Implicit and Exact Path Delay Fault Grading in Sequential Circuits

M. M. Vaseekar Kumar, S. Tragoudas, S. Chakravarty, R. Jayabharathi

Implicit and Exact Path Delay Fault Grading in Sequential Circuits

M.M.Vaseekar Kumar S.Tragoudas
ECE Department
Southern Illinois University
Carbondale,IL 62901
{vaseekar,spyros}@engr.siu.edu

S.Chakravarty R.Jayabharathi
Intel Corporation Intel Corporation
Santa Clara,CA Sacramento,CA
{sreejit.chakravarty,
rathish.jayabharathi}@intel.com

Abstract

[1] The first path implicit and exact non–robust path delay fault grading technique for non–scan sequential circuits is presented. Non enumerative exact coverage is obtained, by allowing any latched error representing a delayed transition to propagate to a primary output with the support of other potentially latched errors. The generalized error propagation is done by symbolic simulation. Appropriate data structures for function manipulation are used. The advantage of the proposed method is demonstrated experimentally with consistent improvement in coverage over an existing pessimistic heuristic despite enforced bounds on the memory requirements.

1 Introduction

The need for verifying the temporal correctness of the manufactured circuit is ever increasing. The objective of delay testing is to detect timing defects which could degrade the performance of a circuit. The path delay fault (PDF) model is considered to be the most accurate and also the most appropriate delay model for testing delay defects in deep sub–micron technology. A PDF can be tested robustly, but in many circuits the majority of PDFs can only be sensitized non-robustly. We consider non-robust fault grading of PDFs in non-scan sequential circuits in this paper. The goal of fault grading is to determine the faults tested by a test set under a specific delay fault model. It is used to reduce the effort of the delay test generator as well as to indicate the completion of the test generation process by verifying that the desired coverage levels have been obtained.

Various techniques have been designed for PDF grading in combinational and sequential circuits. Most of the techniques assume that the flip-flops(*FF*) feeding the combinational part of the circuit are directly observable and controllable using full scan [4, 7]. These full-scan techniques require the usage of enhanced scan flip-flops which are excessively expensive in many circuit designs. For example, microprocessor testing is traditionally done without such DFT support. Area, performance and feasibility considerations may prevent the use of full scan. Furthermore, testing using full-scan, targets many sequentially untestable PDFs.

The fundamental difficulty in PDF testing of non-scan sequential circuits is associated with the propagation of latched errors (each representing a potentially delayed transition along a set of PDFs that end on the same flip-flop) to primary outputs(*PO*). Existing techniques introduce assumptions during the error propagation phase to simplify the process at the expense of pessimism in the grading process [3, 9, 10]. The most restricted method allows for error propagation along a path as long as it does not depend on any other error captured in another flip-flop. This guarantees sequential robustness [3]. A less restrictive sequentially non-robust approach assumes that only one error can be captured at a flip-flop during the PDF sensitization phase [3]. We will consider this method for comparison in the rest of the paper.

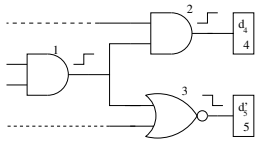

Figure 1: Multiple latching of PDFs

Figure 1 shows two robustly sensitized critical PDFs 1-2-4 and 1-3-5 latched as potentially delayed transitions at flip-flops 4 and 5. Even under the assumption that both paths are critical, with similar gate

[1]This research is supported in part by a grant from Intel Corp.

Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'05)
1530-1591/05 $ 20.00 IEEE

delays, it cannot be guaranteed that when there is a delay fault in PDF 1-2-4, there will also be a delay fault in PDF 1-3-5. Furthermore, delays of the gates vary with different sensitizing patterns and also due to process variations in manufacturing. Therefore the latched value on the end flip-flop of each (critical) sensitizable path should be considered as a potential error that may or may not occur. In the proposed method there are no restrictions on the number of flip-flops that are allowed to have an error during the propagation phase. The method is also exact. The grading methods in [3] cannot handle this generalization unless the underlying grading algorithm is invoked an exponential number of times to the length of the error propagation subsequence.

