Journal of Electronic Testing (2023) 39:189–205
https://doi.org/10.1007/s10836-023-06057-8

Cost-Effective Path Delay Defect Testing Using Voltage/Temperature Analysis Based on Pattern Permutation

Tai Song[1] · **Zhengfeng Huang**[2] · **Xiaohui Guo**[1] · **Krstic Milos**[3,4]

Received: 14 August 2022 / Accepted: 28 February 2023 / Published online: 25 April 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract
As the ICs become more complex, the duration of high-cost specification tests is increasingly important, especially given the total IC expenditure. In our current work, we propose an adaptive test strategy for reducing the duration of delay testing. This method employs pattern permutation with an ML algorithm to improve the test efficiency, followed by an examination of the effect of test performance, temperature, and voltage on the recognition of path delay defects. SPICE simulations under different voltage and temperature conditions with 65-nm CMOS technology were used to validate it. According to the experimental outcomes, when compared to the random ordering method, the proposed method successfully achieves a nearly 7-fold improvement in test quality at identical testing duration or a 25% reduction in the duration at identical test quality. In addition, the method provides the tester with a thorough understanding of the test efficiency contributions.

Keywords Adaptive test · Machine learning · Defect part per million · Learning to rank · Test cost

Abbreviations

AT	Adaptive test
LR	List Reward
TE	Test escape
ICs	Integrated circuits
DPPM	Defect part per million
ATE	Automatic Test Equipment
ATPG	Automatic test pattern generation
VT	Voltage and Temperature
ML	Machine learning
CUT	Circuit under test
FPSA	Failure pattern selecting algorithm
PMatch	Permutation-Matching
PRank	Permutation-Ranking
CNF	Conjunctive normal form
OTA	Operational transconductance amplifier
DPWN	Deep Permutation-Wise Network
V_{DD}	Supply voltage
LNA	Low noise amplifier

1 Introduction

The ICs test, which ensures the elimination of defects prior to product delivery to customers. As the ICs become ever more complex, high-cost specification tests have become an impediment due to their longer duration. When considering the total expenditure on ICs, the duration of testing seems especially crucial.

Circuits faults can be caused by inherent defects (*i.e.*, stuck-at, open, bridge) or by process variations induced defects (*i.e.*, small delay) [5]. Inherent defects are easier to identify because they behave in a defect-activated state and

Responsible Editor: A. D. Singh

Tai Song
baikediguo@163.com

Zhengfeng Huang
huangzhengfeng@139.com

Xiaohui Guo
guoxh@ahu.edu.cn

Krstic Milos
krstic@ihp-microelectronics.com

[1] School of Integrated Circuits, Anhui University, Jiulong 111, Hefei 230601, Anhui, China

[2] School of Microelectronics, Hefei University of Technology, Feicui 420, Hefei 230009, Anhui, China

[3] System Architectures, IHP - Leibniz-Institut fur innovative Mikroelektronik, Im Technologiepark 25, Frankfurt (Oder) 15236, Brandenburg, Germany

[4] Institute of Computer Science, University of Potsdam, An der Bahn 2, Potsdam 14476, Brandenburg, Germany

exhibit an abnormal output response. However, process-induced defects stay in a defect un-activated state sometimes, and the output response is identical to the defect-free state, making the identification of defects less likely. This comes from the fact that the path delay in line degrades the logic states of the circuit as technology advances, leading to the faulty behavior of the chip [4].

From the technological point of view to source of the issue is related to the inter-transistor mismatch of threshold voltage on a chip, as well as the changes in logic gate delay response to fluctuations in dopant atom quantity in the device channel zone and instabilities in gate oxide thickness [21]. The decrease in resistive voltage causes the valid supply voltage to drop and the driving strength of the gate under consideration to weaken, ultimately prolonging its delay [10]. Path delay tests become particularly critical when the accumulated delay effects induced defects must be considered [7].

Even if some very small delay defects are not considered, catastrophic level TEs (faulty circuits go undetected) can occur [24]. Two kinds of path delay defects are depicted in Fig. 1(a), respectively. The delay shaded in green indicates that they are un-activated delay defects and thus called normal path delay. Correspondingly, the red ones mean that they are activated delay defects and thus called excess path delay. As Fig. 1(b) shows, the period of the observed signal (T_{obs}) differs from the reference signal (T_{ref}) by $\triangle$, $2\triangle$ and $3\triangle$, indicating that the difference between them gradually increases as the clock period increases. As shown in path-1 of Fig. 1(a), the delay defect is not detected when clock-1 arrives. Nevertheless, the cumulative delay effect along this path can cause defect behavior when further affected by clock-2 and clock-3 (in the field of cross domain crossing). This comes from the fact that the delay defect in this path accumulates to a certain extent in subsequent clock cycles.

Path delay defects, a common type of manufacturing defect, can seriously jeopardize reliability in the face of process variations [21]. Several methods for reliability improvement have been proposed. A process-variability-aware parametric fault has widely been examined in AT flow [16], and by exploiting the critical path concept used in the logic test [14], at-speed test [26], and delay test [3], the test coverage is improved and the test time is reduced. Although the influence of VT [24], the body-biasing utilization [8], and the effect of VT fluctuations have already been investigated. In the case of FDSOI technology, the analysis is carried out in greater depth [12], with particular attention to the impact of VT.

Motivated by the aforementioned apprehensions, a competent solution of the adaptive test is presented herein for path delay defects. Then, we determine whether any of the specifications should be skipped during the flow of the test program. The data utilized for accomplishing test skipping include the historical information, along with the information about testing duration. Accordingly, tests requiring a high rate of coverage within a short procedural duration are scheduled first, while those featuring a low rate of coverage and a longer duration are scheduled at the end of the test list. This reduces test time while increasing test efficiency. Interleaved data processing is used in the background during measurement to eliminate any timing overhead.

This paper makes two main contributions:

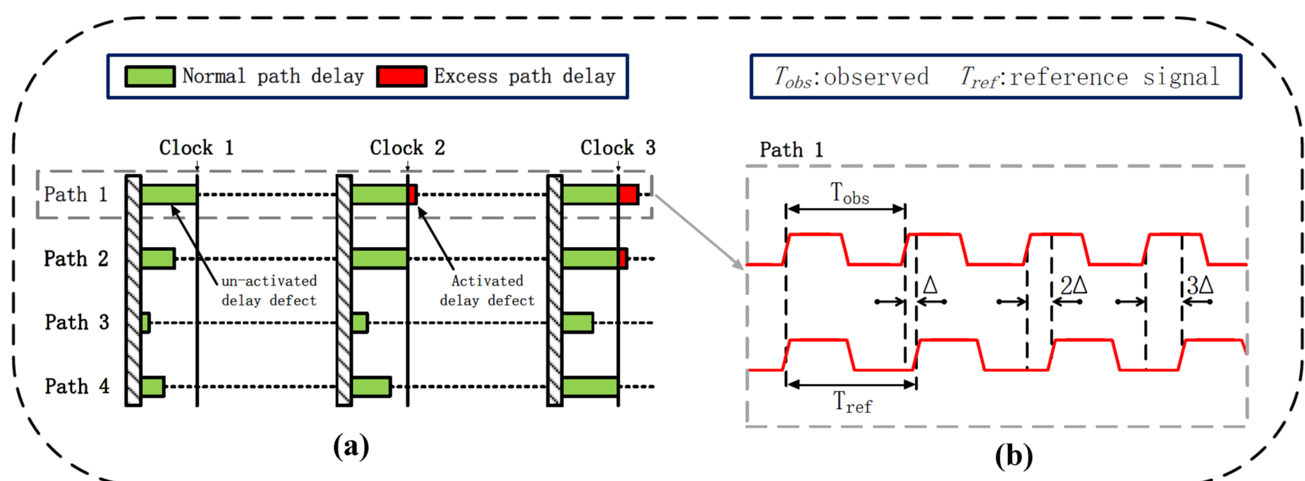

Fig. 1 Path delay fault induced by cumulative effect

1. An ML-based dynamic approach to pattern reordering is presented ensuring valid patterns can be collected and applied first, thereby reducing test time. In this way, the memory overhead of ATE is mitigated by reducing the pattern number.
2. A sensitivity analysis for voltage and temperature is performed, indicating that through a reasonable selection of voltage and temperature, the temporal overhead can be reduced significantly while maintaining the testing performance in path delay defects.

The rest of the paper is assembled as shown below. Section 2 delivers a review of the preliminary and test strategy. In Section 3, the proposed permutation-wise reorder framework is presented. Section 4 elaborates on the experimental setting and outcomes. Section 5 draws the conclusions.

2 Preliminary

2.1 Voltage and Temperature Characteristics for Delay Test

Typically, the delay of circuits varies with the VT [6]. Figure 2 shows the relation of the delay with the VT. During the measurement of a particular path on a constant test pattern, a nearly linear characteristic may appear, given the reliance of the characteristic on the voltage sensitivity of a single path (Fig. 2(a)). In contrast, the path with the longest delay varies with the voltage in the testing process, and the corresponding delay characteristics based on measurements are probably non-linear (Fig. 2(b)). This comes from the fact that the delay path most likely varies with the voltage in the testing process.

