Virtual Compression through Test Vector Stitching for Scan Based Designs

Wenjing Rao*
CSE Department
University of California, San Diego
La Jolla, CA 92093
wrao@cs.ucsd.edu

Alex Orailoglu
CSE Department
University of California, San Diego
La Jolla, CA 92093
alex@cs.ucsd.edu

ABSTRACT

We propose a technique for compressing test vectors. The technique reduces test application time and tester memory requirements by utilizing part of the predecessor response in constructing the subsequent test vector. An algorithm is provided for stitching test vectors that retains full fault coverage while appreciably reducing time and tester requirements. The analysis provided enables significant compression ratios, while necessitating no hardware outlay whatsoever, making the technique we propose particularly suitable for SOC testing. The test time benefits necessitate no MISR utilization, ensuring no consequent aliasing loss. We examine a number of implementation considerations for the new compression technique and we provide experimental data that can be used to guide an eventual commercial implementation. Experimental data confirms the significant test application time and tester memory reductions.

1. INTRODUCTION

The increasing ability to fabricate larger and denser chips is fuelling an electronics revolution, providing the ability to place ever increasing functionality on SOCs. Yet our fabrication capabilities are increasingly checkmated by onerous test requirements as density results in a necessity to consider an increased number of possible faults. A most critical concern is the decrease in the ratio of pinouts to area and chip functionality, thus sharply limiting observability and controllability. Traditionally the decrease in controllability and observability has been resolved through the insertion of scan chains. Nonetheless, the size and quantity of scan chains has forced a dramatic increase in test application time. The situation is compounded by the sheer quantity of test data, itself partially an outcome of the large chip size, forcing an overspecification in a large number of scan cells. While the aforementioned constitute problems in all state-of-art chips, they particularly exacerbate our ability to manufacture SOCs which constitute the forefront of the technology envelope nowadays, both in terms of design and fabrication.

Not only does scan insertion help alleviate the controllability and the observability problems, but it furthermore obviates the necessity to apply the test vectors in a particular order, as it essentially turns the sequential ATPG problem into a combinational one. The lack of order in test vector application constitutes an aspect exploitable in test compression literature. Particularly, in combination with the sizable number of unspecified bits in test cubes, especially prevalent in large SOCs, stitching the test vectors together in a beneficial order promises to provide significant reductions both in test vector application time and in test volume, a significant issue by itself for today's costly ATEs, achieving these reductions essentially at no hardware cost.

A number of challenges though need to be addressed so as to deliver the aforementioned benefits. Perhaps, the more challenging one is an inherent dependency in that the stitching needs to tie together the *response* of the previous test stimulus with the subsequent stimulus rather than two adjacent stimuli, as the responses of the circuit under test are captured typically within the same scan chain. An additional major challenge concerns the observability of the fault captured in a test cycle. Typically faults captured in a clock cycle are assumed to be immediately observable, the possible aliasing characteristics of MISRs notwithstanding. The scheme we propose reduces the number of shifting bits, thus reaping test application benefits, but at the same time reduces the number of scan chain bits being shifted out. For certain faults, it may well be that the fault differentiating effect is strictly contained within the part of the scan chain *not* being shifted out. The feasibility of whether such faults can nonetheless be captured with certainty even though their fault effect differentiating characteristics are buried within the part of the scan chain not shifted out constitutes perhaps the second major challenge of the approach we propose.

As the technique we propose represents a departure from the traditional way of test application, its use depends on the resolution of a number of implementation challenges. Such issues include the determination of the number of bits to be shifted in/out at any test cycle, whether the number of bits being shifted should be fixed or variable, the implementation of possible auxiliary hardware in the form of XOR networks to aid controllability and observability, and the order in which faults are handled during the proposed stitched test generation. We provide both a theoretical and experimental analysis of all these options in the latter part of the paper and proceed subsequently to show experimental data and comparisons, in the amount of compression ratios that the technique we propose can deliver on a number of benchmarks.

While the benefits of the technique we propose focus largely on test application time and test volume reduction, a number of other important benefits in comparison to other techniques previously suggested in the literature are to be expected. Primary among them

*The work of the first author is partially supported by a Cal-(IT)^2 Conexant Fellowship.

