International Journal of Trend in Research and Development, Volume 2(3), ISSN: 2394-9333
www.ijtrd.com

Frame Work For Multiple Fault Diagnosis Based On Multiple Fault Simulation Using Particle Swarm Optimization

[1] S. Shibin, [2]Mr.K. Gopal Ram M.E
[1]PG scholar, [2]Assistant Professor,
M.E- Embedded system technologies,
[1,2]Sun college of Engineering and Technology,
Erachakulam, Nagercoil, Tamilnadu, India.

***Abstract* - In this paper presents multiple error diagnosis algorithms to overcome two significant problems associated with current error diagnosis techniques targeting large circuits: their use of limited error models and a lack of solutions that scale well for multiple errors. Our solution is based on a non-enumerative analysis technique, based on logic simulation (3-valued and symbolic), for simultaneously analyzing all possible errors at sets of nodes in the circuit. Error models are introduced in order to address the "locality" aspect of error location and to identify sets of nodes that are "local" with respect to each other. Theoretical results are provided to guarantee the diagnosis of modeled errors and robust diagnosis approaches are shown to address the cases when errors do not correspond to the modeled types. Experimental results on benchmark circuits demonstrate accurate and extremely rapid location of errors of large multiplicity**

Keyword: Diagnosabili; Wigner-Ville distribution; Gabor Expansion; ILP; Pruning.

I. INTRODUCTION

Fault diagnosis plays a major role in fast yield ramp up of the VLSI IC manufacturing process. In recent years, several works have been reported on multiple-fault diagnosis. In, authors have proposed to generate test patterns to improve the multiple-fault-diagnosis resolution. The works have used a technique known as inject-and-evaluate paradigm. In, the authors have attempted to find a minimal set of modifications that correct the faulty design. The work uses single-location at-a-time patterns to perform the diagnosis for failed responses caused by multiple faults. The works have proposed an incremental multiple-fault simulation strategy. Candidate faults are injected sequentially, ranked by the number of fail- and pass-patterns explained by them. However, if a wrong fault is chosen at any stage, it may lead to a faulty solution. Moreover, the algorithm is inherently based on single fault simulation.

In, an algorithm has been proposed that generates a set of double, triple, and quadruple faults by choosing faults randomly from the set of candidate faults. These fault sets are then simulated and a fault ranking is performed according to the number of failed patterns each can explain. In, a Boolean satisfiability-based method for multiple-fault diagnosis has been proposed. However, the diagnostic resolution achieved by the approach is on the higher side. Recently, other fault models, such as timing aware delay fault model, stuck-open fault model, etc., have been considered. In, a framework has been proposed, which can deal with several fault models at the same time.

II. SYSTEM OVERVIEW

Authors have proposed to generate test patterns to improve the multiple-fault-diagnosis resolution. The works have used a technique known as inject-and-evaluate paradigm. The work uses single-location at-a-time patterns to perform the diagnosis for failed responses caused by multiple faults. The works have proposed an incremental multiple-fault simulation strategy. In this brief, we have proposed a framework for fault diagnosis based on multiple fault simulation. Since the number of faulty sites is unknown, multiple fault simulation algorithms are inherently exponential in nature. In order to explore this exponential search space, we have used a particle swarm optimization (PSO) algorithm. Initially, a list of possible fault candidates is found out by critical path tracing from each failing primary output (PO) and taking a union of them. Next, we try to find a single perfect fault candidate. If

there exists a perfect candidate, we stop and report the result. Otherwise, the faults are sorted (in descending order) according to the number of failed and passed patterns.

A. Fault circuit model

Two signals are connected together. Depending on the logic circuitry employed, this may result in a XOR or wired-AND logic function. Since there are O (n^2) potential bridging faults, they are normally restricted to signals that are physically adjacent in the design.

B. ILP formulation

The following terminologies have been used to formulate the ILP.

1) F: A Boolean array of size equal to the number of collapsed faults present in the circuit. The i^{th} element of F is given by f_i. f_i is "1" if the i^{th} fault is present, "0" otherwise.

2) T: The set of test patterns.

3) C_A: Actual circuit under test.

4) C_F: The faulty circuit with faults indicated in F.

5) Simulate(C, t_i) : A function which takes a circuit C and a test pattern $t_i \in$ T & returns output response obtained when t_i is applied to C .

