Center for
Reliable
Computing

TECHNICAL
REPORT

AVOIDING ILLEGAL STATES IN PSEUDORANDOM TESTING OF DIGITAL CIRCUITS

By Ahmad A. Al-Yamani, Subhasish Mitra and Edward J. McCluskey

02-2
(CSL TR # 02-2)

December 2002

Center for Reliable Computing
Gates Bldg. 2A, Room #236
Stanford University
Stanford, California 94305-9020

Abstract:

Many digital circuits have constraints on the logic values a set of signal lines can have. In this paper, we present two new techniques for detecting the illegal combinations of logic values in digital circuits and two techniques for preventing them from damaging the circuit or corrupting test results. Our techniques require minimal change to the design and impose almost negligible delay overhead. Simulation results show that the area overhead of our techniques is less than 1% for applying the techniques with on-chip hardware. The techniques can be applied from an external tester with one bit per pattern overhead. Unlike previous techniques, the fault coverage of the legal test patterns in a given test set is not sacrificed with our techniques. Furthermore, our techniques are applicable during IC production test, BIST, board-level tests and system-level tests.

Funding:

This work was supported in part by King Fahd University of Petroleum and Minerals, by DARPA under contract No. DABT63-97-C-0024 (ROAR project), and by NSF under contract number CSL-FY00-28.

Imprimatur:

AVOIDING ILLEGAL STATES IN PSEUDORANDOM TESTING OF DIGITAL CIRCUITS

By Ahmad A. Al-Yamani, Subhasish Mitra and Edward J. McCluskey

CRC Technical Report No. 02-2
(CSL TR No. 02-2)
December 2002

CENTER FOR RELIABLE COMPUTING

Gates Bldg. 2A, Room #236
Computer System Laboratory
Department of Electrical Engineering
Stanford University
Stanford, California 94305-9020

ABSTRACT

Many digital circuits have constraints on the logic values a set of signal lines can have. In this paper, we present two new techniques for detecting the illegal combinations of logic values in digital circuits and two techniques for preventing them from damaging the circuit or corrupting test results. Our techniques require minimal change to the design and impose almost negligible delay overhead. Simulation results show that the area overhead of our techniques is less than 1% for applying the techniques with on-chip hardware. The techniques can be applied from an external tester with one bit per pattern overhead. Unlike previous techniques, the fault coverage of the legal test patterns in a given test set is not sacrificed with our techniques. Furthermore, our techniques are applicable during IC production test, BIST, board-level tests and system-level tests.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Correct operation of digital circuits is not guaranteed if illegal combinations of logic values appear on some signal lines. Many digital designs contain logic that is controlled by one-out-of-n (one-hot) signals. *One-hot signals* in digital circuits are a set of signal lines of which no more than one signal should be active at a time. An example is an n-to-1 selector implemented with n transmission gates, enabled by different control signals. Another example is a bus controlled by tri-state buffers. Examples of designs with one-hot signals are reported in [Shoji 86, Suzuki 93, Yano 90, Yano 94].

For a tri-state bus, an illegal state occurs when more than one driver is enabled to drive the bus at the same time; this state is known as a *contention state* or a *multiple-hot* state. In case of contention, if the two drivers are sending two different values, the value of the bus is non-deterministic, which means that it may be zero or one depending on the implementation details. This non-determinism may propagate to the output. A more severe effect is that the circuit may be damaged because pull-up and pull-down transistors are both activated. A similar non-determinism can happen if none of the tri-state buffers is enabled to drive the bus in which case the bus will be *floating* (*zero-hot*).

The scan-path method is a widely used design for testability (DFT) technique [McCluskey 86]. The inclusion of on-chip circuitry to provide test vectors or to analyze output responses is called *built-in self test* (*BIST*). One BIST technique is pseudo-random testing using a linear feedback shift register (LFSR) [McCluskey 85].

Illegal states can happen during scan in and out because the patterns are shifted serially through the bistables. This problem may cause the circuit to be damaged. Several solutions are available in the literature for this problem. For example, bistables that may cause illegal states can be controlled with control points [Hetherington 99], removed from the scan chain or bypassed and loaded with a legal value [Raina 00]. The techniques we present in this paper are intended for illegal states that occur during test pattern application rather than during scan in and out.

Illegal states during test pattern application may cause the circuit to be untestable or hard to test and it can also cause it to be damaged. Illegal values during pattern application can be caused by pseudo-random testing. When the circuit is tested using automatic test pattern generation

(ATPG), the one-hot condition can be provided as a constraint to the ATPG tool such that none of the generated patterns results in multiple-hot or zero-hot values in one-hot signals during test pattern application. However, when pseudo-random patterns are applied (externally or internally) or when the ATPG tool does not check for illegal values, some of the patterns generated may cause illegal combinations of values in the one-hot signals.