DATE 2003, March 3–7, 2003, Messe Munich, Germany.

is the dramatic decrease in the hardware necessitated, as all prior compression techniques typically require decompression hardware on chip for test application time reduction and need to couple their decompression techniques with MISR hardware on the output, lest the time benefits that they achieve on the input side end up being completely squandered on the output side. No such output hardware requirement exists for the technique we propose and the frugal hardware requirements if any, on the input side, necessitate no additional hardware on the output side as all the benefits that we deliver accrue as a result of powerful analysis techniques rather than extra hardware.

This paper is organized as follows: In section 2 we present relevant previous work on reducing test time and memory for scan chain based approaches. In section 3, the motivation and basic idea of the proposed approach are illustrated through a simple example. We introduce in section 4 a set of definitions needed to outline the algorithmic framework for test vector generation in section 5. A number of implementation considerations are discussed in section 6. Experimental results are discussed and analysed in section 7, while section 8 draws a set of conclusions.

2. PREVIOUS WORK

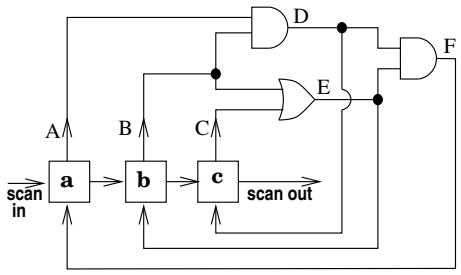
A significant amount of previous work in test compression focuses on breaking a long scan chain into several shorter partitions, thus reducing the length of shifting cycles. These approaches mainly focus on the problem of controllability through the limited number of input pins. The observability of these multiple scan chain partitions through the limited output pins, however, is usually dealt by adding a MISR at the output side, at the possible cost of fault aliasing and significant diagnosis challenges.

In the Virtual Scan Chain approach [1], an LFSR is assigned to each scan chain partition to consequently provide pseudorandom vectors. All the LFSRs are connected to form a chain so that the pseudorandom seed can be shifted in. Each test cycle consists of two stages. First the seeds of all the LFSRs are shifted in. Then test vectors are filled into the scan chain partitions. Among the partitions, only one of them is directly filled in with a test vector from the main input pin, while the remaining partitions are filled by the pseudorandom vector provided by an LFSR. The length of the LFSR chain plus the length of one scan chain partition constitutes the number of bits being shifted in each test cycle.

In a test per clock approach named "designed for high test compression" (DFHTC) core [2], a test pattern is applied to the circuit on every shifting clock. In order to approach this test per clock capability, as a test vector is being shifted into the scan chain, a number of LFSRs are used to provide pseudorandom test patterns for the unshifted part of the scan chain. Thus the data in the scan chain can be applied to the circuit at every clock regardless of whether the expected test vector has been fully shifted into the scan chain.

In a Parallel Serial Full Scan approach [3], a long scan chain is decomposed into several small scan chain partitions. Two test modes, parallel and serial, are provided. In a parallel test mode, the same test vector is shifted into all the scan chain partitions each test cycle to reduce both test time and tester memory. In order to attain full fault coverage, normal serial shifting of the whole scan chain is also retained.

Bayraktaroglu and Orailoglu discuss a compression approach [4] that exploits the unspecified bits of test vectors and applies compressed test vectors into the circuit. The long scan chain is broken into multiple shorter partitions, while a decompression network either on the circuit under test or on the ATE is used to decode test vectors to the multiple scan chain partitions. Through the test compression approach presented by Reda and Orailoglu [5], test application time and test data is compressed by encoding the bits that need to be flipped, thus attaining high compression ratios yet at the cost of decompression hardware insertion.



Test Vector	Result
1 1 0	1 1 1
0 0 1	0 1 0
1 0 0	0 0 0
0 1 0	0 1 0

Figure 1: Circuit With Scan Chain of Length 3 and Test Vectors

An alternative approach attempts to find as many overlaps between test vectors, thus obtaining test data compression [6]. A heuristic algorithm is provided to reorder test vectors in a way that obtains the most overlapping. This work is based on the assumption that two distinct scan chains exist separately for input and output, an unlikely to be fulfilled assumption.

3. MOTIVATION

We illustrate our scheme through a simple example below. The circuit in Figure 1 consists of three gates and a scan chain of length 3. For this circuit, the four test vectors suffice to catch all the faults except for the redundant fault E-F/1. These test vectors and their fault-free results are also shown in Figure 1.

