

A Genetic Algorithm for Multiple Fault Model Test Generation for Combinational VLSI Circuits

T. Arslan and M. J. O'Dare

*School of Engineering
Cardiff University of Wales
Cardiff CF2 1XH*

Abstract

The authors present a genetic algorithm (GA) for the automatic generation of test vector-pairs for the detection of both delay and single stuck-at-fault models in combinational digital VLSI circuits. The GA proves effective in searching the highly complex problem space, which is significantly larger than that with single stuck-at-faults only. The paper describes the GA and results obtained for the ISCAS 1985 benchmark circuits. The initial concept of transitional test pattern generation using a basic GA was introduced by the authors in [1].

1 Introduction

Digital systems produced today are extremely intricate and are ever increasing in complexity, they are required for use in a widening range of domestic and industrial applications [2]. The systems in which complex digital Integrated Circuits are used may be medical, military or flight control systems which depend heavily on their correct and reliable operation, in order to ensure reliability of these circuits it is necessary to test their performance to identify any defects prior to using them in a fully operational environment.

In order to generate a test for a fault condition it is necessary to model the fault condition and simulate the circuit operation with the modelled fault condition present. By using a fault model the physical defect can be represented by altering the characteristics of a circuit to enable it to perform as if a fault condition were present. This paper considers the generation of tests that detect gross delay faults in combinational digital circuits [3]. The basic elements of such circuits are gates that produce a steady state output denoted by 0 or 1, which is purely dependant on the combination of its input values. The gates within a combinational digital circuit considered in this paper have the primitive functions NAND, AND, NOR, OR and NOT. The interconnection of the gates in a combinational circuit have no feedback loops, and therefore have outputs that are totally independent of their previous values. The most common fault model is the Single Stuck-at-fault model [3], which is simply used to represent an output or input node of a gate permanently held at the logic state 0 (Stuck-at-zero)

or logic state 1 (Stuck-at-one). Test pattern generation for single-stuck-at faults was presented by the authors in [3], the stuck-at fault model however cannot be used to model delay defects, as the actual size of the defect needs to be considered [3].

2 The Gate Delay Fault Model

In the gate 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. A delay defect of this order may cause an error at the primary output of the circuit, figure 1 is used as a simple example of this fault condition. A series of connected elements in a combinational circuit are shown, at time t_0 the circuit is initialised by applying a signal (logic 0) to its primary inputs; in this example all nodes will reflect the logical state of the input. At time t_1 a second signal (logic 1) is applied to the primary input of the circuit, and at time t_2 the state of the primary output is observed. Consider $t_0, t_1, t_2 \dots t_n$ to be the active edges of a system clock, and $t_n - t_{n-1}$ to be the clock interval τ . The clock interval has a duration greater than the sum of gate delays, i.e. if each gate A-D has a gate delay Δ , then $\tau > 4\Delta$ for the example circuit shown, therefore, with a circuit having g gates in its longest path (from primary input to primary output), then $\tau > g\Delta$.

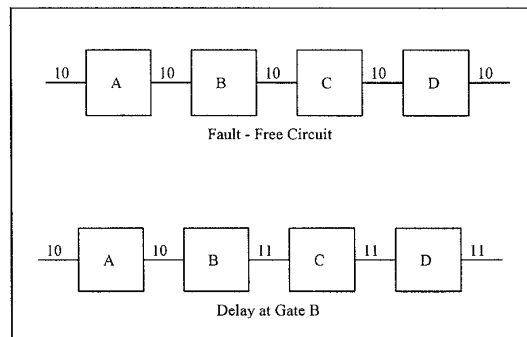


Figure 1: Delay model example.

In this example if a delay defect was present at the output of gate B causing a slow to rise fault condition, it's effect will be observed at the primary output of the circuit. This statement is only true of course if the delay defect size together with the sum of the delay sizes for each gate is greater than the system clock interval, however, if the delay defect size is greater than the system clock interval it will be detected if the fault condition is propagated to an observable primary output [3]. A delay fault with these characteristics is referred to as a gross delay defect, which is the condition considered by the authors in this paper, another assumption of the model used is that there is only one fault condition present at any one time, the model is therefore referred to as non-robust, Park *et al* [3].

In order to test for delay defects it is necessary to initialise the state of all nodes within the circuit, before a test pattern may be applied to the primary inputs of the circuit. A valid test for delay defects therefore comprises of a pair of test patterns $\langle P1, P2 \rangle$, applied in ordered sequence. P1 being the test pattern that initialises the circuit and P2 the test pattern that causes the required transition at the test node within the circuit. The problem is considerably more complex than that of stuck-at-fault testing, with an increase in the search space from 2^n to 2^{2n} for an n input circuit, this problem is made even more complex by the fact that the test patterns need to be applied in ordered sequence. GAs are suited to the NP-completeness of test pattern generation having already proved successful for single-stuck-at fault detection in both combinational and sequential circuits [4,5,6].

