A Comprehensive Learning-Based Flow for Cell-Aware Model Generation

Pierre D'hondt, Aymen Ladhar, Patrick Girard, Arnaud Virazel

A Comprehensive Learning-Based Flow for Cell-Aware Model Generation

P. d'Hondt [1, 2] A. Ladhar [1] P. Girard [2] A. Virazel [2]

[1] STMicroelectronics
Crolles, France
pierre.dhondt@st.com, aymen.ladhar@st.com

[2] LIRMM – Univ. of Montpellier / CNRS
Montpellier, France
girard@lirmm.fr, virazel@lirmm.fr

Abstract— As the semiconductor industry continues to shrink the transistor feature size, new fault models need to be invented and deployed to ensure manufacturing test and diagnostic of the highest quality. The Cell-Aware (CA) test and diagnosis methodology targets the detection of defects inside standard (std) cells, at the transistor level. While becoming an industry standard, the CA methodology, has a large and costly deployment overhead, involving numerous analog simulations. In [1], we presented an innovative flow using Machine-Learning (ML) to reduce the CA test method runtime and ease its adoption for industrial usage. Experiments using different technology nodes demonstrated an over 99% runtime reduction for 80% of combinational cells. In this paper, new elements are presented to more widely take advantage of the ML flow for CA characterization. This includes a new decision algorithm, leveraging ML techniques to decide whether the CA characterization of a new std cell should be ML-based or simulation-based, thus allowing to decrease the CA characterization runtime while maintaining high quality CA models for all cells. Experimental results demonstrate the high performance of the new decision algorithm. The fault coverage on real cell-internal defects of ATPG patterns using ML predicted CA data proves that our predicted CA data can accurately replace those obtained by running extensive analog simulations, thus proving the effectiveness and pertinence of the proposed methodology.

Keywords—Intra-cell defects, Standard cell characterization, cell-aware models, Machine-learning, Test and diagnostic

I. INTRODUCTION

Cell-Aware (CA) test and diagnosis techniques have been introduced to target defects located inside the std cells (referred to as *intra-cell* or *cell-aware defects*) which are only fortuitously covered by traditional fault models and therefore are often found to be the root cause of a significant fraction of test escape with modern technology nodes [2-5]. CA techniques rely on the realistic assumption that the excitation of a defect inside a std cell is correlated with the logic values applied to its inputs [5-6]. When deploying CA methodology for a new technology each std cell is characterized with respect to all possible cell-internal defects. This CA characterization process is traditionally performed using analog (SPICE) simulations identifying which cell-internal defect is detected by which cell-pattern. The simulations results are then organized into a cell-internal-fault dictionary called CA model (also known as CA test model) [7-8]. As std cells may have more than 10 inputs and thousands of std cells with different complexities are used for a given technology, the generation time of CA models for complete std cell libraries of a given technology may reach up to several months, thus drastically increasing the library characterization process cost [9-10]. Decreasing the CA models generation runtime is mandatory to make it a standard in the qualification process of silicon products [11]. To this end, a breakthrough methodology based on Machine Learning (ML) was proposed in [1] for generating CA models for combinational std cells.

To mitigate limitations presented in [1] and allow an ideal use of the fast ML-based CA model generation, this paper presents a new version of the so-called hybrid-flow, which mainly consists of a new decision block orienting cells between the fast ML-based CA model generation flow or the one based on analog simulations. This new decision block is based on a ML algorithm and replaces the structural analysis-based decision block in the hybrid CA model generation flow. Compared to the decision block in [1], this new decision block uses more data. The hybrid flow now considers combinations of (cell; defect type) as candidates for prediction rather than solely the cell itself. This fine-graining increases flexibility and the usage of ML-based CA models generation part of the flow. Experiments carried out on cells from five technologies show that the new decision block is significantly more accurate when selecting cells to be characterized with either the simulation-based CA model generation part of the flow or the ML-based CA model generation part.

