PROJECT REPORT

Timing False Path Identification using ATPG Techniques.

A Project Report submitted to the University of Wisconsin – Department of Electrical and Computer Engineering in partial fulfillment of the requirements for the degree of Master of Science Graduate Program in Electrical and Computer Engineering.

Advisor: Prof. Azadeh Davoodi.

SHREYAS VIJAY PARNERKAR

9057056771

ACKNOWLEDGEMENT

I would like to express my deep and sincere gratitude to my advisor Prof. Azadeh Davoodi, for her extended support and guidance throughout my coursework and this project. Her wide knowledge and logical way of thinking have been of great value for me. Her encouragement and personal guidance provided a good basis for this study.

I owe my loving thanks and gratitude to my parents, Mrs. Kalpana V Parnerkar and Mr. Vijay D Parnerkar who have been extremely supportive throughout my M.S program. I would like to express my gratitude to my fiancée Anushree for her loving support.

Shreyas Parnerkar

Table of Contents

LIST OF FIGURES

Timing False Path Identification using ATPG Techniques.

1. MOTIVATION

Timing analysis tools can report critical paths which are characterized by a transition at each node along the path; however, they cannot generate a “witness” vector which would sensitize the path. This gives possibility of having paths, reported as critical paths, for which there exists no vector to sensitize the path. The goal of the project is to identify these “false critical timing paths” without much overhead, so that the efforts required redesigning and/or optimizing the critical paths is reduced. Static Timing Analysis (STA) Tools are used by designers to determine whether the timing requirements are met. STA has a well-known timing false path problem. A large percentage of timing paths identified as critical may be false due to logic and temporal constraints. The impact is wasted redesign and optimization engineering effort with trade-off of power and/or area for no performance gain. It may also lead to underestimation of the design performance. Hence effective techniques are required to identify the false paths and to reduce the cost of optimization.

2. PROBLEM STATEMENT

The objective is to develop an efficient algorithm to find the longest sensitizable paths. The problem of identifying whether a path (may not be the longest one) is a false path or not is referred to as the general false path identification problem. A false path is often simply referred to as a path which is never activated by any input vector. An algorithm is to be developed that will take in account the timing information of the combinational circuit along with its logic

information during false path identification. We will try to claim that when timing information is taken into account, the total number of identified “logical” false paths will go down.

3. BACKGROUND AND PRELIMINARIES

A critical timing path (P) is characterized by a set of n nodes x1; x2; ….; xn and a set, T = {t1; t2; …; tn}, of signal transitions such that $t_i \in T$ represents the logic value transition on x_i. Node x_i is called a path input for path P. Many of these path inputs are associated with gate devices g_i of path P which can have other inputs which are called side inputs. The transition t_i of each node x_i is characterized by a pair of Booleans <bi; ai> where bi and ai are the initial (or before) and final (or after) Boolean values at node x_i, respectively. bi and ai are always complementary to each other, since we are concerned with the signal transition on every node along the path. {b1; b2; … ; bn} is called the before set and {a1; a2; …. ; an} is called the after set. The time frame associated with the application of the before set is considered the previous time frame, while that associated with the application of the after set is the current time frame.

In Figure 1, there are 8 library cells and/or gates in the circuits, path inputs x1, x2, ..., x7 of gates g1, g2, ..., g8, go through transitions < 0; 1 >, < 1; 0 >, < 1; 0 >, < 1; 0 >, <0; 1 >,< 1; 0 > and < 1; 0 >. Gates gi can have other inputs like x9 for gate g3, x10 for gate g4 which are side inputs of the bold-faced path in the figure.

