| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 7.122|

||Volume 9, Issue 6, June 2020||

Compact Test Generation for Transition Faults

Amit Kumar

Cadence Design Systems, 2655 Seely Avenue, San Jose, CA, USA

ABSTRACT: Test generation procedures for large VLSI designs are required to achieve close to 100% fault coverage using a small number of tests.In this paper, we consider the generation of compact test sets for transition faults. We introduce new measures to guide automatic test generation procedures (ATPGs) to balance between these two contradictory requirements of fortuitous detection and number of specifications. One of the new measures is meant to facilitate detection of yet undetected faults, and the value of the measures is periodically updated. Proposed ATPG reduces external test set sizes by 32% in comparison to that obtained by a state of the art ATPG.

KEYWORDS: Design for Test, Compression, Automatic Test Pattern Generation, VLSI Testing

I. INTRODUCTION

This section contains an introduction to ATPG [1, 2, 3, 4, 5, 6]algorithms and heuristics, key definitions, how it works and the concepts around the ATPG. How the ATPG generates set tests, circuits involved and aspect of modelling faults for ATPG. The objectives of the paper have also been clearly outlined in this section. Automatic Test Pattern Generator (ATPG) is a method that generate test vectors stored in golden responses and ATE. This algorithm and heuristics automatically generate test sets that are capable of separating chips, both those that are faulty and those that are not. ATPG can operate on both circuits that have memory elements (and sequential) and those that do not have memory elements (combinational circuits). To generate test patterns by the ATPG in combinational circuits, the value input of the current is the only requirement, and this makes the process easier than in sequential. This is because in sequential, we need different setting of different elements in sequence to values that have been pre-determined value. This sequential circuit operation in generating tests is thus more complicated than the combining circuits.Scan inserting involves the conversion of combinational design from a sequential one. ATPG uses the scan inserting design to generate tests. Initializing a flip-flop requires the right sequence of initialization. Flip flops are connected in certain design forms to ensure they can be controlled or observed. The connection in shift register forms of flip flops ensures full control of the devices.

Modelling of faults is a critical aspect to aspect to help in detecting problems [1,2]. To simulate the effects of defects, models of logical fault are used. The challenge with fault model is that it cannot model same fault models accurately. Generation of test sets useful in detecting possible faults is achieved through the process of test generation. Fault detection is based on the modelling faults, either using same fault model or mix of faults from the combination of models. Patterns are set in the tester to help in detecting defects that occur in the flip flop during manufacturing. Fault propagation and fault activation are the two main steps of generating sets. Fault activation involves setting the signal of the line to a certain figure based on the fault model being used for transition fault model activation, one fault is given two distinct values to fault the system in two clock cycles consecutively. A value opposite the one given out by the fault is useful in exciting a fault, in stuck-at faults. With fault propagation, proper values are set along a chosen path so that sensitization happens on the path and fault effects can be identified and observed at designated positions.

The research paper aims at

- Proposing novel fault ordering approach. This will depend on fault location in FFR [5,6]. (Fan-out free region) for the given circuit being tested.
- Proposing a new ATPG approach in making testing point aware line and justification of decisions on D-propagation
- Provision of data (quantitative) for designs to be used in industry and in reducing compressed test patterns used in BIST designs available.

Through the research, new methods of fault testing and ATPG algorithms have been analysed. The data volumes will be decreased by 32%, the set size for data in the method will also reduce by the same percentage compared to the ATPG current commercial method.

II. BACKGROUND INFORMATION

The research paper is based on analysing the algorithmic techniques that were available for the ATPG. Literature on the different levels of testing and the background to the current study have been evaluated. All the background information to the study is found in this section. Algorithmic techniques used in minimising size test and experiments conducted have been analysed extensively.Algorithmic techniques can be used to acquire minimal size test. Extra hardware can also be used to achieve the same. Different methods that have been discussed on small test generation in different papers have also been analysed. In normal cases, dynamic or static compaction test procedures, both dynamic and static compaction test protocols are used in analysis [7, 8, 9, 10, 11]. The procedures in static compaction are used in detecting individual faults through merging compatible test cubes generation. It involves use of fault simulation tests in distinct orders different from the ones the tests were generated and dropping off some tests within the set of tests. The tests are dropped ensuring that fault coverage is not lost. The procedures and protocols in dynamic compaction utilize values in test cubes that are unspecified in detection of additional defects. The generation of tests with least numbers of specified values assist in decreasing the number of tests generated using static and dynamic compaction protocols. Testing with the least number of specified values facilitate encoding test tubes for designs with test compression logic. This aid in generating compact test sets in the design [1, 12, 13, 14, 15].

