Norns: Three Guides for Efficient Automatic Post-Fabrication Optimization of Modern NAND Flash Memory

Earl Kim
earl0314.kim@samsung.com
Samsung Electronics
Republic of Korea

Hyunuk Cho
gusdnr9779@postech.ac.kr
POSTECH
Republic of Korea

Sungjun Cho
allencho1222@postech.ac.kr
POSTECH
Republic of Korea

Myungsuk Kim
ms.kim@knu.ac.kr
Kyungpook National University
Republic of Korea

Jisung Park
jisung.park@postech.ac.kr
POSTECH
Republic of Korea

Jaeyong Jeong
jyong.jeong@samsung.com
Samsung Electronics
Republic of Korea

Eunkyoung Kim
yesdama.kim@samsung.com
Samsung Electronics
Republic of Korea

Sunghoi Hur
hursung@samsung.com
Samsung Electronics
Republic of Korea

Abstract

In order to meet the diverse requirements of modern storage systems, flash memory should be optimized by precisely tuning a huge number of internal operating parameters. Although 3D NAND flash memory successfully increases the capacity of storage systems, its complex architecture and unique error behavior make such optimization a more difficult and time-consuming process during NAND manufacturing. This work introduces Norns, a novel method for post-fabrication optimization of NAND flash memory, which is an essential step in the manufacturing process of modern 3D NAND flash memory to simultaneously meet various requirements on reliability, performance, yield, etc. Norns is based on simple machine-learning approaches yet with three key guidelines that leverage (*i*) domain-specific rules, (*ii*) recent optimization results, and (*iii*) online simulation, respectively, to enable quick optimization of a large number of device parameters within the limited product turnaround time (TAT). We evaluate Norns in mass production for 7th-generation QLC NAND flash memory and 8th-generation TLC NAND flash memory. Our Norns can achieve superior optimization over existing post-fabrication optimization techniques by showing significant performance and reliability improvements by up to 8.8% and 12% on average, respectively.

CCS Concepts

• **Hardware** → **External storage**; **Modeling and parameter extraction**.

Keywords

NAND flash memory, post-fabrication optimization, performance, reliability, machine learning, evolutionary algorithm

ACM Reference Format:
Earl Kim, Hyunuk Cho, Sungjun Cho, Myungsuk Kim, Jisung Park, Jaeyong Jeong, Eunkyoung Kim, and Sunghoi Hur. 2024. Norns: Three Guides for Efficient Automatic Post-Fabrication Optimization of Modern NAND Flash Memory. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD '24), October 27–31, 2024, New York, NY, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3676536.3676825

1 Introduction

Post-fabrication optimization is an essential step in the manufacturing process of modern NAND flash memory. After fabricating a chip, manufacturers configure various device parameters (e.g., the voltage/timing parameters for read/program/erase operations) that determine the chip's performance, reliability, and yield. To cope with the continuous reliability degradation in modern NAND flash memory due to aggressive process scaling and multi-leveling, post-fabrication optimization, i.e., determining the best device parameters, becomes increasingly important to meet both performance and reliability requirements while maximizing the yield.

Traditionally, post-fabrication optimization is manually done by experienced engineers, but doing so becomes extremely challenging for modern NAND flash chips for two reasons. First, the number of device parameters to optimize for a single chip has significantly increased. For example, a modern triple-level cell (TLC) NAND flash chip has thousands of device parameters that need to be carefully configured depending on the physical characteristics of target flash cells. Second, post-fabrication optimization is a time-critical task. To meet the limited product *turnaround time (TAT)*, i.e., the amount of time taken to complete the chip production since the start of development, quite a short time (e.g., a few days) is allowed for post-fabrication optimization in many cases. Manually optimizing a large number of device parameters within a limited time requires significant human effort and/or often leads to far-optimal results.

ICCAD '24, October 27–31, 2024, New York, NY, USA

ACM ISBN 979-8-4007-1077-3/24/10
https://doi.org/10.1145/3676536.3676825

To address the challenges, recent studies [7–9] have attempted to *automate* the post-fabrication optimization of NAND flash memory based on two machine learning approaches. In general, the recent proposals use (*i*) a lightweight deep-learning (DL) model to generate initial device parameters that can roughly meet the given design requirements and (*ii*) an evolutionary algorithm (EA) to automatically explore device parameters from the initial values. The prior work has experimentally demonstrated that such a simple combination can effectively find better device parameters compared to manual optimization, with limited datasets and time (which make it extremely challenging to develop a large DL model).

Unfortunately, we identify that the existing techniques become inadequate for modern NAND flash memory due to the significant increases in both the number of device parameters and their performance and reliability impact. The key problem is that the limited time for post-fabrication optimization remains the same, i.e., the optimization needs to reach a *narrower* optimal range within a *wider* search space under the same time constraint. Simple EA is hard to address such a challenge since post-fabrication optimization is significantly bottlenecked by real-device measurements. For example, with our industry-scale infrastructure (from a major NAND flash manufacturer), only around 40,000 parameter sets can be tested on real chips in a week. This means that the number of generations in an EA-based optimization is limited to ~40,000, which is insufficient for optimizing modern quad-level cell (QLC) NAND flash memory only with simple random mutation and crossover.

In this work, we introduce Norns (*NAND Optimization based on Rules, Newborns, and Simulation*)[1], a new post-fabrication optimization scheme that enables optimizing modern high-density NAND flash memory to better meet various performance, reliability, and yield requirements within the quite limited optimization time during NAND manufacturing. **Norns's key idea** is to *more carefully yet more extensively* explore the device parameter sets (i.e., populations) to be tested on real chips for each generation in the EA. Doing so allows the scarce real-device measurement time to be used for more promising populations rather than randomly generated ones, *effectively accelerating* the entire optimization process.

