3-Stage Variable Length Continuous-Flow Scan Vector Decompression Scheme

C.V. Krishna and Nur A. Touba

Computer Engineering Research Center
University of Texas, Austin, TX 78712-1084

Abstract

This paper presents a 3-stage continuous-flow linear decompression scheme for scan vectors that uses a variable number of bits to encode each vector. By using 3-stages of decompression, it can efficiently compress any test cube (i.e., deterministic test vector where the unassigned bit positions are left as don't cares) regardless of the number of specified (care) bits. As a result of this feature, there is no need for any constraints on the automatic test generation process (ATPG) process. Any ATPG can be used with any amount of static or dynamic compaction. Experimental results are shown which demonstrate that the proposed scheme achieves extremely high encoding efficiency.

1. Introduction

Test data compression techniques provide a means to reduce the test data storage requirements on the tester and the test data bandwidth requirements between the tester and chip, thereby allowing less expensive testers to be used as well as reducing test time. A number of interesting test vector compression schemes have been developed using a variety of codes including run-length codes [Jas 98], selective Huffman codes [Jas 99], Golomb codes [Chandra 00], frequency directed codes [Chandra 01], VIHC codes [Gonciari 02], LZ77 [Wolff 02], Mutation codes [Reda 02], packet-based codes [Khoche 02], [Volkerink 02], and non-linear combinational codes [Reddy 02], [Li 03]. Another class of schemes involves reducing test vector volume by encoding test vectors in a longer built-in self-test (BIST) test sequence. These techniques include using hybrid patterns [Das 00], folding counters [Hellebrand 00], [Liang 01], and RESPIN [Dorsch 01].

A third major class of test vector compression schemes are based on linear expansion. The data is decompressed by performing only linear operations. This includes techniques based on linear feedback shift register (LFSR) reseeding and combinational linear expansion circuits consisting of XOR gates. Commercial tools for compressing test vectors including TestKompress from Mentor Graphics [Rajski 02] and SmartBIST from IBM/Cadence [Könemann 01] are based on linear expansion circuits. Linear expansion circuits exploit the unspecified (don't care) bit positions in test cubes (i.e., deterministic test vectors where the unassigned bit positions are left as don't cares) to achieve large amounts of compression. This paper describes a new scheme for compressing test vectors using a linear expansion circuit.

The proposed scheme has a number of nice features. It uses a continuous-flow decompressor so it can be very efficiently connected directly to the tester. The architecture of the decompressor combines three different stages of linear expansion so it is very efficient for any distribution of specified bits in a test cube. It can vary the number of bits that are used to encode each pattern (i.e., it does not use a fixed number of bits to encode every pattern). Thus, it can efficiently compress any test cube regardless of the number of specified (care) bits. As a result of this feature, there is no need for any constraints on the automatic test generation process (ATPG) process. Any ATPG can be used with any amount of static or dynamic compaction. It has important applications for situations where it is not possible or desirable to use a special ATPG that is tailored for the decompressor. For example, if an existing set of test cubes is available, or for circuits where unconstrained static and dynamic compaction can significantly reduce the total number of vectors. The proposed scheme requires low hardware overhead as it configures the decompressor out of the scan cells themselves.

2. Previous Work

Test vector compression schemes that are based on linear expansion use only linear operations to decompress the data. Consequently, each decompressed bit can be represented as a linear (modulo-2) sum of the compressed bits. This idea was originally studied by Könemann for use in LFSR-coding [Könemann 91]. If each bit stored on the tester is represented by a "free-variable" that can be assigned any value (0 or 1), then the basic idea with linear expansion techniques is to send the data from the tester through a linear expansion circuit (which can be combinational or sequential) as it is shifted into the scan chains. The contents of each scan cell can then be represented by a linear equation in terms of the free-variables on the tester. Test cubes (i.e., deterministic test vectors where the unassigned bit positions are left as don't cares) typically have only 1-10% of the bits specified, while the rest are don't cares. Thus, a system of linear equations can be formed consisting of one equation for each specified bit position in a test cube. The solution to this system of linear equations gives an assignment of values to the free-variables on the tester that will produce the test cube in the scan chain (details about this process can be found in [Könemann 91], [Hellebrand 95a], [Krishna 01]). If the number of free-variables used for a test cube is sufficiently larger than the number of specified bits in the test cube, then the probability of not being able to solve the system of linear equations becomes negligible. For LFSR reseeding, it has been shown that if the number of free-variables is 20 more than the number of specified bits, then the probability of not being able to solve the system of linear equations is less than 10^{-6} [Könemann 91]. Thus, linear expansion schemes are very good at exploiting the don't cares in the test cubes to provide high compression. They generally provide a very high *encoding efficiency* which is defined as the ratio of the number of specified bits to the number of bits stored on the tester.