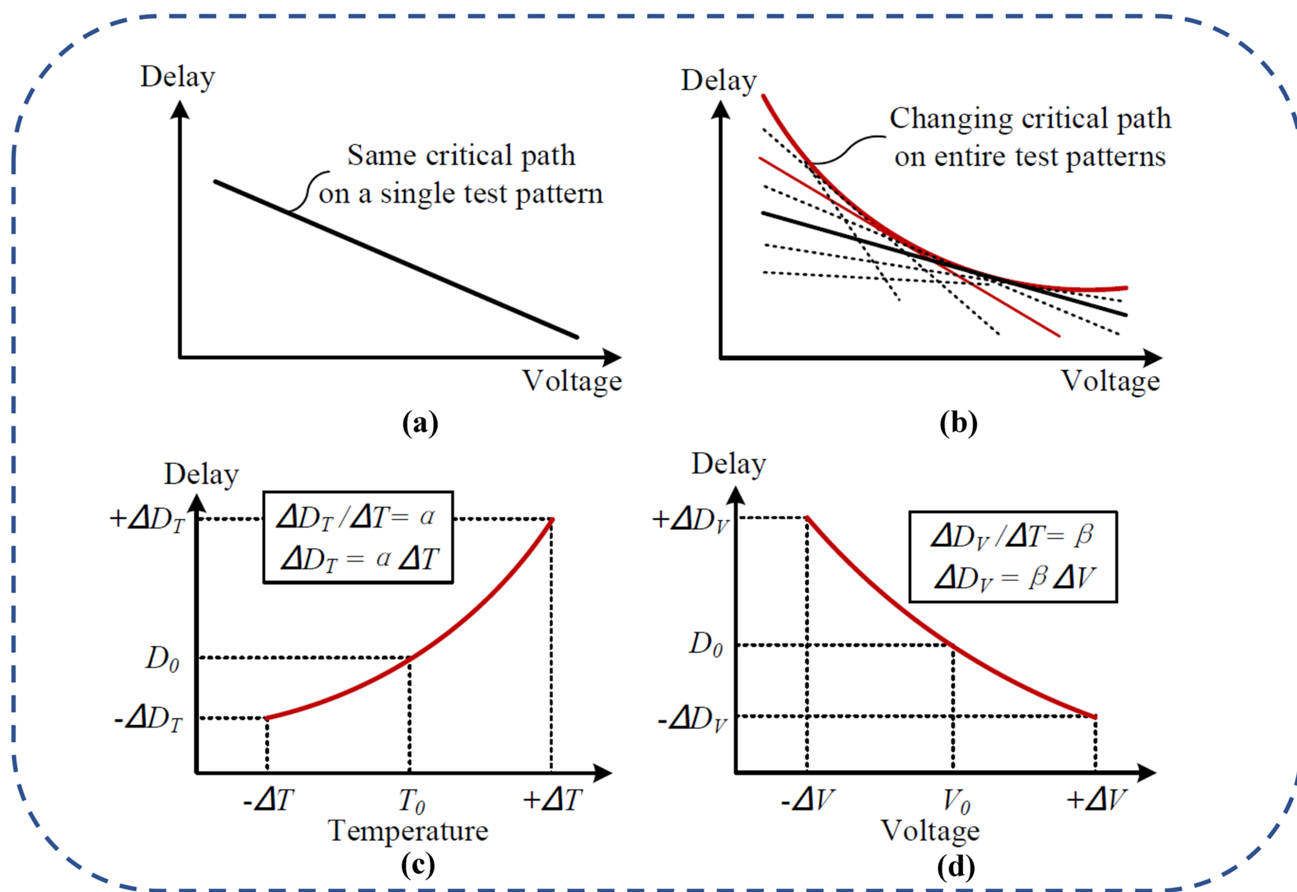

Fig. 2 Delay-voltage relation in the common technology of CMOS Like 65 nm. **a** The identical critical path on a particular path or an individual test pattern. **b** Varying critical paths on the whole test patterns. **c** Relation between delay-temperature relation. **d** Relation between delay and voltage

Figure 2(c), (d) further explained the correlations of delay depending on the voltage and temperature, where T_0 and V_0 stand for the values of temperature and voltage at the initial measurement, and D_0 denotes a delay measured under the T_0 and V_0 conditions. $\triangle D_T$ and $\triangle D_V$ denote the variations in delay caused by T_0-to-$\triangle$T and V_0-to-$\triangle$V temperature changes, respectively. The coefficient α represents the temperature sensitivity for the delay, whereas the coefficient β represents the voltage sensitivity for the delay. Hence, as long as these two delayed coefficients are known ahead of time, the impact of VT can be corrected from delay measurements based on the VT values during the testing process.

2.2 Test Method: Adaptive Test

AT consists of a range of methods for altering the conditions, items, restraints, or outcome of manufacturing tests or the flow of manufacturing automatically to cut the cost of testing. Efficient application of AT requires extra development in the analytics and infrastructure of data, the traceability of die, the design of manufacturing test cell, as well as the coordination among the manufacturers, the test of ICs, and assembly. P. Maxwell [11] proposed the architecture of AT, as shown in Fig. 3.

Figure 3 depicts a model describing the entire end-to-end flow for the to-be-tested parts for AT applications, where "PTAD" stands for "Post-Test Analysis & Dispositioning" and "RT A/O" is short for the "Real-Time Analysis & Optimization". Real-time analysis should be performed during the inspection process for a single die, between successive dies, or within a considerably short temporal window. Figure 3 illustrates the possibilities for feed-forward, feed-back, in-situ, and post-inspection dispositioning that can appear at each stage of testing.

3 Proposed method

The current study creates a pattern reordering flow for the proper acquisition of failing data. Based on this concept, effective patterns will be selected in future tests. This section depicts the pattern-reordering method as follows: Section 3.1 describes the overall framework of the proposed method, and Sections 3.2 and 3.3 detail the specific implementation steps.

3.1 The Framework of the Pattern Reordering Method

Figure 4 shows the failed rate-based ordering of the entire test set, which includes an ML algorithm. Periodical updating of a correlation matrix is possible during the manufacturing ramp-up when new data is available. Initially, our understanding of the circuit is far from adequate. In the absence of a test threshold, we first examine whether the likelihood exceeds a parameterizable pass after entering the test program into a specification measurement routine. If that is the case, that test will be skipped; otherwise, it will be executed. Post completion of a measurement, additional information about the CUT can be obtained, and all likelihood profiles are updated for the unmeasured specifications

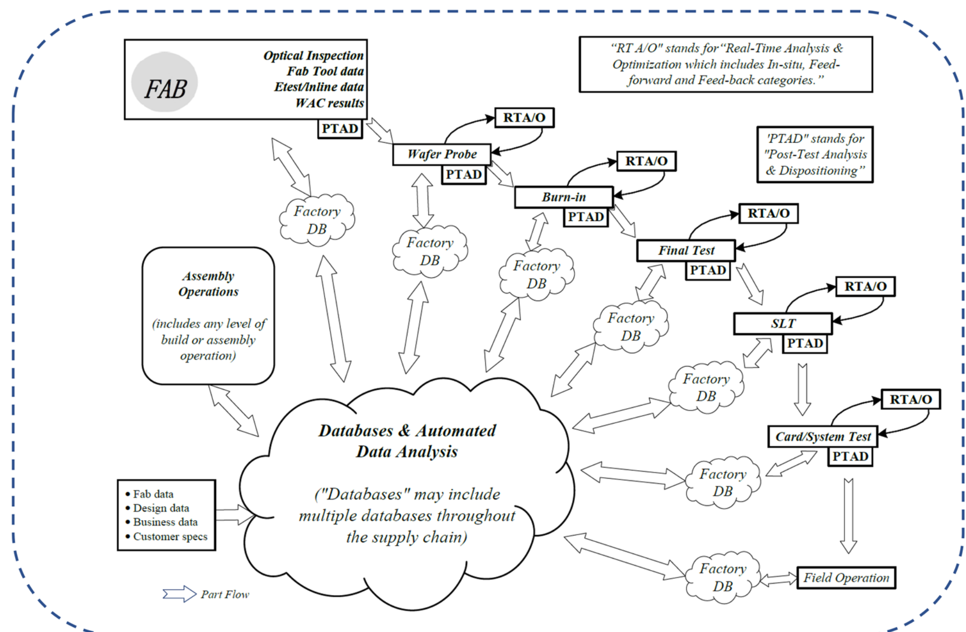

Fig. 3 The AT architecture organizes the test data from each insertion into single or multiple databases. A cascade of manufactured parts can be used to make decisions on test flow, such as insertion, joining, or a database inquiry

