Using Limited Dependence Sequential Expansion for Decompressing Test Vectors

Avijit Dutta and Nur A. Touba

Computer Engineering Research Center
Department of Electrical and Computer Engineering
University of Texas, Austin, TX 78712

Abstract

Existing techniques that incorporate decompressor constraints in the ATPG search/backtrace (e.g., Illinois scan) are based on combinational expansion in which each scan slice must be encoded using only the free-variables arriving form the tester in the current clock cycle. Sequential expansion is more powerful as it allows free-variables across multiple clock cycles to be used, however conventional approaches for sequential expansion that are based on linear finite state machines (LFSRs) and ring generators are not ameniable to including the constraints in the ATPG backtrace because the constraints are too complex. This paper investigates the use of limited dependence sequential expansion to combine the benefits of sequential decompression with the benefits of incorporating the decompressor constraints in the ATPG backtrace. Analytical and experimental results are presented showing the benefits of the proposed approach.

1. Introduction

Test vector compression involves storing a deterministic test set on the tester in a compressed form and using on-chip hardware to decompress it. The test data bandwidth between the tester and chip is generally a bottleneck, so compressing the amount of data that needs to be transferred reduces test time. Test vectors are highly compressible because typically only 1-5% of the bits are specified (care) bits while the rest are don't cares. A number of commercial tools have been introduced in recent years for implementing test vector compression.

One class of test vector compression schemes that is used in a number of commercial tools is based on using a linear decompressor to expand the compressed data coming from the tester to fill the scan chains. Any decompressor that consists of only XORs and flip-flops is a linear decompressor and has the property that its output space (i.e., the space of all possible vectors that it can generate) is a linear subspace. Determining whether a particular test cube (i.e., test vector in which the unassigned inputs are left as don't cares) can be encoded by a linear decompressor can be done by solving a system of linear equations for the specified (care) bits. Combinational linear decompressors [Bayraktaroglu 03], [Mitra 06], use only XOR networks with no flip-flops. If there are b tester channels expanding to fill n scan chains (as illustrated in Fig. 1), then each n-bit "scan slice" is encoded with the b free-variables coming from the tester in the corresponding clock cycle (each bit stored on the tester can be considered a "free-variable" that can be assigned 0 or 1). One drawback is that the worst-case most highly specified scan slices tend to limit the amount of compression that can be achieved because the number of channels from the tester needs to be sufficiently large to encode them. Sequential linear decompressors [Krishna 01, 04], [Konemann 01], [Rajski 04] use linear finite state machines such as linear feedback shift registers (LFSRs) or ring generators [Mrugalski 04] which retain free-variables received from the tester in earlier clock cycles thereby allowing a scan slice to be encoded using free-variables across multiple clock cycles. This allows greater flexibility to handle heavily specified scan slices that may occasionally occur. Consequently, for a fixed number of tester channels, sequential linear decompressors have a higher probability of being able to encode a given test cube.

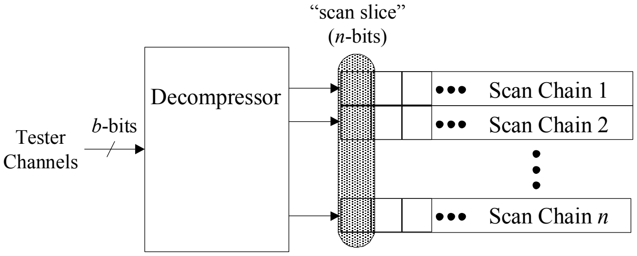

Figure 1. Block diagram of test vector decompression

Another class of test vector compression schemes that is used in commercial tools is based on broadcasting the same value to multiple scan chains. This concept was first proposed in [Lee 98] for scan chains driving independent circuits, and then was adapted for scan chains driving dependent circuits in [Hamzaoglu 99] by adding a serial mode for applying test cubes that cannot be applied in broadcast mode (this has come to be known as Illinois scan). Illinois scan is essentially a special degenerate case of linear decompression in which the decompressor consists of only fanout wires (no XOR gates). The encoding flexibility for the broadcast mode of Illinois scan is less than linear decompressors that use XOR gates. Given a particular test cube, the probability

1-4244-0292-1/06/$20.00 © 2006 IEEE

of encoding it with a linear decompressor that uses XORs is higher than with Illinois scan because it has a more diverse output space with fewer linear dependencies than a fanout network does. However, the fact that faults can be detected by many different test cubes provides an additional degree of freedom. The advantage of Illinois scan is that it is very easy to incorporate the constraints imposed by the decompressor during the ATPG to exploit this degree of freedom in choosing a test cube. This can be done by simply tying dependent inputs together in the circuit description given to the ATPG so that the ATPG algorithm will search only for encodable test cubes. For linear decompressors that use XORs, the conventional approach is to first generate a test cube and then solve the linear equations to see if it is encodable and if it is not then try to find a different test cube with possibly less aggressive dynamic compaction. This is a two step process that does not incorporate the constraints in the ATPG search/backtrace procedure as is done with Illinois scan. So each approach has its advantages. Linear decompressors that use XORs can encode a wider range of test cubes than Illinois scan, but Illinois scan can harness the ATPG to search for encodable test cubes more efficiently.

