Grouped Feature Selection for SMONs Placement in MCU Performance Screening

Grouped Feature Selection for SMONs Placement in MCU Performance Screening

Nicolò Bellarmino*, Riccardo Cantoro*, Martin Huch†, Tobias Kilian† and Giovanni Squillero*

*Politecnico di Torino Torino, Italy
†Infineon Technologies AG Munich, Germany

***Abstract*—In safety-critical applications, microcontrollers (MCUs) must meet stringent performance requirements, particularly in terms of maximum operating frequency ($F_{\max}$). On-chip Speed Monitors (SMONs), often implemented as ring oscillators, are commonly used to determine $F_{\max}$. Previous works showed that frequency from SMONs provide valuable data for predicting $F_{\max}$ via machine learning (ML) models. A common approach is to group SMONs and place them into several spots on the device. However, while increasing the number of groups can enhance predictive accuracy, it also raises production costs, power consumption, and physical area on the chip. Strategically determining the number and placement of SMON groups is therefore essential.**

We leverage Group Lasso regularization to selectively retain only the most informative SMONs, optimizing spatial coverage while controlling feature dimensionality and ML model accuracy. By incorporating a custom optimization metric in the model training process, this approach balances prediction accuracy with groups count, allowing for cost-effective SMON integration. Our method achieves robust performance with fewer SMONs groups, reducing production costs and silicon area requirements in the final MCU design.

***Index Terms*—Fmax, Speed Monitors, Ring Oscillators, Speed Binning, Machine Learning, Device Testing, Manufacturing, Feature Selection**

I. Introduction

Microcontroller (MCU) performance screening aims to identify devices that do not meet the specified characteristics, such as the maximum operating frequency ($F_{\max}$) indicated in the datasheet. Existing literature has demonstrated the effectiveness of Machine Learning (ML) models trained on data correlated with $F_{\max}$ in accurately predicting the operating frequency [1]–[4]. Previous research has advocated for the use of on-chip Ring Oscillators (ROs), also denoted as Speed MONitors (SMONs) to predict $F_{\max}$ values [4]–[7]. In principle, having numerous predictor sensors may increase the information about the device's speed. Another important aspect is where to place SMONs on the chip: depending on their placements, SMONs can capture different information about the speed of the devices.

SMONs have to be spatially distributed to cover the Within-Die (WID) process variation that is increasingly emerging in shrinking technologies [8].

Authors are listed in alphabetical order.

978-1-6654-7763-5/25/$31.00 ©2025 IEEE

In order to handle such requirements, an SMON module is developed such as it contains a heterogeneous set of different SMONs. Different SMON modules are placed in multiple locations on the MCU to ensure spatial coverage, leading to a more accurate estimation of the relation between SMON values and $F_{\max}$.

However, limitations from both technological and statistical front arises: practically speaking, having hundreds of features for ML models potentially causes overfitting because of the *Curse of Dimensionality* (COD): in higher dimensional space, data tends to be sparse, and more and more labeled data are needed to estimate a reliable relation between features and target. But since obtaining accurate $F_{\max}$ value is a time-consuming process, we often have only a limited availability of labeled data, in the scale of hundreds of samples.

Also, since SMONs serve testing purposes exclusively and hold no functional value for customers, minimizing them is essential.

This paper addresses this challenge by focusing on grouped-feature selection, based on group-regularized linear models, with the goal of reducing the number of features and SMONs module (and thus, the number of SMONs) required for building ML models. We compared the proposed approach with a brute force approach.

Experimental results showcase the viability of this approach, demonstrating to be effective in significantly reducing the dimensionality of the feature space without compromising prediction performance. This approach gives test engineers information about the best location in which to place SMONs, in fast and straightforward way with respect to traditional brute force approach.

Reducing the number of SMONs contributes both to efficient prediction models and cost savings and optimized use of silicon space in the microcontroller.

The rest of the paper is organized as follows. Section II presents related work on the topic. Section III describes theory and concept useful for understanding successive experiments; in particular, Section III-A describes the characterization process used to derive the dataset for ML algorithms; Section III-B describes the SMONs used as features, while Section III-C introduces the concepts of ML and Feature Selection and Section III-D describes the regularized linear models setting. In Section IV, the motivations why we need feature selection are given. In Section V, details on the

proposed approach are given. Sections VI and VII presents the experimental setup and the evaluation. Finally, Section VIII draws the conclusions.

