Hybrid Test Point Analysis, Insertion And Benchmarking For DfT Coverage Improvement

Vinay C Chowkimath[1], Ashwini V[2], Swamy Guthal[3]
[1, 2] Dept of Electronics and Communication
[1, 2] BMS College of Engineering,Bengaluru
[3]NXP Semiconductors India Pvt Ltd, Bengaluru

Abstract- *In the present era of very complex integrated circuits, DfT (Design for Testability) plays a very important role in ensuring the reliability of the integrated circuits. The reliability in the context of DfT is measured in terms of test coverage of the design. Due to the increasing complexity of the designs, more and more constraints will get added on the design which will pose challenges for DfT engineers to attain the higher test coverage of the design. As a result, the reliability of the integrated circuits cannot be ensured. In such scenarios, the test points will be used to increase the test coverage by making the undetectable / uncontrollable / unobservable faults as detectable or controllable or observable. The test points can be introduced for mainly two reasons i.e., either for improving the test coverage or for reducing the test patterns. This journal proposes a custom flow for introducing the test points on a design after knowing the test coverage without test points and targeting only the undetected faults for the test point analysis. The project uses a new type of test point analysis methods which is Versa point analysis which targets for both test coverage improvement and test pattern reduction. The benchmarking results of the project shows the advantages of using the Versa points over the traditional test points.*

Keywords- dft, test coverage, test point insertion, versa points

I. INTRODUCTION

In today's complex integrated circuit designs, ensuring the reliability of an integrated circuit is a crucial factor during the development phase of the integrated circuit. Formerly the functional vectors were being used for testing. But, as the complexity of the design increases, the functional vectors will increase exponentially. The testing of the integrated circuits using structural tests can also ensure the reliability of the circuit with some minimal test vectors. The technique of testing the circuits by the structural tests is called Design for Test techniques.

Design for Testability (DfT), is a VLSI technique introduced at the pre-silicon phase of the integrated circuit which ensures the reliability of an integrated circuit at the post-silicon phase by performing tests using DfT structures. The DfT structures include JTAG, Scan Chain, MBIST, LBIST, WBRs, OCCs, Test Points, etc,

Fault coverage is the DfT term which indicates how many faults are being detected among the total faults which may occur during fabrication process. Similarly, test coverage indicates how many faults are detected among the total number of detectable faults in the design. After introduction of standard DfT structures to the core, one can run the ATPG tool to generate test vectors and know the test coverage for the same.

A. Overview of the problem

The following figures depicts the problems incurred during the ATPG. Many nodes will be left unobserved or uncovered due to some constraints or unresolved logics around them.

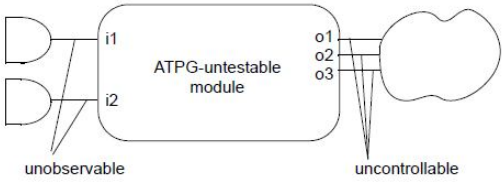

Fig 1.1: ATPG uncontrolled or unobserved module.

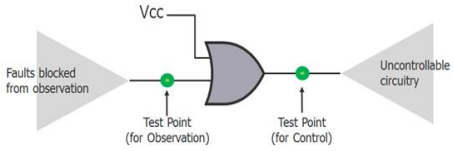

Fig 1.2: Circuits where test points are needed to be inserted.

Even after implementing several DfT structures in and around the functional logic core, due to the constraints on the design several nodes were left undetected. These nodes will be having lost either controllability or observability. Controllability of a node is the ability of the node to be excited to both high and low values by the top-level ports of the design. Observability of a node is the ability of the node that

its value is propagated to the top-level ports of the design. These uncontrolled or unobserved nodes are left uncovered or undetected by the ATPG which results in lower test coverage. For detecting such nodes, the controllability and observability at that node has to be improved by the addition of suitable test points.

Also, some nodes will be having large values of controllability and observability, so that faults at these nodes will be detected by unique test vectors or they need more effort of the ATPG engine to generate test vectors. This will result in increased test vectors or increased run time. Test points can also added at these nodes which are having highest controllability and observability values to reduce the number of deterministic patterns.

