Testing Digital Circuits with Constraints*

Ahmad A. Al-Yamani[1], Subhasish Mitra[2] and Edward J. McCluskey[1]
[1]*Center for Reliable Computing* [2]*Intel Corporation*
Stanford University, Stanford, CA *Sacramento, CA*
{alyamani, smitra, ejm@crc.stanford.edu}

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 the test results. The hardware overhead of our technique is minimal and imposes negligible delay overhead. Simulation results show that the area overhead of our techniques is less than 1%. 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.

1. Introduction

Correct operation of digital circuits is not guaranteed if illegal combinations of logic values appear on some signal lines. For example, 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. Examples include an n-to-1 selector implemented with n transmission gates, enabled by different signals, and a bus controlled by tri-state buffers.

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 writing two different values, the value of the bus is non-deterministic. 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 [1]. 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) [2].

Illegal states can appear during scan in and out because the patterns are shifted serially through the bistables. Several solutions are available in the literature for this problem. For example, bistables that may cause illegal states can be controlled with control points [3], removed from the scan chain or bypassed [4]. The techniques we present are intended for illegal states that occur during test pattern application rather than during scan in and out.

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 (as in FastScan) such that none of the generated patterns results in multiple-hot or zero-hot values in one-hot signals during test pattern

* This work was done at Stanford-CRC and was supported by King Fahd University of Petroleum and Minerals. It was also supported by DARPA under contract No. DABT63-97-C-0024 (ROAR project). The authors would like to thank Chien-Mo Li and Chao-Wen Tseng for their help.

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 states.

In this paper, we present new techniques for detecting illegal states in digital circuits and masking their effects. 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 discuss 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

If the one-hot signals are generated directly from the bistables, those bistables can be designed to hold only one-hot values. Such bistables are called *one-hot bistable*s. A more general solution is to impose constraints so that pass transistor selectors are not used to implement multiplexers [5]. Another approach is to gate the output of the one-hot signals during scan with the *scan enable* (*SE*) signal, resulting in a particular one-hot value enforced on the one hot signals irrespective of the contents of the bistables. Only one of the signals should be OR-ed with the SE signal while all the other signals should be AND-ed with the complement of the SE signal [6]. When SE is 1 during scanning, a particular one-hot value is enforced. This technique ensures safety (one-hot property) 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 used. A similar scheme was used in [7]. 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 limitation of this scheme is that the SE signal has to be specially routed to avoid delays that may cause none one-hot states to occur before the fixing circuit is activated. 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 [3]. A variant of this scheme is described in [4].

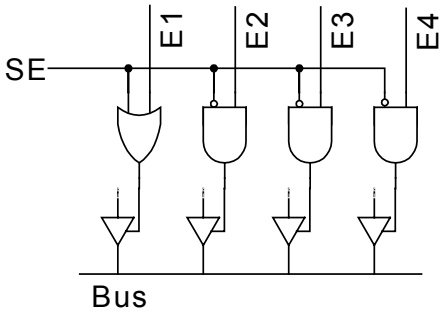

Figure 1: Insuring a one-hot value during scan-in and out.

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 [8, 9]. This technique modifies the original design and adds delay overhead equivalent to several levels of logic.

The L3 latch based Level Sensitive Scan Design (LSSD) technique presented in [10] can also be used to take care of the one-hot signal problem. In [10], 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 [11], 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 bistables. Moreover, the scheme requires additional bistables when the number of one-hot signals is not a power of 2. A similar scheme has been reported in [12]. The techniques described in [9] 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 [13]. 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 applicable.

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, that don't cause an illegal state, 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 whether an input pattern applied to the circuit under test (CUT) 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. However, this technique requires that the test set is known in advance and is not changed later.

3.1 Illegal state detection by back-tracing

The ISD function should be expressed in terms of the primary inputs and the bistable outputs. A reasonable way for finding the ISD function is by extracting 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 current pattern causes an illegal value 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 Fig. 2, $E_1 \ldots E_4$ are connected to the enable inputs of tri-state gates whose outputs are connected to a common bus. From the given combinational logic, we can find the Boolean expressions for the logic functions of $E_1 \ldots E_4$.

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 $E_1 \ldots E_4$; it produces a 1 otherwise. 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 depends on the logic between the tri-state drivers and the scan bistables.

