IDENTIFICATION OF UNSETTABLE FLIP-FLOPS FOR PARTIAL SCAN AND FASTER ATPG

Ismed Hartanto[†]*, Vamsi Boppana*
Coordinated Science Laboratory
University of Illinois
Urbana, IL 61801
{hartanto,vamsi}@crhc.uiuc.edu

W. Kent Fuchs
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285
kfuchs@ecn.purdue.edu

Abstract

State justification is a time-consuming operation in test generation for sequential circuits. In this paper, we present a technique to rapidly identify state elements (flip-flops) that are either difficult to set or unsettable. This is achieved by performing test generation on certain transformed circuits to identify state elements that are not settable to specific logic values. Two applications that benefit from this identification are sequential circuit test generation and partial scan design. The knowledge of the state space is shown to be useful in creating early backtracks in deterministic test generation. Partial scan selection is also shown to benefit from the knowledge of the difficult-to-set flip-flops. Experiments on the ISCAS89 circuits are presented to show the reduction in time for test generation and the improvements in the testability of the resulting partial scan circuits.

1 Introduction

Sequential circuits are known to be much harder to test than combinational circuits. Test generators often spend significant time on undetectable faults. Fast identification of these faults can lead to a smaller test generation time [1, 2].

A powerful technique for proving the undetectability of faults is the identification of illegal states. Formal methods are known to find illegal states [3, 4]. A significant drawback of many such approaches is the assumption of a fault-free reset state. Recent approaches [1, 5] also use BDD's, but do not require the fault-free reset state assumption. These methods find illegal states by identifying states that do not have an incoming arc in the state transition graph.

Sometimes it is not necessary to find illegal states; finding states that are difficult to traverse with test generation tools can be sufficient to speed up test generation for sequential circuits. In this paper, we present a technique that can rapidly identify states that are difficult to traverse by sequential circuit test generation algorithms. This is achieved by utilizing a deterministic test generator and simple circuit modifications. The main advantage of the proposed method is its ability to adapt to specific test generation tools. Two applications that benefit from the knowledge of the state space are presented along with experimental data on ISCAS89 circuits.

The first application is rapid sequential circuit test generation. During test generation, if the test generator needs to set a value on a state line, a check is done to ensure consistency with the state space knowledge. This early identification of conflicts will be shown to be useful in forcing early backtracks. The next application is the selection of partial scan elements. It will be shown that state space knowledge provides a good indication of the testabilities of the state elements.

2 Three-Value-Unsettable Flip-Flops

Let us consider a sequential machine M with n state elements s_1 *to* s_n. The sequential circuit is assumed to have no global fault-free reset line.

Definition 1 (Illegal State) *A state S_{ill} of a sequential machine M is* **illegal** *when there exists an initial state S_i from which there is no sequence of input vectors that can bring the machine M from S_i to S_{ill}.*

Definition 2 (Unsettable Flip-Flop) *A state element s_u in a sequential machine M is* **unsettable** *to v if there exists an initial state S_i from which there is no sequence of input vectors that can bring the machine M from S_i to a state where s_u is v.*

Definition 2 states a general condition for unsettable flip-flops that is difficult to prove. A more practical condition is defined next.

Definition 3 (Three-Value Illegal State) *A state S_{ill} of a sequential machine M is a* **three-value illegal state** *if there exists no input sequence (evaluated by three-valued logic simulation) that can bring the machine M from the fully unspecified initial state (consisting of all Xs and corresponding to the entire state space) to S_{ill}.*

Definition 4 (Three-Value-Unsettable Flip-Flop) *A state element s_u in a sequential machine M is* **three-value-unsettable** *to v if there exists no input sequence (evaluated by three-valued logic simulation) that can bring the machine M from the fully unspecified initial state (consisting of all Xs and corresponding to the entire state space) to a state where s_u is v.*

This research was supported in part by the Semiconductor Research Corporation (SRC) under grant 95-DP-109, by the Office of Naval Research (ONR) under grant N00014-95-1-1049, and by an equipment grant from Hewlett-Packard.

[†] Presently with Design Technology Center, Hewlett-Packard Co., Palo Alto, CA.