Ideally, it would be nice to combine the advantages of both approaches. In other words, have greater encoding flexibility than Illinois scan, but retain the ability to include the constraints in the ATPG search/backtrace so the ATPG can efficiently find encodable test cubes. Some work has been done in this direction. One approach is to provide the ability to reconfigure the broadcast mode in Illinois scan to change the constraints. This can be done statically (on a per scan basis) by either reconfiguring the scan chains [Pandey 02] or reconfiguring the fanout network [Samaranayake 03], [Tang 03], [Mitra 06]. Or, it can be done dynamically (on a per shift basis) [Sitchinava 04] where MUXes are placed in front of each scan chain and the control signals for the MUXes are driven by tester channels. In [Wang 04], a combinational network that includes XOR gates is also used and included in the ATPG backtrace.

All of the previous schemes that include the decompressor constraints in the ATPG backtrace are based on combinational decompression where each scan slice must be encoded using only the *b* free-variables arriving from the tester in a single clock cycle. In this paper, we investigate how to use sequential decompression in a way that the constraints are included in the ATPG search/backtrace. The advantage of sequential decompression is that free-variables across multiple clock cycles can be used to encode each scan slice thereby providing greater flexibility and alleviating the problem of the worst-case most heavily specified scan slice limiting the encodability of a test cube. Conventional sequential linear decompressors based on LFSRs or ring generators are not amenable to including the constraints in the ATPG backtrace. The reason is that typically the value of each scan cell depends on the XOR of a large number of free-variables. Including these types of constraints in the ATPG backtrace would greatly increase the search complexity for the ATPG resulting in a large number of backtracks and aborts thereby rendering the ATPG ineffective. To get around this problem, this paper proposes the use of limited dependence sequential expansion which keeps the constraints to a minimum to allow effective ATPG backtrace while still retaining the advantages of sequential expansion in terms of using free-variables across multiple clock cycles.

The contributions of this paper include the following:

- A systematic study of different ways of increasing the flexibility of decompressors for a fixed number of tester channels.
- A new decompressor design that uses limited dependence sequential expansion is proposed, and a synthesis procedure is presented.
- The probability of encoding a test cube with different decompressor designs is analyzed, and the advantages of sequential decompression are quantified.
- Experimental results for benchmarks are shown comparing different compression schemes in terms of the ATPG runtime and the amount of compression achieved.

The paper is organized as follows: Sec. 2 analyzes the encoding flexibility provided by different combinational decompressor designs. Sec. 3 investigates the use of sequential decompressors and shows the advantages compared with combinational decompressors. Sec. 4 discusses some of the issues involved in selecting a decompressor design. Sec. 5 presents a synthesis procedure for synthesizing limited dependence sequential linear decompressors. Sec. 6 show the experimental results. Sec. 7 is a conclusion.

2. Combinational Encoding Flexibility

Illinois scan, where a fanout network from the tester channels is used, provides the simplest constraints for ATPG since it involves simply tying inputs together. However, it has limited encoding flexibility because if two specified bits in a scan slice have opposite value and are fed by the same tester channel, they cannot be encoded. If the number of tester channels is *c*, and the expansion ratio (i.e., the ratio of scan chains to tester channels) is *k*, then the probability of not being able to encode two specified bits in a scan slice is:

$$\frac{k-1}{2(ck-1)}$$

Increasing the encoding flexibility requires adding some gates to the decompressor. Consider adding one 2-input gate to drive each scan chain. To maximize the output space of the decompressor (and hence its encoding flexibility), the logic driving each scan chain should have an output space that is equally balanced between 0's and 1's. This rules out using an AND or OR gate. The only 2-input gate whose output space is equally balanced is a

2-input XOR/XNOR gate. Note that the presence or absence of inversion does not change the probability of encoding an arbitrary test cube, so without loss of generality, only XOR will be considered. If each scan chain is driven by a 2-input XOR of a unique combination of tester channels, then if there are exactly $ck = C_2^c$ scan chains, all scan slices with 2 specified bits can be encoded and the probability of not being able to encode 3 specified bits will be less than:

$$\frac{1}{2(ck-1)}$$

So it is more likely to be able to encode 3 specified bits with 2-input XOR gates than it is to encode 2 specified bits with Illinois scan which is a considerable improvement. The cost is that the constraints during ATPG now require adding 2-input XOR gates into the ATPG backtrace for each pseudo-primary input (pseudo-PI) corresponding to a scan cell. This is illustrated in Fig. 2 with a small example where 2-input XOR gates are used to drive each scan chain, and the constraints for the decompressor are expanded into the circuit given to the ATPG (note that A_i, B_i, and C_i are the free-variables arriving from the tester during clock cycle i). To justify a 0 on a pseudo-PI, there are now 2 different ways to do it (assign 00 to the inputs of the XOR gate driving it or assign 11). This increases the search space for the ATPG thereby slowing it down a little. However, in comparison to Illinois scan, the ATPG has more flexibility when targeting a fault which can lead to better dynamic compaction and less need for resorting to serial mode to detect faults.