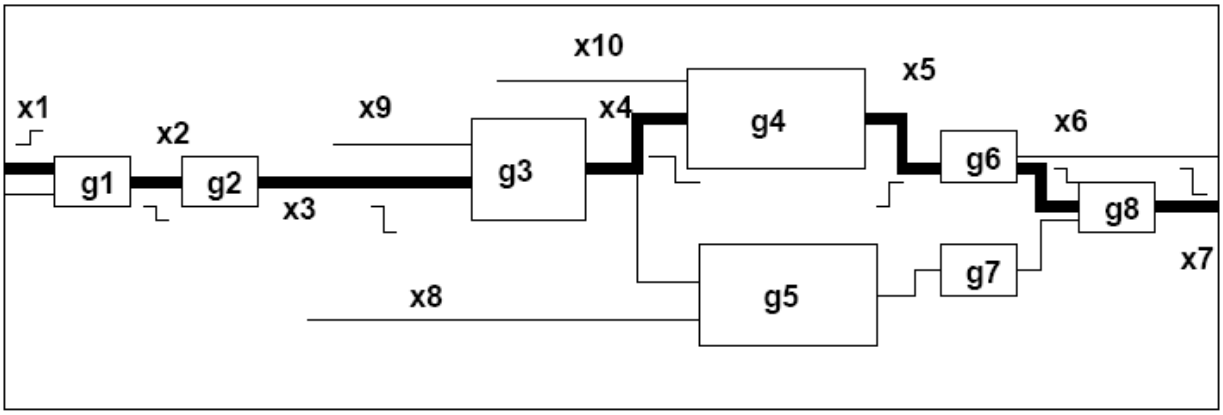


Figure 1 : Critical Timing Path

DEFINITIONS:

A false path is a path along which a specified logic transition cannot be sensitized. Timing paths identified using structural timing analyses are considered to start from the launch point and end at the capture point. Each node xi is either the primary input or output of a sequential element, library component or custom macro.

Timing paths with outputs of sequential elements as their launch points and inputs of sequential elements as their capture points are called latch to latch timing paths.

Timing slack is defined as the difference between the required arrival time and the actual arrival time at a capture point of a timing path.

Static timing analysis can be configured to run so that only the single worst case timing path is generated for each capture point in the circuit. These paths are called **main paths**. Main paths can be ordered based on their timing slacks at the capture points with the path with the worst slack showing up first in the report. Besides the main path for a particular capture point, paths converging to the same capture point, with differences in timing slacks from that of the main path within a given threshold can be generated. These paths are called **sub-paths**. Sub-paths and their corresponding main path form a group of converging paths.

3.1 LOGIC PATH SENSITIZATION

In order to check whether a transition can be sensitized through a path P, we need to check if every xi on the path can take up the values bi as well as ai, which is equivalent to checking the satisfiability of the following Boolean functions.

ea = True iff for all i, xi = ai can be justified simultaneously when evaluated in the current time frame,

eb = True iff for all i, xi = bi can be justified simultaneously when evaluated in the previous time frame.

The subscripts in the above terms ea, eb help to indicate the evaluation of different criteria. We call the boolean function which checks the satisfiability of all the ai's as the after value criterion, while the corresponding one for bi's as the before value criterion.

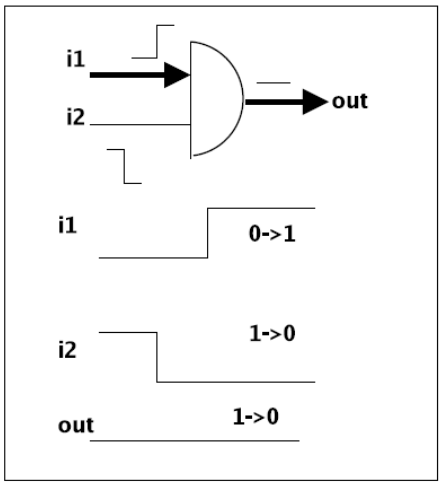


Figure 2 : Logical False Path due to after value violation.