Testing for delay faults can be done so that all patterns are applied at the rated clock (at-speed), (see [8], among others) or using a variable clock method (see [3, 9], among others). The latter is a pessimistic method where only one pattern in the test sequence is applied at a rated clock and will be the approach followed in this paper. Let this pattern be p_2 and the predecessor pattern p_1. The sub–sequence prior to pattern p_1 initializes the flip-flop in appropriate states. The slow application of p_1 allows for all signals to settle to an appropriate logic prior to the rated-clock application of p_2. Delayed path propagations are then captured as errors on flip-flops or observed directly at the *PO*s. The subsequence following pattern p_2 is subsequently applied with a slow clock to propagate any captured errors to *PO*s. *Any vector in this sub-sequence is called a propagation vector*.

Our technique is based on the derivation of functions by symbolic simulation and their appropriate manipulation which ensures that any sensitized and observable PDF will be detected with or without the support of other latched error combinations representing potentially delayed transitions. Such functions essentially provide with parallelism of the traditional fault simulation procedure in [3]. This also eliminates the exponential time behavior of an exact grading simulation method based on the procedure in [3].

The formed functions, model the propagation of the delayed transitions from a flip-flop to another flip-flop or *PO*. Their generation is independent to the number of PDFs. They are stored and manipulated using Binary Decision Diagrams (BDDs). In an initial step, the proposed non-enumerative PDF grading method uses the *implicit* technique in [7] for combinational circuits to store the PDFs that are sensitized during the rated clock phase in different ZBDDs. The distinction is made upon the ending flip-flop or *PO* on each sensitized PDF. That way, a latched error represents a set of implicitly stored PDFs. Subsequently the proposed fault grading method systematically and implicitly identifies whether a propagation vector v_i, results to an error at any *FF* or *PO* denoted by $'F'$, and for each $'F'$ determines which error prior to applying v_i propagates to $'F'$. This is done with the help of the functions formed by symbolic simulation. Thus, if $'F'$ is observable we are able to implicitly determine all PDFs propagating to it through the corresponding error.

This paper is organized as follows. Section 2 presents an overview of the symbolic simulation method required for the proposed fault grading technique. Subsequently, it outlines the fault grading methodology and illustrates it with an example. The methods are presented without explicit reference to the data structures that are used to represent the functions. Section 3 gives experimental results that demonstrate the need for an exact grading scheme. The improvement over existing heuristics is very significant for all benchmarks we experimented with. Section 4 concludes.

2 The proposed method

2.1 Overview

Let f_j denote a flip-flop. Let d_j be an error representing a potentially delayed rising transition and d'_j be an error representing a potentially delayed falling transition latched at f_j at the end of the rated clock. The four different cases that may occur are shown in Table 1.

Table 1: Encoding delayed transitions

value	Good	Faulty
0	0	0
1	1	1
d_j	1	0
d'_j	0	1

The propagation of an error captured at a flip-flop (at the end of the respective sensitized PDF) is determined by symbolic simulation. Consider the application of the *first propagation vector*. Since an error at a flip-flop (encoding a delayed transition) may or may not occur, we maintain a variable f_j at the output of the flip-flop f_j to denote a d_j error. Similarly a variable f'_j for a flip-flop f_j denotes a d'_j error. Note that a variable is allowed to have any of the two assignments 0 or 1 and likewise an error may or may not occur. For convenience of representation, we use the same name for the flip-flop and its variable.

Consider a portion of the circuit as shown in Figure 2. Let f_1, f_2, f_3 be three input flip-flops, and f_x

Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'05)
1530-1591/05 $ 20.00 IEEE

be the next-state flip-flop. Now assume that errors d_1, d'_2, d_3 are present at the beginning of the slow clock. We do a symbolic simulation and for each next-state flip-flop f_x we obtain the function F_x at its input. Thus F_x will be a function of (f_1, f_2, f_3). Assume that $F_x = f_1.f_2.f'_3 + f'_1.f_3$. We conclude that error d_x is latched at f_x either due to f_1 (in the first cube) or f_3 (in the second cube). This indicates the existence of a possible input assignment that could propagate both the errors latched at f_1 and f_3, respectively to f_x. The union of PDFs represented by d_1 and the PDFs by d_3 will yield the PDFs represented by error d_x.

