[6] ——, "Optimized pipelined networks of associative and commutative operators," *IEEE Trans. Computer-Aided Design.*, vol. 13, no. 11, Nov. 1994.
[7] K. K. Parhi, "High-level algorithm and architecture transformations for DSP synthesis," *J. VLSI Signal Processing*, no. 9, pp. 121–143, 1995.
[8] M. Janssens, F. Catthoor, and H. De Man, "A specification invariant technique for operation cost minimization in flow–graphs," in *Proc. 7th Int. Symp. High-Level Synthesis*, 1994, pp. 146–151.
[9] ——, "A specification invariant technique for regularity improvement between flow–graph clusters," in *Proc. European Design Automation Conf.*, 1996, pp. 138–143.
[10] T. Kim, W. Jao, and S. Tjiang, "Circuit optimization using carry-save-adder cells," *IEEE Trans. Computer-Aided Design*, vol. 17, pp. 974–984, Oct. 1998.
[11] V. G. Oklobdzija, D. Villeger, and S. S. Liu, "A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach," *IEEE Trans. Comput.*, vol. 45, pp. 294–306, Mar. 1996.
[12] P. Stelling, C. U. Martel, V. G. Oklobdzija, and R. Ravi, "Optimal circuits for parallel multipliers," *IEEE Trans. Comput.*, vol. 47, pp. 273–285, Mar. 1998.
[13] V. G. Oklobdzija and D. Villeger, "Improving multiplier design by using improved column compression tree and optimized final adder in CMOS technology," *IEEE Trans. VLSI Syst.*, vol. 3, pp. 292–301, June 1995.
[14] P. Stelling and V. G. Oklobdzija, "Design strategies for optimal hybrid final adders in a parallel multiplier," *J. VLSI Signal Processing*, vol. 14, no. 3, pp. 321–331, Dec. 1996.
[15] ——, "Implementing multiply-accumulate operation in the multiplication time," in *Proc. Int. Symp. Computer Arithmetic*, 1997, pp. 99–106.
[16] A. Mathur and S. Saluja, "Improved merging of datapath operators using information content and required precision analysis," in *Proc. ACM/IEEE Design Automation Conf.*, 2001, pp. 462–467.
[17] J. Um and T. Kim, "An optimal allocation of carry-save-adders in arithmetic circuits," *IEEE Trans. Comput.*, vol. 50, pp. 215–232, Mar. 2001.
[18] Z. Yu, M.-L. Yu, and A. N. Willson Jr., "Signal representation guided synthesis using carry-save adders for synchronous datapath circuits," in *Proc. ACM/IEEE Design Automation Conf.*, 2001, pp. 456–461.
[19] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Comput.*, vol. 13, pp. 14–17, 1964.
[20] K. Hwang, *Computer Arithmetic: Principles, Architecture and Design*. New York: Wiley, 1979.
[21] E. Swartzlander, *Computer Arithmetic*. Piscataway, NJ: IEEE Press, 1990, vol. 1&2.
[22] N. D. Dutt. High-level synthesis design repositories. [Online]http://ftp.ics.uci.edu/pub/hlsynth
[23] C. Fiduccia and R. Matteyses, "A linear-time heuristic for improving network partitions," in *Proc. ACM/IEEE Design Automation Conf.*, 1982, pp. 175–181.
[24] Synopsys Inc., Design Compiler User Guide, 2000.
[25] LSI Logic Inc., G10-p Cell-Based ASIC Products Databook, 1996.

Test Data Compression Technique for Embedded Cores Using Virtual Scan Chains

Abhijit Jas, Bahram Pouya, and Nur A. Touba

***Abstract*—This paper presents a design-for-test (DFT) technique to implement a "virtual scan chain" in a core that looks (to the system integrator) like it is shorter than the real scan chain inside the core. A core with a "virtual scan chain" is fully compatible with a core with a regular scan chain in terms of both the external test interface *and* tester program. The I/O pins of a core with a virtual scan chain are identical to the I/O pins of a core with a regular scan chain. For the system integrator, testing a core with a virtual scan chain is identical to testing a core with a regular scan chain (no special modes, control signals, or timing sequences are needed). The only difference is that the virtual scan chain is much shorter so the size of the scan vectors and output response is smaller resulting in less test data as well as less test time (fewer scan shift cycles). The process of mapping the virtual scan vectors to real scan vectors is handled inside the core and is completely transparent to the system integrator.**

***Index Terms*—Automatic test equipment, data compression, design for testability, digital system testing.**

I. Introduction

The semiconductor market is becoming increasingly more competitive with shorter design cycles. The rapidly growing number of small fab-less design houses has resulted in an abundance of choice when it comes to buying semiconductor products. Integration of entire systems on a single chip [system-on-chip (SOC)] has significantly helped reduce effective manufacturing costs of semiconductors in recent years. However, the complexity of verifying and testing such SOCs has gone up significantly. Designing SOCs with predesigned and preverified cores is becoming the norm to reduce design turn-around time. These cores could be in-house or purchased from commercial vendors and range from a wide variety of embedded processor cores, memory cores, DSPs, and other ASICs. Multiple commercial vendors sell cores with similar functionality, thereby creating a competitive environment. One characteristic of a core that emerges as an important distinguishing factor is test complexity. Given two cores with similar functionality, the core that can be thoroughly tested with the smallest amount of test data and the simplest tester program has a significant competitive advantage because it reduces manufacturing test costs.

