Unknown Blocking Scheme for Low Control Data Volume and High Observability

Seongmoon Wang Wenlong Wei Srimat T. Chakradhar
{swang,wwei,chak}@nec-labs.com
NEC Labs. America, Princeton, NJ

Abstract

This paper presents a new blocking logic to block unknowns for temporal compactors. The proposed blocking logic can reduce data volume required to control the blocking logic and also increase the number of scan cells that are observed by the temporal compactors. Control patterns, which describe values required at the control signals of the blocking logic, are compressed by LFSR reseeding. In this paper, the blocking logic gates for some groups of scan chains that do not capture unknowns are bypassed. Since all the scan cells in these scan chain groups are observed without specifying the corresponding bits in control patterns, fewer specified bits are required and more scan cells are observed. The seed size is further reduced by reducing numbers of specified bits in the densely specified control patterns. The proposed method can always achieve the same fault coverage that can be achieved by direct observation of scan chains. Experiments with large industrial designs clearly demonstrate that the proposed method is scalable to large circuits. Hardware overhead for the proposed blocking logic is very low.

1 Introduction

Continuous scaling of semiconductor technology has increased IC chip complexity. As the design complexity increases, test data volume also increases rapidly. Since test data volume is a major factor that determines test cost, several test compression techniques including commercial tools [6, 3] to reduce both volume of input test patterns and output responses have been developed. Spatial compaction [7, 4, 10] reduces response data volume by reducing the number of outputs (typically outputs of scan chains) that are observed by the automatic test equipment (ATE). Another approach, temporal compaction, reduces response data volume by compressing a long sequence of responses into a single signature, which is smaller than the size of even a single output response. Hence using a temporal compactor such as multiple input signature register (MISR) can drastically reduce response data volume.

The presence of unknown values (*unknowns* for short) in output responses of scan test patterns creates a lot of complications for test data compression. Especially, entrance of any unknown into a temporal compactor can be catastrophic since it corrupts the signature of output responses over the entire period of testing time. Unknowns can occur due to the presence of non-scan flip-flops, embedded memories, or tristate buffers. Limitation in accuracy of simulation can also produce unknowns. To prevent corrupting the signature, every unknown that appears at outputs must be blocked before it enters the temporal compactor. Techniques to block unknowns for temporal compaction are proposed in [1, 5, 6, 9, 11]. Since blocking unknowns for temporal compaction requires control data that also contribute to overall test data volume, it is important to reduce control data volume.

A selective compactor scheme where only one scan chain output is observed (not blocked) at any scan shift cycle is proposed in [6]. The enhanced selective compactor proposed in [8] can observe multiple scan chains at a scan shift cycle at the expense of higher area overhead and larger control data volume. The simple channel masking technique is commonly used in Logic Built-in Self-Test (LBIST) [1]. In this scheme, fault effects that are scanned out at scan shift cycles when all scan chains are blocked cannot be observed. An enhanced channel masking scheme presented in [1] improves observability of the simple channel masking scheme.

Naruse et al. [5] propose an unknown blocking scheme that is based on LFSR reseeding [2] for LBIST. In order to reduce the number of seeds that are stored in an on-chip memory, a best feedback polynomial of LFSR is searched from a set of different degrees of polynomials. The critical drawback of this method is its prohibitive run time. Wang et. al. [11] show that run time for computing control data and control data sizes can be significantly reduced by using an efficient algorithm. Another unknown masking technique is proposed in [9]. This technique uses a combinational logic called XML to generate blocking control signals. Since the algorithm used in [9] to minimize hardware overhead for the XML is based on traversing the large fault isolation table, the main drawback of this method is its huge memory space requirement. Since the hardware (XML) for this method is customized for a specific set of test responses, the whole XML should be redesigned for any design change.

All unknown blocking schemes introduced above block not only unknowns but also errors (fault effects) for modeled and unmodeled faults. Consider an unknown blocking scheme based on LFSR reseeding such as [5]. The signal probability of every output of an LFSR is 0.5. In other words, if a long sequence of control patterns is generated, the output of every LFSR stage will be set to 1 (0) in 50 % of clock cycles. Hence, 50 % response data that are scanned out of outputs of scan chains will be blocked, i.e., not be observed. Not observing some scan cells can result in decrease in modeled and/or unmodeled fault coverage.

A large IC chip is comprised of several sub-blocks. Sources of unknowns are typically located in only a few sub-blocks. For example, if a sub-block requires very tight timing, some flip-flops in the sub-block are not scanned so that they function as unknown sources during scan based testing.