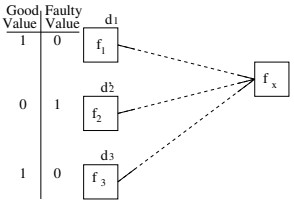

Figure 2: Example

We also conclude that error d'_x is latched either due to f_2(in the first cube) or f'_3(in the first cube) or f'_1(in the second cube). Thus there exists a possible input configuration that could also bring a d'_x at f_x. As far as PDF coverage linked to error d'_x we take the union of PDFs represented by d'_2, d'_3 and d'_1. It should be noted that the set of PDFs represented by both d_1 and d'_1 is the same. Then for the next slow clock propagation vector we will have both d_x and d'_x in flip-flop f_x for symbolic simulation.

Let $F_x^{f_1}$ represent all the cubes that contain variable f_1 and $F_x^{f_3}$ represent all the cubes that contain variable f_3 in F_x. Functions $F_{x'}^{f_2}$,$F_{x'}^{f'_3}$, $F_{x'}^{f'_1}$ are derived similarly for error d'_x. Consider another flip-flop f_y and without loss of generality assume that only error d_y is latched. Error d_y is the same as d_x if and only if $F_x^{f_1} = F_y^{f_1}$ and $F_x^{f_3} = F_y^{f_3}$, and, *in addition, f_1 and f_3 must be the only variables that allow error d_y*. If this is the case, for the next slow clock propagation vector, d_y is denoted as d_x.

Likewise d_y is the complement of d_x if and only if $F_y^{f_2} = F_{x'}^{f_2}$ and $F_y^{f'_3} = F_{x'}^{f'_3}$ and $F_y^{f'_1} = F_{x'}^{f'_1}$, and, *in addition, f_2, f'_3, f'_1 are the only variables that allow error d_y*. If this is the case, for the next slow clock propagation vector, d_y is denoted as d'_x. The above mentioned variable bindings are needed to ensure that the use of symbolic simulation during any propagation vector is correct and does not overestimate the coverage. It can be shown that the number of BDD operations required to determine all variable bindings prior to any symbolic simulation is in the order of $O(F^3)$ where F is the number of flip-flops. However, the procedure is much faster in practice.

The same process is followed for any subsequent propagation vector. A minor difference is that at the beginning of the simulation a flip-flop may have both errors latched. A flip-flop f_i that has both errors d_i and d'_i latched, is still treated as variable f_i. Assume that after symbolic simulation we obtain function $F_x = f_i(A) + f'_i(B)$. Then both d_i and d'_i propagate due to d_x and d'_x as long as either $'A'$ or $'B'$ is not $NULL$. We calculate the PDFs covered under d_x (respectively, d'_x) as before, by taking the union of the PDFs in errors for d_i(respectively, d'_i) as depicted by variable f_i.

2.2 Illustration

The grading method supports any number of initialization and propagation vectors provided that only one vector is applied at the rated clock to detect PDFs, and is illustrated with an example. For each propagation vector we show how to use symbolic simulation method to implicitly identify any detectable PDF. The example also shows that the approach in [3] is quite pessimistic.

Consider the circuit shown in the Figure 3. Assume a four vector sequence *v1(1110000100), v2(0010000011), v3(1011100011), v4(1111100011)* where *v2* is applied at the rated clock, and *v3*, *v4* are propagation vectors. Figure 3 shows the symbolic simulation values at different clocks, separated by commas. Testable PDFs are detected using vectors *v1,v2* as in [7] and are stored as a ZBDD. PDFs ending in flip-flops are latched as errors representing potentially delayed transitions. Table 2 (left part under clock 2) shows the delayed transitions, and the PDFs represented by them in columns $1-3$. Symbols $'f'$(respectively, $'r'$) denotes a falling transition (respectively, rising transition) on an edge of a PDF. Also note that all PDFs ending at the same flip-flop are represented by a symbol as shown in column 3 of Table 2. Let F_k_list be a list of symbols that represents all PDFs detected by $v2$ at flip-flop f_k. For example the symbol $P\_d'_4$ is listed in $F_{f_4}\_list$ instead of the ZBDD representation.