Norns generates device parameters using three key guidelines. First, it ensures that each population meets a set of *predefined rules* based on domain-specific knowledge from experienced engineers. If the EA generates a value that violates any rules, Norns picks a random value that meets all related rules. Second, Norns also ensures that every device parameter is in a desirable value range dynamically determined based on the *recent populations* (i.e., *newborns*) that provide the highest performance and reliability levels. Third, Norns *simulates* the optimization level of each population based on a simple Bayesian model (trained online with real measurement results) to select the best populations to be tested on real chips. Doing so enables Norns to effectively evaluate orders of magnitude larger number (e.g., ~10^7) of populations compared to the limited measurement times (e.g., ~40,000).

We evaluate Norns using real state-of-the-art 3D TLC/QLC NAND flash chips in mass production. Our results show that Norns improves the performance and reliability of the 7th-generation QLC NAND flash chips [6] by 12% and 8.8% on average, respectively, compared to the state-of-the-art post-fabrication optimization technique [8] that cannot meet the reliability requirements in the given time. Norns outperforms the state-of-the-art technique also for the 8th-generation TLC NAND flash chips [12], improving the performance and reliability by 6.2% and 11%, respectively.

This work makes the following key contributions:

- To our knowledge, this work is the first to enable automatic post-fabrication optimization to simultaneously meet various requirements in modern high-density NAND flash memory.
- We introduce Norns, a new post-fabrication optimization scheme that effectively accelerates the optimization process using predefined rules, intermediate results, and simulation.
- We demonstrate Norns's effectiveness throughout real-device evaluation of state-of-the-art 3D TLC/QLC NAND flash chips in mass production, showing large performance and reliability benefits over the state-of-the-art technique [8].

2 Background

We provide a brief background on modern NAND flash memory, which is necessary to understand the rest of the paper.

2.1 Overview of NAND Flash Memory

NAND Flash Organization. Figure 1 depicts the hierarchical organization of a NAND flash chip. A set of flash cells form a NAND string that is connected to a bitline (BL), and NAND strings of different BLs compose a block. The control gate of each flash cell at the same vertical location in a block is connected to a wordline (WL), so all the cells at the same WL concurrently operate. Thousands of blocks constitute a plane while sharing all the BLs in the plane. There are typically two or four planes in a single NAND flash chip (or die).

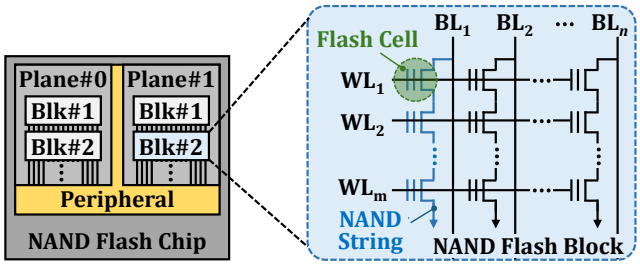


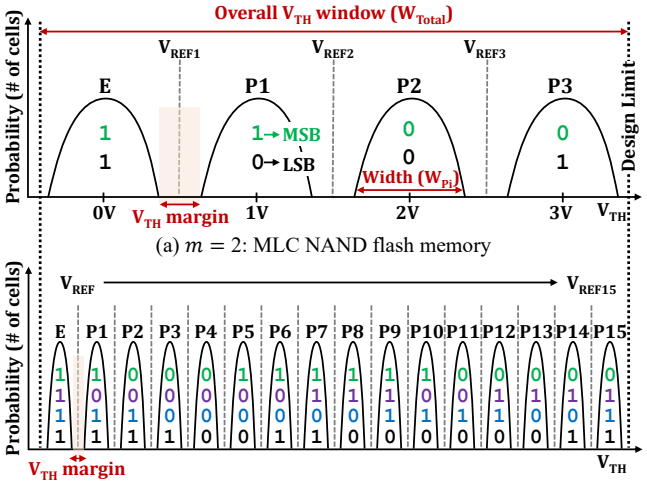
Figure 1: An organizational overview of NAND flash memory.

NAND Flash Operations. A flash cell stores bit data as a function of its threshold voltage (V_{TH}) level that highly depends on the amount of charge in the cell's charge trap layer; the more electrons in the charge trap layer, the higher the cell's V_{TH} level. There are three basic operations to access or modify the data stored in NAND flash memory: (*i*) program, (*ii*) read, and (*iii*) erase operations. The program operation, which increases V_{TH} of selected flash cells, injects electrons from the substrate into the charge trap layer of the selected flash cells using FN tunneling [17] by applying a high voltage (> 20V) to WL gates (i.e., changing the state of the flash cell from '1' state to '0' state). On the other hand, to erase programmed

[1]The word *Norns* refers to the three deities in Norse mythology, Urd, Verdandi, and Skuld, who are responsible for shaping the course of human destinies based on the past, present, and future, respectively.

cells (i.e., change the state of the flash cell from '0' state to '1' state), a high voltage (> 20V) is applied to the substrate (while WL gates are set to 0V) to remove electrons from the charge trap layer, which decreases the V_{TH} of the flash cells. The stored data can be read out by sensing the V_{TH} level of the target flash cells on the selected WL using a proper read reference voltage (V_{REF}).

Multi-level Cell Technology. Since the mid-2000s, multi-level cell techniques have been widely used to continuously increase the capacity of NAND flash memory. Multi-level cell technology aims to store multi-bit information in a single flash cell. Figure 2 illustrates V_{TH} distributions for 2^m-state NAND flash memory, which stores m bits within a single flash cell by using 2^m distinct V_{TH} states. For example, m is 2 for MLC (i.e., storing 2 bits per cell), m is 3 for TLC, and m is 4 for QLC NAND flash memory, respectively [2, 5, 6, 12].



(a) $m = 2$: MLC NAND flash memory

(b) $m = 4$: QLC NAND flash memory

Figure 2: V_{TH} distributions of 2^m-state NAND flash memory.