978-3-9810801-2-4/DATE07 © 2007 EDAA

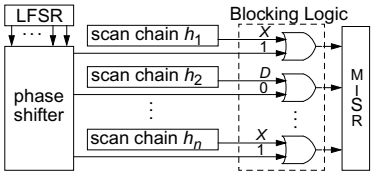

Figure 1. An LFSR Reseeding-based Blocking Logic

Each scan chain is usually routed with flip-flops in the same sub-block since mixing flip-flops in different sub-blocks together into one scan chain makes diagnosis difficult. Hence typically most unknowns are captured in scan chains for a few sub-blocks [7]. This paper proposes a new unknown blocking scheme based on LFSR reseeding that can minimize control data volume and maximize the number of scan cells that are observed, especially for designs where unknowns concentrate in a part of the design.

The rest of this paper is organized as follows. Section 2 introduces the LFSR reseeding based unknown blocking scheme. In Section 3, architecture of the proposed blocking logic is described. Section 4 describes the proposed algorithms that minimize the number of specified bits in control patterns and maximize observability. Experimental results are presented in Section 5. Section 6 gives conclusions.

2 LFSR Reseeding Based Blocking Scheme

Figure 1 describes the LFSR reseeding based unknown blocking scheme used in prior work [5, 11]. Every unknown value that is scanned out of the output of a scan chain (see X's on the outputs of scan chains h_1 and h_n) in a shift cycle must be blocked to prevent it from corrupting the signature by setting the control input of the corresponding blocking logic gate to a 1. On the other hand, errors that need to be observed should propagate to the MISR through the blocking logic gate. For example, in Figure 1, since an error (denoted by D) that needs to be observed is scanned out of the output of h_2 in the current cycle, the control input of the blocking logic gate for h_2 is set to a 0. The LFSR should be loaded with appropriate seeds to block all unknowns and propagate the errors that need to be observed to the MISR. These seeds should be stored in the ATE memory prior to test application along with test data. Hence, if the size of seeds is large, it will increase overall data volume to be stored in the ATE memory. (In this paper, we use a scheme that loads a new seed into the control LFSR for each test pattern.) When test patterns are compressed by LFSR reseeding, normally the size of seeds, i.e., the number of stages of LFSR, is determined by the number of specified bits in the most specified test pattern among all test patterns. If the number of specified bits in the most specified test pattern is S_{max}, the number of stages of LFSR required is given by $S_{max} + M$, where M is a margin to ensure that the equations are solvable. Hence, it is important to minimize the number of specified bits in the most specified pattern.

To minimize volume of control data, control pattern c_i that has minimal number of care bits is computed for every test pattern p_i. Every scan cell that captures an unknown should be assigned 1's in c_i (assume that the blocking logic is comprised of only OR gates). The number of specified bits in each control pattern is minimized by minimizing the number of bits that are specified to 0's to observe errors. To reduce

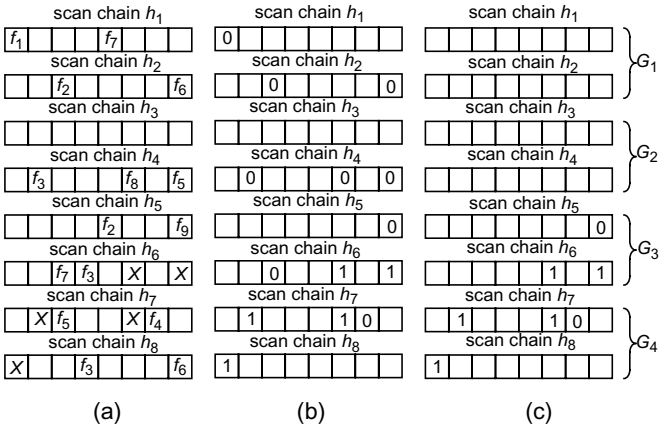

Figure 2. (a) Response Loaded into Scan Cells (b) Corresponding Control Pattern c_i (Prior Work) (c) Control Pattern c_i for the Proposed Blocking Logic

the number of care bits in c_i, we reduce the number of faults that are targeted by test pattern p_i. Note that if a fault f is detected not only by p_i but also by many other test patterns, then even if f is not observed when p_i is applied, there is a high chance that fault f will be detected by other test patterns. Once a set of faults F_i that should be detected by test pattern p_i is determined, then the minimum number of scan cells required to observe all faults in F_i are selected. Fault set F_i is called the *target fault list* of p_i. Typically, a fault is captured into multiple scan cells and some scan cells capture fault effects for multiple faults. Since observing only one fault effect is enough to detect the fault, only one fault effect is selected for observation for every fault in F_i.

