PEPR: Pseudo-Exhaustive Physically-Aware Region Testing

Wei Li*, Chris Nigh*, Danielle Duvalsaint*, 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

Abstract—**Recent reports indicate that existing fault models and test metrics result in substantial manufacturing test escapes that cause major system-level challenges such as silent data corruption resulting from incorrect computations. Such test escapes are often detected today after system deployment (*e.g.*, in the field) using a variety of synthetic and application workloads. In this work, a new test metric is investigated for detecting defects that escape existing test approaches. PEPR (Pseudo-Exhaustive Physically-Aware Region) testing comprehensively analyzes both the physical layout and the logic netlist to identify single- or multi-output sub-circuits. The resulting sub-circuits are exhaustively tested to detect timing-independent combinational (TIC) defects. Analyses demonstrate that PEPR-based scan tests detect TIC defects perfectly (100%) when examining fail data from over 30,000 14nm failing chips. In contrast, existing fault models and test metrics might result in up to 92% of TIC defects being detected fortuitously. Strategies for addressing increased test pattern count resulting from the pseudo-exhaustive nature of PEPR testing are also discussed.**

Index Terms—**Test metrics, fault models, scan test, defects.**

I. Introduction

This paper is inspired by pseudo-exhaustive testing research over the past four decades. The growing need for highly thorough testing, beyond today's practices, to ensure robust operation of increasingly complex digital systems has motivated us to revisit pseudo-exhaustive testing. For example, several companies (AMD, Google, Intel and Meta) have recently highlighted the challenge of silent data corruption (*i.e.*, undetected incorrect outputs) caused by substantial manufacturing test escapes [1]–[5]. This paper focuses on defects that are timing- and sequence-independent (TIC [6]). TIC defects are sequence-independent, and a defective combinational circuit remains combinational.

1) A significant portion of TIC defects are often detected fortuitously using existing test techniques. Our fail data from 14nm chips reveals that up to 92% of TIC defects may be detected fortuitously.
2) Classical pseudo-exhaustive testing mostly focuses on TIC defects (although extensions to sequence-/timing-dependent defects are possible).
3) Thorough analysis of silicon data to quantify the effectiveness of pseudo-exhaustive testing for TIC defects forms the foundation for subsequent analysis of sequence- and timing-dependent defects.

An exhaustive test, which applies all possible input combinations to a combinational circuit and checks all corresponding output values, detects all (detectable) TIC defects. An attractive feature of exhaustive testing is that it does not require a detailed analysis of TIC defect behaviors. However, exhaustive testing is not feasible (and likely not necessary) for large designs. To avoid test pattern explosion, many flavors of pseudo-exhaustive testing have been published (such as [7]–[16], and many others). Examples include verification testing, multiplexer and sensitized partitioning, segmentation techniques including physical layout-aware segmentation, gate-exhaustive testing, input pattern faults, partial pseudo-exhaustive testing, and region-exhaustive testing. However, industrial use of pseudo-exhaustive testing has been limited (*e.g.*, [17]) due to test data volume and test-time constraints imposed by the manufacturing test floor. As a result, test research has instead focused on a variety of fault models and test metrics. For example, [18] discussed reduction in test patterns using cell-aware testing compared to gate-exhaustive testing (the latter being a form of pseudo-exhaustive testing).

Recent reports (*e.g.*, [1]–[3]) indicate that existing fault models and test metrics result in substantial manufacturing test escapes (*e.g.*, one or more in a thousand CPUs [2]) that cause major system-level challenges such as silent data corruption from incorrect computations. Test escapes are often detected today after system deployment (*e.g.*, in the cloud) using a wide variety of synthetic and application workloads (*e.g.*, [3], [19]). System-level testing is difficult for several reasons:

1) Each chip might undergo several hours of system-level testing [3].
2) While substantial test escapes are detected, the thoroughness of such system-level tests is unclear.
3) Systematic ways of creating thorough system-level tests with quantified coverage are in their infancy.

Hence, it is important to revisit highly thorough and systematic testing techniques, such as pseudo-exhaustive testing, despite their (increased) test data volume and test time characteristics. Pseudo-exhaustive testing may be applied during manufacturing testing or in the field (for systems with in-field scan testing support, *e.g.*, [20], [21] and others).

In this paper, we introduce Pseudo-Exhaustive Physically-Aware Region (PEPR) testing and demonstrate its effectiveness and practicality using real design and tester data. PEPR explicitly analyzes **both** the physical layout and the logic netlist of the circuit-under-test (CUT) to identify single- or multi-output sub-circuits (unlike many pseudo-exhaustive techniques that focus solely on the logic/physical netlist and/or single-output sub-circuits). These sub-circuits are tested exhaustively for TIC defects (similar to sensitized partitioning [8] although PEPR sub-circuits overlap with each other as discussed later). Extraction of each PEPR sub-circuit from the CUT is determined by several factors: (1) three-dimensional physical layout regions with parameterized length, width and height dimensions, (2) parameterized number of CUT logic levels driving the inputs of and receiving the outputs from the physical layout region, (3) input-output parameters for fanout and isolated nets (*i.e.*, nets not connected to the sub-circuit) within a layout region, and (4) parameterized overlap between various sub-circuits. Unlike many pseudo-exhaustive techniques, PEPR layout regions are intentionally made to **overlap with each other** in all three dimensions to ensure detection of TIC defects that may span multiple PEPR sub-circuits. PEPR parameters do not require detailed simulations of various TIC defect characteristics and derivation of defective circuit-level behaviors.

In this paper, we make the following contributions:

1) We introduce PEPR and its various parameters, and present techniques for (a) extracting PEPR sub-circuits, (b) PEPR sub-circuit collapsing for significantly reducing the overall sub-circuits to be tested exhaustively (analogous to fault collapsing) without sacrificing PEPR thoroughness, and (c) parallelizing PEPR sub-circuit extraction and collapsing for fast runtimes. We demonstrate the practicality of our PEPR algorithms using an industrial 14nm chip design with 18.7 million logic gates. Specifically, a novel tensor-based representation of layout polygon coordinates enables a neighborhood search strategy that reduces computational complexity from O(n^2) to O(dn), where d is a small constant and n is the total number of sub-circuits. Thus, we are able to extract and collapse over 12 billion sub-circuits down to 1.3 billion in less than an hour using an 8-GPU machine. Without this reduction, the number of faults associated with the 1.3 billion sub-circuits would be approximately 114 billion instead of the 37.1 billion examined in this work. In addition, with a CPU-based implementation the run-time would exceed 150 hours.
2) We demonstrate PEPR's effectiveness by using design and tester data from over 30,000 tested 14nm chips. Our results demonstrate that all TIC defects can be detected by PEPR. **Most importantly, PEPR ensures a high degree of thoroughness for TIC defects without relying on serendipitous defect detection (unlike many existing fault models and test metrics).** In contrast, stuck-at, cell-aware and gate-exhaustive faults account for detection of approximately 4.8%, 8.2% and 83.4% of TIC-affected chips, respectively, when serendipitous detection is not included.
3) We demonstrate that PEPR parameters can be quickly tuned based on silicon failures. While other fault models may require detailed failure analysis, PEPR uses high-level physical and logical parameters to determine PEPR sub-circuits and their corresponding inputs and outputs. For our experiments, this enables a rapid 10-minute investigative tuning process that includes (i) extraction of relevant sub-circuits from the full set of 1.3 billion, (ii) construction of new sub-circuits under various parametric settings, and (iii) targeted simulations for updated parameter evaluation. For the examined design, application of this process to 47 failing chips (a mere 0.14% of the 32,723 total) that were not accounted for under PEPR's initial settings led to improved PEPR parameter update. This feedback loop promotes optimal PEPR deployment for new design and process environments with potentially unknown defect types.