To validate our overall workflow, this paper also presents novel experiments showing how CA models obtained using the new version of the hybrid flow are used to generate ATPG patterns for some ST industrial test chips. Results show that the test coverage achieved by using these patterns on actual defects is as high as the test coverage achieved by using patterns obtained only from the simulation-based CA models, thus demonstrating the quality of the ML-based CA models (now referred to as *ML-CA-models*) and the relevance of our method.

The rest of this paper is organized as follows. Section II presents the new version of the hybrid flow to overcome limitations presented in [1]. Section III presents experimental results using the new hybrid flow as well as some validations of the ML-CA-models quality on industrial designs. Section IV concludes this paper.

II. NEW DECISION BLOCK IN THE HYBRID FLOW

To exploit all the possibilities of a CA model generation based on ML, a new decision block with major novelties has

been developed. This new decision block replaces the one based on the structure analysis of cells presented in [1]. Its objective remains the same, i.e., orienting a std cell through the hybrid flow by choosing if the CA model for the new cell should be simulated or can be predicted by ML. A comparison of performance of the new decision block versus the previous one is proposed in Section III. Figure 1 is a schematic of the updated hybrid flow with the new decision block.

A. Presentation of the new decision block

The first major novelty relies on a new paradigm. Indeed, the decision block of [1] considered each std cell as atomic, meaning that a cell is characterized either through a ML-based CA model generation or through a simulation-based CA model generation. However, we observed that inside each cell, the prediction accuracies of each defect type vary. Table I illustrates this phenomenon. It contains averaged prediction accuracy for several types of cell-internal defects. The row labelled as "known structure" reports the values of accuracy for cells which have at least a cell with an identical or similar structure in the std cells database. The row labelled as "unknown structure" reports values for the other cells.

From Table I, we noticed that (i) the presence or absence of a cell with an identical/similar structure in the ML training group has a real impact on the prediction accuracy, confirming the result from [1]; and (ii) in some cases, a single badly predicted intra-cell defect type brings down the prediction accuracy of the entire cell under the quality criterion, preventing the cell from using a ML-based CA model generation. To alleviate this issue, fine-graining has been introduced in this work to mitigate this effect. Rather than considering the entire cell as a candidate for ML-based CA model generation, several combinations of (cell; cell-internal defect type) are considered independently. All components of the hybrid-flow have been modified to allow such fine-graining during the training and prediction or simulation phases, and then merging the results in a single CA model.

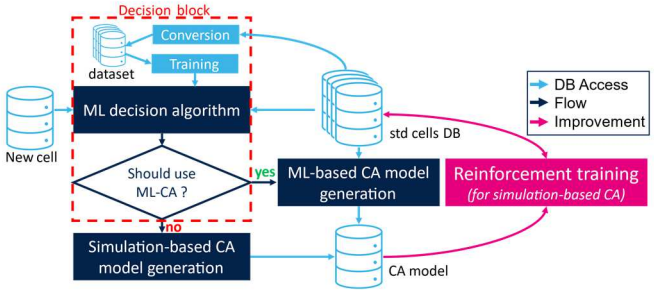

Fig. 1: Updated hybrid flow with the new decision block based on machine learning

The decision block now analyzes every combination of (cell; defect type) and decides which ones should have their behavior predicted by ML and which ones should be electrically simulated. After generation, the predicted and simulated results are merged in the CA model of the cell. This fine-graining allows to use of the ML algorithm more extensively, thus decreasing the number of analog simulations, and ultimately decreasing the run-time and cost. Note that the quality criterion is still used on a (cell; defect type) combination basis rather than simply on the cell basis, meaning that a badly predicted defect type will no longer prevent the cell to be entirely ML predicted, i.e., all eligible defect types will go through ML while the remaining ones will go through simulation. From now on, any % of population should be understood as a % of the total number of (cell; defect type) combinations. Moreover, every decision by the decision block is taken on a (cell; defect type) basis.

TABLE I. AVERAGE PREDICTION ACCURACY PER DEFECT TYPE FOR A POPULATION OF C40 CELLS.