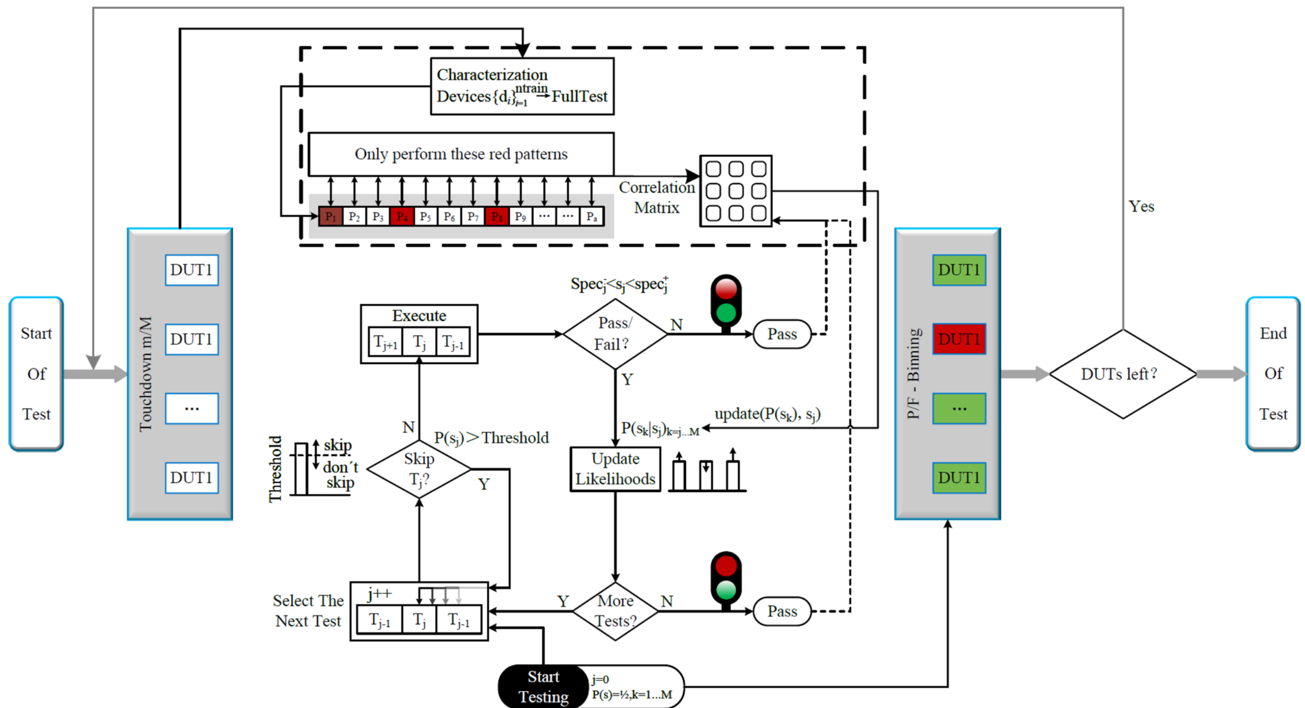

Fig. 4 The Framework of pattern reordering method

until the test sequence ends, or the circuit fails. Intuitively, our strategy has more optimal testing time than techniques that use identical test sets for all devices because we spend more resources on the marginal devices without extending the testing duration.

3.2 Effective Pattern Identification

To perform optimized pattern identification at a low test cost, failing patterns that lead to fast defect identification should be selected first. As illustrated in Fig. 5, all failing results for each CUT are recorded.

Figure 5 gives an illustration of the method to construct new test list from failing patterns, in which each group consists of all failing patterns when failure occurred, and the test pattern number at which the failure is detected. When a CUT generates multiple erroneous outputs, each of failing pattern is selected. For example, in Group *1*, all patterns are applied to each die in CUT#1 until it fails one test, in this case, the corresponding failing patterns (*i.e.*, 4th or 8th) are recorded. Similarly, all

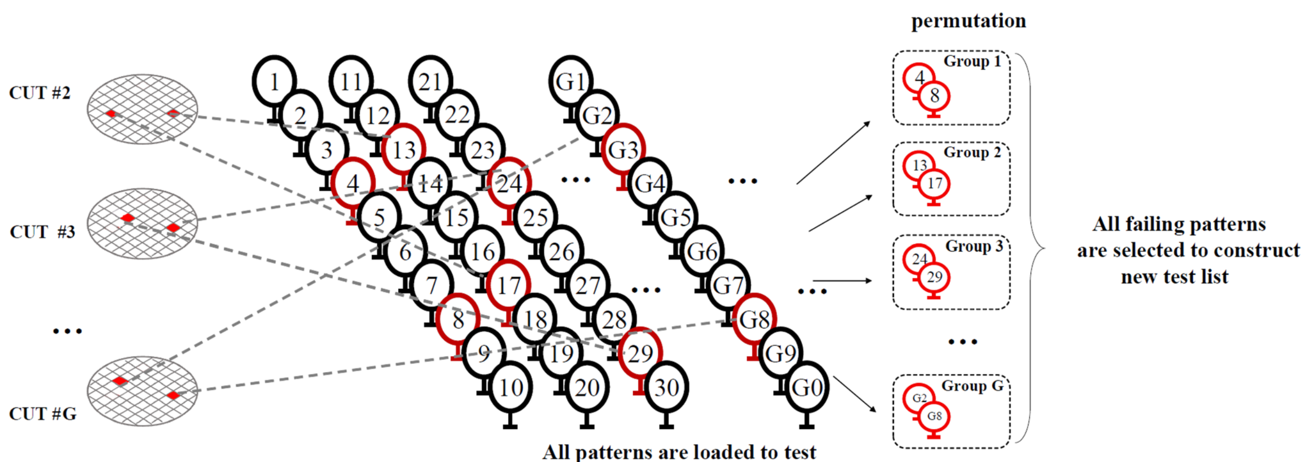

Fig. 5 Method for selecting effective patterns, where the red grids in the CUTs represent the failing dies. The corresponding failing pattern numbers are also marked in red

CUTs (*i.e.*, from #2 to #G) are tested by all patterns and the failing patterns are recorded to construct new test list, indicates the permutation is composed from Group *1* to Group *G*.

3.3 The Architecture of Permutation-wise Framework

Although the failing patterns can be selected to construct the permutation in Section 3.2, different permutations will result in different test times. To find optimal permutation, failing chips during production can provide valuable information for the following tests, and it is desired that all patterns are applied, and all failing tester responses (all failing patterns and erroneous output bits) be recorded.

We present the permutation-wise framework as illustrated in Fig. 6, which sequentially consists of two stages: PMatch and PRank. The goal of the PMatch stage is to obtain a set of candidate lists, with diverse lists produced by proposing an FPSA algorithm via goal-oriented and permutation-wise beam search. Subsequently, at the PRank stage, the LR metric is introduced for the set of candidate lists, which is a uniform criterion on permutation-wise ranking. Its computation is based on rating scores of specially designed DPWN. They eventually recommended the list be the one with the highest LR score.

As a first step, the representations of patterns, the fundamental inputs of the proposed architecture, are carried out. Following the previous research, the available profiles are parameterized into vectors. The labeled interaction records of lists are denoted mathematically as: $R = \{(u, C, V, yCTR, yNExT \mid u \in U, V \subset C \subset I)\}$ by utilizing the item set I and the user set U. In this formula, C and V separately denote the recorded $n-$item input ranking list for the re-ranking phase, as well as the labeled $o-$item final item list shown to user U. Let a pattern V be associated with dense features x_u^d and sparse features x_u^s; every value of a sparse feature is embedded into the d-dimensional space. The lists C and V are naturally expressed separately as $C = \left[x_c^1, \ldots x_c^m\right]$ and $V = \left[x_v^1, \ldots x_v^n\right]$, where m denotes the item counts in C and n denotes the item counts in V. The PMatch and Prank of the developed permutation-wise model will be zoomed into the sections below.

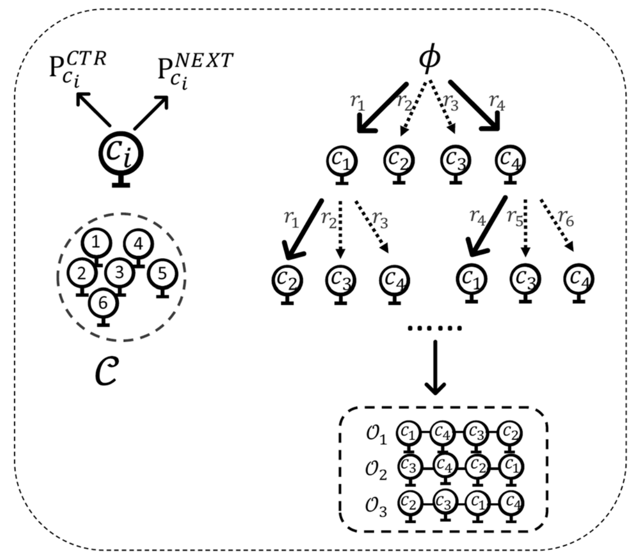

Fig. 7 Illustration of the FPSA algorithm

3.3.1 PMatch stage

The PMatch stage was designed with the goal of more effectively acquiring diverse valid candidate lists. The creation of item lists is possible through the parallel deployment of multiple algorithms, as shown in Fig. 6's left portion, and the lists are then fused into the set of candidate lists. The valid current endeavor ignores the permutation-variant impact within the final item list. The FPSA, a goal-oriented and permutation-wise scheme, is used to effectively leverage permutation-variant impact.

As illustrated in Fig. 7, we consider the re-ranking task as selecting items from the input ranking list sequentially till the pre-defined length. Beam search is a common

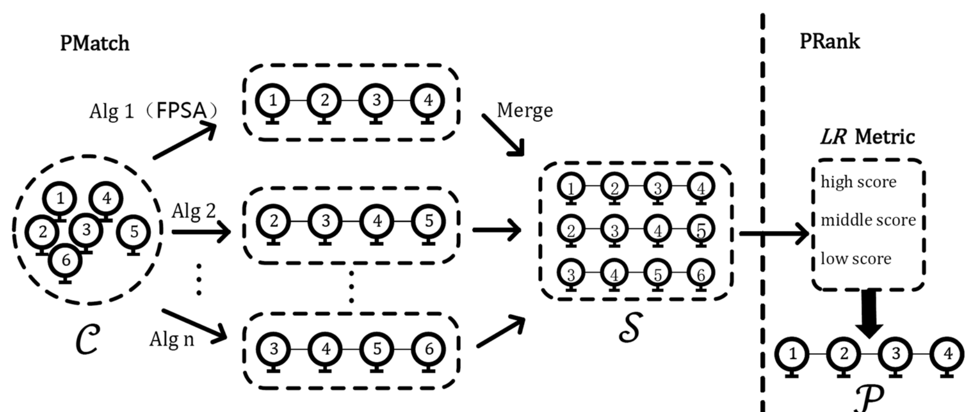

Fig. 6 The overall architecture of the Permutation-Matching (PMatch) Stage

technology to generate multiple effective lists by exploring and expanding the most promising candidates in a limited set. In the FPSA algorithm, we implement the beam search algorithm in a goal-oriented way, that is, we select the lists with the higher estimated reward at each step.