A previous research and experiment were conducted on justification of particular circuit node values in generating some test to in detection of targeted faults. This was used in designing logic compression for on-chip test data. The results of this experiment proved that sizes decrease by 5% averagely on test set from the described method. Before this, it was clear that for generation of compact test sets, while ensuring the generating test identify faults that had been targeted, it was important to ensure facilitation of undetected faults. Rotating back trace was useful in achievement of this. In bid to change the ability to control and observe measures useful in guiding line justification alongside fault propagation, the steps of testing procedures were adhered to. The particular ones, compact controllability and Compact observability figures of lines of the circuit were used to achieve the experiment. The latter measures were acquired through modification of the famous SCOAP [16, 17, 18] values (observability and controllability values). These values (observability and controllability values) are computed for each line 1 circuit afresh each time before each new test is generated and the guidelines are aimed at facilitation of detection of the faults that are yet to be detected on line 1. In the process of derivation of the C- controllability and C-observability for the line 1, these guidelines are changed to ensure there is inclusion of fault that have not been detected on that same line 1 alone. From the experimental data, it can be shown that utilisation of the SCOAP (C-controllability and C-observability), the sizes of the tests are decreased by about 6% averagely. Other methods used in derivation of compact test don’t alter the generation procedures of testing discussed ere. An example in place is the definition of group of faults that posses’ conditions that may be compatible and important in detection. Through a single test, there is a test generated to identify every cluster of faults.

III. PROPOSED WORK

The section contains the proposed work to be done in order to come up with findings in the area of research. The section has different procedures that have to adhere to and followed in achieving the objectives of the paper. It contains fan-out free region-based fault detection method proposed. The description of Automatic Test Pattern Generator measures and guidelines [19] in usage as well as ways of line justification alongside decisions of D-propagation decisions.

In this paper, consideration is made in test generation for a stuck-at faults in scan design through use of D-algorithm. To generate a test for a stuck-at fault, may be for line r stuck at-a. a=0 or 1; this starts through implication of a value a^{-} on the given line r, whereby a^{-} is obtained as the complement of the. This leads to creation of a D or D^{-} on line t utilising the notation introduced before that is D. What follows is the justification through a value on line r via assignment of appropriate values. This assignment is to the input(s) of the gate that drive line r. Then D^{-} or D found on line t is propagated.

This propagation is to the outputs that have been observed by selecting intervening gates. D-frontier and J-frontier which form two data structures facilitates the line justification alongside the propagation of Ds. For entries in the J-frontier, they form an output lines of gates, these gates outputs have given values that have not yet been implied through the values of the gates’ inputs. For the entries in the D-frontier, there are gates whose input(s) have D or D^{-} values with their outputs not being specified. To justify the needed value on the line t in the J-frontier, assignment of proper values is done to one of the inputs of the gate that drive r.

Propagation of D or D^{-} to an observed output is done through selection of a gate amidst the D-frontier and the propagation of D^{-} or D from its input(s) to its output. Selecting a proper input for assigning in justification of a line value in the J-frontier and selection of a proper gate from the D-frontier in propagation of D or D^{-} is critical in obtaining tests. These are done with least number of specified values, tin obtaining least size test sets and reducing times of test generation. Typical methods to select proper inputs for line justification and D propagation use controllability and observability measures or select randomly with equal probability one of the unspecified inputs of gates driving lines on the J-frontier and for D propagation select randomly with equal probability one of the gates from the D-frontier. Even though line justification and D propagation decisions based on controllability

