International Journal of Electronics Engineering, 1(2), 2009, pp. 197-201

A Genetic Approach to Detect Bridging Faults, Transitional Delay Faults and Stuck at Faults in LSI 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: *A novel algorithm based on Genetic approach for diagnosing Bridging faults, Transitional Delay faults and stuck at faults in LSI circuits is presented. The method of devising a universal set for detecting Bridging faults in testable circuits is investigated. We outline a method that utilizes the information from the stuck-at fault model to accurately diagnose the bridging faults that affect two lines. The proposed method uses the observation that the bridging fault response matches the stuck at fault responses on the shorted lines for the failing test vectors. By using the information from the pass test vectors a further reduction in size of test set is possible. The algorithmic approach allows us to have a random search of test vectors without being caught in a local minima or maxima. The fitness scheme allows us to have a selection of test vectors with high fault coverage and with large fault detection scores.*

Keywords: *Bridging Faults, Genetic Algorithm, Automatic Test Pattern Generator, LSI, Transitional Delay Faults, Fault Coverage, Combinational Circuit.*

1. INTRODUCTION

The systems in which complex digital integrated circuits are used depend heavily on their correct and reliable operation. Before an system or circuit is to be sent to its customer it has to be tested rigorously for its correctness by subjecting it to different test vectors [1, 2]. It means that the test generators must be able to generate all possible test sequences in fully functioning environment. Physical defects in logic circuits are modeled by using logical faults. These tests are performed after the design has been fully fabricated [3, 4]. The stuck at fault model applies to gate level circuits, or a block of a sequential circuit which can be separated from the storage elements. The stuck at fault model assumes that only one input on one gate will be faulty at a time, assuming that if more are faulty, a test that can detect any single fault, should easily find multiple faults. Delay faults on the other hand can be due to slowly responding output nodes, for it to be tested a test vector pair would be used the first vector would initialize or decide the logical value at faulty node and the second vector would cause a transition at that node, the fault is said to be detected if the transition would have been propagated to the output node also. The model assumes that only fault at a time is present in a circuit [5, 6, 7]. The Bridge faults can be somehow compared with stuck at faults since in both the logic node is permanently tied to a logic value. However in case of Bridge faults the final value is depending on either of the two voltages if different values are present on the connected (shorted) conductor lines. Fault simulation consists of logic simulation in the presence of a fault. By comparing the simulation results for the faulty circuit and the good circuit, it is possible to determine whether the fault is detected by the given test [8, 9, 10]. In this paper, a genetic based approach has been adopted to detect the possible fault occurrences. Depending on whether or not multiple fault simulation results in primary output values that agree with the observed values, faults are added to or removed from a set of suspected faults. Fault identification process is affected by the addition and place of occurrence of faults [11, 12].

*Corresponding Author: dhirajsangwan@gmail.com

1.1. Genetic Algorithms Overview

Genetic algorithms mimic the evolution by natural selection. It is based on Darwin theory of Survival of the fittest. The basic idea of genetic algorithms is very simple. One of the ways to implement this idea in computer programs is to represent individuals as strings of binary digits. Each bit in the string represents one gene. Individuals with low fitness get eliminated and the strongest get selected. By transforming the previous set of individuals to a new one, the algorithm generates a new set of individuals that have better fitness than the previous set of individuals. The offspring replace the old population and a generation is complete. This process is repeated until certain criteria are met [13]

2. PROBLEM FORMULATION

As the complexity of the VLSI circuit increases so is the number of fault possibilities. The different types of faults have their different origin sources like stuck at faults are mainly due to process defects like unwanted short to power rails. Delay faults are mainly due to slowly responding logic gates due to resistive or capacitive effects and Bridging faults have origin roots lie deep in mask misalignment during metallization step of fabrication [14, 15].

2.1. Stuck at Fault Model

Stuck at faults can be stuck-on faults (stuck-at 1) or stuck-open faults (stuck-at 0). Hence, physical testability of VLSI circuits for stuck-at faults is very important [14]. Figure 1, shown below gives an approximation about stuck at faults and their identification method.