In this paper, a novel design-for-test (DFT) technique that allows core vendors to reduce the test complexity of the core they are trying to market is presented (preliminary results were published in [14]). The idea is to create a core with a "virtual scan chain" which looks (to the system integrator) like it is shorter than the real scan chain inside the core (as illustrated in Fig. 1). The I/O pins of the core with the virtual scan chain are identical to the I/O pins of a core with a real scan chain.

Manuscript received October 5, 2002; revised May 19, 2003. The work was supported in part by the National Science Foundation under Grant MIP-9702236, and in part by the Texas Advanced Research Program under Grant 1997-003658-369.

A. Jas was with the Computer Engineering Research Center, Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712-1084 USA. He is now with Intel Corporation, Austin, TX 78746 USA.

B. Pouya was with the Computer Engineering Research Center, Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712-1084 USA.

N. A. Touba is with the Computer Engineering Research Center, Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712-1084 USA (e-mail: touba@ece.utexas.edu).

Digital Object Identifier 10.1109/TVLSI.2004.830911

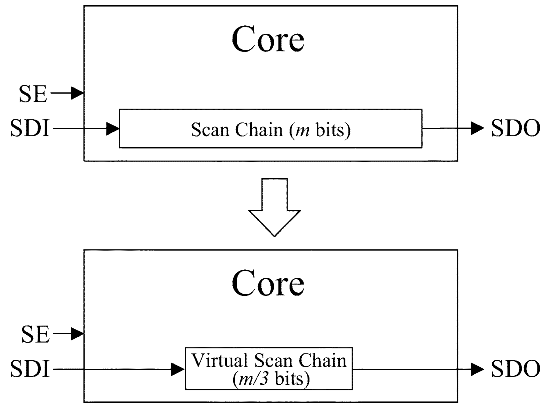

Fig. 1. Concept of virtual scan chain.

There is a scan data in (SDI) pin, scan data out (SDO) pin, and a scan enable (SE) pin to control the scan chain. This means that the external test interface of a core with a virtual scan chain is identical to one with a regular scan chain. Moreover, there are no special modes, external control signals, or timing sequences that are required (as are commonly required in other test data compression schemes), so the tester program is exactly the same as is used for conventional scan testing. This is a very significant advantage as it implies that the system integrator does not need to change existing test methodologies, tools, and flows to accommodate a core with a virtual scan chain in his SOC design. The only difference is that a core with a virtual scan chain appears to have much shorter scan chains inside the core and results in shorter test vectors and output responses. This directly translates to reduced test data volume and reduced test time (fewer scan shift cycles). The real scan chain inside the core, however, is longer than the virtual scan chain. The process of mapping the virtual scan vectors to real scan vectors is handled inside the core and is completely transparent to the system integrator. One nice feature of a virtual scan chain is that it hides intellectual property (IP) because it encodes the core's scan vectors and disguises the real number of scan cells.

Without loss of generality, a single virtual scan chain replacing a single real scan chain will be described, but obviously, if the core has multiple real scan chains then they could be replaced with multiple virtual scan chains.

II. Related Work

The problem of reducing test data and test time for SOCs in general and cores in particular, have been attacked from several different angles in recent literature. These techniques can broadly be classified into the following categories.

1) Techniques that relate to compression in a full-scan environment. These include scan chain architecture techniques [1], as well as the use of hardware decompressors to decode precompressed stored test patterns [3], [4], [7], [12], and [13]. All of theses techniques apply to cores with regular scan chains and, hence, also apply to cores with virtual scan chains since a core with a virtual scan chain is fully compatible to a core with a regular scan chain from the system integrator's point of view.
2) Techniques that are used in an automatic test pattern generation (ATPG) framework to produce test vectors that can more effectively be compressed and later decompressed in hardware [6], [10], [11].