6) Equal (O_1, O_2): A function that takes two output responses $O_1$and $O_2$and returns "1" if they are same. Otherwise, it returns "0."

C. PSO model

Particle Structure: The probable candidate faults are first identified using a critical path tracing method and are given as input to the PSO. A particle is an n -bit binary array, n being equal to the total number of candidate faults. A "1" position indicates that the i^{th} fault in the candidate list is present in the circuit; a "0" indicates its absence.

Initial Positions: The value of this variable is taken as an input and is generally not more than ten (depends on the manufacturing process). Let there be n possible faults and k particles. For $4* k /5$ particles, we make sure that the initial number of faults in each particle is in the range of 1 to MAX_POSS_FAULT.

Fitness of a Particle: Fitness of a particle is calculated in terms of the number of test patterns. The faults depicted by the particle are injected into the circuit. Both fail and pass patterns are simulated in presence of these faults and the responses are compared with the tester responses. If the two responses for a particular test pattern match, the test pattern is said to be fully explained by the particle.

GBest and Pbest: Both GBest and PBest (for a particle) are updated after each iteration.

Mask Operator and New Position of a Particle: Mask operator is calculated by comparing bit-by-bit the GBest with the particle's current position. If i th fault is present/absent in both GBest and the particle, i^{th} bit is set to "0," otherwise it is set to "1."After the mask operator has been calculated, a bitwise- XOR is performed between particle's current position and the mask operator. The final position obtained after applying the PBest mask operator is considered as the particle's new position.

D. ILP Formulation of Fault Diagnosis Problem

The following terminologies have been used to formulate the ILP.

1) F: A Boolean array of size equal to the number of collapsed faults present in the circuit. The i^{th} element of F is given by f_i. f_i is "1" if the i^{th} fault is present, "0" otherwise.

2) T: The set of test patterns.

3) C_A: Actual circuit under test.

4) C_F: The faulty circuit with faults indicated in F.

5) Simulate(C, t_i): A function which takes a circuit C and a test pattern $t_i \in$ T , and returns the output response obtained when t_i is applied to C .

6) Equal (O_1, O_2): A function that takes two output responses $O_1$and $O_2$and returns "1" if they are same. Otherwise, it returns "0."

III. PSO for Multiple Fault Simulation

PSO is a population-based stochastic technique. We have used a discrete PSO (DPSO) formulation to solve the multiple fault simulation problem. For i^{th} particle, the position is denoted as p^i_k. Let p $best^i$ be the local best solution that particle i has seen so far and $gbest_k$ be the global best particle of iteration k .

In the above expressions, a →b represents the exclusive-or sequence to be applied on components of a to transform it to b .For example, if a =⟨1,1,00⟩ and b ⟨1,1,00⟩ a →b =⟨0,1,0,1⟩ The operator is the fusion operator. Applied on two exclusive-or sequences, is equal to the sequence in which the sequence of exclusive-or operations in a is followed by those in b.

Let us consider a permanent faulty router that cannot be corrected. This router is permanently disabled. Similarly, during the reconfiguration of a PRR, no packet can be sent inside the area being reconfigured. Thus, these PRRs are dynamically isolated. However, these isolations can lead to data packet losses or increase packet transmission latency. More precisely, these drawbacks occur when routers containing data packets in their output buffers have their neighboring nodes unavailable due to a dynamic reconfiguration or permanent fault detection. Thereby, these data packets remain stored in the output routers until the end of the reconfiguration (dynamic implementation case) or are lost, in the case of detection of a permanent faulty node.

However, these isolations can lead to data packet losses or increase packet transmission latency. More precisely, these drawbacks occur when routers containing data packets in their output buffers have their neighboring nodes unavailable due to a dynamic reconfiguration or permanent fault detection. Thereby, these data packets remain stored in the output routers until the end of the reconfiguration (dynamic implementation case) or are lost, in the case of detection of a permanent faulty node.

First, initialize the positions. The value of this variable is taken as an input and is generally not more than ten (depends on the manufacturing process).Next, initialize particle and velocity. Fitness of a particle is calculated in terms of the number of test patterns, the particle can explain. The faults depicted by the particle are injected into the circuit. Both fail and pass patterns are simulated in presence of these faults and the responses are compared with the tester responses. For each particle, the number of test patterns explained by the particle is used as its fitness. If fitness is better both Gbest and Pbest (for a particle) are updated after each iteration. Update both velocity and its position.

