On-the-Fly Machine Learning Model for Future 3D NAND Trim Optimization

Hyo Jung Son (Rachel Son), Youtian Zhang, Wei Cao, Xiang Yang, Deepanshu Dutta
Technology Development Engineering, Sandisk Technologies, Inc.
Milpitas, CA USA
Email: rachel.son@sandisk.com

Abstract

Trim optimization is a pivotal process in NAND flash memory development to meet stringent performance, power, and reliability requirements. Traditionally, this process relies heavily on the expertise and experience of device engineers, demanding significant development time to identify the optimal trim set. As NAND technology progress, the complexity of trim optimization exponentially increases due to the growing number of trim parameters across generations. In this work, we present a novel On-the-Fly Trim Optimization (OFTO) methodology leveraging a Bayesian optimization algorithm to overcome these challenges. Applied to our latest technology BiCS8 node, OFTO enhances performance and reliability by approximately 20% beyond the best results achieved with conventional trim optimization methods. Notably, the trim development cycle is reduced by 80%. This innovative approach represents a significant advancement in streamlining design efficiency for future 3D NAND technologies.

Introduction

The rapid expansion of Artificial Intelligence (AI) and Machine Learning (ML) technologies has created an unprecedented demand for storage solutions that offer high performance, large capacity and exceptional reliability, which has been the driving force of the continuous scaling of 3D NAND technology. The scaling of 3D NAND is primarily enabled by stacking more wordline (WL) layers. As shown in **Error! Reference source not found.**, the number of WL layers in Bit Cost Scalable (BiCS) 3D NAND technology has increased by ~350% in the past six generations. In addition to physical scaling, logic scaling, in the form of Triple-Level Cell (TLC), Quad-Level Cell (QLC) [1-3] and even Penta-Level Cell (PLC) [3], has been introduced to further increase the bit density and lower the cost per bit. It is worth noting that cost reduction always comes with trade-offs. Increasing the number of WLs and packing more bits into a single cell inevitably leads to degraded performance and reliability. To mitigate these challenges, post-fabrication on-chip functionalities, commonly known as trims have been widely adopted and proven effective in flash memory technology. However, the complexity of process and inherent variation has led to a steady increase in the number of parameters required with each new generation (**Error! Reference source not found.**). Notably, the number of trim parameters has increased by 124% from Bics 2 to Bics8. Such trend underscores the escalating importance and complexity of the trim optimization process in advancing 3D NAND technology.

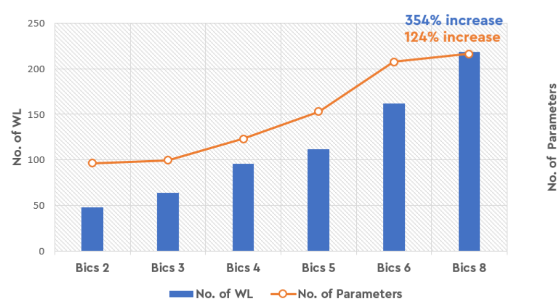

Fig. 1. 3D NAND scaling trend for the number of WL (left-side axis) and the number of parameters (right-side axis) over BiCS generations.

Trims play a critical role in regulating biases and timing during erase, program, and read operations in NAND flash memory. Many parameters exhibit interdependencies, meaning the optimal value of one parameter is influenced by the settings of others. Such interdependence introduces substantial complexity to the optimization process, making brute force approaches highly impractical. For example, with 10 interdependent parameters, each configured with 8 bits, the total number of possible trim parameter combinations reaches 2^{80}, requiring an exhaustive search that is computationally infeasible.

Traditionally, device engineers rely on their experience and knowledge to reduce the search dimension and narrow down the search range. While this approach worked for earlier generations of NAND flash memory, it is unlikely to be effective for future generations due to the significantly increasing complexity. To tackle this challenge, leveraging machine learning is crucial to minimize human effort in the trim optimization process. There have been some prior effort in this direction. Kim et al. [4] employed a two-step optimization process: using a Variational Autoencoder (VAE) to generate configurations satisfying target objectives, followed by a Genetic Algorithm (GA) to refine these configurations. This method requires a substantial offline training dataset for each operating characteristic, making it resource-intensive. Moreover, it is limited to configurations generated by VAE, and cannot optimize trims on-the-fly.

