Improving Encoding Efficiency for Linear Decompressors Using Scan Inversion

Kedarnath J. Balakrishnan and Nur A. Touba

Computer Engineering Research Center
Department of Electrical and Computer Engineering
University of Texas, Austin, TX 78712-1084
E-mail: {kjbala, touba}@ece.utexas.edu

Abstract

The output space of a linear decompressor must be sufficiently large to contain all the test cubes in the test set. The idea proposed in this paper is to use scan inversion to transform the output space of a linear decompressor so as to reduce the number of inputs required thereby increasing the encoding efficiency while still keeping all the test cubes in the output space. Any existing method for designing a linear decompressor (either combinational or sequential) can be used first to obtain the best linear decompressor that it can. Using that linear decompressor as a starting point, the proposed method improves the encoding efficiency further. The key property used by the proposed method is that scan inversion is a linear transformation of the output space and thus the output space remains a linear subspace spanned by a Boolean matrix. Using this property, a systematic procedure based on linear algebra is described for selecting the set of inverting scan cells to maximize encoding efficiency. Experiments indicate that significant improvements in encoding efficiency can be achieved.

1. Introduction

As system-on-chip designs become increasingly complex containing many cores and requiring large amounts of test data, the test data storage requirements on the tester and the test data bandwidth requirements between the tester and chip are growing rapidly [Khoche 00]. Test data compression techniques provide a means to reduce these requirements thereby allowing less expensive testers to be used as well as reducing test time. Compressing the output response is relatively easy since lossy compression techniques can be employed, e.g., using a multiple input signature register (MISR). However, compressing test vectors is much more difficult because lossless compression techniques must be used. Recently, as reducing test vector volume has become such an important problem, a lot of research has been done on lossless compression techniques for test vectors.

An important class of test vector compression schemes involves using a linear decompressor which uses only linear operations to decompress the test vectors. 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], SmartBIST from IBM/Cadence [Könemann 01], and DBIST [Chandramouli 03] from Synopsys are based on linear decompressors. Linear decompression exploits 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. The encoding efficiency of the linear decompressor is defined as the ratio of the number of specified bits in the test set to the number of bits stored on the tester. Higher encoding efficiency means more compression. This paper describes a new technique that can significantly improve the encoding efficiency of a linear decompressor. It is applicable to any linear decompressor including both combinational and sequential.

If there are c scan cells, then the space of all possible scan vectors is 2^c. The *output space* of a linear decompressor is the set of scan vectors that can be generated by the linear decompressor. Each bit stored on the tester can be thought of as a "free-variable" that can be assigned any value (0 or 1). Consider the case where the linear decompressor receives an input sequence from the tester consisting of n free-variables when generating a scan vector. Assuming the linear decompressor is always initialized to the same state before generating each scan vector (if it is a sequential circuit), then the size of the output space of the linear decompressor is less than or equal to 2^n (since that is the number of possible unique input sequences that could be applied to it). The output space will be equal to 2^n if every input sequence maps to a

unique scan vector, and less than 2^n if some input sequences map to the same scan vector. In the degenerate case where the linear decompressor is just a set of wires directly connecting each scan chain to a tester channel, then $n=c$ and the content of every scan cell is equal to a

of all possible scan vectors. Linear decompressors have some nice properties compared with non-linear decompressors. They generally have a larger output space for the same n because of the use of XOR gates which minimize the number of input sequences that map to the same scan vector. Another very important property is that the output space of a linear decompressor is a linear subspace spanned by a Boolean matrix, $\boldsymbol{A}_{c \times n}$. The Boolean matrix $\boldsymbol{A}$ can be obtained by symbolic simulation of the linear decompressor (see [Krishna 01] for details). Each row in $\boldsymbol{A}$ corresponds to a scan cell and each column corresponds to a free-variable in the input sequence. The advantage of having the output space be defined by $\boldsymbol{A}$ is that determining whether a particular test cube is contained in the output space and the corresponding input sequence to generate it can be quickly done by solving a set of linear equations using Gaussian elimination.

In order to be able to compress a test set, the output space of the linear decompressor must contain all the test cubes in the test set. When using an LFSR as a linear decompressor, it has been shown that if the number of free-variables used to generate a test cube is 20 more than the number of specified bits in a test cube, then the probability of the test cube not being in the output space is less than 10^{-6} [Könemann 91]. However, for a given test set, the number of free-variables can be reduced further provided the corresponding reduced output space still contains all the test cubes in the test set. Reducing the number of free-variables increases the encoding efficiency and hence compression. As the number of free-variables are reduced, the output space becomes smaller and smaller until a point is reached where one or more test cubes are no longer in the output space. This point terminates the reduction in free-variables and limits the encoding efficiency that can be achieved by the linear decompressor for a particular test set.