A number of test data compression schemes based on linear expansion have been proposed. They can be categorized based on two attributes. One is whether they receive data from the tester in a continuous-flow (i.e., "streaming" data) or whether they are periodically-loaded (i.e., require some delay between loads), and the other is whether they use a fixed number of free-variables per test cube or whether they can vary the number of free-variables used per test cube. Continuous-flow schemes can be directly connected to the tester and operate very efficiently since they simply receive the data as fast as the tester can transfer it. From a tools integration standpoint, this is very nice since it mimics the standard behavior of normal scan chains. Periodically-loaded schemes typically can achieve slightly higher encoding efficiency, but they require some test scheduling to fully utilize the tester bandwidth during the time period between loads. They are better suited for an SOC environment where multiple cores are being tested concurrently and the data from the tester can be scheduled to be delivered to each core as needed so that the tester bandwidth is never wasted.

Schemes that use a fixed number of free-variables per test cube have less control complexity. Each test cube is treated the same, so no extra control information is needed per test cube. The drawback is that in order to encode all test cubes, the number of free-variables used per test cube must be sufficient to encode the most specified test cube (i.e., the worst-case). If s_{max} is the number of specified bits in the most specified test cube, then the number of free-variables used to encode each and every test cube is determined by s_{max}. If s_{avg} is the average number of specified bits per test cube, then if there is a significant difference between s_{max} and s_{avg}, the encoding efficiency will be substantially degraded. To avoid this, constraints on the ATPG process (especially on static and dynamic compaction) are needed to keep s_{max} from becoming too large. This may result in more test vectors than would be generated if the ATPG was not constrained. Schemes that can vary the number of free-variables used for each test cube avoid the need for constraints on the ATPG process. These schemes have high encoding efficiency regardless of s_{max} and s_{avg}. They do not require any special ATPG tool support, and may require fewer test vectors since more compaction can be done during ATPG. The drawback of these schemes is the need for additional control bits for each test cube which may offset some of the gains, as well as some extra hardware complexity.

Periodically-loaded schemes with a fixed number of free-variables per test cube:

The earliest linear expansion schemes were periodically-loaded schemes that used a fixed number of free-variables per test cube. These schemes used full reseeding of LFSRs [Könemann 91] or multiple-polynomial LFSRs [Hellebrand 95a]. More recent work uses partial LFSR reseeding [Krishna 01].

Periodically-loaded schemes with a variable number of free-variables per test cube:

The schemes described in [Zacharia 95, 96], [Rajski 98a] varies the number of free-variables per test cube by reseeding variable-length LFSRs. The scheme in [Hellebrand 95b] tries to encode multiple test cubes per seed. The scheme in [Krishna 02] compresses LFSR seeds using a statistical code, so the number of bits stored on the tester for each test cube varies depending on the amount of seed compression achieved.

Continuous-flow schemes with a fixed number of free-variables per test cube:

The scheme described in [Jas 00] is a continuous-flow scheme that internally uses LFSR reseeding. Its encoding efficiency is limited by the fact that it uses small LFSRs and directly feeds one scan sub-chain. The schemes in [Bayraktaroglu 01, 03] and [Mitra 03] use a combinational linear expansion circuit to decompress one "bit-slice" (see Fig. 1) of a set of multiple scan chains at a time. Their encoding efficiency is limited by the fact that it uses a fixed number of free-variables per bit-slice. The scheme described in [Rajski 02] injects free-variables from the tester into a ring generator [Mrugalski 03] (which is similar in nature to an LFSR) which then passes through a phase shifter and into the scan chains. This method is tightly coupled with the ATPG software to ensure that each test cube that is generated can be encoded. This scheme is used in the TestKompress tool from Mentor Graphics and has been shown to achieve excellent compression for industrial circuits. One other scheme that technically can be classified as a linear expansion scheme is serial/parallel scan chains (Illinois scan architecture) [Hamzaoglu 99], [Hsu 01]. Here the linear expansion circuit does not have any XOR gates, but simply routes the same scan-in line to multiple scan chains. If there are conflicting specified values, then the test vector is applied in serial mode with no compression.

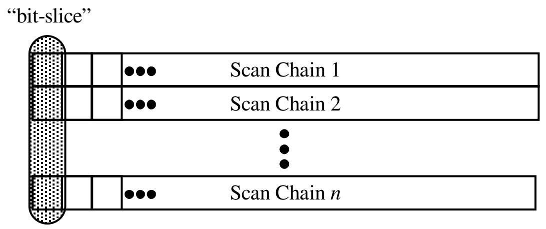

Figure 1. Definition of a "Bit-Slice" of Multiple Scan Chains

Continuous-flow schemes with a variable number of free-variables per test cube:

The first continuous flow scheme that used a variable number of free-variables per test cube was part of the SmartBIST family of decoders described in [Könemann 01]. This method involves injecting free-variables from the tester into a set of LFSRs that collectively have as many stages as the number of scan chains being driven. One channel from the tester is used for controlling a clock gating mechanism that controls when the scan chains are loaded from the set of LFSRs receiving the free-variables. When a sufficient number of free-variables have been injected into the LFSRs to solve the linear equations for the specified bits in the next bit-slice of the scan chains, then the clock gating mechanism allows the scan chains to be loaded. Most of the time, the linear equations will be solved immediately, so the clock gating mechanism is only used occasionally when a bit-slice has too many specified bits to solve right away. This scheme is capable of applying any test cube regardless of the number of specified bits it has. The clock gating control signal is controlling how many free-variables are used per bit-slice and hence per test cube. One nice feature of this scheme is that it has a very modular design. A drawback is that one tester channel needs to be completely devoted to providing the clock gating signal. A scheme based on static LFSR reseeding was described in [Volkerink 03] which uses the same control mechanism as [Könemann 01], but uses a large LFSR (e.g., 500 bits) along with a shadow register to allow it to decompress more than one bit-slice at a time. A scheme based on seed overlapping was described in [Rao 03] which is based on the method in [Bayraktaroglu 01] but allows seeds of different sizes to be used. It also decompresses one scan bit-slice at a time and requires a special constrained ATPG.

The proposed scheme is a continuous-flow scheme with a variable number of free-variables per test cube which combines three different stages of linear expansion. Unlike the previous methods, it does not require an extra tester channel for the control information, rather the control information is reduced to only a few bits per test cube thereby freeing up the tester channel so it can be used to inject more free-variables instead. Instead of solving for one scan bit-slice at a time as in [Könemann 01] and [Rao 03], the entire test cube is solved together which allows greater encoding efficiency. Whereas the method in [Volkerink 03] requires a large LFSR along with a shadow register and the number of bit-slices that it can solve for at a time is limited by the size of the LFSR, in the proposed approach, the decompressor is constructed from the scan elements themselves thereby reducing hardware requirements and allowing it to solve for the entire test cube. Unlike the method in [Rao 03], no special constrained ATPG is required for the proposed approach. The proposed 3-stage decompressor provides extremely high encoding efficiency for any distribution of specified bits.

3. Architecture for Proposed Scheme

The architecture for the proposed scheme is shown in Fig. 2. The portion shown in gray is configured from the scan cells themselves. In each clock cycle, a *b*-bit block of free-variables arrives from the tester. The *b*-bit block is expanded through a linear combinational expansion circuit similar to the method in [Bayraktaroglu 01, 03] to load *n* scan chains. Each of the *n* outputs of this linear combinational expansion circuit is generated through a three-input XOR gate which forms a linear combination of the free-variables. In [Bayraktaroglu 01, 03], constraints are placed on the ATPG to ensure that every bit-slice can always be generated (i.e., the linear equations can be solved) immediately through the combinational circuit. This is possible if the number of specified bits in each (*n*-bit) bit-slice of the scan chains is approximately less than *b* (note that it is possible to solve the equations for more specified bits than *b*, but the probability of success diminishes exponentially). While this makes things simple and is an attractive aspect of this approach, it becomes very restrictive as the ratio of *n* to *b* becomes large hence limiting the amount of static and dynamic compaction that can be done resulting in the ATPG generating more test cubes or possibly not being able to detect some faults under these restrictions. To avoid these limitations, and to get a much better encoding efficiency, the proposed method configures the scan cells into a set of LFSRs.

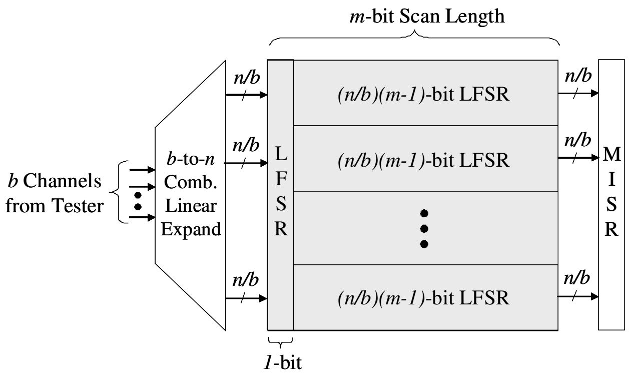

Figure 2. Architecture of Proposed Scheme for *n* Scan Chains

The idea of configuring an LFSR from the scan cells for LFSR reseeding was first described in [Zacharia 95, 96], [Rajski 98a, 99]. In the proposed scheme, a set of LFSRs is configured as shown in Fig. 2. There is a "small vertical LFSR" which in turn feeds a set of *b* "large horizontal LFSRs." The small vertical LFSR is one bit wide and *n* bits long. It is configured by having the output of each scan cell in the leftmost bit-slice of the scan chains be XORed with the scan-in of the next scan chain (the last scan chain wraps around to the first) and then including some feedback taps to implement a primitive polynomial for a modular (type II) LFSR. Each of the large horizontal LFSRs have *(n/b)(m-1)* stages where *m* is the scan length of each scan chain (if the scan chains are not balanced in length, then the size of each LFSR would vary accordingly). The large horizontal LFSRs are configured by partitioning the scan chains into *b* groups of *n/b* scan chains. The final scan-out of a scan chain in a group is XORed with the scan-in of the first scan cell of the next scan chain in the group (the last scan chain in the group wraps around to the first) and then including some feedbacks to implement a primitive polynomial for a modular (type II) LFSR. Each of the large horizontal LFSRs essentially have *n/b* external inputs that are fed by the corresponding *n/b* stages in the small vertical LFSR. Each of these external inputs in the large horizontal LFSR enters at a distance of *m-1* stages away from each other in the LFSR structure. An optional phase-shifter between the small vertical LFSR and the large horizontal LFSRs can also be implemented by adding some additional XOR terms [Rajski 98b].