Application of this set of four test vectors under the traditional scan chain approach necessitates shifting each test vector into the scan chain, and then applying the vector to the circuit under test. The response is stored back into the scan chain and is shifted out while the subsequent test vector is shifted in. At each test cycle, three bits are shifted. The total test time consists of $3 \times 5 = 15$ shift cycles, while the memory requirement is $3 \times 2 \times 4 = 24$ bits.

An analysis of the given test vectors and their results shows it is possible to form the last three test vectors by only shifting in two bits each time. The third bit is attained from the remaining one bit of the preceding response in the scan chain. As the two bits are shifted out, the leftmost bit is shifted to the rightmost scan cell. The first test vector 110 generates the result 111. For the second cycle, we only need to shift in two bits 00, which together with the remaining 1 in the scan cell **c** make up the second test vector 001. The third cycle and the fourth cycle operate similarly by shifting 10 to obtain the third test vector 100 and shifting 01 to obtain 010, the last test vector.[1]

While controllability can be thus achieved by stitching the response of the previous test vector, observability of fault effects requires additional attention. Shifting out only a fragment of the scan chain each time may lead to the existence of *hidden faults*, faults that only produce an effect in the part remaining in the scan chain and thus not immediately observable. Yet observability of a fault does not hinge solely on direct observation of the fault effect differentiating scan cell. If the fault effect differentiating scan cell is not shifted out, then the fault will cause at least the subsequent test vector to differ from the intended one, thus making it all the more likely that the mutated test stimulus will cause the fault to be subsequently caught. We show the effect of such hidden faults in the next

[1]In the example given, the test vectors have been artificially ordered for ease of illustration to induce the relationship described. Typically, no order among vectors needs to be imposed in a scan chain based design in any case.

scan in	110	00		10		01		
fault	cycle 1		cycle 2		cycle 3		cycle 4	
list	TV	RP	TV	RP	TV	RP	TV	RP
correct	110	111	001	010	100	000	010	010
F/0	110	***011***	***000***	***000***				
F/1	110	111	001	***110***	***101***	***110***		
D-F/1	110	111	001	***110***	***101***	***110***		
E-F/1	110	111	001	010	100	000	010	010
D/0	110	***010***						
D/1	110	111	001	***111***				
B-D/1	110	111	001	010	100	***001***		
A/1	110	111	001	010	100	000	010	***111***
B/0	110	***000***						
B/1	110	111	001	010	100	***111***		
E/0	110	***001***						
B-E/0	110	***001***						
C/0	110	111	001	***000***				
E/1	110	111	001	010	100	***010***		
E-b/0	110	***101***						
E-b/1	110	111	001	010	100	***010***		
D-c/0	110	***110***						
D-c/1	110	111	001	010	100	***001***		

TV: Test Vector; RP: Response

Table 1: Fault Behavior Table

paragraph. Table 1 shows the detailed situation of each fault being caught for the example being discussed. Each row keeps track of every fault's behavior through the four testing cycles. The behavior of the fault-free circuit is shown in the uppermost row. For each test cycle, the test vector and the corresponding response are given. An italic bold number indicates a behavior differing from the fault-free circuit. The behavior of a fault that has been observed is no longer tracked in the circuit, indicated by a blank in the remaining part of the row in the table.

With the first test vector 110, seven faults (F/0, D/0, B/0, E/0, B-E/0, E-b/0, D-c/0) generate results differring from the fault-free result. Among these seven faults, six (D/0, B/0, E/0, B-E/0, E-b/0, D-c/0), display responses differring from the fault free result in the last two bits, resulting in their being caught as they are shifted out. The fault F/0, however, generates the result 011, which differs from the correct result 111 only in the bit remaining in the scan chain. Thus, unlike the other six faults, fault F/0 will not be immediately observable and will instead become a hidden fault. Nevertheless, the unobservable fault effect differentiating response bits induce in turn a differentiation on the subsequent test vector to be applied. This can be illustrated by noticing that the second test vector 000, applied to the circuit with the fault F/0, will generate the response 000 in the second test cycle. The expected test vector 001, applied to the fault free circuit, will generate the response 010 instead. These two results differ in the second bit, thus ensuring the observability of the fault effect.

Similarly, in the second test cycle, four more faults generate results differring from the correct one. Among these four faults, F/1 and D-F/1 become hidden faults as their test response in the last two bits, 01, is identical to that of the correct circuit. However, these two faults result in a mutated third test vector 101. The result of 101 in the third test cycle under either of these two faults is 110, which differs from the fault free result 000 in the second bit. Thus these two hidden faults get caught in the third cycle.