Vector *v3* is applied with a slow third clock to propagate the set of latched errors $\{d_1, d_3, d'4\}$. To determine the propagation of these errors, we perform a symbolic simulation with the corresponding variables $\{f_1, f_3, f_4\}$, and *v3*. The function $F_{f'_2}$ is $f'_3.f_4$. Thus at the end of the third clock errors d_3 and d'_4 propagate to flip-flop f_2 and will be represented by error d'_2 for symbolic simulation purposes when considering propagation vector *v4* later. It is important to note that since both d_3 and d'_4 propagate to the same output, a set union operation of the PDFs corresponding to these errors will yield the PDFs covered by both. Thus $F_{f_2}\_list$

Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'05)
1530-1591/05 $ 20.00 IEEE

Table 2: Delayed transitions and PDFs

Clock2			Clock3		
Delayed transition	PDFs represented	Symbol	Delayed transition	PDFs represented	Symbol
d_1	$2f - 12r$	P_d_1	d'_1	$2f - 12r$	P_d_1
d_3	$8f - 18r$	P_d_3			
d'_4	$9r - 16r - 21f$, $10r - 16r - 21f$, $8f - 18r - 21f$	$P\_d'_4$	d'_2	$9r - 16r - 21f$, $10r - 16r - 21f$, $8f - 18r - 21f$, $8f - 18r$	P_d_3 $+P\_d'_4$

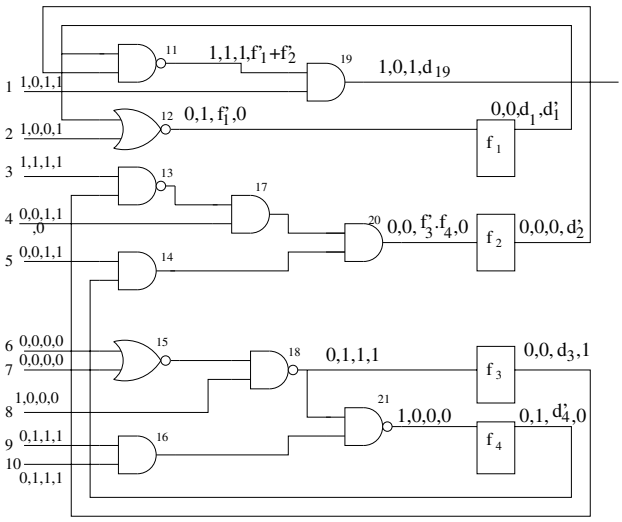

Figure 3: PDF grading

will have both the symbols $P\_d'_3$ and P_d_4. Flip-flop f_2 cannot have a d_2 value as there are no variables in the cube that could latch a d_2 error.

Similarly $F_{f'_1} = f'_1$ and can latch d'_1. It can be seen that variable binding is not possible in this case as both $F_{f'_2}$ and $F_{f'_1}$ have different variables. The above mentioned activities are summarized in the right hand side of Table 2 (under clock3), columns 4-6, when applying the first propagation vector v_3.

This demonstrates the advantage of the method, as the existing non-robust model of [3] could not have propagated either of the latched errors. This is because, the good machine value as considered in [3] at f_4 is 0 (f_3 is 1), which leads to a controlling value to the gate 20, thus in such cases there is a significant loss of fault coverage.

The procedure is repeated for *v4* with variables $\{f_1, f_2\}$ representing $\{d'_1, d'_2\}$ respectively. It is observed that the function formed at *PO* is $F_{19} = f'_1 + f'_2$ denoting a possibility of d_{19} at *PO* due to f'_1 or f'_2. Since PDFs corresponding to different delayed transitions are mutually exclusive, we work in an implicit manner and perform a union of PDFs represented in the $F_{f_1}\_list$ and $F_{f_2}\_list$, which is a constant time operation. PDFs $P\_d'_4$,P_d_3 and P_d_1 are covered by this test. Thus this method yields high coverage in testing non-scan sequential circuits by allowing delayed transitions to propagate to flip-flops in the presence of other potentially delayed transitions, non-robustly and non-enumeratively.

