Changing the Scan Enable during Shift

*Nodari Sitchinava** *Samitha Samaranayake*** *Rohit Kapur**
*Emil Gizdarski** *Fredric Neuveux** *T. W. Williams**

** Synopsys Inc., 700 East Middlefield Road, Mountain View, CA 94043*
*** Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139*

Abstract

This paper extends the Reconfigurable Shared Scan-in architecture (RSSA) to provide additional ability to change values on the scan configuration signals (scan enable signals) during the scan operation on a per-shift basis. We show that the extra flexibility of reconfiguring the scan chains every shift cycle reduces the number of different configurations required by RSSA while keeping test coverage the same. In addition a simpler analysis can be used to construct the scan chains.

This is the first paper of its kind that treats the Scan Enable signal as a test data signal during the scan operation of a test pattern. Results are presented on some ISCAS as well as industrial circuits.

1. Introduction

The recent focus on lowering the number of bits stored on the ATE for deterministic test (test-data-volume, TDV) and lowering the time it takes to apply the test patterns (test-application-time, TAT) has spurred a lot of research [1-8].

Scan chains, the very technology that enables ATPG as the bread and butter DFT method [11] is seen to pose a new challenge. As designs have become more complex, the number of flip-flops that need to be scanned are too many. The stimulus and observe values of test patterns (TDV) are dominated by the values related to the scan chains. With relatively few inputs and outputs of the design that can be used as terminals for the scan chains the number of flip-flops per scan chain has increased dramatically. As a result the time required to operate the scan chains, or the TAT, has increased. To gain an appreciation of the impact of shift time on TAT, consider the typical sequence involved in processing a single scan test pattern:

1. Set up the scan chain configuration.
2. Shift values into the active scan chains.
3. Exit the scan configuration.
4. Apply stimulus to the inputs and measure the outputs.
5. Pulse clocks to capture the test circuit response in Flip-Flops.
6. Set up the scan chain configuration.
7. Shift values out of the active scan chains.
8. Exit the scan configuration.

All of these steps — excluding the shift operations in steps 2 and 7 — take one clock period on the tester. The shift operations, however, take as many clock periods as the longest scan chain. Optimizations (such as overlapping of scan operations of adjacent test patterns) do not change the fact that the test application time is dominated by the scan operation.

It is a commonly known fact that fault detection requires only a small percent of the stimulus and measure points of the inputs, outputs and scan-elements in the design be accessed for test [8]. Prior to the focus on TDV and TAT, the typical practice in the industry was to fill all remaining stimulus points of the pattern (logic X's) with random values. The new methods being developed involve creative ways of treating the randomly filled bits for gains in TDV and TAT. Dynamic Scan uses reconfigurable technology to move scan segments into and out of the active scan chains to bypass the X's of the tests [8]. Illinois Scan uses common scan-ins to allow for the logic-X's of the scan cells to be filled with the same values as those in other scan cells [3, 6]. Another method uses intermediate states of an LFSR coupled with values from an ATE to replace the logic X's [7]. The differences in these technologies show up in the following areas:

- *Area overhead*: Illinois-Scan and Dynamic-Scan have lower hardware overhead than LFSR based methods.
- *Test Application Time and Test Data Volume*: The general rule of thumb is that the more DFT technology is used, the more gains one is expected to achieve. Each method creates a solution with a different trade-off point between area overhead and possible TAT and TDV gains. One needs to understand how much of DFT area overhead one can afford, how much of gains one needs to achieve to select the solution.
- *Complexity of the encoding*: Complexity plays a role when the user (who is not familiar with the IP) needs to utilize the infrastructure for debug. The methods that rely on simpler encoding methods are typically easier to deal with.

This paper extends the low overhead Reconfigurable Shared Scan-in architecture (RSSA) introduced in [9] to provide additional ability to change values on the scan configuration signals (scan enable signals) during the scan operation on a *per-shift* basis. The extra flexibility of

reconfiguring the scan chain every shift cycle reduces the constraints introduced by sharing scan-in pins among several chains and, consequently, the number of different configurations required by RSSA.

The next section describes some concepts behind the common scan-in architecture. In Section 3 the new architecture is described with focus on the ability to change the scan enable signal during the scan operation. Section 4 describes the DFT synthesis procedures which determine the construction of scan chains and the assignment of chains to the scan-in pins. Finally, Section 5 presents some results highlighting the benefits of treating the scan enable as a test data input rather than a test control signal.