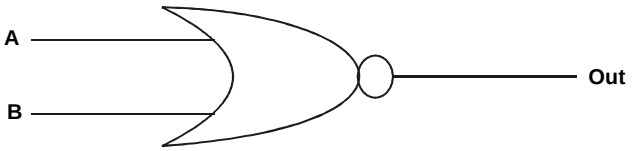

Figure 1: NOR Gate with Stuck at Fault at B

The presence of a fault at any of the input depends on the possibility of its connection with a supply rail. A test vector {1, 1} that is A = 1, B = 1 gives output 0 but the output remains unchanged even after the test vector input is {1, 0} as input B has stuck at one fault. Now, in this case {1, 1}, {1, 0}, {0, 1} are unable to determine the fault whereas {0, 0} is the only test vector which determines the fault. Hence, stuck-at faults can be detected only by specific inputs. Hence, GA provides an optimal solution of test vectors, which can detect all stuck-at faults based on evolutionary algorithm and selection of fitter test vectors [16].

2.2. Delay Fault Model

In the Transitional delay fault model the delay defect is considered to be a slow-to-rise or a slow-to-fall fault condition, occurring at a single node within the circuit due to a delay defect on the gate driving that node. Due to the delay incurring in having a transition at the output node the output may be erroneous. A valid test for delay defects therefore comprises of a pair of test patterns {T1, T2}, applied in ordered sequence. T1 being the test pattern that initialises the circuit and T2 the test pattern that causes the required transition at the test node within the circuit. In the figure shown below a test-pattern pair is applied, pattern {100} is first applied in order to initialise the all circuit nodes, then the second vector {000} is applied to cause any necessary transitions [17, 18].

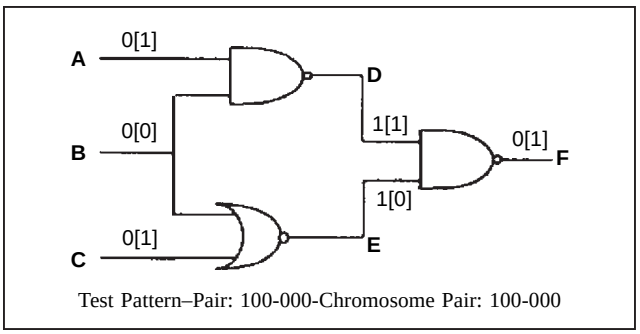

Figure 2: Test-Pattern Pair Applied to Test Circuit

2.3. Bridge Fault Model

When a short is between two-signal lines w1 & w2 both lines have the same voltage level. This defect usually creates a new logic function. The logical fault is called a bridging fault. Depending on the technology one can distinguish OR bridging faults (logic 1 overrides logic 0) from AND bridging faults (logic 0 overrides logic 1). Figure, below presents a bridging fault between signal lines w1 and w2; assuming that logic 1 overrides logic 0. Both w1 and w2 have the same value. Let a0 and b0 be the logic values of w1 and w2 if the circuit was fault free, and af and bf the logic values when the OR bridging fault is present. Due to the wired OR af = bf = a0 + b0. The other Figure, shows an AND bridging fault. Then af = bf = a0.b0.

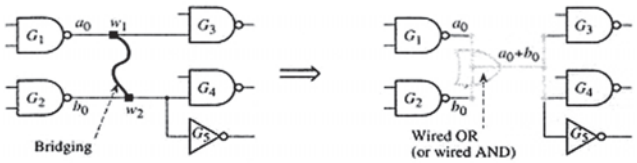

Figure 3: Combinational Fault: Non Feedback Bridging Fault

3. GENETIC IMPLEMENTATION

An objective function is to be formed which decides the probability of selection of the chromosomes on the basis of their fitness. Before the algorithm is to be coded an encoding of the problem is performed. This problem is dealt effectively if a fitness scheme has been developed which effectively isolates those test vectors with greater fault detection ability from those with lesser [20].

3.1. Population Initialization

Initially a random set of population elements are generated called as chromosomes or members of first population. An evaluation step is carried out to monitor that the obtained output whether differs from the expected or not, if it differs that indicates the fault presence at that circuit node.

3.2. Parent Selection

The technique adopted by the system is the roulette wheel method of parent selection, this method ensures that the

fitness values assigned to the test patterns, are proportional to their probability of being selected as parents [4, 9].

3.3. Cross-over and Mutation

Each parent selected will undergo crossover. The method used is referred to as Two point crossover; the two parents selected for crossover exchange information residing between two randomly generated points within the binary string [13]. In order to introduce the newness in the chromosomes mutation is performed which alters the allele.

3.4. Universal Reference Table (URT)

In each run of the algorithm the fittest member is determined and is entered into the test set, only if it improves the fault cover currently achieved by the test set. Once the above procedure is completed, a new population has been created. Before this is attempted a Universal Reference table is established; this table holds information relating to the test patterns that have been chosen for entry into the test set. The URT also plays an important role in providing the information regarding undetected faults in the circuit.