Failing to satisfy any of the ai implies that P is a logical false path. In Fig. 2, it shows a 2-input AND gate with inputs i1 and i2 and output out. Let us consider the critical path section P (i1,

out) and the 0 -> 1 transition at i1. If at the same time, the transition on i2 can only be 1 -> 0, then out cannot assume its final value 1. A delay of the 0 -> 1 transition along path p cannot manifest at the capture point.

On the other hand, failure to satisfy a given bi does not necessarily make a path false. For example, in Fig. 3, for a 2-input AND gate, having inputs i1 and i2 and output out. Let us take the critical path {i1; out} and the 1 -> 0 transition on both nodes. If i1, which is on the critical path, is undergoing a 1 -> 0 and the side input, i2, is undergoing a 0 -> 1 transition then the before value criterion is not satisfied. But if the transition on i2 happens before the transition on i1 (which is a possibility since the path {i1; out} is the critical path) then there exists a functional test for this delay path. To account for the effect of the side inputs, the values in the before set only need to be checked when controlling values on the side inputs of the corresponding gates are assumed. The following criteria take into consideration the violation of the before set when the side-nodes of path P have controlling values.

en = True iff respective non-controlling values can be assigned simultaneously at all side-nodes when evaluated in the current time frame.

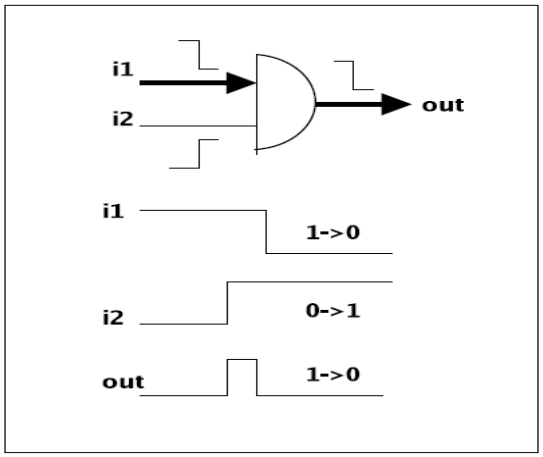


Figure 3 : Effect of Side Inputs

3.2 ALGORITHM FOR IDENTIFYING LOGIC FALSE PATH

This section describes the algorithm [2] used for logical false path identification. Figure 4 shows the algorithm as a flowchart. The decision boxes (black colored) correspond to the after value criteria, ea, before value criteria, eb, and the side value criteria, en defined above. Initially, the after value criteria is checked. If this ea fails, then logically, the path is not able to attain a steady state value (this value is the current time frame value at a node in the path: each path is defined as a set of nodes with <ai,bi>, where ai = after value/current time frame and bi = before value/previous time frame). If ea is true, then we check for the side value, en criteria. This checks if the side inputs at a path can take non-controlling values. If the side inputs can take non-controlling values, the side input is not going to dominate the output of the gate. Hence if en is true, the path is not a logical false path.

However, if en fails, we check for the before value criteria, to see if the nodes at the path can take the previous time frame value. If the nodes fail to take the previous time frame value, the path is a logical false path.

ATPG tools are used for checking the after, before and side value criteria [6] [8].

