Vol 05, Article 10482, November 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

COMPUTING TWO PATTERN TEST CUBES FOR TRANSITION PATH DELAY FAULTS

B.SRENIMA[1], P.SOUNDARYA MALA[2]
[1]STUDENT, [2]ASSOCIATE PROFESSOR, DEPT OF ELECTRONICS AND COMMUNICATION ENGINEERING, GODAVARI INSTITUTE OF ENGINEERING AND TECHNOLOGY, A.P, INDIA
srenima@gmail.com, palivela.soundarya@gmail.com

ABSTRACT

Test information compression are implemented for unspecified tests, Path delay faults are determined by using these unspecified tests or test cubes from their certain specified input values. Compared to other faults path delay faults are very effect on the design, but they have to detect if such values are unspecified. The paper gives the possibility of increasing no .of un specified input values in a test set for path delay faults by un specifying such values in order to make the test set more tractable to test data compression. The results of this paper indicate that some no .of such values exists. The procedure in this paper explains test data compression method and implantation of test for a combinational Circuit.

I. Introduction

Test Data compression methods for full-scan circuits use incompletely-specified tests, or test cubes, to accommodate the constraints of test data decompression logic on the tests applicable to the circuit [1]–[4]. An incompletely-specified test is obtained if test generation for a target fault stops as soon as the fault is detected. However, even in this case, due to the order by which inputs are considered during test generation, a test may contain specified values that are not necessary for the detection of the fault. Due to this possibility, dynamic test compaction procedures include processes that increase the numbers of unspecified values in a test without losing the detection of target faults [5], [6]. Dynamic test compaction procedures use the unspecified values for detecting additional faults by the same test. However, the specification of additional values can be done under the constraints of a test data compression method. The procedures described in [7] and [8] start from a completely specified test set and unspecified as many input values as possible without reducing the number of detected target faults. The unspecified values can then be used for test data compression.
In the procedures described in [5]–[8], the decision to unspecified an input value in a test made based on its effect on the fault coverage. If the fault coverage is reduced, the input retains its specified value. Otherwise, it is unspecified. This approach, where un specifying input values is guided by the fault coverage, is applicable to any fault model (stuck-at faults are considered in [5]–[8]).
When considering path delay faults for the detection of small delay defects, in addition to the fault coverage, another parameter of a test set is related to the lengths of the paths associated with detected path delay faults. In this case, certain specified input values may be needed only for determining the lengths of the paths associated with detected faults. If such input values are unspecified, one or more path delay faults would still be detected for every originally detected fault. Therefore, small delay defects would continue to be detected. However, the detected faults would be associated with shorter paths. The goal of this paper is to explore the possibility of un specifying input values that affect the lengths of the paths associated with detected path delay faults, thus making a test set for path delay faults more amenable to test data compression.

The terminology used in this paper with respect to path delay faults is the following. A small delay defect has an extra delay that is smaller than the clock period. Path delay faults model the case where the accumulation of small delay defects along a path causes the delay of the path to exceed the clock period. A full path starts from an input and ends at an output of the combinational logic of the circuit. A sub path starts from an internal line or an input, and ends at an internal line or an output. A path can be a full path or a sub path. The importance of sub paths to the discussion of path delay faults results from the fact that the percentage of detectable path delay faults can be low [9]–[13]. To allow small delay defects to be detected even when path delay faults are not detected, path delay faults that are associated with full paths as well as path delay faults that are associated with sub paths are considered in [14]–[16].
To define the conditions under which a path delay fault associated with a sub path is considered to be detected, the transition path delay fault model is used in [16]. A test for a transition path delay fault requires the detection of a transition fault on every line along the path associated with the fault. Tests for transition path delay faults are a special type of strong non-robust tests for path delay faults. A strong non-robust test sensitizes the path by assigning non-controlling values to off-path inputs during the second pattern of the test. In addition the test creates a 0 1 or 1 0 a transition on every line of the path corresponding to the transition at the source of the path. To detect a transition path delay fault the test is also required to detect each corresponding transition fault along the path.