The idea proposed in this paper is to alter and reshape the output space of a linear decompressor using scan inversion. This can allow the number of free-variables to be reduced further while still keeping all the test cubes in the output space thereby increasing the encoding efficiency. The encoding efficiency can often be increased above 1 using the proposed approach. Any method for designing a linear decompressor can be used first to obtain the best linear decompressor that it can. Using that linear decompressor as a starting point, the proposed method reduces the number of free-variables further to improve the encoding efficiency by incorporating scan inversion. The key property used by unique free-variable such that the output space of the linear decompressor contains all possible scan vectors. However, in order to get compression, n needs to be less than c, and thus in the general case, the output space of the linear decompressor will be a subset
the proposed method is that scan inversion is a linear transformation of the output space and thus the output space remains a linear subspace spanned by a Boolean matrix. Using this property, a systematic procedure based on linear algebra is described for selecting the set of inverting scan cells to maximize encoding efficiency. A nice feature of the proposed method is that it can be implemented without any hardware overhead (this is described in detail in Sec. 3).

The paper is organized as follows: Sec. 2 describes how this paper relates to previous work. Sec. 3 discusses how scan inversion can be implemented in hardware. Section 4 presents a procedure for selecting which set of scan cells to invert based on linear algebra. Sec. 5 explains how the proposed method can be used to improve encoding efficiency for linear decompressors. Sec. 6 shows experimental results. Sec. 7 is a conclusion.

2. Related Work

A number of different techniques for designing linear decompressors have been proposed in the literature. The original idea of using an LFSR as a linear decompressor and solving for test cubes using linear algebra was described in [Könemann 91]. Several techniques for improving the encoding efficiency for linear decompressors based on LFSRs were described in [Venkataraman 93], [Hellebrand 95ab], [Zacharia 95, 96], [Rajski 98], and [Krishna 01, 02].

Linear decompressors that can receive data from the tester in a continuous-flow (i.e., "streaming" data) are especially useful for test data compression. Continuous-flow linear decompressors 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. There is no need for any special scheduling or synchronization. A number of techniques for designing both combinational and sequential continuous-flow linear decompressors have been proposed. Combinational continuous-flow linear decompressors are described in [Hamzaoglu 99], [Hsu 01], [Bayraktaroglu 01, 03], [Mitra 03], and [Krishna 03]. Sequential continuous-flow linear decompressors are described in [Jas 00], [Könemann 01], [Rajski 02], [Volkerink 03], [Rao 03], and [Krishna 04].

The method described in this paper can be used in conjunction with any linear decompressor including all of the ones referenced above to improve the encoding efficiency further. It transforms the output space of the

linear decompressor in a way that allows a smaller number of free-variables from the tester to be used to encode the test set. The proposed idea has some relation to the idea of transforming the output space of a test pattern generator to encode test cubes which has been investigated in the past in the context of built-in self-test (BIST). Techniques for designing a non-linear circuit to transform the output space of a pseudo-random generator to encode test cubes were described in [Touba 95ab, 01], [Chatterjee 95], [Wunderlich 96], and [Kiefer 97, 98]. The proposed method differs significantly from these methods in that it uses linear transformations, is based on linear algebra, is targeted towards lossless test vector compression, and can be implemented without any overhead.

3. Scan Inversion

The proposed method involves inverting a set of scan cells to transform the output space of the linear decompressor. Implementing inverted scan cells can be accomplished either by explicitly inserting inverters in the scan chains, or by simply using the *Q'* output instead of the *Q* output when connecting the output of one scan cell to the next scan cell. An example of inverting the contents of a scan cell is shown in Fig. 2. Figure 1 shows a normal scan chain with no inversion, while Fig. 2 shows the scan chain with the third scan cell inverted. This is accomplished by inverting before and after the third scan cell. If the same scan vector was shifted into both the normal scan chain in Fig. 1 and the scan chain in Fig. 2, the contents of the third scan cell in Fig. 2 would have the opposite value from what it would be in the normal scan chain while the contents of all other scan cells would be the same. Also, when the output response is shifted out, the bit corresponding to the third scan cell will have the opposite value from what it normally would have.

While the example in Fig. 2 only involves inverting one scan cell, any set of scan cells in the scan chain can be inverted by making appropriate connections from either *Q* or *Q'* to the next scan cell. To invert the first scan cell in a scan chain, either an inverter can be placed on the *scan_data_in (SDI)* input or the XOR (XNOR) gate that is driving the SDI input can simply be changed to an XNOR (XOR) gate.

If the inverted scan cells are implemented by simply changing some connections from *Q* to *Q'* and changing some XOR gates to XNOR gates, then there is effectively no overhead for using the proposed method. Even if explicit inverters are used, the overhead would still be small.

The proposed method does not involve any special ordering of the scan cells. The scan chains can be ordered in any manner to optimize the layout.

