Faulty Function Extraction for Defective Circuits

Chris Nigh*, Ruben Purdy*, Wei Li*, Subhasish Mitra†, R.D. Blanton*
*Electrical and Computer Engineering Department
Carnegie Mellon University, Pittsburgh, Pennsylvania
†Department of Electrical Engineering and Department of Computer Science
Stanford University, Stanford, California

2024 IEEE European Test Symposium (ETS) | 979-8-3503-4932-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/ETS61313.2024.10567760

***Abstract*—It is well-known that understanding the behavior of silicon failures is an essential step in yield learning. It is also becoming more important for producing high-quality silicon due to the increasing number of defects detected fortuitously. In order to meet this need, a new approach for extracting the precise faulty function from defective logic circuits is described. The approach is applied to nearly a 1,000 14nm failures and one use case of the results on improving ATPG is discussed.**

I. Introduction

Silent data corruption (SDC) has and continues to receive a great deal of attention from the research community [1]–[3] and even the popular press [4]. In particular, the work in [1], [2] clearly demonstrates that SDCs are the result of hardware defects in the logic circuits that escaped manufacturing test. Obviously, an escaped defect is due to its behavior deviating from what is predicted by the models and metrics utilized for test generation. For example, in Section IV, it is shown that chip fails detected using tests based on the stuck-at and cell-aware fault models deviate from stuck-at and cell-aware behavior 99.4% and 95.7% of the time, respectively. This means, of course, that a significant proportion of defective ICs are detected fortuitously.

Fortuitous detection is nothing new; the test community has long understood that most defects are fortuitously detected. For example, N-detect [5] and other test metrics [6], [7] do not attempt to model defects but instead attempt to increase the likelihood of fortuitous detection. The amount of fortuitous detection has changed dramatically over the years however. Table I reports the percentage of defects detected by stuck-at tests that exhibit non-stuck-at behavior extracted from several papers published between 1990 and 2022. This all implies that the semiconductor industry is relying more and more on fortuitous detection to establish the ever-increasing chip quality levels demanded by a world increasingly dominated by electronic systems.

Node (nm)	**Non-stuck-at (%)**	**Data source**
15000	20.2	1990, Bell-Northern Research, [10]
130	63.1	2001, IBM, [11]
90	80.0	2008, NXP, [12]
14	99.4	2022, GLOBALFOUNDRIES, [8]

TABLE I: Percentage of defects that exhibit stuck-at behavior.

It may be the case that our reliance on fortuitous detection has run its course and that is now time to examine the phenomenon in a precise, data-driven manner. That is, we believe it is imperative to understand precisely the misbehavior caused by defects in actual failing ICs in order to measure quantitatively the deviation they exhibit with respect to the fault models and test metrics deployed for test generation. Why? A significant gap between models/metrics and actual defect behavior intuitively increases the likelihood of defect escape and (maybe) the SDC phenomenon experienced by super-silicon system owners such as Google, Meta and others.

At first thought, EDA-based diagnosis comes to mind as an ideal approach for precisely understanding logic circuit failures. Diagnosis is insufficient however; while diagnosis is extremely adequate for localization it generally has not been developed to precisely derive the misbehavior caused by defects. We therefore propose a methodology based on the Pseudo-Exhaustive Physically-Aware Region (PEPR) testing metric [8]. PEPR is an ideal choice because it is a parameterizable approach that assumes that the influence of a defect can be bounded by a region in the design that has both physical (i.e., layout) and logical aspects. We develop an approach to identify the region that bounds a defect, then characterize the change in behavior of the subcircuit contained in the region caused by the defect. Here, behavior consists of not only function but timing and state as well. In this work, we focus solely on circuit function however.

There is one other work we are aware of whose goal is the precise characterization of faulty functionality due to the presence of a defect. Specifically, in [9], a function describing the impact of a defect on circuit behavior is derived based on faulty nets and the signals that influence those nets. The most significant difference between that work and what is proposed here is that [9] assumes that only a single net can be erroneous during any given test. For PEPR, any number of nets within the parameterized region are allowed to be simultaneously erroneous.