4. RESULTS AND DISCUSSION

The technique has been successfully applied to detect faults in three test circuits. The Bridge faults and stuck faults have some characteristics in common such that both have their final values fixed which do not at all depend on the stimulus applied.

Test Circuit First (TCI): 7485: 4 Bit Magnitude Comparator

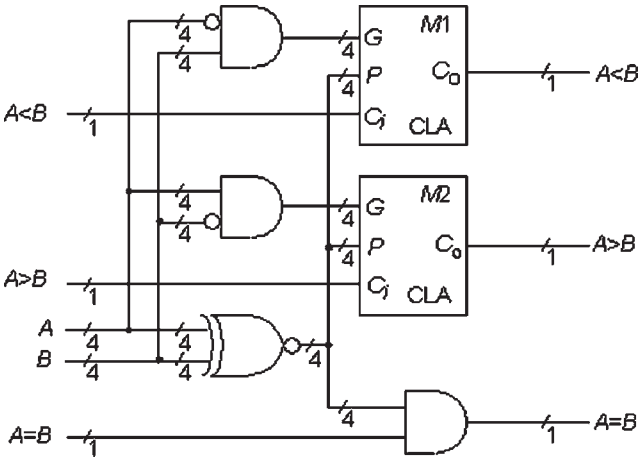

Figure 4: Four Bit Magnitude Comparator

Statistics: 11 inputs; 3 outputs; 33 gates

Function: The 74L85 magnitude comparator can be functionally modeled as above. The GAs performance is illustrated with the 11 inputs circuit shown in figure above. On this occasion the GA has produced test patterns that provide 100% fault coverage for our circuit, this information is represented below.

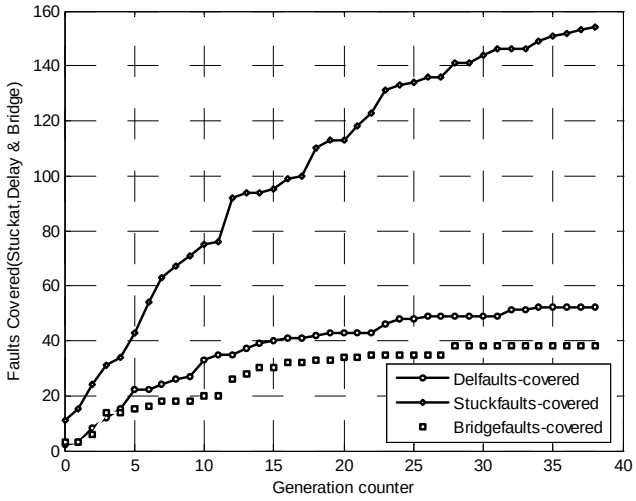

Figure 5: Generation Counter Vs Remaining Faults TCI

The above Figure, shows how the faults remain as the counter progresses during each iteration of genetic algorithm. In total it needs thirty-eight iterations to cover all the 154 stuck at faults and 52 delay faults and 38 bridge faults.

Test Circuit Second (TCII) : 74182 Carry Look-Ahead Circuit(CLA)

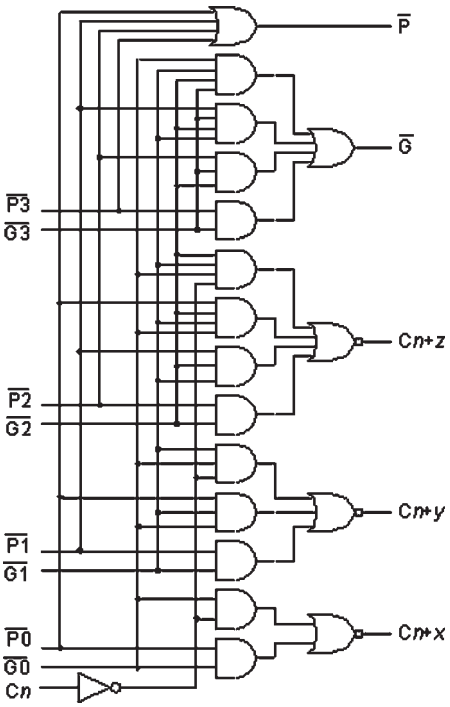

Figure 6: Carry Look Ahead Generator

Statistics: 9 inputs; 4 output; 19 gates

Function: Given carry-in (Cn), generate (G) and propagate (P) signals, the circuit produces three carry out signals, plus two P and G signals used to cascade into another CLA block.