The proposed FPSA is clearly described and depicted in Algorithm 1. Initially, using the converged CTR and next forecast models $M^{CTR}(v \mid u; \ominus^{CTR})$ and $M^{NEXT}(v \mid u; \ominus^{NEXT})$, each item c_i in the list C are attached with 2 forecasted scores, namely the probability of click-through $P^{CTR}_{(c_i)}$ and the probability of continue-browsing $P^{NEXT}_{(c_i)}$. Subsequently, as shown in lines 1-17 of Alg. 1, the remaining candidates for each list in the set $\mathcal{S}$ are expanded thoroughly at every step, and the top k of candidate lists are retained based on the respective reward estimates. At each step of the iteration through the list item, the probability of transitive expose P^{Expose} multiplied by the former items' NEXT scores (lines 18-28) is computed. Upon completion of the iteration, the PV reward r^{PV} and IPV reward r^{IPV} are computed, and their sum is represented by a reward r^{sum}. The r^{PV} and r^{IPV} herein separately denote the *PV* and *IPV* estimates, while the hybrid reward of the list is denoted by r^{sum}.

3.3.2 PRank Stage

The design objective of the PRank stage is to offer a consistent criterion for permutation-wise ranking for the set of candidate lists generated during the PMatch stage. As depicted in Fig. 6's right portion, the permutation-optimal list is chosen from the set of candidate lists (provided during the PMatch stage) by introducing the LR metric, whose computation is accomplished based on the rating scores with the elaborately developed DPWN model. Figure 8 schematizes the holistic architecture of DPWN, which is responsible

Algorithm 1 Fast Permutation searching algorithm (FPSA)

Algorithm 1 Fast Permutation searching algorithm (FPSA)

Input: Input ranking list $\mathcal{C}$; CTR score list $\mathcal{P}^{CPT}$; NEXT score list $\mathcal{P}^{NEXT}$; Output length n; Beam size integer k; Fusion coefficient float α , β.
output: Candidate list set $\mathcal{S}$.
1: New candidate list set $\mathcal{S}$=[ϕ], estimated reward set $\mathfrak{R} = \{ \}$;
2: **//Beam search.**
3: **for** i = 1,2, . . . , n **do**:
4: New candidate list set $\mathcal{S}_t$= $\mathcal{S}$:
5: Clear $\mathcal{S}$ and $\mathfrak{R}$;
6: **for** $\mathcal{O} \in \mathcal{S}_t$ **do**:
7: **for** $\mathcal{C}_i \in \mathcal{C}$ **do**:
8: **if** $\mathcal{C}_i \notin \mathcal{C}$ **then**:
9: New $\mathcal{O}_t$ by appending $\mathcal{C}_i$ to $\mathcal{O}$;
10: Reward ∇ = Calculate-Estimated-Reward ($\mathcal{O}_t$);
11: $\mathfrak{R} \Leftarrow \mathfrak{R} \cup \{\mathcal{O}_t\}$;
12: $\mathfrak{S} \Leftarrow \mathfrak{S} \cup \{\mathcal{O}_t\}$;
13: **end if**
14: **end for**
15: **end for**
16: According to $\mathfrak{R}$, $\mathcal{S} \Leftarrow$ top k of $\mathcal{S}$;
17: **end for**
18://**Calculate reward.**
19: **function** CALCULATE-ESTIMATED-REWARD ($\mathcal{O}$)
20: Estimated PV reward $r^{PV} = 1$,IPV reward $r^{IPV} = 0$;
21: Transitive expose probability $\mathcal{P}^{Expose} = 1$;
22: **for** $\mathcal{C}_i \in \mathcal{O}$ **do:**
23: $r^{PV} += n^{Expose} * n^{NEXT}_{ci}$;
24: $r^{IPV} += n^{Expose} * n^{CTR}_{ci}$;
25: $n^{Expose} *= n^{NEXT}_{ci}$;
26: **end for**
27: $r^{sum} = \alpha * r^{PV} + \beta * r^{IPV}$;
28: **return** r^{sum};
29: **end function**

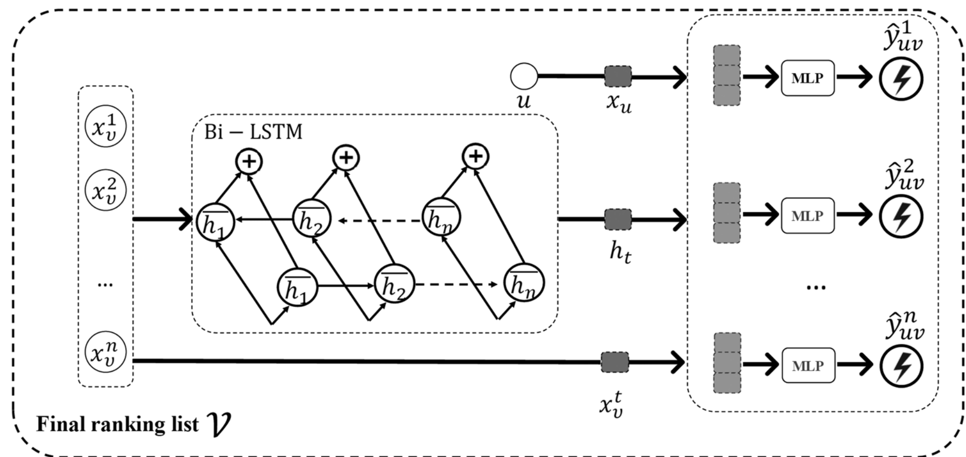

Fig. 8 Technical architecture of the DPWN model

for capturing and modeling the permutation-variant impact embodied in the final item list.

The overall architecture of DPWN is shown in Fig. 8. DPWN is proposed to capture and model the permutation-variant influence contained by the final item list. By taking such motivation into consideration, DPWN $M(x_v^t \mid u, V)$, predicts the permutation-wise interaction probability between user u and the t-th item x_v^t in the final item list V. Mathematically, the forward output state for the t-th item x_v^t can be calculated as follows:

$$i_t = \sigma(\omega_{xi}x_v^t + \omega_{xi}h_{t-1} + \omega_{ci}c_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma(\omega_{xf}x_v^t + \omega_{hf}h_{t-1} + \omega_{cf}c_{t-1} + b_f) \tag{2}$$

$$c_t = f_t x_v^t + i_t tanh(\omega_{xc}x_v^t + \omega_{hc}h_{t-1} + b_c) \tag{3}$$

$$o_t = \sigma(\omega_{xo}x_v^t + \omega_{ho}h_{t-1} + \omega_{co}c_t + b_o) \tag{4}$$

$$\mathbf{h_t} = o_t tanh(c_t) \tag{5}$$

where i, f, o, and c are the input gate, forget gate, output gate and cell vectors which have the same size as x_v^t. Shape of weight matrices are indicated with the subscripts. Similarly, we can get the backward output state $vech_t$. Then we concatenate the two output states $h_t = \overrightarrow{h_t} \oplus \overleftarrow{h_t}$ and denote h_t as the sequential representation of x_v^t . Due to the powerful ability in modeling complex interaction in the CTR prediction field, we integrate the multi-layer perceptron (MLP) into DPWN for better feature interaction. Hence, we formalize the Bi-LSTM as follows:

$$M(x_v^t \mid u, V) = \sigma(f(f(f(x_u \oplus x_v^t \oplus h_t)))) \tag{6}$$

As previously stated, this paper proposes a unified permutation-wise DPWN-based LR metric. The most permutation-optimal list solution is obtained from the set of candidate lists that are generated during the PMatch stage. Specifically, the LR score is computed for every list $\mathcal{O}t$ in the set of candidate lists $\mathcal{S}$ as shown below:

$$LR(\mathcal{O}_t) = \sum_{x_o^i \in o_t} M(x_o^i \mid u, \mathcal{O}t) \tag{7}$$

Thereafter, the highest-scoring list P is chosen and recommended eventually to obtain the permutation-optimal list that best meets the user's demands. Finally, the portion of the PMatch and PRank stages in the proposed framework is marked as the learned scheme of re-ranking ρ, so that the permutation-optimal list P can be created from the list C.

4 Experimental setup and verification

4.1 Test Chip Architecture

The digital CMOS (65 nm)-based test chip was designed to validate the effectiveness of pattern permutation, which conducted according to reference [12] by Yousuke Miyake. Table 1 lists the main functions made possible by the chip, which was mounted on an evaluation board with a short bonding wire to minimize parasitic inductance.

The difference in delay among diverse delay elements is exploited during the design of the clock generator for the variable test. The test conditions or outcomes get saved in internal memory, and there is also a mounted I/O controller. The VT for the chip was evaluated using a thermostatic bath and a tester. The thermostatic bath regulates the chip temperature, whereas the tester is responsible for controlling the input clock velocity, supply voltage, as well as chip operation.