4. Selecting Set of Inverted Scan Cells

The proposed method involves selecting a set of scan cells to be inverted. If there are *c* scan cells, then there are 2^c different ways the scan cells can be inverted each of which results in a different transformation of the output space of the linear decompressor. Given a particular linear decompressor, the goal is to select the set of inverted scan chains so that the resulting transformed output space of the linear decompressor will contain all of the test cubes in the test set. This section describes a systematic procedure based on linear algebra for doing this.

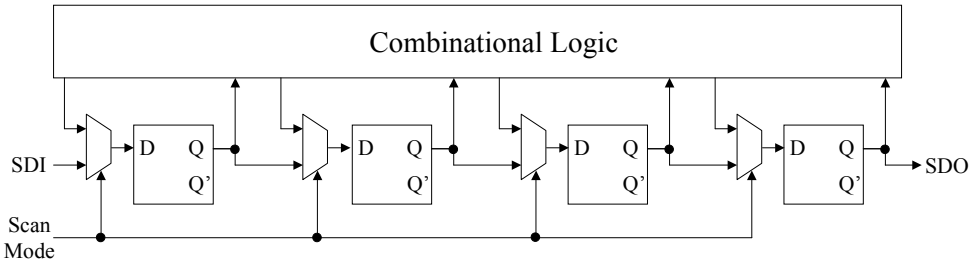

Figure 1. Normal Scan Chain

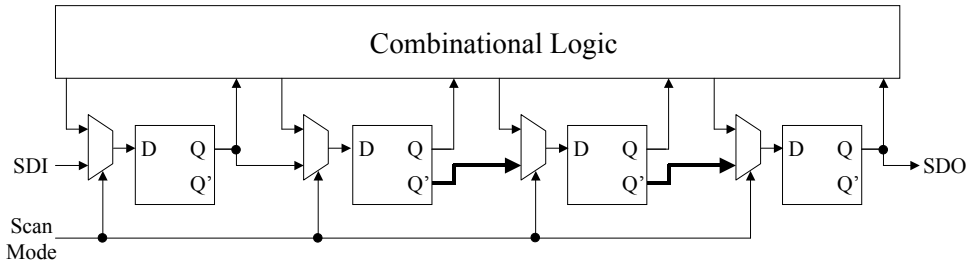

Figure 2. Scan Chain with 3rd Scan Cell Inverted

The first step is to obtain the Boolean matrix ***A*** that spans the output space of the linear decompressor. This can be obtained using symbolic simulation (see [Krishna 01] for details).

In order for a test cube to be in the output space of the linear decompressor, there must exist a solution to the system of linear equations, ***Ax=b*** where ***x*** is an assignment of values to the free-variables that are inputs to the decompressor when generating the test cube, and ***b*** is the values of each bit in the test cube. Note that for the unspecified bits in the test cube, there is no need to solve the linear equations, so it is only the linear equations (rows) corresponding to the specified bits in ***b*** that need to be considered. Gauss-Jordan Elimination [Cullen 97] can be used to perform rows operations that transform a set of columns into an identity matrix (these columns are called the *pivots*). An example of a system of linear equations for a test cube t_1 is shown in Fig. 3, and the corresponding system after Guass-Jordan Elimination is shown in Fig. 4. The vector ***b*** after Guass-Jordan Elimination will be referred to as ***z*** to eliminate confusion. The rows after Gauss-Jordan Elimination can be classified as either *pivoted rows* or *linearly dependent rows*. The pivoted rows have pivots while the linearly dependent rows are all 0. In the example in Fig. 4, the first three rows are pivoted while the last two are linearly dependent. If all rows are pivoted, then a solution to the system of linear equations exists, and hence the test cube is in the output space of the linear decompressor. If some of the rows are linearly dependent, then a solution only exists if all of the corresponding values in ***z*** (the vector ***b*** after Guass-Jordan Elimination) are equal to 0 for the linearly dependent rows. If there is a linearly dependent row whose corresponding value in ***z*** is equal to 1, then a solution does not exist. In Fig. 4, the fourth row is linearly dependent, but the corresponding value in ***z*** is 0 so that is okay. However, the fifth row is also linearly dependent, but the corresponding value in ***z*** is 1, and thus there is no solution. This is easy to see because in the original system of linear equations in Fig. 3, both rows 2 and 5 are identical in ***A***, however, the corresponding outputs for those two rows in ***b*** have opposite values. Obviously there is no assignment to ***x*** that will simultaneously solve the linear equations for both rows 2 and 5.