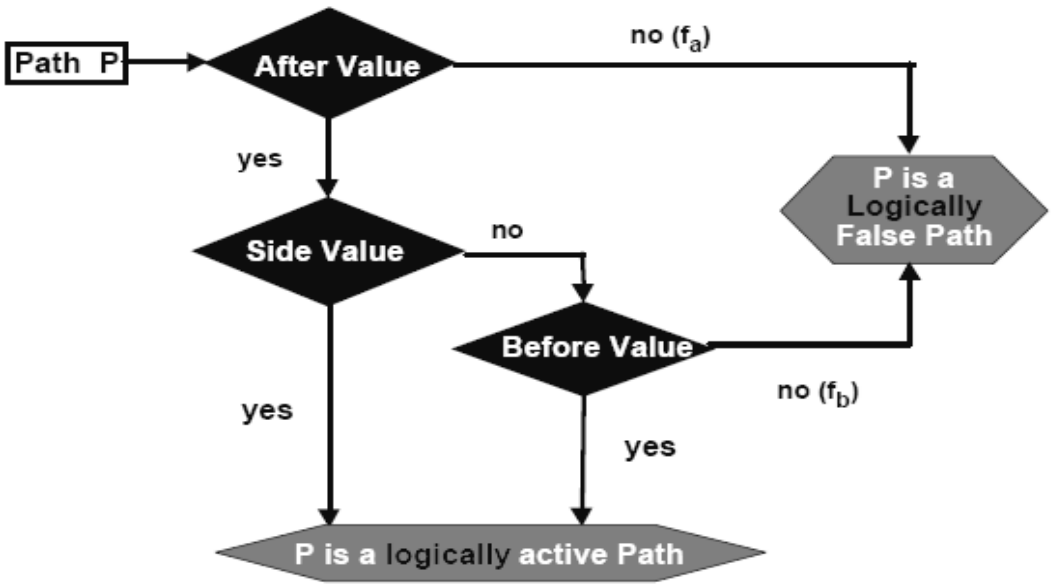


Figure 4: ALGORITHM A

4. PROPOSED METHOD

There are a few potential problems in identifying false paths using the "Logical False Path Identification" technique mentioned above. Functional false path detection techniques do not consider the delay information for the gates and rely only on the logic information. This makes this analysis underestimate the performance. A path might not be logically sensitizable but when we take the delay information for the gates into account, the transition can still propagate through this path. The following example demonstrates this limitation.

Consider path i1-out with rising transition at the input (i1). The side input i2 takes a controlling value (0 in this case). Hence the side value criterion (en) will be violated and hence i1 – out will be termed as a functional false path.

But as shown in Fig. 5, since i2 (side input) takes the controlling value (0) after the transition occur at i1, the transition still propagates from i1 to out and we see a 0 to 1 transition at output.

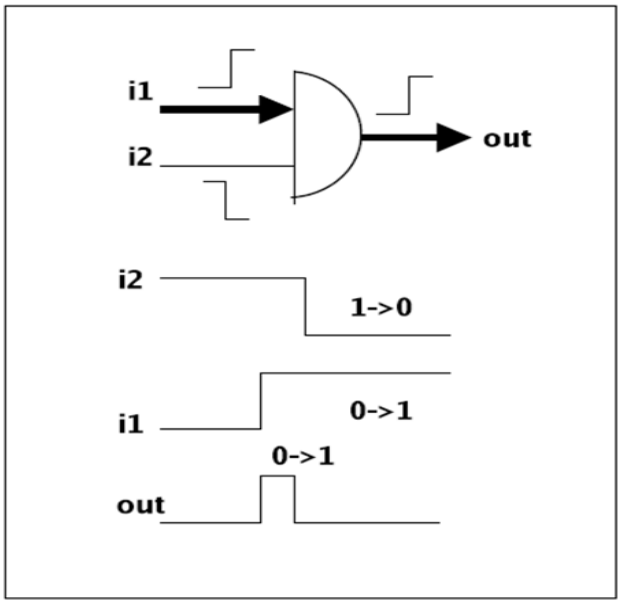


Figure 5: Effect of Side Inputs Considering the delay information

4.1 DEFINITIONS

4.1.1 RACE CONDITIONS

The delay values are analyzed at each node on the path. The worst case delay for "controlling value" at the side nodes is calculated. If the transition on the path at node under consideration occurs after the worst case delay for controlling value, this node is considered as a race condition node.

The upper bound for nodes with race condition is as shown above. This helps in faster recognition of nodes with race conditions.

