International Journal of Electronics Engineering, 1(1), 2009, pp. 119-121

A Genetic Implementation of Stuck-at-fault & Delay Fault Identification in Digital Circuits

Dhiraj Sangwan[1*], Rajesh Kumar[2] & Mukesh Kumar[3]

[1]*Electronics & Communication Engg. Deptt, FET, MITS, Lakshmangarh, INDIA*

[2]*Electrical Engg. Deptt, MNIT, Jaipur, INDIA*

[3]*Electronics Science Deptt, Kurukshetra University, Kurukshetra, INDIA*

Abstract: *In this paper the authors present a genetic algorithm (GA) based approach to find the minimal set of Test vectors to find all the possible stuck at faults and delay faults in a digital circuit. A genetic algorithm has been used which search for minimum number of vector set with high detection capability from a large solution space. The possible number of interconnections and hence the number of faults is having an approximately exponential relationship with the number of primary inputs and circuit nodes. All the faults which have been originated because of process defects or circuit delays which may either lead to permanent connection of circuit nodes to power supply rails or can cause the output to be slowly rising or falling. Such faults must be identified before the chip is fully packaged to prevent the malfunctioning of a system.*

Keywords: *Genetic algorithm, Automatic Test pattern generator, Universal Reference Table, Fault Coverage, Combinational circuit.*

1. INTRODUCTION

Digital circuits are made of logic gates. As the complexity of the circuit increases so is the number of logic gates which increases the number of circuit's nodes proportionately. The fault identification method adopted to verify the possible fault occurrences have to be rigorously tested for their reliability [6,11]. A Fault Model is used to replicate the behavior of the physical fault in a software form. When a line is stuck it is called a fault. The fault model is used in digital circuits for post manufacturing testing, not design testing [14]. A stuck-at fault implies that an input or output line of a logic gate is permanently stuck at logic value 1 or 0 (stuck-at-1 or stuck-at-0) [4, 5]. Delay faults just like stuck at faults are also introduced during the manufacturing stage of an Integrated circuit and can create the timing or delay problems at the output nodes. To detect such faults the faulty node must have a transition in logic values and the same transition must be propagated to the output node. A fault is untraceable if no sequence of input vectors produces a fault effect at any primary output of the circuit under test. Otherwise, the fault is detectable [4,11]. A pattern set with 100% stuck-at fault coverage consists of tests to detect every possible stuck-at fault in a circuit. 100% stuck-at fault coverage does not necessarily guarantee high quality, since faults of many other kinds – such as bridging faults, opens faults, and transition (delay) faults – often occur [9,10]. Even though random pattern generation is simple, it is not easy to generate a test for a hard-to-detect fault, since it does not take into account any information about the circuit to be tested [1,7,8].

*Corresponding author: dhirajsangwan@gmail.com

1.1 Genetic Algorithms Overview

Genetic algorithms feature populations of individuals that evolve with the use of the principles of selection, variation and inheritance [3]. One of the ways to implement this idea in computer programs is to represent individuals as strings of binary digits. Each individual is assigned a numerical evaluation of its merit by a fitness function. The fitness function determines how each gene of an individual will be interpreted and, thus, what specific problem the population will evolve to solve. Once all individuals in the population have been evaluated, their fitness values are used for selection [5, 12, 13].

2. PROBLEM FORMULATION

Figure 1 shown below gives an approximation about stuck at faults and their identification method. Input pattern $A = 0$, $B = 1$ detects the presence of stuck at fault at A.

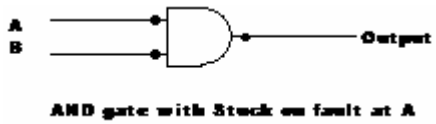


Figure 1: AND Gate with Stuck at 1 Fault at A.

3. GENETIC IMPLEMENTATION

A test circuit (Fig. 2) has been shown below to outline the procedure adopted to implement the algorithm. We try to determine minimum number of test vectors in minimum

number of iterations from the solution space using GA so that all faults are detected. The procedure adopted to detect the delay faults i.e. slow to rise and slow to fall consist of a pair of test vectors [4,12,15]. This test vector set along with detecting the delay faults also detects stuck at faults at the faulty nodes.

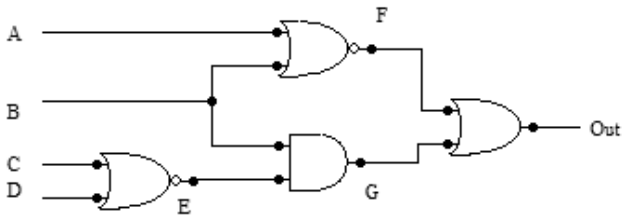


Figure 2: TestCircuit-1 Four Primary Inputs and One Primary Output.