The total numbers of stuck at faults detected successfully are 76 and delay fault number is 28 and bridge faults are 29. In total it requires 23 iterations to cover all the faults occurred.

The results for Test circuit II are shown in Figure 7.

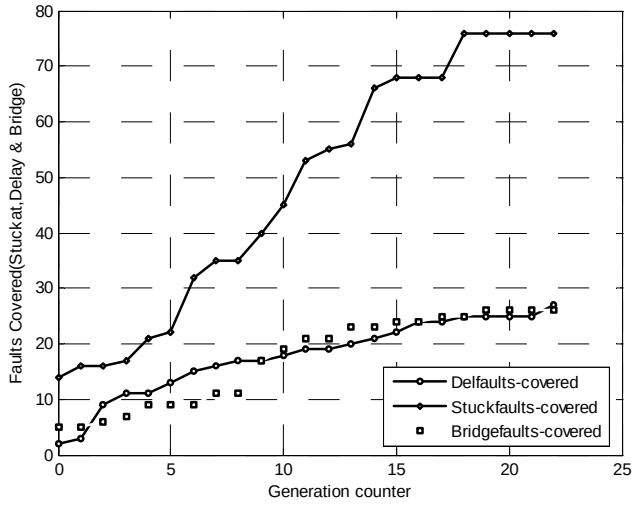

Figure 7: Generation Counter Vs Faults Covered II

Test Circuit Third (TCIII) : 74283 Fast Adder Circuit

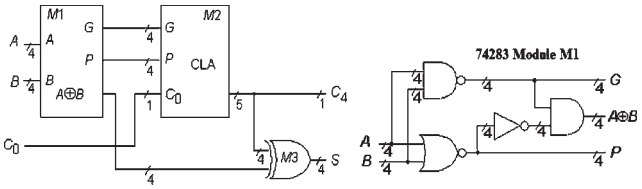

Figure 8: Four-Bit Adder

Statistics: 9 inputs; 5 outputs; 36 gates.

Function: The module M1produces the generate, propagate, and XOR functions. The module M2 is similar to the 74182 (CLA) The XOR word gate M3 produces the sum function. The third test circuit used though has number of inputs equal to 4 but since the number of intermediate nodes are 10 so the chances of physical defects are much more in this case because stuck at defects mainly occur due to the faulty metallization steps which result in either permanent connection to supply 1 or supply 0. The total numbers of stuck at faults detected successfully are 172 and delay fault number is 65 and bridge faults are 47. As the number of inputs increases, so is the exponential increase in the possible number of faults occurred. In total it requires 23 iterations to cover all the faults occurred.

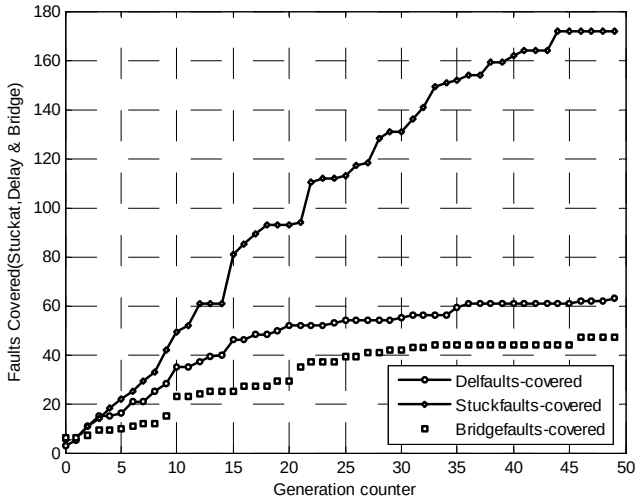

Figure 9: Generation Counter Vs Faults Covered III

The population size with which we have started has an important effect on the number of iterations occurred to find the total faults. Figure 10, shown below follows the same argument.