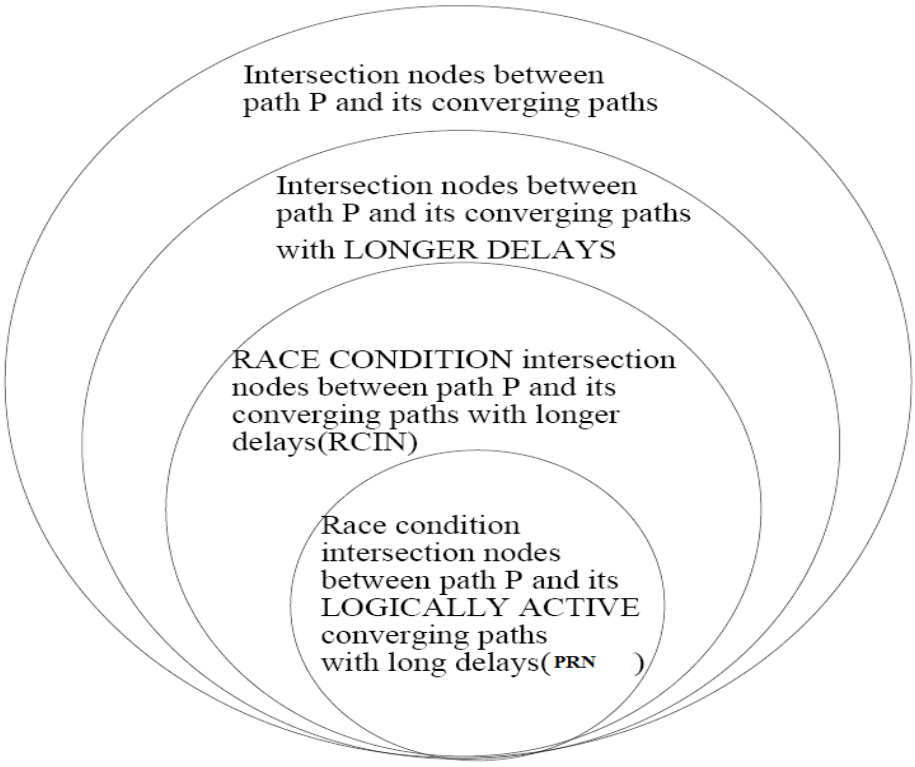


Figure 6 : Upper Bound Identified for Race Conditions

4.1.2 REVISED NON-CONTROLLING VALUE (SIDE VALUE) CRITERION (ERN)

Once we find the nodes with race condition, these nodes are the potential nodes that can block the transition on the path under consideration. Hence these nodes need to be further analyzed. We define a parameter ern.

ern = true iff non-controlling values can be assigned at nodes which have Race conditions.