In the remaining sections of this paper, we describe our methodology for deriving faulty functionality, and its application to fail data from a large population of failing ICs. Specifically, in Section II, what is meant by faulty function extraction is described followed by details on the approach developed for identifying/deriving the precise changes in circuit function due to a defect in Section III. Also in Section III, we describe the iterative approach for identifying the minimal region that bounds a defect in a failing logic circuit. Application of the

approach is described in Section IV, where we specifically apply our methodology to nearly 1,000 failures from test chips fabricated in a 14nm technology. Furthermore, we demonstrate how the extracted faulty functions are used to inform a data-driven approach for PEPR-based ATPG[1]. Finally, in Section V, we summarize our contributions and discuss on-going and future work.

II. Faulty Function Definition

As alluded to earlier, the objective of diagnosis is localization and defect attribution. That is, diagnosis answers the questions, what is the x-y-z location of the failure and what is its nature, *i.e.*, open, bridge, cell-internal, *etc.*? The goal of diagnosis however is not precise faulty function extraction, evidenced by the fact that there are typically scores provided with the diagnostic outcomes that characterize the deviation from what was observed on the tester and predicted by model(s) employed by the diagnosis algorithm.

Our goal instead is to identify the precise change in circuit function due to a defect. One way of course is to check if defective circuit function matches a defect model (*e.g.*, stuck-at, cell-aware, *etc.*). This is accomplished simply by fault simulating known fault models using the circuit test set, and comparing the bit-level errors to those observed on the tester which is, in essence, what almost all diagnosis algorithms perform. As we demonstrate later however, very few failing circuits behave as predicted by conventional fault models. As a result and as already mentioned, we utilize the PEPR test metric because it assumes the behavior of a region's subcircuit can arbitrarily change (due to a defect) and the size of the region is parameterizable.

A. Pseudo-Exhaustive Physically-Aware Region Testing

The Pseudo-Exhaustive Physically-Aware Region (PEPR) testing metric [8] rasterizes a design into overlapping regions each of which consist of a physical (layout) component, and additional but optional logical levels. PEPR requires that the subcircuit in each region be exhaustively tested to ensure that it is functioning correctly. Associated with the physical portion of a region are three parameters x, y, and L which represent the width, length and the number of layers that defines the dimensions of the cuboid derived from the design layout. There are also three other parameters s_x, s_y and s_L that represent the step-size for layout rasterization along the three dimensions. Parameters s_x and s_y define the step size in units of measure (*e.g.*, nanometers) while s_L defines step size in the z direction in terms of the number of layers. Rasterization begins in one corner of the design layout and steps in the x, y, and z directions, one dimension at a time, until the entire design is covered.

The subcircuit corresponding to a layout cuboid can be augmented with m levels of logic added to the inputs of the circuit and/or n levels of logic added to the outputs of the circuit.

[1]Faulty function extraction is not necessary for PEPR; the point is that region parameters observed from faulty function extraction can be used to help select region parameter values for future applications of PEPR.

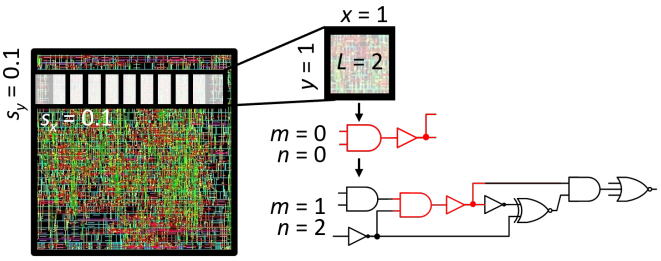


Fig. 1: PEPR region and circuit derivation.

		F_1	F_2	F_3	...	F_{255}
00	00	01	10	11	:	11
01	00	00	00	00	:	11
10	00	00	00	00	:	11
11	11	11	11	11	:	00

Fig. 2: The faulty functions for a region circuit with $p = q = 2$.