2.3 The fault grading algorithm

The procedure of the fault grading algorithm is given below. C is the circuit and t is the test set consisting of three types of vectors. Let t_{in} denote the set of initialization vectors, t_r the single activation vector and t_p the set of propagation vectors, respectively.

```
PROCEDURE Fault_Grading( C, t)
initialization(C,t_in)
( PDFs_detected, X_list)=grading(C,t_r)
For every t_pj ∈ t_p do
  For every output X ∈ (PO/FF) do
    C_sub=cone(C,X)
    F_a = symbolic_sim (C_sub, t_pj)
    For every f_i ∈ C_sub with possible (d_i or d'_i) do
      If (check_propagation(f_i, F_a)) then
        variable_binding(f_i, C)
        If( X ∈ FF) then update(X_list, f_i_list)
        else If( X ∈ PO) then
          add( PDFs_detected, f_i_list)
```

The $initialization()$ function initializes the flip-flops by simulating t_{in}. The $grading()$ function identifies the PDFs tested, with t_r applied at the rated clock, as per [7]. PDFs ending at the *PO* are added to the ZBDD $PDFs\_detected$ while those ending at flip-flop are stored as ZBDDs and added as symbols to their respective X_list.

The $cone()$ function returns the sub-circuit cone C_{sub} with respect to X. Working with every output cone of the circuit reduces the complexity of the algorithm. The $symbolic\_sim()$ function performs the symbolic simulation with the flip-flop variables and t_{p_j} and

Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'05)
1530-1591/05 $ 20.00 IEEE

returns F_a as explained in section 2.1 and 2.2 where $'a'$ can be either X or X' or both. The $check\_propagation()$ function checks whether the d_i or d'_i in f_i propagates as d_X or d'_X to X. If the latched d_i or d'_i propagates to X then the $binding()$ function checks for possible partial variable binding with previously computed d_X or d'_X errors in the circuit $'C'$. At the end of the loop any possible variable binding is detected and the symbols are substituted.

If X is a flip-flop the $update()$ function updates the X_list with the PDFs in f_i_list. If X is a *PO* the PDFs in f_i_list are added to $PDFs\_detected$. The procedure is further simplified by updating the PDFs detected as a list of symbols, rather than a ZBDD and performing a final ZBDD Union operation, as no new PDFs are introduced in the propagation phase. The fault grading procedure ends when all the propagation vectors have been applied. This procedure is repeated for other test sets and $PDFs\_detected$ is updated. The total number of PDFs stored in the $PDFs\_detected$ will give the coverage of PDFs for the set of test patterns. The time complexity of the proposed algorithm is polynomial. The method is exact as it guarantees the detection of any sensitized PDFs, provided they are sensitized in only one clock.

3 Experimental Results

We present experimental results for the non-scan version of some ISCAS'89 and ITC'99 sequential circuits to demonstrate the enhancement in the fault coverage over the existing method in [3]. To ensure low memory requirements by the BDD data structure, we implemented a restricted version of the proposed algorithm where flip-flops are greedily partitioned into sets of size utmost ten. The propagation of an error latched at flip-flop f_i in set $'A'$ can only be assisted by errors on flip-flops within set $'A'$. All other flip-flops that are not in set $'A'$ are assigned error free good machine values. Thus very simple functions F_x are formed and the approach can execute with less than 256 MB RAM. This however, may increase the execution time for some benchmarks because for each propagation vector we have to perform as many symbolic simulation as the number of sets. Note that the method in [3] requires $O(F)$ logic simulations per propagation vector where $'F'$ is the number of flip-flops. The proposed grading tool has been implemented in the C language.