II. PEPR Implementation

Sub-circuit extraction is a major component of PEPR. After extraction, PEPR sub-circuits are transformed into faults that are compatible with commercial ATPG tools. The details of PEPR sub-circuit extraction, fault modeling, and test generation are described next.

A. Layout to Sub-circuits

Definition. PEPR examines overlapping, three-dimensional regions, referred to here as voxels, of the physical layout (PVs for short) and its associated logic (LVs). The physical aspects of a voxel are specified by three parameters, namely, l, w and h, where l and w define the length and width of the voxel, respectively, and h defines the height of a voxel in terms of the number layers (*i.e.*, $h \in \{1, ..., p\}$, where p is the total number of layers in the design layout). The circuit extracted from a given PV can also be augmented with additional levels of logic based on two additional parameters, m and n, where m is the number of logic levels added to the sub-circuit inputs, and n is the number of levels added to sub-circuit outputs. The additional levels of logic dictated by m and n are referred to as the logical voxel, LV for short. There are two additional parameters, $f, q \in \{0, 1\}$, for specifying the input-output treatment of some nets that have fanout to more than one location, and isolated nets, respectively, within a PV sub-circuit. Unless otherwise stated, $m = n = f = q = 0$ is used throughout this work, that is, the physical portion of voxels are not augmented with additional levels of logic, and fanout and isolated nets are not designated as sub-circuit outputs. Next, the methodology for extracting the nets/gates within each PV is described.

Physical Voxel Extraction. The physical portions of all voxels are obtained by sliding a three-dimensional sub-region over every targeted multi-layer region. Figure 1 illustrates a region (highlighted in red) with $h = 3$ and $l = w$. As mentioned,

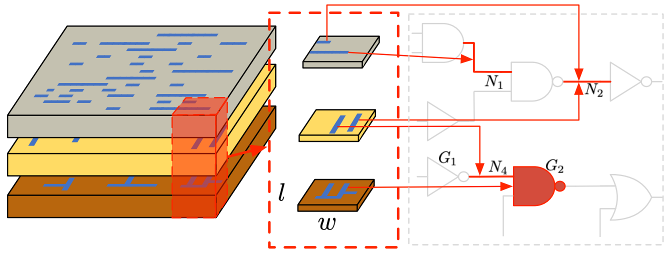


Fig. 1: Physical voxel with $h = 3$, and its corresponding sub-circuit.

Algorithm 1 Physical Voxel Extraction.

```
Input: Multi-layer region with dimensions $(max_x, max_y, h)$
Input: $h, w, l, max_x, max_y$ and $\beta$
Output: Set of PVs $P$
 1: $P = \emptyset$
 2: sub-region size $= (l \times w \times h)$
 3: Initial sub-region location: $(x, y, z)$, where $x = y = 0$
 4: repeat
 5:     repeat
 6:         Place sub-region at $(x, y, z)$
 7:         $p_i$ = polygons in sub-region
 8:         $n_i$ = nets/gates in $p_i$
 9:         If $n_i \nsubseteq n_j$ for all $n_j \in P$ then $P = n_i \cup P$
10:         $x = x + \beta$
11:     until $x \geq max_x$
12:     $x = 0$
13:     $y = y + \beta$
14: until $(x \geq max_x)$ and $(y \geq max_y)$
```

voxels will overlap, and the amount of overlap is not only a function of l, w, and h, but also the step-size parameters β_l, β_w and β_h for each of the three dimensions. In our experiments, we set $\beta_l = \beta_w = \beta$ and $\beta_h = 1$, and $h = 3$.

Given the the layout and a mapping from polygon identifiers to net/cell identifiers, the goal is to determine the nets and cells in each PV, and its relationship to other PVs. Each multi-layer region that is targeted is examined using a sub-region defined by l, w and h using the procedure described in Algorithm 1. First, a sub-region is initially placed at $(x = 0, y = 0)$ within the targeted multi-layer region (line 1). The set of nets and gates corresponding to the polygons p_0 found within the first region are stored in n_0 (lines 6 and 7). If n_i is not already in the PV set P, and is not a subset of any other n_j that is already in P, then n_i is added to P (line 8). The sub-region then moves horizontally to the adjacent location $(\beta, y = 0)$, where again the sets p_1 and n_1 in this next sub-region are compared with the existing sets in P. The loop continues until the end of the row (*i.e.*, $x \geq max_x$) (line 10). At this point, the sub-region moves to the next row (lines 11 and 12) for continued PV extraction and comparison. The inner loop (lines 4-8) continues until the entire multi-layer target is completed (line 13).

All PVs found using Algorithm 1 are net/gate-unique, that is, there are no set instances in P where $n_i \subseteq n_j$. This outcome is result of line 6 of Algorithm 1 which prevents (i) duplicate sets (*i.e.*, no $n_i = n_j$ and (ii) sets that are proper subsets of other sets (*i.e.*, no $n_i \subset n_j$). Preventing duplicate net/gate sets in P means duplicate sub-circuits are not included in subsequent testing tasks such as ATPG, fault simulation, *etc.* A net/gate set n_i that is the proper subset of another set n_j means the logic circuit c_i corresponding to n_i is completely contained within the circuit c_j corresponding to n_j. Exhaustively testing c_j also means the exhaustive test of c_i, which means c_i does not have to be explicitly considered in subsequent testing tasks. Removal of equivalent and subsumed net/gate sets is akin to fault dropping.

Efficiency via Parallelism. Executing line 6 of Algorithm 1 (*i.e.*, the removal of equivalent and subsumed sub-circuits), while straightforward, is an immense computational task due to the sheer number of PVs which can easily be in the billions. To ensure that all equivalent and subsumed circuits are identified, every pair of net/gate sets $n_i, n_j \in P$ would have to be compared, which requires a quadratic computation over an immensely large set.

