IARJSET

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

ISO 3297:2007 Certified

Vol. 4, Issue 10, October 2017

A new seed finding algorithm for testing VLSI circuits

V.M.Thoulath Begam

Professor, Department of ECE, A.R. College of Engineering & Technology, Tirunelveli, India

Abstract: In this paper a new algorithm is described which is the combination of GA and PSO. They have been used to find the best seeds for CUTs. Using GA, first the initial random patterns are generated and high fitness test patterns are stored. Then PSO is applied on these patterns to obtain final best seeds. The effectiveness of this approach is demonstrated on some sequential and combinational benchmark circuits. The results proved that this algorithm finds high fitness seeds and also reduces testing time.

Keywords: GA, PSO, seed finding, Testing.

I. INTRODUCTION

Nowadays digital systems are extremely complex in nature. The increase in density and complexity of circuits cause a very high demand on the reliability of such circuits. Different Automatic test pattern generators are used for test pattern generation. The most common faults found in VLSI circuits are stuck-at faults. These faults are electrical failures due to physical mechanism. Detection of faults requires an optimum number of vectors to be selected from all possible combinations. Test time and fault coverage are the major objectives which can be improved by selecting high fitness vectors and applying them as seeds to test the circuits.

Best seeds generate high fitness patterns. High fitness test patterns are the patterns that find more number of faults in less time. Many methods and algorithms have been proposed for finding the seeds. Deterministic method leads to high computational complexity in its initialization and test set generation. In some methods Particle swarm algorithm is used to initialize the sequential circuits and ant colony algorithm is used to generate the corresponding test set which consumes more time. The proposed algorithm is formed by combining GA and PSO to find the best seed/seeds for VLSI circuits. Test pattern generation is initialised using GA. Then the best seed is found using PSO.

This paper is organised as follows. In section II basics of test generation is presented. Initial test pattern generation is explained in section III. PSO flow chart is given in section IV. Seed finding algorithm procedure is explained with examples in section V. In section VI simulation results are displayed and explained. Section VII presents conclusions.

II. TEST GENERATION BASICS

In non-converging state method [1] a certain number of populations had been generated and evaluated and the system was assumed to be in a non-converging state. But the test patterns in a test sequence influence the area of the modification logic. Different types of faults can be identified by genetic algorithm as shown in [6, 9] Genetic algorithm is used for the timing circuit initialization. An improved PSO algorithm [2] is used in timing circuit initialization phase. Timing circuit test sequence is composed of ‘0’, ‘1’ string. The initialization sequence is a special kind of test sequences. The PSO and ACTSG algorithms are able to generate a vector set at the stage of initialization and test set generation for sequential circuit and their efficiency is better than the contrasted algorithms was given in [7]. An approach to find a minimum length reset sequence was presented in [8]. It consists of an OBDD-based approach combined with a heuristic algorithm for preventing a memory overflow and the tool is a routine to quickly decide the non-resetability of a design. Many low power testing methods were analysed in this survey [3, 4] and the results of all the methods were compared. An ATPG design, based on genetic algorithm [5], is a deterministic test pattern generator which takes more time to find the essential sequences. A GA based sequential circuit fault simulator is employed to evaluate fitness of each candidate vector and select best vector to apply in each time frame [10, 14]. For a sequential circuit, the state of the circuit depends on previous patterns [11]. The proposed algorithm in [12] has two phases. In the first phase the test vectors used to initialise the flip-flops are generated and the second phase detects faults. A gate level simulation using GA is proposed in [13] in which random search of test vector is possible without being caught in a local minima or maxima.

For sequential circuits, initialization can be achieved through the use of a global reset signal which is connected to all state elements generating an initialization sequence that is able to initialize all the flip-flops to a known value. The initialization sequence is a special kind of test sequence. In the timing circuit, the initial state of each storage element is unknown. Usually, initialization vector and the vector in the evolutionary process of the groups are together looked as the quasi - initialization vector.

