Reducing Test Time for 3D-ICs by Improved Utilization of Test Elevators

Sreenivaas S. Muthyala and Nur A. Touba

Computer Engineering Research Center
University of Texas, Austin, TX 78712
sreenivaas@utexas.edu, touba@utexas.edu

Abstract

A highly efficient test compression scheme for 3D-ICs is proposed, which uses sequential linear decompressors local to each core. The compressed test data is brought from the tester over the test access mechanism (TAM) to the cores where they are decompressed. The idea is to provide flexibility in the utilization of the free variables (i.e., bits stored on the tester that can be assigned 0 or 1), so that the free variables that are not used in one core can be used to encode test cubes in other cores. The decompressors are daisy-chained, such that some of the free variables brought in to one decompressor are passed on to the other decompressors. Consequently, the free variables are also shared with the decompressors in other layers. This enables better utilization of free variables, since free variables not used in one decompressor can be used by any of the other decompressors with which they are shared. The encoding efficiency improves considerably when the free variables are shared with other cores. This reduces test time and tester storage without any additional control. In addition, this architecture also minimizes the number of test elevators required to transfer the test data across layers. The scan chains driven by a decompressor are local to the layer in which the decompressor is present. Hence, only the input to the decompressor, i.e., the compressed test data, is transferred across layers, which reduces the number of test elevators. Furthermore, it is also shown how the number of through-silicon-vias (TSVs) can be reduced further by implementing a test data serializer in the sending layer driving the TSVs and a deserializer converting the serialized data from the TSVs back to the original form.

1. Introduction

Three-dimensional integrated circuits (3D-ICs) using through-silicon vias (TSVs) are an important new technology that provides a number of significant advantages including increased functional density, shorter interconnect, higher performance, and lower power. Stacking in a 3D-IC can be done wafer-to-wafer (W2W), die-to-wafer (D2W), or die-to-die (D2D). 3D-IC designs are typically composed of reusable cores, each of which must be thoroughly tested. Cores can be synthesizable designs (i.e., soft cores) in which the design-for-test (DFT) architecture can be customized (e.g., number and length of scan chains, etc.), or they can be layouts (i.e., hard cores) in which the DFT architecture is fixed. For intellectual property cores (IP cores), it may not be possible to perform ATPG, in which case a set of *test cubes* (i.e., test vectors in which the unassigned inputs are left as don't cares) is provided that must be applied during test.

Testing a core-based design typically involves designing wrappers to go around the cores, providing test access mechanisms (TAMs) for transporting data from the tester pins to the cores, and developing a test schedule for determining which cores are being tested at different times. Many techniques for designing wrappers, TAMs, and test schedules have been developed, see [Xu 05] for a survey.

3D-IC testing differs from conventional 2D testing in two important ways. One is that transporting test data between layers requires the use of TSVs (i.e., *test elevators*) which are expensive in terms of both area overhead as well as impact on yield. Consequently, it is important to consider the number of test elevators when optimizing a test architecture. Optimization procedures for minimizing test time under a constraint on the number of test elevators has been proposed in [Wu 08] and [Noia 10]. A second difference for testing 3D-ICs is that pre-bond testing is performed on each die individually, to ensure defective dies are not used in stacking. Then post-bond testing is performed on the final stack to ensure the 3D-IC as a whole is not defective. In some cases, testing might also be performed on partial stacks. In order to do pre-bond testing on the non-bottom layers, it is necessary to add probe pads for test purposes. This is because the TSV tips as well as the microbumps are too small to be probed and are sensitive to scrub marks [Marinissen 09]. These probe pads are a test overhead that take up a lot of space and limit the locations where TSVs can be placed which puts constraints on the design and floorplan of a die. Consequently, the number of probe pads for non-bottom layers is typically very limited resulting in less bandwidth being available from the tester for pre-bond testing than available in post-bond testing. Techniques for handling this include either having separate TAMs for pre-bond and post-bond testing [Jiang 12], or having a single TAM with a bandwidth adapter [Lee 13].

