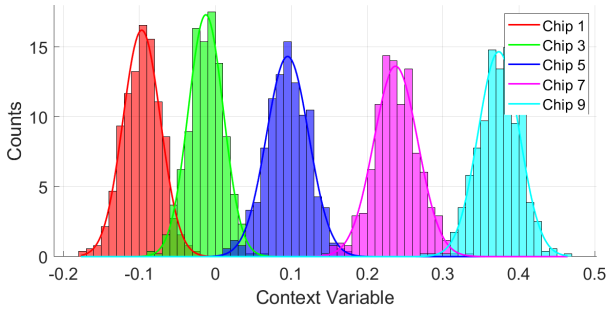
FIGURE 13. Transfer loss (%) by different methods.

mode under varying memory channel states due to process variations. The accuracy deviation by chips is large, showing no adaptability on inter-chip variation. Similarly, decision boundaries of SVC optimized collectively on multiple source chips are not adaptive, so the prediction accuracy fluctuates by target chips under process variations. TT method shows the highest prediction accuracy since it is trained and tested on the same chip environment, i.e., without process variation, which is infeasible in reality SS method trains the model with source chip $\mathcal{C}_5$ and applies to different chips. The prediction accuracy of SS is the worst among ML-based methods since the model is highly optimized to its source chip. Especially, the degradation of prediction accuracy is severe in $\mathcal{C}_1$ and $\mathcal{C}_9$ since the σ_{VTH} is highly different from the source chip $\mathcal{C}_5$. MS method trains the prediction model with multiple source chips. However, the basic ML-based model cannot learn to distinguish the input/output relationships by all different chips. Therefore, the prediction of MS method is inaccurate even though the testing is done on the target chip included in the source chip set $\mathcal{C}$. ADDA and CAVIA methods can predict optimal decoding mode with over 90 % accuracy on average since ADDA and CAVIA can adapt to a target chip through their adaptation methods. ADDA is pre-trained on source chip $\mathcal{C}_5$ and is adapted to target chips by domain adaptation procedure. In particular, the domain adaptation of ADDA has an advantage in that it does not require labeled data on the adaptation phase, unlike other methods. On the other hand, CAVIA requires labeled data in both off-line meta training and on-line context variable adaptation phase. Nevertheless, the number of data required in the adaptation is much smaller than ADDA. In addition, the performance represented by prediction accuracy in the target chip is higher.

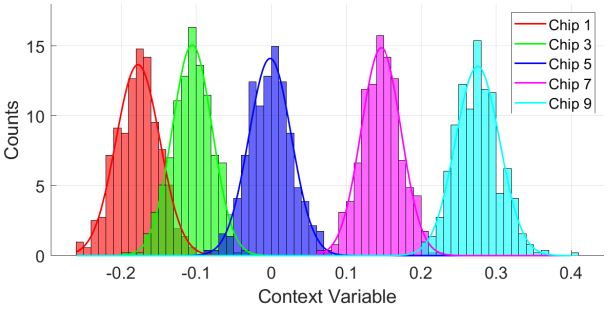
The knowledge transfer performance of ADDA can be shown with the transfer loss as shown in Fig. 13. The transfer loss is a metric about how much the trained model lack information about the target domain [60]. Transfer loss is calculated with transfer error and in-domain error,

TABLE 3. Prediction accuracy (%) of CAVIA by source chips.

Source Chip (Training)	Target Chip (Test)					
	1	3	5	7	9	Avg
1, 3, 5, 7, 9	90.2	92.3	93.6	92.7	92.0	92.2
1, 5, 9	90.3	91.6	93.5	92.5	91.6	91.9
3, 7	89.2	90.6	93.4	92.3	90.5	91.2



(a) CAVIA meta trained with $\{\mathcal{C}_1, \mathcal{C}_3, \mathcal{C}_5, \mathcal{C}_7, \mathcal{C}_9\}$.



(b) CAVIA meta trained with $\{\mathcal{C}_3, \mathcal{C}_7\}$.