Not counting the feedback taps for implementing the primitive polynomials or optional phase shifter, the number of XOR gates required for configuring this architecture consists of the following. The small vertical LFSR has *n* 2-input XORs, each of the large horizontal LFSRs has *n/b* 2-input XORs, and the linear combinational expansion circuit has *n* 3-input XORs. The number of XOR gates required for implementing the primitive polynomials will be *(b+1)* times *(number of terms per polynomial - 1)*. Each of the XORs in the small vertical LFSR has one of its inputs (the one creating feedback) ANDed with a control signal (*enable_small_LFSR*). The same is done for the large horizontal LFSRs with a control signal (*enable_large_ LFSRs*). The *enable_small_LFSR* control signal is set to 0 to disable the feedback of the small vertical LFSR during the first clock cycle in which the first bit-slice is loaded. This is done so that the output response from the previous scan vector does not interfere with the state of the small vertical LFSR. Essentially, the small vertical LFSR gets initialized in the first clock cycle. In subsequent clock cycles, the *enable_small_LFSR* control signal is set to 1 so that it operates as an LFSR. For the large horizontal LFSR, the *enable_large_LFSRs* control signal is set to 0 to disable its feedback during the first *m* clock cycles. Again, this is done to allow the output response to be shifted out and the large horizontal LFSRs to get initialized with linear combinations of the free variables coming in from the tester. Starting in the *m+1* clock cycle, the *enable_large_LFSR* control signal is set to 1 so that it operates as an LFSR. Note that the *enable_large_LFSR* and *enable_small_LFSR* signals are controlled by the on-chip test controller.

4. Operation of Proposed Decompressor

The architecture for the proposed scheme (described in Sec. 3) combines three different stages of linear expansion. The **first stage** is the linear combinational expansion circuit which expands the *b* free-variables that arrive from the tester each clock cycle to fill each bit-slice of the multiple scan chains. This is similar to [Bayraktaroglu 01, 03]. For test cubes where every bit-slice has sufficiently fewer than *b* specified bits, the proposed scheme can decompress the test cube in *m* clock cycles (the minimum number of cycles needed to fill the scan chains). Thus it performs very well for lightly specified test cubes.

For test cubes where one or more bit-slices have too many specified bits to solve with only *b* free-variables, the scheme in [Bayraktarouglu 01, 03] cannot be used for decompression. Thus, the scheme in [Bayraktaroglu 01, 03] either has to reduce the ratio of *n* to *b* (thereby reducing the amount of compression), or rely on special constrained ATPG to avoid such test cubes if possible. The proposed scheme avoids this problem by having a **second stage** of linear expansion through the use of the small vertical LFSR. Note that this is similar to the scheme in [Rajski 02]. The advantage of having the small vertical LFSR is that it will cycle the free-variables that are injected into it so that the set of linear equations for each bit-slice of the scan chains will depend not just on b free-variables as in [Bayraktaroglu 01, 03], but on free-variables injected across multiple clock cycles. Thus the excess free variables in lightly specified bit-slices can be used to solve for the highly specified bit-slices. This adds a lot of flexibility in solving the equations thus enabling a higher encoding efficiency. It also allows the ratio of n to b to be scaled much higher without greatly reducing the space of possible test cubes that can be generated.

A minimum of m clock cycles are required to fill the scan chains. This will inject mb free-variables into the small vertical LFSR. In the best case, it may be possible to solve the system of linear equations for the test cube after m clock cycles. However, if the number of specified bits in the test cube is more than mb, then more free-variables are needed. In this case, additional clock cycles beyond what is needed to initially fill the scan chains with data are required. The key idea is that the tester keeps shifting free-variables into the LFSR until it is possible to solve the system of linear equations for the test cube. Whereas the scheme in [Rajski 02] uses a fixed number of clock cycles for generating all test cubes (whether that many are needed or not), the proposed scheme uses only as many clock cycles as are needed for each test cube. As soon as it is possible to solve the system of linear equations, then the test cube is generated and applied to the circuit-under-test (CUT). The output response can be shifted out and compacted with a multi-input signature register (MISR) as the next test cube is generated. The solution to the system of linear equations gives the values for the free-variables (i.e., the bits stored on the tester) that will produce the test cube.