Figure 9 show a test chip architecture and the CUT is Open Cores minSoC, in which a standard scan design is

Table 1 Main functions mounted on the test chip

Module	#Gates	Description	Note
Test Controller	8.2 k	Controller of test architecture	-
Test Internal Memory	21.0 k	Storage of test condition and result	-
Test Clock Generator	1.0 k	Generation of variable test clock	-
TVS Controller	1.1 k	Controller of RO oscillation	-
TVS	1.5 k	Temperature and voltage sensor	3 Ros * 4 Units
Logic BIST (CUT, TPG, RA, etc.)	69.6 k	Logic BIST architecture	MinSoC

taken using commercial EDA tools. Then, typical logic BIST is constructed by connecting an LFSR as a TPG, and a MISR as a response analyzer (RA) to the CUT. Two CUTs are implemented to consist the chip with multiple clock domains. The variable test clock generator is designed based on a method that uses delay difference between multiple delay elements as explained in the next subsection. The TVS consists of three pair of RO and counter, and four TVSs are mounted on the chip. The test controller that manages an entire test circuit, an internal memory used to store test conditions or results, and an I/O controller are also mounted.

A variable test clock generator as shown in Fig. 10 is designed which changes an interval between two clock signals. An input clock passes through two variable delay paths while two output clocks TCLK_C and TCLK_L are used as capture and launch signals in logic BIST, respectively. The variable delay path consists of buffer chains, and the number of the buffers through which the clock passes can be controlled by a parameter DLYC. It can be seen that the number of buffers corresponding to DLYC are 2^0, 2^1 ,2^2, …2^{i-1}, for i =bit of DLYC. Since the implemented DLYC has 8-bits, the delay amounts are 1(2^0), 2(2^1), 4(2^2) ..., 128(2^7). For example, when DLYC are set to all 0s, the clock passes through four selectors in the variable delay path. When it is set to 10110000, the output clock delayed by 7 buffers in addition to the selectors. TCLK_L used as the launch signal can pass through a delay path including some buffers. On the other hand, capture clock TCLK_C always passes through a delay path such that DLYC is set to all 0s. Thus DLYC can delay TCLK_L, the interval between launch and capture signals can be shortened as shown in below part of Fig. 10. Since the test timing becomes smaller by shortening the interval, it suggests that the test clock become faster.

For fabricated chip, the resolution can be measured from a pass/fail boundary of logic BIST when changing an input clock speed sequentially. An evaluation using an actual test chip is performed the variable test clock generator and the delay of one buffer for actual chip is 26.30 ps.

For simulation, the designed clock generator is evaluated using the SPICE simulation based on CMOS (65nm) technology. The delay quantity of a buffer corresponding to a count in DLYC can be calculated from the increase in delay caused by changing the DLYC. It can be seen that the average buffer is 23.88 ps in Table 2, which is very close to the fabricated chip result (26.30 ps), indicating that the effectiveness of simulation method.

4.2 Test Pattern Generation

The goal of test pattern generation is to achieve critical path sensitization and transition propagation to the path at predefined times. The Boolean SAT solver is used to activate the defect-triggered delay on a critical path [25]. SAT problem solving is the process of seeking a solution of f(x_1,x_2,x_3

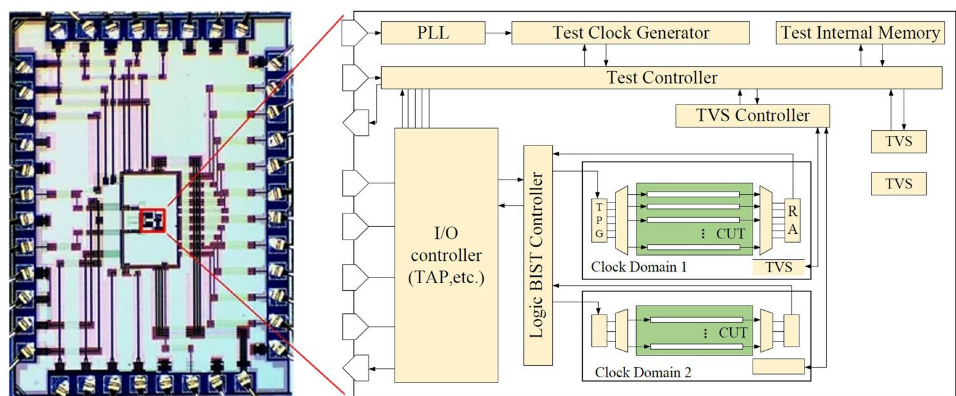

Fig. 9 Test chip architecture with CMOS technology

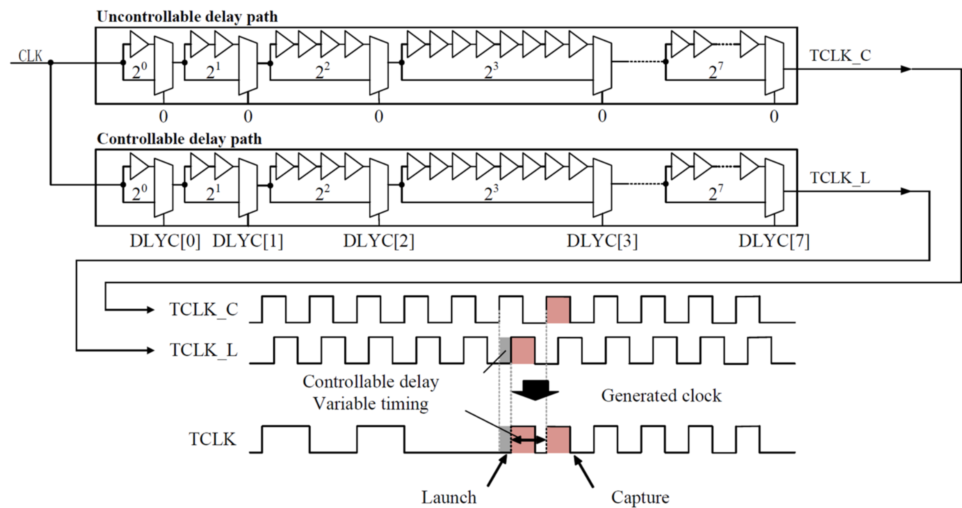

Fig. 10 Variable delay paths in the variable test clock generator

...,x_n)= 1 , where f ( ) is frequently expressed in the CNF, that is, as a product of sums. A clause set C_1,C_2,...C_n is contained in a CNF [9], where each clause is a disjunction of a literal set, *i.e*., a variable or its negation.

The key task of applying an SAT solver to test generation, is to model the test generation problem in CNF. For SAT-based test generation, static timing information is used for sub-path selection. Since accumulative extra delay strongly depends on whether the anterior delay effects are activated, delay estimation in static timing analysis causes inaccuracy for sub-path selection. If the actual activated rate is low, the accuracy of the SAT-based method will not be assured. Based on the above problems, we add a pre-processing phase before sub-path selection. In this phase, the aggressors whose sensitizing constraints conflict with those of the critical path under test are deleted. This way ensures that this method has enough activated rate and accuracy.

Table 2 Evaluation of variable delay path by simulation

DLYC	#Buffer	Delay [ps]	DLYC	#Buffer	Delay [ps]
0	0	-	1111111	127	3014.53
1	1	21.82	10000000	128	3005.37
10	2	35.85	10000001	129	3029.85
11	3	67.23	10000010	130	3057.13
100	4	91.35	10000100	132	3099.47
101	5	116.27	10001000	136	3194.37
110	6	136.75	10010000	144	3394.54
111	7	159.18	10100000	160	3846.26
1000	8	186.59	11000000	192	4557.24
1111	15	339.87	11111101	253	5826.59
10000	16	359.79	11111110	254	5957.82
11111	31	719.43	11111111	255	6089.43
100000	32	776.57			
111111	63	1383.39	Average of buffer 23.88		
1000000	64	1426.89			

4.3 Test time Calculation Method

The number of test patterns influences the test duration because the number of patterns changes the test time. Accordingly, reducing the number can shorten the duration of the test, as shown in Fig. 11.

In this figure, the bars arranged on the horizontal axis are used to represent the durations of 8 tests, while the clock signs arranged there above are used to indicate the time slots corresponding to the test execution or skip. Thus, the T1 (1st test) clock indicates that the execution was done in the 1st time slot. In the case of the 1st device, the procedure begins with the execution of T1. The next step is the comparison of the T2 (2nd test) with the threshold (pass without test) level, and where the threshold is exceeded, this test is skipped. Note that the threshold depends on the mean failure rate of the test pattern. Where T2's likelihood falls below the threshold, the test is executed in the 2nd time slot. Thereafter, T3 shows a likelihood of going beyond a threshold, which is thus skipped. Subsequently, the succeeding tests are sought until a test kept below the threshold is found. In case T4 is less than the threshold, it is run in the 3rd time slot. Later on, the T5, T6, and T7 are skipped since they go beyond the threshold. The procedure continues until the test list is completed either by execution or skipping. For the example extension purpose, the test skipping is diagrammatic for devices #2 and #3 as well. Two crucial features of the algorithm are provided by this diagram. The first feature is the application of tests depending on the device's performance traits. The test is skipped when sufficient confidence has been attained for it. The second feature is that the pattern of test skipping varies from device to device because of