Adding logic to the circuit inputs is meaningful in cases where defect activation may depend on the precise voltages driving the circuit inputs which can slightly vary depending on the number and characteristics of the active cell transistors in the added logic. Logic added to the circuit outputs is justified for situations where a defect creates intermediate voltages near the transistor threshold values of driven cells. Intermediate voltages may have to pass through several levels of logic before they have resolved to a logic zero or one where they can then be reliably propagated. Figure 1 illustrates layout region extraction, circuit identification, and circuit augmentation with additional logic.

Assuming a region and its corresponding circuit bounds the impact of a defect within the logic circuit under test, our objective is to identify the faulty function of the region circuit. For a circuit with p inputs and q outputs, there are $2^{(2^p \times q)}$ possible functions, one of which is the intended function while the remaining are the possible faulty functions that PEPR assumes a defect can cause. Figure 2 illustrates these concepts for an example region circuit.

B. The Input Pattern Fault Model

To identify the faulty function of a region circuit, the circuit input pattern (IP) faults [13] are fault simulated for comparison with tester data. An IP fault $ip \rightarrow o/o'$, where ip is one of the possible 2^p set of input values, o is the correct output circuit response corresponding to ip, and $o' \neq o$ is one of the possible $2^q - 1$ set of erroneous output values produced when ip is applied to a faulty circuit. By fault simulating all possible $2^p \times (2^q - 1)$ IP faults and comparing their simulation responses to the tester data, the faulty function, if it exists, can be derived.

If none of the $2^{(2^p \times q)} - 1$ possible faulty functions have a fault-simulation response that exactly matches what has been observed on the tester, then no faulty function exists for the circuit region. There are several reasons why this may happen, specifically:

- The region does not bound the defect. That is, the defect affects more of the logic circuit under test than the subcircuit corresponding to the region. In Section III, we describe the approach deployed to identify the region that bounds the defect which in theory must exist, if the defect obeys the assumption that it only affects functionality (*i.e.*, no impact on timing, no additional state, *etc.*).
- Because it is not possible to know whether a defective logic circuit satisfies the aforementioned assumptions, it is therefore possible that a faulty function simply cannot be derived because it does not exist.
- The user-defined fault model (UDFM) capability [14] is used to perform the IP fault analysis described above. The UDFM capability however allows only one net to be erroneous which means that the number of possible erroneous responses for a given circuit input pattern ip is q instead of $(2^q - 1)$. Thus, any faulty function that causes two or more circuit outputs to be erroneous cannot be modeled using a set of user-defined faults. Later in Section III-C an approach to somewhat overcome the limited capability of UDFMs is described.

III. Faulty Function Extraction

Faulty function extraction for a failing logic circuit requires (i) identification of a specific circuit region, and (ii) derivation of a logical description of the changed functionality of the subcircuit contained in the region. Given the large number of possible faulty functions, it quickly becomes apparent that an efficient analysis approach is required for faulty function extraction. This section describes the use of diagnosis for initial defect localization, the deployment of fault models and test metrics for faulty function extraction, a dictionary-based implementation that reduces fault simulation, and a parameter search for region identification.

A. Defect Localization

The first step in identifying a faulty function is to find the potential circuit regions that bound the defect. It is assumed that there is only a single defect that affects the logic circuit under test. However, a situation can be imagined where a single identified region bounds multiple defects, or multiple identified regions collectively bound all existing defects. Diagnosis is a proven method for localizing a defect with high confidence [15]. In this work, layout-aware diagnosis is used to locate potential defect site(s) in the design layout, offering an excellent starting point for bounding a defect. Also, limiting analyses of failures labeled by diagnosis as a single defect with an associated single candidate ensures correctness of faulty function extraction, and reduces computational effort. While commercial diagnosis offerings are capable of handling multiple defects, there is significant complexity and a reduced confidence in the reported "callout" especially when defect sites interact. In light of the aforementioned, it is possible to derive a faulty function for each defect and combine the functions similarly to the approached described in [9].