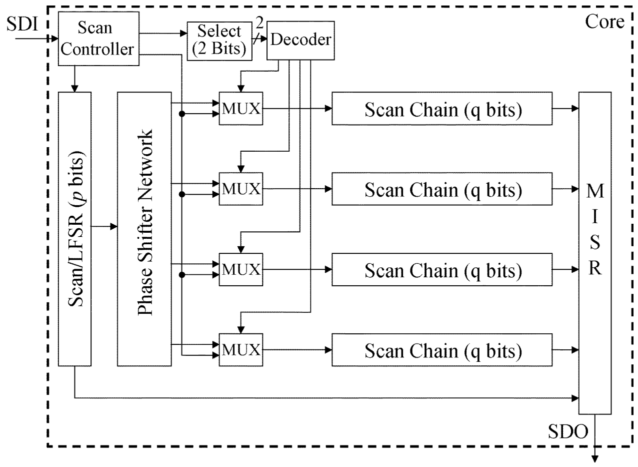

Fig. 2. Example of virtual scan chain that is $p + q + 2$ bits long.

3) Hybrid techniques that use built-in self-test (BIST) structures to aid external testing. Several such techniques are described in [5], [9], [15]–[17], and [20]. The technique described here falls in this category. Two techniques that are closely related to the technique described here are described in [2] and [8]. Reconfiguration techniques for such architectures are described in [18]. In [8], an approach called parallel serial full scan (PSFS) is presented. The idea is to have two modes for loading a scan chain from a single scan data in (SDI) pin: parallel and serial. In parallel mode, the same test vector is shifted into multiple scan chains. In [2], a linear XOR gate based "expander" network is implemented to expand N input channels from the tester to feed M scan channels in the chip $(N < M)$. This results in both test data and test-time reduction. Although conceptually the techniques described here and in [2] are similar, there are several differences in the corresponding scan architectures. Reference [2] puts a restriction on the number of specified bits in each scan shift cycle, whereas the technique described here puts a restriction on the number of specified bits over the entire test vector. This gives more flexibility to the ATPG tool or any static compaction tool that is used as a post-processor on the ATPG generated test cubes. The technique described in [8] is a special case of the technique described in [2] (degenerate case of the expander network). In general, the technique in [8] requires the use of two different modes, serial and parallel, and thus, is not fully compatible with normal conventional scan testing as the technique described here is (i.e., it is not transparent as the user needs to modify the tester program to account for this).

III. Implementing a Virtual Scan Chain

Having described the concept of a virtual scan chain, now the details of how it can be implemented inside the core will be discussed. It is best explained with an example. Fig. 2 shows how an m-bit long real scan chain can be implemented as a $(p + q + 2)$-bit long virtual scan chain, where $m = p + 4q$ and $p + q + 2 < m$. The real scan chain (consisting of m scanned flip-flops in the core) is divided into one p-bit linear feedback shift register (LFSR) and four smaller scan subchains. Each scan subchain is q bits long. The LFSR is formed by reconfiguring p bits of the original real scan chain into an LFSR during testing. Only $p + q + 2$ bits will be shifted in from the SDI pin (since that is the size of each virtual scan vector), and after that the system clock will be applied to capture the response back into the LFSR and the scan subchains. During those $p + q + 2$ scan cycles, all $m = p + 4q$ scan

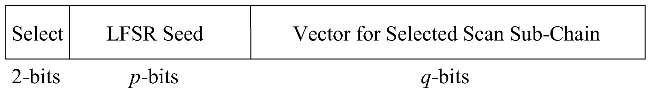

Fig. 3. Format of virtual scan vector.

elements must be filled with the real scan vector. The way this is done will now be described.

There is a scan controller which is a simple finite state machine (FSM) that controls which of the scan subchains/LFSR the SDI input is being shifted into during each scan cycle. During the first two scan cycles, the SDI input is shifted into a 2-bit select (SEL) register. During the next p scan cycles, the SDI input is shifted into the portion of the scan chain which will later on be configured as a p-bit linear feedback shift register (LFSR). During the last q scan cycles, the SDI input is shifted into one of the 4 q-bit scan subchains (selected by the 2-bits shifted into the SEL register earlier). The format of the virtual scan vector is shown in Fig. 3. Once the p bits have been loaded, the LFSR is configured to operate for the next q scan cycles. The remaining 3 q-bit scan subchains are loaded from the p-bit LFSR through a phase-shifter network. The LFSR operates in autonomous mode during the final q scan cycles. Thus, at the end of $p + q + 2$ scan cycles, all $m = p + 4q$ scan elements are loaded with a test vector. Note that the state of the LFSR after q scan cycles form part of the test vector. The seed (initial state) that is loaded into the LFSR is "expanded" by running the LFSR in autonomous mode. The process of finding a virtual scan vector that will map to a desired test vector is described in the next section.

As the next virtual scan vector is shifted in, the response of the previous vector gets shifted out. The responses of the LFSR and the multiple subchains need to be compacted as they are shifted out since there is only one SDO output. This can be done using a multiple-input signature register (MISR) with its feedback line connected to the SDO output. This will make the output response of the virtual scan chain look like that of a regular scan chain. For the last test vector in the test set, the last few bits of the output response information may get stuck in the flip-flops of the MISR and not get shifted out. This problem can be solved by adding an extra "dummy" test vector to the end of the virtual scan vector test set to effectively "flush" the contents of the MISR out. Using a MISR introduces the possibility of losing fault coverage due to aliasing. This can be avoided by either doing fault simulation with the MISR when generating the test vectors, or by choosing the size of the MISR so that the probability of aliasing is sufficiently low.