Figure 2 (b) gives control pattern c_i obtained by using the procedure described above for the test response shown in Figure 2 (a). The scan cells that capture unknown values are assigned 1's in c_i. Assume that the target fault list of p_i contains faults $f_1, f_2, \ldots, f_7$. Hence c_i is specified to observe fault effects for these 7 faults. Only one fault effect is selected for each fault for observation. For example, although fault effects of f_2 are captured into the third scan cell of scan chain h_2 and fifth scan cell of scan chain h_5, only the third scan cell of scan chain h_2 is selected for observation and assigned a 0 in c_i. c_i requires 14 specified bits.

After a control pattern c_i that has minimal number of specified bits is computed for test pattern p_i, a seed is computed for c_i by using a linear solver. Then the LFSR pattern cr_i that will be generated by the LFSR/phase shifter from the seed for c_i is computed by simulating the LFSR/phase shifter. During the simulation, the LFSR is loaded with the seed for c_i and clocked for l cycles, where l is the number of scan cells in the longest scan chain.

3 Architecture of the Proposed Method

Figure 3 depicts a scan design that employs the proposed unknown blocking technique. There are a few differences between the blocking logic of the proposed technique and that of prior work [5, 11], which is shown in Figure 1. Note that the blocking logic of the proposed method has an extra 2-input AND gate before each OR gate of the blocking logic. One input of each 2-input AND gate is driven by an output

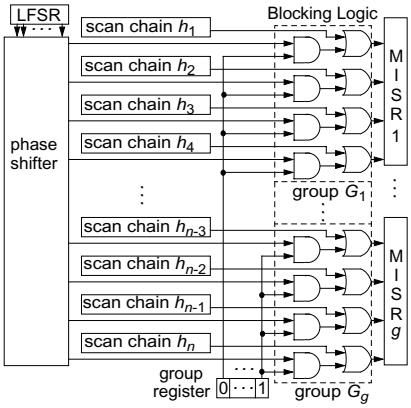

Figure 3. Architecture of the Proposed Method

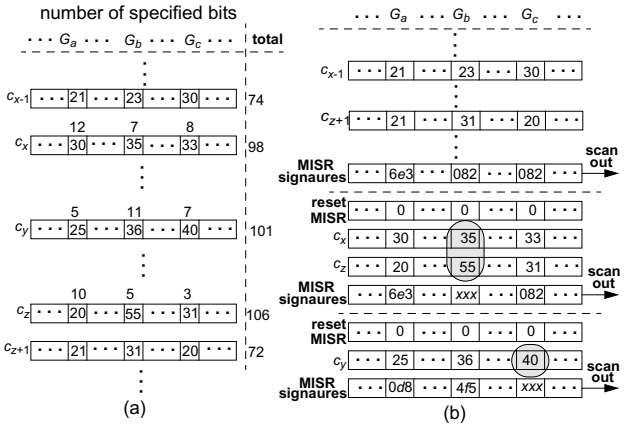

Figure 4. (a) Number of Specified Bits for UCGs (b) Sorted Control Patterns and Signatures

of the phase shifter and the other input is driven by an output of the group register. The n scan chains are divided into g groups and all the 2-input AND gates that are connected to the outputs of scan chains that belong to the same group are driven by a common output of the group register. Hence if i-th bit of the group register is assigned 0, then the values captured in all scan cells in i-th group enter the corresponding MISR, independent of the output states of the phase shifter. The outputs of scan chains in each group are connected to a separate MISR. Each group may contain a different number of scan chains although in the example shown in Figure 3, every group consists of 4 scan chains. In this paper, all scan chains that can capture unknowns are placed into a few scan chain groups. These scan chain groups, which are assigned 1's in the group register such as G_g of Figure 3, are called *unknown capturing scan chain groups* or *UCGs*. On the other hand, scan chain groups that capture no unknowns are called *unknown free scan chain groups* or *UFGs* and assigned 0's in the group register such as G_1.

4 Control Patterns for the Proposed Method

The proposed blocking logic can significantly reduce data volume for control patterns and improve observability over [5, 11].