B. Fault Models and Test Metrics

Successful faulty function extraction for a failing logic circuit results in a logical model of the functionality change for a circuit region that, when simulated, produces a response that exactly matches the tester response. While this is proposed as a posterior analysis of existing logic circuit fails, similar logical models are regularly postulated *a priori* in the form of fault models, like stuck-at, cell-aware, or bridge. For example, a given stuck-at fault identifies a signal line as a region, and uses a constant-1 or -0 faulty function for the region; a given cell-aware fault identifies a cell instance as a region, and uses a circuit model of the cell with some internal defect like a bridge or a transistor open to obtain a faulty function; a given bridge fault model typically identifies two signal lines as a circuit region, and assumes some faulty functionality when the two signals have opposite logic values.

Fault models are intended to represent a specific defect with a corresponding faulty function. If the impact of a defect is accurately captured by a fault model, then the errors (*i.e.*, incorrect primary and scan outputs) observed via the tester exactly match the errors predicted by fault simulation. If there is any mismatch, the faulty function assumed/defined by the fault model is obviously not the same as that caused by the defect which means detection is fortuitous which inherently implies that other defects with similar behavior outside of the fault model may not be detected and thus escape.

In contrast to fault models, test metrics, such as gate-exhaustive [16], N-detect [5], physically-aware N-detect [17], and PEPR [8], are developed to measure test quality [15]. Test metrics typically assume a wide range of faulty functionality, making them more comprehensive (and also more costly) than a fault model that may target the same circuit regions. For example, cell-aware assumes faulty functions that correspond to certain circuit perturbations of the cell while gate exhaustive assumes any faulty function is equally possible. Test metrics typically prioritize identification of the circuit region that contains the defect rather than a specific faulty function which implies the region should be tested exhaustively (gate-exhaustive, PEPR) or multiple times (N-detect, physically-aware N-detect). Because there is not a single faulty function to compare against, test metrics must be evaluated differently than fault models.

As an example, consider the physically-aware N-detect test metric [17], used to measure the proportion of stuck-at faults that have been detected N times under unique physical neighborhood states. This metric does not assume a particular faulty function, but rather claims that the activation of a defect that affects a single net may be dependent on the logic values of nets within close proximity of the defective net, and thus it is beneficial to detect a stuck-at fault for varying neighborhood states. Similarly, gate-exhaustive and PEPR identify a logical

and physical circuit region, respectively, that should be exhaustively tested. Psuedo-exhaustive test metrics assume that any input pattern applied to the circuit region may produce a faulty response. Without an assumed faulty function, a simple comparison of errors observed in simulation and errors observed on the tester is not applicable.

To allow for the uncertainty associated with test metrics, the concept of *consistency* introduced in [9] is utilized here. In essence, a test metric is consistent if the the circuit region is capable of producing a faulty function that exactly matches the errors observed on the tester. If a test metric is inconsistent, then there is some assumption made by the metric that does not hold. For example, as discussed in Section II, PEPR can be inconsistent if the region does not, for instance, bound the defect.

Consider the concept of consistency applied to IP faults. Consistency is defined with respect to what is observed on the tester. Specifically, an IP fault $ip \rightarrow o/o'$ is consistent if one of the following is true:

- For each t_i that detects $ip \rightarrow o/o'$, the corresponding fault simulation response must exactly match the errors observed on the tester, OR
- For each t_i that detects $ip \rightarrow o/o'$, there are no errors observed on the tester.

Circuit regions that have a consistent set of IP faults has a corresponding faulty function; regions that do not, do not bound the defect.

C. Dictionary Analysis

Input pattern faults associated with region subcircuits can be fault simulated directly using UDFMs if the number of faulty outputs is limited to just a single output at a time as described in Section II. However, performing fault simulation to obtain the precise response produced at the primary and secondary (scan) outputs is significantly more computationally expensive than pass-fail simulation. For example, in our experiments, pass-fail fault simulation is 3500X faster than obtaining the detailed fault simulation response. This slowdown however is easily mitigated through parallelism.