In this paper, we present two new techniques for detecting illegal states in digital circuits and two new techniques for masking their effects. The first technique is based on fixing the illegal state on the one-hot signals. It causes a small delay overhead and slightly changes the circuit under test (*CUT*). The second technique is based on skipping the illegal pattern. It has no delay overhead and does not alter the CUT. Unlike previous techniques, our techniques have no impact on the fault coverage achieved with the legal patterns of a given test set. Although we will be discussing our techniques in the context of one-hot signals, the techniques are directly applicable with arbitrary constraints on logic values that can appear on a set of signal lines.

In Sec. 2 of this paper, we give an overview of the previous work. We present our techniques in Sec. 3 and discuss the simulation results in Sec. 4. We conclude in Sec. 5.

2. PREVIOUS WORK

One-hot signals are sometimes generated directly from the bistables outputs of a design. In that case, the bistables are designed to hold only one-hot values. Such bistables are called *one-hot bistable*s. Several approaches have been proposed for handling one-hot bistables in a scan path based design. The simplest is to impose special testability constraints so that pass transistor-based selectors are not used to implement multiplexers [Abadir 99].

Another approach is to gate the output of the one-hot bistables during scan with the *scan enable* (*SE*) signal, resulting in a particular one-hot value enforced on the bistable outputs irrespective of their contents. Only one of the bistable outputs should be OR-ed with the SE signal while all the other bistable outputs should be AND-ed with the complement of SE signal [Synopsys 97]. When SE is 1 during scanning, a particular one-hot value is enforced. This technique ensures safety during scan-in and scan-out operations, but multiple-hot or zero-hot values may appear on the one-hot signals if pseudo-random patterns are loaded into the bistables. A similar scheme was used in [Levitt 95]. Figure 1 shows how this scheme is implemented to insure a one-hot value on E_1, E_2, E_3 and E_4 during scan-in and scan-out. A generalization of this approach is to enforce a particular one-hot value on the one-hot signals throughout the test mode of operation by using a special signal. Although this solution avoids illegal states during scan-in and scan-out operations and also when pseudo-random patterns are used to test the circuit, the fault coverage can fall drastically because the logic may not be sufficiently tested since the enforced one-hot value does not change during testing [Hetherington 99]. A variant of this scheme is described in [Raina 00].

Another technique is to use a priority encoder for the one-hot signals. The priority encoder takes n arbitrary inputs and produces n one-hot outputs [Fleming 92, Mitra 97]. The problem with this technique is that it modifies the original design and adds delay overhead equivalent to several levels of logic.

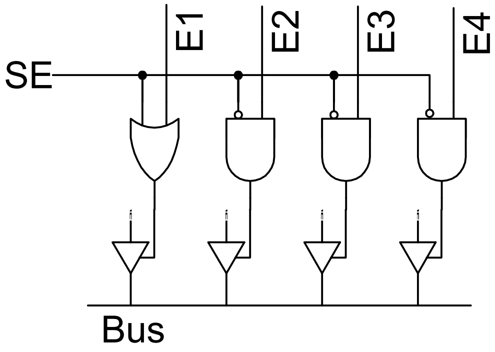

Figure 1: Insuring a one-hot value during scan-in and out.

The L3 latch based Level Sensitive Scan Design (LSSD) technique presented in [Das Gupta 81] can also be used to take care of the one-hot signal problem. In [Das Gupta 81], an L3 latch is used to break the path from an LSSD to a non-LSSD network. A *non-LSSD network* is a part of the circuit that contains non-LSSD latches. This way the non-LSSD logic does not interfere with the testing of the LSSD logic. This scheme does not ensure safety when pseudo-random patterns are used to test the circuit under test (CUT) during test mode because the final scanned-in value may violate the one-hot condition (because it is a random pattern). In [Pateras 95], a technique is presented where the one-hot bistable outputs are encoded and then decoded such that they only hold one-hot signals. This scheme overcomes the problem of pseudorandom pattern application. However, the scheme has limitations due to the fact that it assumes that one-hot signals are produced directly from the one-hot latches. Moreover, the scheme requires additional latches when the number of one-hot signals is not a power of 2. A similar scheme has been reported in [Pyron 00]. The techniques described in [Mitra 97] target the one-hot problem during synthesis and require changes in the logic design of the circuit.

Chiang and Gupta presented a technique for designing a test pattern generator (TPG) to test the board level interconnects via boundary scan. Their technique takes care of the one-hot constraints and the fault coverage constraints in designing the TPG [Chiang 97]. In board level interconnect testing, the TPG directly controls the nets with tri-state controllers. There is no logic between the TPG and the interconnects. This technique is not applicable if the tri-states are on-chip because of the logic between the TPG and the tri-state drivers. Most of the previous work relies on the assumption that the TPG directly controls the tri-state drivers. Our technique eliminates this assumption so it's more widely usable and it reduces the constraints the designer has to consider when designing the system.

The technique we present in this paper does not impose restrictions on the original design. It satisfies the one-hot constraints during test pattern application so it complements the techniques that satisfy the constraints during scan-in and out. It also does not affect the fault coverage for the legal patterns in a given test set. While we describe our technique in the context of one-hot signals, the technique can be directly applied for circuits with arbitrary constraints on logic values that can appear on a set of signal lines.