Note that while the example in Fig. 2 shows 4 q-bit scan subchains, any number of such subchains can be used (e.g., 8, 16, 32, etc.). Selecting the number of subchains and the size of the LFSR will be discussed in Section V after the test generation procedure is described.

IV. Constructing Virtual Scan Vector Test Set

In this section, a procedure for finding a minimal set of virtual scan vectors that provides the desired fault coverage is described. It is assumed that the virtual scan chain architecture (i.e., the number of scan subchains and size of the LFSR) has already been selected. The process of choosing a virtual scan chain architecture for a particular core will be discussed in the next section as many of the issues for that relate to the test generation procedure described in this section.

Each virtual scan vector gets mapped to a test vector in the real scan chain by using the LFSR and the phase-shifter network. The idea of expanding an LFSR seed into a scan vector was first proposed in [17]. If the LFSR has k-stages, then a system of linear equations can be solved to find a seed that will generate a particular test cube.

A. Test Generation

The procedure for finding a virtual scan vector test set that provides the desired fault coverage is as follows. First, random test generation is used to find virtual scan vectors that detect the easy-to-detect faults. This is done by simply simulating random virtual scan vectors, and those that detect previously undetected faults are added to the test set. For the hard-to-detect faults, regular ATPG is done to find test cubes (i.e., the unspecified inputs are left as X's). The linear equations for the LFSR are then solved to find a virtual scan vector that will map to the test cube as described in [17]. The size of the LFSR needs be selected to be sufficiently long so that a solution always exists for the linear equations. In [9], it was shown that if the number of stages in the LFSR is 20 greater than the number of specified bits in the test cube, then the probability of finding a seed is 0.999 999. Note that one of the subchains can be selected as the one to be directly loaded from the SDI input, so this reduces the number of specified bits that need to be generated by the LFSR and, thus, allows a smaller LFSR to be used. The procedure for selecting an optimal size for the LFSR is discussed in Section V.

B. Static Compaction

Static compaction involves merging compatible test cubes to reduce the total number of test cubes. Two test cubes are compatible if they do not have conflicting values at any bit position (conflicting values are when one has a specified 1 and the other a specified 0 in the same bit position). Two test cubes are merged by specifying all bit positions in which either has a specified value, hence merging test cubes increases the number of specified bits. For a virtual scan chain, the amount of static compaction that can be done is limited because static compaction specifies additional X's which may cause the resulting test cube to no longer be mappable to a virtual scan vector. As a result, the number of virtual scan vectors required for a particular fault coverage may be more than the number of regular scan vectors. However, because each virtual scan vector is much shorter (has fewer bits), the overall amount of test data is still greatly reduced (as will be shown in the experimental results).

The constrained static compaction procedure for a virtual scan chain must check when two test cubes are merged, that the additional specified bits do not cause the system of linear equations for the LFSR to become unsolvable. A threshold on the total number of specified bits (which depends on the size of the LFSR) can be used as a heuristic on whether the linear equations will be solvable. Static compaction of test cubes can proceed until the total number of specified bits for all the scan subchains, minus the one scan subchain that is directly fed from the SDI pin, exceeds the threshold. Whichever scan subchain has the largest number of specified bits can be chosen as the one that is directly fed from the SDI pin to maximize the amount of static compaction.

After static compaction, the linear equations for each test cube can be solved to find a virtual scan vector that maps to the test cube. The virtual scan vector for each test cube can then be expanded (by simulating the LFSR) to the corresponding fully specified test vector which can then be fault simulated to possibly drop additional undetected faults.

V. Selecting Virtual Scan Chain Architecture

The three important parameters that define a virtual scan chain architecture are the number of scan chains n, the size of the LFSR p, and the total number of test cubes after constrained static compaction N. Once these three parameters are determined, the total amount of virtual scan test data is given by $(p + q + s) * N$, where $s = \lceil \log_2 n \rceil$

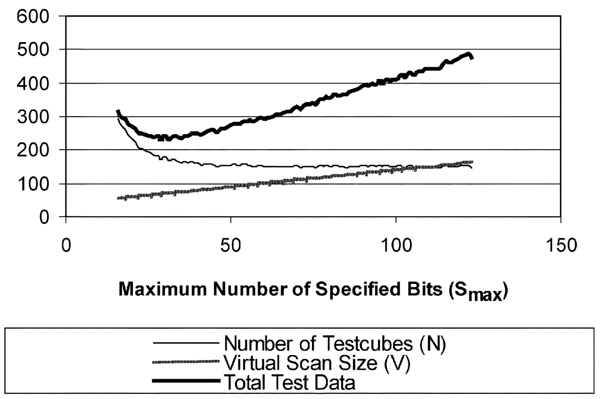

Fig. 4. Variation of V, N and *Total Test Data* with $S_{\max}$, for s5378 with $n = 16$ scan chains.