In order to improve efficiency, a dictionary-based IP fault analysis is developed. An IP fault $(v_0 v_1 \dots v_{p-1}) \rightarrow (o_j{=}v)/(o_j{=}v')$ for a region subcircuit with p inputs and q outputs that faults a single output o_j is detected by a test if (i) the values $(v_0 v_1 \dots v_{p-1})$ are applied to the p inputs, and (ii) the output o_j is sensitized to some observable point (*i.e.*, a primary or secondary output). Two dictionaries are created to evaluate these two detection requirements. The first is a signal dictionary that contains the logic values for the inputs of all region circuits of interest for the test set of the logic circuit under test. The second is a stuck-at fault dictionary that contains a single bit indicating the detection status of stuck-at faults located at region circuit outputs.

The two dictionaries are constructed for signals and faults associated with region circuits believed to bound defects for a population of failing logic circuits. Querying both dictionaries for a region circuit's IP faults enables the identification of a set of consistent IP faults (if they exist) and its corresponding faulty function.

In general, current commercial offerings of fault simulation do not allow multiple signals to be faulty simultaneously. This limitation, as described in Section II, prevents the analysis of all possible faulty functions. One approach to mitigate this limitation is to union the fault simulation responses of individual faulty signals to approximate errors stemming from IP faults that have more than one faulty output. This approach is not guaranteed to overcome the shortcoming however, if errors from multiple faulty outputs interact constructively or destructively. If the capability for multiple faulty signal simulation existed, faulty function extraction success would, in theory, improve.

D. Minimal Region Identification

Extracting a faulty function caused by defect within a logic circuit under test first requires the identification of a region that bounds the defect. The layout bounding boxes reported by layout-aware diagnosis are used as starting points. That is, regions with the parameters described in Section II are overlaid on these locations to ensure they are covered in a manner consistent with the deployment of PEPR for ATPG purposes. The region layout (x, y and L) and logic (m and n) parameters are systematically searched to identify minimal values that lead to a faulty function that produces a fault simulation response that exactly matches errors observed on the tester, if possible.

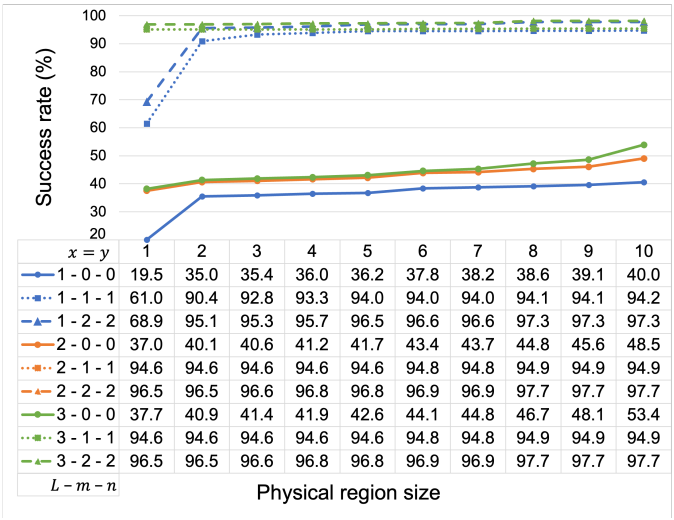
IV. Experiments

Faulty function extraction is performed on fail data from a 18.7M gate test chip fabricated in a 14nm technology. The design has a total of 136.6M stuck-at faults and a 3-detect test set that achieves 98.3% coverage. We have fail data for over 30,000 fail logs but focus, as described in Section III-A, only on the 989 single-defect, single-candidate fail logs for one of the twelve cores in the chip. Particularly, we determine how many of these failures have a faulty function precisely predicted by a given fault model or test metric. In addition, we analyze the correlation that exists between (i) PEPR region parameters that successfully lead to faulty function extraction, and (ii) properties of the design and wafer location.

A. Models and Metrics Evaluation

The stuck-at and cell-aware fault models are evaluated as described in Section III-B. That is, stuck-at and cell-aware faults corresponding to locations reported by diagnosis are simulated, and the corresponding response compared to the data observed on the tester. An exact match between the simulation response and the tester data indicates the model has correctly predicted the exact faulty function. For the 989 fail logs examined, 99.4% and 95.7% have faulty functions that deviate from stuck-at and cell-aware behavior, respectively.