	Open defects			**Short defect**		
	Source	**Drain**	**Gate**	**S-D**	**S-G**	**G-D**
Known structure	~97%	~97%	~90%	~99%	~93%	~81%
Unknown structure	~87%	~87%	~78%	<60%	<60%	<60%

The second major novelty of the new decision block is the use of ML. As already pointed out, the decision block is in charge of orienting (cell; defect type) combinations to the ML-based CA model generation or to the simulation-based CA model generation parts of the flow. The decision is taken by a new ML algorithm which is independent from the one used in the ML-based part of the hybrid flow. This ML-based decision algorithm (now referred to as *ML-decision*) should decide whether or not each (cell; defect type) combination should go through the ML-based part of the flow depending on what is available in the std cells database. This new ML-decision algorithm uses supervised learning and its specificities regarding the training phase will be presented hereafter.

Let us first consider the usage of such a ML-based decision block. To be oriented, the structure of the new cell is computed and compared to the ones available in the std cells database. The number of times the new cell's structure is present in the std cells database is used by the ML-decision algorithm. The ML-decision algorithm is bound to answer the question "Considering the content of the std cells database, should this (cell; defect type) combination use the ML-based CA model prediction part of the flow?". A 'yes/no' Boolean answer is expected. To this end, the ML-decision algorithm receives the following information as inputs: the cell's (i) name, (ii) function, (iii) drive strength, numbers of (iv) inputs and (v) transistors and (vi) structure. The ML algorithm also receives information about (vii) the presence or absence of the cell structure in the std cells database and (viii) the number of times the cell structure is present in the std cells database. The (ix) considered defect type is also provided.

B. Training of the new ML-decision algorithm

The supervised training of this ML-decision algorithm is the most challenging part of this work. The creation of a dataset for the training and evaluation of the ML-decision algorithm should be carefully considered, as it should account for the numerous situations that the ML-decision algorithm may encounter in the field. The dataset is a list of samples, each of them contains the information given as inputs to the ML-decision algorithm along with the expected 'yes/no' answer. The dataset allows both the training and the evaluation of the ML-decision algorithm.

The dataset should prepare the ML-decision algorithm to orient the (cell; defect type) combinations according to what is available on the std cells database at the time of decision. In a

real industrial application, the content of the std cells database depends on the history of the company (e.g., electrically simulated CA models anterior to the deployment of the proposed methodology) and on the history of the hybrid-flow usage, i.e., the simulated CA models obtained using the proposed methodology and added to the std cells database by the improvement loop introduced in Fig. 1. To deal with this changing database, the ML-decision algorithm should have a fast-training phase so it can be quickly trained every time the database changes or be compatible with reinforcement training. Our chosen ML-decision algorithm is again a Random Forest Classifier (RFC), which can be trained in a fraction of second over the dataset. Creating the first version of the dataset by scanning the std cells database takes a few minutes. Updating the dataset because a new cell has been added to the std cells database by the reinforcement loop takes a few seconds (which is negligeable compared to the simulation-based CA model generation run time). All these timings make the usage of the ML-decision algorithm realistic and appropriate.

