International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Novel Test Point Insertion Mechanism Using Timing Aware Analysis to Target the Challenges in High Complex SOC Designs

Renold Sam Vethamuthu[1], Sivanantham S.[2]

Submitted: 26/04/2023 **Revised**: 27/06/2023 **Accepted**: 08/07/2023

Abstract. The modern complex, SoC designs use Advanced fault modelling like Cell-Aware-Test (CAT), Automotive fault grading (AGF), and Path delay fault models to cover all the possible potential defects in the design. Due to this advanced fault modelling the coverage is reduced and test vector volume increased drastically around 3x-4x times Compared to the standard ATPG approach. It impacts the reliability, overall testing cost and test time of the design. Test point Insertion (TPI) is the optimal solution to overcome the lower coverage and high volume of vector count issues by providing the controllable and observable nodes in the design to cover the uncontrollable and unobservable nodes. TPI is the methodology to identify the areas in a design which has low testability and require more patterns to achieve the targeted test coverage. At the same time, Test Point Insertion will lead to significant area impact and timing critical challenges after synthesis and post-optimization process. To overcome the Area and critical path timing challenges we proposed a Novel Test point Insertion mechanism by using the static timing Analysis (STA) Approach. In this approach, we calculated the most effective test points by using timing aware analysis followed by fault simulations. Based on the coverage, fault count and area overhead target optimal % of inefficient test points is discarded during the final TPI implementation. Finally, effective timing aware TPIs are inserted in the design to overcome the above mentioned challenges. The experiments are performed on the various design blocks, and the results are compared with conventional TPI approaches w.r.t area impact, pattern volume and coverage. It proves that the proposed Test Point Insertion approach gives a significant improvement in TPI area overhead reduction around 75% and the test vector count reduced to 48% with minimal coverage loss as low as 0.2%**.**

Keywords*: Control nodes, Observe nodes, Test point insertion, TPI, Static timing analysis (STA), critical paths, and pattern count..*

1. Introduction

The quality of electronic devices to meet the critical applications, for example, Unmanned air crafts, Self-driving cars, medical equipment etc., must have high standards and reliability. The need for reliability and assurance is a must as a defective device can cost lives. Therefore, these modern electronic devices need to be tested exhaustively before and after manufacturing to detect and prevent faulty behaviour. Automatic Test Pattern Generation (ATPG) is a technique to generate the patterns automatically and identifies the faults in a semiconductor device [1]. It uses different fault models like Stuck-At (SA) fault modes or Transition Delay fault (TDF) modes to generate patterns. These SA and TDF fault models turn out to be insufficient to ensure high reliability and low DPPM [2][3].

Therefore, every complex SoC design uses advanced fault modes like small delay, path delay and actual layout-based fault models in the design and similarly the patterns, like n-detect, embedded multi-detect or cell-aware [1] are used. These fault models lead to a highly inflated fault count [4][5]. The number of patterns, or pattern count, generated to identify all these faults in a device is very high [6]. Due to this high volume of test vectors, the Automatic Test Equipment (ATE) has a large testing time and tester cost. It also results in reduced test coverage due to truncated patterns. The goal of introducing Test Point Insertion (TPI) is to identify the areas in a design which has low testability and require more patterns to achieve the targeted test coverage [6] [9-12]. The addition of test points to a design increases its controllability and observability. It is useful to enhance the quality and reduce the number of patterns generated during ATPG [13-14].

The test point Insertion mechanism provides a great impact on improvement in test coverage and reduction in test vector volume [15-16]]. Along with these greater advantages, TPI also impacts the Area overhead [17], and critical timing paths which lead to an increase in test time, and production cost and reduces the overall yield of the designs [18-19]. Various techniques have been introduced long back ago to overcome the high pattern volume and low-test coverage challenges [21]. Many test point insertion mechanisms are developed by inserting control points, observe points and a combination of control and observe points [20][22]. The addition of control points and observation points provides the random pattern propagation to particularly hard to detect nodes in the design [23]. The controllability and observability increased by adding extra logic into the design during TPI increases the overall area of the design [24].

[1]*School of Electronics Engineering, VIT University, Vellore, India [0000-0001-7386-9864]*
[2]*School of Electronics Engineering, VIT University, Vellore, India [0000-0002-1010-5178]*
renold.edward@gmail.com, Ssivanantham@vit.ac.in
DOI: 10.15598/aeee.v19ix.xxxx

Some test points are inserted in critical paths in the design leading to more challenges during the synthesis and post-optimization process [48]. Various methods are proposed to target the area and timing critical challenges [22]. In [25] proposed a method to reuse existing Flipflops as control points instead of inserting dedicated flipflops as test points. In [26] [27] proposed a TPI technique in order to use the pre-existing logic as control and observe points to save the extra area overhead problems. A shared test point mechanism is introduced in [7] for multiple control and observe points. Analysing and selecting the internal nets to add the control points into the design leads to improving the fault coverage and minimizing the hardware requirement proposed in [25-27]. With the help of fault simulations, identify the untested faults and corresponding logic cone regions and place the test point into the particular regions to propagate the fault effects to external outputs provided in [5]. In [28] By dividing the entire TPI process into multiple phases and calculating the impact of pattern count coverage by activating the first set of test points into the design. Repeating the same procedure until it reaches the required coverage and pattern count targets in the design.

By using a Probabilistic analysis of fault simulations to compute the test point impact w.r.t controllability and observability of the design introduced in [6]. For identifying the best test points various methods Hybrid test point insertion mechanism [29], gradient based TPI [30], multiple signals correlation and convolution technique[36], and successive approximation[31-32] are introduced. Logic BIST techniques use the test points effectively in order to meet the coverage and pattern count by increasing the random testing approach [33]. LBIST techniques use the TPI to ease the diagnosis by introducing more control on complex nodes in the design [29]. Another mechanism to increase the fault coverage is by blocking the X propagation by controllable points into the design without modifying the actual design [35].

Some techniques insert more test points into the design, but some of them are inefficient, to identify such kinds of test points, a logic cone based TPI was introduced [31]. Random self-test [38-39] is a technique by which a random input stimulus is used to target faults rather than an exhaustive pattern set. Random pattern testability can be used to determine which faults can manifest resistance to random input patterns. Once these faults are identified, it is easier to target these faults using test point registers with minimal hardware penalty [40]. Similarly, when there are unknown or corrupted values in the output, the fault coverage will be affected [41]. To block these "x" values at their source, test point registers are added so they cannot propagate into downstream logic. In addition to this, there may exist multi-cycle paths [42] in the design which reduces the testability of the design. In such cases, test point insertion, which is a flip-flop-based design that is clocked with a synchronous clock, reduces the effort of pattern generation and fault targeting.