FIGURE 14. Context variables of CAVIA obtained after adaptation phases.

i.e., $t_e(\mathcal{S}, \mathcal{T}) = e(\mathcal{S}, \mathcal{T}) - e(\mathcal{T}, \mathcal{T})$. The transfer error $e(\mathcal{S}, \mathcal{T})$ is the error when the model trained on source domain $\mathcal{S}$ is applied to the target domain $\mathcal{T}$. On the other hand, in-domain error $e(\mathcal{T}, \mathcal{T})$ is the error when the source and the target domains are the same. The transfer loss for ADDA is the smallest since it retrieves information on the target chip during the domain adaptation phase, unlike SS and MS.

We investigate the adaptability of CAVIA by comparing the prediction performance by reducing the number of source chips available in the meta training phase. The purpose of this experiment is to verify that CAVIA model can adapt to a target chip that has not been encountered in the meta training phase. As shown in Table 3, CAVIA model is trained on three different source chip settings. In the results on the table, the degradation of prediction accuracy is negligible even if the source chips set is smaller.

Results in Fig. 14 give an insight into the high adaptability of CAVIA. Fig. 14 shows histograms of context variables that are obtained on target chips after adaptation phases. The context variables obtained in a single target chip can be different depending on the date used in the adaptation phase. Nevertheless, the context variables for different chips are well separated so that the model can interpret them as indicators of inter-chip variation. Even if the context variable distributions for $\mathcal{C}_1$ and $\mathcal{C}_3$ slightly overlap more in the case of Fig. 14 (b) than in (a), the prediction performances on both chips decrease only a little. The results show that even if only

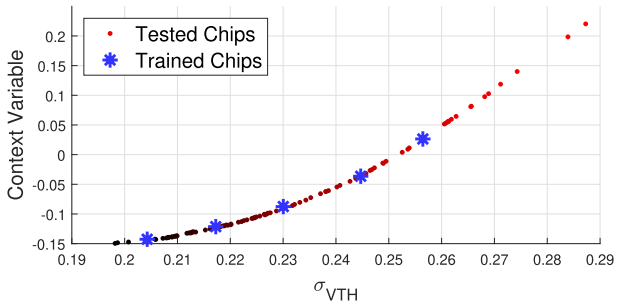


FIGURE 15. Context variables by standard deviation of threshold voltage distribution.

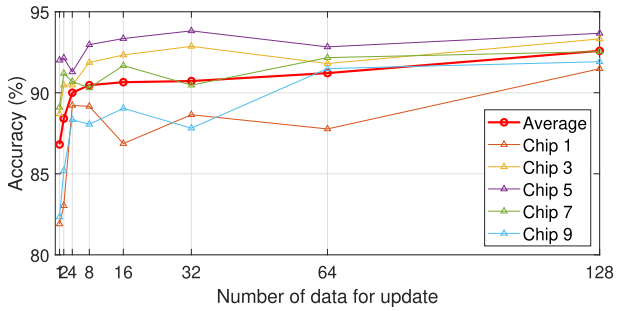


FIGURE 16. Prediction accuracy of CAVIA by the number of data used in adaptation phases.

two source chips are available in the meta training phase, CAVIA model learns to obtain well-distinguished context variables by different chips.

Fig. 15 shows further analysis of CAVIA in the aspect of adaptation. We evaluate context variables for a hundred memory chips with different σ_{VTH} using CAVIA model trained on $\{\mathcal{C}_1, \mathcal{C}_3, \mathcal{C}_5, \mathcal{C}_7, \mathcal{C}_9\}$. Blue stars denote context variables of source chips that are used for training CAVIA model while red dots are the context variables of tested chips. The context variable increases monotonically with respect to the σ_{VTH} values. The result shows that CAVIA model can adapt to different unseen chips by obtaining context variables related to the inter-chip variation. In addition, meta-trained CAVIA model is able to update the context variable as an indicator of the inter-chip variation to optimize the model to the target chip.

Fig. 16 shows the prediction accuracy of CAVIA by the number of data used in the adaptation phases. The average accuracy of CAVIA with a single data for an update is about 86 % which is similar to MS method. The accuracy increases abruptly as the number of data increases to 2, 4, or 8. The average prediction accuracy becomes over 90 % when the number of data for an update is 8, and the increment of the accuracy with the number of data gets smaller after it. From the results, we can verify that CAVIA can adapt to a target chip only using a smaller number of data in the adaptation phase.