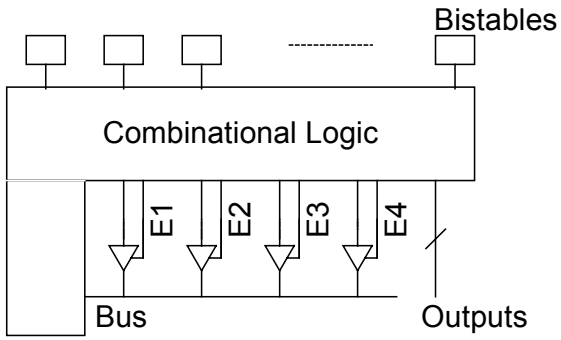

Figure 2: Example of tri-state busses in logic circuits

The way we perform the analysis for a one-hot signal is by extracting the logic cones of such signals. The logic cones can be extracted by tracing the gates whose outputs are connected to the one-hot signals. Then the logic cones are extracted for all inputs signals of all such gates. This recursive process continues until the inputs of the current gates are either primary inputs to the circuit or outputs of bistables in the scan chain.

Figure 3 shows an example for a combinational circuit with one-hot signals E_1 and E_2. By applying the extraction algorithm, we obtain E_1 and E_2 in terms of the primary inputs as shown. Next, the circuits of E_1 and E_2 are pruned and simplified to extract their simplified functions.

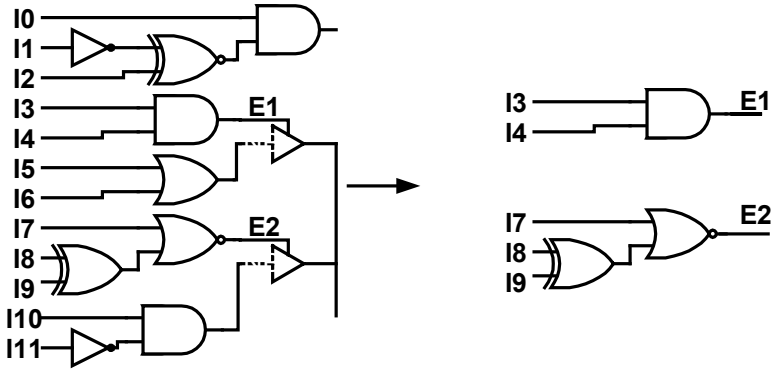

Figure 3: Node analysis for the one-hot signals

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. This is because the shifted pattern may be such that the one-hot condition is satisfied during pattern application (launch) while the contents of the bistables after capture may violate the one-hot condition. This can cause damage to the circuit depending on when the scan enable signal is applied next. This problem never occurs in the normal mode of operation because the circuit operates such that only one signal is activated at time according to the specification.

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 may cause a non-one-hot combination at the one-hot signals.

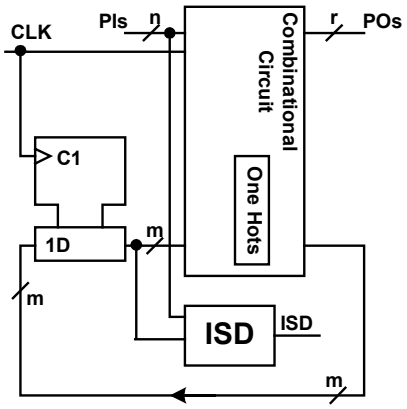

Figure 4: ISD circuit in a finite state machine structure

Figure 5 shows the ISD circuit based on two clock cycles protection for the one-hot signals. The figure only shows a conceptual representation for the bistables that directly control the one-hot signals and the logic around it, and the bistables that control such bistables. The 2nd set of bistables is shown to be considered in designing the ISD circuit such that it detects the illegal state to a depth of 2 cycles.