Many commercial vendor tools provide various types of TPI implementation mechanisms [44-47]. After analysing the commercial tools developed test points, understand many of them are having very less or no impact on coverage. Vendor tools will insert the test points all over the design without prior knowledge of timing critical paths [43]. Since tools always try to achieve the targeted coverage numbers without considering the Area overhead and Timing critical issues. So, we need to take care with respect to Area overhead, critical timing paths and regions while inserting the TPIs into the designs. If any TPI is inserted in the critical path region, it is very difficult to close the timing and in order to meet the timing, the tool will insert huge delay logic in that logic cone, which will create extra area overhead problems. For that, we are proposing a timing aware TPI to use the effective test points and remove the inefficient TPIs in the design. Since many TPIs in the design will take huge area overhead and very less coverage hike, we need to identify those inefficient TPIs in the design and remove them from the final TPI implementation to overcome the area and time critical iss**ues**.

The remaining paper is structured as follows. Section 2 provides a brief overview of different types of test points (control test points and observe test points) along with that explained a real case scenario of where the control and observe test points are needed. Section 3 describes the complete methodology and flow of the proposed timing-aware test point insertion mechanism. Along with that, the calculation of effective test point characteristics is also provided in section 3. Section 4 provides the results and analysis of various design blocks w.r.t proposed TPI mechanism. Finally, the conclusions are provided in section 6.

2. Control and Observe Test Points

Test Point Insertion (TPI) deals with the insertion of logic or Test Points (TPs) to increase the testability of the design. It creates new control and observes points along the functional path of the design [45]. The test points could be used to force, control or observe. The first one, as shown in Fig. 1, is to force a '0/1', i.e. tied to VSS or VDD, So that the other uncontrollable or unobservable node faults can be detected along with the functional path. The other functions are to control the uncontrollable logic by driving a '0/1' at the uncontrollable logic and observe the unobservable logic in the design, as shown in Figs. 2, 3 and 4.

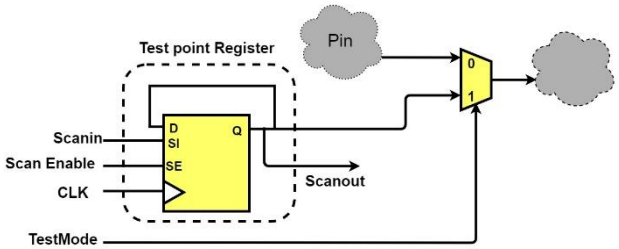


Fig. 1: Test Point Circuit - Force '0/1'

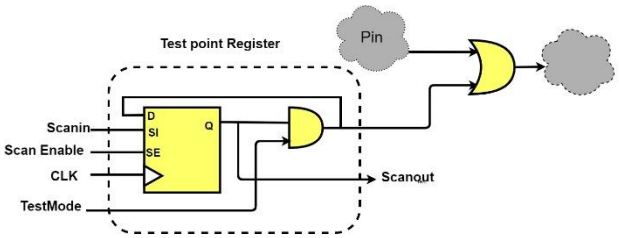


Fig. 2: Test Point Circuit - Control '0/1'

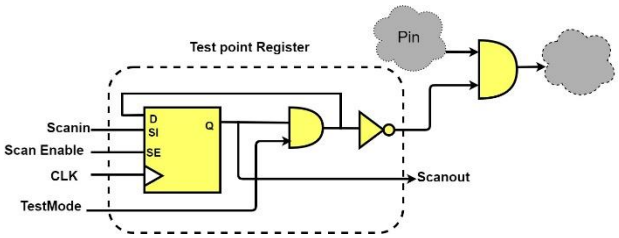


Fig. 3: Test Point Circuit - Control '0/1'

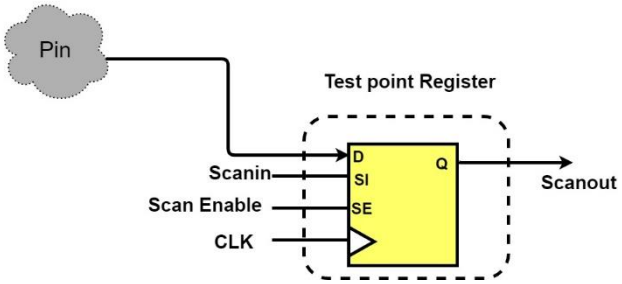


Fig. 4: Test Point Circuit - Observe '0/1'

Though the controllability and observability of the design is improved with test point insertion, the addition of test point registers also increases the area overhead. Test point sharing is a feature where a single test point register, which could either be a source or a sink, can be shared across multiple test point pins [46]. For example, in Fig. 5, inputs of multiple OR and AND gates need to be controlled. Instead of introducing individual test points for every gate, a single source test point register can be used to control the multiple test point pins. Similarly, in Fig. 6, outputs of multiple test point pins need to be observed. In this case, the use of a single sink test point register can ensure observability of the design. The test point sharing feature is useful to ensure improved testability of the design with a negligible area overhead.

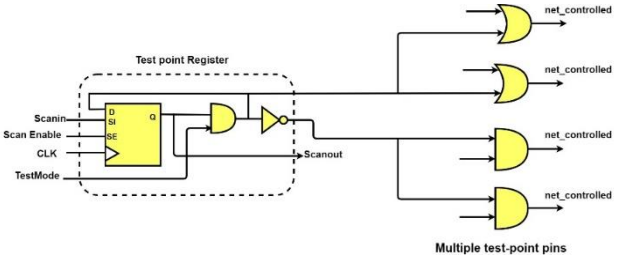


Fig. 5: Test Point Sharing - Control

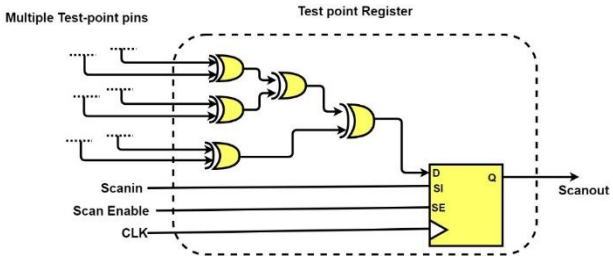


Fig. 6: Test Point Sharing - Observe

Test points that are added to a design could be test point registers, combinational logic or both. In Fig. 7, the Reset of Flipflop (FF) B is uncontrollable during the test. This uncontrollable asynchronous set/reset can be made controllable by adding a test point at the reset of FF B. The test point that is added, in this case, is an OR gate. The inputs of the OR gate are connected to the uncontrollable node and the test mode signal. Based on the assertion or de-assertion of the test mode signal, the reset of FF B is controlled.