In this work, a novel On-the-Fly Trim Optimization (OFTO) technique is developed and demonstrated in the latest 3D NAND BiCS8 technology development. The key contributions include: (1) introducing the advanced Bayesian Optimization (BO) algorithm into 3D NAND trim tuning process to accelerate the search speed of multi-trim optimization; (2) enabling multi-objective co-optimization to balance reliability, performance and power, among other factors, while providing flexible trade-off options through Pareto Frontier; (3) successfully implementing a fully on-the-fly iterative process of testing and model updating, eliminating reliance on human judgment and experience-based decision-making. This new technique demonstrates great potential to enhance the trim development process for future 3D NAND technology.

Background

A. Bayesian Optimization

BO [6] is a powerful framework for optimizing expensive-to-evaluate black-box functions. It utilizes a probabilistic surrogate model, such as a Gaussian Process, to approximate the objective function with uncertainty quantified at the same time. By iteratively updating this model with new observations, BO predicts the behavior of the objective function and guides the search for optimal configurations using an acquisition function, as illustrated in **Error! Reference source not found.**.

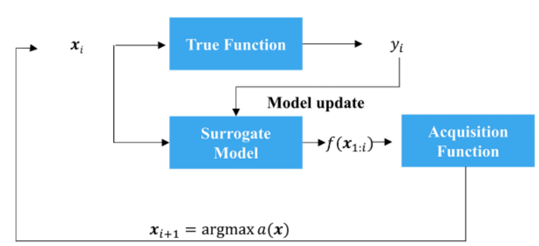

Fig. 2 Schematic illustration of Bayesian Optimization Framework. x_i and y_i are ith iteration trim parameters and the measurement result. The updated model $f(x_{1:i})$ employs the acquisition function $a(x)$ (such as Expected Improvement in (2) to propose the trim parameters x_{i+1} to be tested in the next loop.

BO consists of two key components : the surrogate model and acquisition function.

1) Surrogate model

A surrogate model using supervised learning techniques is employed to predict potential measurement outcomes. This model analyzes existing test results to estimate the behavior of untested trim parameters, thereby guiding the exploration of the design space. Gaussian Process (GP) [7, 8] is the most widely used probabilistic surrogate model for approximating objective functions in BO. A GP represents a probability distribution over possible functions that could fit a given set of n observations $(x_{1:n}, f(x_{1:n}))$, where x_i is a point in the design space (corresponding to one trim set in this work) and $f(x_i)$ is the objective function value (e.g., device performance). GP is an ideal choice for prior probability distributions because of the multivariate normal distribution's advantageous marginalization and conditioning properties. For existing observations $f(X)$, where $X = x_{1:n}$, and the new observations $f(X_*)$, the joint distribution can be written in the form of a multivariate normal distribution,

$$\begin{bmatrix} f(X) \\ f(X_*) \end{bmatrix} \sim N\left(\begin{bmatrix} m(X) \\ m(X_*) \end{bmatrix}, \begin{bmatrix} K & K_* \\ K_* & K_{**} \end{bmatrix}\right) \quad (1)$$

where $m(\cdot)$ represents the expected value of $f(\cdot)$. The matrix $K = \mathrm{K}(X, X)$ is a $n \times n$ covariance matrix, with its elements defined by the covariance function k, i.e. $\mathrm{K}_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The covariance function k is estimated by Radial Basis Function (RBF) kernel. Similarly, K_* and K_{**} are $K_* = \mathrm{K}(X, X_*)$ and $K_{**} = \mathrm{K}(X_*, X_*)$. $f(X_*)$ can then be predicted by existing observations $f(X)$ under GP assumptions: the conditional distribution $f(X_*)|f(X)$ is a normal distribution $N(\mu_*, K_*)$ and the mean and covariance can be calculated as $\mu_* = K_*^T K^{-1}[f(X) - m(X)] + m(X_*)$ and $K_* = K_{**} - K_*^T K^{-1} K_*$, where $m(\cdot)$ terms vanish when zero-mean assumption is adopted for the prior distribution. More detailed information about GP can be found in refs. [9-11].

2) Acquisition Function

The acquisition function, specifically Expected Improvement (EI) employed in this work, determines the next input (updated trim set in this work) to evaluate by prioritizing areas with the highest potential to improve optimization or reduce uncertainty. It is achieved by balancing exploration (investigating less understood regions) and exploitation (refining the optimal in the regions known to be predicted well). EI function quantifies the expected improvement over the current best observed value, given the posterior distribution of the objective function predicted by a surrogate model (e.g., Gaussian Process). For a new point x, EI is calculated as :

$$EI(x) = E[\max(0,\ f(x) - f_{max})] \qquad (2)$$

where $f(\boldsymbol{x})$ is the surrogate model's prediction at $\boldsymbol{x}$ and $f_{\max}$ is the best observed value so far (e.g., maximized hypervolume in the multi-objective case).