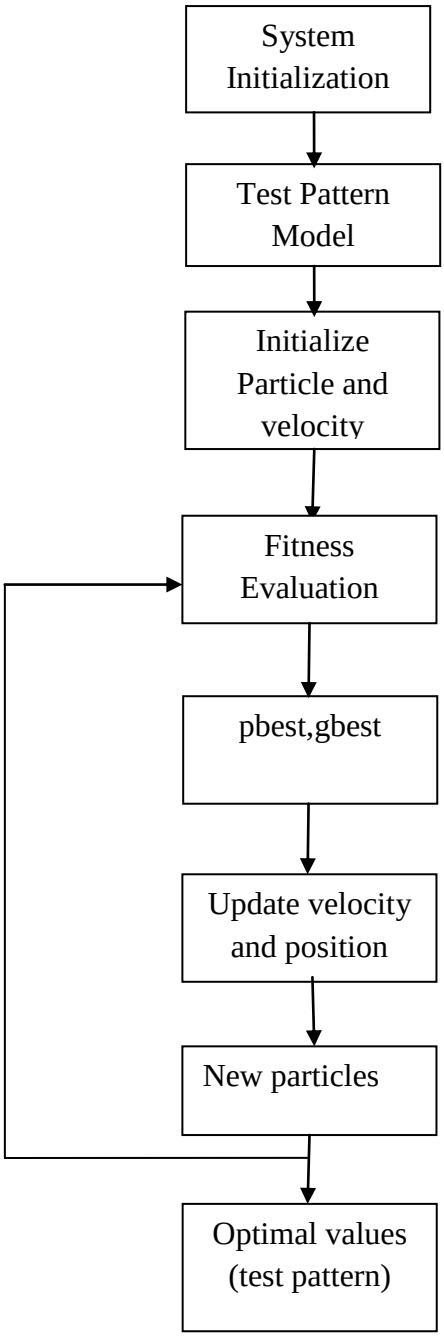

Fig 3.1 Block Diagram of Optimization Based on Multiple Fault Diagnosis

Mask operator is calculated by comparing bit-by-bit the Gbest with the particle's current position. After the mask operator has been calculated, a bitwise- XOR is performed between particle's current position and the mask operator. After applying the mask operator for Gbest, the particle positions go through the same process for their respective Pbest also. The final position obtained after applying the Pbest mask operator is considered as the particle's new position. With each iteration, the

particles modify their positions and move closer to the optimal solution.

A. ALGORITHM

Particle swarm optimization is one of the evolutionary computation techniques. The method has been developed through simulation of simplified social models. PSO learns from scenario and uses it to solve the optimization problems. In PSO, each single solution is a "bird" in the search space. We call it "particle". All particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles.

Data: A Circuit C, A test set TS, PSO solution sets with redundant faults

Result: PSO solution sets removing redundant faults

begin

for Particles having fitness value equal to G best do

prev fitness = fitness of the particle;

for faults present in the particle do

Drop a fault from the particle (by flipping a '1' bit to ' 0');

Simulate C in presence o f t he modified particle (after dropping the fault) f or every pattern i n T and calculate its fitness;

if fitness is less than prev fitness then

Restore the fault (by flipping the corresponding b it back to '1');

else

Drop the fault ;

end

return Pruned PSO results;

IV.RESULTS AND DISCUSSION

A framework for fault diagnosis based on multiple fault simulation. Since the number of faulty sites is unknown, multiple fault simulation algorithms are inherently exponential in nature. In order to explore this exponential search space, we have used a particle swarm optimization (PSO) algorithm. Initially, a list of possible fault candidates is found out by critical path tracing from each failing primary output (PO) and taking a union of them. Next, we try to find a single perfect fault candidate. If there exists a perfect candidate, we stop and report the result. Otherwise, the faults are sorted (in descending order) according to the number of failed and passed patterns.

A. GENERATE RANDOM TEST PATTERN

The probable candidate faults are first identified using a critical path tracing method and are given as input to the PSO. A particle is an n -bit binary array, n being equal to the total number of candidate faults. A "1" at i position indicates that the i^{th} fault in the candidate list is present in the circuit, a "0" indicates its absence.