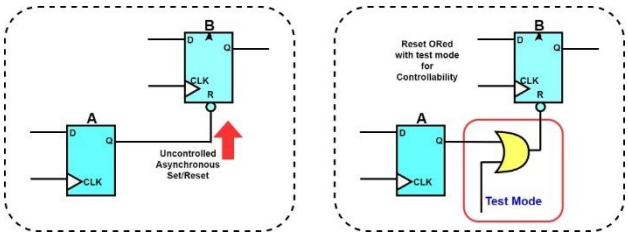


Fig. 7: Figure 7. Uncontrollable Asynchronous Set / Reset made controllable using OR type TPI

Redundant logic introduces redundant faults which may not be testable in ATPG. In Fig. 8, the output of the 3-input AND gate 'D' cannot be tested for Stuck-At '1' (SA1) fault. To ensure testability, a test point register is inserted at this redundant node. This test point register can be used to control, observe or both.

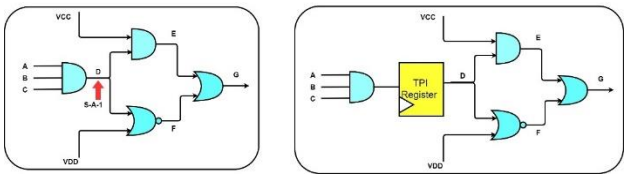


Fig. 8: Untestable redundant fault made testable by adding TPI FF

In Fig. 9, FF A is a clock divider which drives the clock of FF B. In this case, the clock input of FF B is not controllable by the TestClk. To improve controllability, a multiplexer can be added at the uncontrollable clock node. The Clock Mux

will take one input from FF A and another input from the TestClk. The select line of the Clock Mux is controlled by the TestEn signal which, when asserted, ensures that TestClk is selected during test mode. A similar setup using a clock Mux has been implemented in Fig. 10 to ensure that the Testclk reaches FF B during the test with the test mode signal as a select line.

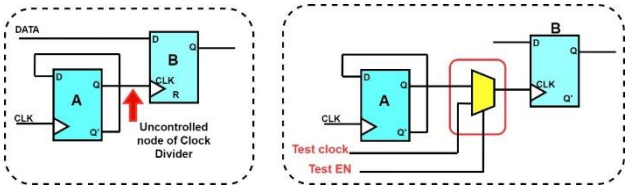


Fig. 9: Uncontrollable Clock MUXed with TestClk for controllability with TestEn as select

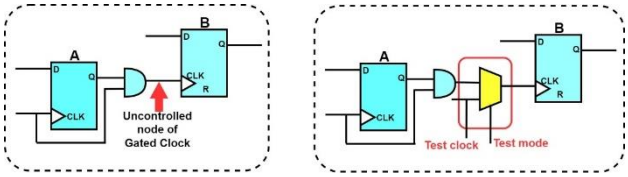


Fig. 10: Uncontrollable Clock MUXed with TestClk for controllability with test-mode as select

The major targets of test point insertion are [45]:

- To increase test coverage: Random- Resistant Fault analysis to reach the maximum achievable test coverage
- To reduce pattern count
- X - Blocking
- To remove multi-cycle paths

The design blocks used for the experimental analysis, some of the design blocks are considered with different area overheads, pattern counts and test coverage. The base performance for these tiles does not reach the targeted performance due to the ATPG Untestable (AU) logic or Un-Controllable (UC) or Un-Observable (UO) logic present in the design. To improve the controllability and observability of the design, Test Point Insertion (TPI) is implemented. Below are some of the nodes or logic cones needed as test points in order to make the uncontrolled and unobservable nodes to testable.

2.1. Control test point

Controllable test points target the uncontrolled faults of the fanout cone in the design and propagate fault response towards the primary output. Control points in the design have the larger fanout cone which means it has high controllability on more pins and cells in the design which are hard to control without that test point. In Fig. 11, the output of the register "write_data_reg" drives the SI of the register "Meb_reg". Here, the register "write_data_reg" is part of the memory module, which makes the input of the register "Meb_reg" to be uncontrollable as there are no wrapper cells for this memory module. For the "control 0/1" driven from" write_data_reg" as shown in Fig. 12, Additional TPI flop "test_point_register" is added to control the input of the register "Meb_reg".

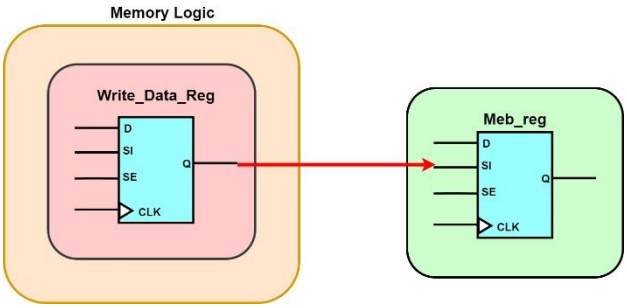


Fig. 11: The Input of Meb_reg is uncontrollable

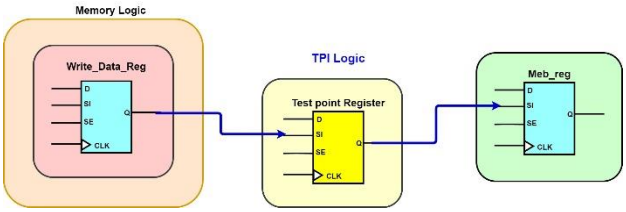


Fig. 12: Additional TPR flop is added to make SI of Meb_reg controllable

In Fig. 13 To force the output of 3 input NAND gate to 0, all three inputs need to be 1. Moreover, the inputs of this NAND gate are connected to a huge combo logic cone. To make the output of the NAND gate easily controllable (control 0), Additional AND gates are added as shown in Fig. 14 to control the propagation of 0 from the 3 input NAND gate.

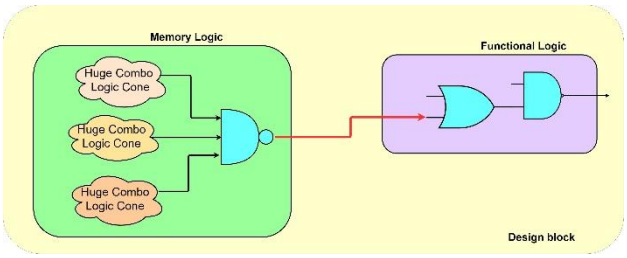


Fig. 13: Inputs of the NAND gate are connected to a huge combo cone which reduces controllability

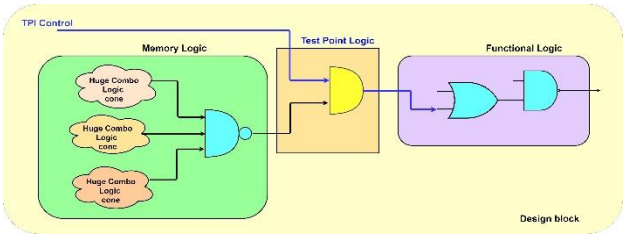


Fig. 14: Additional AND gates are added (as test point) to improve controllability

Similarly, in one of the design blocks consisting some of the uncontrollable nodes like to force the ZN input of the NOR gate to 1 in Fig. 15, the combo logic connected to it must be

forced to drive '0'. Since the inputs of the NOR gate are connected to a huge combo cone, controlling the output of NOR gate to 1 is challenging. For the "control 1" driven from NOR gate, Additional OR gate is added as test point, as shown in Fig. 16 during TPI to control propagation of '1' easily.