Given that it is very unlikely that sub-circuits far apart will have an equivalence or subsumption relationship, we limit sub-circuit comparison to a much smaller set of neighboring PVs to improve efficiency. Limiting comparisons to neighborhoods significantly reduces analysis time with inconsequential penalty. Specifically, if all equivalent and subsumed PVs are not removed, it means that the performance of ATPG, fault simulation, etc. can be negatively impacted due to the inclusion of redundant sub-circuits.

To remove identical and subsumed net/gate sets efficiently, we employ a divide-and-conquer approach based on the assumption that PVs far apart are likely different. To accomplish this, we first divide the layout into multiple sub-layouts as depicted in step 1 of Figure 2. Then, all the sub-layouts are analyzed in parallel to extract PVs from the sub-layouts (Step 2 in Figure 2). Figure 2, steps 3-5, illustrates the removal of redundant PVs within a neighborhood. The parameter d defines the neighborhood for a given PV, where d is the number of sub-region positions away that a PV can be for comparison. In Figure 2, $d = 3$ for the target PV shown in green (step 3) and its PV neighbors (shown in grey). Step 4 illustrates the removal of some neighboring PVs presumably due to equivalence or subsumption.

Later, we demonstrate that this massive amount of parallelism enables efficient extraction of all unique PVs in less than an hour for an industrial design with 18.7M gates. In addition, there is little difference in P when analyzing the entire layout singularly or the sub-layouts in parallel.

Sub-circuit Formation. The nets and gates within a PV imply a sub-circuit. Given that PV extraction does not account for the logic netlist in any way, it is possible that the implied sub-circuit is not fully connected. That is, there can be an isolated net(s) within a PV that is not in the transitive fanin or fanout of any of the other gates/nets in the PV. The sub-circuit is essentially a super gate with inputs and outputs. The identification of the super gate inputs and outputs is outlined in Algorithm 2. To determine which nets correspond to the

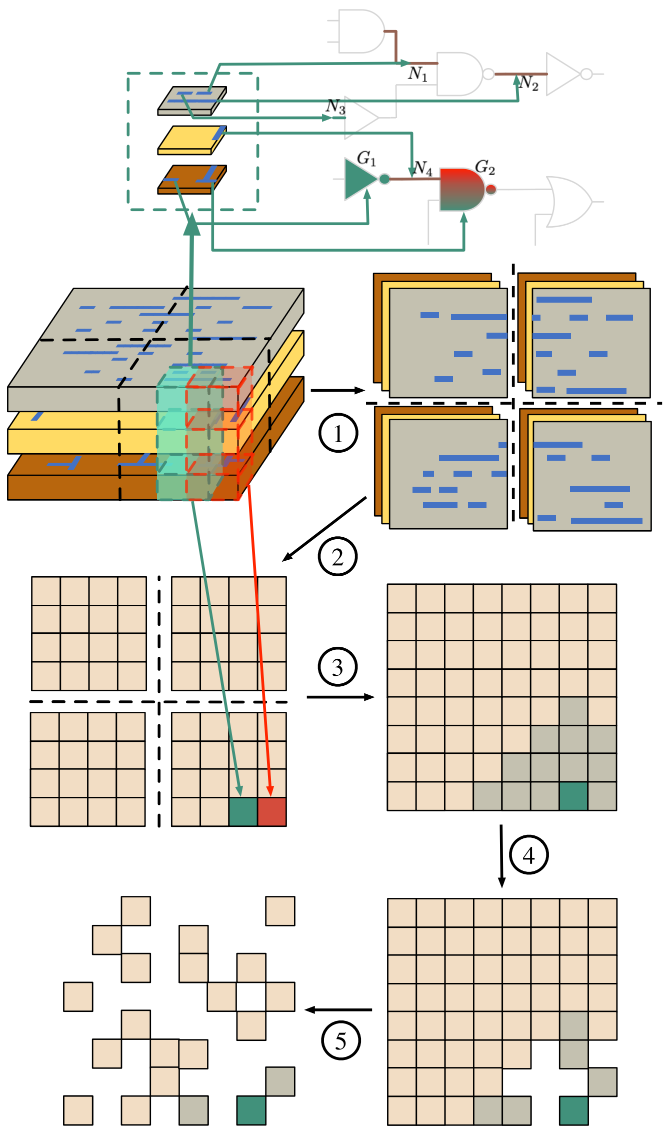


Fig. 2: Illustration of the divide-and-conquer approach used for extracting unique PVs.

inputs and outputs of the sub-circuit, each net/gate is traversed L logic levels forward to obtain the transitive fanout for the PV, and backwards L levels to determine its transitive fanin (lines 4-5).[1] Next, each PV net is analyzed to determine where it falls among four categories, namely: (1) input, (2) output, (3) internal, and (4) isolated. Specifically, lines 7-8 (9-10) determine if a net is a sub-circuit input (output). If an input drives additional fanout that is not connected to a voxel output, it is also assigned as an output when $f = 1$ (lines 11-13). If a

[1]The parameter L is used to limit (i) the search space for identifying the sub-circuit inputs and outputs, and (ii) the resulting size of the sub-circuit. The parameter L should not be confused with the parameters m and n. L does not add logic to the PV but instead is used to determine input-output relationships among nets/gates within the PV. On the other hand, m and n indicate the number of logic levels added to the input and output nets of the PV, respectively, to create the final voxel that has a physical portion (PV) and a logical portion (LV).

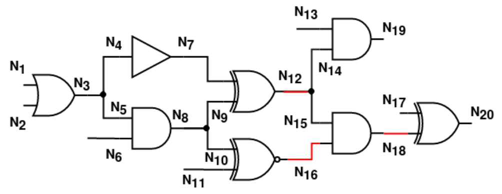
Algorithm 2 SUB-CIRCUIT FORMATION.

```
Input: Net/gate list $n_i$, isolated net parameter $q$, fan-out parameter
    $f$, level search parameter $L$
Output: $inputs$, $internals$, and $outputs$ of sub-circuit $c_i$
 1: $S$ = nets in $n_i$
 2: $S = S \cup$ input-output nets of each cell in $n_i$
 3: $output = \emptyset$, $inputs = \emptyset$
 4: $fanout$ = transitive fan-out of each net in $S$ within $L$ levels
 5: $fanin$ = transitive fan-in of each net in $S$ within $L$ levels
 6: for $s_i \in S$ do
 7:     if $s_i \in fanout$ and $s_i \notin fanin$ then
 8:         $outputs = outputs \cup s_i$
 9:     else if $s_i \notin fanout$ and $s_i \in fanin$ then
10:         $inputs = inputs \cup s_i$
11:         if $f = 1$ and fanout($s_i$) $\not\subset fanin$ then
12:             $outputs = outputs \cup s_i$
13:         end if
14:     else $s_i$ is an isolated net
15:         if $q = 0$ then
16:             $inputs = inputs \cup s_i$
17:         else
18:             $inputs = inputs \cup s_i$
19:             $outputs = outputs \cup s_i$
20:         end if
21:     end if
22: end for
23: $internals = S - inputs - outputs$
```

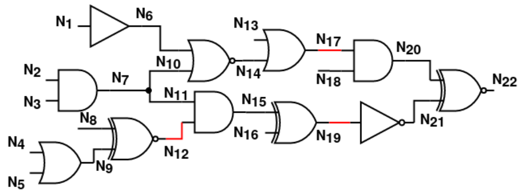
signal is not either in the fanout or fanin, then it is an isolated net (line 14) and treated as an input only (line 16), or as an input and output (lines 18-19) depending on the parameter value of q. Finally, nets that are neither inputs or outputs are internal nets (line 20) which actually are not explicitly needed to describe sub-circuit exhaustive test.