ICCAD '96
1063-6757/96 $5.00 © 1996 IEEE

Definition 4 gives conditions that are practical and relatively easy to find. However, since sequential circuit test generation tools sometimes have difficulties finding *synchronizing sequences* [6], there are cases where a test generator cannot prove the condition in definition 4. The following definitions are given to further simplify those conditions. It should be noted that the following definitions are both tool and experiment specific.

Definition 5 (Difficult-to-Justify State) *A state S_d of a sequential machine M is* **difficult-to-justify** *if a test generator, under a specified time and backtrack limit, does not find an input sequence that can bring the machine M from the fully unspecified initial state (consisting of all Xs and corresponding to the entire state space) to S_d.*

Definition 6 (Difficult-to-Set Flip-Flop) *A state element s_d in a sequential machine M is* **difficult-to-set** *to v if a test generator, under a specified time and backtrack limit, does not find an input sequence that can bring the machine M from the fully unspecified initial state (consisting of all Xs and corresponding to the entire state space) to a state where s_d is v.*

3 Identifying Three-Value-Unsettable FFs

The following procedure finds three-value-unsettable flip-flops that will cause three-value illegal states, and difficult-to-set flip-flops that will cause difficult-to-justify states. First, the circuit netlist is modified by adding primary outputs to each flip-flop. As shown in Figure 1, the modification can be done without affecting the global structure of the circuit; hence, the circuit's topological levelization is not affected. Therefore, the CPU time needed for this step is negligible; furthermore, the circuit modification can be done after the circuit netlist is parsed in memory.

To test whether a state element s_i is three-value-settable to 1, a deterministic test generator is run for the stuck-at-0 fault at the newly created output in the modified circuit. Figure 1 shows such a configuration. It is important that the stuck-at faults are put at the new output and not at the output of the flip-flop; the stuck-at faults should not change the output and next state function of the good circuit.

Since the modified circuit has a primary output at the flip-flops, there will be no propagation problem in the test generation stage of this identification phase. Assuming that the test generator used in this identification phase does not have the over-specification problem [7], if the flip-flop is three-value-settable to 0 in the original circuit, the stuck-at-1 fault at the new output will be detected; and if the flip-flop is three-value-unsettable to 0, the stuck-at-1 fault at the new output will not be detected. Note that three-value-unsettable flip-flops can only be identified using a deterministic test generation tool that does not have the over-specification problem [7]; while in contrast, difficult-to-set flip-flops can be identified using any test generation tool. To identify three-value unsettable flip-flops, the faults at the new POs have to be declared untestable. To identify difficult-to-set flip-flops, aborted faults at the new POs are allowed.

The described method is not restricted to finding one three-value-unsettable flip-flop. The method can be extended to find any combination of flip-flops that causes three-value illegal states.

Since test generation tools for sequential circuits have varying strengths and weaknesses, the set of difficult-to-set flip-flops

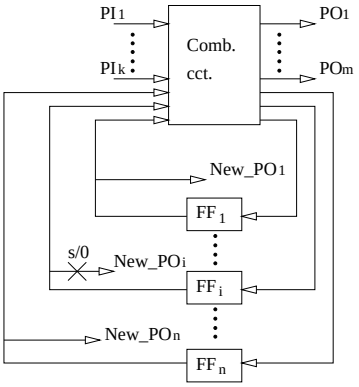

Figure 1: Modified Circuit for Testing whether FF_i is Three-Value-Unsettable to 1

obtained using this method may be different depending on which test generator is used. This fact can be advantageous in one of the applications of difficult-to-set flip-flops, namely, speeding up sequential ATPG.

4 Faster Test Generation Using Difficult-to-Set Flip-Flop Knowledge

One of the problems with deterministic ATPG tools for sequential circuits is the justification of initial states. After the propagation requirements have been met, the starting initial state needs to be justified [8]. There is a possibility that the initial state required for exciting and propagating the fault is not justifiable, and therefore the requirement cannot be satisfied. These unreachable states can be identified quickly by identifying unsettable flip-flops. Early recognition of unreachable states means early backtracking and a smaller search tree to be explored.