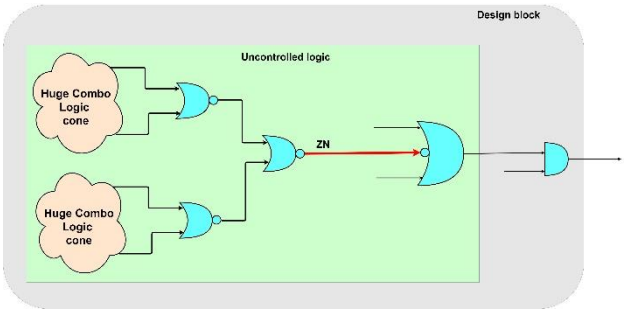


Fig. 15: Inputs of the NOR gate is connected to a huge combo cone becomes uncontrolled node

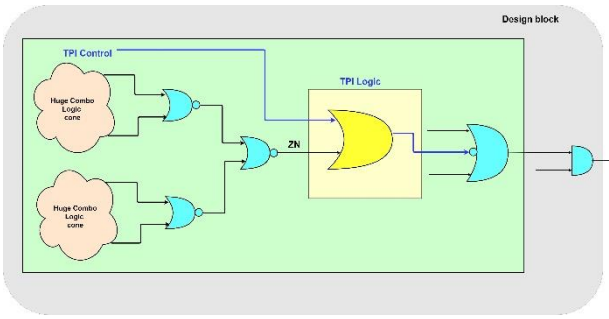


Fig. 16: Additional OR gate is added to the logic cone (as a test point) to improve controllability

2.2. Observe Test Point

Observable test points target the unobservable faults of the nodes which consisting high fan in cone and a greater number of not-detected faults. This observes test points targets all such kinds of unobserved faults and always the inputs propagated towards the inserted signals. Here, in Fig. 17, the flop output is going directly to the memory wrapper and there is no connection or path to propagate that output to IO. In observe mode, multiple observe flops can be used to observe the output of local 0 regs. For the "observe 0/1" from reg: Additional TPI flops are added, as shown in Fig. 18, to observe propagation from the register.

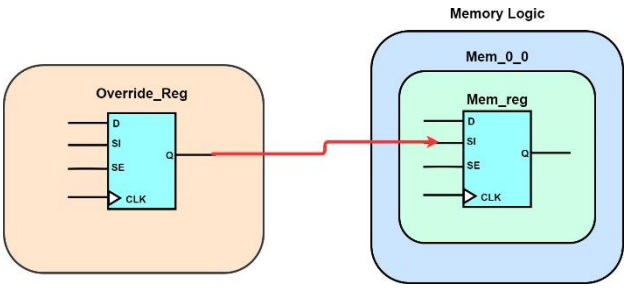


Fig. 17: Register output is going to directly to memory wrapper which is not observable

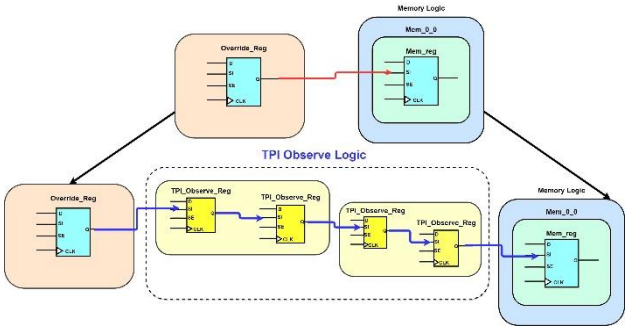


Fig. 18: Additional TPI flops are added (as test point) to observe the output of the register

3. Proposed Timing Aware Test Point Insertion Mechanism

Generally, the TPI techniques identify the signal nodes in combinational logic to add the controllable and observable points for reducing the number of patterns to test the particular logic in the circuit. The control test points target un-controlled faults of the fanout cone in order to increase the controllability of the design. Similarly, observable test points target the unobserved faults along with the fan-in cone of the design to increase the observability of the design.

As per the tool's perspective test points are the logic nodes in the circuit and the tool inserts the extra logic in order to increase the testability of the logic cone in the design. These tools perform random pattern analysis iteratively to get the required coverage and lower pattern counts. Generally, this analysis was performed and inserted test points before the Synthesis and timing optimization of the design. some test points in the design generated and inserted without considering the timing critical paths and long delay paths, it directly impacts slack and path delay of the design. During the synthesis and physical design routing for optimization time, this set of test points creates a huge area overhead and detects a very less number of faults in the design. These kinds of test points are considered inefficient test points. So we are characterizing and defining the most effective test points, these test points identify the more number of undetected faults and create a very minimal effect in the area overhead. During our proposed timing aware test point insertion time considers only the most effective test points into the design by taking into consideration of targeted coverage, pattern count and maximum affordable area overhead limits. At the same time, it removes all other ineffective TPIs from the final set of test points.

Control points in the design have the larger fanout cone means, it has high controllability on more pins and cells in the design which are hard to control without that test point. Similarly, an observable test point consisting of a huge fan-in cone and a greater number of hard-to-observe nodes or cells. So, for identifying the effective test point in the design, consideration of fan in and fanout cone sizes are not sufficient and we must need to consider how many non-

detected faults are covered for the particular cone in the design. from this analysis, a control test point and observe test points are selected based on the how many numbers of non-controlled and non-observable faults covered in the design. therefore, we need to consider the size of the logic cone as well as the number of non-controllable and non-observable faults with respect to each logic cone during the effective test point characteristics calculation.