To achieve even greater flexibility, 3-input gates could be used to drive each scan chain. In this case there are two options for a balanced output space, a 3-input XOR or a 2-to-1 MUX (other balanced functions are equivalent to those two with inversion). In [Mitra 06], it was shown that using 3-input XORs can guarantee that any 3 specified bits in a scan slice can be encoded. For a MUX, one approach would be to partition the tester channels into control and data where the control channels drive the select signal for the MUXs and the data channels drive the data inputs to the MUXs. This is effectively what is done in [Sitchinava 04]. The other option would be to simply connect any combination of 3 tester channels to each MUX with no distinction between control and data.

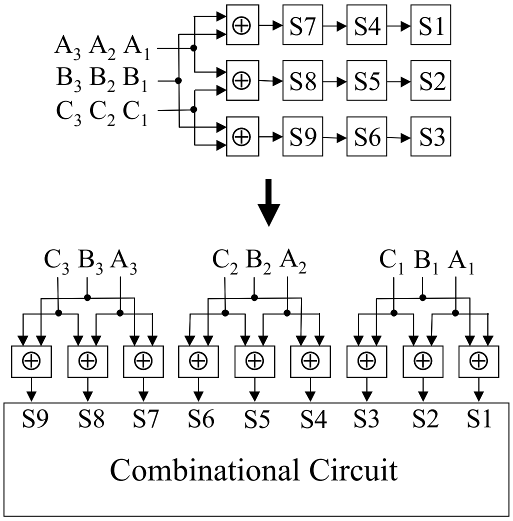

Figure 2. Example of including decompressor constraints at pseudo-PI's for ATPG

The graph in Fig. 3 shows the probability of encoding different numbers of specified bits in a single scan slice for different decompression networks when expanding from 16 tester channels to 160 scan chains (i.e., an expansion ratio of 10). The x-axis is the number of specified bits in the scan slice, and the y-axis is the percentage of all possible combinations of that number of specified bits that can be encoded. As can be seen from the graph, all the decompression networks can always encode 1 specified bit. However, as the number of specified bits is increased, the probability of being able to encode the scan slice drops. Since Illinois has the least encoding flexibility, it has the lowest probability of being able to encode a scan slice. The results for using MUXs are shown for two cases. One is where the control and data lines are separated, i.e., one of the tester channels is dedicated to driving the select line and the other 15 tester channels are used to drive the data lines. The other is where combinations of all 16 tester channels are used to drive either the select or data lines of the MUXs. The results indicate that greater encoding flexibility can be obtained by not having a separate control line. Another interesting result is that using 2-input XOR gates is not as good as using MUXs for low numbers of specified bits, but becomes better than MUXs when the number of specified bits is equal to 10 or more. Using 3-input XORs provides considerably better encoding flexibility although it comes with the tradeoff of adding more complexity to the ATPG than the others.

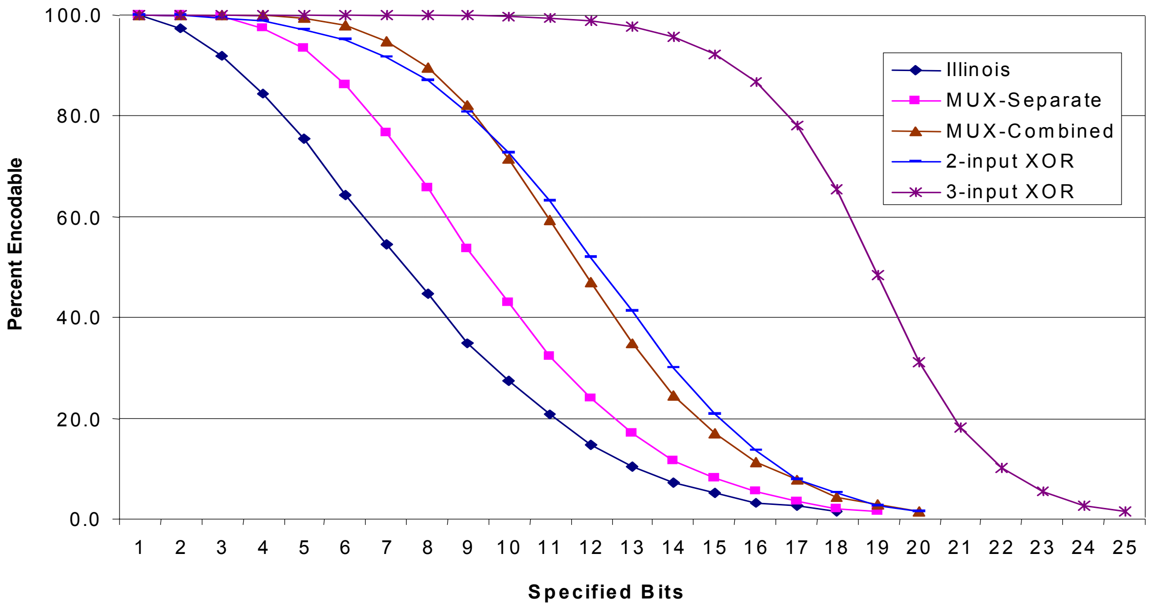

Figure 3. Probability of encoding scan slice for 16 tester channels expanding to fill 160 scan chains