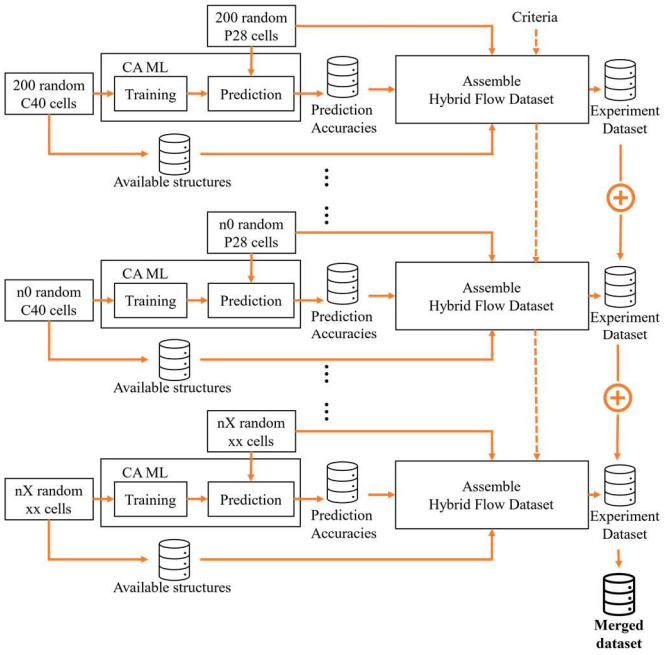

Fig. 2: Process to create a dataset for the ML-decision algorithm. It is composed of several independent experiments with changing parameters. The results of all experiments are merged in a main dataset.

To create a dataset accounting for a wide variety of industrial usages and std cell databases, the **process** depicted in Fig. 2 has been proposed. This process is composed of several **experiments**, each of them represents a different industrial setting and std cells database. Each experiment randomly chooses cells to constitute its own std cells database, then randomly choose cells for which it will generate ML-CA models. The experiment then generates CA models for the chosen cells using its std cells database and assess the quality (*accuracy*) of the prediction. Note that in the framework of this study, every cell has a CA model which has been obtained by simulations and acts as a reference when evaluating the prediction accuracy of the ML-based CA model generation. The accuracy of the prediction for each (cell; defect type) combinations is compared to a threshold (similar to the quality criterion presented in Section II.A). This comparison gives an answer to the (re-phrased) question: "Will this (cell; defect type) combination be correctly predicted, considering the content of the std cells database?". A prediction accuracy higher than the threshold leads to a '*yes*', while lower accuracy leads to a 'no'. Using the content of the cells database and the 'yes/no' answer, the experiment's dataset is assembled.

Figure 2 also shows the process where the experiments are repeated a large number of times with variations in the technologies and number of cells. The main merged dataset is the concatenation of all experiments' datasets. This process leads to a dataset which contains a large number of samples for various (cell; defect type) combinations with different std cells databases to represent the variety of industrial usage. The correctness of our ML-decision algorithm is evaluated using this dataset and the results are presented in the next section.

III. Experimental Results

This section presents experimental results using std cells coming from five technologies. In the first sub-section, the process of Fig. 2 is used to evaluate the new ML-decision block. The second sub-section uses the CA models generated by the hybrid flow with an ATPG to generate test patterns for several IC designs

A. Machine-Learning decision block evaluation

The experiments in Fig. 2 are repeated many times and lead to a dataset for the ML-decision algorithm. With such a dataset, the ML-decision algorithm can be evaluated using conventional ML techniques (e.g., k-fold). Our metric is the **decision error** which is the fraction of badly predicted answers which have a potential impact on the quality of CA model over the total number of samples in the validation dataset (the lower, the better). This fraction represents the number of times the ML-decision block takes the wrong decision when orienting a (cell; defect type) combination through one part of the hybrid flow, considering the content of the cells database. Among the two possible mis-decisions, only the false-positive one can have a negative impact on the quality of the CA models. The false-positive error consists in sending a (cell; defect type) combination through the ML-based CA model generation even though the content of the cells database is not sufficient to reliably predict the behavior of the defects in the cell. The other type of mis-decision consists in sending a (cell; defect type) combination through the simulation-based CA model generation even though its defective behavior could have been accurately predicted by the ML-based CA model generation part of the flow (false-negative). This false-negative mis-decision does not negatively impact the quality of the generated CA model and is therefore not accounted for in the decision error.

The experiments presented in this sub-section mimic the usage of the proposed hybrid flow, i.e., when the cells of a new library/technology need to be oriented toward one part of the hybrid flow by the decision block, the std cells database only contains cells from older technologies. We used library cells from five technologies (from older to recent): C40, C28, P28, P18 and FinFET16.