We can see from Table 1 that after four test cycles all the faults except the redundant E-F/1 have been caught, thus ensuring perfect attainable fault coverage. The total test time with this new approach is $3 + 2 \times 4 = 11$ shift cycles. The memory requirement is $3 + 2 \times 3 + 2 \times 4 = 17$ bits. Compared to the 15 shift cycles and 25 bits memory requirement of the original approach, the new scheme reduces test time by 27% and test memory requirement by 32% in this simple example. There is no additional hardware overhead nor any loss of fault coverage.

4. DEFINITIONS

To generate test vectors with this new approach, we need to keep track of the behavior of faults, such as which faults have been caught, which faults constitute the current hidden fault set, when a hidden fault will become observable, and so on. We classify the set of all faults into three distinct sets according to their behavior in response to the test stimuli. Each fault at any time belongs to one and only one of the three sets.

- f_c: the *caught fault set*, which contains all the faults that have already been captured.
- f_h: the *hidden fault set*, which contains all the current hidden faults. The current response for a hidden fault differs from that expected for a fault free circuit, but any such differentiation is contained strictly within the bits remaining within the scan chain and not shifted out. It is necessary for each fault in this set to capture the faulty response so as to trace the corresponding subsequent test vector actually being applied.
- f_u: the *uncaught fault set*, which contains all the faults that have not been differentiated by any test vectors applied up to that point.

For a fault free circuit $C_{correct}$, we denote its response under a test vector T as $R_{correct} = C_{correct}(T)$. The response of a faulty circuit with the fault F under the test vector T is denoted as $R_F = C_F(T)$. For an existing hidden fault, f_e, we denote the subsequent test vector affected by f_e to be T_{f_e}. Also, the circuit with fault f_e is denoted as C_{f_e}. Thus the response of the next test vector under the effect of hidden fault f_e is denoted as $R_{f_e} = C_{f_e}(T_{f_e})$.

5. ALGORITHMIC FRAMEWORK

We present in this section an algorithm for generating a sequence of test vectors for the proposed approach. The flowchart of the algorithm is shown in Figure 2.

At the beginning of test vector generation, the hidden fault set f_h and the caught fault set f_c are empty, while the uncaught fault set f_u contains all the irredundant faults in the circuit. The algorithm iterates until all the faults are caught, that is, f_u and f_h become both null and f_c equals the whole fault set. In each iteration a new test vector is generated that incorporates the remaining response of the last cycle in the scan chain.

In each iteration, when a new test vector $T_{correct}$ is generated, the effect of $T_{correct}$ on the fault sets is analysed. All the faults that $T_{correct}$ can differentiate from the correct circuit should be put into f_c or f_h according to the location of the differentiating part. Also, instead of $T_{correct}$, every existing hidden fault f will have its own T_f and will generate a possibly distinct response R_f under the stimulus of the mutated test vector. Some of the hidden faults will be driven to become observable, while others may generate exactly the same response as the correct circuit, thus becoming uncaught faults.

For every fault from the set f_u that the new test vector $T_{correct}$ is able to differentiate, if there is any differentiating part that resides in the shifted out part, the fault should be regarded as a caught fault and put into f_c. Otherwise, it becomes a hidden fault, and is put into f_h.