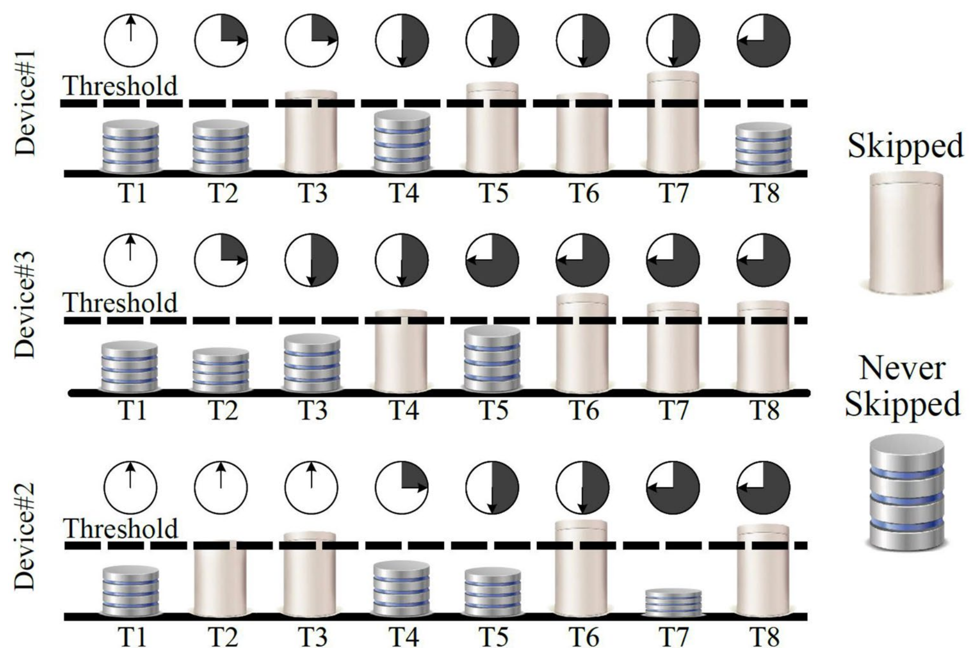

Fig. 11 Test time elimination illustration. The elimination process is demonstrated for three devices. Applied tests are selected based on the performance of the devices being tested

diversity. Thus, with the help of AT higher-efficiency device adaptation and testing execution can be achieved.

Time complexity analysis revealed efficient performance with the PMatch and Prank operations. Table 3 summarizes the time complexities for the time complexity as detailed.

Here *n* represents the total number of the possible value choices with all variable, *m* represents the number of all variables in the algorithm; *a* represents the number of variables in the PMatch calculation; and *b* represents the number of variables in the PRank calculation. Transformation of the raw input values into a coded value between 0 or 1 requires multiple values for each variable to be visited once and stored in an array with a time complexity of O(n), was *n* represents the total number of the possible value choices with all variables. Similarly, reordering the coded values of *m* variable stored in the array had a complexity of O(m), and *m* represents the number of the variable. In total, the time complexity for "PMatch and Prank" is limited to O(n). To verify the validity of the input values, the application checks whether each input is logically meaningful. This operation is done in constant time O (1), through simple comparisons

Table 3 Time complexity analysis of proposed Algorithm

Main feature	Time complexity
calculation	O(n)
Missing data handling	O(n)
Input validation	O (1)
Reset functions	O (m)
PMatch calculation	O (m+ a)
PRank calculation	O (m+ 2b)

and matching between the input and the pre-defined requirement. The "Reset" operation had O(m) time complexity as all input operations must be visited once when their values were cleared. The "PMatch calculation" operation is with a time complexity of O (m + a), as it produces different permutation and test files (where a and b, respectively, represent the number of variables using in PMatch calculation and PRank calculation). While all the variables were calculated once, the items requiring follow-up actions were calculated multiple in separately (*i.e.*, original input and action needed files).

4.4 Test Performance Comparison Method

Besides of test time, DPPM (test quality) is another key indicator for test performance. Reducing test time without jeopardizing test quality obviously can indicate the quality of our method. Therefore, both test time and DPPM are considered together to compare the test performance for different method. In order to calculate DPPM, the number of defect escape (TE) is computed and then scaled to 1 million devices according to (Refer Eq. 8).

$$DPPM = \frac{\#\ \ of\ \ defect\ \ escape\ \ devices}{total\ \ \#\ \ of\ \ devices} * 10^6 \quad (8)$$

Table 4 compares the results of DPPM under identical time levels or duration of test for the same DPPM to the previous techniques. Table 4(a) shows that our proposed method has the least DPPM (number: 97) for the same test time level (approximately1.6s), and Table 4(b) shows that our proposed method has the least test time (1.2s) for the same DPPM (624 ~ 750), indicating that our proposed method achieved the

Table 4 Comparison with the previous techniques

Method	DPPM	Time	Method	DPPM	Time
AT	**97**	1.58	AT	640	**1.2**
RF [20]	650	1.6	RF [20]	650	1.6
Dynamic [22]	310	1.6	Dynamic [22]	655	1.35
AT Flow [23]	100	1.8	AT Flow [23]	640	1.8
PV [15]	700	1.6	PV [15]	700	1.6
TC based [2]	4 k	1.9	TC based [2]	750	0.9
Tutorial [13]	1 k	1.5	Tutorial [13]	699	1.5
Specification [1]	450	1.68	Specification [1]	687	2.1
Relearning [17]	524	1.7	Relearning [17]	624	2
(a) DPPM for the same test time level			(b) Test time for the same DPPM		

best test performance between test quality and test cost. The results demonstrate that the proposed method improves the test quality nearly 7 times over the random ordering method, with DPPM dropping from an average of 650 to 97. Moreover, the testing duration is shortened by 25% in contrast to the pattern ordering approach, where test time dropped from the traditional test of 1.6 s to 1.2 s.

The results derived from the proposed algorithm are compared with the LNA circuit assessments derived from the random ordering-based test list to validate it [20]. The threshold-dependent graphs of DPPM, as depicted in Fig. 12, with the solid and dashed curves, are derived separately using the proposed algorithm and a randomly generated initial list of tests. As is clear, the algorithm can improve DPPM by several orders of magnitude. This advantage is more noticeable at lower DPPM levels.

The comparison of testing duration in Fig. 13 shows that the outcomes based on our algorithm are significantly shorter over the entire range of threshold values. The dependability, shown in Figs. 12 and 13, is another important feature. Vertical bars represent 99.5% confidence intervals. The considerably smaller confidence region with the proposed algorithm at all interest points suggests that the reliability of our method is several folds better than the random ordering method.

4.5 Performance Evaluations of Proposed ML Method

In order to evaluate the test performance of the proposed ML method, the benchmark circuits are simulated, emulating as chips under test, as shown in Table 5. Five benchmark circuits are analyzed to demonstrate the viability of our method for a variety of circuit types. The circuits include c499 and c7552 from the ISCAS'85 benchmarks, s5378 and s7552 from the ISCAS'89 sequential circuits, and b12 from the ITC'99 benchmark suite. The defect simulation is used to create a population of realistic failures from various benchmark designs. The resulting simulation

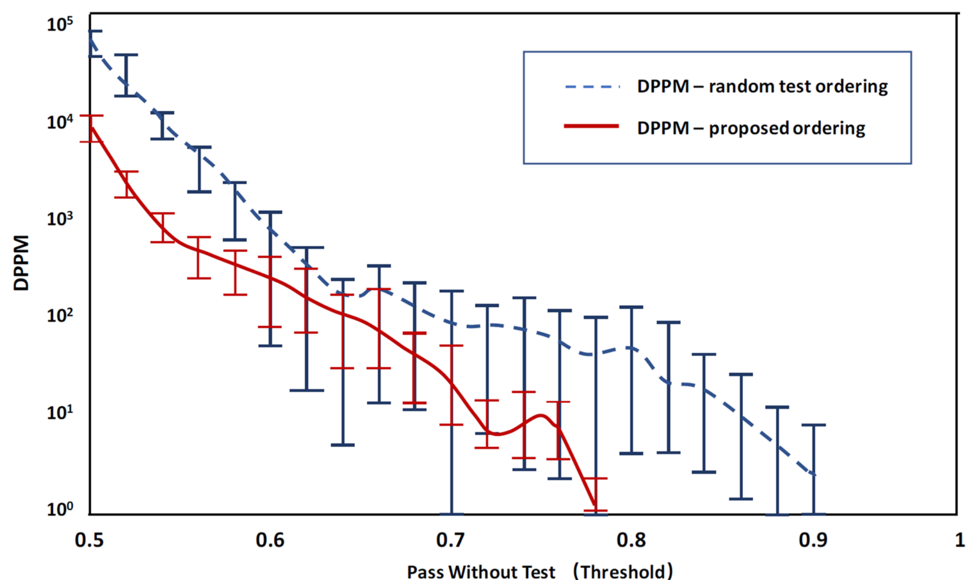

Fig. 12 Effect of threshold on DPPM

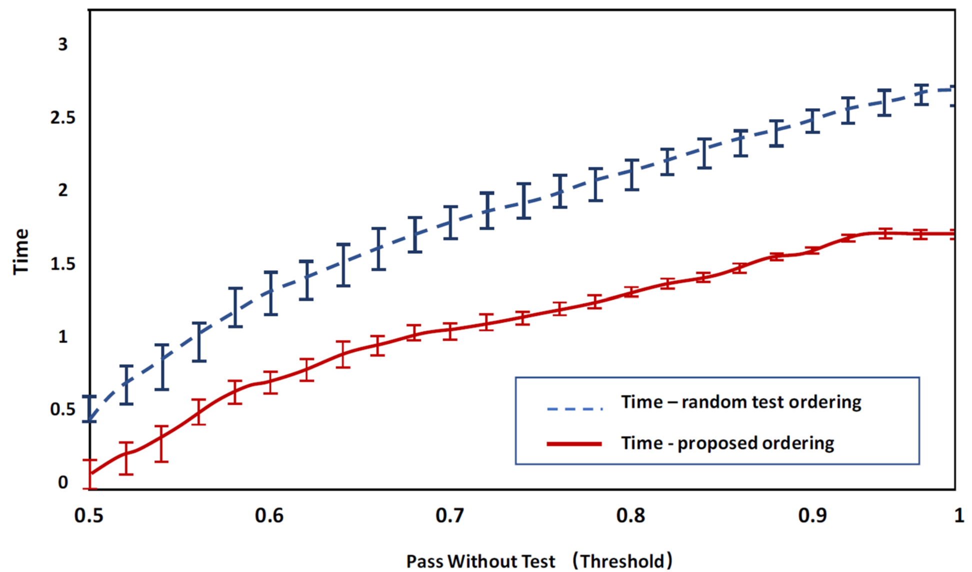

Fig. 13 Effect of threshold on test time

responses form the virtual test data for the failing population of circuits created. Note: Failing chips take up a small portion from the entire set of test chips applied.