DOI10.17148/IARJSET.2017.41016

Binary decision diagram method is used to generate the initialization sequence but this method requires a lot of memory when describing a circuit and therefore is applicable to smaller circuits only. The method in [8] combines the binary decision diagram and Greedy algorithm to generate the initialization sequence. Ant colony algorithm [2] was applied for timing the circuit initialization process successfully. In his method the trigger, which is looked as food, generates the initialization sequence using ant path. But this method only used one individual ant and so can not realize parallel execution and the execution time increases when the circuit scale increases.

III. Initializing Test Pattern Generation

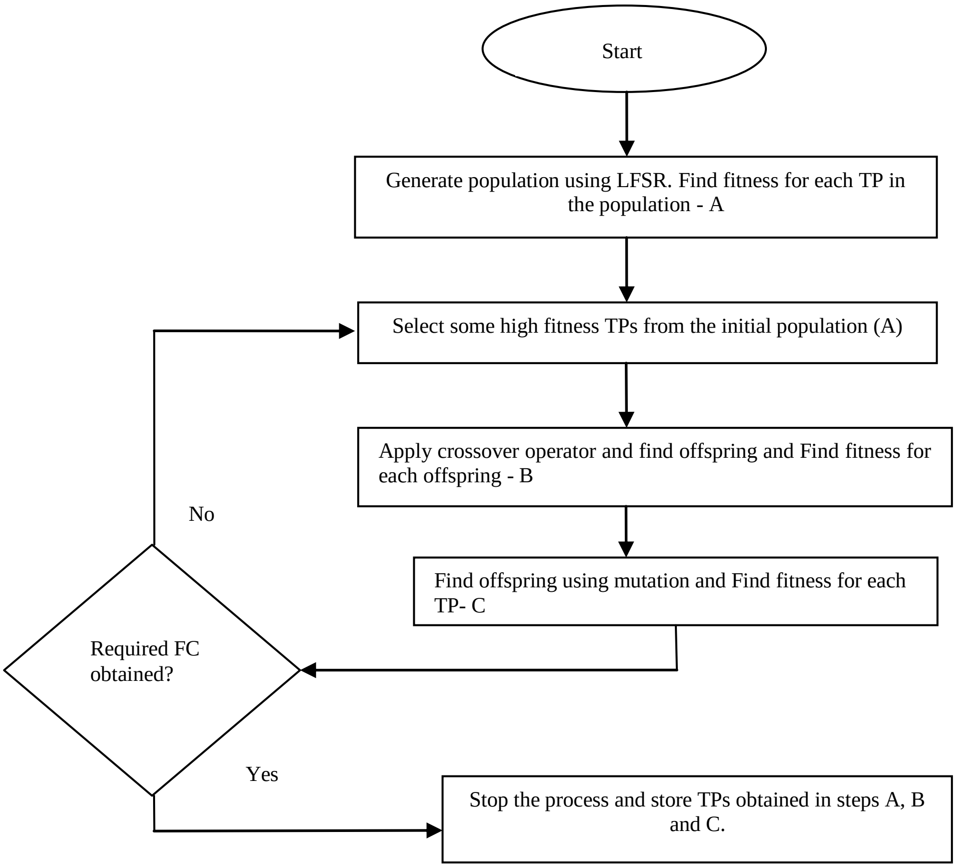

Start
Generate population using LFSR. Find fitness for each TP in the population - A
Select some high fitness TPs from the initial population (A)
Apply crossover operator and find offspring and Find fitness for each offspring - B
Find offspring using mutation and Find fitness for each TP- C
Required FC obtained?
No
Yes
Stop the process and store TPs obtained in steps A, B and C.