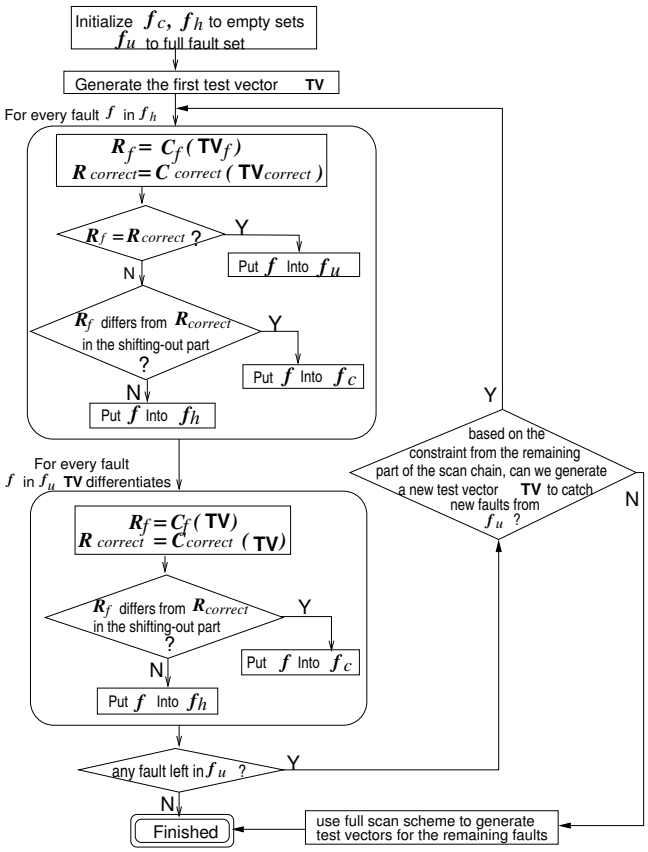


Figure 2: Algorithm flowchart

Secondly, every fault f in the hidden fault set f_h is to be analysed and should be placed into a proper set. With the fault f in the circuit, the new test vector T_f will generate the response R_f, which is compared to the correct response $R_{correct}$. According to the difference between $R_{correct}$ and R_f, fault f will possibly get caught, remain hidden, or become an uncaught fault again. The three situations are discussed below:

- R_f differs from $R_{correct}$ in the part being shifted out, making f observable and resulting in f being placed in f_c.
- R_f differs from $R_{correct}$ only in the part that remains in the scan chain. In this case, f continues to be a hidden fault and is remanded to the set f_h. Its faulty response is updated.
- R_f is exactly the same as $R_{correct}$, resulting in the loss of observability of fault f and its being placed back into the set f_u.

While faults can circulate between f_u and f_h, f_c will consistently increase in size. The size of f_h fluctuates typically across test vector application iterations, but tends to decrease overall as more test vectors are applied. If no test vector can be generated to catch any new fault from f_u, a traditional fully shifting scheme can be used to complete test vector generation for all the remaining faults in f_u.

6. IMPLEMENTATION CONSIDERATIONS

The proposed scheme reduces test time and memory by efficiently stitching test vectors to obtain a considerable portion of overlapping in the scan chain, thus shortening the number of bits shifted in and out each test cycle. The implementation of this idea, however, demands increased attention on fine-tuning a number of factors. Since the test vector generation is based on the specified bit constraint imposed by the previous response, together with the limited shifting-out bits, controllability and observability need to be exploited for a better result.

6.1 Size and Type of Shift

Perhaps a highly important issue to consider in applying this technique is the number of bits to be shifted every cycle. Fixing the shifted bits to a small number each cycle lowers the test time and memory, while a higher number may increase controllability and observability, thus generating test vectors that can catch more faults, yet at the cost of increased test application time per vector.

An alternative strategy is to change the number of shifted bits. At the beginning stage of the test vector generation algorithm, easy-to-test faults dominate. The number of bits shifted out can be set to a very small fraction of the scan chain, thus obtaining maximum decrease in testing time in terms of shifting cycles. When no test vector can be generated to catch any new faults due to the constraints implied by the large fixed part of the scan chain, the shifting bits can be increased to obtain more controllability and observability.

A benefit conferred by the variable shift strategy is a reduction in the amount of restrictions imposed by particular patterns of the fixed part. A circuit may tend to produce a certain value on a particular output bit most of the time. When shifting a constant number of bits each cycle, these relatively fixed patterns will be always fixed in the same bit location in the remaining part of the scan chain, thus leading to fewer possibilities of generating test vectors with increased variability. Allowing the number of bits shifted each cycle to change adds increased diversity on the patterns in the remaining part, thus increasing the variety of test vectors to be generated for catching more faults.

6.2 Hidden Fault Observability

The limitation on observability imposed by shifting out a part of the scan chain is embodied in the existence of hidden faults. In order to improve the observability, as many as possible hidden faults should be driven to the caught fault set f_c. Or, even if the fault effect is not observable through the shifted out part, its effect should be preserved to later cycles, thus improving the chances of its being caught through the stimuli later on. Since the differentiating part of a hidden fault f remains within the scan chain and will constitute a distinct next test vector T_f, when the response of T_f is written back into the scan chain, the differentiating part existing in T_f will be overwritten by the response $R_f = C_f(T_f)$. Thus whether the effect of the hidden fault f is preserved depends on whether R_f is different from $R_{correct}$.