For the example in Fig. 3, let the specified values in ***b*** correspond to scan cells c_1 through c_5. If either scan cell c_2 or scan cell c_5 were inverted, the system of linear equations would become solvable. For example, if scan cell 5 was inverted, then the last row in ***b*** would be changed from a 1 to a 0. Now the fact that row 5 is the same as row 2 in ***A*** is not a problem because the corresponding values in ***b*** for those two rows are identical. This is an example of how scan inversion can be used to make a test cube solvable.

$$\mathbf{A}\ \mathbf{x} = \mathbf{b} \qquad c_1\ c_2\ c_3\ c_4\ c_5$$

$$\begin{pmatrix} 1&0&1&1&1&0\\ 0&1&0&1&0&1\\ 0&0&1&1&1&1\\ 1&0&0&0&0&1\\ 0&1&0&1&0&1 \end{pmatrix} \begin{pmatrix} x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6 \end{pmatrix} = \begin{pmatrix} 0\\ 0\\ 1\\ 1\\ 1 \end{pmatrix} \vdots \begin{pmatrix} 1&0&0&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0\\ 0&0&0&1&0\\ 0&0&0&0&1 \end{pmatrix}$$

Figure 3. System of Linear Equations for Test Cube t_1

Pivots | **z** | Dependency

Pivoted Rows	$\begin{pmatrix} 1&0&0&0&0&1\\ 0&1&0&1&0&1\\ 0&0&1&1&1&1 \end{pmatrix}$	$\begin{pmatrix} 1\\ 0\\ 1 \end{pmatrix}$	$\begin{pmatrix} 1&0&1&0&0\\ 0&1&0&0&0\\ 0&0&1&0&0 \end{pmatrix}$
Linearly Dependent	$\begin{pmatrix} 0&0&0&0&0&0\\ 0&0&0&0&0&0 \end{pmatrix}$	$\begin{pmatrix} 0\\ 1 \end{pmatrix}$	$\begin{pmatrix} 1&0&1&1&0\\ 0&1&0&0&1 \end{pmatrix}$

Constraints: $i_1 \oplus i_3 \oplus i_4 = 0$; $i_2 \oplus i_5 = 1$

Figure 4. Gauss-Jordan Reduction for Test Cube t_1

$$\mathbf{A}\ \mathbf{x} = \mathbf{b} \qquad c_1\ c_2\ c_6\ c_7$$

$$\begin{pmatrix} 1&0&1&1&1&0\\ 0&1&0&1&0&1\\ 1&1&1&0&1&1\\ 0&1&0&0&0&1 \end{pmatrix} \begin{pmatrix} x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6 \end{pmatrix} = \begin{pmatrix} 0\\ 0\\ 1\\ 1 \end{pmatrix} \vdots \begin{pmatrix} 1&0&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&0&1 \end{pmatrix}$$

Figure 5. System of Linear Equations for Test Cube t_2

z | Dependency

Linearly Dependent	$\begin{pmatrix} 1&0&0&0&1&1\\ 0&1&0&0&0&1\\ 0&0&0&0&0&0\\ 0&0&0&1&0&0 \end{pmatrix}$	$\begin{pmatrix} 1\\ 0\\ 1\\ 0 \end{pmatrix}$	$\begin{pmatrix} 1&1&0&1\\ 0&1&0&0\\ 1&1&1&0\\ 0&1&0&1 \end{pmatrix}$

Constraints: $i_1 \oplus i_2 \oplus i_6 = 1$

Figure 6. Gauss-Jordan Reduction for Test Cube t_2

$$\mathbf{C}\ \mathbf{i} = \mathbf{z}$$

$$\begin{matrix} \text{Constraints for } t_1 \\ \text{Constraints for } t_2 \end{matrix} \begin{pmatrix} 1&0&1&1&0&0&0\\ 0&1&0&0&1&0&0\\ 1&1&0&0&0&0&1 \end{pmatrix} \begin{pmatrix} i_1\\ i_2\\ i_3\\ i_4\\ i_5\\ i_6\\ i_7 \end{pmatrix} = \begin{pmatrix} 0\\ 1\\ 1 \end{pmatrix}$$

Figure 7. Constraint Matrix for Test Set {*t1*, *t2*}

Let ***i*** = $(i_1, i_2, \ldots, i_c)$ be a vector in which if i_j =1 then scan cell j is inverted and if i_j =0 then scan cell j is not inverted. For a given test cube, a set of constraints on ***i*** for which the test cube is solvable can be determined as