2. Illinois Scan

Traditional scan chains have a unique scan-input per scan segment. Thus, the length of scan chains, in a balanced scan architecture, is equal to the number of scan flip-flops divided by the number of scan inputs. Figure 1 shows a common scan-in architecture (Illinois Scan) as it relates to traditional scan chains [3, 6]. In Illinois Scan the scan-inputs of a chain are tied together to allow for many short scan chains to be accessed through the same number of scan-inputs as conventional scan architectures. On the output side a MISR is used to compact responses.

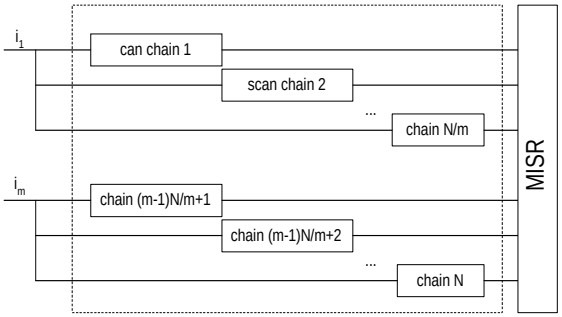


Figure 1: Illinois scan architecture.

In this architecture, the scan chain segments that share the same scan-in have dependencies in values and are limited in the values they can take on. Figure 2(a) and (b) give a pictorial view of these dependencies. The picture shows the values in the scan elements after the complete scan operation is completed. Cells shaded in the same color take on the same value in a scan operation. The dependencies in the scan elements are apparent in the figure. Benefits are achieved in TDV and TAT when these dependencies do not conflict with the test patterns requirements.

Test patterns that do not require conflicting values from scan-elements that have the same value can use this architecture. If a test pattern can use the architecture in Figure 2(a) the TDV for that test pattern is reduced by a factor of four. Consequently, the TAT compared to the equivalent number of scan-ins in the traditional scan architecture would also be lower by a factor of four. Test patterns that cannot use this architecture because of the dependencies in values would have to be applied in an alternate way. If the alternative is to apply the test pattern through the traditional serial scan chain then the test pattern would not benefit from any TAT or TDV reduction.

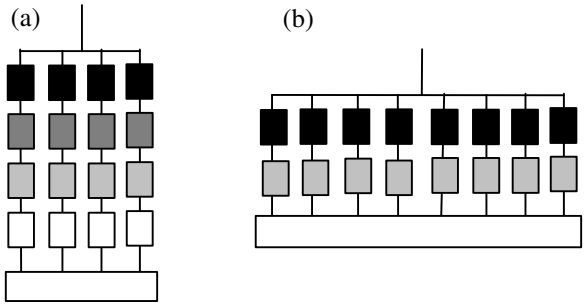


Figure 2: Value dependencies for the scan chains that share the same scan input.

An important aspect of this architecture is that when there is a more aggressive sharing of the scan-ins, there are more dependencies in the scan values (shown in Figure 2(b)) and fewer tests can utilize this architecture. These test patterns that could not utilize the architecture would have to use the traditional single scan chain method to apply the tests. To alleviate this the common scan-in architecture was improved to provide two configurations that use different scan configurations [6]. In the reconfigurable architecture a test pattern can either use one configuration or the other. This method is shown to provide a small improvement over the original non-reconfigurable common scan-in architecture.

3. Reconfigurable Architecture that Allows for Changing Scan Enable

The architecture which allows a single test pattern use multiple configurations of the scan chains is shown in Figure 3 and was originally presented in [9].

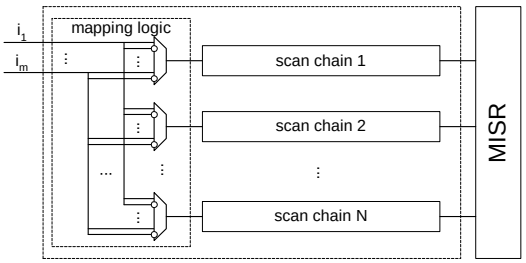
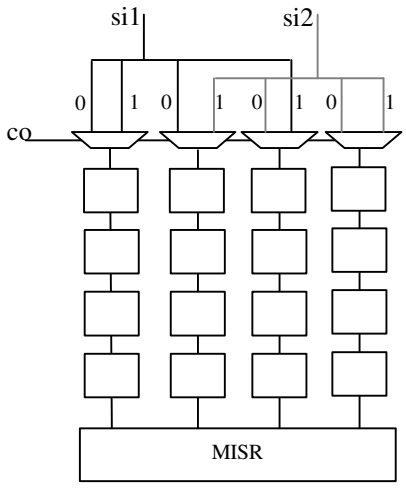


Figure 3: A Reconfigurable architecture that allows for a single test pattern to use multiple configurations to load the scan chains.