As shown in figure 1, The entire proposed Timing aware Test point insertion mechanism is divided into 3 stages. In the first stage, the RTL design is synthesized and converted to Gate level Netlist along with the full scan insertion was performed. This scan inserted netlist given as an input to the two parallel operations, one is Test point insertion into the design and the other operation is ATPG pattern generation and fault simulation in order to analyze the overall pattern count, coverage and non-detected faults due to un-controllable and un-observable nodes In the design. In the second stage, we consider the list of tools inserted test point into account and will calculate each and every test point characteristic in order to identify the mostery test point characteristic in order to identifies the most

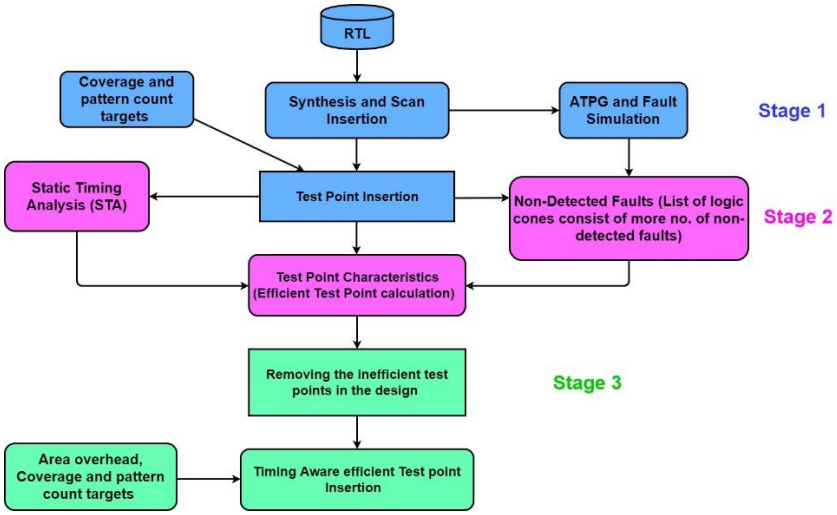


Fig. 19: Proposed Timing aware TPI flow diagram with different stages

effective test points in the design. Here the test point characteristics are calculated with the help of Static Timing Analysis (STA) and the non-detected faults identified from the ATPG fault simulations step. At the last stage of this proposed flow, divide the total test points which are inserted by the toll into two categories, one is the most efficient test point which has a high impact on the fault coverage and with not involved in high timing critical paths. The other category of test points are inefficient test points, it has more area overhead and very less impact on coverage and it targets the most critical timing paths, which leads to a high impact in area overhead during the synthesis post-optimization process. With the help of the above calculations, we remove the inefficient test point from the design and keep only the most effective test points in the final timing aware test point insertion mechanism. The impact of area overhead, coverages and the pattern count metrics were calculated by comparing the results without implementing the TPI, with TPI and with proposed TPI metrics.

3.1. Effective Test point calculation by using Static timing analysis

By using static timing analysis, we can observe different timing critical paths based on their path slack values. the difference between path delay and clock period is called slack of the node. always keeping the positive slack is a good option since it indicates that within the required time only the signal will reach the endpoint of the node. The negative slack paths also called failing paths have negative slack values and do not meet the timing requirements. A huge number of test points are inserted into the design along with timing critical paths If we are not considering these kinds of negative slack or timing critical paths into account.

During the synthesis process, the tool needs to work harder in order to meet the design constraints affected by these test points. Due to this a huge logic like gates, buffers, and registers are added while implementing the test points into the design. This will lead to a huge area overhead and extra time to test this logic and even there could be a chance of dropping the coverage also due to this extra logic.

By considering these challenges into account we proposed timing aware test point insertion flow with static timing analysis. In this approach, we have considered different control points based on the timing and number of non-detected faults covered, and with that, we have calculated the characteristics of the test point. so, we included the control point which has a high number of non-control faults and no timing critical paths, and some control points have timing critical paths but have more non-controllable faults. Similarly, few control points have no timing critical paths and not much impact on non-controllable faults. Few controls point will have a greater number of timing critical paths and a very less number of non-controllable faults. So,

the proposed algorithm will consider all these characteristics and calculates the efficient test points with a weighted factor based on slack values of timing critical paths.

$$ETP_{control} = \text{No. of non-control\_faults}_{fanout\_cone} \quad X \quad M_{Slack}$$

Where $ETP_{control}$ is the Efficient controllable test point and M_{slack} is the weighted multiplication factor, the value is depends on diffenrt slack values of the timing path. Similarly, the efficient observable test point can be characterized as

$$ETP_{observe} = \text{No. of non-observable\_faults}_{fanin\_cone} \quad X \quad M_{Slack}$$

Where $ETP_{observe}$ is the efficient observable test point

After calculating the characteristics of all the test points which is inserted by the tool, the most efficient test points are arranged in order to select the appropriate marginal number of efficient test points into the final design in order to meet the targets of vector volume, fault coverage and with minimal impact of the area overhead. As we analysed previously, the control and observation test points are affected by timing optimization. Since it must need to meet the design timing constraints provided during the initial phase. In the case of high critical timing paths having negative slack values needed a greater number of gates at the fan in and fan out paths, for such kind of timing critical paths have a smaller weighted factor.

4. Results and Analysis

In this Section, the performance of the proposed timing aware test point insertion by using a static timing analysis mechanism is verified on six industrial design blocks and using the 5-nm technology node. Each design block consists of the dedicated DFT and physical domain layout regions. The size of each individual design blocks w.r.t total gate count and the corresponding base ATPG outcomes in terms of pattern count and test coverages are reported in Table I. each design block consists of highly complex logic and a huge number (millions) of gates in the design. The ATPG tool generated a High volume of patterns dues to various advanced fault modes used in the highly complex designs in order to cover all the possible defects.

Here we used the most common and commercially available tools from Cadence (RTL Compiler) and Synopsys (Tetramax) for inserting the initial set of test points into the design. The performance of the proposed methodology was analyzed and compared on three metrics w.r.t pattern count, test coverage and area overhead with help of the conventional TPI approach. here the maximum number of controllable and observable test points is limited to 5% of the total scan flops in the design.

4.1. Impact on Area Overhead

When we are inserting the TPI by using commercial tools without prior knowledge of the critical timing paths and the tool will insert the test points randomly into the design. Many test points are inserted on the negative slack or timing critical paths. During Synthesis and timing optimization process the synthesis tool inserts a large number of gates, buffers, and registers into the logic cones in order to meet the design and timing-related constraints applied to the system. Due to that, we have seen much difference in the total area of the design after inserting the test points into the design. The % of Area overhead increased is 2.49%, 3.15%, 2.56%, 1.32%, 1.84% and 2.12% corresponding to design block1 to design block6 as shown in Table II under the conventional TPI area overhead column. The average increased area overhead among these 6 design blocks is 2.25%. when we are applying the proposed timing aware test point insertion mechanism to all the above-mentioned design blocks, we have seen a less area overhead around 0.48%, 0.85%, 0.4%, 0.3%, 0.62% and 0.59% respectively. So overall the average area overhead with the proposed approach is 0.54%. it shows that the proposed TPI approach provides a significant improvement of 75.5% in the total area overhead reduction when compared to the conventional TPI approach. in table III, % total area saved column shows the % area overhead saving for individual design blocks when applying the timing aware TPI into the design. The critical paths and timing slacks are analysed for all the design blocks by using static timing Analysis (STA). With that, the inefficient test points which give less impact on test coverage and huge impact area overhead are removed from the overall TPI list with the help of a timing aware TPI algorithm until it reaches desired test coverages and minimal area overhead targets.