4.2 ALGORITHM B

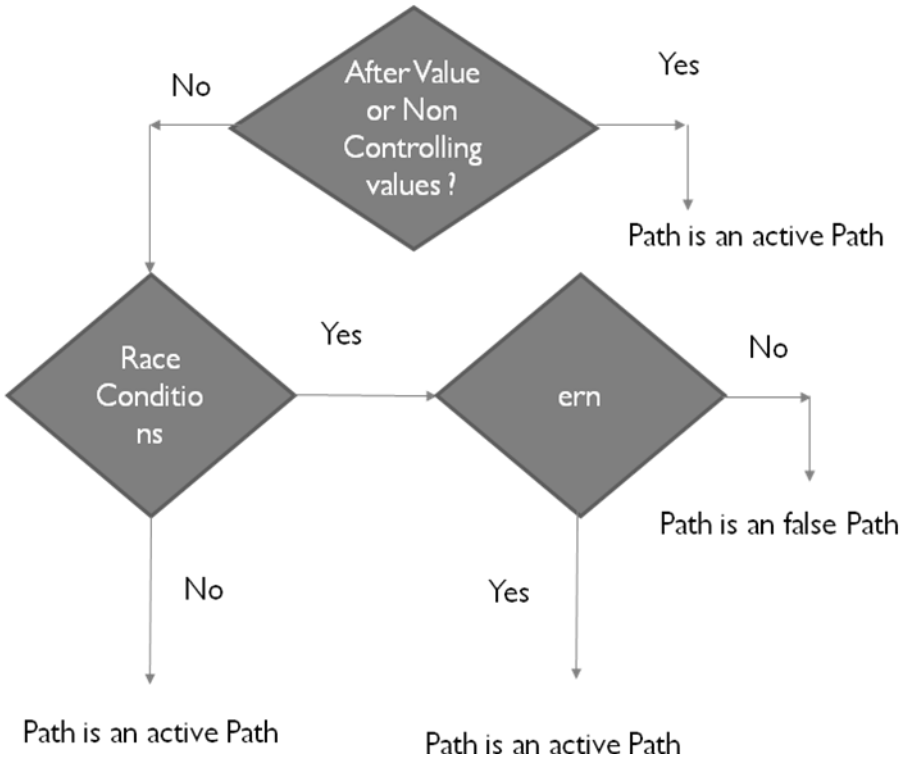


Figure 7 : ALGORITHM B

Initially after and non-controlling values are checked at all nodes on the path. If these conditions are true, the path is an active (true) path.

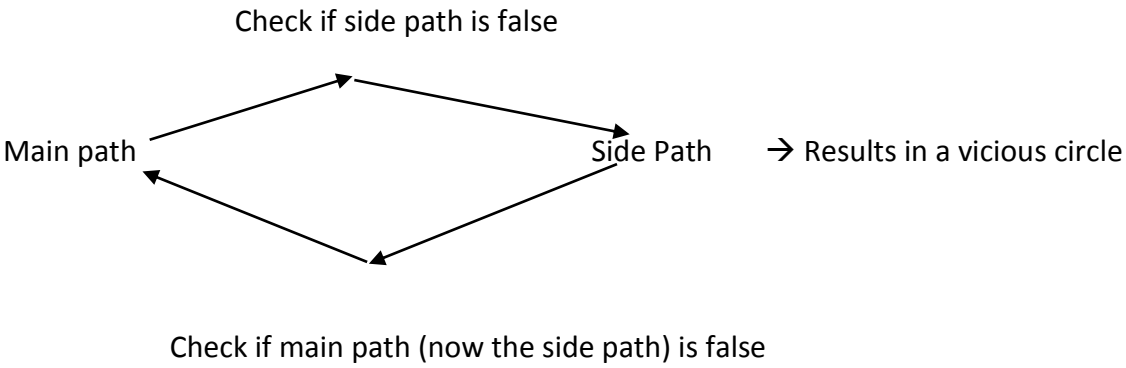
If these conditions fail, race conditions are found at all nodes on the path. If some nodes have race conditions, then these nodes are checked again for non-controlling values (assigning non-controlling values simultaneously on these nodes). If non-controlling values cannot be assigned simultaneously at race condition nodes, then the path is a false path.

The path is true if we can assign non-controlling values at nodes with race conditions. This ensures that there will be some side paths which will allow the propagation of transitions on the path under consideration.

4.3 IMPROVED ALGORITHM

We found that Algorithm B in figure 7, still identifies a lot of paths to be false paths (refer Section 5 for simulation results). We studied the potential reasons for algorithm B to identify more number of paths to be false paths. We anticipated that, since we were considering the worst case delay value at the side inputs, there could be a chance of pessimism in this approach. Another reason that we feel is causing Algorithm B to identify a large number of paths to be false paths is that the side paths are not checked for it to be a false path. If the side paths are false paths, the side path cannot block the transition at the main path.

The problem here is that if we check for the side paths to be a timing false path or not, then the main path becomes a side path and we need to check whether it is a timing false path or not. This is actually the original problem that we need to solve.



The way around that we are using to get rid of this vicious circle, is to test the side paths using logical false path identification algorithm.