follows. For the rows that are pivoted, there are no constraints on $\boldsymbol{i}$ since it doesn't matter what the corresponding value of $\boldsymbol{z}$ is for those rows since they are always solvable. For the linearly dependent rows, the corresponding value of $\boldsymbol{z}$ must be 0, so whatever scan inversion takes place must ensure that the value for that row is 0 after Guass-Jordan Elimination. This places constraints on $\boldsymbol{i}$. In the example in Fig. 4, the last two rows are linearly dependent, so constraints must be placed on $\boldsymbol{i}$ to ensure that the corresponding values in $\boldsymbol{z}$ for those rows are always 0. These constraints can be determined by augmenting the linear system with a dependency matrix which is a square matrix with the number of rows and columns equal to the number of rows in $\boldsymbol{b}$. The dependency matrix is initially an identity matrix since each value in $\boldsymbol{b}$ depends only on itself. The row operations that are performed during Gauss-Jordan Elimination are also applied to the dependency matrix. After Gauss-Jordan Elimination, the dependency matrix indicates which set of scan cells each value in $\boldsymbol{z}$ depends on. For each linearly dependent row, the constraints on $\boldsymbol{i}$ can be obtained by looking at the dependency for the value in $\boldsymbol{z}$ for that row. The value in $\boldsymbol{z}$ for each linearly dependent row must be 0, so a linear equation in terms of $\boldsymbol{i}$ can be written to ensure that the value in $\boldsymbol{z}$ will be 0. In the example in Fig. 4, the 4th row is linearly dependent and the corresponding value in $\boldsymbol{z}$ depends on scan cells c_1, c_3, and c_4. Since the corresponding value in $\boldsymbol{z}$ in the normal scan chain is 0, the constraint on scan inversion is that $i_1 \oplus i_3 \oplus i_4$ should be 0. In other words, either none of scan cells c_1, c_3, and c_4 should be inverted or two of them should be inverted (thus canceling each other out). If one or all three of them are inverted, then the value in $\boldsymbol{z}$ for that row will become 1 and no solution for the test cube will exist. Similarly, a linear equation for the constraint due to the 5th row can be obtained. In this case, for the normal scan chain the value of $\boldsymbol{z}$ for this row is 1, so it is necessary that one of the scan cells that it depends on be inverted, either c_2 or c_5 in order to get a solution for the test cube.

The procedure described above can be used to obtain a set of constraints on $\boldsymbol{i}$ for each test cube to make it solvable. Consider test cube t_2 whose system of linear equations is shown in Fig. 5. Note that this test cube has only 4 specified bits and in this case the specified bits are in scan cells c_1, c_2, c_6, and c_7. The system of linear equation after Guass-Jordan Elimination is shown in Fig. 6. As can be seen, there is one linearly dependent row in this case. The corresponding set of constraints on $\boldsymbol{i}$ can be obtained from the dependency matrix.

In order for the linear decompressor to be able to generate all of the test cubes in the test set, there needs to be a solution for all test cubes. Thus, $\boldsymbol{i}$ must be selected to allow all test cubes to be simultaneously solved. A solution for $\boldsymbol{i}$ can be obtained by forming a constraint matrix $\boldsymbol{C}$ that incorporates all of the constraints for all of the test cubes. Each constraint for each test cube can be added as a row in $\boldsymbol{C}$. For example, suppose the test set consisted of test cube t_1 and t_2 whose constraints were obtained in Figs. 4 and 6. Then the first two rows in $\boldsymbol{C}$ would correspond to the two linear constraint equations from Fig. 4, and the last row in $\boldsymbol{C}$ would correspond to the linear constraint equation from Fig. 6. The resulting constraint matrix is shown in Fig. 7. Gauss-Jordan Elimination can then be used to find a solution to the system of linear equations $\boldsymbol{Ci}=\boldsymbol{z}$. The solution for $\boldsymbol{i}$ gives the set of scan cells that need to be inverted so that the test set can be generated by the linear decompressor. If no solution exists, then it is not possible for the linear decompressor to be used to generate the test set under any set of inverted scan cells.

The complexity of the Gauss-Jordan Elimination method for solving a set of n linear equations with m variables is of the order of $O(nm^2)$. In the proposed scheme, as with any other linear test vector compression scheme, a set of linear equations needs to be solved for each test cube. Hence, if s is the number of specified bits in a cube, and c is the number of compressed bits for that cube, then solving for that cube will require $O(sc^2)$ time. The only additional task involved in the proposed scheme is to solve for the constraint matrix. The time complexity for this additional step will depend on how many equations there are in the constraint matrix and the number of scan cells that are included in the inversion constraints. Note that there is no need to include the scan cells that are not present in any inversion constraint for any test cube.

5. Using Scan Inversion to Increase Encoding Efficiency

Given a linear decompressor, the previous section described how to select a set of inverting scan cells so that all the test cubes in the test set will be in the output space (if such a set of inverting scan cells exists). This procedure can be used in two ways to increase encoding efficiency. One is to start with an initial decompressor design and reduce the number of free-variables that it receives per test cube from the tester as much as possible while still keeping the test set in the output space through scan inversion. The other is to use scan inversion to relax the constraints on the ATPG (automatic test pattern generation). The number of free variables per test cube is kept constant but each cube can have more specified bits. Hence the ATPG can do more static and dynamic compaction. Both of these applications will increase encoding efficiency and hence compression. They are discussed in detail in the following subsections.