4.1 Reducing 0's in Control Patterns

The control pattern for prior work [5, 11] shown in Figure 2 (b) requires 14 specified bits. In the following, we show that the number of 0's in the control pattern for the same test response can be significantly reduced by using the proposed blocking logic. Consider computing a control pattern c_i for the proposed blocking logic (see Figure 2 (c)) for the test response shown in Figure 2 (a). Assume that scan chains are clustered into 4 groups, $G_1, \ldots, G_4$. Only the scan cells in groups G_3 and G_4 capture unknowns and no other scan cells capture unknowns. Hence G_1 and G_2 are assigned 0's in the group register and all scan cells in these scan chain groups can be observed without specifying any bit of c_i to 0 for them. All the 7 faults except 2 faults f_4 and f_9 in the target fault list of p_i can be detected by observing G_1 and G_2. Although the proposed scheme can reduce the number of 1's too (see the next section), assume that all scan cells that capture unknowns are assigned 1's in the control pattern for now. Hence, 5 bits are assigned 1's to block the unknowns in the control pattern. Only two additional bits need to be specified to 0's to observe fault effects for f_4 and f_9. In consequence, total only 7 bits (5 1's and 2 0's) need to be specified in the control pattern for the proposed blocking logic.

As described above, data for the group register are determined according to scan chains that capture unknowns. A simple method to identify scan cells that can capture unknowns is to simulate the design with a set of random patterns. During the simulation, all scan chains that contain scan cells that capture unknown(s) in the response to any random pattern are identified. These scan chains are grouped together into 2-3 UCGs (UCGs are assigned 1's in the group register). Since the number of groups, i.e., the number of stages of the group register, is small and the group register need not be updated often with different data (in most cases, the group register needs to be loaded only once during the entire test session), data volume for the group register is negligible.

4.2 Reducing Specified Bits in Densely Specified Control Patterns

In LFSR reseeding, the size of seeds (or the number of stages of the LFSR) is normally determined by the number of specified bits in the most specified pattern among all patterns. Hence even if there is only one control pattern that has large number of specified bits and all the other responses have very few specified bits, the blocking logic will require large seed size, which in turn results in large control data volume.

The numbers of specified bits in densely specified control patterns are reduced as follows. Initially, a control pattern that requires minimum number of specified bits is computed for every test pattern. Assume that an LFSR with $S_{max} + M$ stages is used to generate control sequences for the blocking logic. First, we select a set of control patterns that have more than S_{max} specified bits. Let this set be C_d and the set that includes all the other control patterns be C_s. Next, we compute a seed for every control pattern c_i in C_s and calculate the LFSR pattern cr_i from the computed seed by simulating the LFSR operation as described in Section 2. We apply cr_i to the blocking logic and drop all faults that are observed not only from the target fault list of p_i but also from target fault lists of all the other test patterns. The signatures of all MISRs are updated. Hence, many faults will be dropped from target fault lists of test patterns for which control patterns are in C_d when processing all control patterns in C_s is complete. Then

the signatures of all MISRs are reset.

Now we start processing control patterns in C_d. Numbers of specified bits of these control patterns are reduced by unspecifying the bits that are specified for scan cells in one or more UCGs, i.e., by not observing scan cells in one or more UCGs. The UCG(s) for which bits are unspecified in c_i are called the *unobserved UCGs* of c_i. For every control pattern c_i in C_d, we first drop the faults that are captured in scan cells in UFGs, i.e., unknown free scan chain groups (these faults are always observed independent of control sequences generated by the LFSR). The unobserved UCGs are selected to avoid decrease in fault coverage. We first unspecify the specified bits for all the UCGs from c_i. The number of remaining target faults of p_i that are captured in each UCG is counted. Then we select a UCG G_m that captures the largest number of target faults of p_i, mark the UCG, and specify c_i to block all unknowns and observe all target faults of p_i that are captured in G_m. If the number of specified bits of c_i is small enough for the linear solver to find a seed for c_i, then we drop the faults from the target fault list of p_i whose fault effects are captured in the scan cells that are assigned 0's in c_i. Otherwise we unspecify back the bits that are specified for G_m from c_i. Then we select another UCG among the unmarked UCGs that captures the largest number of target faults of p_i. We repeat the procedure described above in this paragraph until all UCGs are marked. When all UCGs are marked, we select the next control pattern and determine the unobserved UCGs for it. This is repeated until a set of unobserved UCGs is determined for every control pattern in C_d.

If in the response to test pattern p_i, an unobserved UCG captures at least one target fault that is captured only in that UCG (the fault is captured in no other scan chain group), then not observing the UCG will make some faults undetected. If this is the case for a control pattern c_i (it does not occur often since many faults in the target faults of p_i have already been dropped and very few faults remain), then c_i is applied twice during test application. In each of the two applications, a different UCG is selected as the unobserved UCG. This guarantees detection of all target faults of p_i.