Algorithm B is modified in the final step (checking the ern condition). When we check the nodes with race conditions for the revised side value criteria, the side paths are also checked for the logic falsity. If the side path is a logic false path, it is assumed that the side path is not blocking the transition at the main path.

We call this condition as RSVFPC (revised side value false path condition). If RSVFPC is true, which means side path is a false path, then the main path is an active path, else it is a false path. Figure 8 shows this new algorithm.

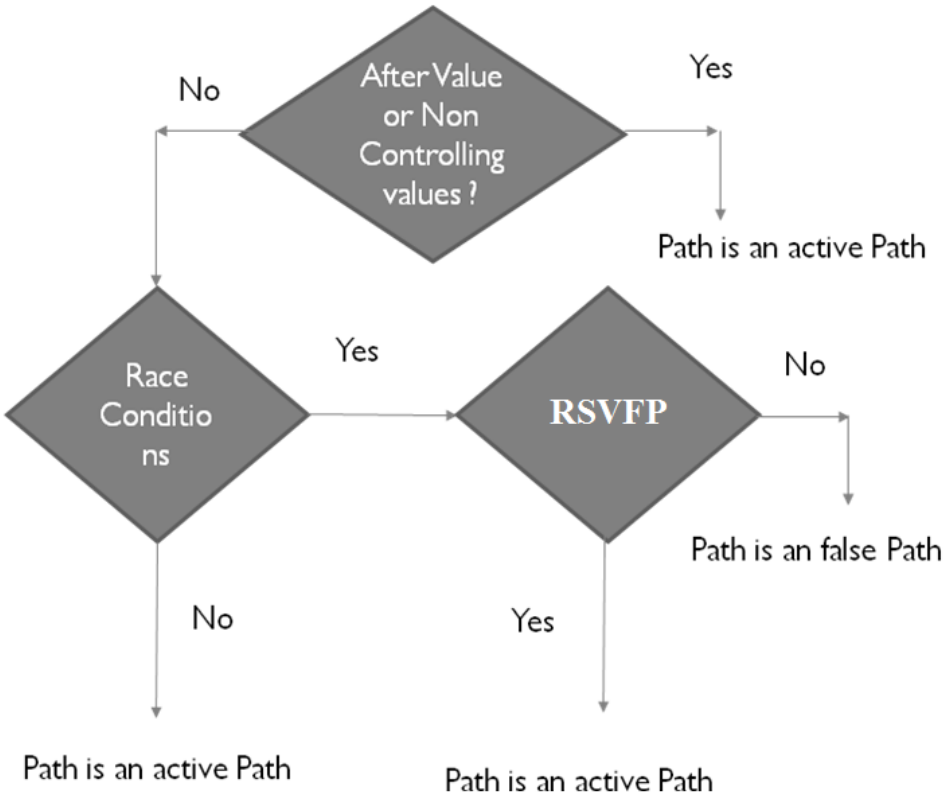


Figure 8 : ALGORITHM C

5. SIMULATION RESULTS

5.1 SIMULATION CONFIGURATION

5.1.1 Delay Models

The delay models are input pins to output delay with rise and fall delays considered separately.

5.1.2 ATPG Tool

Atalanta ATPG tool [8] is used to evaluating the after value, before value and the side value conditions.

5.1.3 Benchmarks

ISCAS-85 benchmark circuits (combinational) are used for the false path analysis. The circuits are synthesized using Synopsys Design Vision and net-list is generated. Nan-Gate Open Cell Library PDK v1.2 (2008) is used for synthesis.

5.2 EXPERIMENT 1

The first experiment is carried out to compare the original algorithm A (logic false path identification) with Algorithm B, which does the timing false path identification.

The table below shows that the total number of false paths goes down, as expected in the timing false path identification.

The timing information helps to reduce the total number of false paths.

The results show about 48% reduction in the total number of Timing false paths.