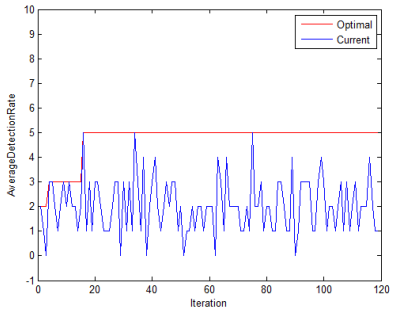

Fig 4.1 Variation of Avg.Detection Rate Iteration

B. PSO OPTIMIZATION

The value of this variable is taken as an input and is generally not more than ten (depends on the manufacturing process). Let there be n possible faults and k particles. For 4∗ k /5 particles, we make sure that the initial number of faults in each particle is in the range of 1 to MAX_POSS_FAULT.

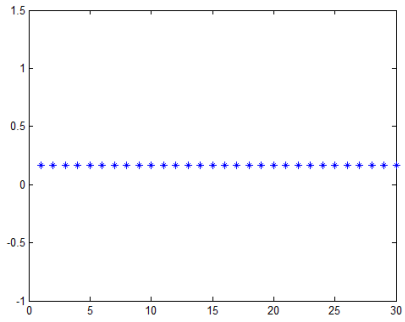

Fig 4.2 Particle Swarm Optimization

C. SELECTED PATTERN

Fitness of a particle is calculated in terms of the number of test patterns. The faults depicted by the particle are injected into the circuit. Both fail and pass patterns are simulated in presence of these faults and the responses are compared with the tester responses. If the two responses for a particular test pattern match, the test pattern is said to be fully explained by the particle.

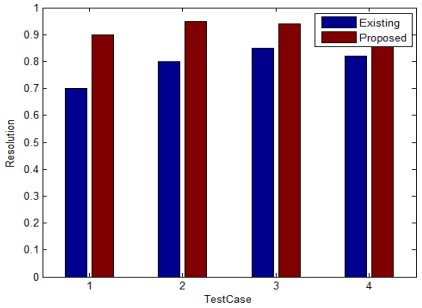

Fig 4.3 Variation of Resolution with Test Case

D. FAULT HIT RATIO AND TESTCASE

Mask operator is calculated by comparing bit-by-bit the Gbest with the particle's current position. If i fault is present/absent in both Gbest and the particle, i bit is set to "0," otherwise it is set to "1."After the mask operator has been calculated, a bitwise- XOR is performed between particle's current position and the mask operator. After applying the mask operator for Gbest, the particle positions go through the same process for their respective Pbest also. The final position obtained after applying the Pbest mask operator is considered as the particle's new position.

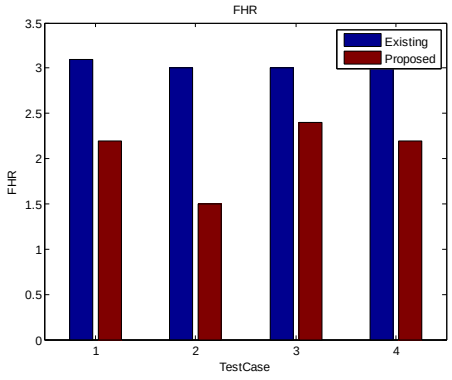

Fig 4.4 Variation of FHR with Test Case

E. DELAY AND TESTCASE

If for the last 30 generations, the best result that is the particle, which has maximum fitness, do not change, we stop running the algorithm. We also use a maximum iteration condition, after which we stop, even if solution is still improving.

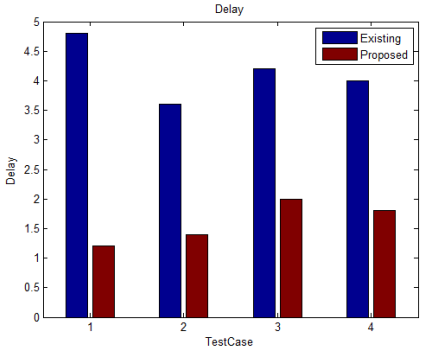

Fig 4.5 Variation of Delay with Test Output

Selected Test pattern is -->

00000 11011 00100 11010 01100

ans = 6

V. CONCLUSION

In this brief, we proposed a frame work for the diagnosis of multiple stuck-at and transition faults. The algorithm has a very high first hit rank and diagnose ability with small resolution. Comparison with previous works proved the effectiveness of the algorithm. We are currently working on extending the algorithm for different fault models, such as bridging faults and other un-modeled faults. Since, the approach is an effect-cause based one; we believe that these fault models can easily be incorporated into the framework.