3.1 Transition Fault ATPG

Test generation of transition faults requires initialization of the fault site in the first-timeframe and activating a transition at the fault site and propagating the fault effect to PO/PPO's for observation in the second time frame. The generation of tests for a transition fault slow-to-rise (fall) start with justifying a value 1(0) at the fault site. A value 1(0) is added to the J-frontier for slow-to-fall (rise) faults for the first frame for initialization of the fault. Next, the value on the line is justified by assigning appropriate values to the input(s) of the gate driving the faulty line.Next, in the second time frame depending on the transition fault type a transition needs to be activated and propagated. In other words, for the second time frame- for slow-to-fall (rise) fault a 0(1) needs to be assigned to the fault site and the fault effect (transition) needs to be propagated. During the second time frame the problem translates to the detection of a stuck-at-1(0) fault in the second time frame. In other words, in the second time frame, a for slow-to-fall (rise) fault, a 0(1) value needs to be justified at the fault site (added to the J-frontier) and a D^{-} (D) needs to be added to D-frontier to propagate the fault effect. During the test generation, any values in the second time frame are satisfied from PPIs/PIs in the first-time frame. Primary inputs cannot be specified in the second time frame and primary outputs cannot be observed in the first-time frame, when launch-on-capture testing schemes are used. The entries in the D-frontier are gates for which some input(s) have D or D^{-} values and the outputs are unspecified. Justifying the required value on a line in the J-frontier is done by assigning the proper value to one of the inputs of the gate driving in the frame under consideration.

All propagation decisions take place in the second (propagation) time frame. Propagating D or D^{-} to an observed output in the frame and selecting a proper gate from the D-frontier to propagate D or D^{-} in the propagation frame is important to obtain tests with a minimum number of specified values, to obtain minimal size test sets, and to reduce test generation times. The ordering of faults also has an impact on the overall test length. One would think that targeting faults that help in the generation of tests for other undetected faults first is a good idea. The ordering should help with the following: Fortuitous Detection: Faults should be targeted in an order to increase fortuitous detection. Tests can be generated for large numbers of other, yet to be detected faults while generating a test for originally targeted fault. Hard to Detect Faults: A few faults are hard to detect; typically, a test generated for a hard-to-detect fault does not result in fortuitous detections. A hard-to- detect fault, if targeted in the end, results in low fortuitous detections and typically increases pattern count. A balance needs to be created to target faults in such a way that large numbers of faults are detected fortuitously, while keeping pattern counts in reasonable limits. In the next sub-section, we propose a new fault ordering mechanism that balances between targeting faults with high fortuitous detection and hard to detect faults.

3.2 Fault Ordering Based on Test Flow

A new fault ordering mechanism was developed, but before that two new measures were proposed: Test Flow Measures for Propagation (TFP) and Test Flow Measure for Justification (TFJ) to help classify faults in to one, having a high fortuitous detection or hard-to-detect fault. Consider an example of a three input AND gate with inputs A, B, C, output D, and stuck at fault model. It is important to note that all stuck-at-1 or stuck-at-0 faults on A, B, and C, respectively, can be initialized at the same time. However, for launching the final transition value at the fault site and propagating the fault effect, it is different. All slow-to-rise faults can be tested simultaneously, as each of them require the same inputs (1, 1, and 1) on (A, B, C), respectively. However, for slow-to-fall faults on (A, B, C), we require three vectors (0,1, 1), (1,0,1), and (0,1,1), respectively. Note that initialization and final launching of the transition during the propagation phase are comparatively different as, during initialization phase, the fault location only needs to initialize. All the values in the propagation frame are controlled by initialization frames it is much easier to justify a value in the initialization frame; hence, we will consider only propagation frame. Two test flow measures are used, that estimates number of faults whose detection is facilitated by a decision, these measures are named Test Flow Measures (TF measures)

```
Algorithm 1: Fault Ordering
  Input: Set TFJ1 TFJ0 to 0 for all lines
  Result: LISTf is the final fault order targeted for test generation by
          ATPG
  for every net in the design do
      Total(TET), which is measure of fortutious detection possible when
        a test for a given fault is generated For every line i we have two
        total test flow measures TFTi1(0) which is sum of test flow
        measure for propogation TFPi1(0) multiplied by test flow measure
        for justification TFJi1(0), respectively and store it in a list.
      Sort LISTA in descending order according to Total Test
        Enumeration Measures
  end
  //Detection of hard to detect fault and fortutious detection is balanced
    in next step.
  Make a final list of faults Listf , by selecting a fault from beginning of
    ListA and moving it from LISTA to LISTf and then a fault from the
    end of ListA and moving it from ListA to LISTf.
  LISTf is the final fault order targeted for test generation by ATPG
```