II. Related Work

Several approaches to performance prediction have been proposed in the past [9]–[12]. In the literature, using indirect measures to predict circuit parameters is called 'alternate test' and has been widely studied for analog circuits [13]–[16]. The core idea is to learn a mapping between indirect measurements and some circuit parameters, and to use only the indirect low-cost measurements to predict circuits parameters during production testing.

The authors of [1]–[3] worked on building ML models for $F_{\max}$ prediction, to be used in MCU performance screening. In [2], they correlated the frequency values of 27 on-chip SMONs to functional $F_{\max}$. Dimensionality reduction is a well-covered topic both in the ML [17], [18] and CAD communities, with several works in the realm of alternate tests for analog circuits [19], [20]. The importance of feature selection in MCU performance screening was firstly addressed in [21]. However, existing methods often rely on filtering or wrapper approaches. Filtering approaches are usually univariate (considering only one feature at once). Wrapper methods like Sequential with Forward and Backward feature selection or RFE, instead, usually have a high computational cost and dependency on the model and data, with the risk of overfitting [22].

Two main techniques exist for dimensionality reduction: *features selection* (FS) and *features extraction* (FE). FS selects subsets of features based on some criteria (like the effectiveness in predicting the target label). FE builds a new and smaller set of features as a combination of the original ones, compacting the information on the dataset. Principal Component Analysis (PCA), a feature extraction technique [23], has been effectively used in related works on SMONs [4].

III. Background

A. Microcontroller Characterization Process

Frequencies from on-chip ROs, referred to also as SMONs, provide features for the ML model. These frequencies are accurately measured during production using a stable, fast, and straightforward process. As measuring SMON frequencies is part of the regular production test, these features are potentially available for every produced Microcontroller (MCU). However, training ML models necessitates a properly labeled dataset, requiring MCU characterization.

The labeling process is time-consuming, involving measuring each MCU individually with functional test patterns [2]. This process is conducted on a small subset of manufactured devices. The labeling procedure includes mounting each MCU on a board and executing a specific functional pattern with a low frequency, incrementally increasing until a functional failure occurs, with the last working frequency $F_{\max}$ being recorded [24]. This process is repeated using various functional test patterns, resulting in a multi-label dataset. For each device, the most critical pattern is the one with the lowest $F_{\max}$ value. In order to ensure robustness in the measurement process and exclude outliers from the training procedure, devices with $F_{\max}$ deviating by more than 2.5 standard deviations from the wafer median in at least one functional pattern are eliminated. This ensures a high-quality set of labeled devices for ML training.

B. The SMONs

The structure and sensitivity of the SMONs significantly influence the performance prediction model. In order to address this, the goal is to incorporate diverse SMON variants into the MCU. They also need to be spatially distributed to account for Within-Die (WID) process variation, which becomes increasingly prominent as the feature sizes of manufacturing technologies are further reduced [8]. In the early stage of production, test engineers must decide which SMONs to place on board and where.

Thus, an SMON module is designed to contain a heterogeneous set of various SMONs. Multiple instances of identical SMON modules are strategically placed throughout the MCU to ensure comprehensive spatial coverage.

An SMON module comprises both generic and design-dependent Ring Oscillators (ROs). Generic ROs include inverter gates, NAND gates, and NOR gates, which are sourced from the standard cell libraries used in the design of the MCU. Design-dependent ROs, on the other hand, are replicated functional paths derived directly from the design.

It is possible to choose from 130 different monitors to be placed on compact blocks distributed across the die. The location of these SMONs is fundamental, as it is linked to spatial variations, also known as "Global Mismatch."

The feature set includes SMONs from multiple SMON modules, providing a robust foundation for understanding the chip's behavior across various scenarios.

To streamline the information from different SMON modules, aggregation can be employed. For each SMON in various modules, a single value is extracted using an aggregation function, such as the mean or the median. This approach, referred to as the "Virtual Module (VM)" [21], serves the dual purpose of reducing the number of SMONs analyzed by the ML model while capitalizing on the correlations and variations among SMONs at different locations. The VM strategy allows for a simplified ML algorithm, even in the presence of multiple SMON modules, while reducing variance and removing potential outliers from the measured SMON values.