Two delay faults have been shown:

(1) Slow to fall at node F which is detected if input pairs are "0001" & "1101"

(2) Slow to rise at node E which is detected if input pairs are "1101" & "1100"

First the fault free circuit response is to be evaluated which will act as a reference for distinguishing the faulty free behavior from the faulty ones [13]. Implementation is achieved by following the algorithmic steps of Population Initialization, Population selection, Crossover, Mating.

3.1 Population Initialization

The initialization of chromosomes for the delay fault model is similar, as the test patterns are directly represented as chromosomes, however, each test consist of a pair of test patterns, it was therefore necessary to generate chromosome - pairs to directly represent the potential delay test pattern pairs [2].

3.2 Parent Selection

This step is performed to favor the survival of the fittest rule. Basically it is a selection procedure to have a probabilistic based selection of chromosome with higher fitness value which further depends on their fault detection capability [1,2]. Once the states of the circuit nodes have been set by the first chromosome the values are stored in the Universal Reference Table (URT) [11. From a population of individuals, GA demands selection of those individuals for reproduction, which are better than others in detecting faults [4, 9].

3.3 Cross-over and Mutation

The next step performed is to have a generation of new members of the future generations the technique adopted is called as crossover which can explore the combinations of the current mating pool [13].

Test Circuit II: The total number of stuck at faults (stuck at 0 and stuck at 1) is 28 and the total number of delay faults (slow to rise and slow to fall) is 12.

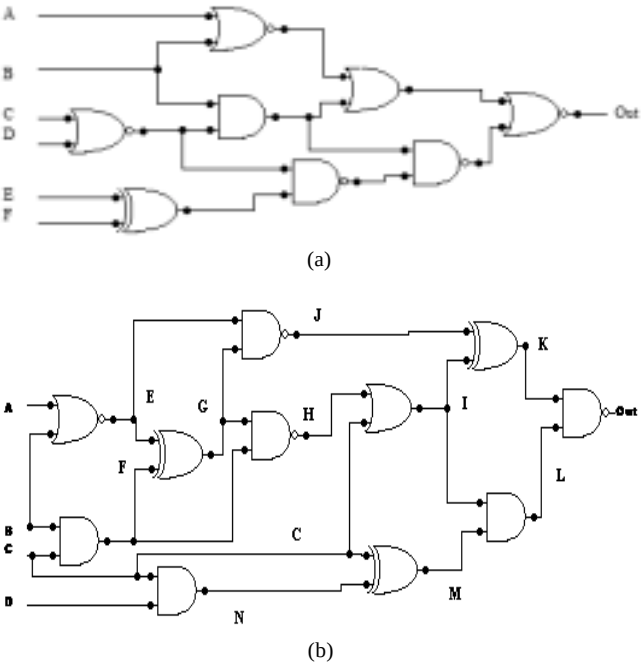


Figure 3: (a) Test Circuit-II, Five Primary Inputs and One Primary Output.
(b) Test Circuit-III, Four Primary Inputs and One Primary Output

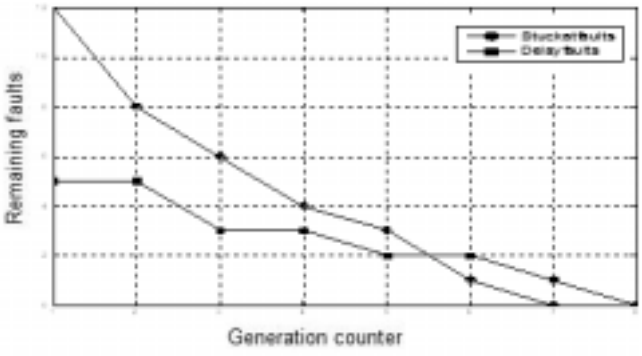
Test Circuit III: The total number of input combinations is 16 but the number of possible stuck at faults increases to 30 due to the 15 possible circuit nodes, which can be stuck at 0 or stuck at 1 respectively. The total number of delay faults both slow to rise and slow to fall in this case is 20.

4. RESULTS AND DISCUSSION

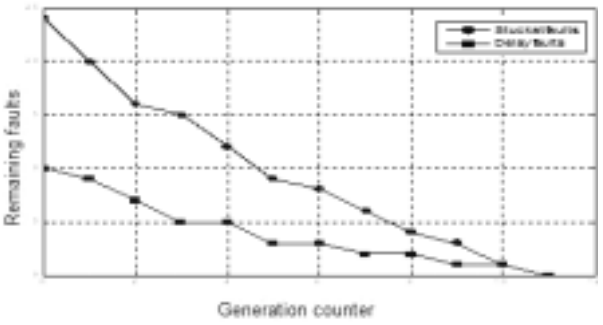
Three test circuits have been used to prove the procedure used for identifying the faults and to show the results obtained. The GAs performance is illustrated with the four input circuit shown in figure 4.

Test Circuit First (TCI)

The GAs performance is illustrated with the four input circuit shown in figure 4(a).