Table II presents the **decision error** of the ML-decision algorithm using a growing dataset detailed in the first column. The decision error is computed when orienting the cells of the

new technology (right side of the arrow), considering cells databases filled with cells from the technologies mentioned on the left side of the arrow. For each row in Table II, the process of Fig. 2 has been repeated at least five times to form the dataset. The expected 'yes/no' answer for (cell; defect type) combination sample depends on the quality threshold introduced in the previous section. Accepting a lower prediction accuracy of the ML-based CA model generation algorithm means that the predicted CA models can widely differ from the reference simulated CA model. We intuitively consider that the lower the threshold, the more (cell; defect type) combinations will use the ML-based CA model generation part of the flow. The decision block should be efficient at orienting the cells through the hybrid flow irrespective of the chosen quality criterion. Table II contains decision errors for several quality thresholds (respectively 85%/93%/97%) for each experiment.

TABLE II. DECISION ERROR (%) OF THE ML-DECISION BLOCKS, FOR SEVERAL EXPERIMENTS AND SEVERAL QUALITY THRESHOLD (85%/93%/97%)

Experiment	ML-decision	Structural analysis
C40→ C28	5/3/2	11/18/22
C40 + C28→ P28	3/6/5	8/17/29
C40 + C28 + P28→ P18	4/6/4	4/11/22
C40 + C28 + P28 + P18→ FF16	3/3/3	12/22/26

The quality of the new ML-based decision block is compared to the old one based on structural analysis. To this end, the **decision error** of the old decision block is computed. The meaning of the decision error remains the same, i.e., how often does the decision block orient a (cell; defect type) combination through the ML-based CA model generation even though the content of the cells database is not yet sufficient to reliably predict the behavior of the defects in the cell. Note that the decision of the old decision block is based only on the content of the cells database and its decision for a given (cell; defect type) combination does not change with the quality threshold. However, the expected decision for the sample changes with the threshold, making ultimately the decision error of the old decision block to vary with the quality threshold.

The last column of Table II reports the decision errors of the old decision block based on structural comparison. Table II clearly shows the supremacy of our new decision block over the structural one. The new ML-decision block will more often orient a (cell; defect type) combination through the appropriate CA model generation part of the flow, resulting in a better usage of the quick ML-based CA model generation part of the flow, and allowing to quickly get CA models for compatible cells and guarantying CA models of high quality for cells that would be badly predicted considering the current content of the cells database.

While studying the low number of (cell; defect type) combinations which have been badly oriented by the ML-decision block, we noticed that these combinations mostly have a prediction accuracy close to the threshold, making the 'yes/no' decision difficult and explaining the misprediction of the developed ML-decision algorithm. Providing more information about the cells and about the content of the cells database as inputs to the ML-decision algorithm may help to differentiate these cases, thus further decreasing the decision error and increase the global quality of the ML-based decision block.

The mis-decision of sending a (cell; defect type) combination through the ML-based CA model generation part of the flow is susceptible to lead to a bad prediction of the behavior of these defects in this cell. This is an improvement compared to the previous work presented in [1] as a mis-prediction by the structural decision block would have sent the entire cell through the inappropriate CA model generation part of the flow. The new fine-graining and the marginal number of mis-predictions lead the behavior of other defect types in the same cell to be accurately predicted or simulated. **The CA model of the entire cell will only contain a few percentages of error.** For example, on a population of nearly 2000 cells, sending all cells through the ML-based CA-model generation part of the flow would generate CA models that are on average 91.3% identical to the reference (i.e., simulations only CA models). This average is 92.4% when the old structural hybrid flow ensure quality by orienting some cells through the simulation-based CA model part of the flow. Finally, when fine graining and the new ML-based decision block are used, the average resemblance of the hybrid generated CA models is **99.1%**. These few percentages of error **can be tolerated by the industry.** The next sub-section will demonstrate this point.

B. Industrial validation : ATPG with ML-based CA models

The ultimate goal of this work is to generate CA models for std cells from the hybrid flow, and use them in an exigent industrial environment. This sub-section demonstrates the appropriateness of our method to obtain CA models of quality that can replace the ones obtained by SPICE simulations only.