As m increases to store more bits within a flash cell, more V_{TH} states should be put into the limited V_{TH} window, which makes a V_{TH} margin (i.e., a gap between two neighboring V_{TH} states calculated as $W_{\text{Total}} - \sum_{i=0}^{k} W_{\text{Pi}}$) inevitably narrower. Since narrow V_{TH} margin makes NAND flash memory more vulnerable to various noise effects, in turn, significantly degrades the flash reliability, more careful management is required for multi-level cell NAND flash memory to form finer V_{TH} states, as m increases [11, 18].

Shaping a narrow V_{TH} state can be achieved by reducing ISPP step voltage, but it directly degrades the flash performance (e.g., program latency) [10, 23]. In higher m-bit multi-level cell NAND flash memory, such a trade-off between performance and reliability of NAND flash memory makes flash optimization more complex and difficult. Therefore, to meet various requirements of a storage system (e.g., performance, requirement, and yield), it is essential to efficiently optimize NAND flash memory by precisely tuning internal NAND operating parameters.

2.2 Tuning NAND Operating Parameter

Just after a NAND flash chip is fabricated, its characteristics are insufficient to work as a commercial memory product. Hence, the internal operating parameters of individual NAND flash chips should be carefully tuned to ensure device requirements. Key internal operating parameters (e.g., voltage levels or timing values) can be modified by setting a value of *electrical fuse (eFuse)* that is placed in the peripheral circuit, enabling quick access during NAND operations [19].

Figure 3 shows how internal operating parameters can be tuned using the 8-bit eFuse. The 8-bit eFuse can contain a value from 0 to 255. The analog voltage generator or operation timer circuit is connected to the eFuse, and its corresponding output is determined based on the eFuse's setting value. For example, when the eFuse value is incremented sequentially from 0 to 255, the analog voltage level output can linearly increase from 5V to 15.2V with 40mV resolution. Similarly, an 8-bit eFuse can change the timing parameter from 40μs to 10.6ms with 40μs stepping.

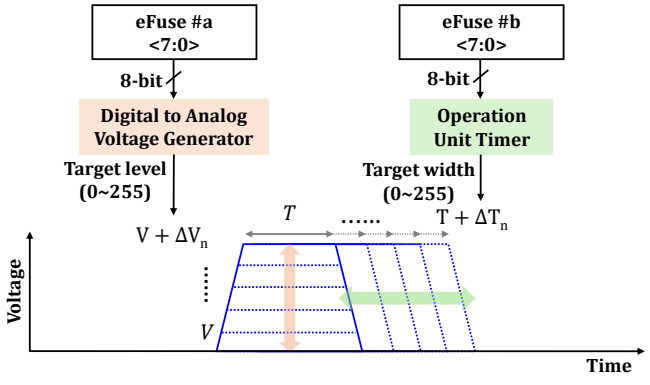


Figure 3: Overview of NAND eFuse operation.

3 Motivation

We introduce an overview of post-fabrication optimization for NAND flash memory and discuss the limitations of the state-of-the-art techniques [7–9].

3.1 NAND Flash Post-Fabrication Optimization

Post-fabrication optimization is an essential step of NAND flash manufacturing process to meet various requirements on reliability and performance while maximizing the yield. After fabricating a NAND flash chip, manufacturers should carefully configure various *device parameters* that specify the voltage levels and timing values used in read/program/erase operations. Each configured value is stored inside the NAND flash chip using an electrical fuse (eFuse) described in Section II, so that the chip can refer to related parameter values when performing a NAND flash operation.

Figure 4 shows how the device parameters for program operation are set in post-fabrication optimization, which significantly affects the NAND flash chip's reliability and performance. Modern NAND flash memory commonly adopts the *incremental step pulse programming (ISPP)* scheme [23] to secure sufficient margins between adjacent threshold voltage (V_{TH}) states that encode different bit data. The ISPP scheme performs multiple programming steps, gradually increasing the program voltage of each step from

V_{PGM1} by a certain amount of step voltage ΔV_{PGM}. At the end of each programming step, the ISPP scheme performs multiple verify operations to estimate each cell's current V_{TH} level. If a cell's V_{TH} level has sufficiently increased to be higher than the target verify voltage V_{VFY_i}, the ISPP scheme excludes the cell from the next programming steps. How to set ISPP parameters (e.g., V_{PGM1}, ΔV_{PGM}, and V_{VFY_i}) significantly affects the reliability and programming performance of NAND flash memory. For example, using a smaller ΔV_{PGM} value leads program operation to increase the V_{TH} levels of target cells more precisely, which enables wider margins between adjacent V_{TH} states (i.e., higher reliability), but at the cost of a longer programming latency (i.e., lower performance).

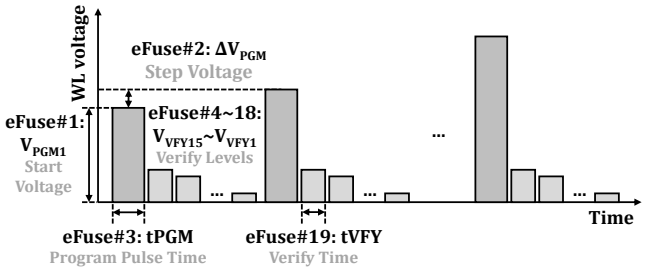


Figure 4: ISPP mechanism and related operating parameters.

Traditional post-fabrication optimization is done manually by experienced engineers, but doing so becomes extremely challenging for modern NAND flash memory due to two reasons. First, the number of device parameters to optimize for a single NAND flash chip has significantly increased. Modern NAND flash memory exhibits high process variations [21], i.e., the performance and reliability characteristics of flash cells significantly vary depending on the cell location as well as the operating conditions, so it is difficult to simultaneously meet various requirements using a single set of device parameters designed for the worst WLs. To address this, manufacturers group WLs that have similar characteristics and design a NAND flash chip to use different voltage/timing parameters for different WL groups, which significantly increases the number of device parameters; for example, the number of program-related parameters is around a hundred in relatively-old generations of triple-level cell (TLC) NAND flash memory, but it increases to more than 2,000 in recent quad-level cell (QLC) NAND flash memory.