Fig. 1 Flow chart for initial TP generation using GA

In this method the initial test patterns are generated (see Figure 1) using genetic algorithm. Genetic algorithm is very efficient in generating test patterns for sequential circuits. The population is started from the test patterns generated by an LFSR. Fitness of some initial test patterns is found and the high fitness patterns are chosen as parents from the initial population. The next generation is produced by cross over and mutation. For each generation the fitness factor is found and the best chromosomes are stored as initial pattern on which the PSO algorithm is to be applied. Initially test patterns are generated by LFSR.

The LFSR common polynomial is given as

$$T(x) = 1+x+x^n \quad (1)$$

The test patterns are applied to a circuit with faults and the fitness is found. The patterns with high fitness are chosen as parents. From these parent patterns new trails are obtained by cross over and mutation. The fitness for each pattern is found which is generated using cross over and mutation. The patterns with high fitness are stored for future usage .The patterns obtained only using genetic algorithm can cover nearly 80% faults. Therefore these patterns are used to initialize the sequence generation.

DOI10.17148/IARJSET.2017.41016

IV. PSO ALGORITHM

The patterns stored in the previous sections are taken as initial patterns. In the first pattern two successive bits are flipped and its fitness is found. This process is continued for the remaining bits and for each time the fitness is noted and the bit change which improves the fitness more is determined. The same bit changing is applied on remaining stored test patterns. The highest fitness pattern among them is finally chosen as the best seed for that CUT. The application of same bit modification on other patterns results better improvement in fitness factor.

Only one pattern is analyzed for highest fitness and the same change is done in the remaining patterns. This method can improve fault coverage with less time consumption. These final patterns (seeds), which are having high fitness, are used as the test inputs during test mode.

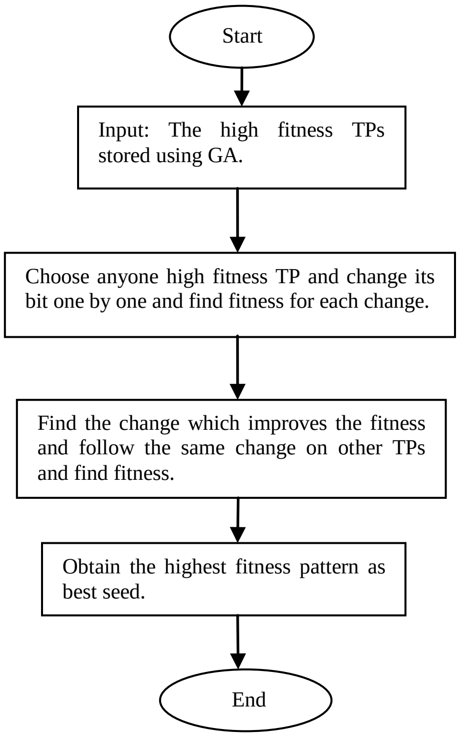

Start
Input: The high fitness TPs stored using GA.
Choose anyone high fitness TP and change its bit one by one and find fitness for each change.
Find the change which improves the fitness and follow the same change on other TPs and find fitness.
Obtain the highest fitness pattern as best seed.
End

Fig. 2 Flow chart for finding best seeds using PSO Algorithm

Figure 2 shows the flow chart of PSO algorithm for finding the best seed. Initial test patterns are obtained using GA. Among them high fitness pattern is chosen and each bit (successive two bits) of that pattern is complemented. For each modification the fitness is determined. The modification which improves fault coverage is noted. The same bit is modified on the remaining patterns which are already stored. The process is continued to achieve maximum fault coverage. The final pattern obtained using the proposed algorithm covers more number of faults in less time and it is chosen as the best seed.

V. PERFORMANCE OF THE PROPOSED ALGORITHM

C17 is a combinational bench mark chosen as to CUT. It has five inputs, two outputs and six NAND gates. The maximum number faults that can occur are 22. Stuck-at-faults are inserted and the best seed found is 01111. Maximal length fifth order polynomial yields 31 patterns. TPs 00101, 01101, 01110, 10110, 10010, 10111, 11001, 11011, 11100 generate repeated patterns therefore they cannot be applied as seeds. Among 31 patterns only 22 test patterns are applied and “01111” is found as the best seed by the proposed algorithm (see Table 1). The obtained seed finds all faults within nine clock pulses.