One important way to reduce test time is by using test compression [Touba 06]. The conventional way of implementing test compression in core-based designs is for each core to have its own local decompressor. This allows compressed data to be brought over the TAM lines to each core. Note that if a decompressor was shared among multiple cores, then uncompressed data would need to be routed from the output of the decompressor to the cores

978-1-4799-6016-3/14/$31.00 ©2014 IEEE

which would require much larger TAMs thereby greatly increasing routing complexity. In addition, for 3D-ICs if the uncompressed test data is shared across layers, the number of test elevators required to transfer the uncompressed test data will be much higher. The amount of compression achieved with sequential linear decompressors depends on the *encoding efficiency,* which is defined as the ratio of the number of free variables (bits stored on the tester) versus the number of care bits in the test data. The encoding efficiency that can be achieved when using the conventional approach of having fixed size TAMs feeding multiple independent core decompressors is limited by the fact that the TAM bandwidth to each core decompressor needs to be large enough to bring enough free variables to encode the worst-case test cube with the largest number of care bits. To maximize encoding efficiency, a special ATPG procedure is used to generate test cubes in a way that limits the number of care bits in each test cube. This typically comes at the cost of more test cubes being generated. Moreover, there can still be a considerable amount of variance in the number of care bits per test cube (profiles for industrial circuits showing how the percentage of care bits varies can be found in [Janicki 12]). The inherent drawback of the conventional approach is that the number of free variables brought in for decompressing each test cube is fixed by the worst-case test cube with the most care bits, and so many free variables are wasted for test cubes with fewer care bits.

Some recent work has looked at ways to improve encoding efficiency by dynamically adjusting the number of free variables sent to the decompressor in each core on a per test cube basis to try to better match the number of free variables sent with the number of care bits. This can be done by dynamically allocating the number of tester channels used to feed each decompressor [Janicki 12] or by selectively loading decompressors in each clock cycle [Kinsman 10], [Muthyala 13]. [Larsson 08] presents selective encoding of test data for testing SOCs using codewords to represent the care bit profile of the test cubes. The conventional approach is to have independent decompressors for each core such that each of them have their own set of free variables so that the linear equations for each decompressor can be solved independently. This helps to reduce computational complexity. The drawback is that the unused free variables are wasted when the decompressor is reset between test cubes. This happens in all of the existing methods except [Muthyala 13], which allows limited sharing of free variables between decompressors during a few cycles in such a way that the linear equations are mostly independent and can be solved with only a small increase in computational complexity.

The existing methods for dynamically adjusting the number of free variables sent to each decompressor have the drawback that they increase the amount of routing (i.e. TAM width) required between the tester channels and the core decompressors. This is less suited for 3D-ICs because routing between layers requires additional test elevators (i.e., extra TSVs) which are expensive. This paper proposes a new architecture and methodology for test compression in core-based designs, which is better suited for 3D-ICs. It can be used to either provide greater compression for the same number of test elevators or reduce the number of test elevators while maintaining the same compression compared to existing methods.

Whereas conventional methods for test compression in core-based designs are based on an architecture where tester channels are routed directly to independent core decompressors, the proposed method uses a daisy chain architecture as described in Sec. 2.

In conventional testing, the scan shift frequency is typically much slower than the functional clock frequency and the maximum ATE (automatic test equipment) clock frequency, due to power dissipation limitations as well as the fact that the test clock may not be buffered and optimized for high speed operation. In [Khoche 02], the ATE shifts in test data n times faster than the scan shift. An n-bit serial-to-parallel converter is used to take in serial data at the fast ATE shift rate and convert it to n-bits in parallel at the slower scan shift frequency. This allows a single ATE channel to be used to fill n scan chains in parallel. This idea was combined with a test compression scheme in [Wang 05], where faster ATE channels are sent to a time division demultiplexing (TDDM) circuit which feeds the inputs of a VirtualScan decompressor [Wang 04], and the compacted output response is fed through a time division multiplexing (TDM) circuit to go to the ATE.

This paper proposes using a similar concept for test elevators when transferring test data from one layer to another layer in a 3D stack. The idea is to use a serializer in the layer sending the test data that accepts data in parallel at the scan shift frequency, and generates a serial output at the functional clock frequency which is sent over the test elevator to a deserializer on the receiving layer which converts the serial data coming in at the functional clock frequency into parallel outputs at the scan shift frequency. This approach is described in detail in Sec. 3.

This paper is organized as follows: The proposed architecture is explained in Sec. 2. Implementation of serializer driven TSVs to further reduce number of test elevators is detailed in Sec. 3. Sec. 4 tabulates the experimental results and Sec. 5. is a conclusion.