C. Machine Learning

Training an ML model involves establishing a relationship between inputs and outputs based on available labeled data. The model evaluates certain features to generate an output. For instance, simple linear regression algorithms combine input features linearly using weights assigned during training.

The number of samples required to accurately estimate a function increases exponentially with the number of input

variables, a phenomenon known as the curse of dimensionality [25]. In simpler terms, having more features necessitates a larger dataset to construct robust ML models. To address this issue, dimensionality reduction techniques such as FS can be employed [18]. There are three main categories of feature selection methods: filtering, wrapper, and embedded methods [18].

In particular, some ML algorithms are capable of performing feature selection internally; these are referred to as *embedded methods*. In such methods, feature selection is an automatic part of the model training process, eliminating the need for separate feature selection steps [26]. Among the various embedded selection methods [27], Regularized Linear Models stand out as a fast and straightforward approach for performing feature selection during model training.

D. Regularized Linear Models

Regularized linear add a penalty term (usually the L_p norm) computed on the coefficients' vector in the ordinary least-square linear regression objective function:

$$\min_{w} \frac{1}{2n_{\text{samples}}} ||Xw - y||_2^2 + \alpha \left(\sum_{i=1}^{n} |w_i|^p \right)^{\frac{1}{p}} \tag{1}$$

Least Absolute Shrinkage and Selection Operator (Lasso) Regression, ElasticNet, and Orthogonal Matching Pursuit (OMP) [28]–[30] are popular regularized linear models with internal feature selection. Lasso is a linear regression model with L_1 regularization. It tends to drive some coefficients to exactly zero, effectively performing FS. ElasticNet is an extension of Lasso that combines L_1 and L_2 regularization. While L_0 regularization is non-convex and computationally challenging, some optimization methods like OMP approximate the fit of a linear model with constraints imposed on the number of non-zero coefficients (ie. the L_0 pseudo-norm).

Group regularization is an extension of standard regularization techniques that is particularly useful when features or variables are structured in groups. Traditional regularization methods, such as Lasso, impose penalties on individual coefficients to encourage sparsity and prevent overfitting. However, in many real-world applications, variables are naturally organized into groups based on prior knowledge or inherent structure. Group regularization takes this grouping into account by encouraging sparsity at the group level rather than at the individual feature level.

One of the most widely used forms of group regularization is the *Group Lasso*. Introduced in [31], Group Lasso extends the Lasso by applying an ℓ_2 norm to each group of coefficients and an ℓ_1 norm across groups. This results in either all coefficients within a group being selected or none of them, promoting group-level sparsity. The optimization problem for Group Lasso can be written as:

$$\min_{w} \frac{1}{2n_{\text{samples}}} \|Xw - y\|_2^2 + \lambda \sum_{g=1}^{G} \|w_g\|_2, \tag{2}$$

where y represents the response or target vector, X is the design matrix, w is the vector of coefficients, λ is a regularization parameter, and w_g refers to the coefficients of the g-th group. Group Lasso is particularly effective in scenarios where variables in the same group are correlated or where it is desirable to select groups of features together.

It should be noted that when there is only one group, ($G = 1$), Group Lasso is equivalent to Ridge; when each weight forms an independent group, i.e., ($G = m$, with m the number of features), Group Lasso becomes Lasso.

IV. Motivations

As discussed in Section III-B, incorporating a greater number of SMONs at various locations on the MCU could enhance the handling of WID process variation, providing more comprehensive insights into MCU performance. Having a multitude of SMONs might contribute to superior performance predictions: they can catch several information about the devices speed, and placing them at different device location would permit to include positional information in the features. However, practical constraints limit the feasibility of this approach for three key reasons:

- **Physical Area and Overhead**: Each additional SMON module integrated into the design occupies physical space on the chip. Since, each module compromised hundreds of SMONs, incorporating a multitude of them may leads to a noticeable area overhead, contributing to current leakages. Given that SMONs serve testing purposes exclusively and hold no functional value for customers, minimizing the occupied area is essential.
- **Curse of Dimensionality**: From an ML perspective, an increased number of SMONs results in a higher feature count for the predictive models. Depending on the product under consideration (and thus, on the dataset), this may lead to suboptimal models due to the COD, or models with a higher footprint than needed (in terms of number of coefficients).
- **Interpretability** : The higher the number of SMONs, the more difficult is to catch which of them is relevant for the performance prediction task.