A circuit may not be sensitive on the bits that T_f differs from $T_{correct}$. Furthermore, if f itself is not able to be differentiated by the stimulus of $T_{correct}$ or T_f, then most likely the hidden fault f will generate exactly the same result as the fault-free circuit and become an uncaught fault, because of the same two responses $R_f = C_f(T_f)$ and $R_{correct} = C_{correct}(T_{correct})$.

A way to keep the differentiating part of hidden faults from being eliminated in this manner is to XOR the response with the existing test vector in the scan chain. We denote this strategy as *Vertical XOR* (VXOR) and illustrate an example of the vertical XOR versus the non-XOR implementation in Figure 3.

With the vertical XOR scheme, a hidden fault f will be eliminated if and only if the condition $R_{fault} \oplus T_{fault} = R_{correct} \oplus T_{correct}$ holds. This situation happens when the fault differentiating part of R_f and $R_{correct}$ is exactly in the same bit position as

		3/8 info					5/8 info					7/8 info					variable shift		
circ	aTV	shift	TV	ex	m	t	shift	TV	ex	m	t	shift	TV	ex	m	t	TV	m	t
s444	32	5/21	14	23	0.88	0.82	11/21	29	3	0.64	0.57	18/21	32	0	0.88	0.86	54	0.73	0.53
s526	61	5/21	14	23	0.88	0.82	11/21	58	5	0.66	0.58	18/21	59	0	0.85	0.83	99	0.72	0.53
s641	58	/	/	/	/	/	1/19	34	25	0.80	0.46	13/19	39	2	0.62	0.49	56	0.68	0.24
s953	88	/	/	/	/	/	11/29	88	0	0.63	0.38	23/29	88	0	0.88	0.79	96	0.52	0.14
s1196	137	/	/	/	/	/	6/18	138	0	0.63	0.34	14/18	139	0	0.89	0.79	135	0.49	0.10
s1423	68	21/74	36	38	0.76	0.71	42/74	44	28	0.82	0.78	63/74	56	1	0.73	0.72	131	0.63	0.43
s5378	255	45/179	62	211	0.92	0.89	99/179	118	137	0.83	0.79	152/179	225	0	0.77	0.75	274	0.57	0.45
s9234	380	60/228	48	348	0.96	0.95	127/228	182	212	0.84	0.82	194/228	266	3	0.61	0.60	478	0.68	0.63
Ave	135		35	129	0.88	0.84		86	51	0.73	0.59		113	1	0.78	0.73	165	0.63	0.38

Table 2: Varying the Size and Type of Shifting

the differentiating part of T_f and $T_{correct}$. The cost of additional hardware with the vertical XOR scheme scales with the length of the scan chain in the circuit, as it necessitates the insertion of an XOR gate for each scan cell.

The vertical XOR scheme tries to increase observability by preserving the hidden fault effect within the scan chain for later cycles. We denote an alternate strategy that aims at increasing observability as *Horizontal XOR*. The strategy attempts to increase the number of hidden faults observable at each cycle but requires only a small fixed number of XOR gates. Instead of shifting out the data in the last scan cell each cycle, the horizontal XOR scheme XORs several scan cells for shifting out.

Figure 4 shows an example of the horizontal XOR scheme. In this example, three scan cells are XORed for shifting out. Therefore shifting out one third of a scan chain will make most of the hidden faults observable. By shifting out the XOR result of multiple scan cells, this scheme effectively improves the observability of hidden faults.

6.3 Selection of test vectors

In the algorithm described above, we generate a test vector each cycle to catch new faults from f_u under the constraint of having a part of the scan chain fixed to a certain value. Various schemes can be applied in regards to the selection of a proper test vector.

A greedy way of choosing a test vector is to pick up the one that catches most faults from f_r. Experimental data shows that significant improvements can be thus achieved in most circuits.

An alternative strategy of choosing a test vector is to give increased priority to hard-to-test faults. For the hard-to-test faults, fewer possible test vectors can be generated to catch them. With the additional constraint of a number of bits being fixed, the possibility of generating a test vector for them is even lower. Thus, in trying to generate a test vector to catch faults from f_u, those hard-to-test faults should be considered at a higher priority. The generation of test vectors is done through the priority-ordered fault list, and the opportunities for catching hard-to-test faults are thus improved.

A way to combine these two schemes consists of assigning a weight for every fault according to its test difficulty. The weight of a test vector is defined as the sum of all the weights of the faults it can catch from f_u. At every iteration, the test vector with the highest weight that fits the constraint can be then selected.