For proving the proposed algorithm for sequential benchmarks S27, S298 are considered. S27 is a sequential circuit benchmark with 4 inputs and one output. The number faults inserted here is 26. The tool takes a finite number of iterations for providing the best seed value. Out of all the generations the best fit individuals improve the optimum value. In the above S27 circuit 26 stuck-at-faults were inserted at all primary inputs, outputs and some of the interconnections. The number input of S27 is 4.Therefore 16 input patterns can be applied as test input. All the 15 input patterns are applied to LFSR and the generated patterns are tested for fault coverage.

TABLE 1 FINDING THE BEST SEED USING THE PROPOSED ALGORITHM FOR C17 BENCHMARK

No. of faults inserted	**TPG used**	**TPs required find faults**	**Seed applied**	**No. of clocks required to cover all faults**	**Best seed found by the proposed algorithm**
22	LFSR	11x1x	00001	11	01111
	x^5+x+1	0111x	00010	14	
		0xx1x/ x0x1x	00011	11	
		xxx01	00100	13	
		01xxx	00110	17	
		10xxx	00111	11	
		0101x	01000	14	
		X000x/0x00x	01001	20	
		Xx000/ x0x00	01010	17	
			01011	10	
			01100	17	
			01111	9	
			10000	11	
			10001	15	
			10011	21	
			10100	18	
			10101	18	
			11000	16	
			11010	19	
			11101	19	
			11110	20	
			11 111	21	

After the end of traversing through all generations the tool provides the seed value “1111” as the fittest condition. The patterns generated using the above obtained seed find all faults within six clock pulses. Table 2 shows the LFSR seed value and those test vector sequences with their respective time taken for fault detection. Among 15 seeds the best seed is found by the proposed algorithm.

TABLE 2 FINDING THE BEST SEED FOR VARIOUS CUTS USING THE PROPOSED ALGORITHM

CUT	**No of I/Os and gates**	**No. of faults inserted**	**No. of clocks required to cover all faults**	**Best seed found**
C17 benchmark	4 inputs, 1 output, 3 D-type FFs 2 inverters.	22	9	01111
S27 benchmark	4 inputs, 1 output, 3 D-type FFs 2 inverters, 8 gates (1 ANDs + 1 NANDs + 2 ORs + 4 NORs)	26	7	1111
S298 benchmark	3 inputs, 6 outputs, 14 D-type FFs 44 inverters, 75 gates (31 ANDs + 9 NANDs, + 16 ORs + 19 NORs)	47	4	101

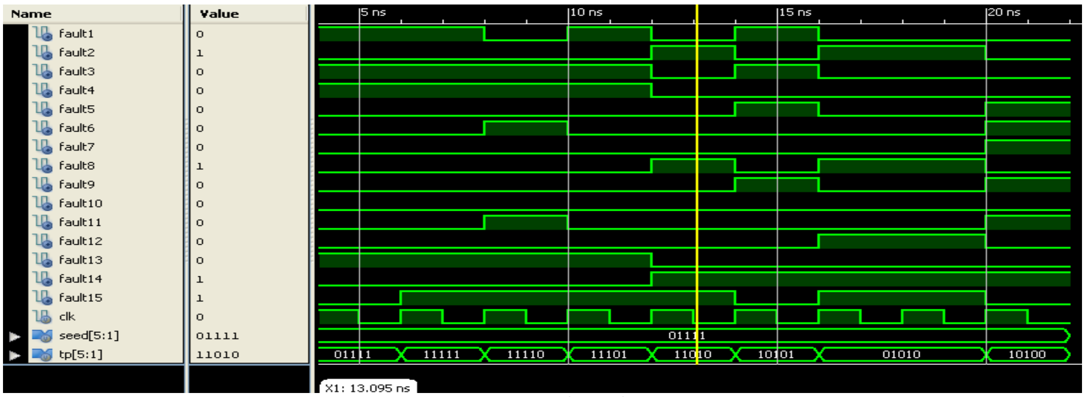
All fifteen patterns were applied to the test pattern generator and it is clearly proved that the seed found using the proposed algorithm covers all faults with minimum clock pulses. The same method was followed for S298 sequential bench mark to obtain the best seed. This circuit has 3 inputs only. Therefore 7 test vectors can be applied as test inputs. Among them test patterns generated using '101' as seed covers all inserted faults within four clock pulses.