Technologically, a reduced feature set lowers production costs by incorporating only the most informative sensors into future products, and also mitigates the associated physical space and current leakage concerns. Also, having an idea on the best SMONs module location help test engineers in essential decision during design phase.

Reducing the feature set also contributes to enhanced generalization performances of ML models, allowing for simpler models that operate on fewer inputs. This strategic reduction aligns with both technological and machine learning considerations, facilitating cost-effective and efficient integration of SMONs into MCU designs.

Also, opting for a linear model over more sophisticated methods has several advantages including, simplicity and interoperability, computational efficiency, avoidance of overfitting, and small data requirements.

A. Limitations of the Brute-Force Approach

The brute-force approach, as detailed in [21], while providing optimal SMON module placement, suffers from several limitations that make it impractical for wide application:

- **Computational Complexity**: It requires training a model for every possible combination of SMON modules. As the number of SMONs grows, the number of combinations increases exponentially. For example, chosing k SMON modules across in n modules requires evaluating $\binom{n}{k}$ combinations, which makes the approach computationally intractable as n and k rise. The computation time for evaluating every combination may take several days on a dedicated server, as evidenced by previous work [21].
- **Scalability**: Each time a new device family is introduced, the evaluation process should be repeated. For large-scale manufacturing processes involving multiple device families, this repetition significantly impacts the time-to-market for new products. As the complexity of the design increases, the computational time for brute-force search also escalates.
- **Inefficiency**: The brute-force method explores many redundant or suboptimal SMON combinations, wasting computational resources on configurations that are unlikely to provide useful insights. Many of the combinations evaluated may contribute little to overall predictive accuracy or even introduce noise into the models, leading to overfitting.

  The large computational overhead associated with brute-force search not only increases time and resource usage but also escalates costs. Delaying the development cycle is not justifiable when alternative methods can provide nearly equivalent solutions in a fraction of the time.

For these reasons, an alternative approach is required to achieve optimal SMON placement.

V. Proposed Approach

The proposed approach is based on the use of Group Lasso regularization to address the limitations of the brute-force method. Group Lasso is particularly suited for this task as it enables the selection of entire groups (or modules) of SMONs, rather than individual sensors, by adding a regularization term that penalizes the inclusion of unnecessary groups. This allows for efficient pruning of SMON modules, reducing the physical overhead on the MCU.

Group Lasso regularization technique allows for an efficient selection of SMON modules while still achieving satisfactory predictive performance. However, the key to successful implementation lies in controlling the regularization strength, λ, in the Group Lasso objective function, Eq. (2). This plays a critical role in determining the balance between the number of SMON modules included and the model's predictive performance. A high value of λ increases the penalty on the number of SMON modules, leading to a smaller selected set but potentially sacrificing accuracy. Conversely, a low value of λ results in higher accuracy but at the expense of a larger number of SMON modules.

To account for both factors—prediction error and the number of SMON modules—we propose modifying the objective function in a grid search process as follows:

$$\text{score} = \text{MAE} + \alpha \times \text{SM} \tag{3}$$

In this equation:

- MAE is the mean absolute error of the predictive model, which reflects the prediction accuracy.
- α is a hyperparameter that determines the trade-off between prediction accuracy and the number of SMON modules included in the final design.
- SM represents the number of SMON modules selected by the Group Lasso model.

By adjusting α, we can prioritize either minimizing the number of SMON modules or maximizing the predictive accuracy, depending on the design constraints and performance targets of the MCU, achieving a balance that meets both technological and performance requirements.

The equation means that for each additional SMON module added, we accept a reduction in the Mean Absolute Error (MAE) of α Hz. In other words, the parameter α controls the trade-off between prediction accuracy and the inclusion of new SMON modules. A higher α implies greater tolerance for a decrease in prediction accuracy in exchange for fewer SMON modules, while a lower α emphasizes predictive accuracy, leading to the selection of more SMON modules.

The advantages of this approach over the brute-force one are several: by linking the number of SMON modules with the MAE, this approach allows us to decide a priori how much accuracy we are willing to sacrifice by reducing the SMON modules. In other words, we are introducing a direct control on the amount of predictive power we can sacrifice to prioritize the reduction in the number of models chosen. This flexibility enables engineers to make informed decisions based on the specific technological constraints and performance requirements of the MCU design, ensuring an optimal balance between accuracy and resource utilization.