3. Sequential Encoding Flexibility

Previously proposed schemes that include the decompressor constraints in the ATPG backtrace are limited to combinational decompression where each scan slice is encoded using only the free-variables arriving from the tester in a single clock cycle. To achieve greater flexibility to handle the heavily specified scan slices, the use of sequential decompression is investigated here since it allows free-variables across multiple clock cycles to be used in encoding each scan slice. The constraints for conventional approaches for sequential linear decompression that use LFSRs or ring generators are very complex because each scan cell can depend on the XOR of a large number of free-variables. Incorporating such complex constraints in the ATPG backtrace can greatly increase the search complexity of the ATPG. Consider a pseudo-PI whose value depends on the XOR of q free-variables. In order to justify a logic value at the pseudo-PI, q inputs need to be assigned, and the number of possible ways to assign them to get either a 0 or 1 would be 2^q-1. As q increases, this search space grows exponentially. For this reason, conventional approaches that use LFSRs or ring generators do not attempt to include the constraints in the ATPG backtrace. Instead they do ATPG and then check the constraints afterwards. The drawback of this approach is that if an encodable test cube for a fault exists, there is no guarantee that it will be found, and dynamic compaction may need to be done less aggressively in order for the linear equations to remain solvable.

In order to efficiently include the decompressor constraints in the ATPG backtrace, the constraints need to be limited to a dependence on only 2 or 3 free-variables to keep the search space manageable. If the b bits coming from the tester in each clock cycle are defined as a “tester slice”, then one way to perform sequential decompression is to store the last one or two tester slices in a register and use either 2 or 3 input XOR gates to drive each scan chain. The inputs to these XOR gates can come from the domain of the current tester slice and any previous tester slice stored in a register (this is illustrated for two registers in Fig. 4). This provides two benefits. One is that free-variables across 2 or 3 tester slices are used to encode each scan slice which gives more flexibility by providing access to a larger pool of free-variables to handle an occasional heavily specified bit slice, and the second benefit is that a larger number of unique free-variable combinations can be used to drive the scan chains each clock cycle. For example, if there are only 8 tester channels and 2-input XORs are used, then for a combinational decompressor there are only C_2^8 = 28 unique free-variable combinations in each clock cycle which must be broadcast to multiple scan chains if there are more than 28 scan chains. However, if one register is used to store the previous tester slice, then there are C_2^{16} = 120 unique free-variable combinations in each clock cycle, or if two registers are used there are C_2^{24} = 276 unique free-variable combinations. This allows more scan chains to be driven with unique combinations of free-variables which provides greater diversity in the output space and gives more encoding flexibility.

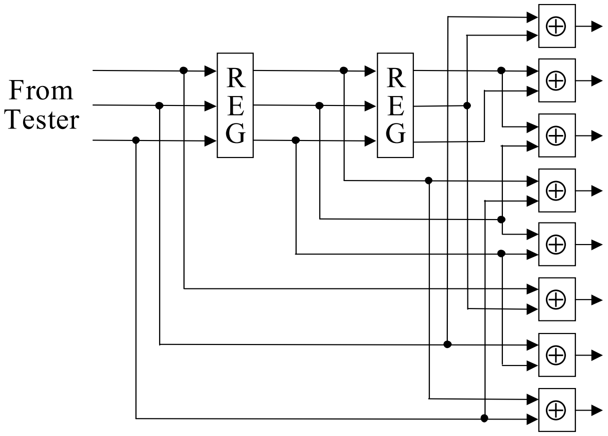

Figure 4. Example of limited dependence sequential decompressor with two registers

The benefit of using sequential decompression versus combinational decompression is shown in Fig. 5. A scan architecture consisting of 80 scan chains each 100 bits long was driven using 8 tester channels. The probability of encoding test cubes with different percentages of specified bits using different decompressor designs is shown. The *x*-axis is the percent of the bits in the test cube that are specified, and the *y*-axis is the probability of encoding the test cube expressed as a percentage. As expected, Illinois scan has the lowest probability of encoding and using an LFSR has the highest probability of encoding (a 64-bit LFSR was used with dynamic reseeding). Using 2-input XORs is shown for the case where only combinational expansion is used and then when 1 and 2 registers are used. As can be seen, the probability of encoding a test cube goes up considerably by adding the registers to perform sequential decompression. Using 2-input XORs with one register is better than using a combinational decoder with 3-input XORs. This is an interesting result because the ATPG search complexity is less with 2-input XORs than with 3-input XORs. Another significant result is the very large improvement that is achieved for 3-input XORs when one or two registers are used to store the previous tester slice. The results begin to approach what an LFSR can achieve, but in this case each pseudo-PI depends on only 3 free-variables thereby making it practical to include the constraints in the ATPG backtrace. To make the comparisons in Fig. 5 fair, the same number of free-variables were used for each decompressor (i.e., a total of 100 tester slices were used for encoding each test cube). No extra shifts were used to pre-load the sequential decompressors. Instead, the sequential decompressors were bypassed in the first clock cycle for the designs with 1 register and the first two clock cycles for the designs with 2 registers. The LFSR was bypassed for the first clock cycle. If one or two extra shifts are used to pre-load the sequential decompressors, the results are slightly better, but there is not much difference.