At any point, the small vertical LFSR can only store as many linear combinations of free-variables as the size of the LFSR (i.e., n). Thus if the test cube has too many specified bits, it may not be possible to solve the system of linear equations even if it is run for an infinite number of clock cycles because of linear dependencies. To avoid putting any limits on the number of specified bits per test cube, the proposed scheme uses a **third stage** of linear expansion through the use of the large horizontal LFSRs. Each of these large horizontal LFSRs has n/b external inputs coming from the small vertical LFSR and indirectly from the outputs of the linear combinational expansion circuit. Let one of the n/b inputs to each of the b large horizontal LFSRs be labeled as I_1, I_2, …, I_b, as shown in Fig. 3. Then in the proposed scheme, the linear combinational expansion circuit is designed in such a way that this set of b inputs taken together form a linearly independent set of equations in terms of the b free-variables arriving each clock cycle. Since there are b free-variables arriving each clock cycle, and b large horizontal LFSRs, it is always possible to do this. In the trivial case, each of the inputs, I_1, I_2, …, I_b, could be directly connected to one of the b inputs coming from the tester (in the general case, linearly independent equations in terms of say 3 free-variables each could be constructed). This ensures that the contents of each of the b large horizontal LFSRs can be loaded independently with any specified set of values. In at most *(n/b)(m)* clock cycles, the set of large horizontal LFSRs could be put into any possible state. In the worst case, it is equivalent to having a separate serial input to each of the b large horizontal LFSRs that serially loads each LFSR with whatever the desired state is.

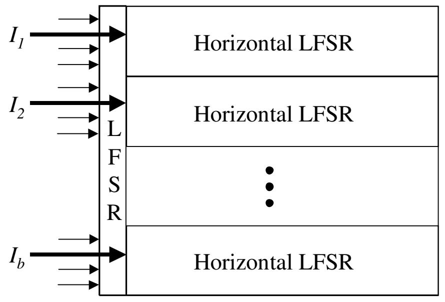

Figure 3. Set of b Linearly Independent Inputs

If it is necessary to use the proposed scheme to load fully specified test cubes, then the small vertical LFSR could be implemented separately from the scan cells such that all the scan cells are included in only the large horizontal LFSRs. In this case, any test cube (even fully specified test cubes) can be encoded with no more than *(n/b)(m)* clock cycles. Note that *(n/b)(m)* is equal to the number of clock cycles that would be required to load the scan chains with b channels from the tester if no compression was used. When the small vertical LFSR is constructed from the scan cells themselves as shown in Fig. 2, then there may be some fully or nearly fully specified test cubes that could not be encoded because of linear dependencies between the small vertical LFSR and the large horizontal LFSRs. In general, the proposed scheme would not be used for nearly fully specified test cubes, and thus constructing the small vertical LFSR from the scan cells themselves will not cause any problem.

One question that may arise is given the fact that all the test cubes can be generated just using the large horizontal LFSRs, why bother with the small vertical LFSR. The reason is that for less specified test cubes, using only the large horizontal LFSRs may have much less encoding efficiency because it takes m cycles before the large horizontal LFSRs are initialized and the feedback is enabled. Thus the large horizontal LFSRs are much slower in cycling the free-variables from multiple clock cycles compared with the small vertical LFSR. One of the powerful aspects of the proposed scheme is combining all three stages of linear expansion together so that it achieves high encoding efficiency regardless of the distribution of specified bits in the test cubes.

Given a test cube with s specified bits, it can be assumed that at least $\lceil s/b \rceil$ clock cycles will be needed before there is any chance of solving it. The linear decompressor is symbolically simulated for $\lceil s/b \rceil$ clock cycles. The system of linear equations is then formed for the s specified bits and a linear equation solver is called to try to solve it. If it cannot be solved, then the linear decompressor is simulated for another clock cycle (adding more free-variables) and a new system of linear equations is formed and an attempt is made to solve it. This continues until a solution to the system of linear equations is found. Note that the symbolic simulation process, forming the system of linear equations, and solving it can all be done very quickly.

5. Transmitting Control Information

In the proposed scheme, since the number of clock cycles needed to decompress each test cube is variable, this information needs to be transmitted to the on-chip test controller so it knows when to apply the system clock. The on-chip test controller has a *SCAN_COUNTER* which counts how many clock cycles have elapsed since the last scan vector was applied to the CUT. There also is a *FINAL_COUNT* register that stores the number of clock cycles that it will take to generate the current test cube. This is compared with the contents of the *SCAN_COUNTER* to determine when the test cube should be applied to the CUT. The value of the FINAL_COUNT for each test cube can be transmitted in the first clock cycle(s) when generating the test cube. The first bits coming from the tester for each test cube can be loaded into the FINAL_COUNT register to initialize it for each test cube. Alternatively, if the test cubes are ordered so that the value of FINAL_COUNT monotonically increases between test cubes, then only the increment in FINAL_COUNT between two test cubes needs to be transmitted.