Error! Reference source not found. shows an example of applying BO to minimize a 1D black-box function. The top row shows observations (red dots), the predicted mean function (dotted green line) and variance (green shaded region) as provided by the GP. The bottom row shows the acquisition function, EI (blue lines), and the corresponding new point for the next iteration (blue dot). In the 3rd iteration shown in **Error! Reference source not found.**(a), the acquisition function suggests $x = 3$ as the next sampling point. In the 4th iteration in **Error! Reference source not found.**(b), the observation at $x = 3$ is added to the surrogate model and the deviation around the new point is reduced. The updated EI calculation then identifies $x = 5$, where the maximum EI value occurs, as the next sampling point. In this way, the function minima can be found with low deviation.

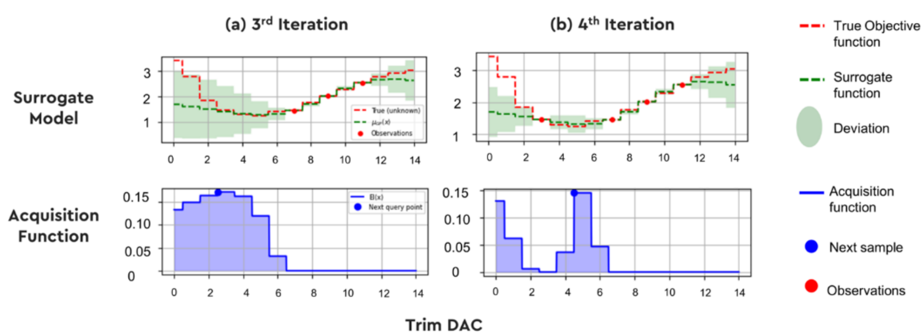

Fig. 3 Application of BO on a single objective optimization. (a) and (b) show the 3rd and 4th iteration results, respectively. DAC represents the digital values of trim parameters.

B. Multi-Objective Optimization

In general, common objectives such as performance, power and reliability should all be optimized. Unfortunately, it is highly unlikely that a single solution exists that can simultaneously optimize all objectives. Moreover, depending on the application, the priority of these objectives may vary significantly. Therefore, it is essential to implement multi-objective optimization (MOO) and offer flexible solutions that address the diverse requirements of different customers. Here we utilize the concept of hypervolume to obtain the Pareto Frontier (PF) to serve this purpose. As illustrated in **Error! Reference source not found.** for a two-objective case, hypervolume is the area in the objective space between a reference point and PF. This measure effectively captures both the quality and diversity of solutions in MOO. A higher hypervolume indicates the better PF.

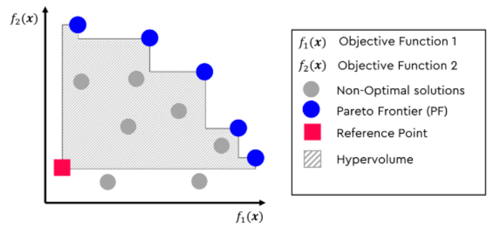

Fig. 4 Pareto Frontier and hypervolume illustration when maximizing both objective functions $f_1(x)$ and $f_2(x)$. The blue points on the Pareto Front (PF) dominate the grey non-optimal solutions, where dominance in general implies superior performance in at least one objective while being at least equal in all others.

Implementation In NAND Development

To fully get rid of human intervention, we developed the OFTO technique by integrating BO in the iterative loop between NAND tester (customized internally in Sandisk) and server on which BO is running. As illustrated in **Error! Reference source not found.**, BO sends updated trim set to NAND tester, and NAND tester sends back measured Si data for surrogate model update, forming an iterative loop until the PF evolution saturates.