2010-2014– IJVES

Vol 05, Article 10482, November 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

This paper considers transition path delay faults associated with full paths and sub paths when determining values that can be unspecified under a two-pattern (broadside) test set for a full-scan circuit. The ability to define the detection conditions of a transition path delay fault based on a set of transition faults that need to be detected facilitates the proposed procedure. Starting from a completely-specified test set, denoted by the procedure focuses on scan-in values that can be unspecified. The procedure consists of two parts, as follows.
The first procedure described in this paper ensures that the transition path delay faults that are detected by will continue to be detected after scan-in values are unspecified under. Let P denote the set of transition path delay faults detected by .This procedure does not allow the detection of any fault from P to be lost when is unspecified. This is similar to requiring that the fault coverage of with respect to transition path delay faults would not decrease. It provides a baseline where as many values as possible are unspecified without reducing the fault coverage. This baseline represents the case where the test generation procedure produces incompletely-specified tests to detect a set of target path delay faults. The main contribution of this paper is in allowing additional values to be unspecified in case the test set cannot be compressed by a selected test data compression method, as discussed next.
To allow additional scan-in values in to be unspecified, the second procedure described in this paper allows the lengths of the paths associated with the transition path delay faults in P to be reduced as T is unspecified. For a fault p_j that is associated with a path of length L_j, it is possible to consider sub paths of lengths $1 <= l <= L_j$. The proposed procedure allows path lengths to be reduced to one. It thus allows the detection of a transition path delay fault P_j to be replaced by the detection of each transition fault included in P_j individually. By unspecifying scan-in values gradually, the procedure ensures that the path lengths in P are reduced gradually.
Test data decompression logic applies completely-specified tests, where the unspecified values of a test cube may be filled randomly or based on other, specified values. Randomly filling the unspecified values of a test set produced by the proposed procedure imitates this.
Due to variations in path lengths associated with detected path delay faults that occur when unspecified values are filled randomly, the paper defines a matching between the transition path delay faults detected by the completely-specified test set T to which the proposed procedure is applied, and the test sets produced by the proposed procedure. The matching provides a more accurate assessment of the effects of unspecified values, and of randomly filling these values, on the path lengths associated with detected faults.
Experimental results indicate that randomly filling the unspecified values of a test set obtained by the proposed procedure increases the path lengths associated with detected faults. For the first few test sets in the series produced by the proposed procedure from a test set T, the path lengths are equal or close to those of T.
The discussion in this paper is independent of any particular test data compression method, and targets only the number of unspecified values in the test set. Additional unspecified values are assumed to improve the ability to compress a test set. It is also possible to use the constraints of a particular test data compression method for guiding the proposed procedure to unspecified certain input values.