After unobserved UCGs are determined for every control pattern in C_d, the control patterns in C_d are sorted by their unobserved UCGs such that test patterns for which control patterns have the same unobserved UCGs are consecutively applied during test application. After all test patterns for which control patterns have the same unobserved UCGs are applied, the signatures in all MISRs are scanned out and the MISRs are reset before the next group of test patterns are applied.

Example: Figure 4 (a) shows numbers of specified bits in each control pattern required for the UCGs G_a, G_b, and G_c and the total number of specified bits (the column "total"). For example, control pattern c_{x-1} requires respectively 21, 23 and 30 specified bits for G_a, G_b and G_c and total 74 specified bits. Assume that $S_{max} = 75$ is given for the control LFSR. All control patterns other than c_x, c_y, and c_z require fewer than 75 specified bits. Hence $C_d = \{c_x, c_y, c_z\}$ and $C_s = \{$*all control patterns except* $c_x, c_y,$ and $c_z\}$. An LFSR seed is computed for every control pattern in C_s and the faults that are observed are dropped from target fault lists of all test patterns including test patterns for which control patterns are in C_d. As more faults are dropped, fewer specified bits are required in each control pattern. After all test patterns except p_x, p_y, and p_z for which control patterns are in C_d are applied to the scan chains, the signatures in the MISRs are scanned out for comparisons with good signatures. Then all MISRs are reset.

Now control patterns in C_d are processed to compute seeds for them. Control pattern c_x is taken first from C_d and all specified bits of c_x that were specified for the three UCGs are unspecified. The faults that are captured in UFGs are dropped from the target fault list of p_x. Since G_a captures the largest number of faults, 12 (in Figure 4 (a), the number over the number of specified bits required for each UCG represents the number of target faults captured in the UCG), the bits for G_a are specified to block unknowns and to observe the 12 faults. G_a is marked. Since the number of specified bits of c_x is only 30 (smaller than $S_{max} = 75$), the linear solver finds a seed for c_x. The faults whose fault effects are captured in the scan cells that are assigned 0's in c_x are dropped from the target fault list of p_x. Since G_c captures more faults than G_b, G_c is marked next and the bits for G_c are additionally specified in c_x. Since the number of specified bits of c_x is only 63 (the sum of bits that are specified for G_a and G_c), the linear solver still finds a seed for c_x. The faults whose fault effects are captured in the scan cells that are assigned 0's in c_x are dropped from the target fault list of p_x. Since the only unmarked UCG is G_b, G_b is marked next and the bits for G_b are specified in c_i. Since the number of specified bits of c_x now becomes 98, which is far greater than S_{max}, the linear solver does not find a seed for c_x. The bits that are specified for G_b are unspecified back and G_b is determined as the unobserved UCG of c_x. If there are faults in the target fault list of p_x that can be detected only by observing G_b, then p_x will be applied one more time. In the control pattern for the second application of p_x, a UCG other than G_b is selected as the unobserved UCG. A seed is computed for c_x and observed faults are dropped from all target fault lists.

Since there are no further unmarked UCGs, we take the next control pattern c_y from C_d and process the UCGs in the order of G_b, G_c, and G_a (according to the number of faults captured in each UCG) to determine the unobserved UCGs. Since specifying c_y for G_c after G_b makes the number of specified bits of c_y 76, which is greater than S_{max}, the bits of c_y that are specified for G_c are unspecified back. Specifying bits for the remaining unmarked UCG, G_a, makes the number of specified bits of c_y only 61. Hence G_c is determined to be the unobserved UCG of c_y. Using the same procedure that were used for c_x and c_y, G_b is determined to be the unobserved UCG of c_z.

Control patterns in C_d are now sorted into two different groups. For the first group, which includes c_x and c_z, G_b is not observed and for the second group, which includes only c_y, G_c is not observed. Since they are not observed (compared with good signatures), the signatures for unobserved UCGs are denoted by *xxx* in Figure 4 (b). □

4.3 Improving Observability

Another advantage of the proposed scheme over prior work [5, 11] is better observability. If an LFSR is used to generate control signals for the blocking logic, then on an average only 50% of scan cells are observed. In contrast, in the proposed scheme, all scan cells in the groups that are as-

signed 0's in the group register are observed. For example, in Figure 2 (c), since all scan cells in UFGs, G_1 and G_2, are observed, 75 % scan cells can be observed (assume that approximately 50 % scan cells of G_3 and G_4 are observed).