Two sets of experiments were performed on a 750MHz SunBlade 1000 workstation and are shown in Table 3. In all the experiments, the initial state of flip-flops is assumed to be 0. In both experiments the PDFs are stored as ZBDDs. BDDs are used for generating the functions, determining variable binding and determining the propagation of latched delayed transitions.

The first set of experiments, listed in the left-hand side of Table 3 (columns $2-6$), are conducted considering three vector sequences for sequential grading. The first two vectors are obtained from function–based ATPGs [5, 6] for PDFs in combinational or full-scan circuits, and are used as the initialization and fault activation vectors in the sequence. The second vector is applied at the rated clock to sensitize the PDFs non-robustly. Test sets for the first three circuits were derived from a compact ATPG [5]. As it was time consuming to derive test sets for larger circuits using [5], we used the ATPG from [6], which is less compact. The last propagation vector which is applied at a slow clock is the same as the second vector of the ATPGs [5, 6]. The PDFs represented by the latched errors propagating to a primary output are immediately detected. The latched errors that propagate to next state flip-flops, by the third vector are assumed to be shifted out to an observable point, for determining the possible coverage. We experiment with such three vector sequences because we were unable to obtain the sequential ATPG tool in [3].

Column 1 of Table 3 gives the circuit names. Column 2 presents the number of test sets (each test set has three vectors). Column 3 and 4 compare the coverage of PDFs with the non-robust model of [3]. The 5^{th} column shows the improvement in the coverage as a percentage. It can be observed that the proposed methodology yields a better coverage than [3]. Often the improvement is very significant and over 20%. The results for s6669, s38417 and s38584 confirm the scalability and implicitness of the method. It also shows that the method can handle path intensive circuits. It should be noted that, though the percentage improvement is not so high in s38417 and s38584, the number of PDFs covered has significantly increased over the heuristic method. It is also important to note here that appropriately derived error propagation sequences (by a sequential ATPG for PDFs) could favor the presented method, and the improvement would have been more drastic, since they could allow for many errors to propagate to *PO* or flip-flop. Column 6 in Table 3 gives the execution time in seconds. The listed time includes the time for generating all the required functions, but most of the execution time is spent on the actual fault grading process.

The second set of experiments, listed in Table 3 from columns $7-12$, uses test sequences derived from a genetic algorithm based ATPG (GATTO) [2] for stuck at faults in non-scan sequential circuits. The columns provide information similar to columns $2-6$. Each test sequence, which has n vectors, was repeatedly executed

Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'05)
1530-1591/05 $ 20.00 IEEE

Table 3: Result PDF grading

	Single Propagation Vector using ATPG [5, 6]					Multiple Propagation Vectors using ATPG [2]					
Benchmarks	Test Sets	Coverage [3]	Exact Coverage (proposed)	Improvement (%)	Execution Time (sec)	Test Sets	Coverage [3]	Exact Coverage (proposed)	Improvement (%)	Average # Propagation Vectors	Execution Time(sec)
s641	947	1794	2039	13.65	10.54	265	366	401	9.56	11	2.16
s713	403	2429	3078	26.72	4.57	265	414	463	11.83	14	2.08
s820	753	883	968	9.62	2.25	240	192	215	11.98	19	1.56
s1269	10182	7125	8017	12.52	31.02	422	2888	4196	45.29	16	18.64
s3271	7707	2508	2807	11.92	15.23	1424	6634	9958	50.11	18	373.58
s6669	115272	280377	341677	20.71	913.36	438	104738	144907	38.35	15	383.01
s38417	39799	76451	81073	6.05	1583.20	704	7911	10718	35.48	21	6607.60
s38584	92239	90163	99533	10.39	2676.16	1266	14422	16294	12.98	19	4593.99
b03	1244	988	1213	22.77	6.95	151	163	198	21.47	10	1.35
b04	6152	5702	6078	6.59	14.32	627	6059	6537	7.89	18	36.86
b09	1058	1048	1392	32.82	7.47	456	211	243	15.16	21	10.77
b10	831	676	828	22.48	1.16	467	342	383	11.98	16	4.51
b11	3689	3310	3725	12.53	28.99	467	660	830	25.75	22	23.83
b12	6152	10025	10783	7.56	91.79	254	247	324	16.42	20	2.08