Figure 3 illustrates an example of identifying the I/O of a sub-circuit for differing search values of L. The red signals indicate nets within a PV. For $L = 0$, both the $fanin$ and $fanout$ sets (lines 4 and 5 of Alg. 2) are empty. This means all three nets will be deemed as isolated and labeled as either an input or both an input and output depending on the value of parameter q. For Figure 3(a), for $L = 1$, the $fanin$ and $fanout$ sets are $\{N_7, N_8, N_9, N_{10}, N_{11}, N_{12}, N_{15}, N_{16}\}$ and $\{N_{14}, N_{15}, N_{18}, N_{20}\}$, respectively, which results in N_{12} and N_{16} being inputs, and N_{18} functioning as the sub-circuit output. If the parameter f is one, N_{12} is labeled as an output because it drives fan-out, N_{14}, that is not in the path of the sub-circuit output. In Figure 3(b), for $L = 1$, the $fanin$ and $fanout$ sets are $\{N_8, N_9, N_{15}, N_{16}, N_{13}, N_{14}\}$ and $\{N_{20}, N_{21}, N_{15}\}$, respectively, which means PV nets N_{12}, N_{17} and N_{19} are deemed isolated nets. For $L = 2$, the $fanin$ and $fanout$ sets are $\{N_4, N_5, N_7, N_{11}, N_{12}, N_{10}, N_6\}$ and $\{N_{19}, N_{22}\}$, respectively, which results in N_{12} and N_{17} being deem inputs, and N_{19} functioning as the sub-circuit output. If the parameter q is set to one, the isolated net, N_{17} is both an input and output of the sub-circuit.

Performing path tracing to identify any logical connection between nets within a given PV is not required, meaning that $L = 0$ can be used. If $L = 0$, each net in a PV is treated as an input and an output. However, treating each net as both an input and output leads to a higher IP fault count, many of which are untestable due the likelihood that nets in a PV are logically connected. Therefore, $L = 2$ is used in all subsequent experiments.



(a)



(b)

Fig. 3: (a) Three nets (shown in red) from a PV and its corresponding circuitry. All three nets (N_{12}, N_{16} and N_{18}) are all within one logic level from each other. (b) In this second circuit, N_{12} and N_{19} are separated by two logic levels, and N_{17} is neither connected to N_{12} nor N_{19}.

B. Fault Modeling

Generating test vectors that exhaustively test a voxel-induced sub-circuit requires that all possible faulty circuit functions be efficiently modeled. For this purpose, the input pattern (IP) fault model [22] is employed. An IP fault is defined as $ip \rightarrow (o, o')$, where ip is a circuit input pattern, o is the expected output, and o' is the erroneous output that results if the circuit is faulty. For an n-input, m-output circuit there are $2^n \cdot (2^m - 1)$ possible IP faults. Generating tests for all possible IP faults for a circuit is equivalent to pseudo-exhaustively testing the circuit.

Unfortunately, existing commercial ATPG tools cannot model arbitrary faults that lead to more than one error site. For example, the IP fault $000 \rightarrow (00, 11)$, which is meaningful for a full-adder circuit, cannot be analyzed because both outputs are erroneous. So instead of performing test generation for all $2^n \cdot (2^m - 1)$ IP faults, only the $(2^n \cdot m)$ IP faults with a single output is erroneous is considered. For example, for the 3-input, 2-output full-adder circuit, the total number of IP faults is $2^n \cdot (2^m - 1) = 2^3 \cdot (2^2 - 1) = 24$, while the number of single-error faults is $2^n \cdot m = 2^3 \cdot 2 = 16$. Although multiple-error IP faults cannot be analyzed explicitly, it is likely the case that tests for the single-error faults also detect multiple-error faults. The reason being is that any propagation of multiple errors due to the activation of multiple-error fault are unlikely to completely mask. This means the detection of the single-error faults would also detect multiple-error faults.

C. Evaluation

When developing a new fault model or test metric, it is essential to evaluate its effectiveness in detecting defects. Conventionally, new metrics/models have been evaluated empirically, specifically with experiments in which (production) ICs are tested using multiple test sets generated using new and existing metrics and models. Typically, the "best" is equated to the model/metric that detects the largest number of defects. In addition, the number of defective ICs that are uniquely detected, typically depicted in the form of a Venn diagram, is also considered to be a strong indicator of relative effectiveness [23]–[26]. However, as discussed in [27], deeper analysis is required to really understand the effectiveness of various fault models and test metrics.

A conventional test experiment to evaluate PEPR on silicon will be conducted in the near future. But here, we use the diagnosis-based technique described in [28], [29] to assess the quality of the stuck-at and cell-aware fault models, and the gate-exhaustive and PEPR test metrics. The work in [29] challenges the belief that model/metric effectiveness is correlated to the number of failing ICs detected. For instance, a stuck-at test set typically detects a large percentage of failing ICs, but it is well known that few defects truly exhibit stuck-at fault behavior. In other words, stuck-at tests serendipitously detect other fault types, which means there are no guarantees that stuck-at tests will detect defects.

Model/metric effectiveness in [29] examines each failing chip individually. For each failing chip the failure behavior measured by the ATE (referred to as a fail log), and the expected behavior predicted by a given defect model, or the possible behavior assumed by a test metric are compared. Through this comparison, we can easily conclude if the model/metric is effective for detecting this particular chip failure, or is fortuitous. Applying this approach to a large number of fail logs provides significantly more data for evaluating model/metric effectiveness.

In this evaluation, we compare the stuck-at [30] and cell-aware [26] fault models, and the gate-exhaustive [13] and PEPR test metrics. The flow diagram in Figure 4 describes the details of the comparison procedure. Starting with a fail log obtained from testing with a production test vector/pattern set V, the resulting tester response is used to identify "tester failing patterns" $V_T \subseteq V$. Physically-aware diagnosis is used to identify circuit locations that are likely failure sources, from which a comprehensive fault list is extracted for the key fault models and test metrics. That is, all the faults associated with the locations are identified to determine which faults are detected by the test pattern set V. Specifically, the faults are simulated, and the resulting failing patterns are obtained for each fault i, referred to as "simulator failing patterns" $V_{S_i} \subseteq V$.