II. Transition Path Delay Faults

Path delay faults model small delay defects whose accumulation along a path cause the path to fail. The transition path delay fault model was defined in order to capture the situation where the accumulation of small extra delays is sufficient for causing faulty behavior after a transition is propagated through a sub path. As a result, the model captures both small and large delay defects. This is accomplished as follows.
Similar to a path delay fault, a transition path delay fault is associated with a path and a transition on its first line. When the transition is propagated along the path, it defines corresponding transitions for all the lines of the path. Based on these transitions it is possible to define transition faults along the path. A transition path delay fault is detected when all the single transition faults along the path are detected by the same test.
For the discussion in this paper, a transition path delay fault is denoted by $p_j=f_{j,0}-f_{j,1}-\dots f_{j,L-1}$,where $f_{j,0}, f_{j1}, \dots f_{j,L-1}$ are the transition faults that need to be detected in order to detect p_j. The relationship between the faults is the following.
Let $f_{j,l}$ be $v_{j,l}$ $v'_{j,l}$ transition fault on line $g_{j,l}$. Then $g_{j,o}-g_{j,1} \dots g_{jL-1}$ forms a path .In addition $v_{j,l} = v_{j0}$, if the number of inverters between $g_{j,0}$ *and* $g_{j,l}$ is even, and $v_{j,l}=v'_{j,0}$ if the number of inverters between $g_{j.l}$ *and* $g_{j,0}$ is odd.
If p_j is associated with a full path, a test for p_j is a strong non-robust test for the path delay fault associated with $g_{j,0}-g_{j,1}-\dots g_{j,L-1}$ and the $v_{j,0}$ $v'_{j,o}$ transition on $g_{j,0}$. If $g_{j,0}$ *or* $g_{j,Lj-1}$ is an internal line, the requirement to detect a transition fault on $g_{j,0}$ guarantees that a test for p_j will cause a transition to occur on $g_{j,0}$ even if it is an internal line. The transition is propagated to $g_{j, Lj-1}$ under the strong non-robust propagation conditions. The requirement to detect a transition fault on $g_{j, Lj-1}$ guarantees that a test will propagate fault effects from $g_{j,Lj-1}$ to an output $g_{j,Lj-1}$ if is an internal line.
The ease of dealing with sub paths under the transition path delay fault model, and the uniformity in dealing with full paths and sub paths, is one of the reasons for using the transition path delay fault model in this work.

Vol 05, Article 10482, November 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

The disadvantage of using this model is that fault simulation requires simulation of transition faults under every test, as discussed below. With an appropriate definition for the detection conditions of faults associated with sub paths, the procedures developed in this paper can be applied to other path delay fault models, including ones for which fault simulation requires only logic simulation.

Given a two-pattern test *ti*, to find the transition path delay faults detected by *ti*, the procedure from [16] first simulates all the transition faults under *ti*. Next, the procedure marks all the lines with detected transition faults. It then identifies a set of lines were detected transition path delay faults begin. The set is denoted by *B*.A line b ϵ B is associated with a detected transition fault. In addition it satisfies one of the following conditions, which ensure that a detected transition path delay fault starting at cannot be extended towards the inputs.

1) *b* is an input.

2) If *b* is a gate output, none of the gate inputs is marked as having a detected transition fault.

3) If *b* is a fan-out branch, the fan-out stem of is not marked as having a detected transition fault.

III .COMBINATIONAL BENCHMARK CIRCUIT C432

In this suggested technique we test 36-bit feedback ISCAS-85 C432 27-channel interrupt contraller operator. In this technique we are test any IC are regular indicate tour. This document reveals that on-chip creation of efficient broadside assessments can be done using a simple and set components framework, with some factors that need to be designed to a given circuit, and can accomplish great conversion mistake protection for testable tour. With the suggested on-chip test creation technique, the circuit is used for producing obtainable declares during test program. This relieves the need to estimate obtainable declares off-line.

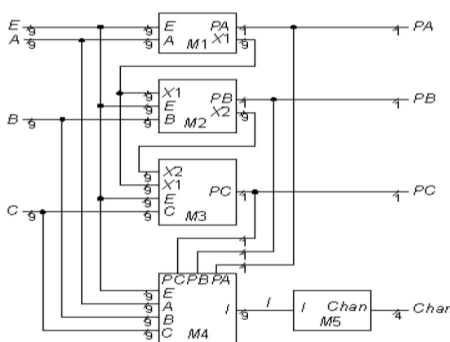


Figure 1 C432 Combinational Circuit

SCANING CIRCUITS

D f\f"s

The D flip-flop is the most common flip-flop in use today. It is better known as information or wait flip-flop (as its outcome Q looks like a wait of feedback D).The Q outcome requires on the condition of the D feedback at this time of a positive advantage at time pin (or adverse advantage if time feedback is effective low). It is known as the D flip-flop for this reason, since the outcome requires the value of the D feedback or information feedback, and setbacks it by one time pattern. The D flip-flop can be considered as a basic storage mobile, zero-order hold, and wait range. Whenever time impulses, the value of Qnext is D and Qprev otherwise.