In our first phase, the difficult-to-set and three-value unsettable flip-flops are identified using the method in Section 3. The information collected is stored and then used in the second phase.

In the second phase, every time the test generator needs a value at a flip-flop to excite/propagate the fault, the list of difficult-to-set flip-flops is checked. If the required value is known to be unsettable, then the ATPG algorithm has to backtrack. This second phase is similar to HITEC's [8] failed states. However, HITEC's identification of failed states is done for both the good and the faulty circuit and hence has to be cleared before the processing of the next fault. Our identification of difficult-to-set flip-flops is done for the good circuit only and hence is valid for every fault. Furthermore, since we identify difficult-to-set flip-flops one by one in the first phase, the check for the difficult-to-set flip-flops can be done each time there is a requirement for a state assignment, and does not have to wait until forward fault propagation and excitation are complete.

Since the difficult-to-set flip-flops are identified for the good circuit, the information can also be used during the fault propagation and excitation stage of the test generator. Every time the test generator makes an assignment on a state line, the list of difficult-to-set flip-flops is checked. If the assignment is made on a state line that is known to be a difficult-to-set flip-flop, the ATPG has to backtrack.

For each difficult-to-set flip-flop, a flag indicating whether the

faults at the new output are aborted or declared untestable is maintained. This aborted/untestable flag is used in the second phase. If a fault is declared untestable in the second phase, and no aborted difficult-to-set flip-flops are used while performing test generation for this fault, then the fault can be declared untestable. However, if at least one aborted difficult-to-set flip-flop is used, then the fault has to be declared aborted.

The results in Section 6 indicate that the proposed method speeds up a deterministic test generation by up to 50%. It is also interesting to note the improved detection offered by the method.

5 Difficult-to-Set FFs and Partial Scan

For each unsettable flip-flop, there will be at least one undetectable fault. We describe a method that applies the identification of difficult-to-set flip-flops to the selection of scan elements for partial scan. Previous methods for selecting flip-flops are based on one or more of the following three techniques: testability analysis [9–11], structural analysis [10, 12–16] and test generation [12, 17]. The approach proposed in this paper combines the above three techniques. First, a test generation technique is run to find difficult-to-set flip-flops, then, structural analysis is performed to rank flip-flops; and finally, some difficult-to-set flip-flops are scanned with the objective of improving testability.

The first stage of the proposed method is the identification of difficult-to-set flip-flops as described in Section 3. The objective of this stage is to categorize flip-flops as either hard or easy to control. If there are no difficult-to-set flip-flops in the circuit, other measures, such as the number of backtracks performed by the test generator to detect the fault at the new primary output of the modified circuit (Figure 1), can be used. The next step is to rank the controllability of each flip-flop; for this purpose, the S-graph [13] is used.

In the second stage, the S-graph of the circuit is created and each flip-flop is assigned a score of 0. For each difficult-to-set flip-flop s_i, a set of flip-flops that influence s_i is identified. The scores of the flip-flops that influenced s_i are incremented by 1. The objective of this stage is to have a ranking of the controllability of the flip-flops, from hardest to easiest. The ranking of the flip-flops is reflected in the scores of each flip-flop. The flip-flop with the highest score is the hardest to control.

Results in Section 6 show the effectiveness this approach to partial scan.

6 Experimental Results

The following experiments on a SPARCstation 20 illustrate the effectiveness of the proposed techniques in enhancing ATPG and partial scan selection. First, HITEC [8] is utilized to find the difficult-to-set flip-flops. Table 1 shows the number of difficult-to-set flip-flops identified after running HITEC with time limits of 1 and 100 seconds and with respective backtrack limits of 10000 and 1000000. For comparison, the times needed by HITEC to generate tests for these circuits are also given [18]. From Table 1, it is clear that the number of difficult-to-set flip-flops is a good indicator of the circuit testability. If the circuit has difficult-to-set flip-flops, HITEC requires considerable running time. Circuits s641 and s713 seem to be exceptions, since the difficult-to-set flip-flops found in these circuits are easily proven to be unsettable by the difficult-to-set flip-flop identification program, as can be seen from the small execution time. For circuits that are difficult to test (s526, s1423, s5378), the running time for the difficult-to-set flip-flop identification routine is only about 2-4% of the total running time of the original test generator.