5.1 Reducing Free-Variables per Test Cube

Any method can be used to design the best linear decompressor for a normal scan chain. Then the number of free-variables that are input to the decompressor per test cube can be incrementally reduced and the procedure in Sec. 4 can be used to see if it is possible to still solve for all the test cubes using scan inversion. If so, then this process of incrementally reducing the number of free-variables and checking for a solution is repeated until a point is reached when no further reduction in the number of free-variables per test cube is possible while still being able to solve for all test cubes. For example, for a continuous-flow decompressor (either sequential or combinational), the number of free-variables per test cube can be reduced by simply holding one tester channel input to a constant 0 when doing symbolic simulation to obtain the matrix $\boldsymbol{A}$ that spans the output space of the linear decompressor. This will reduce the rank of $\boldsymbol{A}$ and hence reduce the output space. However, if the procedure in Sec. 4 can still solve for all test cubes using scan inversion, then the linear decompressor design can be simplified by eliminating that one tester channel input that was held to 0 since it is no longer needed. This can be repeated to try to eliminate additional tester channel inputs. The end result will be a linear decompressor that generates the exact same test set, but uses fewer tester channels thereby reducing tester storage and bandwidth requirements.

5.2 Increasing Specified Bits per Test Cube

If the number of tester channels that are allocated for feeding the linear decompressor is fixed, then another way that scan inversion can be used is to increase encoding efficiency is to allow more specified bits per test cube. Some test compression methodologies (e.g., [Bayraktaroglu 01, 03], [Rajski 02]) involve fixing the decompressor design and then constraining the ATPG so that the resulting test cubes will be in the output space of the decompressor. The constraints on the ATPG reduce the amount of static and dynamic compaction that are performed and therefore can result in more test cubes and hence more test time. The scan inversion method proposed here can be used to allow more specified bits per test cube while still being able to solve for the test cube. This can be used to relax the constraints on the ATPG and thereby allow more static and dynamic compaction. Each time a test cube is generated by the ATPG any additional constraints for solving the test cube via scan inversion can be added to the constraint matrix $\boldsymbol{C}$ described in Sec. 4. If the additional constraints cause the constraint matrix to no longer have a solution, then that test cube cannot be accepted and the ATPG must find a less specified test cube. If the constraint matrix still has a solution, then the test cube can be accepted. The end result of relaxing the constraints on the ATPG is that more static and dynamic compaction can be performed thereby reducing the total number of test cubes and hence reducing both test time and tester storage requirements.

6. Experimental Results

Two sets of experiments were performed to evaluate the effectiveness of the proposed method. The first set of experiments was done on randomly generated test cubes for large industrial-size scan architectures. The second set of experiments was done on the largest ISCAS 89 benchmark circuits [Brglez 89]. For the randomly generated test cubes, experiments were performed using two different types of linear decompressors. The first was a combinational linear decompressor as shown in Fig. 8 using 512 scan chains and 1024 scan chains. The results are shown in Table 1. For the combinational decompressor with normal scan chains without inversion, the number of channels from the tester was 32. For each randomly generated test cube, the number of specified bits was incrementally increased until it could no longer be solved (i.e., it was no longer in the output space). The average percentage of specified bits per test cube that could be solved for is shown along with the corresponding encoding efficiency. This measures the best encoding efficiency that can be achieved for normal scan chains without scan inversion. Recall that the *encoding efficiency* is defined as the ratio of the number of specified bits in the test set to the number of bits stored on the tester. Note that since each test cube is encoded independently, there is no dependence on the number of test cubes. Results are then shown for scan inversion using it in the two ways described in Sec. 5. The first is using scan inversion to reduce the number of tester channels while still encoding the same set of test cubes as before. The number of reduced tester channels along with the resulting encoding efficiency that is achieved are shown in columns 6 and 7 respectively. The second way that scan inversion is used is to keep the tester channels at 32 and increase the percentage of specified bits per test cube as much as possible until it is no longer possible to solve for all the test cubes. The maximum percentage of specified bits and the resulting encoding efficiency are shown in the last two columns of Table 1. Note that when using scan inversion, the effectiveness reduces as the number of test cubes increases since it is harder to keep all of the test cubes in the output space. Results are shown for several different numbers of test cubes.

A number of interesting observations can be made from Table 1. Scan inversion is remarkably effective at improving the encoding efficiency for combinational decompressors (it is more than doubled in most cases). Typically the encoding efficiency for combinational decompressors is fairly low because of the fact that they

Table 1. Results for Combinational Linear Decompressor

Scan Chains	Num. Test Cubes	Combinational without Scan Inversion			Combinational with Scan Inversion			
		Tester Channels	Percentage Specified	Encoding Efficiency	Reduced Channels	Encoding Efficiency	Increased % Specified	Encoding Efficiency
512	200	32	2.7%	0.43	11	1.26	6.2%	0.99
	400	32	2.7%	0.43	13	1.06	5.7%	0.91
	600	32	2.7%	0.43	15	0.92	5.5%	0.88
	1000	32	2.7%	0.43	16	0.86	5.2%	0.83
1024	200	32	1.3%	0.42	11	1.21	3.5%	1.12
	400	32	1.3%	0.42	11	1.21	3.1%	0.99
	600	32	1.3%	0.42	13	1.02	2.9%	0.93
	1000	32	1.3%	0.42	14	0.95	2.8%	0.90