The IP faults for gates and PEPR regions with $x{=}y{=}5$, $L{=}3$ and $m{=}n{=}0$ corresponding to locations reported by diagnosis are analyzed for consistency. The percentage of the 989 fail logs that have faulty functions successfully extracted is 5.5%



$x = y$	1	2	3	4	5	6	7	8	9	10
1 - 0 - 0	19.5	35.0	35.4	36.0	36.2	37.8	38.2	38.6	39.1	40.0
1 - 1 - 1	61.0	90.4	92.8	93.3	94.0	94.0	94.0	94.1	94.1	94.2
1 - 2 - 2	68.9	95.1	95.3	95.7	96.5	96.6	96.6	97.3	97.3	97.3
2 - 0 - 0	37.0	40.1	40.6	41.2	41.7	43.4	43.7	44.8	45.6	48.5
2 - 1 - 1	94.6	94.6	94.6	94.6	94.6	94.8	94.8	94.9	94.9	94.9
2 - 2 - 2	96.5	96.5	96.6	96.8	96.8	96.9	96.9	97.7	97.7	97.7
3 - 0 - 0	37.7	40.9	41.4	41.9	42.6	44.1	44.8	46.7	48.1	53.4
3 - 1 - 1	94.6	94.6	94.6	94.6	94.6	94.8	94.8	94.9	94.9	94.9
3 - 2 - 2	96.5	96.5	96.6	96.8	96.8	96.9	96.9	97.7	97.7	97.7
$L - m - n$	Physical region size									

Fig. 3: Faulty function extraction success as a function of PEPR parameters x, y, L, m and n.

and 43.3% for the gate-exhaustive and PEPR test metrics, respectively. PEPR, with these parameter settings, is able to successfully explain 10X more failures than the cell-aware fault model.

B. PEPR Parameter Exploration

The PEPR parameters used in Section IV-A are arbitrarily chosen. This means that some of the 43.3% of faulty functions extracted, while accurate, may be for regions that are unnecessarily large with respect to bounding the defect. For the remaining fail logs, there could be other region parameters for which a faulty function does indeed exist. The unique parametrization properties of the PEPR test metric enables search for a region that leads to successful faulty function extraction. In this context, faulty function extraction is attempted for regions where the parameters are swept from 1X to 10X the minimum layout feature size for x and y, 1 to 3 for the number of layers L, and 0 to 2 for m and n, the additional levels of logic added to the inputs and outputs of the corresponding region circuit, respectively. The outcomes of this analysis is presented in Figure 3.

A key takeaway from Figure 3 is that adding even one logic level to the inputs and outputs (*i.e.*, $m{=}n{=}1$) to a region circuit significantly increases the success of faulty function extraction. Also, while the largest parameters $\{x{=}y{=}10;\ L{=}3;\ m{=}n{=}2\}$ leads to successful faulty function extraction for 97.7% of failures, much smaller regions ($\{x{=}y{=}2;\ L{=}2;\ m{=}n{=}1\}$) are very successful at 94.6%. Finally, we are further investigating the 23 failures that did not have a faulty function successfully extracted. Specifically, questions being considered include: Will a larger region lead to success? Are these failures sequence- or timing-dependent? Is diagnosis incorrect leading to regions that have zero chance of bounding the defect?

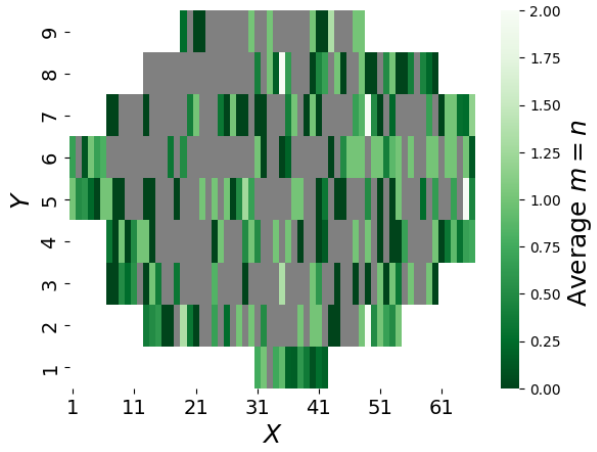


Fig. 4: Wafer map detailing average added logic levels $m = n$ required for faulty function identification. (Locations absent of failure data are shaded grey.)