Second, post-fabrication optimization is a time-critical process. The *turnaround time (TAT)*, i.e., the total amount of time taken to *complete* the production of new NAND flash chips, is quite limited for developing a new generation of NAND flash memory. As the TAT includes the time for not only post-fabrication optimization but also many other tasks, e.g., the layout design, fabrication, and verification of chips, only a week or even fewer days are given for post-fabrication optimization in many cases. Manually optimizing a large number of parameters within such a limited time requires significant human effort and fails to achieve near-optimal results.

3.2 Limitations of the State-of-the-Art

To address the above challenges, recent works [7–9] attempt to automate post-fabrication optimization of NAND flash memory by combining a lightweight deep-learning (DL) model and an evolutionary algorithm (EA). Figure 5 depicts the state-of-the-art technique's optimization process that consists of two phases [8]. In the first phase, the DL model generates *initial* device parameters that can *roughly* meet the given performance and reliability requirements. It uses (*i*) a variational autoencoder (VAE) to encode the performance and reliability characteristics of target chips into latent values and (*ii*) a multi-layer perceptron (MLP) to generate initial parameters from the latent values. In the second phase, the EA performs parameter exploration from the initial values by repeating (*i*) random mutation/crossover on the previous populations and (*ii*) evaluation of new populations on target chips. The EA terminates the exploration either when the given time for post-fabrication optimization is over or when it finds a parameter set that can meet all requirements.

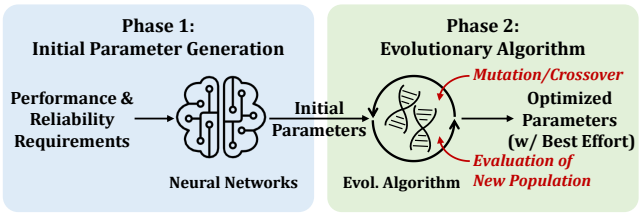


Figure 5: Automated post-fabrication optimization [8].

One may think that a large DL model trained for post-fabrication optimization could be more effective than the existing approach that relies on EA, but developing such a model is difficult in practice. This is because data generation of post-fabrication optimization requires real-device measurement, which is highly time-consuming. For example, in our industry-scale infrastructure (from one of the major NAND flash manufacturers), only around 40,000 parameter sets can be tested in a week. Such a small amount of data makes it extremely difficult to successfully train a large model (even the simple MNIST dataset contains more than 60,000 samples [14]), which, in turn, likely causes *no* optimization of device parameters. Note that datasets obtained from a certain NAND flash generation are hard to use for future post-fabrication optimization of another generation because the set of device parameters and their impact significantly vary across generations. In contrast, the EA progressively optimizes device parameters immediately using each real-device measurement, thereby providing best-effort improvements within the given time in most cases.

Even though the recent studies have demonstrated the effectiveness of their proposals in post-fabrication optimization of TLC NAND flash memory [7–9], we identify that the existing techniques are hard to use for optimizing more-advanced NAND flash memory. In QLC NAND flash memory, we observe that the existing techniques *cannot* meet the target reliability requirements within the given time. According to our analysis, the key problem is that the optimization progresses *too slowly* to meet both the performance and reliability requirements within the limited time. This is because aggressive process scaling and multi-leveling have increased not only the number of device parameters to optimize but also each parameter's performance and reliability impact in modern NAND

flash memory. As a result, the EA-based optimization needs to reach a *narrower* optimal range within a *wider* search space, while the maximum number of generations remains the same because it is dictated by the number of real-device measurements that can be performed within the given time. Even though the EA itself is an effective approach to guarantee progressive optimization with a limited amount of data, it may waste the scarce real-device measurements for randomly-generated populations.

4 Norns: Three Guides for Efficient Post-Fabrication Optimization

We introduce Norns (*NAND Optimization based on Rules, Newborns, and Simulation*, a new post-fabrication optimization scheme for modern high-density NAND flash memory which leverages three guides called (*i*) Urd, (*ii*) Verdandi, and (*iii*) Skuld, respectively.

4.1 Design Overview

We design Norns by extending the state-of-the-art post-fabrication optimization technique for NAND flash memory [8]. **The key idea** of Norns is to efficiently generate the device parameters for each generation in the EA so that more promising parameters can be tested on real chips. Doing so improves the optimization degree per generation, thereby accelerating the entire optimization process.

Figure 6 shows how Norns generates the next-generation populations from the previous populations (i.e., ancestors).[2] The high-level process of Norns is almost the same as a typical EA's process that consists of three steps: (*i*) ancestor evaluation, (*ii*) parent selection, and (*iii*) new population generation. The first two steps are straightforward; Norns first calculates each ancestor's *achievement score* that indicates how well the ancestor's device parameters meet the performance/reliability requirements and then selects the high-score parents to use for new population generation (§4.2). In the third step, Norns effectively explores a large number of populations using three guides, i.e., Urd, Verdandi, and Skuld, along with random mutation and crossover (§4.3). As high-level descriptions, Urd provides the conditions that every population must meet, which can be obtained from domain-specific knowledge; Verdandi specifies a desirable range for each parameter based on the recent populations with high achievement scores; Skuld predicts the achievement score of each candidate population (generated by random mutation and crossover with the other two guides) via simulation and selects the ones with the highest simulated scores as the next generation.

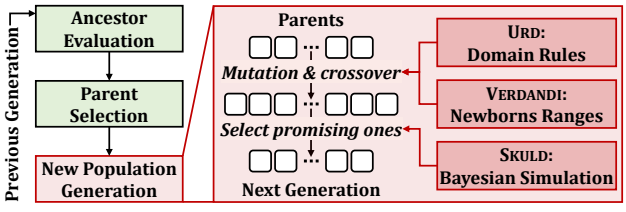


Figure 6: Overview of optimization process in Norns.