Also, with a single run of training, it is possible to select the best combination of SMON modules. This speed up not only the training process but the whole development of new products, as test engineers can have key information about SMON modules placement.

VI. Experimental Setup

The proposed methodology has been validated on a dataset is composed of 1148 labeled samples. In this product, 6 different SMONs modules are placed on the chip, each compromising of about 130 SMONs.

To validate our approach, we eavluated Eq. (3) with 21 values of α. For each α, the evaluation was performed with a 5-folds CV stratified per wafer, that permitted chosing the optimal value of the parameters λ of Eq. (2) across a grid of 20 values.

All experiments were performed in Python. Experiments run on a server equipped with a dual-socket AMD® EPYC 7301 16-Core CPU @ 3.20GHz, 128GB of RAM.

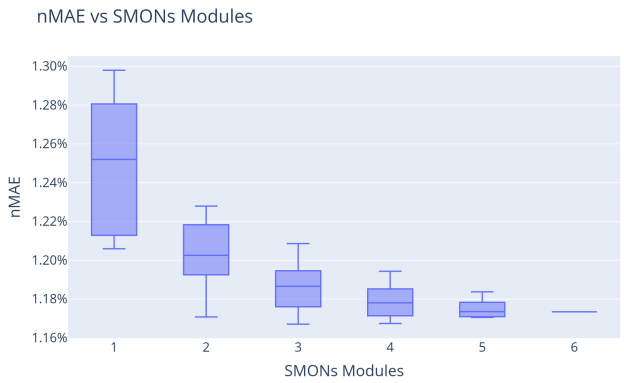

Fig. 1. Box plot of nMAE across different SMON module counts. The distribution shows a decrease in median nMAE and variance as the number of modules increases, indicating improved accuracy and stability.

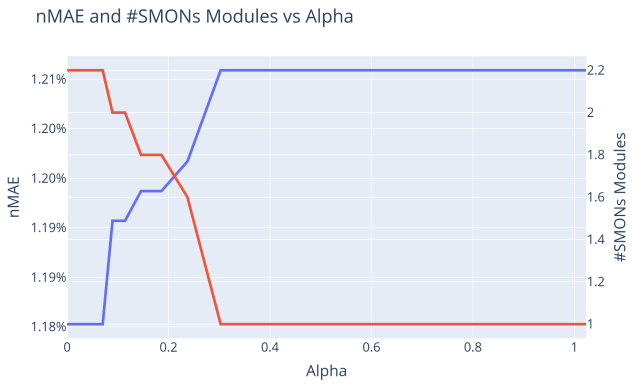

Fig. 2. Alpha vs SMONs modules. Red curve is the number of modules while blue curve is the nMAE, varying alpha in Eq. (3).

Features are standardized by removing the mean and scaling to unit variance, as a pre-processing step (*Standard Scaler*).

As the final model, we used a pipeline composed of the Standard Scaler and a Linear Ridge Regressor.

Results are presented in terms of normalized Root Mean Square Error (nRMSE), normalized Mean Absolute Error (nMAE) and the coefficient of determination (R^2). RMSE and MAE are popular regression performance indexes [32] but in this context, we normalized them by the mean value of $F_{\max}$ in the test set, i.e. $nRMSE = RMSE(y_{true}, y_{pred})/mean(y_{true})$, $nMAE = MAE(y_{true}, y_{pred})/mean(y_{true})$ to obtain a percentage of the error concerning the mean frequency of the samples. R^2 is the proportion of the variation in the dependent variable that is predictable from the independent variable(s) [33]. A regressor that perfectly fits the data would have an R^2 score of 1 (or 100%). A dummy regressor that always predicts the mean value of the target has an R^2 score of 0.

VII. Experimental Results

A. Brute Force Baseline

The impact of SMON module selection on the nMAE in performance prediction was thoroughly evaluated through a brute-force approach, where all possible combinations of modules were tested to determine the effect on prediction accuracy. In other words, the whole powerset of all the SMONs modules should be considered (i.e. is the set of all subsets). For a set with X elements, the powerset (excluding the empty set) consists of $2^X - 1$ subsets. For 6 SMONs modules, we must consider 63 combinations. For 10, these increase to 1023, making the evaluation not tractable.