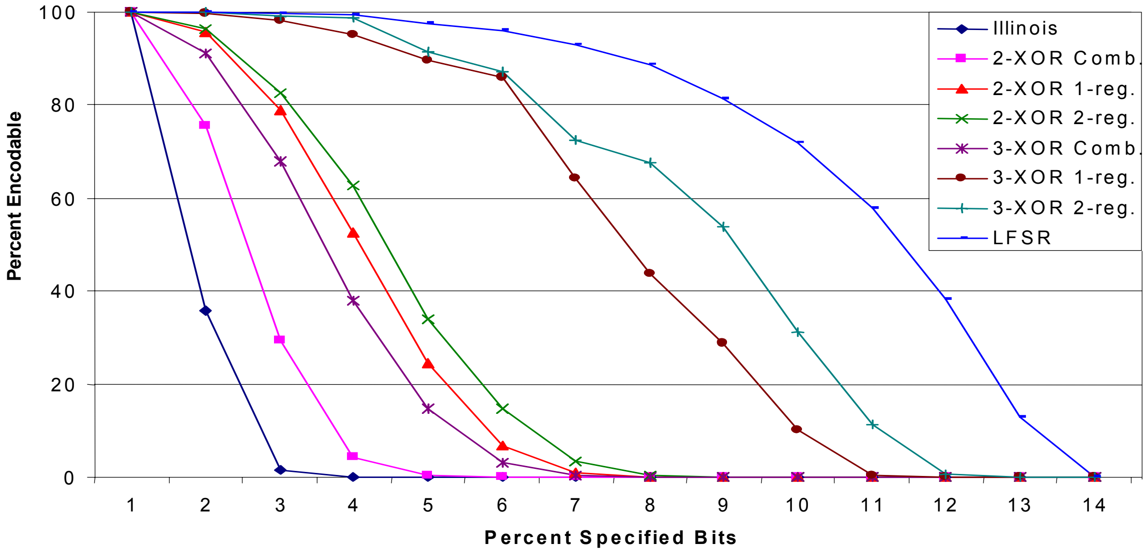

Figure 5. Probability of encoding test cubes for 8 tester channels expanding to fill 80 scan chains

4. Selecting Decompressor Design

As seen in Fig. 5, the addition of registers to store previous tester slices significantly improves the encoding flexibility of the decompressor. This section discusses some of the issues involved in selecting which decompressor design to use. The first issue is whether to use Illinois scan, 2-input, or 3-input XORs. There is a tradeoff in terms of the area and ATPG time versus the amount of compression achieved. The area for all of the decompressor designs is fairly small and probably not a significant factor. The ATPG runtime is a one time cost. If some additional ATPG runtime can be handled, then the reduction in test time that can be achieved with greater compression may be very worthwhile as that reduces test costs for every chip manufactured.

The results in Fig. 5 show that adding registers to enable sequential decompression gives a significant boost which comes with little additional cost in ATPG runtime. The ATPG runtime will mainly depend only on whether 2 or 3 input XORs are used. Adding more registers provides a diminishing marginal improvement. Adding the first register give a big improvement, and then adding the second register give much less improvement. Using more than 2 registers will give some minor improvement, but probably not worth the cost at that point.

Another issue is how to handle the first *r* scan slices if *r* registers are used. The registers are reset between each test cube to decouple them so that each test cube is encoded with its own independent set of free-variables. This means that in the first *r* clock cycles for each test cube, some or all of the *r* registers will not yet be filled with free-variables and thus will not be ready to drive the scan chains. The simple solution is to just use *r* extra shifts when decompressing each test cube. The extra shifts fill the *r* registers before the first scan slice is decompressed. If *r* is very small relative to scan length, then this will not have much impact on the test time. The other alternative is to use MUXes to bypass the sequential decompressor when decompressing the first *r* scan slices. The scan chains can be driven during those clock cycles with a combinational decompression network that depends only on the current tester slice. This second approach does not require any extra shifts, and this is what was used for generating all the results in this paper to provide a fair comparison with combinational decompressors (because in this case the same number of free-variables are used to encode each test cube). However, from an implementation standpoint, the first approach of using extra shifts is probably more attractive since it simplifies the hardware.

5. Synthesis Procedure for Decompressor

The procedure for synthesizing a limited dependence sequential linear decompressor for expanding *b* tester channels to fill *n* scan chains using *r* tester slice registers and *q*-input gates driving each scan chain is as follows:

1. Generate all $C_q^{b(r+1)}$ combinations of the current tester slice bits and the tester slice registers' bits.
2. For each scan chain, select an unused combination whose individual components have collectively been used as inputs to the fewest gates. Mark that combination as used. Create a gate to drive the scan chain using the selected combination as the inputs.
3. If there are more scan chains than combinations, then fan out the output of each gate corresponding to a combination to multiple scan chains. Keep the number of fanouts for each gate as balanced as possible.