6. Experimental Results

To see how the encoding efficiency for the proposed 3-stage scheme varies with respect to different distributions of specified bits and different scan architectures, the first set of experiments was performed on randomly generated test cubes. The results are shown in Table 1. The number of channels from the tester, b, was 16. Keeping the total number of scan cells constant across the different architectures, the number of scan chains, n, was taken as 64, 128, 256, 512, and 1024. Randomly generated test cubes were encoded and the encoding efficiency was calculated as the number of specified bits divided by the number of bits required to encode them. Results are shown for both the 3-stage scheme as well as a 2-stage decompressor (only combinational expander and vertical LFSR) both using a variable number of free-variables per test cube. The reason for showing both the 2-stage and 3-stage case is to illustrate the importance of the 3rd stage in achieving high encoding efficiency for non-uniform specified bit distributions where the number of specified bits per test cube varies. The first row for each scheme corresponds to the case where the number of specified bits per test cube varied randomly in the range between 2-5%. The second row is for 2-10%, followed by 2-20%, and 2-50%.

While the 3-stage scheme is able to fully encode all the test cubes regardless of the scan architecture and the specified bit distribution, the two stage scheme is unable to perform similarly ('--' indicates the cases for which encoding was not possible). Note that as the number of scan chains increases, the encoding efficiency of the 3-stage scheme increases significantly. This also has the advantage of reducing the test time, since the length of each scan chain is smaller for the architectures having larger number of scan chains. For the two stage scheme

Table 1. Comparison between Two-Stage and Three-Stage Linear Decompressors for Non-Uniform Specified Bit Distributions

Linear Decompressor	Random Test Cubes	Scan Architecture				
	Specified Bit Distribution	n=64 m=2048	n=128 m=1024	n=256 m=512	n=512 m=256	n=1024 m=128
2-Stage	2%-5%	.14	.28	**.56**	--	--
	2%-10%	.24	**.48**	--	--	--
	2%-20%	**.44**	--	--	--	--
	2%-50%	--	--	--	--	--
3-Stage	2%-5%	.14	.28	.56	.90	**.99**
	2%-10%	.24	.48	.80	.95	**.99**
	2%-20%	.44	.74	.90	.97	**.99**
	2%-50%	.77	.90	.96	**.99**	**.99**

to be able to encode all the test cubes, the number of scan chains needs to be small enough so that the most specified test cubes can be encoded. However, having a smaller number of scan chains reduces the encoding efficiency for the less specified test cubes. For example, for test cubes with 5% specified bits, the 2-stage scheme cannot encode them with 512 scan chains, so the number of scan chains has to be reduced to 256. However, this limits the encoding efficiency that can be achieved for the 2% specified test cubes in the same test set. Thus, the overall encoding efficiency for a range of test cubes between 2%-5% is limited to 0.56. The 3-stage scheme also has similarly limited encoding efficiency with 256 scan chains, however, the advantage of the 3-stage scheme is that the number of scan chains can be increased to 1024 while still being able to solve for the most specified test cubes. By using 1024 scan chains, an encoding efficiency of 0.99 can be achieved even for 2% specified test cubes. Thus with the proposed 3-stage scheme, the encoding efficiency can consistently be made high regardless of the distribution of specified bits by using a scan architecture with a sufficiently large number of scan chains. Note that if the number of specified bits is kept relatively uniform across all test cubes by constraining the ATPG (as is done in the methodology proposed in [Rajski 02]), then a 2-stage scheme works very well and there is no need to go to a 3-stage scheme. The advantage of the 3-stage scheme comes only when the specified bit distribution is non-uniform.

A second set of experiments were performed on the largest ISCAS 89 benchmark circuits. For each circuit, ATPG was performed to generate test cubes for the non-redundant faults. The set of test cubes was encoded using the proposed scheme assuming 8 channels coming from the tester and expanding to fill the scan chains. The results are shown in Table 2. The total number of specified bits in each test vector is shown followed by the number of bits that need to be stored on the tester for the proposed scheme. Note that the proposed method has a high encoding efficiency bringing the test data storage requirements down close to the number of specified bits. The encoding efficiency is not as high as in Table 1 because for these small circuits, s_{avg} is small such that the control bits have a more significant impact on the encoding efficiency.

A comparison was made with the scheme in [Bayraktaroglu 03] in Table 3. This scheme uses only a linear combinational circuit. It requires a special constrained ATPG to ensure all test cubes can be encoded. As can be seen, the proposed scheme significantly reduces the tester storage requirements by configuring the scan cells into an LFSR.

Table 4 shows a comparison of the results for the proposed scheme with a variety of other test data compression schemes. The proposed scheme requires less test data storage for all circuits in comparison with these approaches. Note that it was not possible to directly compare results for the proposed scheme with those in [Könemann 01], [Rajski 02], and [Volkerink 03], since those papers only show results for circuits that are not publicly available.