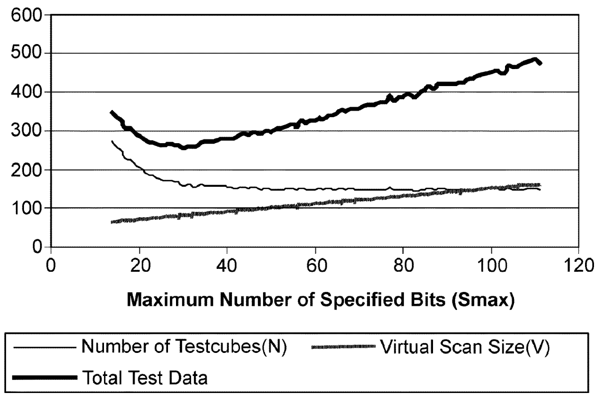

Fig. 5. Variation of V, N and *Total Test Data* with $S_{\max}$, for s5378 with $n = 8$ scan chains.

is the width of the select register. The length of each scan subchain q ($q = \lceil (m - p)/n \rceil$) gets fixed for a given n, where m is the total number of scan elements in the core. The objective in selecting a virtual scan architecture is to minimize the total test data. The total test data can be expressed as $V * N$, where $V = p + q + s$ is the length of each virtual scan vector. Note that for a given n, V depends directly on p since both q and s depend only on n and p. So for a given number of scan subchains n, the total test data will depend on the tradeoff between the number of test cubes N and the size of the LFSR p.

N can be decreased by statically compacting the test cubes. However, as the test cubes are statically compacted, the number of specified bits in any particular test cube increases. This implies that to solve the linear equations to find an LFSR seed, which when expanded will generate the given test cube, the size of the LFSR, p, needs to be increased. So there is a tradeoff between N and p. Let S_i denote the number of specified bits for test cube T_i that needs to be solved for when finding an LFSR seed. Note that S_i is less than the total number of specified bits in T_i because it does not include the specified bits in the one subchain that is directly fed by the SDI pin. Let $S_{\max} = \max\ S_i$, for all test cubes T_i belonging to the test set. The size of the LFSR p that is needed to solve the equations for all the test cubes is directly determined by $S_{\max}$. The larger the $S_{\max}$, the larger that p must be, which in turn means that the virtual scan vector size V also increases. However an increase in $S_{\max}$ causes more static compaction and consequently decreases N. Thus finding the optimal virtual scan architecture involves finding the optimal value of $S_{\max}$ which minimizes the total test data. The impact of $S_{\max}$ on V, N, and the total test data is explained below with an example.

Fig. 4 below plots the variation of the total number of test cubes (N), the virtual scan length (V), and the total test data with $S_{\max}$. The graph shown is for the circuit s5378 with $n = 16$ scan subchains. Initially the test set (with no static compaction) has $N = 291$ test cubes with $S_{\max} = 16$. As $S_{\max}$ is increased more static compaction occurs, which decreases N. This is illustrated in Fig. 4 by the middle line. Initially, with increasing $S_{\max}$, N decreases rapidly up to a point when increasing $S_{\max}$ has little or no effect on N. Finally, when the constrained static compaction degenerates into unconstrained static compaction (due to a sufficiently large value of $S_{\max}$) the curve flattens out completely.

To exactly determine the size of the LFSR for a given $S_{\max}$, a system of linear equations needs to be solved [17]. However, in [9], it was shown that if the number of stages in the LFSR is 20 greater than the number of specified bits, the probability of finding a seed is 0.999 999. Thus $S_{\max} + 20$ gives an estimate of the size of the LFSR p that is required to successfully find seeds for all the test cubes. This estimate for p is used to plot the variation of the virtual scan length $V = p + q + s$, with $S_{\max}$ in Fig. 4 (the bottommost line). Note that plotting this curve avoids the expensive computation of solving the linear equations to actually find a seed by suitably estimating the LFSR size p. This is an important feature of this methodology that makes the process computationally efficient.

Once V and N have been plotted, the curve for the total test data $(V * N)$ can be very easily obtained. This is shown in Fig. 4 by the topmost line. The total test data value has been scaled down by 200 to fit the y-axis scale (scaling leaves the shape of the curve unchanged), so that the comparison can be made on the same graph for visual clarity. It is clearly seen from the figure that the plot of total test data has a very well defined global minimum. This is the point that defines the optimum virtual scan architecture for s5378 with 16 scan subchains. This happens to be the point for which $S_{\max} = 31, N = 167, V = 66$ and $\text{total test data} = 22\,044$ bits.