The domain of possible inputs to the gates is the *b*-bits in the current tester slice combined with the *b*-bits in each of the registers storing the previous *r* tester slices. Combinations of these are selected to drive each scan chain in a way that balances the use of each input evenly. This spreads the use of the free-variables evenly.

The design could be optimized if it is customized for a particular circuit-under-test. If structural information is known about the circuit-under-test and the scan chain ordering is known, then it is possible to choose the combinations of inputs that drive each scan chain in a way that maximizes the probability of encoding a test cube. The synthesis procedure here assumes no information about the circuit-under-test, and thus generates a decompressor that is applicable for any circuit-under-test.

6. Experimental Results

In this section, experimental results are presented for using different decompessors. Table 1 shows details for the circuits that were used. Experiments were performed on one ISCAS benchmark circuit (s38584) and two industrial circuits (Design A and B). The number of scan cells, the total number of faults, and the number of ATPG vectors required for 100% coverage of detectable faults are reported in Table 1.

Tables 2 through 4 report the results obtained for each of the designs listed in Table 1. In each case, different decompressors were used to expand 8 tester channels to fill the number of scan chains shown in the column header of the fourth, fifth, and sixth columns. The decompressors are Illinois scan, combinational 2-input XOR gates driving each scan chain, combinational 3-input XOR gates driving each scan chain, and the proposed limited dependence sequential linear decompressors with one or two tester slice registers using either 2-input XOR gates driving each scan chain or 3-input XOR gates driving each scan chain. In generating the results, the constraints for each decompressor were added to the circuit description given the ATPG tool. A commercial ATPG tool was used to generate all the results reported here (although any ATPG tool can be used).

In each table, results are first shown in the upper part of the table for using a single configuration. The results include the compression ratio that is achieved (i.e., normal uncompressed tester storage for a test set generated with no constraints divided by compressed tester storage), the number of parallel and serial vectors that are used, and the coverage that is obtained if only parallel vectors are used. In the lower part of each table, results are shown for using 4 configurations with static reconfiguration where the configuration is changed only on a per scan basis. These results were obtained by first detecting as many faults as possible with the first configurations, and then using each subsequent configuration to detect any faults that still remain undetected. The results that are reported for using 4 configurations include the amount of compression, and the number of parallel and serial vectors that are required. At the bottom of the table, the ATPG runtime is shown. This is the time that it takes to run ATPG for the first configuration (subsequent configurations are much faster since most of the faults are already detected).

In the results, it can be seen that Illinois scan has the shortest ATPG runtime as expected, but it also provides the lowest amount of compression. The proposed limited dependence sequential decompressors require longer ATPG runtimes, but achieve much better compression. As can be seen, the addition of the tester slice registers to perform sequential decompression significantly improves the results.

Table 1. Design Details

Design	Scan cells	Faults	Fullscan ATPG vectors
s38584	1464	105298	135
Design A	7654	239902	796
Design B	856	53689	154

Table 2. Results for s38584

		Num. Scan Chains		
	Decompressor	192	224	256
Compression (1 config.)	Illinois	3.4	3.2	2.0
	2-xor comb	5.2	4.4	3.3
	3-xor comb	5.6	5.5	5.1
	2-xor, 2 reg	6.0	6.3	5.3
	3-xor, 2 reg	6.1	6.7	6.2
Parallel Vectors (1 config.)	Illinois	286	280	293
	2-xor comb	332	330	385
	3-xor comb	437	454	499
	2-xor, 2 reg	339	425	442
	3-xor, 2 reg	441	447	452
Serial Vectors (1 config.)	Illinois	27	31	56
	2-xor comb	11	18	28
	3-xor comb	5	7	10
	2-xor, 2 reg	5	5	11
	3-xor, 2 reg	3	3	7
Coverage with parallel vectors only (1 config.)	Illinois	93.2	94.7	93.2
	2-xor comb	99.6	98.2	93.1
	3-xor comb	99.7	98.1	93.2
	2-xor, 2 reg	99.8	98.0	93.1
	3-xor, 2 reg	99.8	99.6	95.6
Compression (4 config.)	Illinois	3.9	3.7	2.4
	2-xor comb	6.5	4.8	4.0
	3-xor comb	6.5	6.3	5.3
	2-xor, 2 reg	6.8	6.3	5.9
	3-xor, 2 reg	6.7	7.5	6.3
Parallel Vectors (4 config.)	Illinois	458	470	466
	2-xor comb	403	412	465
	3-xor comb	472	512	588
	2-xor, 2 reg	451	510	490
	3-xor, 2 reg	465	472	590
Serial Vectors (4 config.)	Illinois	13	18	42
	2-xor comb	3	12	18
	3-xor comb	0	2	6
	2-xor, 2 reg	0	2	7
	3-xor, 2 reg	0	0	2
ATPG Runtime	Illinois	3.14	3.26	3.44
	2-xor comb	5.40	3.20	5.24
	3-xor comb	6.65	7.20	7.24
	2-xor, 2 reg	5.20	4.76	6.25
	3-xor, 2 reg	7.10	7.14	7.24