3 Genetic Implementation

Genetic algorithms (GAs) have grown in popularity following Holland's work in the field, Holland [7]. The technique adopts the mechanics of the biological genetic process, using binary strings as a substitute for *chromosomes*. In nature the characteristics of the living organism are encoded into chromosomes. GAs have been applied successfully to other areas of VLSI design, for example channel routing [8] and cell placement [9]. Aylor *et al* [10] also employ GAs with a covering heuristic to reduce the size of test sets, that were produced by a random test pattern generation system.

3.1 Population Initialisation

The first stage of the GA is to initialise a population of chromosomes. In the single-stuck-at fault model the population of chromosomes were a direct representation of the test patterns applied to the test circuit, the length of each chromosome being equal to the number of inputs to the circuit. The initialisation 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.

The example circuit of Figure 2 is used to illustrate a test - pattern pair applied to a test circuit, pattern 100 is first applied in order to initialise the all circuit nodes, then the second vector is applied to cause any necessary transitions. The diagram also shows the direct chromosomal representation of the test pattern - pair applied.

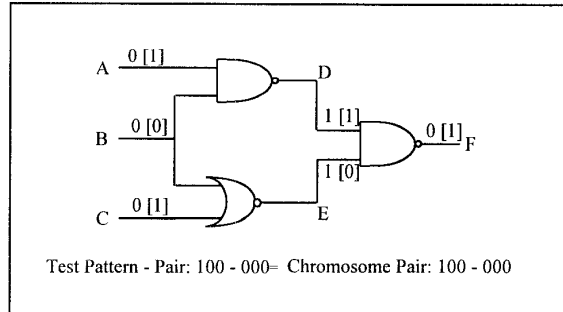


Figure 2: Test - pattern pair applied to test circuit.

3.2 Fitness Evaluation Function

To evaluate the efficiency of the chromosome - pairs as test patterns they are applied in an ordered sequence to the primary inputs of a compiled circuit model. Once the states of the circuit nodes have been set by the first chromosome the values are stored in the Global Record Table (GRT) [6], and the second chromosome is then applied to the test circuit's primary inputs. A comparison of the logical state for each node is made with the value stored in the GRT for the first chromosome, a change in state of 0 to 1 is recorded as a test for a slow-to-fall fault and a 1 to 0 is recorded as a test for a slow to rise fault condition. In order to validate these tests a full fault simulation is required to prove that the potential fault condition would be propagated to an observable primary output. If during the fault simulation the chromosome - pair detects a fault condition by causing the required transition and propagating it to an observable output, a fitness value is assigned to the chromosome - pair reflecting it's efficiency as a delay test, fitness is a biologically derived term used to measure the proficiency of individuals in the quest for producing offspring that have a greater ability to solve a given problem than their parents. The fittest chromosome - pair in the first generation is the first delay test entry into the test set, the test pattern - pair together with it's associated fault coverage is recorded in the GRT, the test pattern - pair may also be a valid test for certain single stuck-at-faults within the circuit, this information is also recorded and updated within the GRT. for subsequent delay test entries into the test set the GRT is updated with the new

fault coverage for the test circuit, a test pattern - pair is only entered into the test set if it improves the overall coverage of the circuit, this procedure ensures that duplicates are not entered into the set and that the fault coverage for the circuit always improves with each entry.

The fitness evaluation function employed by the delay test system is a function of circuit size and complexity, involving a relationship between the number of primary inputs and gates within the circuit being tested. If a chromosome - pair detects delay fault condition that is previously undetected (Pu) a *score* is assigned to the fitness value for that pair:

$$score = 2(\sqrt{G} + I). \quad (1)$$

Where G represents the number of gates in the circuit and I represents the number of inputs to the circuit. The fitness evaluation function, F is represented by:

$$F = \sum_{j=1}^k Pu_j (score) + \sum_{j=1}^k Pd_j. \quad (2)$$

Where k is the number of possible delay faults present in the circuit, Pu and Pd (previously detected fault condition) are both random variables 0 or 1 based on the probability of distribution resulting from fault simulation of a chromosome - pair.

3.3 Parent Selection

Having determined fitness values for the initial population, it is necessary to select parents to engage in crossover. 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 chances of being selected as parents, promoting the survival of the fittest concept [12].