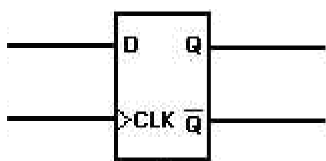


Figure 2 D f\f

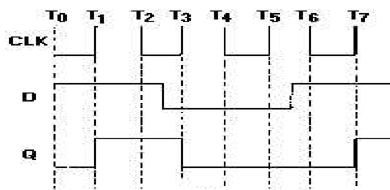


Figure 3Timing diagram of D f\f.

Stuck at „0" Fault

A stuck-at fault is a particular mistake style used by mistake simulators and automatic test pattern generation (ATPG) resources to imitate a production problem within a circuit. Individual alerts and hooks are believed to be trapped at Sensible '1', '0' and 'X'. Furthermore the outcome could be linked with may 0 to style the actions of a faulty circuit that cannot change its outcome not all mistakes can be examined using the stuck-at mistake style. Settlement for fixed risks, namely branching alerts, can provide a circuit untreatable using this style. Also, repetitive tour cannot be examined using this style, since by style there is no modify in any outcome due to only one mistake. In this style, one of the indication collections in a circuit is believed to be trapped at a set reasoning value, regardless of what information are provided to the circuit. Hence, if a circuit has n indication collections, there are possibly 2n stuck-at mistakes described on the circuit, of which some can be considered as being comparative to others. The stuck-at mistake style is may mistake style because no wait information is associated

2010-2014– IJVES

Vol 05, Article 10482, November 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

with the mistake meaning. It is also known as a lasting mistake style because the faulty impact is believed to be long lasting, (e.g. heat range, power source voltage) or on the information principles (high or low volts states) on around indication collections. A style set with 100% stuck-at mistake protection includes assessments to identify every possible stuck-at mistake in a circuit. 100% stuck-at mistake protection does not actually assurance top quality, since mistakes of many other kinds such as connecting mistakes, reveals mistakes, and conversion (delay) mistakes often happen. Fault designs are required to evaluate caused by quality n Sensible Fault : Reflection of the impact of the physical mistakes on the function of the system Only the reasoning operate is usually regarded (not timing) Sensible mistakes allow a statistical treatment of examining and research Presumptions are on side readd to create the research possible. When replicating stuck-at mistakes, a faulty place can don't succeed in one of two ways. A "stuck at- one" causes the place to imitate as a reasoning one, or great, value for the whole test. In the same way," stuck-at-zero" causes the place to imitate as a reasoning zero, or low, value for the whole test Input hooks are more difficult. To be sure you have recognized a stuck-at-one or stuck-at-zero feedback pin, an outcome pin must modify condition due to the feedback being toggled.

Statistics: 36 inputs; 7 outputs; 160 gates; bus translations

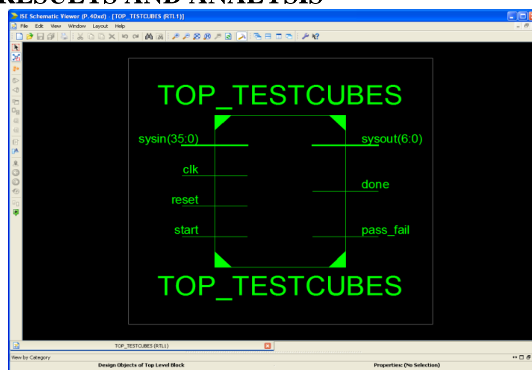
Function: c432 is a 27-channel disrupt operator. The feedback programs are arranged into three 9-bit vehicles (we call them A, B and C), where the bit position within each bus decides the disrupt demand concern. A forth 9-bit feedback bus (called E) allows and hinders disrupt demands within the specific bit roles. The determine above briefly symbolizes the circuit. The determine above contains the segments marked M1, M2, M3, M4, and M5, which contain the actual reasoning.