Once the optimal virtual scan architecture has been obtained from an estimate of the LFSR size, further optimization can be done by actually solving the system of linear equations for all the test cubes. The choice of $S_{\max} + 20$ as the LFSR size virtually guarantees success in finding seeds for the equations. However, in most cases it is a very conservative estimate. As will be seen later in the experimental results section, in most cases the equations can be solved with fewer than $S_{\max} + 20$ bits for the LFSR. It turns out for this example that all the equations can be solved by $S_{\max} + 7$ resulting in a total test data of 17 702 bits with virtual scan length $V = 53$. Nevertheless, the estimate $S_{\max} + 20$ allows quick computation of the optimal values of the parameters for the virtual scan architecture by avoiding the expensive computation for solving the system of linear equations.

All the analyses in the above paragraphs were carried out using a given number of scan subchains n. Similar analyses can be easily carried out for other values of n, and the best value (determined by the total test data) can be chosen. Fig. 5 below shows the same plots for $n = 8$ scan subchains. The best value of total test data obtained in this case is 20 160 with $S_{\max} = 30, V = 63, N = 160$. In this case also, the LFSR size suffices to be $S_{\max} + 7$. However, since $n = 16$ yields better results than $n = 8$, the choice for the optimal virtual scan architecture will be 16 scan subchains. Note that although the final result is different in the two cases, the nature of the graphs is very similar for both the figures. Finding the optimal virtual scan architecture involves finding the low point on the total test data graph.

TABLE I
RESULTS COMPARING VIRTUAL SCAN CHAIN WITH REGULAR SCAN CHAIN

Circuit Name	Regular Scan Chain: Scan Size (bits)	Regular Scan Chain: Compactest Num. of Test Vectors	Regular Scan Chain: Test Data (bits)	Virtual Scan Chain: n	Virtual Scan Chain: Scan Size (bits)	Virtual Scan Chain: Num. Test Vectors	Virtual Scan Chain: Test Data (bits)	% Reduction of Scan Cycles
s5378	214	111	23754	16	53	167	8851	62.7
s9234	247	150	37050	32	67	205	18735	62.9
s13207	700	237	165900	32	65	264	17160	89.7
s15850	611	123	75153	32	91	160	14560	80.7
s38417	1664	95	158080	32	318	117	37206	76.5
s38584	1464	125	183000	32	178	227	40406	77.9

TABLE II
RESULTS SHOWING CPU RUNTIMES FOR FINDING OPTIMAL VIRTUAL SCAN ARCHITECTURE

Circuit Name	Virtual Scan Chain Parameters: n	S_{max} (Min)	S_{max} (Max)	δ	CPU Times (Seconds): Plot	CPU Times (Seconds): Solve
s5378	16	16	66	1	12.2	23.2
s9234	32	56	85	1	2.0	47.0
s13207	32	37	61	1	11.4	49.4
s15850	32	39	88	1	13.4	40.0
s38417	32	86	446	10	44.7	658.3
s38584	32	92	152	4	457.8	853.2

VI. EXPERIMENTAL RESULTS

Experiments were performed for the largest ISCAS 89 benchmark circuits. Table I shows the results comparing a virtual scan chain with a regular scan chain. The fault coverage in both cases is 100% of detectable faults. The test vectors for the regular scan chain were generated with Compactest [19] which uses a powerful dynamic compaction procedure. A virtual scan chain with a certain number of subchains was designed for each circuit and the optimum architecture was chosen as described in Section V. For a regular scan chain, the following are shown: the size of the regular scan chain, the number of test vectors with full (unconstrained) static compaction, and the total amount of test data (that must be stored on the tester). For the virtual scan chain, the following are shown: the number of scan subchains n, the maximum number of specified bits in the statically compacted test vector set $S_{\max}$, the size of the virtual scan chain V (which is much shorter than the real scan chain), the number of test vectors N which results from static compaction under the constraints imposed by the linear dependencies of the LFSRs, the size of the LFSR and the total amount of test data. Lastly, the percentage reduction in the number of scan cycles required for testing each circuit is shown. The percentage is computed as follows:

$$\frac{[(\text{Regular Scan Test Data})–(\text{Virtual Scan Test Data})]}{(\text{Regular Scan Test Data})} \times 100.$$

As can be seen from the results, the number of test vectors is larger for the virtual scan chain because of the constraints on static compaction, however, the number of bits in each vector is much less than that of a regular scan chain. Consequently, the total amount of test data is reduced and the number of scan cycles for testing the circuit is also reduced.