4.2. Impact on Test Coverage

After analysing the overall test coverage results of each design block, we have seen the average increment in Stuck-At (SA) and Transition Delay Fault (TDF) coverage is 1.12% and 2.48% respectively when compared with the standard ATPG base results as shown in Table I and Table III.

Tab.1: Standard ATPG (Base) Characteristics of different design blocks.

design	Gate count (in millions)	Base (Standard ATPG)			
		Pattern count		Coverage	
		SA	TDF	SA	TDF

Design block 1	48	52K	189K	97.29%	89.46%
Design block 2	56	61K	201K	98.19%	92.81%
Design block 3	82	92K	356K	97.82%	91.57%
Design block 4	23	38K	134K	98.37%	91.67%
Design block 5	67	69K	252K	97.24%	93.47%
Design block 6	71	81K	298K	99.02%	92.43%
Average	**57.83**	**66.12K**	**238.54K**	**97.98%**	**91.90%**

After removing all the possible inefficient test points from the design by using the proposed timing aware TPI flow, there is a small reduction in the test coverage, but we have achieved a great impact in overall area overhead reduction. The conventical TPI approaches give 1.34% and 2.93% improvement in the SA and TDF coverages. So, the overall coverage impact is 0.2% and 0.5% w.r.t SA and TDF, but at the same time, we have achieved a 75% reduction in overall area overhead when compared with the conventional TPI approach. The conventional TPI approach takes 2.25% of overall area overhead whereas the proposed TPI takes 0.54%, it shows a significant improvement in overall area overhead. Always we can observe a trade-off between area overhead and coverage numbers in every design. The SA and TDF coverage numbers w.r.t Conventional TPI, as well as Proposed timing aware TPI, are provided in Table II and Table III respectively.

4.3. Impact on Test Vector count

The other important parameter metric is Test vector volume. Due to the advanced fault models, the vector count is increased to a great extent. It impacts the overall test time and cost of the design; it also affects the test coverage due to truncated patterns. The total vector count details with respect to standard base ATPG, conventional TPI and proposed timing-aware TPI are provided in Table II and Table III. While using the proposed TPI mechanism it is observed that there is a significant improvement in overall pattern count, 42% in SA and 48% in TDF patterns when compared with the standard base ATPG patterns. Since we are removing most of the inefficient test points in the proposed TPI

Tab.2: Conventional TPI Characteristics of different design blocks.

design	**Gate count (in millions)**	**Conventional TPI**				
		Pattern count		**Coverage**		**% Of TPI Area over head**
		SA	**TDF**	**SA**	**TDF**	
Design block 1	48	32K	83K	99.01%	92.7%	2.49%
Design block 2	56	34K	102K	99.45%	95.97%	3.15%
Design block 3	82	48K	171K	98.93%	95.03%	2.56%
Design block 4	23	21K	63K	99.82%	93.93%	1.32%
Design block 5	67	40K	138K	98.47%	96.19%	1.84%
Design block 6	71	34K	125K	99.59%	94.75%	2.12%
Average	**57.83**	**35.37K**	**113.97K**	**99.21%**	**94.76%**	**2.24%**

Tab.3: Proposed Timing Aware TPI Characteristics of different design blocks.

design	**Gate count (in millions)**	**Proposed Timing Aware TPI**					**% Of Total area Saved**
		Pattern count		**Coverage**		**%Area overhead**	
		SA	**TDF**	**SA**	**TDF**		

Design block 1	48	35K	92K	98.66%	92.34%	0.47%	81%
Design block 2	56	37K	114K	99.47%	95.74%	0.85%	73%
Design block 3	82	49K	174K	98.92%	94.49%	0.41%	84%
Design block 4	23	21K	68K	99.35%	93.62%	0.30%	78%
Design block 5	67	43K	146K	98.23%	95.63%	0.64%	65%
Design block 6	71	40K	134K	99.51%	94.48%	0.59%	72%
Average	**57.83**	**37.7K**	**121.7K**	**99.12%**	**94.38%**	**0.54%**	**75.5%**

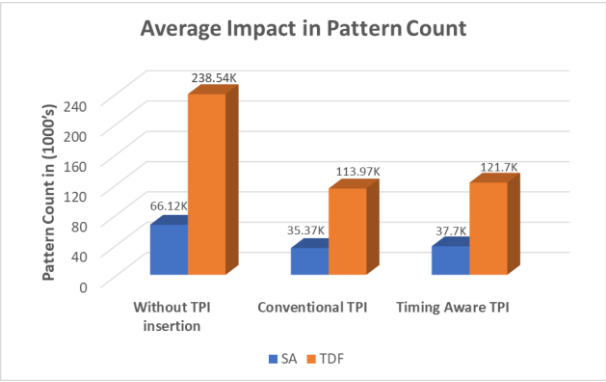
Tab.4: Average of all the design blocks TPI Characteristics.

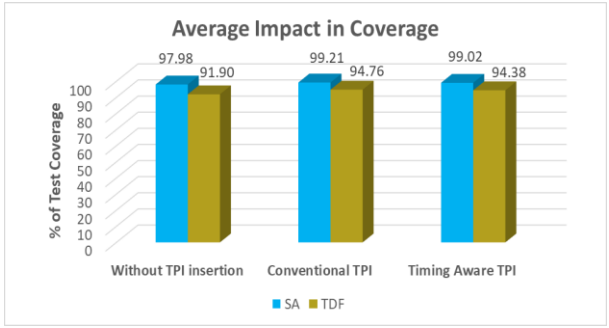
Fault Model Type	Average Coverage			Average Pattern Count Reduction		Average area overhead	
	Without TPI	Conventional TPI	Proposed TPI	Conventional TPI	Proposed TPI	Conventional TPI	Proposed TPI
SA	97.78	99.21	99.06	45.66	40.83	2.25%	0.54%
TDF	91.9	94.76	94.38	52.16	48.5		

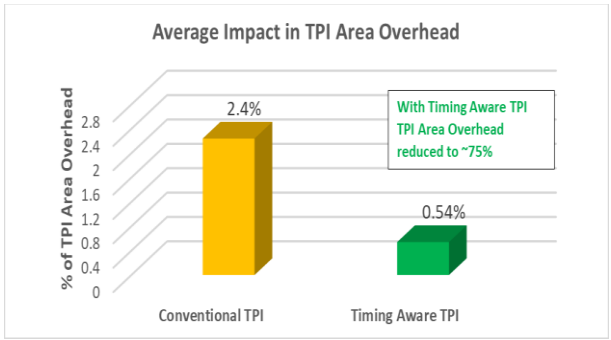
the approach in order to save the area overhead problems, at the same time there is a slight deviation in overall pattern count saving numbers between conventional and proposed approach. the conventional TPI approach gives 45% and 52% pattern count reduction w.r.t SA and TDF.