C. Region to Layout-Process Correlation

Additionally, an analysis is performed to identify how regions that lead to the successful extraction of a faulty function correlate with process and design characteristics. For the correlation analysis, the physical region width and length are set to $x{=}y{=}10$, the number of region layers is set to $L{=}3$, and the number of added levels of logic is varied from $m{=}n{=}0$ to $m{=}n{=}2$. For the process-based analysis, m and n are correlated with wafer lots and $X{-}Y$ die coordinates.

From the available data, there are five lots that have more than 100 failures, details are given in Table II.[2] The five lots are tested at different points in time as denoted by the *Week* column. The average number of required additional logic levels trends upward, suggesting that the number of defects that are voltage dependent may be increasing over time.

	Week	No. of failures	Avg. $m{=}n$
Lot A	A	196	0.41
Lot B	A+1	135	0.45
Lot C	A+4	162	0.44
Lot D	A+5	198	0.49
Lot E	A+20	199	0.50

TABLE II: Lot-based details for average logic levels added for successful faulty function extraction.

The wafer map in Figure 4 shows the spatial behavior for the average number of additional levels of logic required to successfully extract the faulty function. One observation is that regions with defects that are less reliant on added logic levels can be identified, for example, the upper-right and bottom edge.

To understand the relationship between the faulty function identification and design features, a similar analysis is performed based on the diagnosis-reported layout layer, with

[2]The failure data available for analysis is not necessarily all the failure data from these lots.

results reported in Figure 5. From this analysis, we observe that the cell layout layer generally requires more added logic levels than the metal layers, with an average value closer to one. This and other design-specific information information can be used to fine-tune ATPG, redistributing test effort from areas that have smaller regions to others with larger regions. From these results, adding one logic level to region circuits extracted from the cell layout layer is justified, while no additional logic is justified for the metal layers.

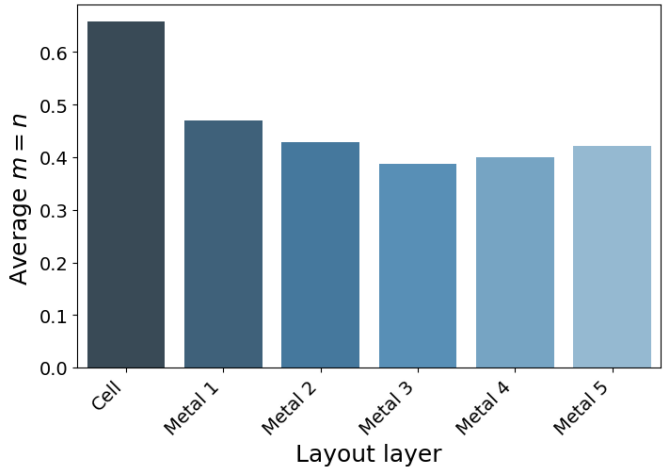


Fig. 5: Comparison of average additional logic levels as a function of layout layer reported by diagnosis.

V. Conclusions

A method for extracting the faulty function for a defective logic circuit has been described and demonstrated. The approach uses the PEPR test metric to bound the impact of the defect both physically (*i.e.*, at the layout level) and logically, and a dictionary-based IP fault analysis to determine changes in functionality due to the defect. Experiments involving actual failure data collected from a 14nm test chip reveal that there is a huge gap between what is predicted by conventional fault models and what is observed on the tester. Importantly, however, extraction of the exact faulty function is successful nearly 98% of the time for the 989 fail logs analyzed.