Table II shows the computation times (CPU seconds) for determining the optimum virtual scan architecture for a given number of virtual scan chains (n). The experiments were run on a 1-GHz Pentium III machine with 1-GB main memory running Linux. Column 2 in Table II shows the number of virtual scan chains for a given circuit, columns 3 and 4 show the minimum and maximum value of $S_{\max}$, which were used to plot the curve as described in Section V. Column 5 (δ) is the increment in the value of $S_{\max}$ that was used to compute the various data points for the curve. Thus the curves were plotted using $\lceil (S_{\max} - S_{\min})/\delta \rceil$ data points. Column 8 shows the CPU time for plotting the curve and column 9 shows the CPU time for the final optimization step where the LFSR size is progressively decreased to check when the equations become unsolvable. Note that the time for plotting the curve can be reduced by increasing the increment (δ), and the time for solving the equations could be reduced by using an incremental solver. Both of these approaches could greatly speed up the procedure at the cost of only a minor reduction in the overall compression.

As mentioned earlier, the two techniques that are most closely related to this work are [8] and [2]. The results obtained using virtual scan chains are compared with those obtained in [8] and [2] in Table III. The table compares the total test data volume for virtual scan chains against the best results (test data volume) obtained in the other two techniques. For the two techniques [8] and [2], Table III shows both the total test data volume (in bits) and the percentage improvement obtained by using virtual scan chains over each of these techniques. As can be seen from the table, the results obtained using the virtual scan chain method are better than those obtained in [8] for all the circuits. The virtual scan chain results are better than those of [2] for all but one of the circuits $(s38584)$. The entries "NA" in the table below denote entries for which comparisons could not be made because the cited work didn't have the corresponding results. Note that the numbers in [2] were obtained using 200 internal scan chains (i.e., $n = 200$). One of the disadvantages of using such a large number of internal scan chains is that the hardware overhead for compacting the output response becomes large (large number of XOR gates or a very large MISR).

VII. CONCLUSIONS

A core with a virtual scan chain reduces test costs for the system integrator. Thus, core vendors may find virtual scan chains a means to achieve a competitive advantage in selling their cores. Note that in this work, the default ordering of the scan chains was used. It was assumed that the core designer would want to choose the ordering of the scan chain based on other criteria such as minimizing routing. However, one way to improve the results would be to specially order the scan chain. The scan cells could be partitioned into scan subchains in a way that equally distributes the specified bits in the test cubes in order to minimize the size of the LFSRs and/or maximize the amount of static compaction that can be performed.

TABLE III
RESULTS COMPARING VIRTUAL SCAN CHAIN TOTAL TEST DATA VOLUME WITH THOSE OF [2] AND [8]

Circuit Name	Virtual Scan Chain	[13]		[4]	
		Test Data	% Improvement	Test Data	% Improvement
s5378	8851	14171	37.5	NA	NA
s9234	18735	17441	21.2	NA	NA
s13207	17160	41273	58.4	25344	32.3
s15850	14560	38015	61.7	22784	36.1
s38417	37206	64866	41.6	89856	58.6
s38584	40406	64790	37.6	38976	-3.7