Table 1: Difficult-to-Set FFs Identification

Cct.	1 Sec. Time Limit		100 Sec. Time Limit		HITEC [18]
	# Diff. FF	# Time (Sec.)	# Diff. FF	# Time (Sec.)	(Sec.)
s382	10	11.6	1	135.5	10980.0
s400	10	11.8	1	138.0	8316.0
s420	32	22.9	32	388.2	–
s444	10	12.1	1	171.6	10224.0
s526	13	14.5	10	1013.5	38520.0
s641	4	0.8	4	0.8	6.4
s713	4	0.8	4	0.8	9.9
s820	0	0.5	0	0.5	360.6
s832	0	0.5	0	0.4	523.2
s1423	38	45.1	35	3603.3	99000.0
s5378	54	256.4	51	3613.4	130680.0

Table 2 shows the results of deterministic ATPG using difficult-to-set flip-flop identification. We implemented our approach on top of the HITEC [8] automatic test generation tool. The selected circuits are those from Table 1 that have at least one difficult-to-set flip-flop in the 100 seconds time limit column. For circuits that are hard to test (s526, s1423, s5378), the execution times for the proposed method are about 30-52% of the running time for the original ATPG. In addition to reduced running time, the proposed method also provides additional detections; for example, the number of detected faults for s5378 is increased from 3238 to 3312. These additional detections are direct consequences of the use of difficult-to-set flip-flop information. Using that information, early backtracking and hence a smaller search tree is achieved.

Table 2: ATPG Using Difficult-to-Set FFs Information

Cct.	ATPG w/ Difficult-to-Set FFs			HITEC [18]	
	# Diff FF	Time (Sec.)	# Det. Faults	Time (Sec.)	# Det. Faults
s382	1	6805.5	306	10980.0	301
s400	1	5237.8	346	8316.0	342
s420	32	1405.6	32	–	–
s444	1	5196.9	383	10224.0	378
s526	10	16513.0	353	38520.0	346
s641	4	10.0	404	6.4	404
s713	4	14.2	476	9.9	476
s1423	35	52041.8	891	99000.0	776
s5378	51	39541.3	3312	130680.0	3238

Table 3 shows the results of partial scan elements selection using difficult-to-set flip-flops. For comparison, the results of fault-oriented OPUS [12] are also listed. We compare our approach with fault-oriented OPUS since our approach is a combination of testability analysis, structural analysis and test generation tech-

niques, and fault-oriented OPUS combines structural and test generation techniques and can be run on large circuits. The number of scanned flip-flops is limited to 10%. HITEC [8] is run for the scanned circuit with a time limit of 1 second and a backtrack limit of 10000. Circuits selected for this experiment are the circuits that have more than 1 difficult-to-set flip-flop when a 100-second time limit is used (column 4 of Table 1). The proposed method has better results for these selected circuits.

Table 3: Partial Scan using Difficult-to-Set FFs

Cct.	# FF Scan.	OPUS Fault Oriented [12]			Partial Scan w/ Difficult-to-Set FF's		
		# Det.	# Abt.	Time (Sec.)	# Det.	# Abt.	Time (Sec.)
s382	3	373	22	39.0	376	17	29.3
s400	3	391	25	37.5	393	20	35.5
s420	2	43	308	348.6	68	290	318.3
s444	3	394	65	108.5	418	37	52.1
s526	3	283	258	266.7	301	235	256.5
s641	2	439	0	7.2	439	0	7.1
s713	2	511	2	9.3	511	2	9.2
s1423	8	969	533	625.9	979	528	639.4
s5378	18	4276	199	308.1	4354	154	263.6

7 Conclusions

This paper demonstrates that identification of illegal states is not always necessary; finding states that are difficult to traverse with test generators may be enough. A method for the identification of difficult-to-justify states through difficult-to-set flip-flops is presented. The method starts by adding primary outputs to each flip-flop; then, test pattern generation is performed on stuck-at-1 and stuck-at-0 faults at these new primary outputs. Two applications that utilize the difficult-to-set flip-flops information are presented. The first application demonstrates the potential usefulness of the identification of difficult-to-set flip-flops in forcing early backtracks in test generation. The second application, selection of partial scan elements, provides an alternative way of selecting flip-flops for partial scan circuits. This second application shows that there is a strong relation between testability of a circuit and difficult-to-set flip-flops.