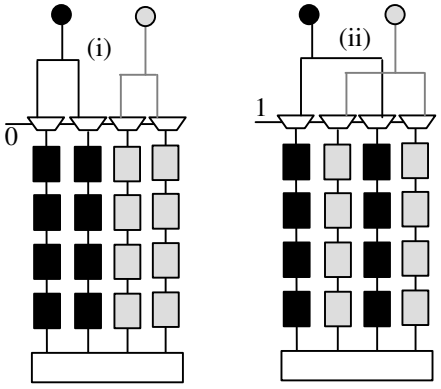
Multiplexers at the beginning of the scan chains allow multiple alliances between the scan inputs and the scan chains. That is, under one setting of the multiplexer control signals, a scan chain is connected to a particular scan input and in another setting the same scan chain is connected to a different scan input. Multiple scan chains can be connected to the same scan input – and hence would get the same scan data. The novelty of the architecture proposed her is that the multiplexer control can be kept constant during the application of a test or be changed while applying the same test. The special case where it is constant is equivalent to the architecture presented in [9] and is conceptually similar but actually quite different to other reconfiguration methods defined in the past [6]. The following example shows how changing the scan enable during the scan operation provides dynamism in the scan chain architecture that is the equivalent of a larger number of static reconfigurations. Thus it is almost guaranteed that a test pattern can be applied through the common scan in configuration.

Figure 4 shows a design with 16 scan cells configured into 4 scan chains. These scan chains are connected to two scan inputs through multiplexers. In one configuration (Figure 4 (b)-(i)) the first two chains are connected to the first scan input and the second two chains are connected to the second scan input. The scan cells are color coded to show their relationship to the scan-ins. In the second configuration (Figure 4 (b)-(ii)) the first and third chain are connected to the first scan input and the second and fourth chain are connected to the second scan input. When the multiplexer control is static for the entire test pattern only two configurations (as shown in the figure) are possible. When the multiplexer control is changed along with the shift operation of the test pattern many more configurations are possible as each shift could take on the available static configurations. An example configuration is shown in Figure 4 (c) where the scan enable signal is applied a logic-1 for the first two shift cycles and then a logic-0 is applied for the remaining two shift cycles. An example test pattern can be very easily constructed that has multiple conflicts in different shift locations such that it cannot use the two configurations in a static manner but can be applied in the dynamic configuration shown in the figure. The ability to have many more configurations at the expense of some test data on the scan enable allows the architecture to provide a more efficient platform for test patterns.

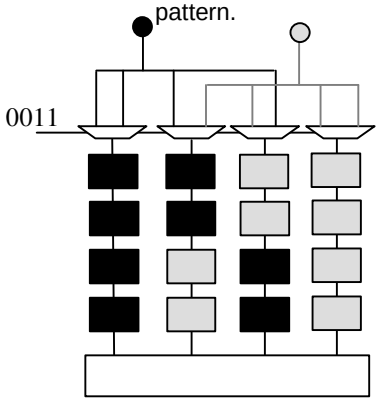
The implementation requires that the timing of the control signal be carefully adjusted to match the shift operation. Since shifting is normally done at much a much slower speed than the operation of the design this is easily achievable.



(a) 4 scan chains connected with two configurations.



(b) The two configurations available to scan data with static control for the test pattern.



(c) Dynamic reconfiguration of chains during a single test pattern.

Figure 4: Example reconfiguration structure and configurations. Static nature and Dynamic nature of the configurations are highlighted. Shading is used to depict the connection between scan inputs and the values in the scan flip-flops.

4. DFT Synthesis

While the explanation describes this architecture to have a particular implementation, a number of variations provide

similar capabilities and results. The architecture described has some multiplexing logic on the scan-inputs and high observability on the scan-outputs. On the input side many different mappings of the scan chain segments to scan inputs could exist. On the output side MISR can be replaced by XOR circuitry to observe values at outputs instead of the configuration shown.