For each benchmark, the average number of failing bits and failing patterns for the failing population are listed in the third and fourth columns, respectively. The numbers of failing bits per failing pattern are given in the fifth column. The sixth column presents the sizes of entire test sets, including passing and failing tests. Running time (measured by seconds) for training models and deploying models for use are also reported in the last two columns of the table, respectively. Whenever a CUT fails a test pattern, the termination point for each CUT is recorded.

In this experiment, the test performance metrics in terms of fault coverage (equivalently, test escape or defect level) and test time savings ($\triangle T$) were studied for the proposed AT flows by varying the sample size β. For a given value of β, the resulting trade-off depends on the particular subset of dies that are included in the sample. Therefore, to report trustworthy metrics, for each value of β many trials were performed, where in each trial we randomly populate the sample from the available dies. Figure 14 shows the average fault coverage and test time savings observed across 50 trials as a function of β for the AT flows. The markers correspond to the average numbers, whereas the lower and upper ends of the intervals correspond to the minimum and maximum numbers, respectively.

For a small β, many tests are expected to be skipped and this is likely to give rise to smaller fault coverage or, equivalently, higher test escape rate. Furthermore, the variance of fault coverage is expected to be higher. This is due to the fact that as β decreases the defect level of the set of dies that go through AT and end up being tested with a reduced test suite increases proportionally, thus increasing the upper bound of test escape across the trials. This indicates that estimating the fault coverage or, equivalently, the test escape rate, is indispensable in order to monitor the test quality and identify alarming situations. On the other hand, as the control sample size β increases, the fault coverage FC increases since the failing statistics are computed on more dies and, thereby, they become more representative of the underlying failure modes. The improvement of FC, however, is at the expense of reduced test time savings $\triangle T$ since, as the control sample size β increases, more dies are fully tested and also the kept test suite is likely to be larger. For a given control sample size β. In short, enabling the options within the AT cycle,

Table 5 Information About Failing Population Used for Test

Design	# Chips	# Bits	# FP*	# Bits/FP	# Test	Training	Use
C499	354	6	6	1.0	78	103.47	0.93
C7552	477	26	18	1.4	108	165.22	1.04
S5378	267	54	31	1.7	131	93.56	0.87
S9234	370	37	31	1.2	145	117.41	0.98
b12	277	33	19	1.7	114	86.29	1.6

**FP* Failing patterns

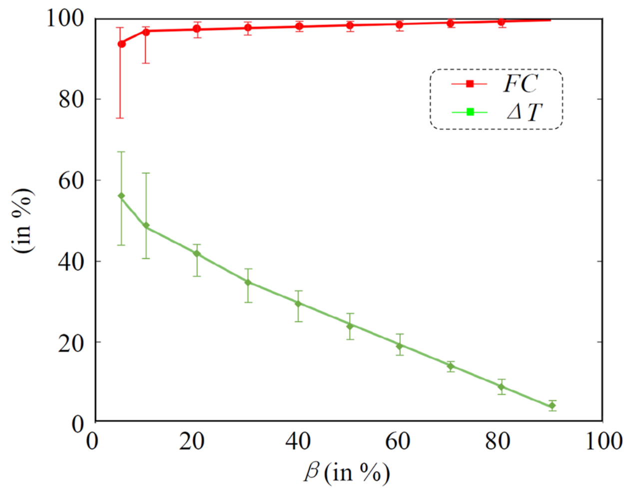

Fig. 14 Fault coverage and test savings as a function of the sample size β for different adaptive test flow scenarios

Table 6 Critical path performance summary

Parameter	Value
Operating Voltage Range	0.8V-1.2V
Frequency Range	50% of Nominal Frequency
Nominal Calibration Delay step	1% of Cycle Time
Standard Deviation of Delay Step	0.1% of Cycle Time
Tracking Accuracy	1.3%
Power Consumption of Voltage Scaling	50 μW
Total Power Consumption	14 mW
Chip Area	$63 \times 53 \mu m^2$

using a large control sample size β, results in a high FC and low $\triangle T$ trad-off. On the other hand, using a low sample size β results in a low FC and high $\triangle T$ trade-off.

4.6 Voltage/Temperature Analysis

Although the experimental results were obtained using ideal VT, the test performance may vary when different VT is used (see Fig. 2). In order to explore the test performance under different VT, and simultaneously, to find an optimal compromise between the quality and cost of the test, test time and TE combined different VT are investigated in Figs. 15 and 16, respectively. Specifically, the voltage range from 0.8 V to 1.2 V is considered. The test time calculation method can be instructed from Section 4.3 and TE calculation method in Section 4.4. The delay circuit's performance indicators are shown in Table 6.

Based on the above VT setup, combined with the test time calculation method in Section 4.3 and TE calculation method in Section 4.4, the results can be obtained as shown in Figs. 15 and 16.

Figure 15 gives the test time concerning the VT values varying from 0.8 to 1.2 V. The two behaviors associated, with the 1.0~1.2 V and 0.8~1.0 V domains are clearly visible in the figure. For the voltage between 1.0 V and 1.2 V, a fast increase in the temporal overhead is noted with the heightening voltage; For the voltage between 0.8 V and 1.0 V, an insignificant variation of the temporal overhead is noted, indicating the low-voltage has little sensitivity to test time in this range. Similarly, the test time sensitivity for temperature in the low-temperature domain (0~50 °C) gives a direct illustration of the temperature having little sensitivity

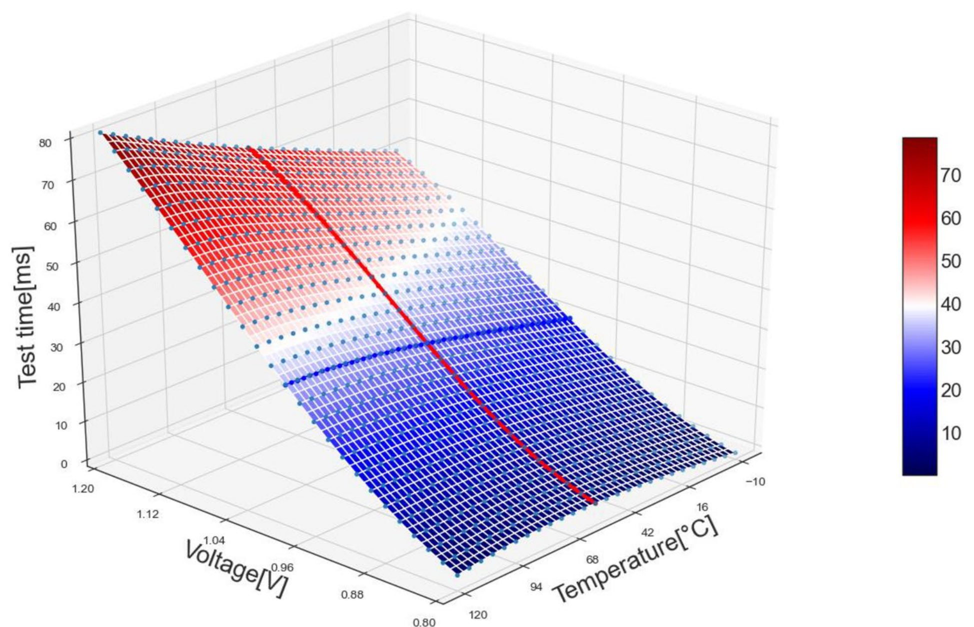

Fig. 15 Sensitivity analysis results for test time overhead

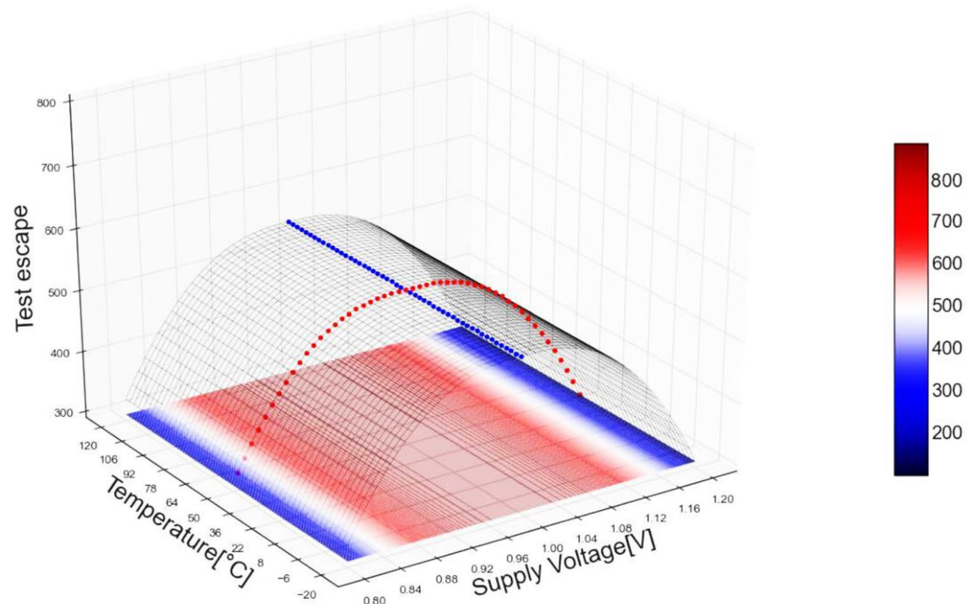

Fig. 16 Sensitivity analysis results for TE

in this range. The previously mentioned facts demonstrate that the sensitivity of the test time for temperature is quite limited when compared to the voltage.