5 Experimental Results

We conducted experiments with sets of test patterns generated for large ISCAS'89 and ITC'99 benchmark circuits and 6 industrial circuits (circuits D1, D2, D3, D4, D5, and D6). Experimental results are shown in Table 1. We first obtained characteristics of test pattern sets we use in the experiments by directly observing responses without output compaction. The results are reported in the columns under the headings *Direct Observ.*. The column *# Out* gives the number of primary and scan outputs in each circuit. For each circuit, we made three versions of scan designs each of which has a different number of unknown sources. Unknown sources were made by excluding some flip-flops for scan insertion. The columns *X src %* give the percentage of non-scan flip-flops in the scan design. Results for the three versions of scan designs are respectively shown in the columns under the headings *low unknown*, *med unknown*, and *high unknown*. The columns X % give the average number of scan cells that capture unknowns in percentage to the total number of scan cells (the percentage of scan cells that were observed is given by $100 - X\%$). The number of test patterns generated is given in the columns *# pat*. All test patterns were generated by an in-house ATPG tool. Fault coverage achieved by each set of test patterns are shown in the columns *FC %*.

We applied the same sets of test patterns and compressed output responses by MISRs through the proposed blocking logic. The results are shown in the columns under the headings *Proposed*. The columns *obs. cell %* report the average number of scan cells that were observed, i.e., whose response values entered the MISRs. For most designs, over 90 % of scan cells were observed. Recall that if the blocking logic is controlled only by the LFSR/phase shifter like prior work [5, 11], then approximately only 50 % of scan cells can be observed. If there are only a few unknowns, then the number of scan cells that propagate through the proposed blocking logic is very close to that of scan cells that can be observed by direct observation. We routed scan flip-flops into scan chains such that about 30 % scan chains capture unknowns in any test pattern for all designs except D1 where about 15 % scan chains captured unknowns. Scan chains that capture unknowns in any test pattern were grouped into 3 UCGs for every scan design. The columns *# stg* show the number of stages of the control LFSR for the proposed blocking logic while columns *old # stg* show the number of stages of the control LFSR for prior work [5, 11] where the blocking logic is controlled only by the LFSR/phase shifter. The number of stages of LFSR was significantly reduced for every scan design by using the proposed blocking logic. It is notable that the number of stages of LFSR for the high unknown version of b22s was reduced from 117 stages to only 14 stages. The number of densely specified control patterns, i.e., the number of control patterns that are placed in C_d (see Section 4.2), are given in the columns *# den pat*. The numbers shown in parentheses in the same columns give the number of test patterns that were applied twice because the unobserved UCG(s) capture at least one target fault that is not captured in any other group. Note that the number of test patterns that are applied twice is very small for every scan design. Fault coverage of the proposed method is not reported since it is always same as fault coverage that can be achieved by direct observation. Run time clearly demonstrates scalability of the proposed method. Run time is about 1 minute or shorter for all ISCAS and ITC 99 circuits. Due to limited page space, run time is reported only for low unknown versions. Run times for med and high unknown versions are close.

If we ignore storage for signatures (since we store very few signatures for an entire test set, storage requirement for the signatures is negligible), the compression that can be achieved by the proposed method is approximately given by (*the number of scan outputs*) / (*the number of stages of LFSR*). The proposed method achieved 440X compression for the low unknown version of D6. In these experiments, a separate seed was stored for each test pattern. However, if we generate multiple LFSR patterns from a single seed, even larger compression can be achieved. Since control pattern for the proposed blocking are very sparsely specified (have large number of don't cares), it will be easy to find compatible control patterns that can share the same seeds.

Table 2 compares results of the proposed method with results of prior work [5, 9, 11]. Since prior work [5, 9] targets LBIST application, accurate comparison of the proposed method with [9, 5] is limited. Except s5378, area overhead of the proposed method is lower than that of [9]. We computed gate equivalents (GEs) of the proposed blocking logic by using the formula used in [9]. Test responses used by [9] have little more unknowns than those used by the proposed blocking logic. If responses have more unknowns, then area overhead for the proposed blocking logic may increase.

Even if responses used by the proposed method have more unknowns (except s13207) than those used by [5], the number of stages of LFSR for the proposed method is smaller than that of stages of LFSR for [5]. The storage amount required for the proposed method (columns *stor bits*) is significantly smaller than [5] for every circuit. The storage amount required for the proposed method is also significantly (about 40-50%) smaller than that required for [11] for every circuit. Run time of the proposed method is several orders of magnitude shorter than [5] (the time unit for results of the proposed method is second while the time unit for results of [5] is hour). As mentioned above, we stored a separate seed for each test pattern. However, if we generate multiple LFSR patterns from a single seed, then even larger compression can be achieved.