3. ILLEGAL STATE DETECTION (ISD)

The purpose of an *Illegal State Detection* (*ISD*) circuit is to detect any input pattern of the circuit under test (CUT) that causes illegal values on a set of signals in the circuit. In this section we present our techniques for designing and implementing the ISD circuit. We also discuss two techniques for fixing the illegal state and taking the system back to a legal state. The first technique is based on static fixing which requires extra hardware and adds logic. The advantage of this technique is its simplicity and wide applicability. The second technique is based on skipping the patterns that cause the illegal state. This is done with no additional hardware in the circuit and with no additional delay.

3.1 Illegal state detection by back-tracing

The ISD function should be expressed in terms of the primary inputs and the bistable outputs. One way to perform this expression is to simulate all the possible values on primary inputs and bistable outputs and combine the patterns that cause contention or floating values to form the detection function. The patterns can be reduced with some logic minimization heuristics. This procedure may be prohibitively time-consuming for large circuits. A combinational circuit with 32 inputs will require more than 4 billion patterns to be simulated. A more reasonable way for finding the same detection function is by analyzing the functions of the one-hot signals in terms of the primary inputs and the bistable outputs.

After analyzing the one-hot signals and expressing them in terms of the primary inputs and the bistables, the illegal state detection circuit is implemented so that it produces logic value 1 if the values on the primary inputs and the bistables cause an illegal value on the one-hot signals and it produces a 0 otherwise.

For the purpose of illustration, let us consider a circuit with full-scan. Suppose that there are 4 one-hot signal lines E_1, E_2, E_3, and E_4. As shown in Figure 2, E_1, E_2, E_3, and E_4 are connected to the enable inputs of tri-state gates whose outputs are connected to a common bus.

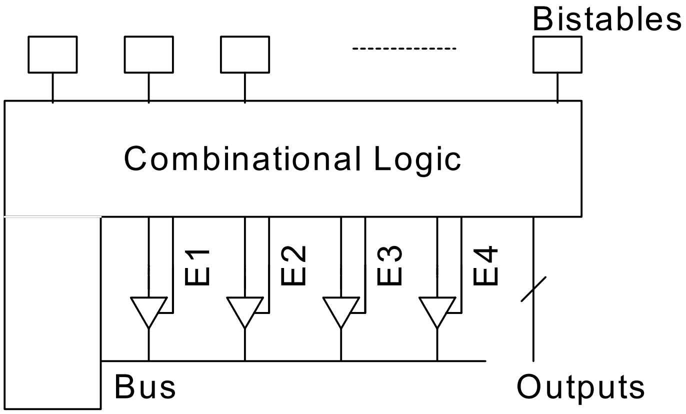

Figure 2: Example of tri-state busses in logic circuits

From the given combinational logic circuit (Figure 2), we can find the Boolean expressions for the logic functions of E_1, E_2, E_3, and E_4 by analyzing those lines and expressing them in terms of the primary inputs and the bistables of the circuit.

Let the Boolean functions corresponding to E_1, E_2, E_3, and E_4 be F_1, F_2, F_3, and F_4, respectively. We form the Boolean function ISD = $(F_1F_2'F_3'F_4' + F_1'F_2F_3'F_4' + F_1'F_2'F_3F_4' + F_1'F_2'F_3'F_4)'$. The ISD function produces a 0 when an input combination guarantees one-hot values on the signal lines E_1, E_2, E_3, and E_4; it produces a 1 when a given input combination produces *non-one-hot* (zero-hot or multiple-hot) values on the signal lines E_1,.., E_4. We can generate the ISD function by synthesizing the ISD circuit expressed in terms of the bistables outputs and the primary inputs. The area overhead of the ISD circuit should be very small if the logic between the tri-state drivers and the scan bistables is shallow. In fact, a large part of the previous work relies on having the bistables directly control the tri-state drivers.

The way we perform the analysis for a one-hot signal is according to the following algorithm:

```
Node Function Analysis (node $n_0$, circuit CUT, function F)
Inputs: $n_0$, CUT;
Outputs: F;            /* an HDL file for $n_0$ in terms of the PIs and FFs */

   1. Let N = {$n_0$};                /* The set of nodes to be analyzed */
   2. Let PI = {all primary inputs of the CUT};
   3. Let FF = {all bistable outputs of the CUT};
   4. For every node $n_i$ in N
         a. Find the gate G whose output is $n_i$ in CUT;
         b. Add G to F;
         c. Let N = N - {$n_i$};
         d. Let the inputs of G be I = {$m_1$, $m_2$, …, $m_k$};
         e. For every node $m_i$ in I
               i. If $m_i \notin$ PI && $m_i \notin$ FF, then
                     Add $m_i$ to N
```

Algorithm 1: Node function analysis algorithm

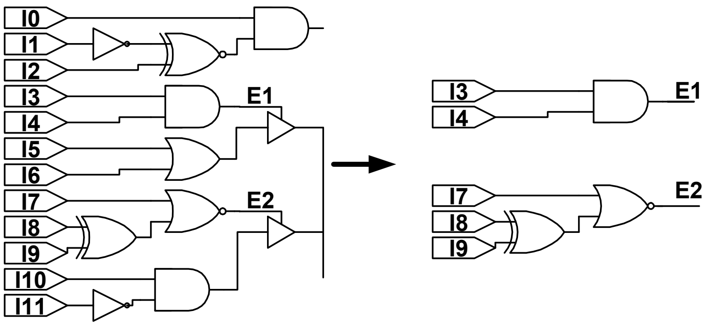

Figure 3: Node analysis for the one-hot signals

Figure 3 shows an example for a combinational circuit with one-hot signals E_1 and E_2. By applying Algorithm 1 to the circuit, we can obtain E_1 and E_2 in terms of the primary inputs as shown in the figure.