The TE fluctuation over the explored VT space is depicted in Fig. 16. The curved surface represents the minimum-to-maximum TE variation. These results show that the extreme value of TE is when the voltage is around 1V, and the temperature is around 36 °C. However, the detectability scope tends to vary significantly by the VT implementations. For instance, lower temperature and supply voltage are prone to make defect detection more difficult, reducing the detectability of path delay defects. The delay defects are only activated under specific VT, which explains the difference. Therefore, a reasonable selection of VT can greatly improve test performance. Certainly, real-world chip defects probably influence the results differently, which is an issue that needs to be investigated further.

4.7 Threats to AT Strategy Analysis

Some potential threats must be considered in future implementation. First, some real-time requirements may have been introduced during the implementation of our proposed pattern reordering method. Compared to the reference [18], we compressed our code to speed up sorting and obtained less test time. Second, while simulation path delay defects are used in our studies for real-world chip defects, their effectiveness has yet to be verified due to the unitary nature of the defect inputs. Future research should focus on more complex defects. Third, the improved method can update the prediction model based on statistical data compared the reference [19], demonstrating the method's adaptive ability. The current work evaluates testing strategies based on the number of test cases required to detect all of the injected defects, as well as the time spent. It is worth noting that our focus herein is on improving the test efficiency and that the extensively used typical measurements are selected for the study. Hence, the adopted measures adequately support our conclusions. In the future, we also intend to investigate more such measures.

5 Conclusion

In this simulation experiment, a cost-efficient AT strategy for reordering patterns is developed to ensure failed data can be collected and applied first. The primary advantage of this approach is the significantly reduced content of test patterns, which makes it suitable for industrial testing using the ML method. The AT strategy attains sorting by exploiting the permutation present among the failing chips. In addition to identifying the most frequently failing patterns, the failure patterns are recorded to build the permutation. The experimental results indicate that an optimal compromise can be achieved between the quality and cost of the test.

Funding This work was supported by China Scholarship Council under the CSC No. 202206505003 and in part by the Cooperative Education Project of the Ministry of Education No.220803303162437 and in part by the National Natural Science Foundation of China under the Granted No. 62274052.

Data Availability The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of Interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Biswas S, Li P, Blanton R, Pileggi L (2005) Specification test compaction for analog circuits and mems [accelerometer and opamp examples]. In Proc Des Automat Test Eur 1:164–169
2. Chen M, Orailoglu A (2008) Test cost minimization through adaptive test development. In Proc. IEEE Int Conf Circuit Des 234-239
3. Cheng X, Song R, Xie G, Zhang Y, Zhang Z (2018) A new FPGA-based segmented delay-line DPWM with compensation for critical path delays. In IEEE Transactions on Power Electronics 33(12):10794–10802
4. Heo J, Kim T (2021) Reusable delay path synthesis for lightening asynchronous pipeline controller. In IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 29, no. 7, pp. 1437-1450
5. Huang NC, Cheng CW, Wu KC (2022) Timing variability-aware analysis and optimization for variable-latency designs. In IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 30, no. 1, pp. 81-94
6. Huang L, Song T, Jiang T (2022) Linear regression combined KNN algorithm to identify latent defects for imbalance data of ICs. Microelectron J 105641, ISSN 0026-2692,.mejo.2022.105641
7. Javvaji PK, Tragoudas S (2019) On the sensitization probability of a critical path considering process variations and path correlations. In IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 5, pp. 1196-1205
8. Karel A, Comte M, Galliere JM, Azais F, Renovell M (2017) Influence of body-biasing, supply voltage, and temperature on the detection of resistive short defects in FDSOI Technology. In IEEE Transactions on Nanotechnology 16(3):417–430
9. Larrabee T (1992) Test pattern generation using Boolean satisfiablity. IEEE Trans Comput-Aided Des Integr Circuits Syst 11(1):4-15
10. Ma J, Tehranipoor M (2011) Layout-aware critical path delay test under maximum power supply noise Effects. In IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 30(12):1923–1934. https://doi.org/10.1109/TCAD.2011.2163159
11. Maxwell P (2011) Adaptive Testing: Dealing with Process Variability. In Proc. IEEE Design & Test of Computers, vol. 28, no. 6, pp. 41-49
12. Miyake Y, Kato T, Kajihara S (2020) Path Delay Measurement with Correction for Temperature and Voltage Variations. In Proc.2020 IEEE International Test Conference in Asia (ITC-Asia), pp. 112-117. https://doi.org/10.1109/ITC-Asia51099.2020.00031
13. Milor L (1998) A tutorial introduction to research on analog and mixed-signal circuit testing. IEEE Trans. Circuits Syst. II: Analog Digital Signal Process., vol. 45, no. 10, pp. 1389-1407
14. Pomeranz I, Reddy SM (2008) Transition path delay faults: A new path delay fault model for small and large delay defects. In IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 16, no. 1, pp. 98-10
15. Shi CJR, Tian M (1998) Automatic test generation of linear analog circuits under parameter variations. In Proc. IEEE/ACM Des. Automat. Conf., pp. 501-506
16. Shintani M, Uezono T, Takahashi T, Hatayama K, Aikyo T, Masu K, Sato T (2014) A variability-aware adaptive test flow for test quality improvement. In IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 33(7):1056–1066
17. Song T, Huang Z, Yan Y (2022) Machine learning classification algorithm for VLSI test cost reduction. Integration. https://doi.org/10.1016/j.mejo.2022.105549
18. Song T, Liang H, Ni T, Huang Z, Lu Y, Wan J, Yan A (2020) Pattern Reorder for Test Cost Reduction Through Improved SVMRANK Algorithm. IEEE Access 8:147965–147972
19. Song T, Liang H, Sun Y, Huang Z, Yi M, Fang X, Yan A (2019) Novel Application of Deep Learning for Adaptive Testing Based on Long Short-Term Memory. VTS 1-6
20. Stratigopoulos HGD, Drineas P, Slamani M, Makris Y (2007) Non-RF to RF test correlation using learning machines: A case study. In Proc. IEEE VLSI Test Symp., pp. 9-14
21. Takahashi T, Uezono T, Shintani M, Masu K, Sato T (2009) On-die parameter extraction from path-delay measurements. In Proc. 2009 IEEE Asian Solid-State Circuits Conference, pp. 101-104. https://doi.org/10.1109/ASSCC.2009.5357189
22. Yilmaz E, Ozev S (2008) Dynamic test scheduling for analog circuits for improved test quality. In Proc IEEE Int Conf Comput Des 227-233
23. Yilmaz E, Ozev S, Butler K (2010) Adaptive test flow for mixed-signal/RF circuits using learned information from device under test. In Proc IEEE Int Test Conf 1-10
24. Yuan X, Owczarczyk P, Drake AJ, Tiner MD, Hui DT (2015) Design considerations for reconfigurable delay circuit to emulate system critical paths. In IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 23, no. 11, pp. 2714-2718
25. Zhang M, Li H, Li X (2011) Path Delay Test Generation Toward Activation of Worst Case Coupling Effects. In IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 19, no. 11, pp. 1969-1982. https://doi.org/10.1109/TVLSI.2010.2075945
26. Zolotov V, Xiong J, Fatemi H, Visweswariah C (2010) Statistical path selection for at-speed test. In IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 29(5):749–759

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Tai Song has been a teacher at the School of Integrated Circuits, Anhui University, since 2021. He received the PhD degree from Hefei University of Technology, China, in 2021. His current research interests include testing of very large scale integration/SoC circuits, design verification, design for reliability, machine learning about adaptive test, hardware Trojan detection and Chiplet.

Zhengfeng Huang received the PhD degree in computer engineering from the Hefei University of Technology in 2009. He joined the Hefei University of Technology as an Assistant Professor in 2004 and became an Associate Professor in 2010, and Professor in 2018. His current research interests include design for soft error tolerance/mitigation. He is a member of Technical Committee on Fault Tolerant Computing that belongs to China Computer Federation.

Xiaohui Guo received his MSc and PhD degrees from Hefei University of Technology in 2015 and 2018, respectively. Currently, he is an associate professor and master supervisor at school of integrated circuits of Anhui University. His main research includes flexible electronics, MEMS sensors, and embedded control system.

Miloš Krstić received the Dr.-Ing. degree in electronics from the Brandenburg University of Technology, Cottbus, Germany, in 2006. Since 2001, he has been with IHP, Frankfurt (Oder), Germany, where he currently leads the Department of System Architectures. Since 2016, he has also been Professor of design and test methodology at the University of Potsdam. For the last several years, his work was focused on fault tolerant architectures and design methodologies for digital systems integration. He has been managing many international and national research and development projects at IHP (GALAXY, EMPHASE, ICNAO, ENROL, RTU-ASIC, SEPHY, DIFFERENT, VHiSSi, and RESCUE). He is also leading and coordinating space activities at IHP. He has published more than two hundred journals and conference papers. He has registered nine patents.