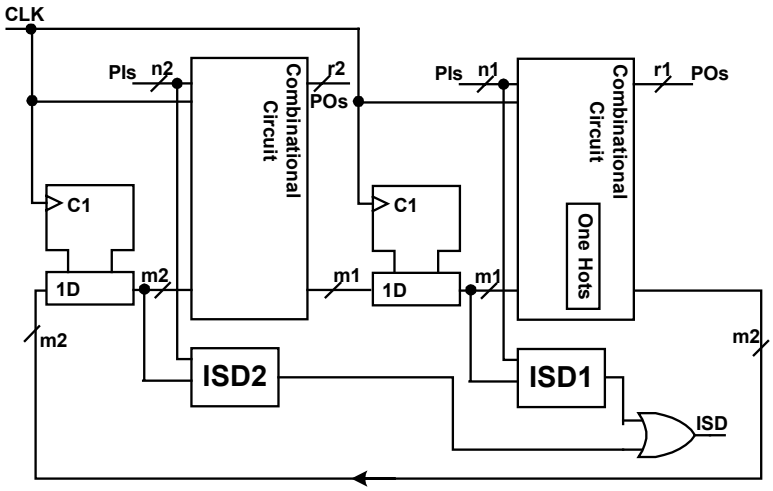

Figure 5: Two-cycle based ISD circuit

We implement the ISD circuit 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.

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 if the test set is known in advance, the ISD circuit may be designed to be the logical sum of the patterns that cause

contention or floating values. Such patterns are found by simulation. This simulation step is not an extra step because it has to be performed anyhow for the signatures to be calculated.

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. 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. If this will cause damage to the circuit, the ISD circuit should be activated to avoid the damage. A second level ISD should be designed 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 cycle is to get the response to the pattern and make sure it's a legal pattern; the 2nd cycle checks if the response to the 1st pulse causes an illegal state.

3.3 Fixing the illegal state

The implementation shown in Figure 6 is one of many possible ways to force a legal state on the one-hot signal lines in test mode. 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 disables the capture cycle, that signal can be used to disable capturing the combinational logic outputs of the illegal patterns. 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.

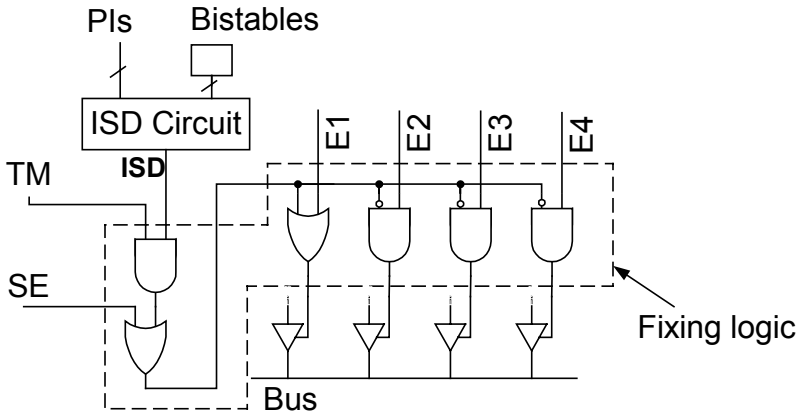

Figure 6: ISD-circuit and its fixing logic control tri-state enables.

If the circuit is BISTed, the TM signal is provided by the BIST controller according to some BIST architectures [14]. During normal operation, the TM and the SE inputs are both 0 and static; hence, the delay overhead introduced is very small because the output of the ISD circuit is blocked by the TM signal. 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.

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.

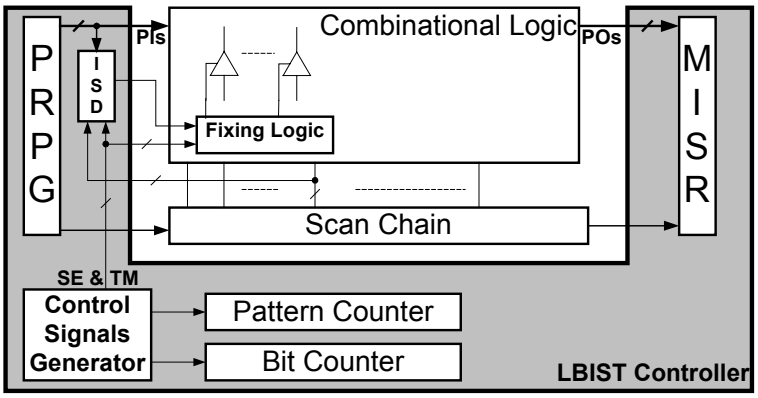

Figure 7: ISD circuit and fixing logic in a system level view of a BIST environment

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. This same pattern skipping technique can be used as a general solution to eliminate the problem of undetermined values (Xs) in pseudo-random testing.