Table 4 shows the prediction accuracy performances when the data for adaptation is labeled with average read latency or read latency samples. It may be difficult to obtain data labeled with optimal decoding mode with minimum average read latency in a target chip. CAVIA adaptation can alternatively use sample data pairs composed of input features and the decoding mode showing minimum read latency in a single static decoding flow. The average accuracy is only decreased by about 0.1 % on average when the alternative data is used in adaptation. Even though the data for updates cannot provide information about input features and corresponding minimum average read latency, CAVIA can adapt to different chips showing high prediction accuracy performances.

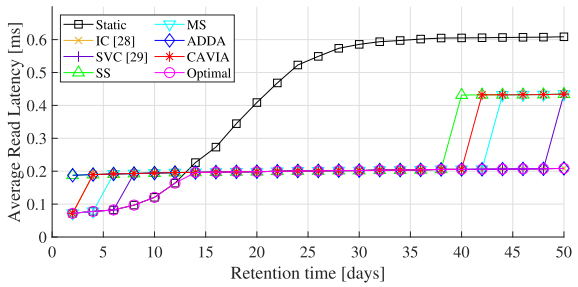
TABLE 4. Prediction accuracy (%) of CAVIA by class labels of data for adaptation.

Training data label	Target Chip (Test)					
	1	3	5	7	9	Avg
$\arg\min_d \mathbb{E}[lat_d]$	90.2	92.3	93.6	92.7	92.0	92.2
$\arg\min_d lat_d$	91.1	94.2	94.1	91.8	89.4	92.1

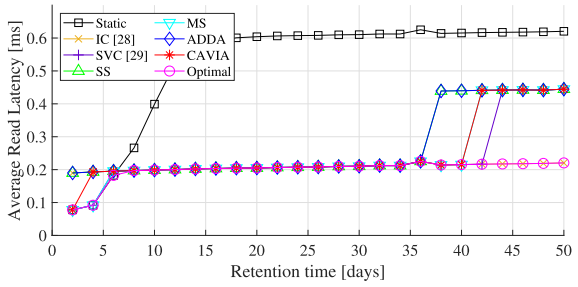
TABLE 5. Average latency reduction ratios (%) by different methods compared to static ERF.

Method	Target Chip (Test)					
	1	3	5	7	9	Overall
IC [28]	16.3	27.9	30.2	29.5	30.4	29.1
SVC [29]	22.7	39.9	48.9	44.1	36.5	39.9
SS	11.3	35.2	49.0	44.8	33.2	37.6
MS	12.2	34.5	48.1	47.4	38.4	40.5
ADDA	22.5	36.1	-	49.0	47.2	45.0
CAVIA	17.9	40.2	48.7	49.6	52.1	48.0
Optimal	35.7	45.2	51.6	54.0	54.1	51.9

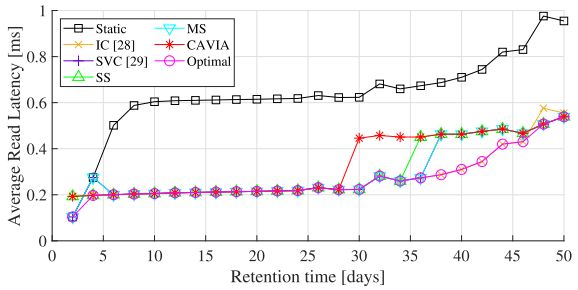
Finally, Fig. 17 illustrates the average read latency of the MSB page at a specific memory realization, i.e., the 8th wordline of the 5th block in chip 1, 3, 5, 7, and 9 with P/E cycle of 2,000. In addition, Table 5 shows the read latency improvement of different methods by providing the average latency reduction ratio compared to static ERF. The table presents the individual latency reduction ratios obtained by testing each chip, and overall ratios obtained by testing collectively chips 1, 3, 5, 7, and 9. We assume the latency for single sensing operation is $t_{\mathrm{sen}} = 100\,\mu$s and decoding latency is $t_{\mathrm{dec}} = 0.5\,\mu$s per iteration [61]. As shown in Table 5, since ADDA and CAVIA can predict optimal decoding modes in different chips, the average read latency can be decreased by 45 % and 48 %, respectively, compared to the static ERF, approaching the performance of the optimal baseline. On the other hand, the average read latency improvement is smaller with IC, SVC, SS, and MS as their predictions on optimal decoding mode are inaccurate compared to ADDA and CAVIA. In Fig. 17(e), the average read latency of ADDA and CAVIA are reduced by about 52.5 % and 52.9 %, respectively. The average read latency improvement ratios of proposed methods are about 48 % to 58 % in Fig. 17(b)-(d). In Fig. 17(a), CAVIA reduced average read latency by about 43.2 % while IC, SVC, MS, and ADDA reduced by about 53.9 %, 53.8 %, 46.4 %, and 52.8 % respectively. Note that although IC and SVC achieve large improvement in chip 1 and 3, it shows poor performance in chip 7 and 9 due to its inability to adapt to characteristics of different chips. From the experimental results, we verify that the accurate prediction of the optimal decoding mode can reduce read latency noticeably. In addition, it has been shown that ERF predictions based