Our current work is investigating several applications for extracted faulty functions. Some examples include:

- **PEPR ATPG:** A powerful feature of PEPR is the capability to change the parameters of the regions and the corresponding subcircuits targeted for pseudo-exhaustive test. Faulty function extraction for a statistically-significant level of failures could be used to inform what parameter values are required to ensure future defective chips are detected deterministically and not fortuitously. This, of course, assumes faulty function extraction is performed on defects that are representative of those that will occur in the future.
- **Defect Characterization:** Faulty function extraction provides both the circuit and change in circuit functionality caused by a defect. Detailed defects can therefore be postulated and simulated (at the transistor level or lower) to determine if it leads to the extracted faulty function.
- **Faulty Function Prediction:** Faulty function extraction applied to a large population of failures produces significant data pairs, each of which consists of a circuit and its associated faulty function. It may be possible to learn a model that takes in as input circuit characteristics and outputs the faulty function. Such a model implies that PEPR ATPG could be relaxed from pseudo-exhaustive to faulty function specific.

VI. Acknowledgements

We would like to thank John Carulli and Rohan Deshpande of Globalfoundries for detailed discussions regarding the design, test and diagnosis of their $14\,\mathrm{nm}$ test chip.

References

[1] H. D. Dixit *et al.*, "Silent Data Corruptions at Scale," *CoRR*, 2021.
[2] ——, "Detecting Silent Data Corruptions in the Wild," 2022.
[3] P. H. Hochschild *et al.*, "Cores That Don't Count," in *Workshop on Hot Topics in Operating Systems*. Association for Computing Machinery, 2021.
[4] J. Markoff, "Tiny Chips, Big Headaches," *The New York Times*, Feb 2022.
[5] S. Ma, P. Franco, and E. McCluskey, "An Experimental Chip to Evaluate Test Techniques Experiment Results," in *IEEE International Test Conference*, 1995, pp. 663–672.
[6] J. Dworak *et al.*, "Enhanced DO-RE-ME Based Defect Level Prediction Using Defect Site Aggregation - MPG-D," in *IEEE International Test Conference*, 2000, pp. 930–939.
[7] Y.-T. Lin *et al.*, "Physically-Aware N-Detect Test Pattern Selection," in *Design, Automation and Test in Europe*, 2008, pp. 634–639.
[8] W. Li *et al.*, "PEPR: Pseudo-Exhaustive Physically-Aware Region Testing," in *IEEE International Test Conference*, 2022, pp. 314–323.
[9] R. Desineni and R. Blanton, "Diagnosis of Arbitrary Defects using Neighborhood Function Extraction," in *IEEE VLSI Test Symposium*, 2005, pp. 366–373.
[10] A. Pancholy, J. Rajski, and L. McNaughton, "Empirical Failure Analysis and Validation of Fault Models in CMOS VLSI," in *IEEE International Test Conference*, 1990, pp. 938–947.
[11] T. Bartenstein *et al.*, "Diagnosing Combinational Logic Designs using the Single Location At-a-Time (SLAT) Paradigm," in *IEEE International Test Conference*, 2001, pp. 287–296.
[12] S. Eichenberger *et al.*, "Towards a World Without Test Escapes: The Use of Volume Diagnosis to Improve Test Quality," in *IEEE International Test Conference*, 2008, pp. 1–10.
[13] R. D. Blanton and J. P. Hayes, "Properties of the Input Pattern Fault Model," *International Conference on Computer Design*, pp. 372–380, 1997.
[14] *Tessent Scan and ATPG User's Manual*, Siemens EDA Software, 2023, Software Version 2023.2.
[15] Y.-T. Lin and R. D. Blanton, "METER: Measuring Test Effectiveness Regionally," *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 7, pp. 1058–1071, 2011.
[16] K. Y. Cho, S. Mitra, and E. J. McCluskey, "Gate Exhaustive Testing," in *IEEE International Test Conference*, Nov 2005, pp. 1–7.
[17] J. Nelson *et al.*, "Multiple-Detect ATPG based on Physical Neighborhoods," in *ACM/IEEE Design Automation Conference*, 2006, pp. 1099–1102.