Figure 1 displays a box plot of nMAE distributions for different counts of SMON modules. As the number of modules increases, the median nMAE decreases, accompanied by a reduction in variance, especially as module count reaches higher levels. This trend indicates that including more SMON modules generally enhances prediction accuracy by capturing more spatial variation across the chip. However, this improvement stabilizes beyond a certain threshold—specifically, with more than three SMON modules, additional modules contribute marginal gains in accuracy. For this product, adding more than three modules yields diminishing returns, emphasizing the importance of a selective approach to SMON placement rather than indiscriminate increases in module count. In particular, it was found that the best combination is composed by module {0,3,5}, with an nMAE of 1.16%. A summary of the results can be found in Table I: including more than 3 modules do not contribute to enhancing the predictive power.

TABLE I
Error metrics (nMAE, nRMSE, R^2) for best SMON module combinations of different sizes, with indication if found by Group Lasso.

# Modules	nMAE (%)	nRMSE (%)	R^2	Found by Group Lasso
6	1.17	1.47	88.45	No
5	1.17	1.46	89.01	No
4	1.16	1.46	89.11	No
3	**1.16**	**1.46**	**89.11**	**Yes**
2	1.17	1.47	89.08	Yes
1	1.21	1.50	89.03	Yes

B. Group Lasso Selection

By employing Group Lasso, we achieve simultaneous model training and SMON module selection. Figure 2 illustrates the trends in both nMAE and the number of selected SMON modules as the regularization parameter α varies in Eq. (3). At very low or very high values of α, we observe a plateau in the number of selected SMON modules. Specifically, when no constraint is applied in the optimization, Group Lasso selects an average of 2 to 3 SMON modules across 5 runs, with a mean of 2.2 modules, resulting in the lowest prediction error. As α increases, fewer SMON modules are selected, highlighting the method's ability to prioritize key modules as regularization strengthens.

Across multiple runs of the Group Lasso procedure that contributed to Figure 2, only four distinct combinations of SMON modules were consistently selected (with 3, 2, and 1 SMON modules, as depicted in Table I). This demonstrates the effectiveness of Group Lasso in identifying minimal yet

highly predictive sets of modules, while excluding ineffective and redundant groups.

The selection of α in Eq. (3) should involve collaboration between design experts, test engineers, and data analysts. If there are no constraints on the number of SMON modules, setting α to zero results in the optimal combination of SMON modules. However, when a tradeoff between predictive power and the number of SMON modules is required, increasing α can effectively prune the SMON modules.

Finally, one advantage of Group Lasso is the ability to avoid a brute-force evaluation of all possible combinations. By fixing α, a single model training can yield the ML model trained with the optimal combination of SMON modules, without considering all other combinations. This is a significant advantage, as the number of combinations increases exponentially.

VIII. Conclusions

We presented an SMON module selection framework to be used in MCU performance screening. SMON frequency values serve as alternative measures for performance prediction, allowing industries and test engineers to leverage a large number of SMONs, grouped into modules, to capture performance variation. Our framework enables the optimal selection of SMON modules based on their importance in the performance prediction task, utilizing group regularization techniques capable of pruning entire sets of SMONs.

The experiments demonstrated that this approach can effectively trace the results of brute-force methods with significantly less computational effort, as the selection of SMON modules is achieved by training the models only once. The training score introduced in this work allows for the determination of a trade-off between the number of selected modules and model accuracy by fixing the parameter α, thereby deciding a priori the cost of placing each additional module.

This approach can be generalized for any scenario where group features need to be pruned.

References

[1] N. Bellarmino *et al.*, "Microcontroller Performance Screening: Optimizing the Characterization in the Presence of Anomalous and Noisy Data," in *IEEE International Symposium on On-Line Testing and Robust System (IOLTS)*, 2022.

[2] R. Cantoro *et al.*, "Machine Learning based Performance Prediction of Microcontrollers using Speed Monitors," in *IEEE International Test Conference (ITC)*, 2020.

[3] N. Bellarmino *et al.*, "Exploiting active learning for microcontroller performance prediction," in *IEEE European Test Symposium (ETS)*, 2021.

[4] N. Bellarmino *et al.*, "A Multi-Label Active Learning Framework for Microcontroller Performance Screening," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2023.