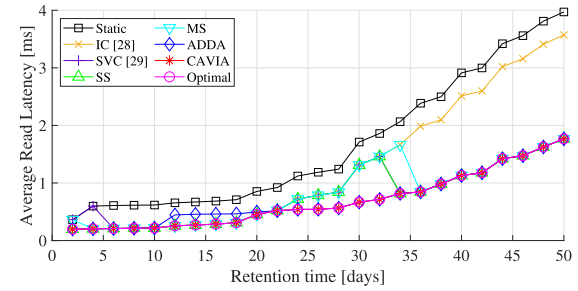
(a) 2000 cycles, Chip 1, Block 5, WL 8, MSB page.



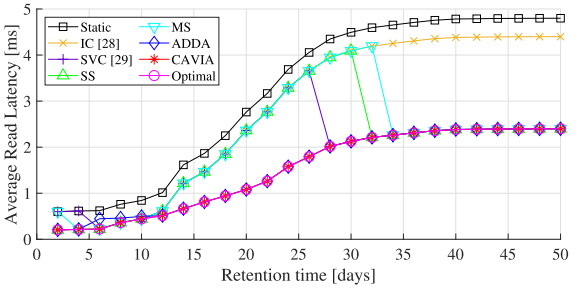
(b) 2000 cycles, Chip 3, Block 5, WL 8, MSB page.



(c) 2000 cycles, Chip 5, Block 5, WL 8, MSB page.



(d) 2000 cycles, Chip 7, Block 5, WL 8, MSB page.



(e) 2000 cycles, Chip 9, Block 5, WL 8, MSB page.

FIGURE 17. Average read latency by different training methods in multi-chips.

on ADDA and CAVIA achieve remarkable performance in multiple chips with different characteristics.

V. CONCLUSION

In this paper, we have presented dynamic ERF prediction based on reusable ML for low latency NAND flash memory considering process variation. First, we have formulated an ERF prediction framework based on ML. Specifically, practical input features with on-cell ratio and output class labeled with optimal decoding modes for the ML model have been proposed. In order to address process variation which makes it difficult to apply ML-based ERF prediction to multiple chips, we have proposed prediction methods based on reusable ML including transfer learning and meta learning. We have considered ADDA and CAVIA for transfer and meta learning methods, respectively, since their fast adaptation with low complexity. Moreover, experimental results have been provided to show the efficiency of the proposed methods. Proposed ADDA and CAVIA-based methods have shown average prediction accuracy of 90.9 % and 92.2 % when tested on five different memory chips. Transfer loss of ADDA was 2.2 % while it was 11.4 % in conventional ML by reusing a pre-trained classifier with domain adaptation. CAVIA could adapt to multiple chips by obtaining context variables based on reused meta trained model parameters. Moreover, CAVIA can successfully adapt to a hundred different chips, and the obtained context variables monotonically increased with σ_{VTH} reflecting characteristics of the memory chips. The average read latency of NAND flash memory could be reduced by over 45 % compared to static ERF when dynamic ERF based on reusable ML is applied. Accordingly, we have verified that ERF prediction methods based on reusable ML can achieve reliable and low-latency performance in NAND flash memory systems even in the presence of process variation.

REFERENCES

[1] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, "Introduction to flash memory," *Proc. IEEE*, vol. 91, no. 4, pp. 489–502, Apr. 2003.