References

[1] D. E. Long, M. A. Iyer, and M. Abramovici, "Identifying Sequentially Untestable Faults Using Illegal States," in *Proc. of the VLSI Test Symposium*, pp. 4–11, Apr. 1995.

[2] M. A. Iyer and M. Abramovici, "Sequentially Untestable Faults Identified Without Search," in *Proc. of the Intl. Test Conf.*, pp. 259–266, 1994.

[3] O. Coudert and J. C. Madre, "Symbolic Computation of the Valid States of a Sequential Machine: Algorithms and Discussion," in *Proc. of the Intl. Workshop on Formal Methods in VLSI Design*, Jan. 1991.

[4] H. Touati, H. Savoj, B. Lin, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Implicit State Enumeration of Finite State Machines Using BDD's," in *Proc. of the Intl. Conf. on Computer-Aided Design*, pp. 130–133, Nov. 1990.

[5] C. Pixley, *A Computational Theory and Implementation of Sequential Hardware Equivalence.* 1990. E. M. Clarke and R. P. Kurshan, editors.

[6] Z. Kohavi, *Switching and Finite Automata Theory.* McGraw-Hill, NewYork, 1978.

[7] K.-T. Cheng and H.-K. T. Ma, "On the Over-Specification Problem in Sequential ATPG Algorithms," *IEEE Trans. on Computer Aided Design*, vol. 12, pp. 1599–1604, Oct. 1993.

[8] T. Niermann and J. H. Patel, "HITEC: A Test Generation Package for Sequential Circuits," in *Proc. of the European Design Automation Conf.*, pp. 214–218, Feb. 1991.

[9] E. Trischler, "Incomplete Scan Path with an Automatic Test Generation Methodology," in *Proc. of the Intl. Test Conf.*, pp. 153–162, Nov. 1980.

[10] V. Chickermane and J. H. Patel, "An Optimization Based Approach to the Partial Scan Design Problem," in *Proc. of the Intl. Test Conf.*, pp. 377–386, Sept. 1990.

[11] P. S. Parikh and M. Abramovici, "Testability-Based Partial Scan Analysis," *Journal of Electronic Testing*, vol. 7, pp. 61–70, Aug./Oct. 1995.

[12] V. Chickermane and J. H. Patel, "A Fault Oriented Partial Scan Design Approach," in *Proc. of the Intl. Conf. on Computer-Aided Design*, pp. 400–403, Nov. 1991.

[13] K.-T. Cheng and V. D. Agrawal, "A Partial Scan Method for Sequential Circuits with Feedback," *IEEE Trans. on Computers*, vol. 39, pp. 544–548, Apr. 1990.

[14] D. H. Lee and S. M. Reddy, "On Determining Scan Flip-flops in Partial Scan Designs," in *Proc. of the Intl. Conf. on Computer-Aided Design*, pp. 322–325, Nov. 1990.

[15] S. T. Chakradhar, A. Balakrishnan, and V. D. Agrawal, "An Exact Algorithm for Selecting Partial Scan Flip-flops," *Journal of Electronic Testing*, vol. 7, pp. 83–94, Aug./Oct. 1995.

[16] T. Orenstein, Z. Kohavi, and I. Pomeranz, "An Optimal Algorithm for Cycle Breaking in Directed Graphs," *Journal of Electronic Testing*, vol. 7, pp. 71–82, Aug./Oct. 1995.

[17] H. K. T. Ma, S. Devadas, A. R. Newton, and A. Sangiovanni-Vincentelli, "An Incomplete Scan Design Approach to Test Generation for Sequential Machines," in *Proc. of the Intl. Test Conf.*, pp. 730–734, Sept. 1988.

[18] E. M. Rudnick and J. H. Patel, "Combining Deterministic and Genetic Approaches for Sequential Circuit Test Generation," in *Proc. of the Design Automation Conf.*, pp. 183–188, June 1995.