Table 3. Results for Design A

		Num. Scan Chains		
	Decompressor	64	128	192
Compression With 1 config.	Illinois	2.9	4.4	4.6
	2-xor comb	3.7	4.7	4.7
	3-xor comb	4.1	5.5	6.0
	2-xor, 1 reg	4.8	5.5	6.2
	2-xor, 2 reg	4.8	5.9	7.2
	3-xor, 1-reg	4.9	5.7	6.4
	3-xor, 2-reg	4.9	5.7	7.6
Parallel Vectors (1 config.)	Illinois	791	810	853
	2-xor comb	776	785	780
	3-xor comb	795	814	830
	2-xor, 1 reg	760	821	787
	2-xor, 2 reg	795	797	883
	3-xor, 1 reg	798	824	807
	3-xor, 2 reg	775	793	804
Serial Vectors (1 config.)	Illinois	168	132	137
	2-xor comb	118	120	138
	3-xor comb	95	93	98
	2-xor, 1 reg	75	92	96
	2-xor, 2 reg	66	74	72
	3-xor, 1 reg	64	88	90
	3-xor, 2 reg	66	89	70
Coverage with parallel vectors only (1 config.)	Illinois	92.1	92.5	90.2
	2-xor comb	93.4	93.5	93.2
	3-xor comb	93.2	93.1	92.9
	2-xor, 1 reg	92.2	92.8	93.4
	2-xor, 2 reg	94.1	93.9	94.2
	3-xor, 1 reg	94.3	94.0	94.6
	3-xor, 2 reg	95.3	97.8	97.6
Compression with 4 configs.	Illinois	3.6	4.5	4.8
	2-xor comb	4.0	4.9	5.4
	3-xor comb	4.8	6.1	6.0
	2-xor, 1 reg	4.8	6.5	7.8
	2-xor, 2 reg	4.8	6.2	7.9
	3-xor, 1 reg	4.7	6.6	7.8
	3-xor, 2 reg	5.5	6.6	8.0
Parallel Vectors (4 config.)	Illinois	807	902	1102
	2-xor comb	798	822	878
	3-xor comb	885	920	995
	2-xor, 1 reg	1133	1512	1588
	2-xor, 2 reg	1139	1556	1498
	3-xor, 1 reg	1188	1590	1588
	3-xor, 2 reg	1002	1603	1616
Serial Vectors (4 config.)	Illinois	121	122	119
	2-xor comb	98	110	112
	3-xor comb	55	73	90
	2-xor, 1 reg	24	27	36
	2-xor, 2 reg	23	30	38
	3-xor, 1 reg	22	21	36
	3-xor, 2 reg	18	20	32
ATPG Runtime (sec)	Illinois	152	148	149
	2-xor comb	155	149	155
	3-xor comb	162	166	164
	2-xor, 1 reg	160	158	150
	2-xor, 2 reg	158	166	171
	3-xor, 1 reg	162	162	159
	3-xor, 2 reg	169	166	169

Table 4. Results for Design B

		Num. Scan Chains		
	Decompressor	64	128	192
Compression (1 config.)	Illinois	1.2	1.3	1.2
	2-xor comb	1.6	1.6	1.5
	3-xor comb	1.5	1.8	1.5
	2-xor, 1 reg	1.8	1.8	1.7
	2-xor, 2 reg	1.8	1.9	1.8
	3-xor, 1 reg	1.9	2.0	2.0
	3-xor, 2 reg	2.4	3.2	2.5
Parallel Vectors (1 config.)	Illinois	106	88	108
	2-xor comb	135	164	122
	3-xor comb	190	189	202
	2-xor, 1 reg	199	238	192
	2-xor, 2 reg	248	294	283
	3-xor, 1 reg	243	287	347
	3-xor, 2 reg	252	330	313
Serial Vectors (1 config.)	Illinois	99	105	118
	2-xor comb	74	80	93
	3-xor comb	74	72	93
	2-xor, 1 reg	50	66	80
	2-xor, 2 reg	54	64	72
	3-xor, 1 reg	50	55	60
	3-xor, 2 reg	36	45	48
Coverage with parallel vectors only (1 config.)	Illinois	91.1	89.6	89.7
	2-xor comb	94.1	92.3	91.1
	3-xor comb	95.3	94.6	93.5
	2-xor, 1 reg	96.6	95.8	93.8
	2-xor, 2 reg	97.2	96.6	96.7
	3-xor, 1 reg	97.8	97.2	96.8
	3-xor, 2 reg	98.2	98.2	97.6
Compression (4 configs.)	Illinois	1.4	1.5	1.3
	2-xor comb	1.7	2.1	1.9
	3-xor comb	1.9	2.3	1.8
	2-xor, 1 reg	2.1	2.1	2.0
	2-xor, 2 reg	2.1	2.6	2.5
	3-xor, 1 reg	2.2	2.6	2.4
	3-xor, 2 reg	2.5	3.3	3.1
Parallel Vectors (4 config.)	Illinois	342	346	366
	2-xor comb	270	351	362
	3-xor comb	270	380	356
	2-xor, 1 reg	309	386	344
	2-xor, 2 reg	301	387	361
	3-xor, 1 reg	299	383	456
	3-xor, 2 reg	269	371	427
Serial Vectors (4 config.)	Illinois	66	75	98
	2-xor comb	54	50	63
	3-xor comb	44	42	63
	2-xor, 1 reg	30	46	60
	2-xor, 2 reg	34	34	45
	3-xor, 1 reg	30	35	42
	3-xor, 2 reg	28	27	30
ATPG Runtime (sec)	Illinois	0.97	0.96	0.98
	2-xor comb	1.08	1.26	1.92
	3-xor comb	2.12	2.22	2.80
	2-xor, 1 reg	1.10	1.87	2.57
	2-xor, 2 reg	1.70	2.14	2.36
	3-xor, 1 reg	2.62	2.81	3.26
	3-xor, 2 reg	2.31	2.76	2.98