Figure 1. Fault Ordering Procedure

3.3 TF Measures for Propagation

Test flow measures for propagation (TFP) measures are of two types: TFP measures to estimate the number of times a justification is required at a line during the propagation frame, and TFP measures for propagation estimates of the facilitation of propagation of transition to observed PO/PPO. On the lines of SCOAP observability measures, we calculate TF measures for propagation (TFP). We use TFJ values to calculate TFP measures; however, TFP values are dependent on the polarity of fault-free values after transition, currently being propagated. Hence, two TFP values are assigned to each line corresponding to 1 and 0, respectively. Figure 2 shows the procedure to calculate TFP measures.

3.3 TF Measures for Justification

Any justification decision in the propagation frame should facilitate placement of the final transition on other fault sites; we estimate this by TFJ measures- each line *l* is assigned two TFJ measures for both 0 and 1, respectively. ATPG is guided more often toward options which facilitate the placement of final transition value 1 for slow-to-rise and 0 for slow-to-fall transition faults. For example, for a 3 input AND gate, TFJ0 will have a value of 3 and TFJ1 will have a value of 1 as slow-to-rise faults on inputs need 1 pattern to test and to create a transition (0s can be initialized simultaneously), but three patterns to test slow-to-fall faults as each slow-to-fall faults on input need a separate pattern. The Procedure to calculate TFJ measures is illustrated in Figure 3.

3.5 Generation of Final Fault List

Total Test Flow Measures (TFT) are used to get the total of fortuitous detection capabilities of a fault. TFT combines estimates of fortuitous detection of a fault during propagation and justification and hard-to-detect faults that would require a separate pattern. The procedure to generate the final fault list targeted by ATPG is given in Figure 3.3.

```
Algorithm 2: Calulation of Flow Measures for Propogation
Input: Set TFP1 TFP0 to 0 for all lines
Result: Final TFP1 and TFP0 for all nets in the design
for All gates in levelized manner from PO/PPO to PI/PPI do
    if Gate is PO or PPO then
        Assign TFP1 =0 and TFP0 =0
    else
        for Input J of gate I do
            for $v \epsilon \{0,1\}$ do
                if If value v for j uniquely determines value for i then
                    if $v = 1$ then
                        $TFP_{J1} = TFP_{I1} + TFJ_{J1}$
                    else
                        $TFP_{J0} = TFP_{I1} + TFJ_{J1}$
                    end
                else
                    for every input other than j required to propagate the
                      value of input j to i store the input in an array L for
                      every line l in array L do
                        if $v = 1$ then
                            $TFP_{J1} = TFP_{J1} + TFJ_{l1} + TFJ_{l0}$
                        else
                            $TFP_{J0} = TFP_{J0} + TFJ_{l1} + TFJ_{l0}$end
                        end
                    end
                end
            end
            If stem add all values of all branches
        end
    end
```

Figure 2. Procedure to generate propagation measures

3.6 ATPG Decisions

ATPG decisions are made in a similar fashion, as the previously- discussed ATPG technique [5]. If the gate is marked, static measure is used for making ATPG decisions, or else appropriate test flow measures are used instead of dynamic measures, keeping the time-frame into consideration. We use a propagation-first ATPG approach to generate tests. The advantages of using a propagation-first ATPG approach for transition faults are:While generating compact tests for circuits with compression the number of specified positions is important. Most of the specified positions are resultant of the creation of a propagation path from fault location to the PO/PPO.

1. Only the initialization time-frame is directly controllable from the inputs. Propagation time frame is controlled by tracing all the values from the propagation frame to the initialization frame PO/PPO. Hence, the impact of a propagation decision is more pronounced.
2. In case of stuck-at faults propagation, first approach is known to work better than justification-first approach.