[2]We adopt the same method as the existing technique [8] to generate the initial population, focusing on improving the efficiency of the EA-based optimization, which has a higher impact on the optimization results.

4.2 Ancestor Evaluation and Parent Selection

For each ancestor A_i that consists of N device parameter values $\{a_0, a_1, ..., a_{N-1}\}$, Norns characterizes target NAND flash chips using the ancestor's device parameters and an object function that calculates the *achievement score* S_i as follows:

$$S_i = \sum_{k=0}^{M-1} w_k \times m_k(A_i), \tag{1}$$

where m_k is k-th evaluation metric (total M metrics), and w_k is the corresponding weight. Examples of m_k include (*i*) VSum (the total margins between every two adjacent V_{TH} states), (*ii*) Disturb (the maximum V_{TH} level of erased cells after programming), and (*iii*) tPROG (the average program latency), all of which can be obtained via real-device measurements with each ancestor A_i. After measuring $m_k(A_i)$ values for all metrics and ancestors, Norns calculates an ancestor's *benefit* for each metric by multiplying the predefined w_k; if a lower value is preferable (e.g., Disturb and tPROG), the weight is negative value; if a higher value is preferable (e.g., VSum), the weight is positive value. Finally, Norns takes the summation of all benefits as the ancestor's achievement score (the higher the score, the better) and probabilistically selects the highest-score ancestors as the parents for the next generation.

4.3 New Population Generation

Norns generates new populations using three key guidelines that enable efficient finding of promising parameters across candidates.

Guide 1. Urd: Domain Rules. Urd specifies the necessary conditions, called *domain rules*, that certain parameters *must* meet in any crossover and mutation process. Domain rules are predefined by experienced engineers based on domain-specific knowledge of well-known, highly correlated parameters. A simple example of domain rules is the relationship among verify levels V_{VFYi} in program operations. Since V_{VFYi} is used for checking if a cell's V_{TH} level is in a range of the i-th program state (e.g., V_{VFY15} is used for the fifteenth program state in QLC NAND flash memory as shown in Figure 4), V_{VFYi} must be lower than V_{VFYj} if $i < j$. Also, recent QLC NAND flash memory [6] employs a *reprogramming* scheme [20] that performs two program operations on a single wordline (WL), coarse programming and fine programming, to enhance reliability; any V_{VFYi} in fine programming must be higher than V_{VFYi} in coarse programming (which is previously performed) since the reprogramming scheme aims to increase the margin between adjacent V_{TH} states via fine programming. Urd prevents the scarce real-device measurements from being wasted for unrealistic values that violate the basic properties but can be generated by simple mutation and crossover. In our current design, Urd consists of five domain rules that are related to ~600 device parameters.

Guide 2. Verdandi: Newborns Ranges. Verdandi specifies desired ranges of certain parameters, called *newborns ranges*, based on the device parameters of ancestors that exhibit the highest achievement scores. To this end, Norns maintains *greatest ancestors list (GAList)* that stores a fixed number of ancestors with the highest achievement scores until the current iteration. For each device parameter, Norns searches the corresponding min./Max. values in GAList among the ones that meet all the related domain rules, which are used as guidance in the crossover process in the EA.

Doing so enables the EA to (*i*) avoid generating undesirable values for highly correlated parameters that can potentially be missed in domain rules and (*ii*) identify near-optimal parameters more rapidly as Norns updates GAList every iteration with ancestors that lead to the highest achievement scores (i.e., it *reinforces* the desired ranges every iteration).

Figure 7 shows an example of how Norns runs the EA using domain rules and newborns ranges. Norns *sequentially* determines device parameters one by one, in order to take into account the rules specified for highly correlated parameters. Suppose that Norns is about to decide the k-th device parameter, V_{VFY3} in fine programming, from P_x and P_y that are selected as the parent, as shown in Figure 7. Norns performs either mutation that generates a random value at a very low probability μ or crossover that selects one of the corresponding values from the parent at a probability of $(1-\mu)$. In both cases, Norns checks all domain rules related to the device parameter, e.g., $V_{VFY3} > V_{VFY2}$. For mutation, Norns repeats random number generation until the value meets all the related domain rules (Case 1 in Figure 7). For crossover, Norns uses the new value *only if* the value meets all the related newborns ranges as well as all the related domain rules (Case 2-1). As shown in Figure 7, Norns derives a newborns range for V_{VFY3} when V_{VFY2}=1V, e.g., $1.2 \le V_{VFY3} \le 2$, based on the min./Max. values for V_{VFY3} when V_{VFY2}=1V in GAList (Case 2-2). If the new crossover value for V_{VFY3} violates any domain rules and newborns ranges, Norns randomly selects a value in the newborns ranges. Note that, even though such newborns ranges strongly guide the EA based on previous results, Norns can escape from local optima via mutation.

Guide 3. Skuld: Bayesian Simulation. Skuld predicts the achievement score of each new population generated via the EA with the other two guides, which effectively mitigates the *fundamental bottleneck* in post-fabrication optimization. As explained in §3.2, post-fabrication optimization is bottlenecked by the time-consuming real-device measurements. For example, even though Urd and Verdandi are effective at generating promising populations, only a fixed number of populations can be explored, which is limited to the number of real-device measurements that can be performed within the given time. To effectively explore a larger number of populations, which, in turn, can lead to further optimized device parameters, Skuld performs simple simulation using a Bayesian model to estimate a population's achievement score.