In the proposed algorithm a single pattern is modified a number of times and the modification which improves fault coverage is followed for the remaining patterns. This method reduces the time needed for new pattern generation with high fault coverage. Experiments were done on small examples and need to be done on larger commercial examples.

Seeds found by the proposed algorithm can be stored in ROM and will be applied to the test pattern generator with regular intervals. Using the proposed algorithm the minimum number seeds that are required to find all faults can be determined. Therefore the memory requirement needed to store the seeds is also minimized by this method.

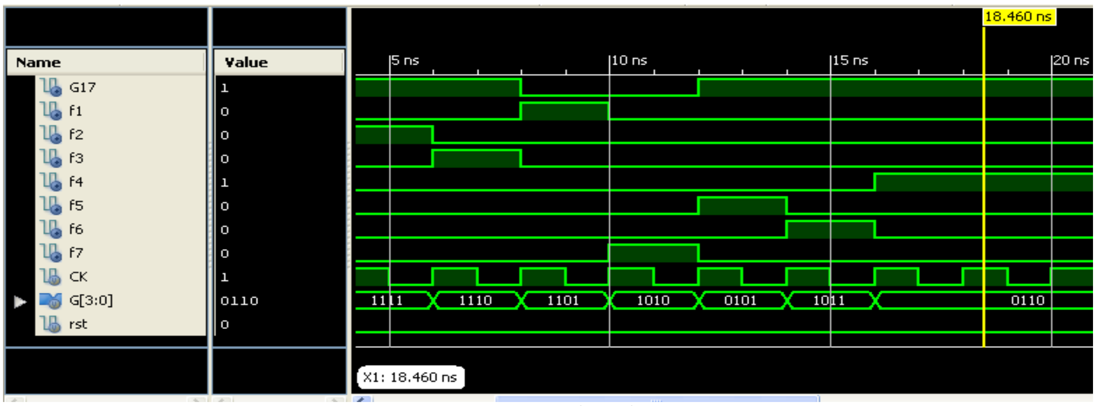
VI. Results and Discussion

The simulation results of proposed algorithm on C17 combinational bench mark and S27 sequential bench mark circuit are shown in Figures 3 and 4. The number of inputs in C17 benchmark is 5. Therefore 31 patterns can be applied as seeds to an LFSR. The pattern 01111 found all inserted faults with in 9 clock pulses. The proposed algorithm found 01111 as the best seed for C17 benchmark circuit.



Name	Value
fault1	0
fault2	1
fault3	0
fault4	0
fault5	0
fault6	0
fault7	0
fault8	1
fault9	0
fault10	0
fault11	0
fault12	0
fault13	0
fault14	1
fault15	1
clk	0
seed[5:1]	01111
tp[5:1]	11010

Fig. 3 Simulation output of C17 for seed 01111



Name	Value
G17	1
f1	0
f2	0
f3	0
f4	1
f5	0
f6	0
f7	0
CK	1
G[3:0]	0110
rst	0

Fig. 4 Simulation output of S27 for seed 1111

The number of inputs in S27 bench mark circuit is 4. Therefore 15 patterns can be applied as seeds to an LFSR. The generated patterns using each seed were applied to CUT as test inputs. The best seed found by the proposed algorithm is "1111" which covers all faults in 7 clock pulses. Output of good circuit was indicated by G17.Seven number of faulty circuits with 25 faults was compared with good circuit output. The difference between the outputs of good and faulty circuits is shown by f1 to f7. The simulation outputs clearly show that the best seed found by the proposed algorithm covers all faults in minimum clock pulses.