The Average values of all the design block results are calculated and kept in Table IV for the performance evaluations of the propose we discussed in previous subs ning aware TPI mechanism provides a significant improvement of an average of 75% in the TPI Area overhead reduction when compared with the conventional TPI approach. the conventional TPI approach gives 2.25% of extra area overhead, whereas in the case of the proposed TPI the average extra area overhead is only 0.54%. similarly, the average improvement is test coverage and Pattern count reduction values provided in table IV.

With Timing Aware TPI
SA Pattern count saving ~42%
TDF Pattern count Saving ~48%







5. Conclusion

Many conventional TPI techniques target the coverage and high pattern counts challenges without prior knowledge on timing critical paths in the design, which also leads to a significant area overhead problem. In This paper, we proposed a Novel test point insertion mechanism by using static timing analysis (STA) in order to target the huge pattern volume, low coverage and area overhead problems. The proposed TPI mechanism calculates the most effective test points by considering the timing critical paths and the

number of non-detected faults into account, at the same time the proposed algorithm discards the inefficient test points which lead to more area overhead and detects a smaller number of faults of the design. Selecting the appropriate marginal number of efficient test points into the final design in order to meet the targets of vector volume, and fault coverage with minimal impact on the area overhead. The experiments are performed and analysed on various industrial design blocks by using the proposed timing aware TPI approach and comparing the results with conventional TPI approaches. It is proven that the proposed Test Point Insertion approach gives a significant improvement in TPI area overhead reduction around 75% and the test vector count reduced to 48% with minimal coverage loss as low as 0.2%.

References

[1] F. Hapke et al., "Defect-Oriented Test: Effectiveness in High Volume Manufacturing," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 40, no. 3, pp. 584-597, March 2021, doi: 10.1109/TCAD.2020.3001259.

[2] A. Acharya et al., "Targeting Zero DPPM through Adoption of Advanced Fault Models and Unique Silicon Fall-out Analysis," 2021 IEEE International Test Conference India (ITC India), 2021, pp. 1-6, doi: 10.1109/ITCIndia52672.2021.9532687.

[3] F. Hapke, W. Redemund, A. Glowatz, J. Rajski, M. Reese, M. Hustava, M. Keim, J. Schloeffel, and A. Fast, “Cell-aware test,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 9, pp. 1396–1409, 2014.

[4] C. Acero *et al.*, “Embedded deterministic test points for compact cell-aware tests,” in *Proc. ITC*, 2015, paper 2.2.

[5] V. S. Iyengar and D. Brand, “Synthesis of pseudo-random pattern testable designs,” in Proc. ITC, 1989, pp. 501–508.

[6] C. Acero et al., "Embedded Deterministic Test Points," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 10, pp. 2949-2961, Oct. 2017, doi: 10.1109/TVLSI.2017.2717844.

[7] S. Eggersglüß, "Towards Complete Fault Coverage by Test Point Insertion using Optimization-SAT Techniques," 2019 IEEE International Test Conference in Asia (ITC-Asia), 2019, pp. 67-72, doi: 10.1109/ITC-Asia.2019.00025.

[8] G. Kim, M. Cheong and S. Kang, "SPAR: A New Test Point Insertion Using Shared Points for Area Overhead Reduction," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, doi: 10.1109/TCAD.2022.3147100.

[9] K. Padmapriya, B. K. S. V. L. Varaprasad and N. Varalakshmi, "A Survey on Test Point Insertion (TPI) Schemes," 2020 International Conference on Inventive Computation Technologies (ICICT), 2020, pp. 107-112, doi: 10.1109/ICICT48043.2020.9112405.

[10] Y. Sun, S. K. Millican and V. D. Agrawal, "Special Session: Survey of Test Point Insertion for Logic Built-in Self-test," 2020 IEEE 38th VLSI Test Symposium (VTS), 2020, pp. 1-6, doi: 10.1109/VTS48691.2020.9107584.

[11] V. Veena, E. Prabhu and N. mohan, "Improved Test Coverage by Observation Point Insertion for Fault Coverage Analysis," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019, pp. 174-178, doi: 10.1109/ICOEI.2019.8862789.

[12] J. Rajski, J. Tyszer and J. Zawada, "On New Class of Test Points and Their Applications," 2018 IEEE International Test Conference (ITC), 2018, pp. 1-9, doi: 10.1109/TEST.2018.8624900.

[13] M. He, G. K. Contreras, D. Tran, L. Winemberg and M. Tehranipoor, "Test-Point Insertion Efficiency Analysis for LBIST in High-Assurance Applications," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 9, pp. 2602-2615, Sept. 2017, doi: 10.1109/TVLSI.2017.2704104.

[14] M. T. He, G. K. Contreras, M. Tehranipoor, D. Tran and L. Winemberg, "Test-point insertion efficiency analysis for LBIST applications," 2016 IEEE 34th VLSI Test Symposium (VTS), 2016, pp. 1-6, doi: 10.1109/VTS.2016.7477314.

[15] H. Konuk, E. Moghaddam, N. Mukherjee, J. Rajski, D. Solanki, J. Tyszer, and J. Zawada, “Design for low test pattern counts,” *Proc. DAC*, 2015, paper 58.4.

[16] S. Remersaro, J. Rajski, T. Rinderknecht, S. M. Reddy, and I. Pomeranz, “ATPG heuristics dependent observation point insertion for enhanced compaction and data volume reduction,” in IEEE International Symp. on Defect and Fault Tolerance in VLSI Systems, 2008, pp. 385–393

[17] E. Moghaddam, N. Mukherjee, J. Rajski, J. Tyszer and J. Zawada, "Test point insertion in hybrid test compression/LBIST architectures," 2016 IEEE International Test Conference (ITC), 2016, pp. 1-10, doi: 10.1109/TEST.2016.7805826

[18] Y. Sun and S. Millican, "Test Point Insertion Using Artificial Neural Networks," 2019 IEEE Computer

Society Annual Symposium on VLSI (ISVLSI), 2019, pp. 253-258, doi: 10.1109/ISVLSI.2019.00054.