(a)



(b)

Figure 4: (a) Generation Counter Vs Remaining Faults TCI (b) Generation Counter Vs Remaining Faults II

Test Circuit Second (TCII)

The total numbers of stuck at faults detected are 28 and delay fault number is 12. The results for Test circuit II are shown in figure 4 (b)

Test Circuit Third (TCIII)

In this case the possible number of stuck at faults is 30 and delay faults have increased to 20. The results for Test circuit II are shown in fig. 5.

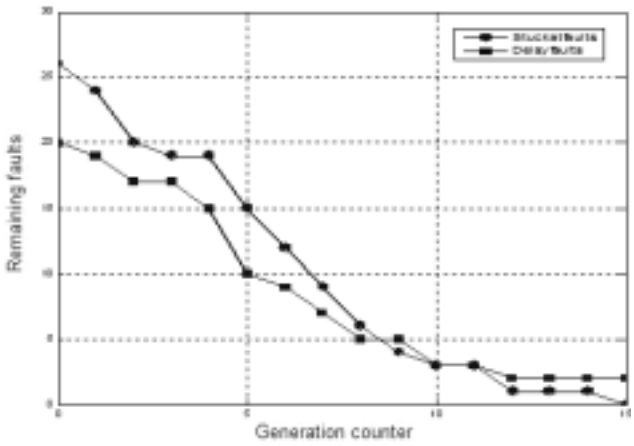


Figure 5: Generation Counter Vs Remaining Faults III

5. CONCLUSION

In this paper a test generation scheme has been presented to detect the presence of stuck at and delay fault identification in digital circuits. The dynamic referencing has been employed with the help of a reference table which updates according to the information of test vectors selected during each run of the GA. The approach can be extended to detect faults in bigger circuits by employing a library or encoding of the logic blocks which have been earlier examined for fault possibilities, this helps in saving of time by not searching for fault occurrences in which results are just alike what they need to be. By using this scheme the test vectors which have detected delay faults are also employed to detect the undetected stuck faults also.

REFERENCES

[1] M. S. Bright and T. Arslan, "A Genetic Framework For The High-Level Optimisation of Low Power VLSI DSP Systems", *IEE Electronics Letters*, **32**, (13), 20th June, 1996, 1150–1151.

[2] M.J. O'Dare and T. Arslan, "Transitional Gate Delay Detection for Combinational Circuits using a Genetic Algorithms", *IEE Electronics Letters*, **32**, (19), 12th (Sept,1996).

[3] Davis "Handbook of Genetic Algorithms," Van Nostrand Reinhold, New York, (1991).

[4] M. J. O'Dare and T. Arslan, "A Genetic Algorithm for Multiple Fault Model Test Generation for Combinational VLSI Circuits", *IEE Conference Publication* No. 446, September 1997.

[5] S. Devadas and K. Keutzer, Validatable Nonrobust, "Delay-Fault Testable Circuits via Logic Synthesis", *IEEE Trans. on Computer-Aided Design*. **11**, (12), (December 1992), 1559–1573.

[6] S. Park and M. R. Mercer, "An Efficient Test Generation System for Combination Logic Circuits," *IEEE Trans. on Computer-Aided Design*. **1**, (7), (July 1992), 926–940.

[7] P. Mazumder and E. M. Rudnick, "Genetic Algorithms for VLSI Design, Layout and Test Automation," *Upper Saddle River*, New Jersey: Prentice Hall PTR, (1999).

[8] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning," *Reading*, Massachusetts: Addison-Wesley, (1989).

[9] J. M. Rabaey, "Digital Integrated Circuits: A Design Perspective," Prentice Hall of India Pvt, Ltd., Inc. (2001).

[10] M. Mitchell, "An Introduction to Genetic Algorithms," MIT Press, Cambridge, MA, (1988).

[11] V. Agrawal and A. Fung, "Multiple Fault Testing of Large Circuits by Single Fault Test Sets," *IEEE Trans. on Computers*, **C-30**, (November 1981), 855–865.

[12]. M.J. O'Dare and T. Arslan, "Generating Test Patterns for VLSI Circuits using a Genetic Algorithm", *IEE Electronics Letters*, **30**, (10), 12th May, 1994.

[13] Rajesh Kumar, Mukesh Kumar, Dhiraj Sangwan "A Genetic Implementation of Fault Identification in VLSI Circuits" *Proceedings of 2nd National Conference On Power Electronics & Intelligent Control*, MNIT Jaipur, Rajasthan, (September 13-14, 2008), 103–108.

[14] Rajesh Kumar, Mukesh Kumar, Dhiraj Sangwan "Multiple Stuck at Fault Identification of Combinational Circuits using Genetic Algorithm", *Proceedings of 2nd International Conference On Soft Computing*, IET Alwar, Alwar, Rajasthan, (November 8-10, 2008), 91–97.

[15] Tekumalla, R.C. Menon, P.R. "Identification of Primitive Faults in Combinational and Sequential Circuits" *IEEE Transactions* **20**, (12), (Dec 2001), 1426–1442.