To determine whether a particular metric/model explains/captures a tester response, a comparison is made between V_T and V_{S_i}. The method employed for comparison is commensurate for the fault models and test metrics in order

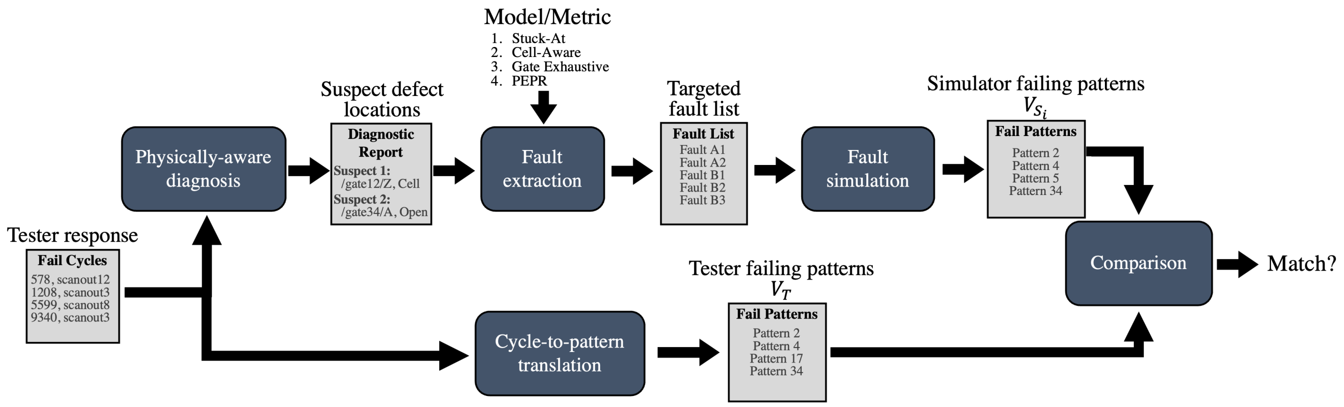
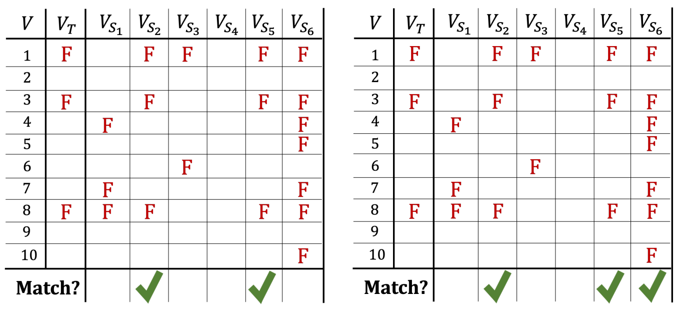


Fig. 4: The flow employed for diagnosis-based model/metric evaluation.



V	V_T	V_{S_1}	V_{S_2}	V_{S_3}	V_{S_4}	V_{S_5}	V_{S_6}
1	F		F	F		F	F
2							
3	F		F			F	F
4		F					F
5							F
6				F			
7		F					F
8	F	F	F			F	F
9							
10							F
Match?			✓			✓	

(a)

V	V_T	V_{S_1}	V_{S_2}	V_{S_3}	V_{S_4}	V_{S_5}	V_{S_6}
1	F		F	F		F	F
2							
3	F		F			F	F
4		F					F
5							F
6				F			
7		F					F
8	F	F	F			F	F
9							
10							F
Match?			✓			✓	✓

(b)

Fig. 5: Illustration of tester-response comparison for (a) a fault model and (b) a test metric.

to align with the intent of the two concepts. A fault model is intended as an accurate representation or predictor of defect behavior, and as a result, a fault simulation response is a match with a tester response when the set of failing patterns for both are equal, that is, $V_T = V_{S_i}$. If $V_T \neq V_{S_i}$, the response of the tester is not explained by a defect assumed by the fault model. An example illustrating this comparison is given in Figure 5a.

Test metrics are intended to evaluate test-set quality, therefore the capability of a test metric should correlate to the number of defects detected that are explained by possible behaviors defined by the metric. As a result, when a given set of signals is identified as relevant to a metric (*e.g.*, gate I/O for gate exhaustive, the signals in a voxel for PEPR), the superset of failures for all values of those signals is used as the set of simulation-failing patterns. Therefore, a match for a test metric occurs when its fail patterns subsumes the set of tester-failing patterns, that is, $V_T \subseteq V_{S_i}$. An example of this form of comparison for a test metric is provided in Figure 5b.

Fault model and test metric evaluation as described requires some details of an IC tester response to be described. First, any tester response (fail log) that reached tester-pin count limits should not be considered because incomplete failure data may confound analysis. Second, as mentioned, we consider only potential fault locations as reported by physically-aware diagnosis to eliminate meaningless comparisons. Third, to avoid potential aliasing of multiple defects, chips with a diagnosis result that indicates multiple defects are also not considered. Application of this evaluation methodology to fail data from a fabricated $14nm$ test-chip design is provided in the next section.

III. Cost-Benefit Experiments

The costs and benefits of applying PEPR to a $14nm$ industrial test chip is explored here. The test-chip design has 12 cores that total to 18.7M gates, and has been particularly designed to improve fault testability and diagnosability. The costs of applying PEPR is examined in terms of voxel identification and ATPG, while the benefits are examined using the diagnosis-based approach described in the previous section. Three points of comparison are included, namely, stuck-at, cell-aware and gate-exhaustive.

To obtain the physical portions of voxels (PVs), the entire chip layout is analyzed to remove unnecessary (equivalent and subsumed) PVs. A step-size of $\beta = 50\ nm$ with a PV of three layers and $125 \times 125\ nm^2$, *i.e.*, $l = w = 125$ and $h = 3$ is adopted. In addition, the search range d for PV comparison is $d = 20$, which means that each PV is compared with neighboring PVs whose Manhattan distance is within $20 \times 50 = 1000\ nm$. PV size is chosen to be approximately $10\times$ that of the chip technology ($14\ nm$).

A. ATPG

The PVs for one of the 12 chip cores are used to create IP faults for the PEPR metric. The signals in the PVs are translated into pin names, and the fan-out and fan-in within $L = 2$ logic levels of each pin are obtained. Using Algorithm 2 (Section II) with $f = q = 0$, the fan-out and fan-in signals are used to determine if a pin in a PV is considered an input and/or an output. The input distribution for each PV extracted from each triplet of layers is depicted in Figure 6. The y-axis gives the number of outputs per PV, while the x-axis provides the number of PVs with that specified number of outputs. The various colors represent the number of inputs for a PV.