Figure 4 shows the ISD circuit in a finite state machine structure of a digital circuit. If the ISD circuit relies only on the bistables that drive the one-hot signals, then there is a chance that the one-hot condition will not be satisfied after the capture cycle, which is the cycle in which the test pattern in the scan chain is applied to the CUT. This is because the shifted pattern may be such that the one-hot condition will be satisfied during pattern application (launch). However, after the capture cycle the contents of the bistables may be such that the one-hot condition may not be satisfied. This can cause damage to the circuit depending on when the scan enable signal is applied next.

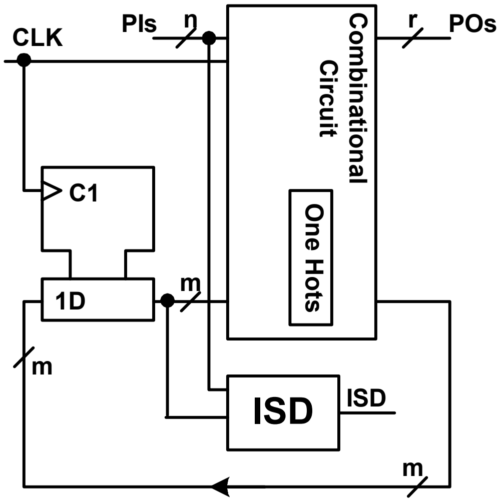

Figure 4: ISD circuit in a finite state machine structure

To avoid the above problem, the ISD circuit should obtain its inputs not only from the bistables that drive the one-hot signals but also from the bistables that control those bistables. In other words, the ISD circuit should rely on the bistables that control the one-hot signals to a depth of two cycles. This way, the ISD circuit is activated one cycle before there is a problem. So, it will be activated when the response of the combinational logic might cause a non-one-hot combination at the one-hot signals. Figure 5 shows the ISD circuit based on two clock cycles protection for the one-hot signals.

The way we implement the ISD circuit is by analyzing the one-hot signals and expressing them in terms of the bistables and primary input. Although the two-cycle based ISD looks like a sequential problem, it's easy to implement it for our purpose because all we need is to continue the analysis to the bistables for two cycles instead of one.

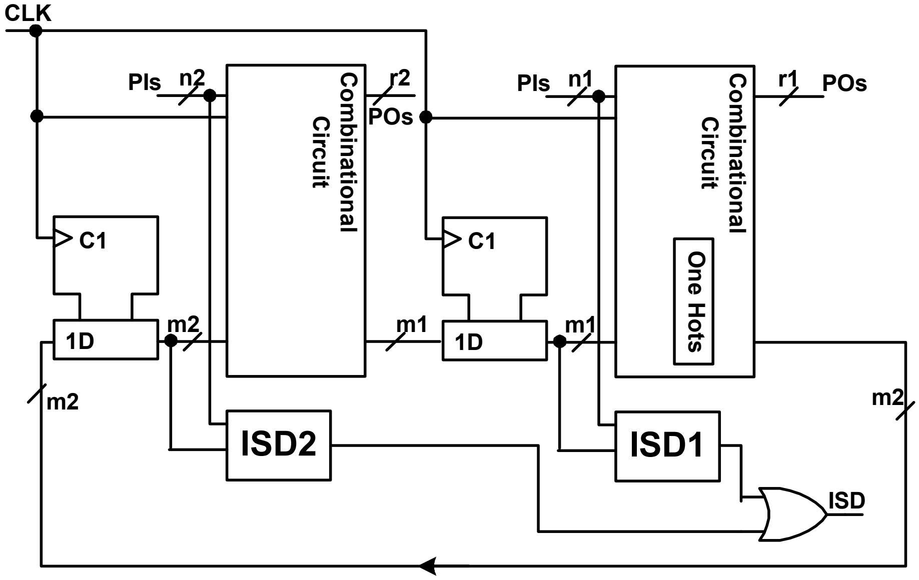

Figure 5: Two-cycle based ISD circuit

The above discussion applies as a full BIST solution where the ISD circuit is synthesized to detect illegal states during the test phase. If the circuit is BISTed with an external tester, there is no need to have the ISD circuit on chip. It can actually be replaced by a single bit per pattern to identify whether the pattern causes an illegal state or not.