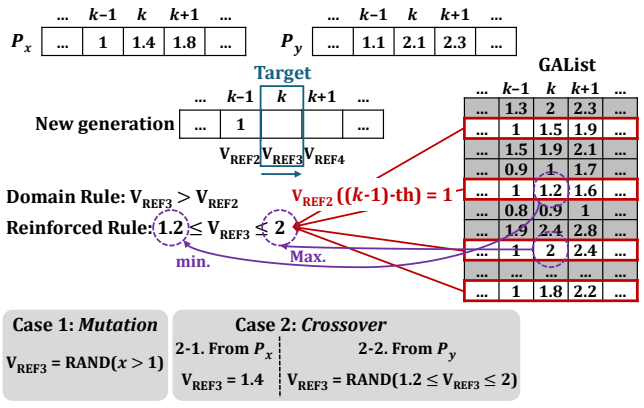


Figure 7: Generation of a new population in Norns.

Figure 8 depicts how Norns decides the final next-generation populations. Skuld maintains a Bayesian model throughout the optimization process which takes a population (i.e., a set of device parameters) as input and returns the estimated achievement score of the population as output. Given the ancestors in the previous generation (the initial parameter sets for the first generation), Skuld updates the Bayesian model (i.e., (re)training) by feeding each ancestor A_i along with its achievement score S_i calculated based on real-device measurement results as described in (1). Then, once a new population is generated via mutation and crossover with the other two guides (shown in Figure 7), Skuld performs inference using the Bayesian model that estimates the new population's achievement score *without* real-device measurements. Finally, Norns selects the new populations that have the highest estimated scores as the ancestors for the next generation.

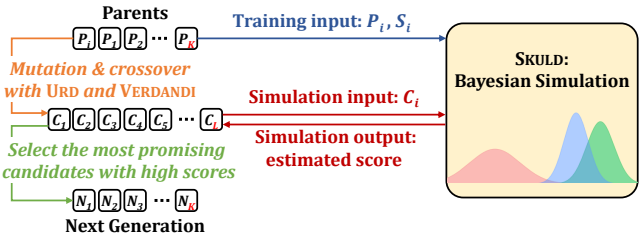


Figure 8: Decision of final populations with Skuld.

Note that Norns can explore significantly more candidate populations than the number of real-device measurements (i.e., $K \ll L$ in Figure 8). This is because Skuld can rapidly perform the Bayesian simulation of a candidate. For example, it takes ~100 minutes per iteration to generate and test all populations on real chips when we set the number of ancestors per generation to 400, whereas ~1,000 candidate populations can be simulated within 1 minute. Details of our Bayesian simulation are summarized in Table 1.

Table 1: Details of Bayesian simulation.

Model	Sklearn GPR (Gaussian Process Regressor)
Kernel	Radial Basis Function Kernel + Constant Kernel
Normalization	Standard Scaling

5 Evaluation

Methodology. We evaluate Norns using two types of modern 3D NAND flash chips in mass production: (*i*) the 8th-generation 3D TLC NAND flash chips [12] and (*ii*) the 7th-generation 3D QLC NAND flash chips [6]. We compare Norns to the state-of-the-art post-fabrication optimization technique (SotA) [8]. To highlight the effectiveness of Skuld (Bayesian simulation), we also evaluate the optimization using only Urd and Verdandi (Norns$_{UV}$). For each type of chip, we perform post-fabrication optimization using one of

NORNS, NORNS$_{UV}$, and SOTA on 40 blocks in 300 chips that are randomly selected from three different wafers. After post-fabrication optimization, we evaluate the final reliability and performance of the selected NAND flash chips with the device parameters obtained from the three techniques. For the evaluation of final reliability and performance, we use four blocks per chip while making the blocks experience half of the maximum program and erase (P/E) cycles and maximum retention time that are prescribed by manufacturers. For different post-fabrication optimization techniques, we use different blocks from the same chips which have similar performance/reliability characteristics.

Overall Reliability and Performance. Figure 9 compares the (a) reliability and (b) performance of the three post-fabrication optimization techniques in the TLC and QLC NAND flash chips. We measure VSum (the higher, the better) and average program latency tPROG (the lower, the better) that represent the reliability and performance of NAND flash memory, respectively. We also show the target reliability and performance requirements (Opt. Target) in Figure 9, and all values are normalized to SOTA.

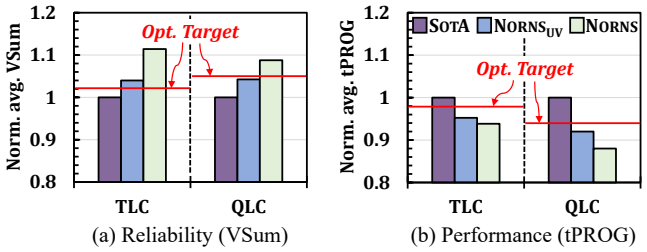


Figure 9: Reliability (VSum) and performance (tPROG) comparisons of three post-fabrication optimization techniques for the 8th-Gen. TLC and 7th-Gen. QLC NAND flash chips.

From Figure 9, we make three key observations. First, NORNS significantly outperforms SOTA, improving reliability by 11% (8.8%) and performance by 6.2% (12%) on average for the TLC (QLC) NAND flash chips compared to SOTA. Improving both performance and reliability at the same time is considered to be quite difficult due to the strong trade-off between performance and reliability, e.g., improving reliability usually comes at the cost of program performance [4]. The performance and reliability benefits of NORNS clearly show the effectiveness of NORNS at identifying near-optimal device parameters that are hard to find via the existing approaches. Second, NORNS is more effective for advanced QLC NAND flash chips. Although the reliability benefit of NORNS over SOTA is higher in TLC NAND flash chips (11%) compared to in QLC NAND flash chips (8.8%), *only* NORNS successfully meets the reliability requirement for QLC NAND flash chips while providing large performance benefits as well, which highlights the effectiveness of NORNS for more challenging optimization targets. Third, SKULD (Bayesian simulation) is the key enabler for NORNS to meet the reliability requirement in more advanced NAND flash memory. NORNS$_{UV}$ also provides considerable benefits over SOTA (4.1% and 6.4% in reliability and performance, respectively, on average across all the chips) but still fails to meet the reliability requirement in QLC NAND flash chips. This clearly shows the effectiveness of SKULD that fundamentally mitigates the real-measurement bottleneck.