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. This logic can be part of the BIST controller. Also, in case of BIST, if multiple scan chains are used with separate SE signals, then we only need to disable capturing for the chain(s) that cause the illegal state. This way the coverage can be further improved.

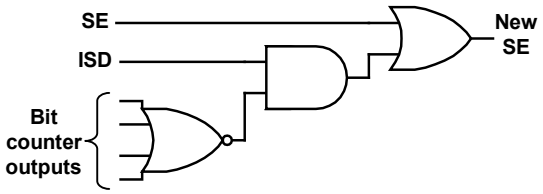

Figure 8: Circuitry for skipping the illegal state

If the design has a separate test clock, the test clock can be ANDed with the complement of ISD such that the pattern is clocked into the scan chain only if it's a legal pattern.

4. Simulation Results

We performed our experiments on I992, which is an industrial pipeline ASIC design from ITC99 benchmark suite [15]. The design has approximately 20,000 gates and four clock domains. There are no internal memories. 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.

I992 is an industrial circuit that has many one-hot signals. This benchmark has 6 internal busses controlled by tri-state drivers. Figure 9 shows a top view of the busses controlled by tri-state drivers in the circuit.

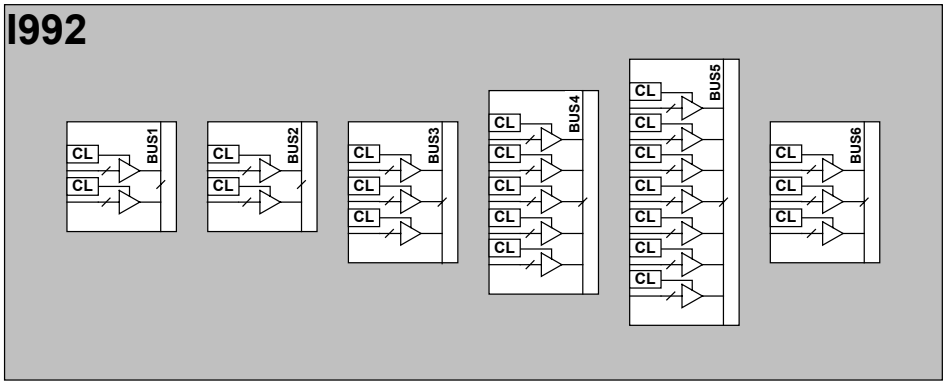

Figure 9: Top view of I992 with its tri-state drivers

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.

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.

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

We also performed our simulations on some circuits from ISCAS 85 and ISCAS 89 benchmarks by randomly adding tri-state drivers at some of their output lines. The results are not shown here due to space limitations. They are discussed in [16]. To briefly summarize the results, although the tri-states were added at the outputs, which means that the logic cones are the largest possible, the area overhead for the ISD-Trace circuit was very low for most of the circuits (1 – 7%) area overhead. There were two cases where the ISD-Trace circuit area was large. For those cases, the ISD-Counter circuit had much lower area overhead. This means that the ISD-Counter method and the ISD-Trace method complement each other.

5. Conclusions

Resolving the illegal states in digital circuits has existed as an obstacle for pseudo-random testing of such circuits for a long time. It has been mostly dealt with using static decoding, which sacrifices the fault coverage and adds an additional level of logic that may not be needed.

The ISD technique is more generally applicable than previous techniques and it does not compromise the fault coverage. The ISD technique can be implemented without changing the given design and is hence, non-intrusive.

The ISD technique is a very low area and delay overhead technique for fixing the illegal states that can occur in pseudo-random testing. 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 are applied. Our technique guarantees correct operation under any patterns.

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

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

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

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

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

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

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

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

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

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

[10] 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.

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

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

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

[14] Dostie, *Design for at-speed test, diagnosis and measurement*, Kluwer publishers, Boston, MA, 2000.

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

[16] Al-Yamani, Ahmad, S. Mitra, and E.J. McCluskey, "Testing Digital Circuits with Constraints", *CRC Technical Report 02-2*, 2002.