3.2 Illegal state detection using BIST pattern counter

In a BIST environment, if analyzing the one-hot signals in terms of the primary inputs and bistables is not possible (e.g. for intellectual property reasons) and the pseudo random pattern generator (PRPG) is known in advance and the seed and test length are also known, the ISD circuit may be designed to be the logical sum of the patterns that cause contention or floating values on the outputs or the bistables. This technique will cause the area of the ISD circuit to be larger than the one obtained by back tracing because some don't cares are not taken into consideration.

The *BIST controller* is the supporting circuitry added to the circuit when it is BIST-tested. It includes the LFSR, the MISR, the counters, the control signal generators, etc. One way to reduce the area of the ISD circuit is to make it depend on the value of the "pattern counter" of the BIST controller. The counter value of those patterns that can cause contention can be used as the minterms for the ISD. If a set of *M* test patterns is applied to the circuit the pattern counter will be of size $\lceil \log_2(M) \rceil$. This can lead to massive reduction in the complexity of the ISD-circuit. Also, in this case we can apply the logic for disabling test application for patterns that cause illegal states. From here on, the ISD circuit based on tracing will be called ISD-Trace and the ISD circuit based on the BIST pattern counter will be called ISD-Counter.

Even with the ISD-Counter circuit, the response of the combinational logic to a legal pattern can be illegal (see the discussion in the previous section). If this will cause damage to the circuit, the ISD circuit should be activated to avoid the damage. One possible solution is to use a second level ISD just like what we did with ISD trace. The 2nd level ISD should be implemented by applying two cycles of logic simulation for the test patterns. The 1st pulse simulation is to get the response of the circuit to the test pattern; the 2nd pulse simulation is to check if the response to the 1st pulse will cause any illegal state. If the 2nd pulse simulation causes an illegal state, the pattern should be included in the ISD circuit implementation even if it is a good pattern.

3.3 Fixing the illegal state

The implementation shown in Figure 6 is one of the many possible ways to force a legal state on the one-hot signal lines in test mode if the applied pattern will cause contention. We will call the added circuitry for forcing the legal state *fixing logic*. Scan enable (*SE*) is a scan signal, which is 1 when patterns are scanned in or out of the scan chains. The test mode (*TM*) signal is 1 in the test mode, i.e. during scan in and out and also during the capture cycle. If the tester (whether external or built-in) has a signal that holds SE at 1 during the capture cycle, that signal can be used to disable capturing the combinational logic outputs of the patterns that cause the illegal state. Otherwise, the TM signal is needed for this fixing logic in order to guarantee the one-hot value during the capture cycle where SE is 0.

If the patterns are generated pseudo-randomly, then the TM signal is needed. If the circuit is BIST-tested, this TM signal is provided by the BIST controller according to some BIST architectures [Dostie 00]. If the circuit is tested using an external tester and the TM signal is not there, then simple logic can be added to implement it. In Figure 6, during the normal mode of operation, the TM and the SE inputs are both 0 and static; hence, the delay overhead introduced is very small.

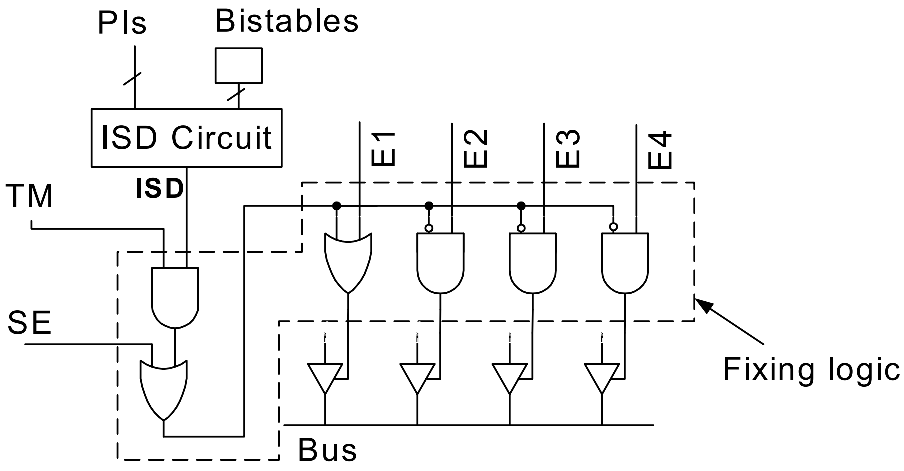

Figure 6: ISD-circuit and its fixing logic control tri-state enables.

The above technique solves the illegal state problem for a set of signals during scan in and out and during the capture cycle because the static fixing logic is active whenever the scan enable is active or the ISD circuit is 1 in test mode. This technique requires much less area and causes less performance overhead than the priority encoder technique

In a BIST environment, the ISD circuit can be part of the BIST controller. Figure 7 shows an overview of the logic BIST controller in a digital circuit. It shows where the ISD circuit and the fixing logic fit in the system level view of a BIST environment.