We also visualize the V_{TH} distribution in the 8th-gen. TLC NAND flash chips to highlight the benefits of NORNS. As shown in Figure 10, NORNS improves the reliability over SOTA significantly, leading to (*i*) narrower distribution of all V_{TH} states and (*ii*) increasing the margin between adjacent V_{TH} states (i.e., more robust against bit errors).

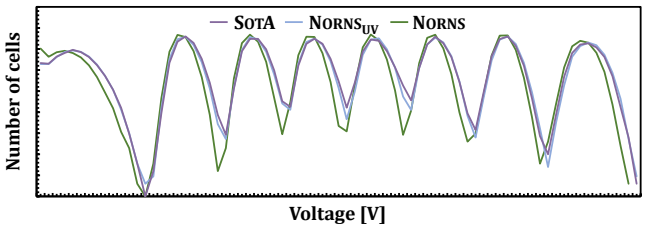


Figure 10: Comparison of V_{TH} distributions resulted from three post-fabrication optimization techniques in 8th-Gen. TLC 3D NAND flash memory.

Reliability and Performance Across Wordlines. To better understand the effectiveness of NORNS, we analyze the reliability and performance across different WLs. Figure 11 shows average VSum and tPROG of WLs at the same WL index in (a) TLC and (b) QLC 3D NAND flash chips, when using the device parameters identified via the three techniques. We sort the WLs in ascending order according to the VSum and tPROG values achieved with SOTA.

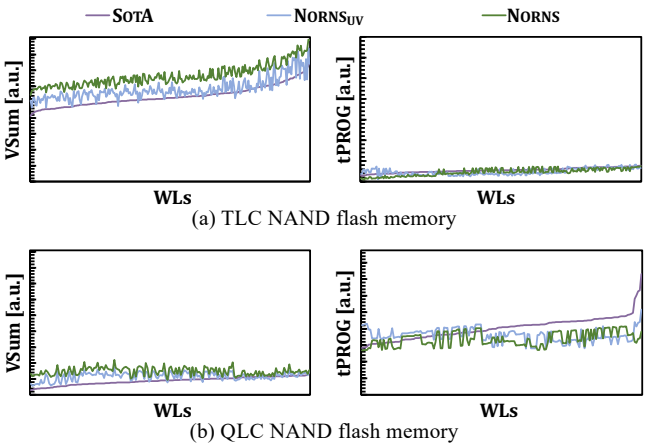
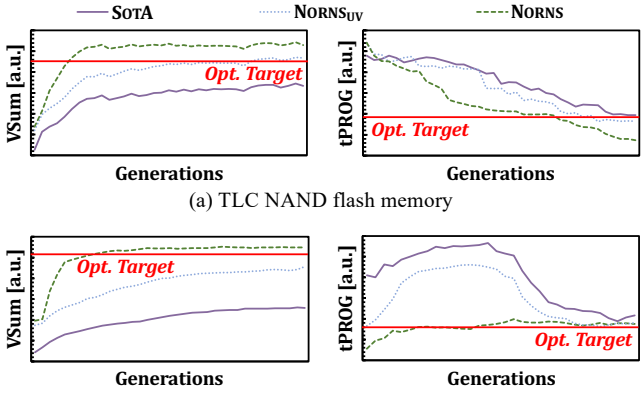


Figure 11: Reliability and performance across WLs for TLC and QLC 3D NAND flash chips.

We make two key observations from Figure 11. First, NORNS improves reliability over SOTA for *all* WLs in both TLC and QLC NAND flash chips, without causing outlier WLs in its post-fabrication optimization. Second, NORNS significantly enhances the performance and reliability of *worst* WLs. This is another important benefit of NORNS from both reliability and performance perspectives because (*i*) the lifetime of a NAND flash block is dictated by the reliability of worst WLs, and (*ii*) it is increasingly important to provide deterministic I/O performance in modern data-intensive applications [13, 15].

Optimization Speed. Finally, we compare the optimization speeds of the three techniques. Figure 12 shows how the three evaluated techniques improve the reliability (left) and performance (right) over time (until 100 generations) for the (a) TLC and (b) QLC NAND flash chips. Note that, as explained §3.2, all three techniques terminate post-fabrication optimization either when all requirements are met or when the given time (e.g., 100 generations) is over.



(a) TLC NAND flash memory

(b) QLC NAND flash memory

Figure 12: Optimization speed comparison for TLC and QLC 3D NAND flash chips.

We make two key observations. First, Norns significantly accelerates the optimization process over SotA. Norns rapidly improves the reliability close to the optimization target within less than 30 generations, while SotA fails to reach the target reliability until 100 generations. Second, Skuld has a significant impact on the optimization speed. At the 50th generation for the QLC NAND flash chips, Norns meets both the reliability and performance requirements, but the reliability and performance levels achieved by Norns$_{UV}$ are quite far from the optimization target. Note that the 50% saved time can be used for optimizing other aspects, e.g., the yield of the target chips. identifying near-optimal device parameters.

Based on our evaluation results and observations, we conclude that Norns is an effective post-fabrication optimization method for modern NAND flash memory which can overcome the limitations of existing approaches.

6 Related Work

To our knowledge, our work is the first to (*i*) propose an automatic post-fabrication optimization technique that provides significant benefits in advanced NAND flash memory and (*ii*) experimentally demonstrate its effectiveness in mass production. Since we already comprehensively discussed the state-of-the-art techniques for post-fabrication optimization of NAND flash memory [7–9] that are highly related to our proposal across the paper, we briefly describe other techniques closely related to our work.