The higher-layer regions, M2-M3-M4 and M3-M4-M5, have fewer equivalent and subsumed PVs, and the distributions for those regions have a higher average number of inputs and outputs as compared to the lowest region M1-M2-M3. For the M1-M2-M3 region, the average number of inputs per PV is three, and the average number of outputs per PV is one. However, for region M3-M4-M5, the average number of inputs and outputs per PV is six. The number of inputs and outputs per PV obviously changes with the size of the PV. For example, with a PV size of $250 \times 250\ nm^2$, the average number of inputs per PV for region M3-M4-M5 is 12.

Table III provides fault counts, test set sizes, achieved fault coverages and ATPG run times for the stuck-at and cell-aware fault models, and the gate-exhaustive and PEPR test metrics. Because the evaluated design is a representative core from a test chip with high testability and diagnosability, we find that there is not a significant difference in test pattern counts between stuck-at, cell-aware, and gate exhaustive. While this design observes a $10\% - 15\%$ increase in number of patterns with each subsequent model/metric, previously-examined industrial designs have observed a $1.5\times$ increase in cell-aware pattern count over stuck-at [26], and a $4\times$ increase in gate exhaustive pattern count over cell-aware [18].

As expected, there is a substantial increase in both ATPG runtime and test set size for PEPR. It should be noted however that the PEPR runtime and test set size are likely inflated due to the need to partition the large number of IP faults (37.1B) that are considered. Specifically, due to compute limitations, it is only possible to perform ATPG on approximately 125M faults (200 GB of fault definition files) at a time. The specific break down of the ATPG runs for PEPR by layer is included in Table I. A total of 183 ATPG runs are required to complete the test generation for 37.1B faults. This situation undoubtedly leads to a larger test set because tests generated for a given partition are not fault simulated against all the faults in other partitions. Moreover, the lack of fault simulation against all faults clearly leads to an increase in ATPG runtime.

To demonstrate the potential for reducing test set size, we conducted the following experiment. For faults in the M1-M2-M3 and M2-M3-M4 regions, 378.4M faults are sampled randomly. ATPG is performed on each sample separately, and runtime and test set size are accumulated. Another ATPG is performed on all the samples simultaneously and the resulting runtime and test size is noted. Both experiments are performed on a machine with 1-TB of memory[2]. Table II gives the results of this comparative experiment. The accumulated runtime and test set size for the separate ATPG runs are 59.1 hours and 4,515 tests, respectively. For the ATPG run that included all samples, the run time and test set size are 90.3 hours and 1,659 tests, respectively, which represents a 63.3% reduction in test set size and an increase of 52.8% in runtime. The increase in runtime is expected because the ATPG tool is running test generation and fault simulation on $4\times$ the amount of faults. If the same level of reduction is applied to Table I, the test set size for all faults in Table I would reduce to 192,257 tests.

[2]Note that the 1-TB machine used for this experiment has more resources than the one used to create the data in Table I.

B. Diagnosis

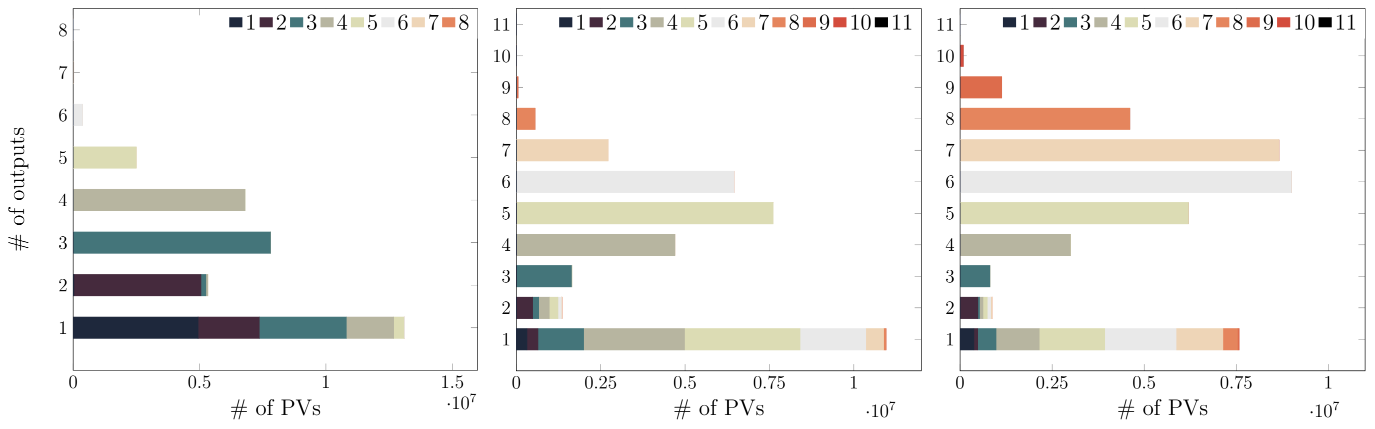
Using the methodology described in sub-section II-C, 32,723 tester responses from all cores of the 14*n*m test chip are examined for fault model/test metric evaluation. The tester responses were collected using a pattern set generated to target multiple-detect static cell-aware faults. For the initial evaluation, a tester response is considered "matched" (as illustrated in Figure 5) if a fault for any of the diagnosis suspect sites align with the tester response. The results of this evaluation are summarized in Table IV.

Only a small portion of tester responses – just 18.7% – aligned with the stuck-at fault model, which is expected as the model is understood to be too simplistic for accurate representation of most defects. The cell-aware fault model matched the tester response for 30.9% of the cases, suggesting that many of the defects are either influenced by signals outside of the suspect cell, or have cell behavior more complex than the provided models for intra-cell defects. In contrast, the gate exhaustive test metric aligns with the tester response for 91.6% of cases. Finally, in more than 99% of cases, initial PEPR settings match the tester response. With a fine-tuning of parameters, PEPR easily achieves a 100% match for all cases. Based on this evaluation, PEPR provides a significant value when evaluating the effectiveness of a test set.

In the left side of Table IV a thermometer relationship is shown to clearly indicate the number of matches among the models and metrics examined. It can be observed that each subsequent model/metric matched all of the tester responses of the one before, plus some additional tester responses. Specifically, the cell-aware fault model matched all the tester responses that match the stuck-at fault model and some more, gate exhaustive matched all tester responses matching cell-aware and some more, and PEPR matched all tester responses matching gate exhaustive and some more.

We also examine a more strict evaluation, in which a tester response is considered matched only if *every* suspect site from a diagnosis report has a corresponding fault that matches with the tester response. This evaluation assumes that the defect could exist at any of the reported suspect locations, and as a result, all should be effectively covered. Detailed in Table V, this form of evaluation shows that PEPR still retains proper alignment for the vast majority of tester responses. In contrast, stuck-at and cell-aware fault models match with just 4.8% and 8.2% of tester responses, respectively, suggesting they may be unable to appropriately represent all observed potential silicon defect behaviors.