There are two parts to our experimental implementation of the architecture. The first part is the creation of scan-chains without any association with scan-ins. The second part is connecting the scan chains to the scan-ins for each configuration. A more detailed analysis of these procedures is presented in [10].

4.1. Determining Scan Chains

This part of the analysis determines membership of scan cells in scan chain segments. This step provides a small constraint on existing scan chain insertion flows which consider other constraints such as clocking, placement and routing. While this step was performed during the experiments described in this paper, it is not a requirement for achieving good results with the architecture described.

The goal is to construct scan chain segments such that the number of *potential conflicts* between scan chains is minimized. A *potential conflict* is defined to exist between any pair of scan cells that belong to the same cone of logic and are placed in two different scan chains. Tests for faults in a cone require values from the scan cells driving the cone. Scan values required from scan cells in the same scan chain can never conflict. The *potential conflict* becomes a *real conflict* when the event occurs that satisfies all of the following additional criteria.

1. Values needed in two scan cells are not compatible (a logic-0 and a logic-1).
2. The two scan cells are in the same shift position relative to the scan-in of the chains.
3. The two scan cells are in scan chains that are sharing the same scan-in.

Thus the cone of influence is used as a simple mechanism of constructing the scan chains. Let us say we are constructing the architecture for a design with F scan-cells and we chose to have N scan chains. Then the length of the scan chains is L, where $L = \lceil F / N \rceil$. Topological logic cones are constructed for every observable point of the design and sorted by size[1]. Starting with the inputs of each cone in the list created, the first L unassigned scan cells encountered are assigned to a partition for the creation of a scan chain. The following L cells are assigned to the next partition for the creation of another scan-chain and the process is continued until all scan-cells are assigned to some scan-chain partition. The scan-chains are constructed by DFT tools that consider routing and other constraints for the cells of each partition. Given the way in which the scan chains were selected, it is likely that most scan cells in a given cone are either in the same scan chain or scan chains immediately before and after the scan chain. Therefore, a majority of the scan cells that have values required by a test pattern are either in the same scan chain or in adjacent scan chains. The overlapping of cones causes scan cells of within a cone to not reside in the same or adjacent scan chains with scan cells in the same cone.

[1] Sorting by size provides lower possibility of conflicts with the scan-chain to scan-in assignment scheme used.

4.2. Assigning Scan-Chains to Scan-Ins for Multiple Configurations

This step involves the creation of a mapping between the scan segments and the available scan inputs.

Given maximum of M scan-in pins, in each configuration we propose to share a scan-in pin among every m^{th} segment, where $2 \le m \le M$. It should be observed that if there is a dependency due to sharing of an input among every i^{th} chain in one configuration, and there is a dependency due to sharing of an input among every j^{th} chain in another configuration, among the two configurations there also will be a dependency among every k^{th} chain, where k is the *least common multiplier* (LCM) of i and j. Thus, for each configuration m should be chosen in such a way as to maximize the LCM of all m's. If the LCM of all m's is larger than the number of internal chains in the design, then every fault that has a single real conflict among all the configurations is guaranteed to be detectable. The dynamism introduced in this paper improves this guarantee to apply to a single real conflict among all configurations on a per-shift basis, thus allowing more faults to be compacted.

While using larger m may result in the larger LCM, using larger number of inputs increases the test data volume. In our experiments we tried to minimize the TDV, therefore, we tried to detect as many faults as possible with smallest m, and increasing m gradually for the faults that could not be detected with smaller m. Thus, we used $m = 2, 3, 5, 7, 11\ldots$

The ability to choose a particular configuration is provided through multiplexing logic between the scan chain terminals and the scan inputs. The control signals of the multiplexer are the scan enable signals that determine the configuration selected.

4.3. Area Overhead

On the input side the mapping of the configurations repeat after the least common multiple of the configurations is achieved. For example, if three configurations are defined then there would be 2*3*5 = 30 unique mappings of scan-ins to scan-segments. Each mapping needs one 3-1 MUX or logic that is equal to 6 two input gate. Thus the total overhead is 180 gates for the input side regardless of the

number of scan segments. This calculation does not include the serial configuration of the scan chains. On the output side, one can implement an efficient non-redundant XORing of the scan chains to the available scan outputs. For that configuration the overhead would be one XOR per scan chain which is 3 two input gate equivalents. A MISR on the output side would result in a different area overhead.