2. Proposed Architecture

The conventional approach for test compression in a core-based design is illustrated in Fig. 1. Each core has its own decompressor operating independently of the other core decompressors. The input test data bandwidth from the Automatic Test Equipment (ATE) is distributed to core decompressors using TAMs. The tester channel allocation is done in such a way that the total test time for testing the entire 3D-IC is minimized.

In the architecture shown in Fig. 1, the test channel allocation is static. In order to make it dynamic, it is

necessary to route the entire ATE bandwidth to all the layers, and control the number of channels allocated by providing additional control signals, which need more test elevators compared to static channel allocation.

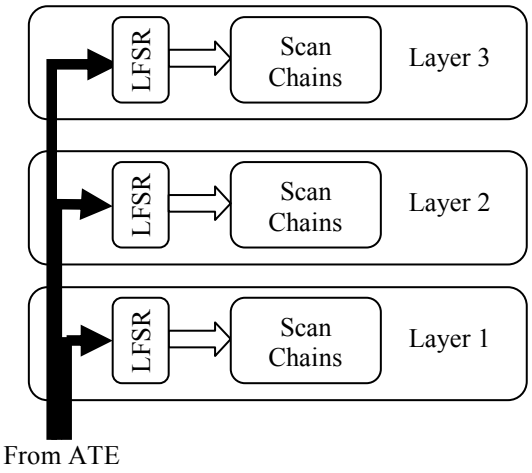

Figure 1. Test Architecture of 3D-ICs with Static Allocation of Tester Channels

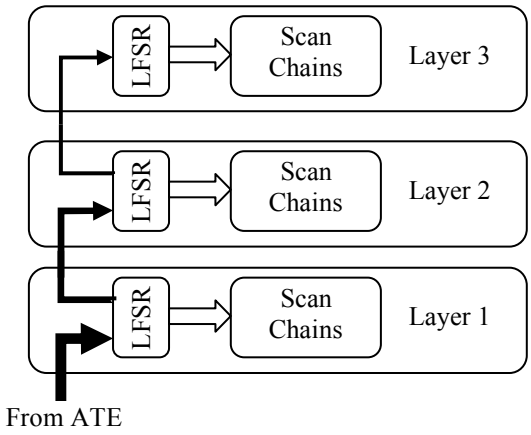

Figure 2. Proposed Architecture with Daisy-Chained Decompressors

The proposed architecture is shown in Fig. 2; the decompressors of the cores are daisy-chained together. In the architecture shown in Fig. 2, the decompressors of the cores are daisy-chained together. The number of channels between layer 2 and layer 3 is less than the number of channels between layer 1 and layer 2, which in turn is less than the input bandwidth to the decompressor in layer 1, i.e. total bandwidth from the ATE. Using this architecture, some free variables from thedecompressor in layer 1 is sent to the decompressor in layer 2 and so on. This provides some flexibility in encoding the test cubes. Free variables unused while encoding a core in a layer are available to encode other cores in the layers above, hence improving encoding efficiency.

The bandwidth feeding each layer should be sufficient to supply enough free variables to encode the test cubes for the core in that layer as well as the free variables to encode the test cubes for the layers above, as the decompressor in each layer is fed via taps from the decompressor of the layer below. Hence, the bandwidth supplied to layer 1 should be big enough to supply free variables to encode test cubes of the cores in layer 1 as well as cores in layer 2 and layer 3. Similarly, the bandwidth supplied to layer 2 should be big enough to supply free variables to encode test cubes of the cores in layer 2 and layer 3. In other words, the number of free variables supplied to decompressor in each layer decreases as we move from the bottom layer to the top layer. Hence, a tapering bandwidth is used, as shown in Fig. 2, proportional to the number of free variables required to be supplied to the decompressor in each layer. Consider an example using the conventional architecture shown in Fig. 1, where the optimal static allocation of the 32-bit channel from the ATE is as follows: a 16-bit TAM for layer 1, 10-bit TAM for layer 2 and a 6-bit TAM for layer 3. The bandwidth assignment for the proposed architecture using daisy-chained decompressors (Fig. 2) is as follows: the full test data bandwidth from the ATE is sent to the first decompressor using a 32-bit TAM. Output taps from the first decompressor are used to drive a 16-bit TAM to the decompressor in layer 2. Similarly, output taps from the second decompressor drive a 6-bit TAM in layer 3.