6 Conclusions

This paper presents a new blocking logic to block unknowns for temporal compactors. The proposed blocking logic can reduce data volume required to control the blocking logic and also improve the number of scan cells that are observed by temporal compactors. Control patterns, which describe values required at the control signals of the blocking logic, are compressed by LFSR reseeding. In the proposed method, scan chains are clustered into several groups and the outputs of the scan chains in each group are connected to 2-input AND gates that are controlled by the same

Table 1. Experimental Results

CKT		low unknown									med unknown								high unknown							
		Direct Observ.				Proposed					Direct Observ.				Proposed				Direct Observ.				Proposed			
Name	# Out	X src %	X %	# pat	FC %	obs. cell %	# stg	old stg #	# den pat	time sec	X src %	X %	# pat	FC %	obs. cell %	# stg	old stg #	# den pat	X src %	X %	# pat	FC %	obs. cell %	# stg	old stg #	# den pat
s5378	228	.56	.43	139	98.7	97.9	8	18	7(0)	2.6	.56	1.2	134	92.4	90.1	7	15	0(0)	5.0	5.7	129	79.1	80.2	22	36	4(0)
s9234	250	.88	.56	206	89.3	92.4	8	16	3(1)	6.9	1.8	2.4	183	83.1	88.0	13	21	3(2)	5.3	7.2	177	78.3	83.1	39	47	2(0)
s13207	790	.15	.13	207	98.2	93.1	8	19	3(0)	9.7	.45	1.0	196	95.5	85.2	17	28	2(0)	2.0	3.0	193	88.5	83.3	34	41	3(0)
s15850	684	.34	.34	140	96.3	96.9	13	31	9(0)	9.7	1.8	1.6	80	83.5	88.3	33	52	6(0)	5.0	7.6	83	72.6	82.5	98	122	6(0)
s35932	2048	.17	.16	25	91.3	88.0	42	142	3(3)	11.9	.98	1.0	26	89.3	87.5	54	200	3(2)	5.0	2.0	28	85.6	87.3	76	176	2(2)
s38417	1742	.18	.36	190	98.2	84.9	17	33	6(1)	45.3	.98	2.1	182	95.0	83.9	51	68	12(0)	2.0	5.3	180	88.4	82.4	111	133	6(0)
s38584	1730	.48	.30	194	95.4	88.1	19	40	2(0)	48.9	2.0	1.3	190	93.8	84.4	34	49	13(2)	5.0	2.1	185	87.9	84.0	51	69	6(1)
b17s	1512	.99	.11	539	91.1	85.7	10	41	5(4)	207	2.0	.15	551	90.2	90.4	12	34	17(2)	5.0	.44	551	88.1	85.4	21	43	26(2)
b20s	512	.41	.11	701	96.1	95.4	8	17	19(3)	64.9	1.8	.18	603	94.8	93.6	7	28	11(4)	5.1	.35	658	93.8	94.4	9	20	9(0)
b21s	512	.82	.06	664	96.6	97.2	7	19	3(2)	60.1	1.8	.21	545	94.2	92.2	8	19	0(0)	5.1	.52	255	86.7	90.0	13	36	2(0)
b22s	757	.95	.09	668	94.2	95.1	7	25	10(2)	110	1.9	.25	674	93.1	91.3	10	20	14(0)	5.0	1.2	691	88.1	87.2	14	117	54(1)
D1	796	.39	.003	333	99.5	99.8	7	18	0(0)	19.1	1.9	.01	331	98.0	99.5	7	23	0(0)	5.1	.03	325	95.0	99.5	7	26	0(0)
D2	2455	.46	.13	418	98.8	98.7	93	377	14(9)	1.2K	2.0	.25	400	97.8	94.4	89	356	15(10)	5.0	.52	395	91.8	94.4	144	349	12(6)
D3	6795	.09	1.0	921	97.8	86.6	170	246	54(0)	1.3K	.19	1.6	858	96.8	84.8	211	345	74(5)	.50	4.7	929	84.5	83.5	630	852	77(4)
D4	5014	.09	.12	540	97.1	85.8	30	58	39(3)	699	.49	1.1	533	96.8	84.5	89	130	46(3)	5.0	3.5	521	84.5	83.2	238	314	42(2)
D5	4980	.08	3.2	545	89.3	83.6	182	245	13(12)	5.6K	.49	5.8	551	84.0	82.2	314	401	8(6)	5.0	6.8	551	81.6	81.7	364	422	4(4)
D6	66K	.10	.14	1443	92.3	85.4	149	311	48(32)	15K	1.0	.96	1330	88.9	99.0	838	1135	88(38)	5.0	2.6	1142	82.2	83.3	2090	2313	83(39)