[2] L. M. Grupp, A. M. Caulfield, J. Coburn, S. Swanson, E. Yaakobi, P. H. Siegel, and J. K. Wolf, "Characterizing flash memory: Anomalies, observations, and applications," in *Proc. 42nd Annu. IEEE/ACM Int. Symp. Microarchitecture*, Dec. 2009, pp. 24–33.

[3] 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," *Proc. IEEE*, vol. 105, no. 1, pp. 1725–1750, Sep. 2017.

[4] R. Chen, Y. Wang, J. Hu, D. Liu, Z. Shao, and Y. Guan, "Unified non-volatile memory and NAND flash memory architecture in smartphones," in *Proc. 20th Asia South Pacific Design Autom. Conf.*, Jan. 2015, pp. 340–345.

[5] K. Takeuchi, T. Tanaka, and T. Tanzawa, "A multipage cell architecture for high-speed programming multilevel NAND flash memories," *IEEE J. Solid-State Circuits*, vol. 33, no. 8, pp. 1228–1238, Aug. 1998.

[6] Y. Park, J. Lee, S. S. Cho, G. Jin, and E. Jung, "Scaling and reliability of NAND flash devices," in *Proc. IEEE Int. Rel. Phys. Symp.*, Jun. 2014, pp. 1–4.

[7] Y. Shim, M. Kim, M. Chun, J. Park, Y. Kim, and J. Kim, "Exploiting process similarity of 3D flash memory for high performance SSDs," in *Proc. 52nd Annu. IEEE/ACM Int. Symp. Microarchitecture*, Oct. 2019, pp. 211–223.

[8] R. Micheloni, S. Aritome, and L. Crippa, "Array architectures for 3-D NAND flash memories," *Proc. IEEE*, vol. 105, no. 9, pp. 1634–1649, Sep. 2017.