Table 2. Results for Proposed Scheme Using 8 Tester Channels

Circuit Name	Num Scan Chains	Num. Test Cubes	Total Num. Bits	Num. Specified Bits	Test Storage (bits)	Enc. Eff.
s13207	175	255	178,500	9,335	11,320	0.82
s15850	153	142	86,762	10,452	11,584	0.90
s38417	185	105	174,720	29,847	30,560	0.98
s38584	183	192	281,088	25,636	27,248	0.94

Table 3. Comparison with [Bayraktaroglu 03]

Circuit		[Bayraktaroglu 03]			Proposed Scheme		
Name	Scan Chains	Num. Chan.	Num. Vect.	Test Storage	Num. Chan.	Num. Vect.	Test Storage
s13207	175	19	258	19,608	8	255	11,320
s15850	153	18	167	12,024	8	142	11,584
s38417	185	19	317	54,207	8	105	30,560
s38584	183	19	185	28,120	8	192	27,248

Table 4. Comparison of Test Data for Different Encoding Schemes

Circuit Name	MinTest [Hamzaoglu 98]		Illinois Scan Architecture [Hamzaoglu 99]		FDR Codes [Chandra 01]		Mutation Encoding [Reda 02]		Seed Overlapping [Rao 03]		Proposed Scheme	
	Num. Vect.	Total Bits	Num. Vect.	Total Bits	Num. Vect.	Total Bits	Num. Vect.	Total Storage	Num. Vect.	Total Bits	Num. Vect.	Total Bits
s13207	233	163,100	273	109,772	236	30,880	274	16,913	272	17,970	255	11,320
s15850	96	58,656	178	32,758	126	26,000	185	14,676	174	15,774	142	11,584
s38417	68	113,152	337	96,269	99	93,466	231	55,848	288	60,684	105	30,560
s38584	110	161,040	239	96,056	136	77,812	220	47,886	215	31,061	192	27,248

7. Conclusions

A new continuous-flow scheme for linear decompression with a variable number of free-variables per test cube was described here. The proposed scheme can achieve high encoding efficiency regardless of the variance between s_{max} and s_{avg}. It has important applications for compressing test data when it is not possible or desirable to use a special constrained ATPG. The hardware requirements for the proposed scheme are small since it makes use of the scan cells themselves to configure the decompressor.

Acknowledgements

This material is based on work supported in part by Intel, in part by IBM, and in part by the National Science Foundation under Grant No. CCR-0306238.

References

[Bayraktaroglu 01] Bayraktaroglu, I., and A. Ogailoglu, "Test Volume and Application Time Reduction Through Scan Chain Concealment," *Proc. of Design Automation Conference*, pp. 151-155, 2001.

[Bayraktaroglu 03] Bayraktaroglu, I., and A. Ogailoglu, "Decompression Hardware Determination for Test Volume and Time Reduction through Unified Test Pattern Compaction and Compression," *Proc. of VLSI Test Symposium*, pp. 113-118, 2003.

[Chandra 00] Chandra, A., and K. Chakrabarty, "Test Data Compression for System-on-a-Chip Using Golomb Codes," *Proc. of VLSI Test Symposium*, pp. 113-120, 2000.

[Chandra 01] Chandra, A., and K. Chakrabarty, "Frequency-Directed Run Length (FDR) Codes with Application to System-on-a-Chip Test Data Compression," *Proc. of VLSI Test Symposium*, pp. 42-47, 2001.

[Das 00] Das, D., and N.A. Touba, "Reducing Test Data Volume Using External/LBIST Hybrid Test Patterns," *Proc. of International Test Conference*, pp. 115-122, 2000.

[Dorsch 01] Dorsch, R., and H.-J. Wunderlich, "Tailoring ATPG for Embedded Testing," *Proc. of Int. Test Conf.*, pp. 530-537, 2001.

[Gonciari 02] Gonciari, P.T., B. Al-Hashimi, and N. Nicolici, "Improving Compression Ratio, Area Overhead, and Test Application Time for System-on-a-Chip Test Data Compression/Decompression," *Proc. of Design Automation and Test in Europe (DATE)*, pp. 604-611, 2002.

[Hamzaoglu 98] Hamzaoglu, I., and J. Patel, "Test Set Compaction Algorithms for Combinational Circuits," *Proc. of International Conf. on Computer-Aided Design (ICCAD)*, pp. 283-289, 1998.

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

[Hellebrand 95a] Hellebrand, S., 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. on Computers*, Vol. 44, No. 2, pp. 223-233, Feb. 1995.

[Hellebrand 95b] Hellebrand, S., B. Reeb, S. Tarnick, and H.-J. Wunderlich," Pattern Generation for a Deterministic BIST Scheme," *Proc. of International Conference on Computer-Aided Design (ICCAD)*, pp. 88-94, 1995.

[Hellebrand 00] Hellebrand, S., H.-G. Liang, and H.-J. Wunderlich, "A Mixed Mode BIST Scheme Based on Reseeding of Folding Counters," *Proc. of International Test Conf.*, pp. 778-784, 2000.

[Hsu 01] Hsu, F.F., K. M. Butler, J. H. Patel, "A Case Study on the Implementation of the Illinois Scan Architecture," *Proc. of International Test Conference*, pp. 538-547, 2001.