The proposed architecture provides flexibility in utilization of free variables; any free variable coming from the tester can be used in any of the layers, provided it is distributed across all the layers. By daisy-chaining the decompressors as shown in Fig. 2, some free variables from the decompressor in layer 1 are fed to the decompressor in layer 2, similarly some free variables in layer 2 are fed to the decompressor in layer 3. Using this architecture, unused free-variables from one decompressor can be used by other decompressors.

Consider the decompressor in layer 3 is encoding an easy-to-encode test cube with few specified bits, which needs less than average free variables, while the test cube being encoded in layer 2 is hard-to-encode with a larger number of specified bits, needing more free variables than average. In the proposed architecture, the free variables which are not required in layer 3 can be used in layer 2 in help encode the hard-to-encode test cube. This reduces the number of free variables required to encode the entire test set for the 3D-IC, without additional hardware and control, with the same number of TSVs as used in the conventional architecture shown in Fig. 1.

To illustrate the encoding advantage of the proposed approach, consider a small example in which a 3D-IC has three layers with each layer having one core. Let the set of test cubes for the cores have care-bit profiles as shown in Table 1. To encode the entire 3D-IC using the conventional architecture shown in Fig. 1, core 1 would need a minimum of 13 free variables per test cube, core 2 would need 11 free variables and core 3 would need 12 free variables per test cube. Since there are 8 test cubes for each core, a total of

(13 + 11 + 12) x 8 = 288 free variables are required to encode the 3D-IC.

Table 1. Example - Care Bit Profile of Test Cubes for the Three Cores of the Example 3D-IC

Test Cube	# Care bits in Test cube		
	Core 1	Core 2	Core 3
1	13	11	12
2	12	11	10
3	10	10	9
4	9	7	8
5	8	6	7
6	7	5	5
7	7	5	4
8	6	4	3

Consider the same 3D-IC, now using the proposed architecture. In this case, encoding is done in groups of three test cubes, with one test cube from each core. Optimizing the test cubes in the group for minimizing the total number of care bits in the group, the test cubes are grouped as shown in Table 2.

Table 2. Care Bit Profile of Test Cube Groups for Encoding Using Proposed Architecture

Test Cube Group	# Care bits in Test cube			Total Care Bits in Test Cube Group
	Core 1	Core 2	Core 3	
1	13	4	7	24
2	12	5	8	25
3	10	5	12	27
4	9	6	10	25
5	8	7	9	24
6	7	10	5	22
7	7	11	3	21
8	6	11	4	21

By encoding in groups according the above table, the maximum number of care bits in any group is 27, and hence, to encode the entire set of eight test cube groups, 27 x 8 = 216 free variables are required using the proposed architecture, which is less than the number of free variables required to encode using the conventional architecture shown in Fig. 1. Hence, it is seen that the proposed architecture gives a better compression and hence reduces the tester storage requirements and test time.

The drawback in the proposed architecture is that the linear equations for scan cells driven by daisy-chained decompressors cannot be solved independently. Consider one test cube from each layer being encoded. The total number of free variables brought in from the tester has to be sufficiently large enough to encode all three test cubes. Hence, creating pivots for care bits in all the three test cubes involves a lot of computation. The total number of XOR operations required to create pivots is $9n^3$ as compared to $3n^3$ XOR operations required to encode using the conventional approach. Hence, this method is reasonable when the additional computation is a reasonable price to pay for the amount of test time reduction achieved.

3. Optimizing Number of Test Elevators by Inter-layer Serialization of Test Data

In this section, an implementation is proposed using a serializer-deserializer structure to further reduce the number of test elevators required to implement the proposed architecture. The scan shift frequency is usually lower than the functional frequency, since the scan clock tree is generally not buffered up for high speeds. Another reason is that during scan, a large percentage of flip-flops toggle, and hence a large amount of power is drawn from the power grid of the chip. This causes a voltage drop in the power lines. To avoid these problems, scan shifting is generally considerably slower than functional frequency.

By using the proposed implementation, the difference between the slower scan shift frequency and the faster functional frequency can be exploited to further reduce the number of test elevators. The idea is to use a serializer at the layer sending test data to serialize the test data from the decompressor taps and drive the test elevators in the driving layer using this serialized data, and a deserializer at the receiving end, which restores the test slices in the receiving layer by converting the serial data back into parallel format (Fig. 3).