Table 2. Comparisons with Prior Work

CKT	Proposed		[9]		Proposed				[5][1]		[11][2]
Name	X %	area GE	X %	area GE	X %	# stg	stor bits	time sec	stor bits	time hour	stor bits
s5378	5.67	843	7.3	338	.43	8	1112	2.6	3090	3	1904
s13207	0.98	937	1.3	1351	.14	8	1544	11.4	11816	13.2	3150
s15850	1.64	860	2.2	1164	.34	13	1820	9.7	2800	10.9	2907
s38584	2.10	2666	2.2	3286	.30	19	3686	48.9	21042	12.3	6240

[1] $X\% = 0.25$ % for all circuits, [2] $X\% = 0.2$ % for all circuits

control signal before the blocking logic gates. If no scan chains in a group capture unknowns, then the corresponding blocking logic gates are bypassed by the 2-input AND gates. Hence no bits need to be specified in the control pattern for that group. This can significantly reduce the number of 0's that should be specified to propagate errors for observation. Since all the scan cells in the group are observed, bypassing can improve observability. Numbers of specified bits in highly specified test patterns are reduced by not observing one or more scan chain groups. The scan chains groups that are not observed are selected such that no decrease in fault coverage results. Experimental results show that control patterns for the proposed blocking logic require very small number of specified bits. The number of scan cells that are observed by the proposed blocking logic is close to that of scan cells that can be achieved by direct observation even under existence of many unknowns in responses. Run time of the proposed method is several orders of magnitude shorter than that of prior work [5]. Experiments with large industrial designs clearly demonstrate scalability of the proposed method. Since only n 2-input AND gates, where n is the number of scan chains in the design, and a small group register are only additional hardware to the blocking logic of prior work, hardware overhead for the proposed blocking logic is very low.

References

[1] V. Chickermane, B. Foutz, and B. Keller. Channel Masking Synthesis for Efficient On-Chip Test Compression. In *Proceedings of International Test Conference*, pages 452–461, 2004.

[2] B. Könemann. LFSR-Coded Test Patterns for Scan Designs. In *Proceedings European Test Conference*, pages 237–242, 1991.

[3] B. Könemann. A SmartBIST Variant with Guaranteed Encoding. In *Proc. of Asian Test Symposium*, pages 325–330, 2001.

[4] S. Mitra and K. S. Kim. X-Compact: An Efficient Response Compaction Technique for Test Cost Reduction. In *Proceedings International Test Conference*, pages 311–320, 2002.

[5] M. Naruse, I. Pomeranz, S. M. Reddy, and S. Kundu. On-chip Compression of Output Responses with Unknown Values Using LFSR Reseeding. In *Proceedings of International Test Conference*, pages 1060–1068, 2003.

[6] J. Rajski and et al. Embedded deterministic test for low cost manufacturing test. In *Proceedings International Test Conference*, pages 301–310, 2002.

[7] J. Rajski, J. Tyszer, C. Wang, and S. M. Reddy. Convolutional Compaction of Test Responses. In *Proceedings of International Test Conference*, pages 745–754, 2003.

[8] H. Tang, C. Wang, J. Rajski, S. M. Reddy, J. Tyszer, and I. Pomeranz. On Efficient X-Handling Using a Selective Compaction Scheme to Achieve High Test Response Compaction Ratios. In *Proceedings International Conference on VLSI Design*, pages 59–64, 2005.

[9] Y. Tang and et al. X-Masking During Logic BIST and Its Impact on Defect Coverage. In *Proceedings International Test Conference*, pages 442–451, 2004.

[10] C. Wang, S. M. Reddy, I. Pomeranz, J. Rajski, and J. Tyszer. On Compacting Test Response Data Containing Unknown Values. In *Proceedings of IEEE International Conference on Computer-Aided Design*, pages 855–862, 2003.

[11] S. Wang, K. J. Balakrishnan, and S. T. Chakradhar. Efficient Unknown Blocking Using LFSR Reseeding. In *Proceedings Design Automation and Test in Europe Conference and Exhibition*, pages 320–325, 2006.