Post-Fabrication Optimization of NAND Flash Memory. Several other studies have also attempted to improve post-fabrication optimization of NAND flash memory. Bongale *et al.* [1] and Xanthopoulos *et al.* [24] propose techniques that optimize individual parameters of analog circuits in NAND flash memory. The proposed techniques lead to better eFuse values by considering inter-chip variations during post-fabrication optimization. While effective, the prior proposals still require significant human effort compared to automatic post-fabrication optimization approaches and are hard to address complex dependency between device parameters.

Turnaround Time (TAT) Optimization. For reducing TAT, recent researchers have attempted to reduce post-fabrication test time for integrated circuits by exploiting neural networks or genetic algorithms (GAs). Golonek *et al.* [3] have explored the optimal test points using a GA-based approach. Lin *et al.* [16] introduced an autoencoder (AE)-based method for test screening escape. Shintani *et al.* [22] introduced an outlier screening method for test escape using a variational autoencoder (VAE) that can avoid the potential risk of overfitting by extracting training data features as a probability distribution. Norns also adopts similar approaches (e.g., VAE and GAs) to enable rapid-yet-efficient post-fabrication optimization of modern NAND flash memory.

7 Conclusion

We propose Norns, a new post-fabrication optimization method for modern NAND flash memory which provides significant reliability and performance benefits over manual optimization. Norns fundamentally addresses the key limitation of state-of-the-art techniques, i.e., being bottlenecked by the time-consuming real-device measurements, by providing three key guidelines that effectively accelerate the optimization process. We experimentally demonstrate the effectiveness of Norns using real state-of-the-art TLC/QLC NAND flash chips in mass production, showing its high benefits over existing techniques.

For better post-fabrication optimization, we also plan to extend our model by constructing a meta-model. We will further improve the predictive power of the model by combining the SKULD of Norns with a Bayesian-based probability estimation model and a tree structure model.

Acknowledgments

We thank anonymous reviewers of ICCAD 2024 for feedback. We also thank the Machine Learning TU at Samsung Advanced Institute of Technology (SAIT) of Samsung Electronics for their feedback and comments. (Corresponding author: Jisung Park)

References

[1] P. Bongale et al. 2016. A Novel Technique for Interdependent Trim Code Optimization. In *VTS*.
[2] J. Cho et al. 2021. 30.3 A 512Gb 3b/Cell 7th-Generation 3D-NAND Flash Memory with 184MB/s Write Throughput and 2.0Gb/s Interface. In *ISSCC*.
[3] T. Golonek and J. Rutkowski. 2007. Genetic-Algorithm-Based Method for Optimal Analog Test Points Selection. *IEEE TCAS II* (2007).
[4] J. Jeong et al. 2014. Lifetime Improvement of NAND Flash-based Storage Systems Using Dynamic Program and Erase Scaling. In *USENIX FAST*.
[5] K Kanda et al. 2012. A 19 nm 112.8 mm^2 64 Gb Multi-Level Flash Memory With 400 Mbit/sec/pin 1.8 V Toggle Mode Interface. *IEEE Journal of Solid-State Circuits* 48, 1 (2012), 159–167.
[6] D. Kim et al. 2020. 13.1 A 1Tb 4b/cell NAND Flash Memory with tPROG=2ms, tR=110µs and 1.2Gb/s High-Speed IO Rate. In *ISSCC*.
[7] J. Kim et al. 2019. Machine Learning-Based Automatic Generation of eFuse Configuration in NAND Flash Chip. In *ITC*.
[8] J. Kim et al. 2021. Machine Learning-Based Optimization Technique for High-Capacity V-NAND Flash Memory. In *ISTFA*.
[9] Jongmin Kim and Earl Kim. 2022. Machine Learning for Product Engineering in NAND Flash Memory. In *ICEIC*.

[10] M. Kim et al. 2018. SARO: A State-aware Reliability Optimization Technique for High Density NAND flash memory. In *Great Lakes Symposium on VLSI*.
[11] M. Kim et al. 2021. RealWear: Improving performance and lifetime of SSDs using a NAND aging marker. *Performance Evaluation* 145, 1 (2021), 102–153.
[12] M. Kim et al. 2022. A 1Tb 3b/Cell 8th-Generation 3D-NAND Flash Memory with 164MB/s Write Throughput and a 2.4Gb/s Interface. In *ISSCC*.
[13] S. Kim and J. S. Yang. 2018. Optimized I/O Determinism for Emerging NVM-based NVMe SSD in an Enterprise System. In *DAC*.
[14] Y. LeCun. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/* (1998).
[15] H. Li et al. 2021. IODA: A Host/Device Co-Design for Strong Predictability Contract on Modern Flash Storage. In *USENIX OSDI*.
[16] F. Lin and K. Cheng. 2017. An Artificial Neural Network Approach for Screening Test Escapes. In *ASP-DAC*.
[17] J. Maserjian and N. Zamani. 1982. Behavior of the Si/SiO_2 interface observed by Fowler-Nordheim tunneling. *Journal of Applied Physics* 53, 1 (1982), 559–567.
[18] M. Rino et al. 2010. *Inside NAND Flash Memories*.
[19] N. Robson et al. 2007. Electrically Programmable Fuse (eFuse): From Memory Redundancy to Autonomic Chips. In *CICC*.
[20] N. Shibata et al. 2008. A 70nm 16Gb 16-Level-Cell NAND Flash Memory. *IEEE JSSC* (2008).
[21] Y. Shim et al. 2019. Exploiting Process Similarity of 3D Flash Memory for High Performance SSDs. In *MICRO*.
[22] M. Shintani et al. 2018. Artificial Neural Network Based Test Escape Screening Using Generative Model. In *ITC*.
[23] K. Suh et al. 1995. A 3.3 V 32 Mb NAND Flash Memory with Incremental Step Pulse Programming Scheme. *IEEE JSSC* (1995).
[24] C. Xanthopoulos et al. 2017. Wafer-Level Adaptive Trim Seed Forecasting Based on E-Tests. In *ISCAS*.