The pin overhead due to additional configurations grows logarithmically with the number of configurations used.

5. Experimental Results

This paper introduces dynamism (as a result of applying test data to the scan enable signal) into the common scan-in reconfigurable architecture. The results of the architecture with only static configurations was presented in [9]. Thus the results of this paper are focused on analyzing the benefits of dynamism over static reconfigurations. A test pattern that cannot be applied exclusively in a static configuration may be applied in a dynamic reconfiguration. Results are limited to configurations that are not created with any significant analysis to show that the implementation flow of DFT – ATPG in realistic scenarios is simple. More elaborate analysis methods in determining the configurations can only add to the results and improve them.

5.1. Method of Collecting the Data

The experiments presented in this paper concentrated on reducing the test data volume. Therefore, the experiments are performed by first attempting to use the configurations statically with the least number of input pins (small *m* to large *m*). Since using less pins (smaller *m*) implies less test data volume, running the experiment with the priority described above highlights the test data volume reduction at the expense of some test application time. Using all pins all the time would improve the test application time provided here at the expense of extra test data volume. the efficiency of using the dynamic reconfigurability of the architecture. Once the static configurations are utilized to their maximum for fault detection, unconstrained ATPG is run on the remaining undetected faults with dynamic switching of all available configurations by changing the scan enable during the scan operation.

The experiments were performed on three industrial designs and some ISCAS benchmark circuits. Table 1 shows the characteristics of the industrial designs.

Table 1: Industrial circuits used in the experiments.

Design	Gates	Faults	Scan cells	Cone size	
				Max. inputs	Max. gates
A	230k	481k	9700	432	1887
B	390k	554k	12500	282	916
C	1083k	2740k	69000	264	5454

5.2. Method of Computing the Results

The data volume reduction (DVR) for the experiments is calculated as follows. In general, the DVR is:

$$DVR = DV_{ATPG} / DV_{NEW}$$

DV_{ATPG} = TestPatterns * ScanChains * MaxChainLength

$DV_{NEW} = DV_{STATIC1} + DV_{STATIC2} + \ldots + DV_{STATICM} + DV_{DYNAMIC}$

$DV_{STATICi}$ = Patterns in configuration i * [ScanPinsUsed (*m*) * MaximumChainLength (*L*) + UnusedScanPins]

$DV_{DYNAMIC}$ = Dynamic patterns * (ScanPinsUsed(m)+ ShiftControlPins(t))* MaximumChainLength (*L*)

In the static configurations, *m* bits of data are loaded *L* times per patterns and each unused scan pin is specified once in each pattern. For dynamic testing, the scan pins (including the control pins) are loaded *L* times for each pattern. The results are presented in Table 2. For the configurations used, the reconfigurable architecture of this paper overcomes the dependencies caused by the common scan-in for significant benefits over the non-reconfigurable static architecture. In case of M = 7 means that 4 configurations were implemented in the architecture (m=2,3,5,7), which require additional t = 2 control pins. Similarly, M = 5 means 3 reconfigurations controlled by 2 pins were created for the architecture described in this paper. When calculating DVR we kept the total number of I/O pins the same as in regular scan by using some of the regular scan-in pins for control purposes.

The test application time (TAT) depends on the length of the longest scan chain during regular ATPG. In general, the TAT is:

$$TAT = TAT_{ATPG} / TAT_{NEW}$$

TAT_{ATPG} = TestPatterns * MaximumChainLength

$TAT_{NEW} = TAT_{STATIC1} + \ldots + TAT_{STATICM} + TAT_{DYNAMIC}$

$TAT_{STATICi}$ = PatternsInConfiguration i * MaximumChainLength (*L*)

$TAT_{DYNAMIC}$ = DynamicPatterns* MaximumChainLength (*L*)

The advantages of using dynamism can be seen in Table 3. In this set of results the number of configurations needed to apply all the patterns through the shared scan-in was determined. Then an execution was performed with fewer static configurations and a cleanup pass using the dynamic configuration to apply all the remaining patterns. The results show that without dynamism the circuit requires more input pins and more static configurations for all faults to be tested using the common scan-in architecture.