[Jas 98] Jas, A., and N.A. Touba, "Test Vector Decompression Via Cyclical Scan Chains and Its Application to Testing Core-Based Designs", *Proc. of Int. Test Conference*, pp. 458-464, 1998.

[Jas 99] Jas, A., J. Ghosh-Dastidar, and N.A. Touba, "Scan Vector Compression/Decompression Using Statistical Coding", *Proc. of IEEE VLSI Test Symposium*, pp. 114-120, 1999.

[Jas 00] Jas, A., B. Pouya, and N.A. Touba, "Virtual Scan Chains: A Means for Reducing Scan Length in Cores", Proc. of VLSI Test Symposium, pp. 73-78, 2000.

[Khoche 02] Khoche, A., E.H. Volkerink, J. Rivoir, and S. Mitra, "Test Vector Compression Using EDA-ATE Synergies," *Proc. of VLSI Test Symposium*, pp. 97-102, 2002.

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

[Krishna 02] Krishna, C.V., and N.A. Touba, "Reducing Test Data Volume Using LFSR Reseeding with Seed Compression ", *Proc. of IEEE International Test Conference*, pp. 321-330, 2001.

[Könemann 91] Könemann, B., "LFSR-Coded Test Patterns for Scan Designs," *Proc. of European Test Conf.*, pp. 237-242, 1991.

[Könemann 01] Könemann, B., "A SmartBIST Variant with Guaranteed Encoding" *Proc. of Asian Test Symposium*, pp. 325-330, 2001.

[Li 03] Li, L., and K. Chakrabarty, "Test Data Compression Using Dictionaries with Fixed-Length Indices," *Proc. of VLSI Test Symposium*, pp. 219-224, 2003.

[Liang 01] Liang, H.-G., S. Hellebrand, and H.-J. Wunderlich, "Two-Dimensional Test Data Compression for Scan-Based Deterministic BIST," *Proc. of International Test Conf.*, pp. 894-902, 2001.

[Mitra 03] Mitra, S., and K.S. Kim, "XMAX: X-tolerant architectures for Maximal Test Compression," *Proc. of International Conference on Computer Design*, pp. 326-330, 2003.

[Mrugalski 03] Mrugalski, G., J. Rajski, and J. Tyszer, "High Speed Ring Generators and Compactors of Test Data," *Proc. of VLSI Test Symposium*, pp. 57-62, 2003.

[Rajski 98a] Rajski, J., J. Tyszer, and N. Zacharia, "Test Data Decompression for Multiple Scan Designs with Boundary Scan", *IEEE Trans. on Comp.*, Vol. 47, No. 11, pp. 1188-1200, Nov. 1998.

[Rajski 98b] Rajski, J., and J. Tyszer, "Automated Synthesis of Large Phase Shifters for Built-In Self-Test", *Proc. of VLSI Test Symposium*, pp. 218-224, 1998.

[Rajski 99] Rajski, J., and J. Tyszer, "A Parallel Decompressor and Related Method and Apparatuses", USA Patent #5,991,909, 1999.

[Rajski 02] Rajski, J., *et al.,* "Embedded Deterministic Test for Low Cost Manufacturing Test," *Proc. of Int. Test Conf.*, pp. 301-310, 2002.

[Rao 03] Rao, W., I. Bayraktaroglu, and A. Orailoglu, "Test Application Time and Volume Compression through Seed Overlapping," *Proc. of Design Automation Conference*, pp. 732-737, 2003.

[Reda 02] Reda, S., and A. Orailoglu, "Reducing Test Application Time Through Test Data Mutation Encoding", *Proc. of Design, Automation, and Test in Europe*, pp. 387-393, 2002.

[Reddy 02] Reddy, S., K. Miyase, S. Kajihara, and I. Pomeranz, "On Test Data Volume Reduction for Multiple Scan Chain Designs", *Proc. of VLSI Test Symposium*, pp. 103-108, 2002.

[Volkerink 02] Volkerink, E.H., A. Khoche, and S. Mitra, "Packet-based Input Test Data Compression Techniques," *Proc. of International Test Conference*, pp. 154-163, 2002.

[Volkerink 03] Volkerink, E.H., and S. Mitra, "Efficient Seed Utilization for Reseeding Based Compression," *Proc. of VLSI Test Symposium*, pp. 232-237, 2003.

[Wolff 02] Wolff, F.G., and C. Papachristou, "Multiscan-based Test Compression and Hardware Decompression Using LZ77," *Proc. of International Test Conference*, pp. 331-339, 2002.

[Zacharia 95] Zacharia, N., J. Rajski, and J. Tyszer, "Decompression of Test Data Using Variable-Length Seed LFSRs," *Proc. of VLSI Test Symposium*, pp. 426-433, 1995.

[Zacharia 96] Zacharia, N., J. Rajski, J. Tyszer, and J. Waicukauski "Two Dimensional Test Data Decompressor for Multiple Scan Designs," *Proc. of International Test Conf.*, pp. 186-194, 1996.