1) Methodology

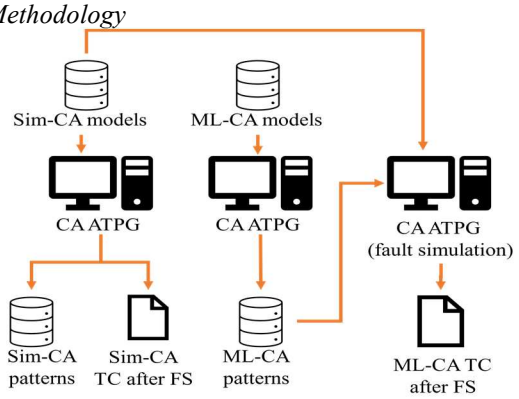

Fig. 3: ATPG flow to assess quality of the CA models obtained by the hybrid flow

The proposed method is sketched in Fig. 3 and uses our CA models with an ATPG to generate test patterns targeting cell-internal defects for several industrial designs. The fault coverage of these patterns is computed and compared to the fault coverage of patterns obtained while using commercial CA models (i.e., those obtained using SPICE simulation). Both sets of test patterns should allow the detection of a high fraction of the real cell-internal defects (ideally 100%). The real cell-internal defects are represented by the fault models contained in the CA models generated by SPICE simulations only. The important point of comparison between the two sets of patterns are the number of patterns and above all the fault coverage of real cell-internal defects. Close values between the two sets are the proof

that the CA models generated by the proposed updated hybrid flow can efficiently replace the ones obtained by simulations only.

2) Experiments

The proposed methodology has been applied on 15 designs from STMicroelectronics. These designs are blocks of a testchip designed using the P28 technology. The blocks have various complexities and are meant to be independently tested by different sets of ATPG patterns. The CA models for the cells used in the designs have been obtained by the updated hybrid flow. To mimic a realistic industrial usage, the CA models for these P28 cells have been obtained with a cells database filled with C40 and C28 cells. The quality criterion has been set to 93%. In our working group of 1932 cells, **79% of (cell; defect types) combinations have had their behavior predicted by the ML-based part of the flow.**

Table III presents the number of patterns in each set as well as their corresponding fault coverage when targeting the real **static** cell-internal defects (due to lack of space, results are reported for 6 designs out of 15, missing lines are similar). Results while targeting real **dynamic** cell-internal defects are similar. The fault coverage is computed by the ATPG as the fraction of detected faults over the entire fault population. The average difference of fault coverage between the two sets of patterns is **0.002 percentage point.** The slight difference in the number of patterns (6% on average) results from the slightly different number of faults marked as detected in the two types of CA models.

TABLE III. NUMBER OF PATTERNS AND ASSOCIATED FAULTS COVERAGES WHEN TARGETING REAL STATIC CELL-INTERNAL DEFECTS IN SEVERAL DESIGNS.

	#cells	With hybrid-flow		With simulations only	
		#patterns	%coverage	#patterns	%coverage
0	44k	2922	97.21%	2837	97.53%
4	44k	2706	97.33%	2596	97.47%
5	4.1k	17	100%	17	100%
9	4.1k	25	100%	26	100%
10	11k	1738	91.48%	1451	91.70%
14	10k	1794	62.18%	1686	62.26%

Those values clearly shows that the CA models obtained by the hybrid flow can be used to target real cell-internal defects with the same coverage that CA models obtained by simulations only. These results demonstrate the pertinence and effectiveness of our approach for CA models generation.