3.3 Propagation Decisions

Propagation decisions are made in propagation frame during test generation for transition faults. Initialization of a fault is comparatively easier than creating transitions at the fault location and propagating it to the PO/PPO. Fault detection for transition faults can be sub-divided into two parts:

- Initialization of fault location (line l) to 0 (1) for slow to rise (fall) fault in initialization frame.
- Detection of a stuck-at-0(1) fault on line l (fault location) in propagation time frame.

Since any line l is controllable only from initialization frame, the second step of detection of stuck-at faults in the propagation

```
Algorithm 3: Test Flow Measures for Justification
Input: Set TFJ1 TFJ0 to 0 for all lines
Result: TFJ1 TFJ0 for all lines
for every net in the design do
    if Single input gate then
        if inverted gate then
            $TFJ0_{op} = TFJ1_{op}$
            $TFJ1_{op} = TFJ0_{op}$
        else
            $TFJ1_{op} = TFJ0_{op}$
            $TFJ0_{op} = TFJ1_{op}$
        end
    else
        Tc is maximum value for any TFJ for controlling inputs
        Tt is sum of all TFJ non-controlling values
        Always one value 1 or 0 is a result of controlling inputs at its
         inputs
        if Output value k due to controlling value then
            TFJk = Tc
        else
            TFJk' = Tt
            k and k' are inversions of each other and are either 0 or 1
        end
    end
end
```

Figure 3. Procedure to generate justification measures

Frame is comparatively difficult to achieve and results in a higher number of specified scan-cells (during the initialization frame). Therefore, to generate compact tests it is important to make prorogation decisions in a manner that facilitates the detection of additional stuck-at-0(1) (for slow-to-rise (fall)) faults in the propagation frame to enhance fortuitous detection. Simultaneously, ATPG decisions should try to create a balance between fortuitous detection and the number of specified positions. Dynamic and static guidance measures proposed in [5] using a probabilistic marking scheme and creates a balance between fortuitous detection and the number of specified positions. We use dynamic and static ATPG guidance measures described above to make propagation decisions as:

- Dynamic measures for propagation facilitate the second step in transition fault tests, the detection of a stuck-at fault in propagation frame. ATPG decisions are made in a manner that the second step is facilitated for many faults.
- Step1 of activating the transition fault is much easier, as all values need to be justified to initialization-frame scan cells.

3.4 Justification Decisions

Justification decisions are made in initialization and propagation frames. In propagation frames, we make justifications decisions based on dynamic guidance measures; this creates a balance between fortuitous detection and the number of specified scan-cells. We analyse the observability of a fault being propagated forward. If the fan-in of the gate on the D-frontier has a fault-free value which is also the controlling value for the gate on the D-frontier, no fault whose D-frontier reaches the off-path inputs can be detected. D-frontier for all such faults is killed by the presence of the controlling value. Hence static measures, which result in the reduction of the number of specified positions, are used for ATPG guidance.

IV. EXPERIMENTAL RESULTS

This section contains the results of all experiments conducted within the paper to analyse different contents and justify the proposals and objectives. The results for the transition fault (launch-on- capture [2]) ATPG discussed in this paper are analysed and discussed. In Table 1, the results of test generation for transition fault model are compared using random decisions, with proposedTransition fault ATPG, the Method-1 ATPG, and Method-2 ATPG for line justification and D-propagation. After the circuit name, we give the fault coverage obtained and test lengths using random decisions, followed by the values when the proposed transition fault ATPG is given. In the next column, we give the percentage reduction in test length obtained by the new transition fault ATPG. Test lengths obtained by Method-1 ATPG [5], followed by percentage reduction and Method-2 ATPG [6] are also given. New transition fault ATPG reduces test length by 32% in comparison to random decision ATPG; only 18% and 22% reduction is achieved by Method-1 and Method-2 ATPG. Figure 4 illustrates step counts normalized in comparison to random ATPG.