The disrupt operator has three disrupt demand vehicles A, B and C, each having nine pieces or programs, and one channel-enable bus E. The following concern guidelines apply: A[i] > B[j] > C[k], for any i, j, k; i.e., bus A has the most important and bus C the smallest. Within each bus, a route with a higher catalog has concern over one with a lower index; for example, A[i] > A[j], if i > j. If E[i] = 0, then the A[i], B[i], and C[i] information are overlooked.

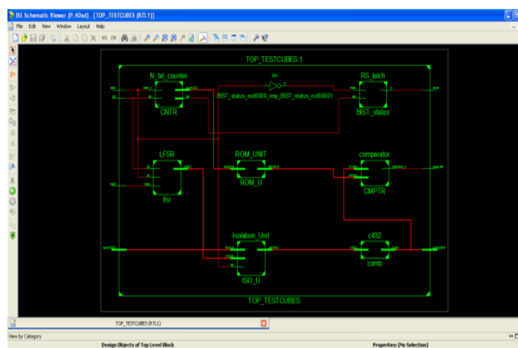
The seven results PA, PB, PC and Chan[3:0] specify which programs have recognized disrupt demands. Only the route of most important in the inquiring bus of most important is recognized. One exemption is that if two or more interferes with generate demands on the route that is recognized, each bus is recognized.

For example, if A[4], A[2], B[6] and C[4] have demands awaiting, A[4] and C[4] are recognized. Component M5 is a 9-line-to-4-line concern encoder. The outcome range designated 421 actually generates the upside down Chan[3] reaction of that proven in the fact desk. We have taken the freedom of including an inverter to outcome 421 to form Chan[3] for this desk (but not in the models).

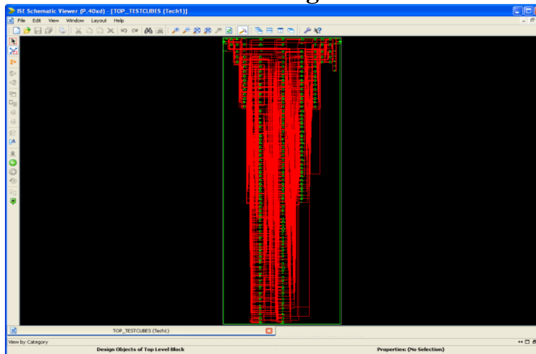
IV. RESULTS AND ANALYSIS

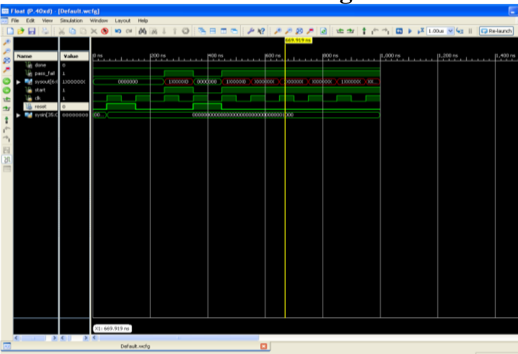


Block diagram



RTL Schematic of Test Cube generator circuit





Technology Schematic of Test cube generator Circuit Simulation Result of Test cube generator Circuit

2010-2014– IJVES

Vol 05, Article 10482, November 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	40	7,168	1%	
Number of 4 input LUTs	152	7,168	2%	
Number of occupied Slices	110	3,584	3%	
Number of Slices containing only related logic	110	110	100%	
Number of Slices containing unrelated logic	0	110	0%	
Total Number of 4 input LUTs	152	7,168	2%	
Number of bonded IOBs	48	141	34%	
Number of BUFGMUXs	1	8	12%	
Average Fanout of Non-Clock Nets	2.81			

Device utilization summary

CONCLUSION

This paper described two techniques for un specifying a two pattern test set for conversion path delay faults in full-scan tour. These techniques un specify scan-in principles. This process assures that the same conversion path delay faults will be recognized by quality set after it is unspecified. To allow additional scan-in principles to be unspecified, and thus create quality set more responsive to evaluate information a combinational bench mark circuit is used for test information compression, Answers are confirmed.