n times. Each time a different vector served as the fault activation vector. The vectors applied before the activation initialize the states of the flip-flops and the vectors applied after the activation vector propagate the captured delayed transitions to an output. Thus each test set would have a varying number of initialization and propagation vectors. The total number of such test sets are reported in column 7. The average number of propagation vectors per test set are reported in column 11, and column 12 gives the execution time taking into account all the propagation vectors for the entire test set. It is noted that the execution time for a single propagation vector is very low and experimentally confirms that the method is implicit and can handle any number of propagation vectors.

The low PDF coverage in both experiments is attributed to the lack of compact and implicit sequential PDF ATPGs where each sequence detects many faults. Such tests would have also better shown the impact of the exact method over heuristic approaches. It is known that the listed benchmarks are not PDF testable by random pairs of vectors even in the presence of full scan. The fault coverage using randomly generated test sequences for sequential PDF testing results in very low coverage and analyzing such an experiment has little value. We were not able to access the sequential ATPG in [3]. However, existing sequential PDF ATPGs are fault enumerative[3]. Thus each generated test sequence targets only one PDF and incidentally may detect a few more, but the PDF coverage still remains low. Therefore, it is not expected that the impact of our exact method over heuristic grading methods [3, 9] will be greater (than what is shown here) when experimenting with sequences from such tools.

4 Conclusions

We propose the first method for *exact* path delay fault grading in non-scan sequential circuits using a variable clock. The propagation of captured errors that represent delayed transitions, which has been treated in a pessimistic manner by previous approaches, is handled accurately using functions. ZBBDs and BDDs are used to store PDFs compactly and manipulate functions, respectively. Experimental results show that the fault coverage is significantly increased over existing heuristic approaches.

References

[1] R. Bryant, *Graph-based algorithms for boolean function manipulation,* IEEE Trans. on Computers, Vol. C-35, No. 8, pp. 677-691, Aug. 1986.

[2] Corno, F.; Prinetto, P.; Rebaudengo, M.; Sonza Reorda, M, *GATTO: a genetic algorithm for automatic test pattern generation for large synchronous sequential circuits*, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Volume: 15 , Issue: 8 , pp. 991 - 1000, Aug. 1996 .

[3] T.J. Chakraborty, V.D. Agrawal,M. L. Bushnell, *On Variable Clock Methods for Path Delay Testing of Sequential Circuits*, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, pp-1237-1249, Nov. 1997.

[4] M. A. Gharaybeh, M. L. Bushnell, and V. D. Agrawal, *The path-status graph with application to delay fault simulation*, IEEE Trans. Computer-Aided Design, vol. 17, pp. 324332, Apr. 1998.

[5] M. Michael and S. Tragoudas, *ATPG for Path Delay Faults without Path Enumeration* , in Proc. International Symposium on Quality of Electronic Design, 12 Clara, CA, Mar. 2001.

[6] S. Padmanaban, S. Tragoudas, *Using BDDs and ZBDDs for efficient identification of Testable Path Delay Faults*, Proc. Design Automation and Test in Europe Conference (DATE), pp. 322-327, Feb. 2004.

[7] Padmanaban S., Michael M. and Tragoudas S., *Exact Path Delay Fault Coverage with Fundamental Zero-Suppressed BDD Operations*, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, pp-305-316, Mar.2003.

[8] I. Pomeranz, S.M. Reddy., *At-Speed Delay Testing of Synchronous Sequential Circuits*, 29^{th} ACM/IEEE Design Automation Conference, pp. 177-181.

[9] I. Pomeranz, S.M. Reddy., *SPADES-ACE: A Simulator for Path Delay Faults in Sequential Circuits with Extensions to Arbitrary Clocking Schemes"*, IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol.13,No.2, pp-251-263, Feb. 1994.

[10] Tekumalla, R.C.; Menon, P.R., *Identification of primitive faults in combinational and sequential circuits* IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume: 20, Issue: 12 , pp. 1426 - 1442,Dec. 2001.