Later we study individual components of the proposed transition fault ATPG and their impact on pattern count reduction; we individually study the impact of dynamic and static measures as well as fault ordering on the final pattern count. We found out that pattern count reduction reduces to 12% if only dynamic measure was used for ATPG guidance; no fault ordering was used. Pattern count reduces by 18% if only static measure was used. Impact of fault ordering is given in Figure 5; justification and propagation decisions were made randomly in this case. Fault ordering reduces pattern count by 5.1%.

Table 1: Results for transition fault ATPG

Ckt	RandomTL	Proposed Trans	%Red	[5] TL	%Red	[6] TL	%Red
D1	23418	19129	18.31	21290	9.09	22067	5.77
D2	19792	16479	16.74	17239	12.90	16380	17.24
D3	25912	15023	42.02	15620	39.72	16023	38.16
D4	3283	1632	50.29	2006	38.90	2068	37.01
D5	22239	12572	43.47	19211	13.62	16329	26.57
D6	18934	13956	26.29	16024	15.37	17021	10.10
D7	9025	4910	45.60	5910	34.52	6921	23.31
D8	14916	9029	39.47	12031	19.34	9982	33.08
D9	21329	13911	34.78	20215	5.22	16021	24.89
Avg.	158848	106641	32.87	129546	18.45	122812	22.69

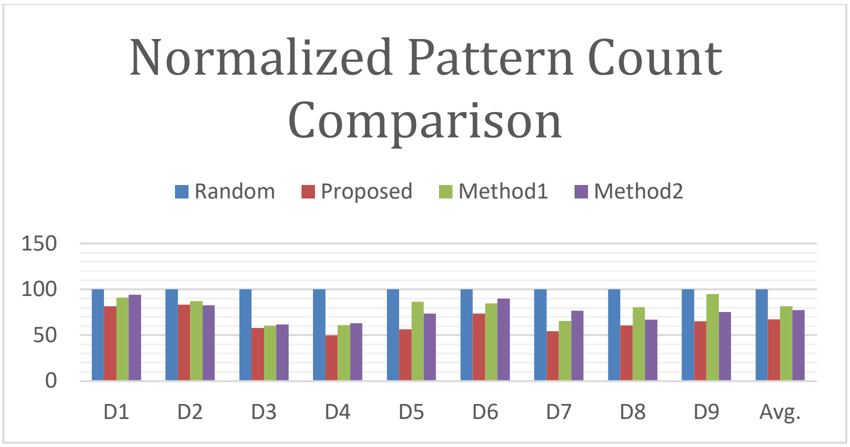

Figure 4: Normalized Test Length

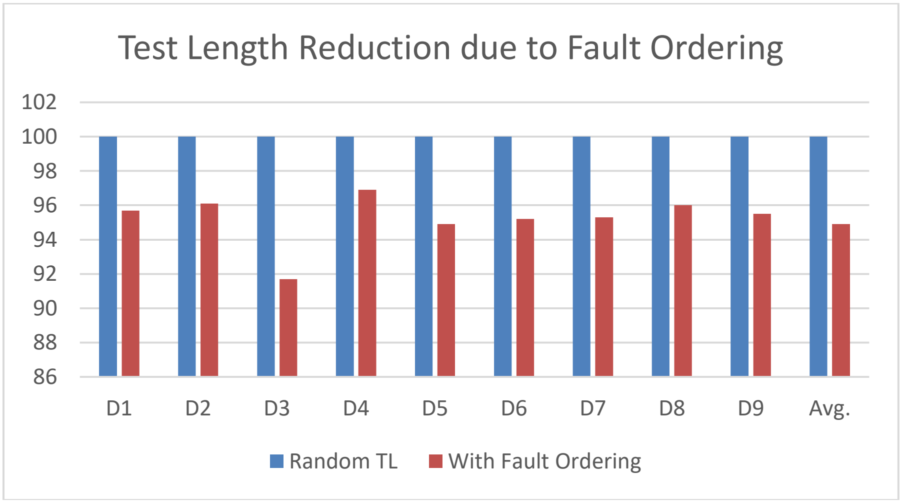

Figure 5: Impact of fault ordering on test length

V. CONCLUSION

Transition faults are difficult to test and often result in longer test lengths, we propose a new fault ordering technique based on test enumeration, this ordering technique and a new guidance approach was also proposed for transition faults. Test set sizes were reduced significantly for both stuck-at and transition fault models. Methods proposed in this work lead to 32% reduction in test length.