Benchmarks	Total Paths	ALGORITHM A	fa	fb	ALGORITHM B
		Static False Paths			Timing False Paths
C432	**16444**	**4974**	3468	1508	**3316**
C499	**7583**	**3269**	2057	1239	**1322**
C1355	**5263**	**2008**	1205	803	**737**
C1908	**4491**	**563**	553	10	**327**
C2670	**2588**	**6**	6	0	**0**
C5315	**3154**	**1722**	1531	191	**920**
C7552	**1989**	**1112**	763	349	**599**

Table 1: Comparison of Algorithm A and Algorithm B

5.3 EXPERIMENT 2

The next experiment is done to compare algorithm B & C, which is comparing the effect of considering the side paths to be logically false or not.

The table below shows some improvement as the number further goes down by some amount.

The results show that there is around 15% further reduction in the total number of false paths identified.

Benchmarks	Total Paths	ALGORITHM B	ALGORITHM C
		Timing False Paths	Timing False Paths + RSVFPC
C432	**16444**	**3316**	**3142**
C499	**7583**	**1322**	**1167**
C1355	**5263**	**737**	**623**
C1908	**4491**	**327**	**235**
C2670	**2588**	**0**	**0**
C5315	**3154**	**920**	**792**
C7552	**1989**	**599**	**474**

Table 2: Comparison of Algorithm B and Algorithm C

6. CONCLUSION

Three different algorithms are implemented for false path identification. Logic false path identification algorithm identifies more number of paths compared to other methods. It clearly shows that this method is being over optimistic and can result in to wrong timing information calculations. It is better to be pessimistic by running the circuit at a slower frequency than to estimate a higher frequency of operation and then fail to meet it. The other two algorithms B & C try to reduce the optimism by reducing the number of false paths identified by using the timing information.

The algorithm C uses logic false path identification for the side paths. This should be changed to consider timing false path identification for the side paths as well.

This then boils down to solving the false path identification technique as a single instance problem rather than solving it in a sequential fashion. An example would be a linear programming approach, where in the ea, eb, en conditions could be put as constraints while the objective would be to find if the path is false or not. But since these values are not evaluated as a programming solution, it is difficult to use such techniques for false path identification. One solution would be to integrate timing information with the linear programming approach to solve the false path identification problem.

7. REFERENCES

[1] *On the General False Path Problem in Timing Analysis.* David H.C. Du, Steve H.C. Yen, S. Ghanta. 1989. ACM. pp. 555-560.

[2] *False Timing Path Identification Using ATPG Techniques and Delay Based Information.* Jing Zeng, Magdy Abadir, Jacob Abraham. s.l. : ACM, 2002. Design Automation Conference. pp. 562-565.

[3] *Integrating Functional and Temporal Domains in Logic Design.* P.C. McGeer, R.K. Brayton. s.l. : Kluwer Academic Publishers, 1991.

[4] *A New Approach to the Use of Satisfiability in False Path Detection.* Felipe S. Marques, Renato P. Ribas, Sachin Sapatnekar, André I. Reis. s.l. : ACM, 2005. GLSVLSI'05.

[5] *Path Verification Using Boolean Satisfiability.* Matthias Ringe, Thomas Lindenkreuz, Erich Barke. s.l. : IEEE.

[6] *A quick and inexpensive method to identify false critical paths using ATPG techniques: an experiment with a PowerPCT" Microprocessor.* Jayanta Bhadra, Magdy S. Abadir, Jacob A. Abraham. s.l. : IEEE, 2000. Custom Integrated Circuits Conference. pp. 71-74.

[7] *False-Path Removal Using Delay Fault Simulation.* Marwan A. Gharaybeh, Vishwani D. Agrawal, Michael L. Bushnell. s.l. : IEEE, 1998

[8] *On the Generation of Test Patterns for Combinational Circuits.* Ha, H. K. Lee and D. S. Dep't of Electrical Eng , Virginia Polytechnic Institute and State University. Technical Report No. 12_93.