In performing the evaluation of the PEPR test metric, there is a small set of 47 tester responses (0.14%) that did not initially match with the PEPR test metric. In all 47 of these cases, diagnosis called out bridge defect suspects. Each mismatching tester response has been investigated to identify the reason why the PEPR test metric, with the parameters used, did not properly align with the observed failure behavior.



(a) (b) (c)

Fig. 6: The input and output distributions for the PVs extracted from (a) the M1-M2-M3; (b) the M2-M3-M4; and (c) the M3-M4-M5 regions. The color of the bars denote the number of inputs in the PV while the y-axis shows the number of outputs. The x-axis provides the number of PVs with each input-output combination.

TABLE I: PEPR ATPG statistics for one core of a 14nm test chip.

Region	No. faults	No. tests	No. ATPG runs	% detected faults	% redundant faults	Test coverage	Fault coverage	Total runtime (h)
M1-M2-M3	1.7 B	5,306	7	59.4%	36.4%	95.4%	59.4%	490.5
M2-M3-M4	9.9 B	124,287	74	52.9%	42.5%	95.4%	52.9%	2,588.6
M3-M4-M5	25.5 B	394,270	102	62.6%	29.2%	91.8%	62.6%	6,489.6

TABLE II: ATPG statistics for parallel and single runs.

No. faults	No. ATPG runs	No. tests	Total runtime (h)
378.4 M	4	4,515	59.1
378.4 M	1	1,659	90.3

For four of the 47 cases, the two signals in the called-out suspected bridge defect have a minimum distance greater than the 125nm voxel size used in layout extraction. Thus, there are no voxels that includes both nets of each bridge, and as a result these four bridges are not part of a pseudo-exhaustive test. For three of the four bridges, the minimum distance between the signals is 160nm, and for the fourth the minimum distance is 184nm. To ensure such a case would align with PEPR, the dimensions l and w could be easily increased for layout extraction. To guarantee that all signals within 184nm will exist within a voxel, it is required that length l, width w, and step-size β are set such that $l - \beta = w - \beta > 184$nm. With a larger voxel, additional nets would be included within the same voxel, at the cost of increased number of signals and complexity for the identified voxels.

For 23 of the 47 cases, we found each to exhibit the following characteristics and behavior related to fan-out signals. The bridged nets are contained in at least one voxel. The bridge nets also exhibited fan-out, and error(s) from the bridge only propagated along the fan-out. The voxel however does not include this fan-out, which means the fan-out nets are not subject to the pseudo-exhaustive test of the corresponding voxel. However, when using the parameter setting of $f = 1$, we can require all voxel signals with fan-outs to be treated as outputs. This would subject those nets to the pseudo-exhaustive test of the larger voxel, which would improve completeness at the cost of an increased number of defined faults for each voxel.

The remaining 20 of 47 cases again revealed that the bridged nets are contained in at least one voxel. In these cases, one of the bridged nets is found to be logically isolated from all of the other signals of the voxel. For the initial parameter settings that include $q = 0$, an isolated net is treated only as an input to the voxel. Because the isolated net is not treated as an output, an error propagating through this net could not be used to detect the fault. To ensure detection of these cases, the parameter setting of $q = 1$ could be used to require isolated signals be defined as both outputs and inputs. This updated parameter setting would come at the cost of an increased number of defined faults for each voxel.

As demonstrated above, the investigation of the mismatched cases (using conventional diagnosis tools) gives us the opportunity to learn "high-level" defect characteristics and tune the key parameters of PEPR. For example, with the knowledge that bridge defects can span distances of 184nm in the examined design and process, the parameters l, w, and β are modified to include such cases. A key aspect of PEPR is that a very detailed defect characterization (e.g., resistance values associated with defects or exact defect locations) is not required. Thus, PEPR is automatically adjustable for trading off cost and test quality. This feedback loop can provide signif-

TABLE III: ATPG execution results for examined models/metrics.

Fault model	No. faults	Patterns	Fault coverage	Runtime
Stuck-at	12.4 M	176	98.3%	0.5 h
Cell-aware	89.3 M	200	96.6%	2.3 h
Gate exhaustive	17.0 M	220	55.3%	4.0 h
PEPR	37.1 B	523,863	59.8%	9,568.7 h

TABLE IV: Matched tester responses for examined fault models and test metrics requiring alignment with at least one suspect.

Model/Metric					Matched responses (of 32,723 total)	
					No. matched	% of total
Stuck-at	0	0	0	6,105	6,105	18.7%
Cell-aware	0	0	4,001	6,105	10,106	30.9%
Gate exhaustive	0	19,853	4,001	6,105	29,959	91.6%
PEPR	2,765	19,853	4,001	6,105	32,723	100.0%

TABLE V: Matched tester responses for examined fault models and test metrics requiring alignment with all suspects.

Model/Metric					Matched responses (of 32,723 total)	
					No. matched	% of total
Stuck-at	0	0	0	1,579	1,579	4.8%
Cell-aware	0	0	1,109	1,579	2,688	8.2%
Gate exhaustive	0	24,615	1,109	1,579	27,303	83.4%
PEPR	5,420	24,615	1,109	1,579	32,723	100.0%

icant value, particularly in new design or process technology environments where defect behaviors may be unknown, or detailed and exhaustive defect characterization may be too complex or infeasible.

In practice, this automated feedback loop can be triggered once a mismatch has been identified, which can arise from the diagnostic evaluation of any set of production test patterns. This process would consist of the following steps: *a*) identification of the reported faulty signals, *b*) localization of the faulty signals within the layout, *c*) construction of new voxels with alterations to layout extraction physical parameters l, w and β, logical parameters m and n, and additional parameters q and f, *d*) translation of new voxels into IP faults, and *e*) fault simulation of tester failing patterns for new IP faults to determine detection status.

In our experience with the examined design, this process can occur rapidly when fully automated. Steps *a*) and *b*) are immediate through a simple lookup. Step *c*) requires an ultra-targeted layout extraction that completes within one minute, and step *d*) simply converts the resulting signals to IP faults. The most time consuming is step *e*), where fault simulation of this targeted list of IP faults is performed for the tester failing patterns. In full, we have found this process to require an average of 10 minutes among these 47 cases, and therefore can easily be performed as needed to continually learn and improve PEPR parametric settings.

IV. Conclusions

At the cost of increased ATPG time and test set size, we have demonstrated that PEPR has the capability to detect all TIC defects deliberately (as opposed to fortuitously). We believe however that both ATPG time and test set size can be substantially reduced with improvements in currently-available EDA tools, and redundancy analysis performed at the voxel level.