[9] Y. Luo, S. Ghose, Y. Cai, E. F. Haratsch, and O. Mutlu, "Improving 3D NAND flash memory lifetime by tolerating early retention loss and process variation," in *Proc. Abstr. ACM Int. Conf. Meas. Modeling Comput. Syst.*, Jun. 2018.
[10] Y. Li and K. N. Quader, "NAND flash memory: Challenges and opportunities," *Computer*, vol. 46, no. 8, pp. 23–29, Aug. 2013.
[11] 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, Autom. Test Eur. Conf. Exhib. (DATE)*, 2013, pp. 1285–1290.
[12] C. A. Aslam, Y. L. Guan, and K. Cai, "Read and write voltage signal optimization for multi-level-cell (MLC) NAND flash memory," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1613–1623, Apr. 2016.
[13] J. Wang, K. Vakilinia, T.-Y. Chen, T. Courtade, G. Dong, T. Zhang, H. Shankar, and R. Wesel, "Enhanced precision through multiple reads for LDPC decoding in flash memories," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 880–891, May 2014.
[14] K. Zhao, W. Zhao, H. Sun, T. Zhang, X. Zhang, and N. Zheng, "LDPC-in-SSD: Making advanced error correction codes work effectively in solid state drives," in *Proc. 11th USENIX Conf. File Storage Technol.*, Feb. 2013, pp. 243–256.
[15] Q. Li, M. Ye, Y. Cui, L. Shi, X. Li, and C. J. Xue, "Sentinel cells enabled fast read for NAND flash," in *Proc. 11th USENIX Workshop Hot Topics Storage File Syst. (HotStorage)*, Jul. 2019, pp. 1–7.
[16] Y. Wu, E. F. Haratsch, Y. Cai, and A.-H. Alhussien, "Flash memory read retry using histograms," U.S. Patent 8 953 373, Feb. 10, 2015.
[17] C. Kim, D.-H. Kim, W. Jeong, H.-J. Kim, I. H. Park, H.-W. Park, J. Lee, J. Park, Y. L. Ahn, J. Y. Lee, and S. B. Kim, "A 512-GB 3-b/cell 64-stacked WL 3-d-NAND flash memory," *IEEE J. Solid-State Circuits*, vol. 53, no. 1, pp. 124–133, Jan. 2018.
[18] H. Choe, J. Jee, S.-C. Lim, S. M. Joe, I. H. Park, and H. Park, "Machine-learning-based read reference voltage estimation for NAND flash memory systems without knowledge of retention time," *IEEE Access*, vol. 8, pp. 176416–176429, 2020.
[19] X. Wang, G. Dong, L. Pan, and R. Zhou, "Error correction codes and signal processing in flash memory," in *Flash Memories*. Rijeka, Croatia: IntechOpen, 2011, ch. 3.
[20] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.
[21] B. Peleato, R. Agarwal, J. Cioffi, M. Qin, and P. H. Siegel, "Towards minimizing read time for NAND flash," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2012, pp. 3219–3224.
[22] M. Zhang, F. Wu, X. Chen, Y. Du, W. Liu, Y. Zhao, J. Wan, and C. Xie, "RBER aware multi-sensing for improving read performance of 3D MLC NAND flash memory," *IEEE Access*, vol. 6, pp. 61934–61947, 2018.
[23] G. Dong, N. Xie, and T. Zhang, "On the use of soft-decision error-correction codes in NAND flash memory," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 2, pp. 429–439, Feb. 2011.
[24] J. Wang, T. Courtade, H. Shankar, and R. D. Wesel, "Soft information for LDPC decoding in flash: Mutual-information optimized quantization," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2011, pp. 1–6.
[25] J. Park, M. Kim, M. Chun, L. Orosa, J. Kim, and O. Mutlu, "Reducing solid-state drive read latency by optimizing read-retry," in *Proc. 26th ACM Int. Conf. Architectural Support Program. Lang. Operating Syst.*, Apr. 2021, pp. 702–716.
[26] G. Dong, N. Xie, and T. Zhang, "Enabling NAND flash memory use soft-decision error correction codes at minimal read latency overhead," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 9, pp. 2412–2421, Sep. 2013.
[27] J. Li, K. Zhao, J. Ma, and T. Zhang, "Realizing unequal error correction for NAND flash memory at minimal read latency overhead," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 61, no. 5, pp. 354–358, May 2014.
[28] J. Kim and W. Sung, "Low-energy error correction of NAND flash memory through soft-decision decoding," *EURASIP J. Adv. Signal Process.*, vol. 2012, no. 1, pp. 1–12, Dec. 2012.
[29] Y.-C. Liao, C.-H. Huang, C. Zeng, and H.-C. Chang, "Data analysis and prediction for NAND flash decoding status," in *Proc. IEEE Int. Memory Workshop (IMW)*, May 2017, pp. 1–4.
[30] B. Kang, J. Jee, and H. Park, "Intelligent error recovery flow prediction for low latency NAND flash memory system," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2020, pp. 1367–1372.
[31] S. Lee, J. Jee, and H. Park, "Machine learning-based error recovery system for NAND flash memory with process variation," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2021, pp. 1537–1541.
[32] Y. Pan, G. Dong, and T. Zhang, "Error rate-based wear-leveling for NAND flash memory at highly scaled technology nodes," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 7, pp. 1350–1354, Jul. 2013.
[33] Y. Pan, H. Zhang, M. Gong, and Z. Liu, "Process-variation effects on 3D TLC flash reliability: Characterization and mitigation scheme," in *Proc. IEEE 20th Int. Conf. Softw. Qual., Rel. Secur. (QRS)*, Dec. 2020, pp. 329–334.
[34] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, pp. 1345–1359, Nov. 2010.
[35] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7167–7176.
[36] S. Thrun, "Lifelong learning algorithms," in *Learning to Learn*. Cham, Switzerland: Springer, 1998, pp. 181–209.
[37] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, Aug. 2017, pp. 1126–1135.
[38] M. Luisa Zintgraf, K. Shiarlis, V. Kurin, K. Hofmann, and S. Whiteson, "Fast context adaptation via meta-learning," in *Proc. ICML*, vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds. 2019, pp. 7693–7702.
[39] Y. Cai, O. Mutlu, E. F. Haratsch, and K. Mai, "Program interference in MLC NAND flash memory: Characterization, modeling, and mitigation," in *Proc. IEEE 31st Int. Conf. Comput. Design (ICCD)*, Oct. 2013, pp. 123–130.
[40] R. A. Cernea, L. Pham, F. Moogat, S. Chan, B. Le, Y. Li, and S. Tsao, "A 34 MB/s MLC write throughput 16 Gb NAND with all bit line architecture on 56 nm technology," *IEEE J. Solid-State Circuits*, vol. 44, no. 1, pp. 186–194, Jan. 2009.
[41] C. M. Compagnoni, A. S. Spinelli, R. Gusmeroli, A. L. Lacaita, S. Beltrami, A. Ghetti, and A. Visconti, "First evidence for injection statistics accuracy limitations in NAND Flash constant-current Fowler-Nordheim programming," in *IEDM Tech. Dig.*, Dec. 2007, pp. 165–168.
[42] K. Takeuchi, T. Tanaka, and H. Nakamura, "A double-level-V_{th} select gate array architecture for multilevel NAND flash memories," *IEEE J. Solid-State Circuits*, vol. 31, no. 4, pp. 602–609, Apr. 1996.
[43] C. M. Compagnoni, R. Gusmeroli, A. S. Spinelli, A. L. Lacaita, M. Bonanomi, and A. Visconti, "Statistical model for random telegraph noise in flash memories," *IEEE Trans. Electron Devices*, vol. 55, no. 1, pp. 388–395, Jan. 2008.
[44] C. M. Compagnoni, M. Ghidotti, A. L. Lacaita, A. S. Spinelli, and A. Visconti, "Random telegraph noise effect on the programmed threshold-voltage distribution of flash memories," *IEEE Electron Device Lett.*, vol. 30, no. 9, pp. 984–986, Sep. 2009.
[45] J.-D. Lee, J.-H. Choi, D. Park, and K. Kim, "Effects of interface trap generation and annihilation on the data retention characteristics of flash memory cells," *IEEE Trans. Device Mater. Rel.*, vol. 4, no. 1, pp. 110–117, Mar. 2004.
[46] Y. Cai, Y. Luo, E. F. Haratsch, K. Mai, and O. Mutlu, "Data retention in MLC NAND flash memory: Characterization, optimization, and recovery," in *Proc. IEEE 21st Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2015, pp. 551–563.
[47] K. Takeuchi, T. Tatsumi, and A. Furukawa, "Channel engineering for the reduction of random-dopant-placement-induced threshold voltage fluctuation," in *IEDM Tech. Dig.*, 1997, pp. 841–844.
[48] A. Asenov, A. R. Brown, J. H. Davies, S. Kaya, and G. Slavcheva, "Simulation of intrinsic parameter fluctuations in decananometer and nanometer-scale MOSFETs," *IEEE Trans. Electron Devices*, vol. 50, no. 9, pp. 1837–1852, Sep. 2003.
[49] A. Spessot, C. Compagnoni, F. Farina, A. Calderoni, A. S. Spinelli, and P. Fantini, "Compact modeling of variability effects in nanoscale NAND flash memories," *IEEE Trans. Electron Devices*, vol. 58, no. 8, pp. 2302–2309, Aug. 2011.
[50] F. Ferrarese, "Statistical analysis of total ionizing dose response in 25-nm NAND flash memory," Ph.D. dissertation, School Eng., Univ. Padova, Padua, Italy, 2014.
[51] K. Pradeep, T. Poiroux, P. Scheer, A. Juge, G. Gouget, and G. Ghibaudo, "Analysis and modeling of wafer-level process variability in 28 nm FD-SOI using split C-V measurements," *Solid-State Electron.*, vol. 145, pp. 19–28, Jul. 2018.
[52] J. P. Cain and C. J. Spanos, "Electrical linewidth metrology for systematic CD variation characterization and causal analysis," in *Proc. SPIE*, vol. 5038, May 2003, pp. 350–361.
[53] K. Qian and C. J. Spanos, "A comprehensive model of process variability for statistical timing optimization," in *Proc. SPIE*, vol. 6925, Mar. 2008, pp. 430–440.