7. Conclusions

The results in this paper show that by using limited depth sequential decompression, a significant improvement in compression can be achieved. A number of commercial test compression schemes are based on incorporating the decompressor constraints in the ATPG search/backtrace. The proposed method provides a simple and practical way to boost the effectiveness of such schemes by incorporating tester slice registers to allow the use of free-variables across multiple clock cycles.

One area for future research would be to investigate how structural information about the logic cones in the circuit-under-test could be used to improve the design of limited depth sequential decompressors.

Acknowledgements

The authors would like to thank SynTest Technologies (special thanks to David Wang) for providing Design A and Design B on which the experiments were performed. This material is based on work supported in part by the National Science Foundation under Grant No. CCR-0306238.

References

[Bayraktaroglu 03] I. Bayraktaroglu and A. Orailoglu, "Concurrent Application of Compaction and Compression for Test Time and Data Volume Reduction in Scan Designs," *IEEE Trans. on Computers*, Vol. 52, No. 11, pp. 1480-1489, Nov. 2003.

[Hamzaoglu 99] I.Hamzaoglu and J.H.Patel, "Reducing Test Application Time for Full Scan Embedded Cores," *Proc. Int. Symp. on Fault-Tolerant Computing*, pp. 260-267, 1999.

[Krishna 01] C.V. Krishna, A. Jas, and N.A. Touba, "Test Vector Encoding Using Partial LFSR Reseeding," *Proc. Int. Test Conf.*, pp. 885-893, 2001.

[Krishna 04] C.V. Krishna, N.A. Touba, "3-Stage Variable Length Continuous-Flow Scan Vector Decompression Scheme," *Proc. of VLSI Test Conf.*, pp. 79-86, 2004.

[Könemann 01] B. Koenemann, C. Barnhart, B. Keller, T. Snethen, O. Farnsworth, and D. Wheater, "A SmartBIST Variant with Guaranteed Encoding," *Proc. Asian Test Symp.*, pp. 325-330, 2001.

[Lee 98] K.-J. Lee, J.J. Chen, and C.H. Huang, "Using a Single Input to Support Multiple Scan Chains," *Proc. Int. Conf. on Computer-Aided Design*, pp. 74-78, 1998.

[Mitra 06] S. Mitra and K.S. Kim, "XPAND: An Efficient Test Stimulus Compression Technique," *IEEE Trans. on Computers*, Vol. 55, No. 2, pp. 163-173, Feb. 2006.

[Mrugalski 04] G. Mruglaski, J. Rajski, and J. Tyszer, "Ring Generators – New Devices for Embedded Test Applications," *IEEE Trans. on Computer-Aided Design*, Vol. 23, No. 9, pp. 1306-1320, Sept. 2004.

[Pandey 02b] A.R. Pandey and J.H. Patel, "Reconfiguration Technique for Reducing Test Time and Test Volume in Illinois Scan Architecture Based Designs," *Proc. VLSI Test Symposium,* pp. 9-15, 2002.

[Rajski 04] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherje, "Embedded Deterministic Test," *IEEE Trans. on Computer-Aided Design*, Vol. 23, No. 5, pp. 776-792, May 2004.

[Samaranayake 03] S. Samaranayake, E. Gizdarski, N. Sitchinava, F. Neuveux, R. Kapur, and T. W. Williams, "A Reconfigurable Shared Scan-In Architecture," *Proc. VLSI Test Symposium*, pp. 9-14, 2003.

[Sitchinava 04] N. Sitchinava, S. Samaranayake, R. Kapur, E. Gizdarski, F. Neuveux, and T.W. Williams, "Changing the Scan Enable During Shift," *Proc. VLSI Test Symposium*, pp. 73-78, 2004.

[Tang 03] H. Tang, S.M. Reddy, and I. Pomeranz, "On Reducing Test Data Volume and Test Application Time for Multiple Scan Designs," *Proc. Int. Test Conf.,* pp. 1079-1088, 2003.

[Wang 04] L.-T. Wang, X. Wen, H. Furukawa, F.-S. Hsu, S.-H. Lin, S.-W. Tsai, K. S. Abdel-Hafez, and S. Wu, "VirtualScan: A New Compressed Scan Technology for Test Cost Reduction," *Proc. Int. Test Conf.*, pp. 916-925, 2004.