The ISD circuit can take its inputs from the bistables in the circuit scan chain. If adding fan-outs to the scan chain bistables is not desirable, the ISD circuit can have extra bistables to store the contents of the bistables that determine the value of ISD. In a BIST environment, the "bit counter" is used to count the bits shifted into the scan chain. The value of the bit counter can be used to identify the bits that will go into the bistables that control the one-hot signals. These bits should be stored in the extra bistables to determine the value of ISD. So, the ISD circuit bistables can take their inputs from the pseudo random pattern generator (PRPG) and the bit counter through simple control logic. The above technique is generally applicable and is guaranteed to solve the illegal state problem during test application

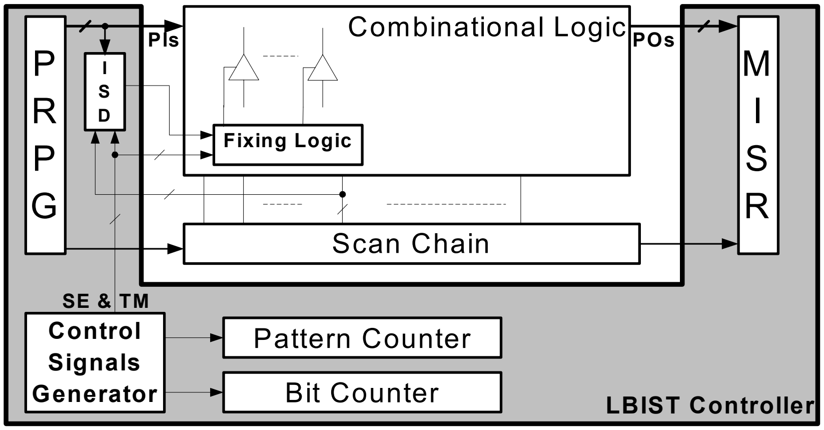

Figure 7: ISD circuit and fixing logic in a system level view of a BIST environment

3.4 Skipping the illegal state

A small logic circuit can be added to the BIST controller such that the patterns that cause the illegal states are not applied to the circuit. In this case, there is no need for any intrusion and no delay overhead, not even the negligible overhead, is caused by the ISD circuit.

Normally, the test patterns are scanned into the bistables with value 1 on the scan enable (SE) signal. Once the pattern is shifted in, SE is turned off for one clock cycle such that the scan chain stores the results of the combinational circuitry. Then SE is turned on again for these results to be shifted out. The logic added to the BIST controller should keep the SE signal 1 during the test application cycle in case the pattern will cause an illegal state. This way the pattern is not applied to the circuit and no corrupted output is read out.

Consider a BIST controller that uses the bit counter to count the bits shifted into the scan chain. Assume that for every scan-in and scan-out sequence, the bit counter is loaded with the count of bistables in the scan chain. It gets decremented every cycle. SE is kept 1 until the bit counter reaches zero. To avoid applying the pattern that causes an illegal state, we need to set SE to one if (1) ISD is 1 and (2) the bit counter is 0. Figure 8 shows an example for the logic needed for this purpose. As shown in the figure, the overhead is negligible and this logic can be part of the BIST controller.

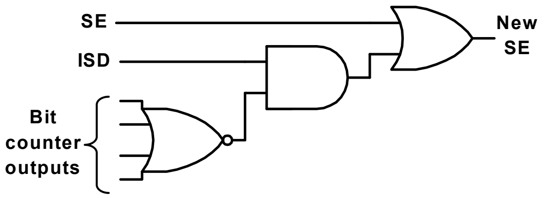

Figure 8: Circuitry for skipping the illegal state

During scan-in and scan out, some of the combinations occurring at the bistables may cause an illegal state on the one-hot signals. If the illegal state will not cause damage to the circuit, no action is needed. However, if, based on the implementation, the illegal state may cause damage, then the one-hot signals should be protected during scan in and scan out using static decoding.

Figure 9 shows a building block diagram for the blocks that interact to perform the illegal state detection and fixing.

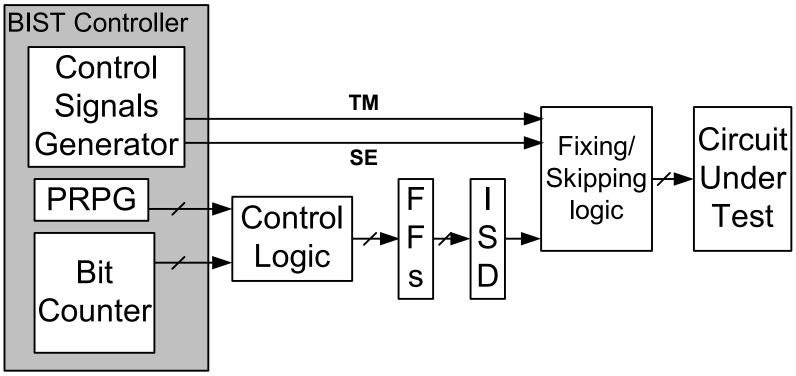

Figure 9: Building blocks for ISD and fixing logic

Another solution for the illegal state problem is to use logic mapping to transform the patterns that cause contention to some ATPG generated patterns that target the faults that are not covered by the random patterns. The ATPG patterns are generated with the one-hot constraint on the one-hot signals. We implemented this technique and observed that it resulted in large area overhead. The area of the resulting circuit was sometimes comparable to and sometimes larger than the area of the original circuit. The reason for this is the fact that by applying purely random patterns to the circuit, and if the patterns are equally probable, the probability of having a legal state on n one-hot signals is $n/2^n$ and the probability of having an illegal state is $(2^n\text{-}n)/n$. If the circuit has two one-hot lines, the probability of an illegal state is 0.5. If it has four one-hot lines, it is 0.75 and so on. This means that most of the patterns need to be mapped and that's why the area of the mapping logic is high. If we use a smaller set of mapping patterns then the mapping area will be smaller but the improvement in the coverage will be sacrificed.