[54] Y. Di, L. Shi, C. Gao, Q. Li, C. J. Xue, and K. Wu, "Minimizing retention induced refresh through exploiting process variation of flash memory," *IEEE Trans. Comput.*, vol. 68, no. 1, pp. 83–98, Jan. 2019.
[55] Y. Pan, G. Dong, Q. Wu, and T. Zhang, "Quasi-nonvolatile SSD: Trading flash memory nonvolatility to improve storage system performance for enterprise applications," in *Proc. IEEE Int. Symp. High-Perform. Comp Archit.*, Feb. 2012, pp. 1–10.
[56] S. Park, H. Jang, O. Simeone, and J. Kang, "Learning to demodulate from few pilots via offline and online meta-learning," *IEEE Trans. Signal Process.*, vol. 69, pp. 226–239, 2021.
[57] Y. Luo, L. Zheng, T. Guan, J. Yu, and Y. Yang, "Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2507–2516.
[58] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2794–2802.
[59] C.-C. Hsieh, H.-T. Lue, Y. C. Li, T.-W. Chen, H.-P. Li, and C.-Y. Lu, "A novel dichotomic programming algorithm applied to 3D NAND flash," in *Proc. Symp. VLSI Technol. (VLSI Technol.)*, Jun. 2015, pp. T180–T181.
[60] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 513–520.
[61] R.-S. Liu, M.-Y. Chuang, C.-L. Yang, C.-H. Li, K.-C. Ho, and H.-P. Li, "Improving read performance of NAND flash SSDs by exploiting error locality," *IEEE Trans. Comput.*, vol. 65, no. 4, pp. 1090–1102, Apr. 2016.