There are many test point insertion flows are followed for improving the test coverage. The journal focuses on a hybrid/custom flow. Because, in all the existing flows one has to decide on many factors for the test points at the very initial stage of the DfT insertion. Prior to knowing the test coverage without test points, one has to decide on the number of test points to be inserted on the design. Also the test point flops are added in a separate chain. To avoid the above scenarios and implement the test point insertion technique without any extra scan chains, the journal proposes a custom/hybrid flow for the test point analysis and insertion. And also the journal focuses on the use of versa points combined with observe point sharing technique which increases the effectiveness of test points on the design and reduces the area overhead incurred by the addition of test points.

II. PROPOSED FLOW

The journal proposes a custom or hybrid flow which emphasizes on inserting the test points on a design post ATPG (i.e., after knowing the actual test coverage and pattern count) without adding new chains to accommodate the test point flops. The flow is implemented by performing the test point analysis and insertion process based on the results of the ATPG process. Also, the analysis and insertion processes are performed in different EDA tools to ensure efficient test point insertion.

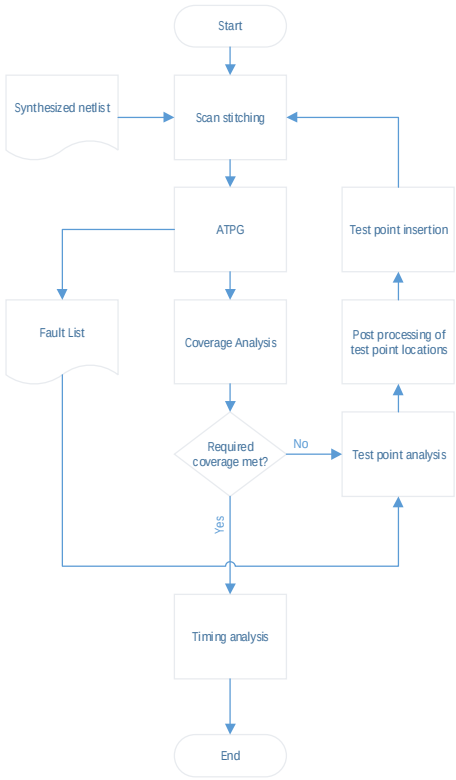

Fig 2.1: Proposed flow for efficient test point insertion for test coverage improvement.

Fig 2.1 shows the flow chart for the proposed hybrid test point analysis and insertion flow for test coverage improvement. The flow chart gives the overview of the algorithm used.

III. DESIGN AND IMPLEMENTATION

This section precisely describes the entire algorithm used for the insertion of test points for the DfT test coverage improvement. The hybrid test point analysis and insertion flow proposed in this project involves the following steps.

A. ATPG on the scan inserted netlist

The test points mainly targets the stuck at fault models. Hence test patterns are generated for the stuck-at fault type in both compression and bypass modes during the ATPG process. After the completion of the ATPG process the final fault list is dumped out. This fault list is written out to use it in the further process of coverage analysis and test point analysis.

B. Coverage analysis

Coverage analysis is performed on the fault list dumped out by the ATPG process. The fault list will be having

information of all the covered and uncovered faults of the design. It also gives the information of fault sub-class of every faults in the design. In this step, the root cause for the faults to remain uncovered is analyzed and the uncovered faults are segregated into separate bins. ATPG is re-run with some modified settings or test procedures to cover the remaining uncovered faults.

C. Test point analysis

After performing the coverage analysis, depending on the short comes of the ATPG process, the type of test point analysis is decided and the same is carried out to detect the optimal test point locations. To do test point analysis, the netlist of the design is read and analysis settings are made such as number of test points to be added, number of target patterns, type of the test point analysis, no-test blocks etc. The fault list from the recent ATPG configuration run is read to target the uncovered faults. The result of the test point analysis is written out to a file. This file will contain information of the test points to be inserted in the design to achieve the purpose of test point analysis performed. These test points should be inserted in the netlist and then the test point flops should be made scannable and then stitched inside the existing scan chains. Later the ATPG process is repeated on the test point inserted netlist to see the effects of test points on the final test coverage.