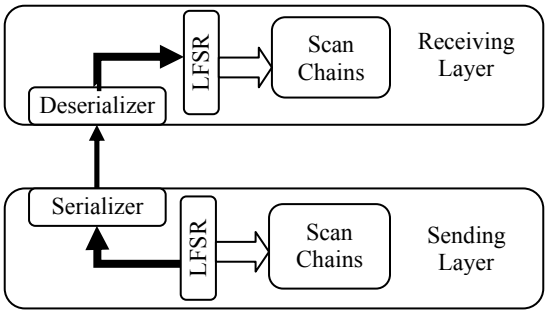

Figure 3. Proposed Implementation of Inter-layer Serialization of Test Data

If the test elevators can be operated at *n* times the scan shift frequency, then instead of having *m* test elevators to transfer data across layers, $\lceil m/n \rceil$ test elevators are required. On the other hand, it is also possible to have the same number of test elevators and increase the effective bandwidth by *n* times, i.e., *m* x *n* bits of data can be shifted in one shift cycle using *m* test elevators implementing inter-layer serialization, as compared to *m* bits of data shifted in one shift cycle using *m* test elevators without serialization. Hence, depending on the constraints on the number of test elevators, this architecture can be used at an advantage to either increase the effective bandwidth or reduce the number of test elevators required in the design.

Consider a serializer driven by *m* taps from the LFSR in the lower layer. Let the functional clock be *n* times faster than the scan clock. Hence, as explained previously, the number of test elevators required would be *m/n.* Let the number of test elevators required be represented as *t,* which, here, is equal to *m/n.* Inter-layer serialization would require

an *m* x *t* serializer in the sending layer driving the test elevators between the layers and a *t* x *m* deserializer driving the LFSR in the receiving layer. The simplest way of implementing an *m* x *t* serializer in the sending layer is by using a *m:t* multiplexer controlled by a modulo *m* counter driven by the faster functional clock (Fig. 4).

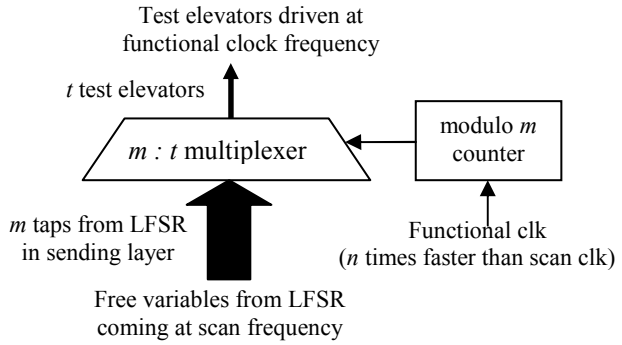

Figure 4. Serializer Implementation for Functional Clock Operating *n* Times Faster than Scan Clock, Where Number of Test Elevators, *t = m/n*

This ensures that the test data coming in at scan shift frequency from the LFSR iscoupled to the test elevators at the faster functional clock frequency. Similarly, the deserializer can be implemented as an *n* bit shift register (*n* is the ratio between functional clock frequency and scan clock frequency) driven by the faster functional clock, whereas the data in the shift register is sampled at the slower scan clock rate (Fig. 5).

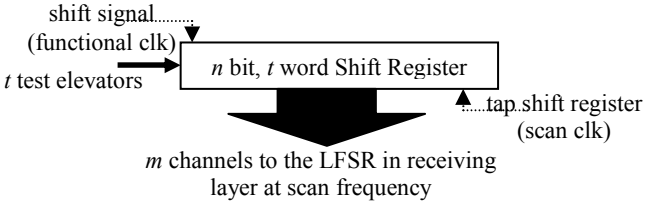

Figure 5. Deserializer Implementation for Functional Clock Operating *n* Times Faster Than Scan Clock, Where Number of Test Elevators, *t = m/n*

As discussed above, inter-layer serialization has a small area overhead, hence it can be used in cases where the advantages of using it outweighs the additional cost of implementing the required architecture in the core.