VII. Conclusion

The proposed algorithm combines GA and PSO for generating high fitness seeds. The proposed algorithm has the advantages of both GA and PSO. It has high fault coverage and obtains the final solution in minimum time. The proposed algorithm is applied to some combinational and sequential benchmark circuits and the results prove that the

ISSN (Online) 2393-8021
ISSN (Print) 2394-1588

best seeds can be obtained for all combinational, sequential circuits that have maximum fault coverage with less testing time.

Larger circuits with more number of inputs required large test pattern generator. Determination of test patterns that used to cover all faults is a complex process. In future, test set for large circuits will be generated using the proposed algorithm.

REFERENCES

[1] Dobai, R 2011, 'Test Generation for Asynchronous Sequential Digital Circuit', Information Sciences and Technologies Bulletin of the ACM Slovakia, vol. 3, no. 1, pp. 73-83.

[2] Fu Xin 2012, 'A Sequential Circuits Test Set Generation Method Based on Ant Colony Particle Swarm algorithm', National Conference on Information Technology and Computer Science (CITCS 2012), pp. 205-209.

[3] Girard, P 2002, 'Survey of Low-Power Testing of VLSI Circuits', IEEE Design and Test of Computers, vol. 19, no. 3, pp.80-90.

[4] Gong, Y, He, R & Li, X 2003, 'A low power BIST TPG design', in Proceedings of the IEEE 5th International ASIC Conference, Beijing, China, pp.1136–1139.

[5] Gregor Papa 2007, 'Deterministic Test Pattern Generator Design with Genetic Algorithm Approach', Journal of Electrical Engineering, vol. 58, no. 3, pp.121–127.

[6] Chayanika Sharma 2013, 'A Survey on Software Testing Techniques using Genetic algorithm', International Journal of Computer Science Issues IJCSI, vol. 10, no. 1, pp. 381-393.

[7] Chen Guangyu, Li Zhi & Xu Chuanpei 2002, 'Synchronous sequential circuits based on ant algorithm initialization research', Journal of electronic measurement and instrument, vol. 16, no. 4, pp. 33-37.

[8] Keim, M 1996, 'On the Resetability of Synchronous Sequential Circuits', Proceedings of 14th IEEE VLSI Test Symposium, Princeton, NJ, USA, pp.240-245.

[9] Irith Pomeranz & Sudhakar M Reddy 2010, 'TOV: Sequential Test Generation by Ordering of Test Vectors', IEEE Transactions on Computer-Aided Design Of Integrated Circuits And Systems, vol. 29, no. 3, pp. 454-465.

[10] Elizabeth M. Rudnick, Janak H. patel, Gary S.Greenshen, Thomas M.Niermann, "Sequential Circuit test generation in a genetic algorithm frame work", Proceedings of the 31st annual design Automation Conference, pp. 698-704,1994.

[11] M. Srinivas and L.M. Patnaik, "A simulation-based test generation scheme using genetic algorithms ", Proc. International Conference VLSI Design, pp. 132-135,1993.

[12] Xiaoming Yu Abramovici M., "Sequential Circuit ATPG using combinational algorithms", IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, pp. 1294-1301, 2005.

[13] Dhiraj Sangwan, Seema Verma, Rajsh Kumar, "A genetic algorithm based two phase fault simulator for sequential circuit", International Journal of Computer Applications, Vol. 12, no. 10, pp. 17-24, 2011.

[14] E.M. Rudnick, J.H. Patel, G.S.Greenshen, T. M.Niermann, "A genetic algorithm frame work for test generation ", IEEE Transactions on Computer Aided Design Of Integrated Circuits and Systems , pp. 1034-1044, 1997.

DOI10.17148/IARJSET.2017.41016