4. Simulation Results

We performed our experiments on I992, which is an industrial design from ITC99 benchmark suite [ITC 99]. I992 is the only circuit with tri-state drivers in the ITC benchmark suite. The other benchmark suites (e.g. ISCAS and MCNC) don't have any circuits with tri-state drivers.

4.1 I992 Circuit

I992 is an industrial circuit that has many one-hot signals. This benchmark has 6 busses controlled by tri-state drivers. Figure 10 shows a top view of the busses controlled by tri-state drivers in the circuit.

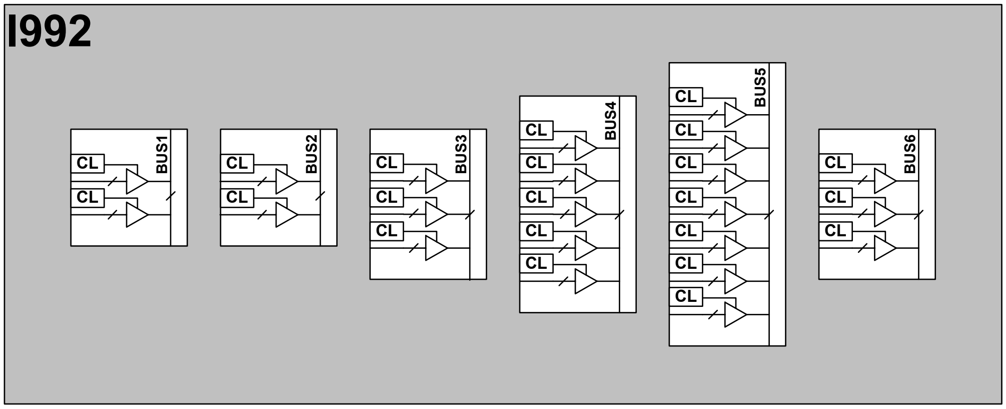

Figure 10: Top view of I992 with its tri-state drivers

I992 is a pipeline ASIC of approximately 20,000 gates. It's the largest benchmark we had with tri-state drivers. The design uses a four-phase clock. There are no internal memories. There are 6 internal busses with tri-state drivers in the design.

Table 1 shows the main features of the I992 circuit. It shows the primary inputs, primary outputs, inouts, the number of bistables, the number of busses driven by tri-state drivers and the cell area. The cell area is given in LSI cell area units where an inverter is equivalent to 1 unit. The library used for technology mapping is LSI Logic G10p library [LSI 96].

Table 1: I992 Characteristics

PIs	POs	InOuts	Bistables	Busses	Area
31	88	176	508	6	29817

We traced back the tri-state enables to the bistables and primary inputs. Then we synthesized the ISD circuits for all the busses. Table 2 shows the details of all the tri-state busses in the circuit. It also shows the areas of the ISD circuits and their overhead relative to the area of the I992 circuit. The total area overhead of the 6 ISD circuits of I992 is only 0.892% of the total area of the circuit. This is an extremely low area overhead for our ISD circuits that will completely eliminate the contention problem in the circuit with a full BIST solution. As explained earlier, if the circuit will be tested with an external tester, only one bit per pattern is required to identify illegal patterns.

Table 2: I992 ISD Circuits' Area Overhead

Busses	Bus width	Tri-state drivers	ISD circuit area	%Area overhead
Bus 1	17	2	10	0.034
Bus 2	17	2	10	0.034
Bus 3	6	3	40	0.134
Bus 4	6	5	82	0.275
Bus 5	6	7	93	0.312
Bus 6	6	3	31	0.104

5. Conclusions

The ISD technique is a very low area and delay overhead technique for fixing the illegal states that can occur in pseudo-random testing of digital circuits. The ISD technique can be used not only during IC production test, but also during board-level or system-level tests when arbitrary test sequences may be applied. Our technique guarantees correct operation when any vectors are applied.

The major distinction between the ISD technique and previous work is that ISD technique is more generally applicable and it does not compromise the fault coverage achieved with the legal patterns in a given test set. The ISD technique can be implemented without changing the given design and is hence, non-intrusive.

The techniques presented in this paper are applicable to any circuit with constraints on the values a set of nodes can take. Furthermore, they can be combined with any technique for improving fault coverage or reducing test length in cases of pseudo-random testing.

References

[Abadir 99] Abadir, M., and R. Raina, "Design-for-test Methodology for Motorola PowerPC^{TM} Microprocessors," *Proc. International Test Conference*, pp. 810 –819, 1999.