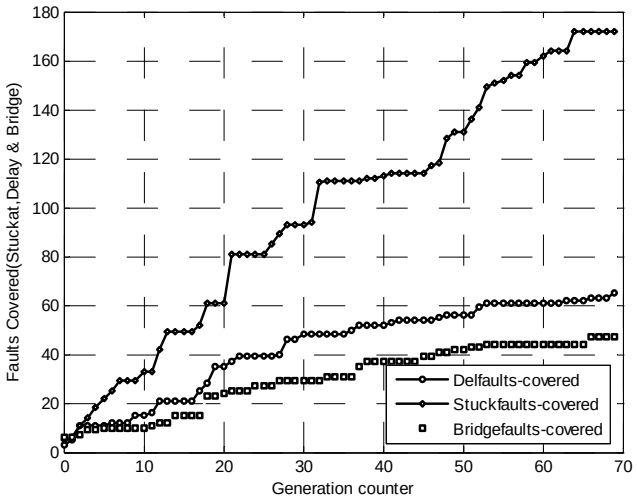

Figure10: Effect of Reduction in Population Size on Generation Counter

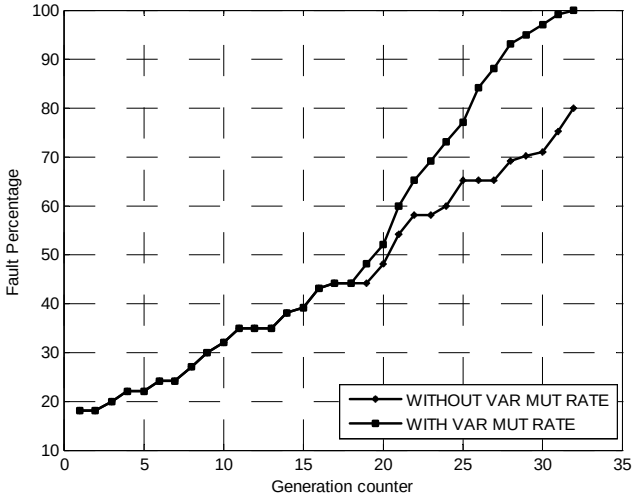

Figure 11: Effect of Changing and Not Changing Mutation Rate of Fault Detection

Along with finding all the faults the following detail has also been obtained that instead of having a fixed Pc and Pm it would be more beneficial especially in denser circuits to have a variable rate if no significant fault detections are obtained.

Figure 11, describes clearly that in lesser generations higher fault coverage can be achieved by using varying mutation rate is used instead of fixed mutation rate if no significant fault coverage improvement is achieved in 3 successive generations.

5. CONCLUSION

Genetic algorithm provides a probabilistic based selection and random search of points in a large solution space. The random search of test vectors by using GA helps in finding the transitional delay faults by having a random generation and recombination or population members. The technique adopted has been successfully applied to all the three Test Circuits and faults have been detected. It has been found that instead of concentrating on fixed crossover and mutation rates if variable assignment is used then the detection rate can be considerably increased.

REFERENCES

[1] B. Chess, D. B. Lavo, F. J. Ferguson, and T. Larrabee, "Diagnosis of Realistic Bridging Faults with Single Stuck-at Information," in *Proc. Int. Conf. Computer-Aided Design*, (1995), 185–192.

[2] D. B. Lavo, B. Chess, T. Larrabee, F. J. Ferguson, J. Saxena, and K. M. Butler, "Bridging Fault Diagnosis in the Absence of Physical Information," in *Proc. Int. Test Conf.*, (1997), 887–893.

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

[4] Bechir Ayari and Bozena Kaminska, "A New Dynamic Test Vector Compaction for Automatic Test Pattern Generation" *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **13**, (3), (1994), 353-358.

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

[6] 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, (1997).

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

[8] J. M. Rabaey, "Digital Integrated Circuits: A Design Perspective," Prentice Hall of India Private Limited, Inc., (2001).

[9] M. Schulz, E. Trishler, and T. Sarfert, "SOCRATES: A Highly Efficient Automatic Test Pattern Generation System", *IEEE Trans. on Computer Aided Design*, **7**, (1), (1988), 126-137.

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

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

[12] J. H. Aylor, J. P. Cohoon, E. L. Feldhousen and B. W. Johnson, "A Genetic Algorithm for Compacting Randomly Generated Test Sets", *International Journal of Computer Aided VLSI Design*, **3**, (1991), 259-272.

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

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

[15] Puneet Gupta and Michael S. Hsiao, "ALAPTF: A New Transition Fault Model and the Atpg Algorithm", Proceedings of International Test Conference (ITC'04), IEEE Computer Society, (2004), 1053-1060.

[16] 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), (1996), 1150-1151.

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

[18] 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, (2008), 103-108.

[19] 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, (2008), 91-97.

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