6. Conclusions

In this paper we present a dynamic reconfiguration method that allows changing the scan enable signal during the scan operation of a single test pattern. A specific implementation of this architecture is used for the experiments that do not require significant analysis in the

creation of the configurations. Dynamism is compared to the static configurations and shown to be superior in the ability to apply more test patterns with the same number of configurations. This benefit can be viewed in multiple ways:

1. Dynamism allows less sensitivity of the scan-cell to scan-chain membership. As a result a simpler and, therefore, faster, analysis can be used to create the configurations for the same results of static reconfigurations. In this paper cell membership and reconfigurability was created without any design analysis.
2. Dynamism allows for the need for fewer configurations for the same level of analysis and results of the static configurations.

This paper shows significant benefits can be achieved by simple low overhead architectures.

7. References

[1] C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, B. Keller and B. Koenemann, "OPMISR: The Foundation for Compressed ATPG Vectors," Proceedings of the International Test Conference, 2001, pp. 748-757.

[2] M. J. Geuzebroek, J. Th. van der Linden and A. J. van de Goor, "Test Point Insertion that facilitiates ATPG in reducing test time and data volume," Proceedings of the International Test Conference, 2002, pp. 138-147.

[3] I. Hamzaoglu and J. H. Patel, "Reducing Test Application Time for Full Scan Embedded Cores," Proceedings of the International Symposium on Fault Tolerant Computing, 1999, pp. 260-267.

[4] A. Jas and N. A. Touba, "Test Vector Decompression via Cyclical Scan Chains and Its Application to Testing Core-Based Designs," Proceedings of the International Test Conference, 1998, pp. 458-464.

[5] A. Jas, B. Pouya and N. A. Touba, "Virtual Scan Chains: A means for reducing scan length in cores," Proceedings of the VLSI Test Symposium, 2000, pp. 73-78.

[6] A. R. Pandey and J. H. Patel, "Reconfiguration Technique for Reducing Test Time and Test Data Volume in Illinois Scan Architecture Based Designs," Proceedings of the VLSI Test Symposium, 2002, pp. 9-15.

[7] J. Rajski, J. Tyszer, M. Kassab, N. Mukerjee, R. Thompson, K. H. Tsai, A. Hertwig, N. Tamarapalli, G. Mrugalski, G. Eide and J. Qian, "Embedded Deterministic Test for Low Cost Manufacturing Test," Proceedings of the International Test Conference, 2002, pp. 301-310.

[8] S. Samaranayake, N. Sitchinava, R. Kapur, M. B. Amin and T. W. Williams, "Dynamic Scan: Driving Down the Cost of Test," IEEE Computer, October 2002, pp. 63-68.

[9] S. Samaranayake, E. Gizdarski, N. Sitchinava, F. Neuveux, R. Kapur, T.W. Williams, "A Reconfigurable Shared Scan-In Architecture", Proceedings of the VLSI Test Symposium, 2003, pp. 9-14.

[10] S. Samaranayake, "A Reconfigurable Shared Scan-In Architecture", Master's Thesis, Massachusetts Institute of Technology, 2003.

[11] K. D. Wagner, "Robust Scan-Based Logic Test in VDSM Technologies," IEEE Computer, November 1999, pp. 66-74.

Table 2: Comparing the proposed architecture to the original Illinois Scan architecture

Circuit	Internal chains	L	Regular ATPG			Illinois Scan					Proposed architecture				
			M	t	Patterns	M	t	Broad. Pat.	Serial Pat.	DVR = TATR	M	t	Patterns	DVR	TATR
Circuit A	487	20	9	0	966	9	0	2138	218	4.27	7	2	2910	64.42	18.04
Circuit B	516	26	9	0	748	9	0	1452	438	1.69	7	2	2767	48.00	15.48
Circuit C	537	135	9	0	2361	9	0	3185	418	5.49	7	2	3715	138.6	35.82
S13207	80	11	9	0	149	9	0	157	78	1.80	7	2	305	9.30	4.31
S38417	129	14	7	0	137	7	0	406	60	2.07	5	2	731	9.14	3.43
S38584	139	13	9	0	230	9	0	286	142	1.57	7	2	595	13.56	2.97

Table 3: Static and dynamic Testing

Design	Static Only			Static +Dynamic			
	Patterns	Configurations	Pins (M)	Total Patterns	Dynamic Patterns	Configurations	Pins (M)
A	2922	6	13	2910	22	4	7
B	2780	5	11	2767	34	4	7
C	3712	5	11	3715	25	4	7
s13207	305	5	11	305	6	4	7
s38417	735	6	13	731	16	3	5
s38584	562	6	13	595	45	4	7