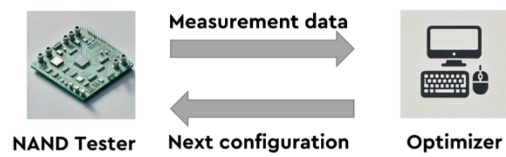

Fig. 5 Implementation of the OFTO technique for NAND trim development

OFTO technique supports multi-objective co-optimization. Tested objectives include, but are not limited to Vt window, state margin, read (t_R) and write (t_{PROG}) performance, Fail Bit Count (FBC). To minimize sample-to-sample variation, testing can be conducted on the same sample (e.g., the same block) if the total number of iterations is limited to a few hundred. For larger iteration counts, similar blocks can be pre-selected and alternated during testing to mitigate the effects of degradation.

A useful feature of BO for NAND trim development is its capability to reveal trim interaction. **Error! Reference source not found.** illustrates a contour plot depicting the relationship between Trim A and B, along with the corresponding objective values. The plot provides a visual representation of how variations in these trim parameters influence the optimization objectives. This visualization aids in understanding the interaction between trim parameters and their combined impact on objectives, serving as a valuable tool for making informed engineering decisions.

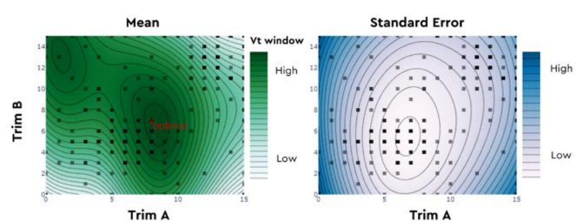

Fig. 6 Objective dependence on the interactive trim parameters provided by BO.

Results and Discussion

On our state-of-the-art BiCS8 platform, over 20 parameters are selected for the OFTO efficiency test. These parameters encompass various electrical bias, clock timings (which control wordline/bitline settling time), and NAND programming algorithms. The optimization focuses on two key objectives: Vt-window, defined as the cumulative span of non-overlapping regions within the threshold voltage distributions, serving as a proxy for cell state separation and inversely proportional to bit error rate (BER); and t_{PROG}, representing program speed and inversely proportional to program bandwidth. Traditionally, this task with a complexity of 2^{160} could require an experienced engineer several months to complete. However, with OFTO, convergence is achieved in less than 200 iterations, reducing the runtime to approximately 24 hours.

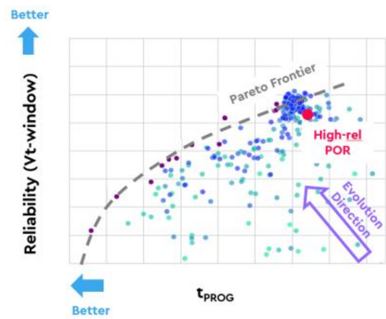

Fig. 7 BO Optimal Candidates vs. Trim selected by traditional method on optimization of performance and reliability metric.

Error! Reference source not found. shows the obtained Vt-window t_{PROG} in the last several searching steps (indicated by different colors) prior to convergence. Further iterations do not improve hypervolume and PF much after passing the convergence criteria. The advantage of OFTO technique is that it provides many optimal trim sets along the PF, depending on the priority of objectives (performance vs. reliability in this case) for different applications. In other words, multiple optimal trim sets can be obtained simultaneously. In contrast, objective priority has to be fixed (high reliability in this case) in traditional method, and hence only one optimal trim set can be obtained in one optimization cycle (red dot in **Error! Reference source not found.**, marked with "High-rel POR"). By comparing the PF from OFTO and the "High-rel POR" point, the best operation point at PF shows ~20% improvement in Vt window, thus significantly lower bit error rate, with the same t_{PROG}. The obtained Vt distribution from OFTO and traditional method are plotted in **Error! Reference source not found.**.

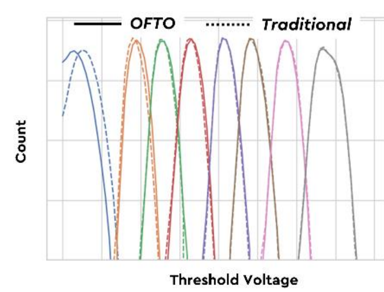

Fig. 8 TLC Vt distribution obtained from OFTO and traditional method. Arbitrary units are used in the plot.