[19] S. Hellebrand, J. Rajski, S. Tarnick, S. Venkataraman, and B. Courtois, "Built-in test for circuits with scan based on reseeding of multiple polynomial linear feedback shift registers," IEEE Trans. Comput., vol. 44, no. 2, pp. 223–233, Feb. 1995.

[20] A. Kumar, J. Rajski, S. M. Reddy, and T. Rinderknecht, "On the generation of compact deterministic test sets for BIST ready designs," in Proc. ATS, 2013, pp. 201–206.

[21] Y. Liu, E. Moghaddam, N. Mukherjee, S. M. Reddy, J. Rajski, and J. Tyszer, "Minimal area test points for deterministic patterns," in Proc. ITC, Nov. 2016, paper 2.4.

[22] S. Eggersgl¨uß and R. Drechsler, "High Quality Test Pattern Generation and Boolean Satisfiability". Springer, 2012.

[23] R. Drechsler, S. Eggersgl¨uß, G. Fey, A. Glowatz, F. Hapke, J. Schloeffel, and D. Tille, "On acceleration of SAT-based ATPG for industrial designs," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 27, no. 7, pp. 1329–1333, 2008.

[24] S. Udar and D. Kagaris, "Minimizing observation points for fault location," in IEEE International Symp. on Defect and Fault Tolerance in VLSI Systems, 2009, pp. 263–267.

[25] J.-S. Yang, N. A. Touba, and B. Nadeau-Dostie, "Test point insertion with control points driven by existing functional flip-flops," *IEEE Trans. Comput.*, vol. 61, no. 10, pp. 1473–1483, Oct. 2012.

[26] H. Ren, M. Kusko, V. Kravets, and R. Yaari, "Low cost test point insertion without using extra registers for high performance design," in Proc. ITC, 2009, paper 12.2.

[27] R. Sethuram, S. Wang, S. T. Chakradhar, and M. L. Bushnell, "Zero cost test point insertion technique to reduce test set size and test generation time for structured ASICs," in Proc. ATS, 2006, pp. 339–348

[28] N. Tamarapalli and J. Rajski, "Constructive multi-phase test point insertion for scan-based BIST," in Proc. ITC, 1996, pp. 649–658.

[29] H.-C. Tsai, C.-J. Lin, S. Bhawmik, and K.-T. Cheng, "A hybrid algorithm for test point selection for scan-based BIST," in *Proc. DAC*, 1997, pp. 478–483.

[30] B. H. Seiss, P. Trouborst, and M. Schulz, "Test point insertion for scan based BIST," in *Proc. ETC*, 1991, pp. 253–262.

[31] K.-T. Cheng and C.-J. Lin, "Timing-driven test point insertion for full-scan and partial-scan BIST," Proc. ITC, 1995, pp. 506-514.

[32] M. Nakao, K. Hatayama, and I. Highasi, "Accelerated test points selection method for scan-based BIST," Proc. ATS, 1997, pp. 359-364

[33] C. Schotten and H. Meyr, "Test point insertion for an area efficient BIST," in International Test Conf., 1995, pp. 515–523.

[34] S. Udar and D. Kagaris, "Minimizing observation points for fault location," in IEEE International Symp. on Defect and Fault Tolerance in VLSI Systems, 2009, pp. 263–267.

[35] D. Erb, M. A. Kochte, S. Reimer, M. Sauer, H.-J. Wunderlich, and B. Becker, "Accurate QBF-based test pattern generation in presence of unknown values," IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, vol. 34, no. 12, pp. 2025–2038, 2015.

[36] A. Jas, C.V. Krishna, and N.A. Touba, "Weighted pseudorandom hybrid BIST," *IEEE Trans. VLSI*, vol. 12, pp. 1277-1283, 2004.

[37] F. Muradali, V. K. Agarwal, and B. Nadeau-Dostie, "A new procedure for weighted random built-in self-test," Proc. ITC, 1990, pp. 660-669.

[38] N.A. Touba and E.J. McCluskey, "Bit-fixing in pseudorandom sequences for scan BIST," IEEE Trans. CAD, vol. 20, pp. 545-555, 2001.

[39] A.-W. Hakmi, H.-J. Wunderlich, C.G. Zoellin, A. Glowatz, F. Hapke, J. Schloeffel, and L. Souef, "Programmable deterministic built-in self-test," Proc. ITC, 2007, paper 18.1.

[40] S.-J. Wang, K.-T. Yeh, and K. S.-M. Li, "Exploiting distribution of unknown values in test responses to optimize test output compactors," *Integration*, vol. 65, pp. 389–394, 2019.

[41] S. Hellebrand, H.-G. Liang, and H.-J. Wunderlich, "A mixed mode BIST scheme based on reseeding of folding counters," Proc. ITC, 2000, pp. 778-784.

[42] M. Syal, M. S. Hsiao, S. Natarajan, and S. Chakravarty, "Untestable multi-cycle path delay faults in industrial designs," in *14th Asian Test Symposium (ATS'05)*. IEEE, 2005, pp. 194–201.

[43] M. T. He, G. Contreras, M. Tehranipoor, D. Tran, and L. Winemberg, "Test point insertion efficiency analysis for LBIST applications," in *Proc. VLSI Test Symp. (VTS)*, Apr. 2016, pp. 1–6.

[44] Design Compiler User Guide. [Online]. Available: https://solvnet.synopsys.com

[45] M. G. Corporation, *Tessent® Scan and ATPG User's Manual*. Mentor Graphics Corporation, Siemens Industry Software Inc.

[46] TestMAX DFT User Guide. [Online]. Available: https://solvnet.synopsys.com

[47] TestMAX ATPG and Diagnosis User Guide. [Online]. Available: https://solvnet.synopsys.com

[48] N. A. Touba and E. J. McCluskey, "Test point insertion based on path tracing," in Proceedings of 14th VLSI Test Symposium. IEEE, 1996, pp.2–8.

[49] Aluka, M. ., Dixit, R. ., & Kumar, P. . (2023). Enhancing and Detecting the Lung Cancer using Deep Learning. International Journal on Recent and Innovation Trends in Computing and Communication, 11(3s), 127–134. https://doi.org/10.17762/ijritcc.v11i3s.6173

[50] Wanjiku , M., Levi, S., Silva, C., Ji-hoon, P., & Yamamoto, T. Exploring Feature Selection Methods in Support Vector Machines. Kuwait Journal of Machine Learning, 1(3). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/131

[51] Mr. Dharmesh Dhabliya, M. A. P. (2019). Threats, Solution and Benefits of Secure Shell. International Journal of Control and Automation, 12(6s), 30–35.