REFERENCES

[1] T. W. Bartenstein, D. Heaberlin, L. Huisman, and D. Sliwinski, "Diagnosing combinational logic designs using the single location at-a-time (SLAT) paradigm," in *Proc. IEEE Int. Test Conf.*, Nov. 2001, pp. 287–296H..

[2] V. Boppana, R. Mukherjee, J. Jain, M. Fujita, and P. Bollineni, "Multiple error diagnosis based on xlists," in *Proc. 36th Ann. ACM/IEEE Design Autom. Conf.*, Jun. 1999, pp. 660–665.

[3] P. Y. Chung, Y. M. Wang, and I. N. Hajj, "Diagnosis and correction of logic design errors in digital circuits," in *Proc. 30th Int. Design Autom. Conf.*, Jun. 1993, pp. 503–508.

[4] X. Fan, W. Moore, C. Hora, and G. Gronthoud, "Stuck-open fault diagnosis with stuck-at model," in *Proc. 10th IEEE Eur. Symp.*, May 2005, pp. 182–187.

[5] I. Pomeranz and S. M. Reddy, "On correction of multiple design errors," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 14, no. 2, pp. 255–264, Feb. 1995.

6]. S.-Y. Huang, "On improving the accuracy of multiple defect diagnosis," in *Proc. 19th IEEE VLSI Test Symp.*, May 2001, pp. 34–39.

[7] A. Veneris, J. Liu, M. Amiri, and M. Abadir, "Incremental diagnosis and correction of multiple faults and errors," in *Proc. Design, Autom. Eur. Conf. Exhibit.*, 2002, pp. 716–721.

[8] Z. Wang, M. Marek-Sadowska, K.-H. Tsai, and J. Rajski, "Analysis and methodology for multiple-fault diagnosis," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 3, pp. 558–575, Mar. 2006.

[9] H. Takahashi, K. O. Boateng, K. K. Saluja, and Y. Takamatsu, "On diagnosingmultiple stuck-at faults using multiple and singlefault simulation in combinational circuits," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 21, no. 3, pp. 362–368, Mar. 2002.

[10]A. Smith, A. Veneris, M. Ali, and A. Viglas, "Fault diagnosis and logic debugging using Boolean satisfiability," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 24, no. 10, pp. 1606–1621, Oct. 2005.

[11] V. J. Mehta, M. Marek-Sadowska, K.-H. Tsai, and J. Rajski, "Timing-aware multiple-delay-fault diagnosis," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 28, no. 2, pp. 245–258, Feb. 2009.

[12]Takahashi, N. Yanagida, and Y. Takamatsu, "Enhancing multiple fault diagnosis in combinational circuits based on sensitized paths and EB testing," in *Proc. 4th Asian Test Symp.*, Nov. 1995, pp. 58–64

[13] B. Bosio, P. Girard, S. Pravossoudovitch, and A. Virazel, "A comprehensive framework for logic diagnosis of arbitrary defects," *IEEE Trans. Comput.*, vol. 59, no. 3, pp. 289–300, Mar. 2010.

[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4. Nov.–Dec. 1995, pp. 1942–1948.

[15] L. Wang, C. Wu, and X. Wen, *VLSI Test Principles and Architectures: Design for Testability*, 1st ed. Amsterdam, The Netherlands: Elsevier, 2006.

[16] S. Kundu, S. Chattopadhyay, I. Sengupta, and R. Kapur, "Multiple fault diagnosis based on multiple fault simulation using particle swarm optimization," in *Proc. 24th Int. Conf. VLSI Design*, Jan. 2011, pp. 364–369.

[17] K. Wang, L. Huang, C. Zhou, and W. Pang, "Particle swarm optimization for traveling Salesman problem," in *Proc. 2nd Int. Conf. Mach. Learn. Cyberm.*, vol. 3. Nov. 2003, pp. 1583–1585.

[18]L. Guilan, Z. Hai, and S. Chunhe, "Convergence analysis of a dynamic discrete PSO algorithm," in *Proc. 1st Int. Conf. Intell. Netw. Intell. Syst.*, Nov. 2008, pp. 89–92.

[19]*TetraMAX ATPG Guide*, Synopsys, Mountain View, CA, USA, 2006.