[5] J. Chen *et al.*, "Data learning techniques and methodology for Fmax prediction," in *IEEE International Test Conference (ITC)*, 2009.

[6] J. Chen *et al.*, "Selecting the most relevant structural Fmax for system Fmax correlation," in *28th VLSI Test Symposium (VTS)*, 2010.

[7] M. Sadi *et al.*, "SoC Speed Binning Using Machine Learning and On-Chip Slack Sensors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2017.

[8] S. Asai, Ed., *VLSI Design and Test for Systems Dependability*. Springer Japan, 2019.

[9] K. von Arnim *et al.*, "An Effective Switching Current Methodology to Predict the Performance of Complex Digital Circuits," in *IEEE International Electron Devices Meeting (IEDM)*, 2007.

[10] G. Sannena *et al.*, "Low overhead warning flip-flop based on charge sharing for timing slack monitoring," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2018.

[11] T. B. Chan *et al.*, "DDRO: A novel performance monitoring methodology based on design-dependent ring oscillators," in *Thirteenth International Symposium on Quality Electronic Design (ISQED)*, May 2012.

[12] F. Angione *et al.*, "Test, reliability and functional safety trends for automotive system-on-chip," in *2022 IEEE European Test Symposium (ETS)*, 2022.

[13] H. Ayari *et al.*, "Making predictive analog/rf alternate test strategy independent of training set size," in *IEEE International Test Conference (ITC)*, 2012.

[14] P. Variyam *et al.*, "Prediction of analog performance parameters using fast transient testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2002.

[15] H.-G. Stratigopoulos *et al.*, "Error moderation in low-cost machine-learning-based analog/rf testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 2008.

[16] J. Brockman *et al.*, "Predictive subset testing: Optimizing ic parametric performance testing for quality, cost, and yield," *IEEE Transactions on Semiconductor Manufacturing*, 1989.

[17] W. Jia *et al.*, "Feature Dimensionality Reduction: a Review," *Complex & Intelligent Systems*, Jun. 2022.

[18] I. Guyon *et al.*, "An Introduction to Variable and Feature Selection," *The Journal of Machine Learning Research*, Mar. 2003.

[19] S. Larguech *et al.*, "Efficiency evaluation of analog/rf alternate test: Comparative study of indirect measurement selection strategies," *Microelectronics Journal*, 2015.

[20] M. J. Barragan *et al.*, "A procedure for alternate test feature design and selection," *IEEE Design and Test*, 2015.

[21] N. Bellarmino *et al.*, "Feature Selection for Cost Reduction In MCU Performance Screening," in *IEEE 24th Latin American Test Symposium (LATS)*, 2023.

[22] U. M. Khaire *et al.*, "Stability of feature selection algorithm: A review," *Journal of King Saud University - Computer and Information Sciences*, 2022.

[23] I. T. Jolliffe *et al.*, "Principal component analysis: a review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Apr. 2016.

[24] R. McLaughlin *et al.*, "Automated Debug of Speed Path Failures Using Functional Tests," in *27th IEEE VLSI Test Symposium*, 2009.

[25] R. Bellman, *"Adaptive Control Processes: A Guided Tour"*, 1961.

[26] N. Bellarmino *et al.*, "Embedded Feature Selection in MCU Performance Screening," in *IEEE 2nd International conference on Design, Test and Technology of Integrated Systems (DTTIS)*, 2024.

[27] Q. Lv *et al.*, "Enhanced-Random-Feature-Subspace-Based Ensemble CNN for the Imbalanced Hyperspectral Image Classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2021.

[28] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, 1996.

[29] H. Zou *et al.*, "Regularization and Variable Selection via the Elastic Net," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 2005.

[30] J. A. Tropp *et al.*, "Signal Recovery From Random Measurements Via Orthogonal Matching Pursuit," *IEEE Transactions on Information Theory*, 2007.

[31] M. Yuan *et al.*, "Model Selection and Estimation in Regression with Grouped Variables," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, Dec. 2005.

[32] T. Chai *et al.*, "Root Mean Square Rrror (RMSE) or Mean Absolute Error (MAE)?– Arguments Against Avoiding RMSE in the Literature," *Geoscientific Model Development*, Jun. 2014.

[33] D. Chicco *et al.*, *The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation*, en, Jul. 2021.