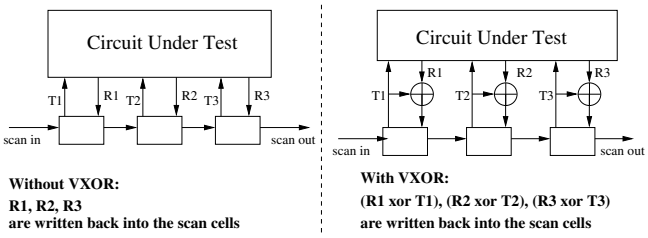


Figure 3: Vertical XOR example

7. EXPERIMENTAL RESULTS

Our implementation on ISCAS89 benchmark circuits uses Perl script language for the flow of algorithm. ATALANTA [7] is used as a test vector generation tool and HOPE [8] is used for fault simulation.

Table 2 shows the comparison of various shifting size and shifting types. Throughout the tables of experimental results, **t** denotes the test time ratio in terms of shifting cycle of our approach to the full shifting scheme, while **m** denotes the test memory requirement ratio. In Table 2, aTV denotes the number of test vectors needed for the traditional full shifting approach that ATALANTA generates. TV denotes the test vector number needed to apply for our approach. The column named **ex** shows the number of extra full shifting test vectors needed to cover the remaining faults. The **shift** column shows the number of bits being shifted every cycle versus the length of the scan chain. For fixed size shifting, we consider three sizes of shifting in terms of the data ratio needed for each cycle. As fundamentally all compression techniques rely in representing a longer sequence in terms of a shorter one, a fair comparison necessitates keeping the bit compression ratio constant. The **info** column provides a comparison at three possible points, namely **3/8**, **5/8** and **7/8**. As specified bits are not limited to scan chain bits only, the consideration of input bits in this computation results in certain of these ratios to be unattainable, particularly in the case of benchmarks with scan chain lengths relatively shorter in comparison to the number of input bits; we denote the resultant inapplicability with a '/' in Table 2.

The results from the experimental data are consistent with our analysis. With fewer number of bits shifted, controllability is limited, resulting in the necessity to employ an increased number of extra test vectors; with a large number of bits shifted, the reduction in test time and memory goes down. Thus in general **5/8** info gives better results among the fixed size shifting schemes while varying the shifting size produces significantly better results due to its flexibility.

Table 3 shows a comparison of the various schemes on improving hidden fault observability. The column NXOR denotes the

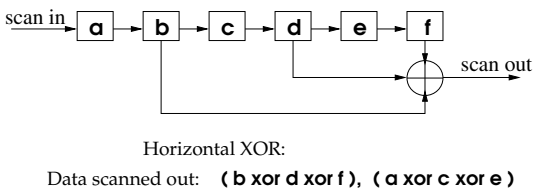


Figure 4: Horizontal XOR example

	NXOR		VXOR		HXOR	
circ	m	t	m	t	m	t
s444	0.88	0.65	0.68	0.47	0.89	0.65
s526	0.74	0.57	0.77	0.62	0.66	0.49
s641	0.89	0.33	0.73	0.23	0.86	0.32
s953	0.59	0.25	0.59	0.25	0.52	0.13
s1196	0.59	0.22	0.49	0.10	0.55	0.17
s1423	0.72	0.53	0.75	0.52	0.68	0.48
s5378	0.76	0.57	0.60	0.49	0.65	0.51
s9234	0.75	0.68	0.67	0.63	0.71	0.65
Ave	0.74	0.48	0.66	0.41	0.69	0.43

Table 3: Hidden Fault Observability

plain implementation without any XOR schemes. VXOR denotes the vertical XOR scheme used and HXOR denotes the horizontal XOR scheme used. The table indicates that improved results in general are produced with XOR implementations.

Table 4 shows a comparison between various methods of selecting test vectors. The column **Random** denotes selecting test vectors by going through a randomly ordered fault list. The column **Hardness** denotes selecting test vectors by going through ordered fault list by hardness to test. The column **Most-faults** denotes selecting test vectors that catches most faults each cycle. The results show that the greedy approach of **Most-faults** usually gives the best result.