Though BO is a powerful tool for optimizing expensive block-box function, it faces limitations in very high-dimensional problem due to the curse of dimensionality, which increases computational cost and reduces surrogate model accuracy as dimensionality grows. Specifically, BO model has a computational complexity of $O(n^3)$, where n represents the total number of evaluations [12]. Hence, optimizing acquisition function like EI becomes computationally expensive. In our demonstration, we limit the number of parameters to 20 to maintain the computational burden. Note that dimensionality reduction techniques, such as Subspace-Augmented BO (SASSBO) [13] have been recently developed to manage computational costs in high-dimensional problem. To reduce the complexity of the optimization problem and accelerate the optimization process, parameters known to be independent can also be decoupled and optimized in parallel. In addition to BO, evolutionary algorithm [14], such as Genetic Algorithms [15-17], known for their effectiveness in high-dimensional optimization and multi-objective optimization using non-dominated sorting, offer promising solutions for managing a large number of trim parameters.

Conclusion

In this study, we developed an on-the-fly machine learning model utilizing the Bayesian Optimization algorithm to advance the trim development. Applying this technique to our latest technology has led to substantial improvements in both performance and reliability, achieving device margins approximately 8% higher than those optimized by human experts. Moreover, the elimination of manual iterations has reduced the trim development cycle by up to 80%. This benefit is expected to scale further as the number of parameters continues to grow in future 3D NAND. This approach sets the stage for highly efficient trim development, paving the way for 1,000-layer 3D NAND.

References

[1] A. Goda, “3-D NAND Technology Achievements and Future Scaling Perspectives,” in IEEE Transactions on Electron Devices, vol. 67, no. 4, pp. 1373-1381, April 2020, DOI: 10.1109/TED.2020.2968079.

[2] K. Parat, A. Goda, “Scaling Trends in NAND Flash” 2018 IEEE International Electron Devices Meeting(IEDM). IEEE, 2018.

[3] R. Frickey, J. Doller, R. Norton, R. Sancho, R. Sayyad, and D. Ustinov , “Comparing the Reliability of Solid-State Drives Based on TLC and QLC NAND Flash Memories,” 2024 IEEE International Reliability Physics Symposium (IRPS), Grapevine, TX, USA, 2024, pp. 3A. 1-1-3A. 1-7.

[4] A. Khakifirooz et al., "A 1.67Tb, 5b/Cell Flash Memory Fabricated in 192-Layer Floating Gate 3D-nand Technology and Featuring a 23.3 Gb/mm2 Bit Density," in IEEE Solid-State Circuits Letters, vol. 6, pp. 161-164, 2023, DOI: 10.1109/LSSC.2023.3285508.

[5] J. Kim, J. Lee and S. Yoo, "Machine Learning-Based Automatic Generation of eFuse Configuration in NAND Flash Chip," 2019 IEEE International Test Conference (ITC), Washington, DC, USA, 2019, pp. 1-9, DOI: 10.1109/ITC44170.2019.9000162.

[6] F. Peter “ A tutorial on Bayesian optimization”, 2018, arXiv preprint: 1807.02811.

[7] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

[8] J. Wang, “An intuitive tutorial to Gaussian processes regression” Computing in Science & Engineering, 2023.

[9] K. P. Murphy, Machine learning : a probabilistic perspective. Cambridge (Ma): Mit Press, 2012.

[10] D. Duvenaud, “Automatic model construction with Gaussian processes.” Doctoral dissertation, 2014.

[11] D. Jones, M. Schonlau, and W. Welch, “Efficient global optimization of expensive black-box functions.”,1998, Journal of Global Optimization, 13(4):455-492.

[12] M. Malu, G. Dasarathy, and A. Spanias, “Bayesian Optimization in High-Dimensional Spaces: A Brief Survey,” IEEE Xplore, Jul. 01, 2021.DOI: https://ieeexplore.ieee.org/document/9555522.

[13] D. eriksson, M. Jankowiak, “High-Dimensional Bayesian Optimization with Sparse Axis-Aligned Subspaces.”, 2021, Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence.

[14] E. Zitzler, K. Deb, and L. Thiele, “ Comparison of multiobjective evolutionary algorithms: Empirical results.” 2000, Evolutionary computation, 8(2) : 173-195.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[16] K. Deb and H. Jain, “An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints,” IEEE Transactions on Evolutionary Computation, vol. 18, no. 4, pp. 577–601, Aug. 2014, DOI: https://doi.org/10.1109/tevc.2013.2281535.

[17] H. Seada and K. Deb, “A Unified Evolutionary Optimization Procedure for Single, Multiple, and Many Objectives,” IEEE Transactions on Evolutionary Computation, vol. 20, no. 3, pp. 358–369, Jun. 2016, DOI: https://doi.org/10.1109/tevc.2015.2459718