REFERENCES

[1] K. Lee, J. Chen, and C. Huang, "Using a single input to support multiple scan chains," in *Proc. Int. Conf. Comput.-Aided Design*, 1998, pp. 74–78.
[2] C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, B. Keller, and B. Koenemann, "OPMISR: The foundation for compressed ATPG vectors," in *Proc. Int. Test Conf.*, 2001, pp. 748–757.
[3] B. Koenemann, C. Barnhart, B. Keller, T. Snethen, O. Farnsworth, and D.Wheater, "A smartBIST variant guaranteed encoding," in *Proc. Asian Test Symp.*, 2001, pp. 325–330.
[4] J. Rajski, J. Tyszer, M. Kassab, N. Mukherjee, R. Thompson, K.-H. Tsai, A. Hertwig, N. Tamarapalli, G. Mrugalski, G. Eide, and J. Qian, "Embedded deterministic test for low costmanufacturing test," in *Proc. Int. Test Conf.*, 2002, pp. 301–310.
[5] P. Goel and B. C. Rosales, "Test generation and dynamic compaction of tests," in *Proc. Test Conf.*, 1979, pp. 189–192.
[6] I. Pomeranz, L. N. Reddy, and S. M. Reddy, "COMPACTEST: A method to generate compact test sets for combinational circuits,"*IEEE Trans. Comput.-Aided Design*, vol. 12, no. 7, pp. 1040–1049,Jul. 1993.
[7] S. Kajihara and K. Miyase, "On identifying don't care inputs of test patterns for combinational circuits," in *Proc. Int. Conf. Comput.-Aided Design*, 2001, pp. 364–369.
[8] A. El-Maleh and A. Al-Suwaiyan, "An efficient test relaxation technique for combinational & full-scan sequential circuits," in *Proc. VLSI Test Symp.*, 2002, pp. 53–59.
[9] K. Fuchs, F. Fink, and M. H. Schulz, "DYNAMITE: An efficient automatic test pattern generation for path delay faults," *IEEE Trans. Computer-Aided Design Integr. Circuits Syst.*, vol. 10, no. 10, pp. 1323–1335, Oct. 1991.
[10] U. Sparmann, D. Luxenburger, K.-T. Cheng, and S. M. Reddy, "Fast identification of robust dependent path delay faults," in *Proc. Design Autom. Conf.*, 1995, pp. 119–125.
[11] S. Kajihara, K. Kinoshita, I. Pomeranz, and S. M. Reddy, "A method for identifying robust dependent and functionally unsensitizable paths," in *Proc. VLSI Design Conf.*, 1997, pp. 82–87.
[12] K. Heragu, J. H. Patel, and V. D. Agrawal, "Fast identification of untestable delay faults using implications," in *Proc. Int. Conf. Comput.-Aided Design*, 1997, pp. 642–647.
[13] S. Padmanaban and S. Tragoudas, "A critical path selection method for delay testing," in *Proc. Int. Test Conf.*, 2004, pp. 232–241.
[14] K. Heragu, J. H. Patel, and V. D. Agrawal, "Segment delay faults: A new fault model," in *Proc. VLSI Test Symp.*, 1996, pp. 32–39.
[15] M. Sharma and J. H. Patel, "Testing of critical paths for delay faults," in *Proc. Int. Test Conf.*, 2001, pp. 634–641.
[16] I. Pomeranz and S. M. Reddy, "Path selection for transition path delay faults," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 3, pp. 401–409, Mar. 2010.