MINYOUNG HWANG received the B.S. degree from the School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2021, where he is currently pursuing the M.S. degree.

His research interests include NAND flash memory systems, machine learning, simultaneous wireless information and power transfer, and intelligent reflecting surface for wireless communications.

JEONGJU JEE received the B.S. and M.S. degrees from the School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2016 and 2018, respectively, where he is currently pursuing the Ph.D. degree.

His research interests include massive MIMO systems, hardware-constrained communications, deep learning, and NAND flash memory systems.

JOONHYUK KANG (Member, IEEE) received the B.S.E. and M.S.E. degrees from Seoul National University, Seoul, South Korea, in 1991 and 1993, respectively, and the Ph.D. degree in electrical and computer engineering from The University of Texas at Austin, Austin, TX, USA, in 2002.

From 1993 to 1998, he was a Research Staff Member with Samsung Electronics, Suwon, South Korea, where he was involved in the development of DSP-based real-time control systems. He was with Cwill Telecommunications, Austin, in 2000, where he participated in the project for multicarrier CDMA systems with antenna array. From 2008 to 2009, he was a Visiting Scholar with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA. He is currently working as the Head of the School of Electrical Engineering (EE), KAIST, Daejeon, South Korea. His research interests include signal processing for cognitive radio, cooperative communication, physical-layer security, and wireless localization. He is a member of the Korea Information and Communications Society and the Tau Beta Pi (the Engineering Honor Society). From 2000 to 2002, he was a recipient of the Texas Telecommunication Consortium Graduate Fellowship.

HYUNCHEOL PARK (Senior Member, IEEE) received the B.S. and M.S. degrees in electronics engineering from Yonsei University, Seoul, South Korea, in 1983 and 1985, respectively, and the Ph.D. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 1997.

He was a Senior Engineer (1985–1991) and a Principal Engineer (1997–2002) at Samsung Electronics Company Ltd., South Korea. Since 2002, he has been with the School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, where he is currently a Professor. His research interests include communication theory and machine learning.

SEONMIN LEE received the B.S. degree in electronic convergence engineering from Kwangwoon University, Seoul, South Korea, in 2020, and the M.S. degree in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2022.

She has been with Samsung Electronics Company Ltd., Seoul, since 2022. Her research interests include the error management for NAND flash memory and machine learning.

JINYOUNG KIM received the B.S. and M.S. degrees in semiconductor science from Dongguk University, Seoul, South Korea, in 2003 and 2005, respectively, and the Ph.D. degree in electrical and electronic engineering from Yonsei University, Seoul, in 2017.

She joined Samsung Electronics Company Ltd., Hwaseong, South Korea, in 2005. From 2005 to 2011, she worked in circuit design for DRAM and PRAM. Since 2012, she has involved in NAND flash memory design. Her current interests include error detection, correction, and recovery technologies in flash memory systems.