D. Post-processing of test point analysis results

In the proposed hybrid flow, test point analysis and test point insertion steps are performed in different tools. The test point analysis is performed on the design by the Tessent tool. The Tessent tool is selected for test point analysis as it is having efficient algorithms for analyzing the fault sites which cannot be controlled or observed and also, the tool has different algorithms for different test point usage purposes which is not supported by other tools. But the Tessent tool lacks a significant feature of updating the existing scan chains without compromising on the top-level ports and balance factor of scan chains. Hence, we cannot be able to stitch the inserted test point flops into the existing chains. Hence, we came up with a unique flow in this project. Here we take the test point locations suggested by the Tessent tool and then post process the output files from Tessent to convert the format of the test points as such they are in the RtlCompiler command format. These RtlCompiler commands for inserting the test points are written out to a tcl file.

E. Test point insertion.

Once the tcl file is ready with the test point insertion commands, test points can be inserted by sourcing the tcl file. For doing so, first the database of the RtlCompiler which was previously created when the scan insertion is performed is read to restore the same environment in the tool. Then the test point insertion commands are loaded by sourcing the tcl file generated by the script. After inserting the test points, one can easily hook them up in the existing scan chains with a single command, update_scan_chains. As the test points are inserted by the RtlCompiler tool itself, it supports for the updating of scan chains with the test point flops. In this way we can overcome the pitfalls of Tessent tool in updating the existing scan chains with the newly inserted test point flops.

Once after the test points are inserted into the design, the ATPG process is repeated to see the effects of test points on the design. This process can be done in an iterative way to reach the desired test coverage on the design by trying different number of test points. To ease up on the iterative process, this flow is also automated using the shell script.

IV. EXPERIMENTAL RESULTS

The proposed hybrid test point analysis and insertion flow is implemented on two designs and the results are obtained as expected. The results are shown with respect to these designs. The baseline numbers such as coverage, pattern count, area, etc., of both design A and design B that are compared with the results of the project are shown below.

Design A:

Test coverage	Pattern count	Flop count	Gate count	Area
96.63%	5230	29736	503042	57039514

Table 4.1: Design aspects of design A.

Design B:

Test coverage	Pattern count	Flop count	Gate count	Area
96.51%	1892	7515	124401	733954

Table 4.2: Design aspects of design A.

Based on the analysis of the results with respect to both the designs, few results are considered for benchmarking in different cases and the same are discussed in the below sections.

Case 1: The number of test points inserted on a design will affect the coverage and the pattern count. The same has been benchmarked as shown in the following figure.

Design A:

TP Type	TPs	CPs	Ops	Flops_Added	Coverage	Patterns	Area	% Area increase
LBIST	50	32	18	39	98.92%	5305	57072501	0.057
LBIST	100	58	42	74	98.96%	5223	57101547	0.108
LBIST	150	94	56	115	98.98%	5192	57137831	0.172

Table 4.3: Benchmarking results of incremental LBIST type of test points on design A.

Design B:

TP Type	TPs	CPs	Ops	Flops_Added	Coverage	Patterns	Area	% Area Increase
LBIST	50	36	14	46	97.63%	1927	735220	0.172
LBIST	100	62	38	81	97.97%	1989	736989	0.413
LBIST	150	92	58	116	97.98%	1936	738733	0.651
LBIST	200	112	88	143	97.99%	1869	740388	0.876
LBIST	250	138	112	174	98.22%	1826	742133	1.114

Table 4.4: Benchmarking results of incremental LBIST type of test points on design B.

In the above shown benchmarking results, it is observed that increasing the number of test points yields in more coverage improvement and pattern count reduction.

Case 2: The different type of test points are inserted on the designs with a fixed number of test points to compare the effect of individual types on coverage and test pattern number. The same is benchmarked in the below table.

Design A:

TP Type	TPs	CPs	Ops	Flops_Added	Coverage	Patterns	Area	% Area increase
LBIST	150	94	56	115	98.98%	5192	57137831	0.172
ATPG	150	75	75	150	98.12%	5159	57136164	0.169
VERSA	150	44	106	88	98.23%	3544	57124176	0.148
VERSA	200	57	143	116	98.62%	3297	57149298	0.192

Table 4.5: Benchmarking results of different test point types on coverage and number of patterns on design A.