REFERENCES

[1] A. Kumar *et al.*, "Isometric Test Data Compression," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 11, pp. 1847-1859, Nov. 2015, doi: 10.1109/TCAD.2015.2432133.

[2] F. Hapke, R. Krenz-Baath, A. Glowatz, J. Schloeffel, H. Hashempour, S. Eichenberger, C. Hora, and D. Adolfsson, “Defect-oriented cell-aware atpg and fault simulation for industrial cell libraries and designs,” in International Test Conference, pp. 1–10, 2009.

[3] I. Pomeranz, L. Reddy, and S. Reddy, “Compactest: a method to generate compact test sets for combinational circuits,” IEEE Trans. on CAD, vol. 12, pp. 1040 –1049, jul 1993.

[4] S. Kajihara, I. Pomeranz, K. Kinoshita, and S. Reddy, “Cost-effective generation of minimal test sets for stuck-at faults in combinational logic circuits,” IEEE Transactions on CAD, vol. 14, pp. 1496 –1504, dec 1995

[5] A. Kumar, J. Rajski, S. M. Reddy, and C. Wang, “On the generation of compact test sets,” in ITC, 2013.

[6] A. Kumar, J. Rajski, S. M. Reddy, and T. Rinderknecht, “On the generation of compact deterministic test sets for bist ready designs,” in ATS, 2013.

[7] B. Ayari and B. Kaminska, "A new dynamic test vector compaction for automatic test pattern generation," IEEE Trans. on CAD, vol. 13, pp. 353 –358, mar 1994.

[8] A. Kumar *et al.*, "Isometric Test Data Compression," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 11, pp. 1847-1859, Nov. 2015, doi: 10.1109/TCAD.2015.2432133.

[9] I. Hamzaoglu and J. Patel, "Test set compaction algorithms for combinational circuits," in Procs. of ICCAD, pp. 283 – 289, 1998

[10] M. Konijnenburg, J. van der Linden, and A. van de Goor, "Compact test sets for industrial circuits," pp. 358 –366, 1995.

[11] S. Remersaro, J. Rajski, S. Reddy, and I. Pomeranz, "A scalable method for the generation of small test sets," in Proc. of DATE, pp. 1136 –1141, 2009.

[12] P. Wohl, J. Waicukauski, and T. Finklea, "Increasing prpg-based compression by delayed justification," in Proc of ITC, pp. 1 –10, nov. 2010.

[13] H. Fujiwara, "Fan: A fanout-oriented test pattern generation algorithm," in ISCAS, 1985.

[14] M. Schulz, E. Trischler, and T. Sarfert, "Socrates: a highly efficient automatic test pattern generation system," IEEE Transactions on CAD, vol. 7, no. 1, pp. 126–137, 1988.

[15] W. Kunz and D. Pradhan, "Recursive learning: a new implication technique for efficient solutions to cad problems-test, verification, and optimization," IEEE Transactions on CAD, vol. 13, no. 9, pp. 1143–1158, 1994.

[16] L. Reddy, I. Pomeranz, and S. Reddy, "Rotco: a reverse order test compaction technique," in Euro ASIC, pp. 189–194, 1992.

[17] A. Kumar, M. Kassab, E. Moghadham, N. Mukherjee, J. Tyszer, J. Rajski, S. M. Reddy, and C. Wang, "Isometric test compression with low toggling activity," in ITC, 2014.

[18] A. Kumar, S. M. Reddy, I. Pomeranz and B. Becker, "Hyper-graph-based partitioning to reduce DFT cost for pre-bond 3D-IC testing," *2011 Design, Automation & Test in Europe*, Grenoble, 2011, pp. 1-6, doi: 10.1109/DATE.2011.5763230.

[19] M. Sauer, S. Reimer, I. Polian, T. Schubert, and B. Becker, "Provably optimal test cube generation using quantified boolean formula solving," in ASP-DAC, pp. 533–539, 2013.