Table 2. Results for Sequential Linear Decompressor

Scan Chains	Num. Test Cubes	Sequential without Scan Inversion			Sequential with Scan Inversion			
		Tester Channels	Percentage Specified	Encoding Efficiency	Reduced Channels	Encoding Efficiency	Increased % Specified	Encoding Efficiency
512	200	16	3.0%	0.94	14	1.07	3.6%	1.13
	400	16	3.0%	0.94	15	1.00	3.3%	1.04
	600	16	3.0%	0.94	16	0.94	3.2%	1.01
	1000	16	3.0%	0.94	16	0.94	3.1%	0.98
1024	200	16	1.5%	0.94	11	1.37	2.1%	1.32
	400	16	1.5%	0.94	13	1.16	1.9%	1.20
	600	16	1.5%	0.94	14	1.07	1.8%	1.13
	1000	16	1.5%	0.94	15	1.00	1.7%	1.07

Table 3. 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]		Sequential Decompressor		Sequential Decompressor w/ Scan Inversion	
	Num. Vect.	Total Bits	Num. Vect.	Total Bits	Num. Vect.	Total Bits	Num. Vect.	Total Storage	Num. Vect.	Total Bits	Num. Vect.	Total Bits	Num. Vect.	Total Bits
s13207	233	163,100	273	109,772	236	30,880	274	16,913	272	17,970	266	15,020	266	9,708
s15850	96	58,656	178	32,758	126	26,000	185	14,676	174	15,774	226	16,153	226	10,726
s38417	68	113,152	337	96,269	99	93,466	231	55,848	288	60,684	105	50,365	105	36,864
s38584	110	161,040	239	96,056	136	77,812	220	47,886	215	31,061	192	38,192	192	27,555

must receive enough free-variables every clock cycle to be able to encode each bit-slice of the scan chains. The worst-case most heavily specified bit-slices typically limit the encoding efficiency. As can be seen in Table 1, the encoding efficiency is less than 0.5 without scan inversion. However, with scan inversion, the most heavily specified bit slices can be solved with fewer free-variables per clock cycle thereby allowing the number of tester channels to be reduced substantially. The fact that the encoding efficiency can be increased to around 0.9 with a combinational decompressor is a surprising result. This level of encoding efficiency is on the order of what is typically achieved with sequential decompressors, however, in this case no LFSR is needed thereby reducing the hardware overhead. Combinational decompressors are attractive because they are very simple requiring low hardware overhead. The only drawback to using them has been the substantially reduced encoding efficiency compared with sequential decompressors, however, these results indicate that with scan inversion the gap in encoding efficiency between combinational and sequential decompressors can be significantly reduced. The results for keeping the number of channels at 32 and increasing the percentage of specified bits per test cube did not provide as high of encoding efficiency as reducing the tester channels did.

We also did the same experiments using a continuous-flow sequential linear decompressor as shown in Fig. 9. The results are shown in Table 2. In this case, the number of channels from the tester for the normal scan chain without inversion was 16 and a 64 bit LFSR was used. Because the sequential linear decompressor is very efficient to begin with, the number of channels that can be reduced using scan inversion is not as spectacular as for

combinational decompressors. Nonetheless, a significant increase in the encoding efficiency (above 1 in many cases) can be achieved especially for 1024 scan chains where the percentage of specified bits per test cube is less. Note also that the results for keeping the number of channels at 16 and increasing the percentage of specified bits per test cube were actually better than reducing the tester channels. This is the opposite of what happened for combinational decompressors where reducing channels achieved higher encoding efficiency.

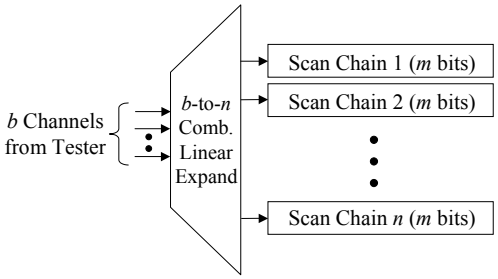

Figure 8. Combinational Linear Decompressor

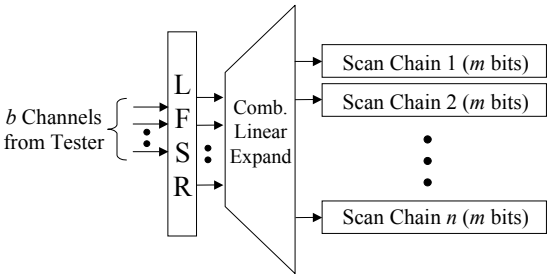

Figure 9. Continuous-Flow Sequential Linear Decompressor

Experiments were also performed for the largest ISCAS 89 benchmark circuits to compare the results with previously published techniques. A scan architecture with 64 scan chains was assumed and a sequential decompressor like the one in Fig. 9 was used with a 64 bit LFSR. Table 3 shows the results obtained by the proposed method along with other test vector compression schemes. The first column shows the circuit name and the next two columns are the number of vectors and the total test size of the dynamically compacted test cubes generated by MINTEST [Hamzaoglu 98]. The next few columns show the number of test vectors and the compressed test size for the Illinois scan architecture [Hamzaoglu 99], frequency directed runlength codes [Chandra 01], mutation encoding [Reda 02] and seed overlapping [Rao 03]. As can be seen from the results, scan inversion substantially reduces the tester storage requirements compared with using the sequential decompressor without scan inversion. The results indicate that the tester storage requirements are less than those in recently published test vector compression papers.