Design B:

TP Type	TPs	CPs	Ops	Flops_Added	Coverage	Patterns	Area	% Area Increase
LBIST	125	75	50	97	97.98%	2003	737806	0.524
ATPG	125	53	72	82	98.09%	1901	737268	0.451
VERSA	125	17	108	51	98.01%	1882	736375	0.329
VERSA	200	29	171	81	98.05%	1839	738281	0.589

Table 4.6: Benchmarking results of different test point types on coverage and number of patterns on design B.

In the above case, it is observed that the Versa point type of test point analysis and insertion has produced more effective results when compared to the LBIST and ATPG type of analysis. The Versa points are the hybrid test points, which targets for both the test coverage improvement and also the pattern count reduction. The same can be observed in the above benchmarking results.

Case 4: The observe point sharing also is a factor to be considered which can affect the effect of test points. The following table shows the effect of observe point sharing on the coverage and number of patterns.

TP Type	TPs	CPs	Ops	Flops_Added	Coverage	Patterns	Area	% Area increase
LBIST	200	128	72	200	99.01%	5197	57176776	0.240643705
LBIST	200	131	69	157	98.97%	5186	57172120	0.232480943
VERSA	200	57	143	116	98.62%	3297	57149298	0.192470083

Table 4.7: Benchmarking results of observe point sharing effect on the test point area overhead.

In the above benchmarked results of case 4, one can notice that, application of observe point sharing technique can reduce the area overhead on the design due to the test points. The combination of versa point and observe point sharing technique will improve the effectiveness of test points on the design with minimal area overhead.

V. CONCLUSION

The journal mainly focused on a new customized flow of implementing the test points on a design and introduction of new type of test point analysis that is versa point analysis. By analyzing the experimental results, it can be concluded that the proposed hybrid flow implements the test points on any design without affecting the design aspects such as introducing new chains in the design, changing the balance factor of the chains, etc., The test coverage of the design can be increased by doing the analysis considering only the uncovered faults after the ATPG process. Also, the Versa point analysis is the most effective type of test point analysis which will improve the test coverage factor as well as reduce the test patterns. These versa points combined with observe point sharing technology can reduce the total area overhead of the test points on the design.

REFERENCES

[1] Elham Moghaddam, Nilanjan Mukherjee, Janusz Rajski, Jerzy Tyszer, Justyna Zawada “Test Point insertion in

Hybrid Test Compression / LBIST Architectures", INTERNATIONAL TEST CONFERENCE , IEEE, 2016.
[2] C. Acero et al., "Embedded deterministic test points",IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 25, No. 10, October 2017.
[3] L. H. Goldstein and E. L. Thigpen, "SCOAP: Sandia controllability/observability analysis program," in Proc. DAC, 1980, pp. 190–196.
[4] 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.
[5] M.J. Geuzebroek, J.Th. van der Linden and A.J. van de Goor. "Test point insertion for compact test sets". Proc. Of the IEEE Int. Test Conf, 292-301.2000.
[6] H. Vranken, S. S. Sapei, and H.-J. Wunderlich, "Impact of test point insertion on silicon area and timing during layout," in Proc. DATE, 2004, pp. 810–815.
[7] Tessent® Scan and ATPG User's Manual Software Version 2015.3 September 2015
[8] Tessent Shell User's Manual Software Version 2015.3 September 2015
[9] Tessent® TestKompress® User's Manual Software Version 2015.3 September 2015
[10] Tessent® Shell Reference Manual Software Version 2015.3 September 2015
[11] Design For Test in Encounter® RTL Compiler Product Version 14.2 February 2015
[12] Command Reference for Encounter® RTL Compiler Product Version 14.2 February 2015
[13] https://nww.wiki.nxp.com/
[14] https://nww.wiki.nxp.com/display/NXPDfT/NXP+DFT#NXPDFT-NXPDFTTechnicalReferences
l 2014.