Moving forward, the scope and capabilities of PEPR will be expanded in several ways. First, we plan to incorporate timing-dependent and sequence-dependent defects into the PEPR methodology. This means tasks such as voxel collapsing have to be re-examined, for example. Moreover, ideas from small delay defect testing may need to be incorporated. Next, given that our analyses here relied on a $14nm$ test chip, it is important to investigate how PEPR scales with denser technologies. For example, should voxel size remain the same or change? Will sub-circuit inputs and outputs be too large for test generation in terms of run-time and test set size? These and surely other questions must be thoroughly investigated to determine PEPR's practicality in the future. Third, as discussed earlier, PEPR parameters can be customized relatively easily for a given CUT. That is, the insights gathered from conventional diagnosis flows (without detailed analysis of defect characteristics such as resistance values associated with the defects) can be directly used as feedback to tune/update the PEPR parameters for balancing cost and quality. Fourth, the use of PEPR tests does not have to be relegated to the time of chip fabrication. Approaches such as [20], [21] have demonstrated that highly thorough test patterns can be efficiently applied to chips while operating in the field. Finally, a classic test chip experiment is now in the works with an industrial partner. Similar to past experiments, a multitude of test sets will be generated using various fault models and test metrics all of which, will be applied to state-of-the-art commercial chips.

Acknowledgement

We would like to thank and acknowledge John Carulli and Rohan Deshpande of Globalfoundries for detailed discussions regarding the design, test and diagnosis of their $14nm$ test chip. We also like to thank Google for their sponsorship of this project, and in particular, Bharath Parthasarathy, Rama Govindaraju and Peter Hochschild for their on-going input and feedback provided throughout this project.

References

[1] P. H. Hochschild, P. Turner, J. C. Mogul, R. Govindaraju, P. Ranganathan, D. E. Culler, and A. Vahdat, "Cores That Don't Count," in *Workshop on Hot Topics in Operating Systems*. Association for Computing Machinery, 2021.

[2] H. D. Dixit, S. Pendharkar, M. Beadon, C. Mason, T. Chakravarthy, B. Muthiah, and S. Sankar, "Silent Data Corruptions at Scale," *CoRR*, 2021.

[3] H. D. Dixit, L. Boyle, G. Vunnam, S. Pendharkar, M. Beadon, and S. Sankar, "Detecting Silent Data Corruptions in the Wild," 2022.

[4] J. Markoff, "Tiny Chips, Big Headaches," *The New York Times*, Feb 2022.

[5] R. Govindaraju, S. Hansley, S. Sankar, A. van de Van, and S. Mitra, "HW Operation at Scale Reliability to Address Silent Data Corruptions (Panel Discussion)," in *Open Compute Project Global Summit*. Open Compute Project, 2021. [Online]. Available: https://www.youtube.com/watch?v=3yhg4Gt8M_E

[6] E. McCluskey and C.-W. Tseng, "Stuck-Fault Tests vs. Actual Defects," in *IEEE International Test Conference*, 2000, pp. 336–342.

[7] E. McCluskey, "Verification Testing," in *Design Automation Conference*, 1982.

[8] E. McCluskey and S. Bozorgui-Nesbat, "Design for Autonomous Test," *IEEE Transactions on Circuits and Systems*, vol. 28, no. 11, 1981.

[9] E. McCluskey, "Verification Testing — A Pseudoexhaustive Test Technique," *IEEE Transactions on Computers*, vol. C-33, no. 6, pp. 541–546, 1984.

[10] ——, "Quality and Single-Stuck Faults," in *IEEE International Test Conference*, 1993, p. 597.

[11] J. Udell and E. McCluskey, "Pseudo-Exhaustive Test and Segmentation: Formal Definitions and Extended Fault Coverage Results," in *International Symposium on Fault-Tolerant Computing*, 1989, pp. 292–298.

[12] J. M. Acken, "Deriving Accurate Fault Models," Ph.D. dissertation, Stanford University, 1988, copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2022-02-26.

[13] K. Y. Cho, S. Mitra, and E. J. McCluskey, "Gate Exhaustive Testing," in *IEEE International Test Conference*, Nov 2005, pp. 1–7.

[14] A. Jas, S. Natarajan, and S. Patil, "The Region-Exhaustive Fault Model," in *Asian Test Symposium*, 2007, pp. 13–18.

[15] S. Hellebrand, H.-J. Wunderlich, and O. Haberl, "Generating Pseudo-Exhaustive Vectors for External Testing," in *IEEE International Test Conference*, 1990, pp. 670–679.

[16] A. Mumtaz, M. E. Imhof, and H.-J. Wunderlich, "P-pet: Partial pseudo-exhaustive test for high defect coverage," in *IEEE International Test Conference*, 2011.

[17] P. Gelsinger, "Built-in Self-test of the 80386," in *IEEE International Conference on Computer Design*, 1986.

[18] F. Hapke, J. Schloeffel, H. Hashempour, and S. Eichenberger, "Gate-Exhaustive and Cell-Aware pattern sets for industrial designs," in *International Symposium on VLSI Design, Automation and Test*, 2011, pp. 1–4.

[19] K. Serebryany, M. Lifantsev, K. Shtoyk, D. Kwan, and P. Hochschild, "SiliFuzz: Fuzzing CPUs by Proxy," 2021.

[20] Y. Li, S. Makar, and S. Mitra, "CASP: Concurrent Autonomous Chip Self-Test Using Stored Test Patterns," in *Design, Automation and Test in Europe*, 2008.

[21] S. Chakravarty, "Anatomy of an In-Die Tester for Infield Testing," in *Silicon Valley Design for Test Workshop*, 2019.

[22] R. D. Blanton and J. P. Hayes, "Properties of the Input Pattern Fault Model," *International Conference on Computer Design*, pp. 372–380, 1997.

[23] 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.

[24] P. Nigh, W. Needham, K. Butler, P. Maxwell, and R. Aitken, "An Experimental Study Comparing the Relative Effectiveness of Functional, Scan, IDDq and Delay-Fault Testing," in *IEEE VLSI Test Symposium*, 1997, pp. 459–464.

[25] B. Benware, C. Schuermyer, N. Tamarapalli, K.-H. Tsai, S. Ranganathan, R. Madge, J. Rajski, and P. Krishnamurthy, "Impact of Multiple-Detect Test Patterns on Product Quality," in *IEEE International Test Conference*, vol. 1, 2003, pp. 1031–1040.

[26] F. Hapke, et al., "Cell-Aware Test," *IEEE Transactions on Computer-Aided Design of Integrated Circuit and Systems*, vol. 33, no. 9, pp. 1396–1409, September 2014.

[27] R. Guo, S. Mitra, J. Lee, S. Sivaraj, and E. Ameen, "Comparison of Test Metrics: Stuck-at, N-Detect and Gate-Exhaustive," in *IEEE VLSI Test Symposium*, 2006.

[28] Y.-T. Lin and R. D. Blanton, "Test Effectiveness Evaluation through Analysis of Readily-Available Tester Data," in *IEEE International Test Conference*, 2009, pp. 1–10.

[29] ——, "METER: Measuring Test Effectiveness Regionally," *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 7, pp. 1058–1071, 2011.

[30] M. Bushnell and V. Agrawal, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Springer Publishing Company, Incorporated, 2013.