7. Conclusions

A method for improving the encoding efficiency of a linear decompressor using scan inversion was proposed. A systematic procedure based on linear algebra was described for selecting the set of inverted scan cells. Experimental results show that scan inversion can dramatically improve the encoding efficiency of combinational linear decompressors bringing it close to that of sequential decompressors. Scan inversion can also significantly improve the encoding efficiency for sequential linear decompressors. Scan inversion can be implemented with no hardware overhead.

Acknowledgements

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

References

[Bayraktaroglu 01] Bayraktaroglu, I., and A. Orailoglu, "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. Orailoglu, "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.

[Brglez 89] Brglez, F., D. Bryan, and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," *Proc. of International Symposium on Circuits and Systems*, pp. 1929-1934, 1989.

[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.

[Chandramouli 03] Chandramouli, M., "How to Implement Deterministic Logic Built-In Self-Test (BIST)," *Complier: A Monthly Magazine for Technologists Worldwide*, Synopsys, Jan. 2003.

[Chatterjee 95] Chatterjee, M., and D. K. Pradhan, "A New Pattern Biasing Technique for BIST," *Proc. of VLSI Test Symposium*, pp. 417-425, 1995.

[Cullen 97] Cullen, C. G., *Linear Algebra with Applications*, Addison-Wesley, ISBN 0-673-99386-8, 1997.

[Hamzaoglu 98] Hamzaoglu, I., and J. H. Patel, "Test Set Compaction Algorithms for Combinational Circuits," *Proc. of International Conference 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 International 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.

[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 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 00] Khoche, A., and J. Rivoir, "I/O Bandwidth Bottleneck for Test: Is it Real ?," *Proc. of International Workshop on Test Resource Partitioning*, 2000.

[Kiefer 97] Kiefer, G., and H.-J. Wunderlich, "Using BIST Control for Pattern Generation," *Proc. of International Test Conference*, pp. 347-355, 1997.

[Kiefer 98] Kiefer, G., and H.-J. Wunderlich, "Deterministic BIST with Multiple Scan Chains," *Proc. of International Test Conference*, pp. 1057-1064, 1998.

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

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

[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 International Test Conference*, pp. 321-330, 2001.

[Krishna 03] Krishna, C. V., and N. A. Touba, "Adjustable Width Linear Combinational Scan Vector Decompression," *Proc. of International Conference on Computer-Aided Design (ICCAD),* pp. 863-866, 2003.

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

[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.

[Rajski 98] 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 02] Rajski, J., *et al.,* "Embedded Deterministic Test for Low Cost Manufacturing Test," *Proc. of International Test Conference*, 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.

[Sinanoglu 02] Sinanoglu, O., I. Bayraktaroglu, and A. Orailoglu, "Test Power Reduction Through Minimization of Scan Chain Transistions," *Proc. of IEEE VLSI Test Symposium,* pp. 166-171, 2002.

[Touba 95a] Touba, N. A., and E. J. McCluskey, "Transformed Pseudo-Random Patterns for BIST", *Proc. of IEEE VLSI Test Symposium,* pp. 410-416, 1995.

[Touba 95b] Touba, N. A., and E. J. McCluskey, "Synthesis of Mapping Logic for Generating Transformed Pseudo-Random Patterns for BIST", *Proc. of IEEE International Test Conference,* pp. 674-682, 1995.

[Touba 01] Touba, N. A., and E. J. McCluskey, "Bit-Fixing in Pseudo-Random Sequences for Scan BIST," *IEEE Transactions on Computer-Aided Design,* Vol. 20, No. 4, pp. 545-555, Apr. 2001.

[Venkataraman 93] Venkataramann, S., J. Rajski, S. Hellebrand, and S. Tarnick, "An Efficient BIST Scheme Based on Reseeding of Multiple Polynomial Linear Feedback Shift Registers," *Proc. of International Conference on Computer-Aided Design (ICCAD)*, pp. 572-577, 1993.

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

[Wunderlich 96] Wunderlich, H.-J., and G. Kiefer, "Bit-Flipping BIST," *Proc. of International Conference on Computer-Aided Design (ICCAD)*, pp. 337-343, 1996.

[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 Conference*, pp. 186-194, 1996.