3.4 Crossover and mutation

Two - point crossover is employed with each chromosome of the chromosome - pair having a 50% chance of selection, the two parents selected for crossover exchange information lying between two randomly generated points within the binary string. The resulting *offspring* are re-introduced into the population to create a new chromosome - pair that have inherited characteristics from their *parents*, this procedure is repeated until a whole new population of chromosomes have been created.

The next genetic operator to be applied to the population is mutation, employed to introduce diversity into the population. The delay test system uses a mutation function with a bit alteration probability of 1%, however, an *adaptive* mutation function is also employed to introduce further diversity, by increasing the mutation

rate when the GRT records no improvement in fault coverage over three successive generations of the GA. The mutation rate defaults to 1% after a further generation of the GA.

4 Global Record Table

As well as holding the test pattern - pairs and their relative fault coverage the GRT also plays another important function, providing the delay fault simulation function with information necessary to the fitness allocation, by informing the simulation function of undetected delay faults existing in the circuit. The fitness scoring function uses this information to assign a fitness value to chromosome - pairs that are candidates for entry into the test set.

The Fitness and fault simulation function consult the GRT before the scoring function assigns a fitness value to a chromosome - pair. The fittest chromosome - pair is then entered as a valid delay test if the overall fault coverage of the circuit is improved, the GRT is then updated with the new fault coverage and the chromosome - pair is recorded as a new test pattern - pair in the GRT's test set.

5 Results and Discussion

The circuit in figure 3 is an example illustrating the fault simulation procedure adopted by the system, slow-to-fall and slow-to-rise fault conditions are shown at fault sites A and B respectively.

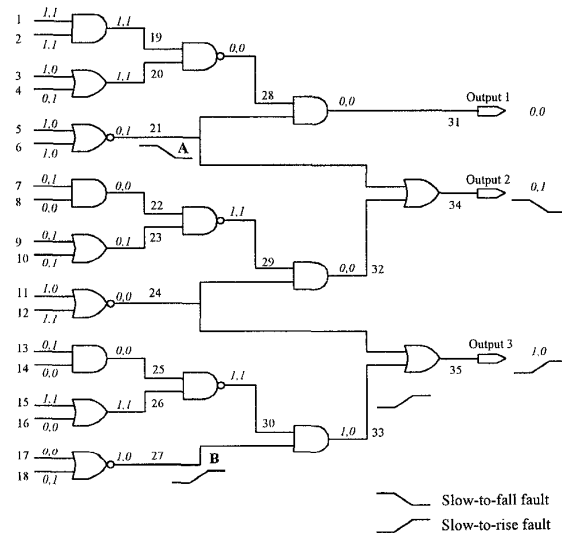


Figure 3: An Example Circuit.

Both of these fault conditions are propagated to an observable output, as well as detecting gross delay faults at the specified fault sites the test vector-pair also detect a single stuck-at-one fault condition at site A , and a single

stuck-at-zero fault condition at site *B*. For the example circuit shown a test set that detects 100% of both gross delay and single stuck-at-fault conditions may be produced in less than 1 second on a Sun Sparc 5 workstation.

Pop Size	Vectors	Generations
10	36	476
20	28	386
30	24	356
40	23	234
50	22	172
100	15	16
200	15	16
400	14	14
1000	14	13

Table 1: population and vector size vs. the number of generations

The compactness of the test set produced is mainly dependant on the population size of the GA, the results for several population sizes are represented in Table 1, the second column of the table indicates the number of test vector-pairs generated for 100% gross delay and single stuck-at-fault coverage. The last column in the table indicates the number of generations taken by the system to reach 100% fault coverage. The results achieved for population sizes up to 200 are shown in figure 4. The graph illustrates the relationship between the population size used in the genetic delay system and the number of generations required to reach a delay fault coverage of 100%, for a population size less than 100 the generations required to reach 100% fault coverage often exceeded 35, but for a population size greater than 100 a fault coverage of 100% was generally achieved in less than 20 generations.

The delay test system has been successfully applied to a number of combinational test circuits and to the ISCAS 1985 combinational benchmark circuits [11], where it produced favourable results when compared to result presented by other researchers, detecting not only delay faults but producing compact test sets that detected a high number of single stuck-at-fault conditions. The results presented in this paper however, do not concentrate solely on the performance of the delay system, but represent also represent the characteristics of the GA in it's application to the delay test problem.

Circuit	Test Set Size	Coverage
C432	28	99.23
C499	50	100
C880	30	100
C1355	98	98.55
C1908	93	99.07
C5315	101	99.94
C6288	31	99.31
C7552	108	98.17

Table 2: Results achieved for the ISCAS Benchmark Circuits.