4. Experimental Results

Experiments were conducted on six different 3D-IC designs and the results are presented in this section. Each of the 3D-ICs has three layers. Three different test compression architectures were experimented using these test chips. In the first test architecture (*arch1*), each core has a 64-bit LFSR, acting as a decompressor. The input tester channels from the ATE were allocated to the decompressors statically. In this architecture, the output cone of the each decompressor is confined to the layer in which the decompressor is present, i.e. test elevators are required to transfer the compressed test data to the decompressors in the non-bottom layers, however, the scan cells which are driven by a decompressor are localized to the layer in which the decompressor is present. Hence, it is enough to have sufficient test elevators to transfer the compressed test data, which is generally less in number.

In the second architecture (*arch2*), the 64-bit LFSRs that were local to each layer in *arch1,* are interconnected and reconfigured to form a big primitive LFSR with number of flip-flops in this big LFSR being the sum of the number of flip-flops in the *arch1* LFSRs in all the three layers. It should be noted that the LFSR in *arch2* is distributed across the three layers, i.e. sections of LFSR are present in each of the three layers and these sections are interconnected using test elevators in such a way that a primitive LFSR is formed and drives scan chains in all three layers of 3D-IC. In this case, the scan chains are confined within the layer, i.e. test elevators are required to transfer compressed test data to the sections of LFSR in the non-bottom layers and to interconnect the sections of LFSR which are in different layers. Hence, more test elevators are required in *arch2* than *arch1*. Using this architecture, the equations for the scan cells of the three layers are solved together and since the pivots for the test cubes are created in common, the free variables are used more efficiently and results in better compression and increased encoding efficiency.

The third architecture (*arch3*) is the proposed architecture using daisy-chained decompressors shown in Fig. 2. In this case, the decompressors used are similar to the ones used in *arch1*, a local 64-bit LFSR in each core acting as a decompressor for the core, with all the decompressors daisy-chained together. The tester channels are allocated to the decompressors as proposed in Sec. 3. This method combines the advantages of *arch1* and *arch2* at the cost of increased computational complexity of encoding the test cubes together. By using this architecture, some of the free variables are distributed to the decompressors in the other layers and the encoding of the test cubes of the three layers are done together, thereby the free variables are used more efficiently and results in better compression and increase in encoding efficiency similar to *arch2*. However, in this method, test elevators are required only to transfer the compressed test data to the non-bottom layers, similar to *arch1,* while providing an encoding advantage similar to *arch2*. In addition, reconfiguration of the local 64-bit LFSRs into a big LFSR for post bond testing of the 3D-IC is also not necessary when using *arch3*.

Experiments were run on six different designs of 3D-ICs, each containing three layers. The test cubes used provided 100% coverage of detectable faults. Static encoding was used to encode the test cubes. The compressed test data, i.e. tester storage required for the three architectures explained above is presented in Table 3. As shown in Table 3, there is reduction in the amount of tester data while using *arch2* when compared to *arch1*.

Table 3. Tester Data for the Three Architectures

Design	Test Vectors	Scan Cells	Local Independent Decompressors (*arch1)*		Global Decompressor (*arch2)*		Proposed Daisy-chained Decompressors (*arch3)*		Percentage Reduction in Tester Data
			Tester Data (# of Bits)	Number of Test Elevators	Tester Data (# of Bits)	Number of Test Elevators	Tester Data (# of Bits)	Number of Test Elevators	
A	838	2578	8272	13	6016	21	6016	13	27.27%
B	606	6747	18245	15	11070	21	11070	15	39.33%
C	686	5662	9512	13	6560	21	6560	13	31.03%
D	751	8724	13314	15	9193	21	9193	15	30.95%
E	803	9432	13583	15	10144	21	10144	15	25.32%
F	807	10538	17538	15	12046	21	12046	15	31.31%

As explained earlier, similar benefit is obtained by using *arch3* as well. This is due to the fact that both *arch2* and *arch3* provide flexible use of free variables across layers and the free variables which are not used in encoding test cube of one layer can be used to encode test cubes of other layers. In addition, by using daisy-chained decompressors (*arch3*), the number of test elevators required is less compared to *arch2*, since the decompressors are local to each layer.

5. Conclusion

By using the daisy-chained decompressor architecture, an increase in compression can be achieved, with efficient usage of test elevators. Experimental results are presented in which the proposed architecture is compared with the conventional architecture. In addition, an implementation is proposed with a serializer-deserializer coupling the test elevators to the decompressors, which further reduces the number of test elevators required to implement the test architecture.