REFERENCES

[1] J. Aerts and E. J. Marinissen, "Scan chain design for test time reduction in core-based ICs," in *Proc. Int. Test Conf.*, 1998, pp. 448–457.
[2] I. Bayraktaroglu and A. Orailoglu, "Test volume and application time reduction through scan chain concealment," in *Proc. Design Automation Conf.*, 2001, pp. 151–155.
[3] A. Chandra and K. Chakrabarty, "Efficient test data compression and decompression for system-on-a-chip using internal scan chains and Golomb coding," in *Proc. Design, Automation, Test in Europe*, 2001, pp. 145–149.
[4] ——, "Frequency-directed run-length codes with application to system-on-a-chip test data compression," in *Proc. VLSI Test Symp.*, 2001, pp. 42–47.
[5] D. Das and N. A. Touba, "Reducing test data volume using external/LBIST hybrid test patterns," in *Proc. Int. Test Conf.*, 2000, pp. 115–122.
[6] R. Dorsch and H.-J. Wunderlich, "Tailoring ATPG for embedded testing," in *Proc. Int. Test Conf.*, 2001, pp. 530–537.
[7] P. Gonciari, B. M. Al-Hashimi, and N. Nicolici, "Improving compression ratio, area overhead, and test application time for systems-on-a-chip test data compression/decompression," in *Proc. Design Automation Test Europe*, 2002, pp. 604–611.
[8] I. Hamzaoglu and J. Patel, "Reducing test application time for full scan embedded cores," in *Proc. Int. Symp. Fault Tolerant Computing*, 1999, pp. 260–267.
[9] S. Hellebrand, J. Rajski, S. Tarnick, S. Venkataraman, and B. Courtois, "Built-in test for circuits with scan based on reseeding of multiple-polynomial linear feedback shift registers," *IEEE Trans. Comput.*, vol. 44, pp. 223–233, Feb. 1995.
[10] H. Ichihara, K. Kinoshita, I. Pomeranz, and S. M. Reddy, "Test transformation to improve compaction by statistical encoding," in *Proc. Int. Conf. VLSI Design*, 2000, pp. 294–299.
[11] H. Ichihara, A. Ogawa, T. Inoue, and A. Tamura, "Dynamic test compression using statistical coding," in *Proc. Asian Test Symp.*, 2001, pp. 143–148.
[12] A. Jas and N. A. Touba, "Test vector compression via cyclical scan chains and its application to testing core-based designs," in *Proc. Int. Test Conf.*, 1998, pp. 458–464.
[13] A. Jas, J. Ghosh-Dastidar, and N. A. Touba, "Scan vector compression/decompression using statistical coding," in *Proc. VLSI Test Symp.*, 1999, pp. 114–120.
[14] A. Jas, B. Pouya, and N. A. Touba, "Virtual scan chains: a means for reducing scan length in cores," in *Proc. VLSI Test Symp.*, 2000, pp. 73–78.
[15] A. Jas, C. V. Krishna, and N. A. Touba, "Hybrid BIST based on weighted pseudo-random testing: a new test resource partioning scheme," in *Proc. VLSI Test Symp.*, 2001, pp. 2–8.
[16] A. Khoche, E. Volkerink, J. Rivoir, and S. Mitra, "Test vector compression using EDA-ATE synergies," in *Proc. VLSI Test Symp.*, 2002, pp. 97–102.
[17] B. Koenemann, "LFSR-coded test patterns for scan designs," in *Proc. European Test Conf.*, 1991, pp. 237–242.
[18] A. R. Pandey and J. H. Patel, "Reconfiguration technique for reducing test time and test data volume in Illinois scan architecture based designs," in *Proc. VLSI Test Symp.*, 2002, pp. 9–15.
[19] I. Pomeranz, L. N. Reddy, and S. M. Reddy, "COMPACTEST: A method to generate compact test sets for combinational circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 1040–1049, July 1993.
[20] M. Sugihara, H. Date, and H. Yasuura, "A novel test methodology for core-based system LSI's and a testing time minimization problem," in *Proc. Int. Test Conf.*, 1998, pp. 465–472.

Improving the Stuck-at Fault Coverage of Functional Test Sequences by Using Limited-Scan Operations

Irith Pomeranz and Sudhakar M. Reddy

***Abstract*—Functional test sequences were shown to detect unique defects in VLSI circuits. This is thought to be due to the fact that they are applied at-speed. However, functional test sequences do not achieve complete stuck-at fault coverage. Therefore, scan-based stuck-at tests, as well as other types of tests, are typically also applied. This increases the amount of test resources required for test application. We describe a procedure for inserting (limited) scan operations into a functional sequence in order to improve its stuck-at fault coverage, thus reducing or eliminating the need for separate scan-based stuck-at tests. Between scan operations, the functional test sequence can still be applied at-speed; however, a higher stuck-at fault coverage is achieved.**

***Index Terms*—Functional test sequences, limited scan operations, scan circuits.**

I. INTRODUCTION

The results reported in [1] and similar studies show that different test sets and different test application schemes, applied to the same scan-based circuit, have unique defects that they detect, i.e., each method detects certain defects that are not detected by any other test set or test application scheme. In particular, functional test sequences were shown to have unique defects that are not detected by any other method. Functional test sequences are designed based on the functional description of the circuit and they do not use scan even if it is available. The characteristic of functional test sequences that makes them effective in testing scan-based sequential circuits is that the test vectors are applied at-speed through the functional path of the circuit. Even if scan is available, a functional test sequence does not employ scan. However, the stuck-at fault coverage of functional test sequences is lower than that achieved by applying a scan-based stuck-at test set. This is due to two reasons: 1) the fault coverage that can be achieved using scan is higher than the fault coverage that can be achieved when scan is not used and 2) some stuck-at faults that can be detected without using scan may not be detected by a functional test sequence designed based on the functional description of the circuit.

Manuscript received June 17, 2002; revised April 13, 2003. The work of I. Pomeranz was supported in part by the National Science Foundation under Grant CCR-0098091 and in part by Semiconductor Research Corporation (SRC) under Grant 2001-TJ-950. The work of S. M. Reddy was supported in part by the National Science Foundation (NSF) under Grant CCR-0097905 and in part by SRC Grant 2001-TJ-949. The paper was presented in part at the International Test Conference, Oct. 2001, pp. 211–220.

I. Pomeranz is with the School of Electrical and Computer Engineering, Purdue University, W. Lafayette, IN 47907 USA.

S. M. Reddy is with the Electrical and Computer Engineering Department, University of Iowa, Iowa City, IA 52242 USA.

Digital Object Identifier 10.1109/TVLSI.2004.830910