While the previous results provide a comparison basis in terms of implementation considerations, it is important as well to provide an assessment of the benefits of the overall scheme in terms of test application time and tester memory reductions. We provide an experimental assessment by utilizing the two previously outlined superior schemes, namely, the variable number of shifted bits and the greedy approach of **Most-faults**, together with the non-XOR approach so as to eliminate any hardware overhead in the comparison basis and apply our technique on some of the largest ISCAS89 benchmarks. The results both in tester memory reduction, **m**, and test application time reduction, **t**, show that appreciable improvements can be attained, ranging between 24% to 80% in **m** and 32% to 93% in **t**. Since testing time is calculated in terms of shifting cycles while memory requirement includes I/O data for each cycle as well, the reduction in test time exceeds that of memory in most experimental results as expected. The implementation on circuit s35932 shows a drastic reduction both in terms of test time and test memory as most faults of s35932 are easy-to-test faults, thus resulting in large reductions by only shifting a small fraction of the scan chain.

	Random		Hardness		Most-faults	
circ	m	t	m	t	m	t
s444	0.81	0.54	0.77	0.50	0.73	0.53
s526	0.86	0.62	0.81	0.58	0.71	0.52
s641	0.88	0.26	0.84	0.24	0.72	0.20
s953	0.70	0.24	0.57	0.17	0.52	0.14
s1196	0.66	0.15	0.53	0.09	0.48	0.09
s1423	0.75	0.50	0.79	0.55	0.68	0.46
s5378	0.73	0.55	0.63	0.48	0.57	0.45
s9234	1.02	0.94	0.98	0.91	0.68	0.63
Ave	0.80	0.48	0.74	0.44	0.64	0.38

Table 4: Selection of Test Vectors

circ	I/O	scan#	m	t
s5378	35/49	179	0.76	0.57
s9234	19/22	228	0.75	0.68
s13207	31/121	669	0.74	0.65
s15850	14/87	597	0.60	0.51
s35932	35/320	1728	0.20	0.07
s38417	28/106	1636	0.60	0.57
s38584	12/278	1452	0.63	0.55
Ave			0.61	0.51

Table 5: Experimental Results for Large Circuits

8. CONCLUSION

We present in this paper a new compression scheme of scan chain based testing, which reduces both test time and memory requirement at no loss of fault coverage. The hardware overhead for this approach is none or negligible. Furthermore, no requirement for the modification of the circuit under test is necessary. Seen from the vantage point of an ATE, the proposed scheme is identical to regular scan based application. The proposed approach exploits the lack of order characteristic of test vectors in a scan chain approach in ATPG and the fact that a large number of unspecified bits exists in test vector generation. Through a well-designed test vector generation algorithm, test vectors can be stitched so as to obtain a large amount of overlapping between a prior response and the subsequent test vector, resulting in high compression ratios. The challenge of observability is solved by an analysis on the behavior of hidden faults. With this approach, no additional hardware, such as a MISR, is needed on the output side; thus the aliasing of faults and the possible loss of information for fault diagnosis is prevented.

For a commercial implementation of the proposed idea, it is important to consider a number of implementation issues. A number of such considerations are discussed in the paper also. Experimental data shows results consistent with the analysis as well as significant improvements on test time and test memory requirements.

9. REFERENCES

[1] A. Jas, B. Pouya and N. A. Touba, "Virtual Scan Chains: A Means for Reducing Scan Length in Cores", in *ITC*, pp. 73–78, 2000.

[2] A. Jas, K. Mohanram and N. A. Touba, "An Embedded Core DFT Scheme to Obtain Highly Compressed Test Sets", in *ATS*, pp. 275–280, 1999.

[3] I. Hamzaoglu and J. Patel, "Reducing Test Application Time for Full Scan Embedded Cores", in *FTCS*, pp. 260–267, 1999.

[4] I. Bayraktaroglu and A. Orailoglu, "Test Volume and Application Time Reduction Through Scan Chain Concealment", in *DAC*, pp. 151–155, 2001.

[5] S. Reda and A. Orailoglu, "Reducing Test Application Time Through Test Data Mutation Encoding", in *DATE*, pp. 387–393, 2002.

[6] C. Su and K. Hwang, "A Serial Scan Test Vector Compression Methodology", in *ITC*, pp. 981–988, 1993.

[7] H. K. Lee and D. S. Ha, "On the Generation of Test Patterns for combinational Circuits", Technical report 12-93, Dep't of Electrical Eng., Virginia Polytechnic Institute and State University, 1993.

[8] H. K. Lee and D. S. Ha, "HOPE: An Efficient Parallel Fault Simulator for Synchronous Sequential Circuits", in *DAC*, pp. 336–340, 1992.