Acknowledgement

This research was supported in part by the National Science Foundation under Grant No. CCF-1217750.

References

[Janicki 12] Janicki, J.; Kassab, M.; Mrugalski, G.; Mukherjee, N.; Rajski, J.; Tyszer, J., "EDT Bandwidth Management in SoC Designs," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems,* Vol. 31, No. 12, pp. 1894-1907, Dec. 2012.

[Khoche 02] Khoche, A.; Volkerink, E.; Rivoir, J.; Mitra, S, "Test vector compression using EDA-ATE synergies," *Proc. of VLSI Test Symposium,* pp. 97-102, 2002.

[Kinsman 10] Kinsman, A.B.; Nicolici, N., "Time-Multiplexed Compressed Test of SOC Designs," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems,* Vol. 18, No. 8, pp. 1159-1172, Aug. 2010.

[Konemann 01] Konemann, B.; Barnhart, C.; Keller, B.; Snethen, T.; Farnsworth, O.; Wheater, D., "A SmartBIST Variant with Guaranteed Encoding," *Proc. of Asian Test Symposium*, pp. 325-330, 2001.

[Jiang 12] Jiang. L.; Xu. Q.; Chakrabarty, K.; Mak, T.M., "Integrated Test-Architecture Optimization and Thermal-Aware Test Scheduling for 3-D SoCs Under Pre-Bond Test-Pin-Count Constraint", *IEEE Trans. on Very Large Scale Integration (VLSI) Systems,* Vol. 20, No. 9, pp. 1621-1633, Sep. 2012.

[Larsson 08] Larsson, A.; Larsson, E.; Chakrabarty, K; Eles, P.; Zebo Peng, "Test Architecture Optimization and Test Scheduling for SOCs with Core-Level Expansion of Compressed Test Patterns", *Proc. of Design, Automation and Test in Europe,* pp. 188-193, 2008.

[Lee 13] Lee, Y.-W.; Touba, N.A., "Unified 3D test architecture for variable test data bandwidth across pre-bond, partial stack, and post-bond test," *Proc. of Defect and Fault Tolerance Symposium,* pp. 184-189, 2013.

[Marinissen 09] Marinissen, E.J.; Zorian, Y., "Testing 3D chips containing through-silicon vias," *Proc. of International Test Conference,* pp. 1-6, 2009.

[Muthyala 13] Muthyala, S.S.; Touba, N.A.; "SOC test compression scheme using sequential linear decompressors with retained free variables," *Proc. of VLSI Test Symposium,* pp. 1-6, 2013.

[Noia 10] Noia, B.; Goel, S.K.; Chakrabarty, K.; Marinissen, E.J.; Verbree, J., "Test-architecture optimization for TSV-based 3D stacked ICs," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems,* Vol. 30, No. 11, pp. 1705-1718, Nov. 2011.

[Touba 06] Touba, N.A., "Survey of Test Vector Compression Techniques", *IEEE Design & Test Magazine,* Vol. 23, Issue 4, pp. 294-303, Jul. 2006.

[Wang 04] Wang, L.-T.; Wen, X.; Furukawa, H.; Hsu, Fei-Sheng; Lin, Shyh-Horng; Tsai, Sen-Wei; Abdel-Hafez, K.S.; Wu, S., "VirtualScan: a new compressed scan technology for test cost reduction," *Proc. of International Test Conference,* pp. 916-925, 2004.

[Wang 05] Wang, L.-T.; Abdel-Hafez, K.S.; Wen, X.; Sheu, B.; Wu, Shianling; Lin, Shyh-Horng; Chang, Ming-Tung, "UltraScan: using time-division demultiplexing/multiplexing (TDDM/TDM) with VirtualScan for test cost reduction," *Proc. of International Test Conference,* pp. 946-953, 2005.

[Wu 08] Wu, X.; Chen, Y.; Chakrabarty, K.; Xie, Y., "Test-access mechanism optimization for core-based three-dimensional SOCs," *Proc. of International Conference on Computer Design,* pp. 212-218, 2008.

[Xu 05] Xu, Q.; Nicolici, N., "Resource-Constrained System-on-a-Chip Test: A Survey", *IEE Proc. Computers & Digital Techniques*, Vol. 152, Issue 1, pp. 67-81, Jan. 2005