[Abramovici 91] Abramovici, M., J.J. Kulikowski and R.K. Roy, "The Best Flip-Flops to Scan", *Proc. of International Test Conference*, pp. 166-173, Oct. 1991.

[Cheng 90] Cheng, K.T. and V.D. Agrawal, "A Partial Scan Method for Sequential Circuits", *IEEE Transactions on Computers*, pp. 544-548, April 1990.

[Chiang 97] Chiang, C.-H., S. Gupta, "BIST TPGs for Faults in Board Level Interconnect via Boundary Scan," *Proc. VLSI Test Symposium*, pp. 376-382, 1997.

[Das Gupta 81] Das Gupta, S., R. G. Walther, T. W. Williams and E. B. Eichelberger, "An Enhancement to LSSD and some Applications of LSSD in Reliability, Availability and Serviceability," *Proc. International Symposium on Fault-Tolerant Computing*, pp. 32-34, 1981.

[Dostie 00] Dostie, B., *Design for at-speed test, diagnosis and measurement*, Kluwer academic punlishers, Boston, MA, USA, 2000.

[Eichelberger 83] Eichelberger, E. B., and E. Lindbloom, "Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test", *IBM Journal of Research and Development*, Vol. 27, No. 3, pp. 265-272, May 1983.

[Fleming 92] Fleming; Lee O. and Walther; John S., "Local tristate control circuit", US Patent 5136185, Aug 1992.

[Hellebrand 92] Hellebrand, S., S. Tarnick, J. Rajski, and B. Courtois, "Generation of Vector Patterns Through Reseeding of Multiple-Polynomial Linear Feedback Shift Registers", *Proc. of International Test Conference*, pp. 120-129, 1992.

[Hetherington 99] Hetherington, G., *et al*., "Logic BIST for Large Industrial Designs: Real Issues and Case Studies," *Proc. International Test Conference*, pp. 358 –367, 1999.

[ITC 99] http://www.cerc.utexas.edu/itc99-benchmarks/bench.html, Benchmark Suite for International Test Conference 1999.

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

[Lee 90] Lee, D.H., and S.M. Reddy, "On Determining Scan Flip-Flops in Partial Scan Designs", *International Conference on Computer Aided Design Proc.*, pp. 322-325, 1990.

[Levitt 95] Levitt, M., *et. a*l., "Testability, Debuggability, and Manufacturability Features of the UltraSPARC TM –I Microprocessor," *Proc. International Test Conference*, pp. 157-166, 1995.

[LSI 96] *G10-p Cell-Based ASIC Products Databook*, LSI Logic, May 1996.

[McCluskey 85] McCluskey, E.J., "Built-In Self-Test Techniques," *IEEE Design & Test of Computers,* pp. 21-28, Apr. 1985.

[McCluskey 86] McCluskey, E.J., *Logic Design Principle*s *with Emphasis on Testable Semicustom Circuit*s, Prentice-Hall, Englewood Cliffs, NJ, USA, 1986.

[Mitra 97] Mitra, S., L. J. Avra and E. J. McCluskey, "Scan Synthesis for One-hot Signals," *Proc. International Test Conference,* pp. 714-722, 1997.

[Pateras 95] Pateras, S. and M. S. Schmookler, "Avoiding Unknown States when Scanning Mutually Exclusive Latches,*" Proc. International Test Conference,* pp. 311-318, 1995.

[Pyron 99] Pyron, C., *et al*., "DFT Advances in the Motorola's MPC7400, a PowerPC^{TM} G4 Microprocessor," *Proc. International Test Conference*, pp. 137 –146, 1999.

[Raina 00], Raina, R., *et al*., "DFT Advances in Motorola's Next-Generation 74xx PowerPC^{TM} Microprocessor," *Proc. International Test Conf*., pp. 131 –140, 2000.

[Shoji 86] Shoji, M., "Elimination of Process-Dependent Clock Skew in CMOS VLSI," *IEEE Journal of Solid-State Circuit*s, pp. 875-880, 1986.

[Suzuki 93] Suzuki, M., *et al*., "A 1.5ns 32b CMOS ALU in Double Pass-Transistor Logic," *Proc. IEEE International Solid-State Circuits Conference*, 1993.

[Synopsys 97] Scan Synthesis User Guide, Synopsys (1997.08).

[Touba 96] Touba, N.A., and E.J. McCluskey, "Altering a Pseudo-Random Bit Sequence for Scan-Based BIST", *Proc. of International Test Conference*, pp. 167-175, 1996.

[Yano 90] Yano, K., *et al.*, “A 3.8-ns CMOS 16x16-b Multiplier using Complementary Pass-Transistor Logic,” *IEEE Journal of Solid-State Circuit*s, vol. 25, No. 2, pp. 388-395, April 1990.

[Yano 94] Yano, K., Y. Sasaki, K. Rikino and K. Seki, “Lean Integration: Achieving a Quantum Leap in Performance and Cost of Logic LSIs,” *Proc. IEEE Custom Integrated Circuits Conference,* pp. 603-606, 1994.