IV. CONCLUSION

CA model generation refers to the process of characterizing cell-internal defects, a key step to ensure high quality for test and diagnosis of digital ICs. In [1], we presented an innovative flow using the power of ML to generate CA models without relying on time consuming analog simulations and costly simulator licenses. To further ease easing the deployment of CA test and diagnosis methodologies, this paper presents a new decision algorithm, which orients std cells, on a defect type basis, towards the quick ML-based CA model generation or the traditional CA characterization flow using analog simulations. This new decision algorithm is an RFC ML algorithm and a takes its decisions by analyzing the cells to be oriented as well as the content of the std cells database. The cells database contains the existing CA models of previously characterized std cells and is used for the supervised training of the ML-algorithm for CA model generation. The new decision algorithm introduced fine-graining, orienting (cell; defect types) combinations through the hybrid flow rather than entire cells. Fine-graining increases the quality of the generated CA models. Experiments mimicking the industrial usage of such a flow were performed using cells belonging to five technology nodes. The results show that the new decision algorithm is efficient at orienting cells across the two parts of the hybrid-flow.

For the first time, we used the CA models obtained thanks to the hybrid flow with an ATPG tool on 15 industrial designs. Hybrid flow lead to 79% of the (cell; defect type) combinations to get their behavior predicted by the ML-based CA model part of the hybrid flow. The fault coverage of these test patterns on real cell-internal defects was computed. The fault coverage has been compared and found equivalent to the faut coverage of patterns using CA models obtained by simulations only. This proves that our predicted CA models can accurately replace those obtained by running extensive analog simulations, thus proving the effectiveness and pertinence of the proposed methodology.

REFERENCES

[1] P. d'Hondt, A. Ladhar, P. Girard, and A. Virazel, "A Learning-Based Methodology for Accelerating Cell-Aware Model Generation," in *Proc. IEEE/ACM Design Automation and Test in Europe*, 2021.

[2] A. Ladhar, M. Masmoudi, and L. Bouzaida, "Efficient and Accurate Method for Intra-Gate Defect Diagnoses in Nanometer Technology," in *Proc. IEEE/ACM Design Automation and Test in Europe*, 2009.

[3] Z. Sun, A. Bosio, L. Dilillo, P. Girard, A. Virazel, and E. Auvray, "Effect-Cause Intra-cell Diagnosis at Transistor Level," in *Proc. IEEE International Symp. on Quality Electronic Design*, 2013.

[4] Z. Gao, M-C Hu, J. Swenton, S. Magali, J. Huisken, K. Goosens, and E.J. Marinissen, "Optimization of Cell-Aware ATPG Results by Manipulating Library Cells' Defect Detection Matrices," in *Proc. IEEE International Test Conference in Asia (ITC-Asia)*, 2019.

[5] F. Hapke, et al., "Cell-Aware Test," *IEEE Transactions on Computer-Aided Design*, vol. 33, no. 9, pp. 1396 - 1409, 2014.

[6] P. Maxwell, F. Hapke, and H. Tang, "Cell-Aware Diagnosis: Defective Inmates Exposed in their Cells," in *IEEE European Test Symp.*, 2016.

[7] F. Hapke, R. Krenz-Baath, A. Glowatz, J. Schloeffel, P. Weseloh, M. Wittke, M. Kassab, and C. W Schuermyer, "Cell-Aware Fault Model Creation and Pattern Generation," US Patent 12/718,799, 2010.

[8] S. Mhamdi, P. Girard, A. Virazel, A. Bosio and A. Ladhar, "A Learning-Based Cell-Aware Diagnosis Flow for Industrial Customer Returns," in *Proc. IEEE International Test Conf.*, 2020.

[9] F. Lorenzelli, Z. Gao, J. Swenton, S. Magali, and E.J. Marinissen, "Speeding up Cell-Aware Library Characterization by Preceding Simulation with Structural Analysis," in *Proc. IEEE European Test Symp.*, 2021.

[10] Z. Gao, S. Malagi, M. Chun Hu, J. Swenton, R. Baert, J. Huisken, B. Chehab, K. Goossens, and E.J. Marinissen, "Application of Cell-Aware Test on an Advanced 3nm CMOS Technology Library," in *Proc. IEEE International Test Conf.*, 2019.

[11] R. Guo, B. Archer, K. Chau, and X. Cai, "Efficient Cell-Aware Defect Characterization for Multi-bit Cells", in Proc. IEEE International Test Conf. in Asia, 2018