The results shown in Table 2 are a sample of some of the results achieved for a range of ISCAS benchmark circuits, the fault coverage is higher in most of the considered circuits when compared to results for single stuck-at fault coverage, presented in [12]. Furthermore, the genetic delay system by comparison often reduces the test set size even further than the test set compaction system presented in [12].

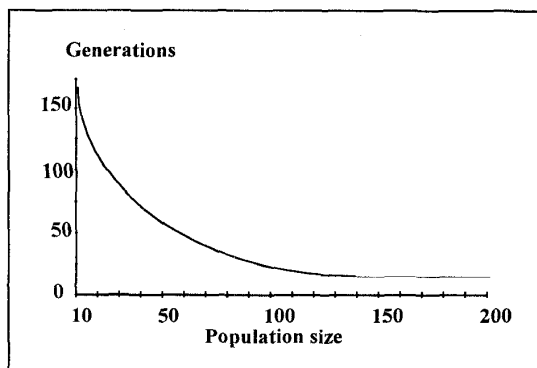


Figure 4: Graph to show population size against the number of generations of the GA.

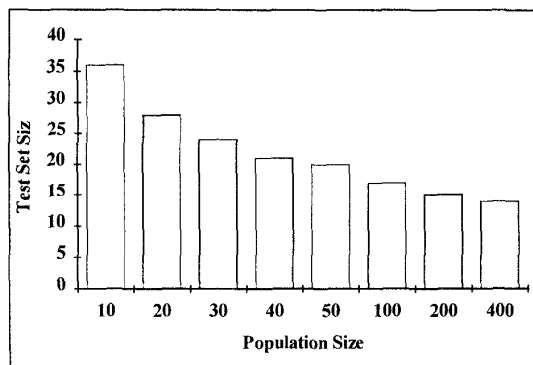


Figure 5: Graph to show population size against the number of test pattern - pairs in the test set.

The graph illustrated in figure 5 represents the relationship between the number of test pattern - pairs in the test set, required to detect 100% of the circuits delay faults and the size of the population used by the genetic system to generate that test set. The maximum number for the population size shown on the graph is 400, beyond this there was no notable improvement in the reduction of the test set size.

6 Conclusions

The Genetically based delay test pattern generation system presented has proved successful in this complex search problem, the results achieved by the system clearly indicate its effectiveness as a fault detection tool for the detection of both gross delay defects and single stuck-at-faults, whilst producing a compact and efficient test set for combinational test circuits. The GRT was instrumental in guiding the GAs search for optimal solutions in an otherwise complex search space.

7 References

1. M.J. O'Dare and T. Arslan, Transitional gate delay detection for combinational circuits using a genetic algorithm, IEE Electronics Letters, Vol 32, No. 19, pp. 1748-1749, September 1996.
2. A. Breuer and A. D. Friedman, Diagnosis and reliable design of digital systems, Computer Science Press Inc. 1977.
3. S. Park and M. R. Mercer, An Efficient Test Generation System for Combination Logic Circuits, IEEE Trans. on Computer-Aided Design. Vol 11, No7, July 1992, pp 926-940.
4. M.J. O'Dare and T. Arslan, Generating test patterns for VLSI circuits using a genetic algorithm, IEE Electronics Letters, Vol 30, No. 10, pp. 778-779, May 1994.
5. S. Devadas and K. Keutzer, Validatable Nonrobust Delay-Fault Testable Circuits Via Logic Synthesis", IEEE Trans. on Computer-Aided Design. Vol 11, No12, December 1992. pp 1559-1573.
6. M.J. O'Dare and T. Arslan, Hierarchical Test Pattern Generation Using A Genetic Algorithm With A Dynamic Global Reference Table, First IEE/IEEE International Conference on - Genetic Algorithms in Engineering Systems: Innovations and Applications. No. 414, 12-14 September 1995. pp. 517-523.
7. J. H. Holland, Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor, 1975.
8. B. Buttitta, P. Orlando, F. Sorbello and G. Vassallo, Monreale: A New Genetic Algorithm For The Solution Of The Channel Routing Problem, IEEE proceedings, CH3001-5, 1991, pp 462-466.
9. U. Hedge and B. Ashmore, A Feasability Study of Genetic Placement, Texas Instruments Technical Journal, Vol. 9, No. 6, Nov-Dec 1992, pp. 72-82.
10. 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, pp. 259-272.
11. B. Brglez and H Fujiwara, A Neutral Netlist of Combinational Benchmark Designs, Proc. Int. Symp. Circuits and Systems, June 1995.
12. B. Ayari, and B. Kaminska, A New Dynamic Test Vector Compaction for Automatic Test Pattern Generation, IEEE Trans. Computer-Aided Design, vol, 13, pp. 353-358, March 1994.