

ARTIFICIAL NEURAL NETWORK BASED MULTIPLE FAULT DIAGNOSIS IN DIGITAL CIRCUITS

A.A. AL-Jumah and T. Arslan

Cardiff University of Wales,
School of Engineering,
Cardiff CF2 3TF, UK
aljumah@cf.ac.uk

ABSTRACT

The paper describes a technique, based on the use of Artificial Neural Networks (ANNs), for the diagnosis of multiple faults in digital circuits. The technique utilises different quantities of randomly *selected* circuit test data derived from a *fault truth table* [1], which is constructed by inserting random single stuck-at faults in the circuit. The paper describes the diagnostic procedure using the technique, the ANN architecture and results obtained with example circuits. Our results demonstrate that when the test data selection procedure is guided by test vectors of the circuit a compact, efficient and flexible ANN architecture is achieved.

1. INTRODUCTION

A local defect in a VLSI device may cause multiple stuck-at faults instead of single stuck-at faults. With the increase in the complexity of such devices the probability of multiple faults occurring has significantly increased. Dealing with multiple faults is a complex task due to the large combination of such faults. For example, a circuit with N nets may have up to $3^N - 1$ faults. Due to the large number of possible multiple faults in the circuit, using single stuck-at fault to detect multiple faults will be easier and more efficient [2]. For this reason there is a demand for effective methods of treating such faults.

Numerous research papers have tackled the issue of multiple faults, however most of these have concentrated on aspects of Test Pattern Generation, for example [3-5].

Testing is to identify whether a circuit is faulty or not, while fault diagnosis is to detect and isolate the faulty component(s) [6]. In some applications a go/no-go test is not sufficient particularly when the circuit under test is to be repaired to the smallest replaceable element in the circuit. In addition diagnostic information is useful even in cases where repair is not needed in order to monitor the performance of the fabrication process [7].

Artificial Neural Networks (ANNs) have been used for the diagnosis of multiple faults after being trained with single

stuck at failure information from a *fault truth table* (FTT) [1]. Such a table usually lists all single-fault/no-fault conditions in the circuit against the resulting state of both internal and external nets. To obtain acceptable diagnosis performance the ANNs need to be trained with all entries in the FTT, which increases exponentially with circuit size. Even then the diagnosis performance is about 81% for single fault detection and 4% for two faults detection [8].

The authors published initial results with a basic ANN configuration in [1]. In this paper we demonstrate that, using an approach based on ANNs trained with test pattern sets generated using an Automatic Test Pattern Generation system can lead to a higher multiple faults diagnosis performance, and a significant reduction in the size of the ANN and the training data, hence enabling the diagnosis of larger and more complex systems. The paper describes the construction of the ANN, the procedure of training it in order to produce a structure which is flexible and robust to cope with the diagnosis of a wide variety of circuits, and provides results with a number of example circuits.

2. IMPLEMENTATION

The performance of the ANN is investigated using three different types of data. These types are derived from data corresponding to all possible input combinations in the FTT, 50% of possible input combinations in the FTT and the use of test vectors alone. In each case the ANN is trained with single fault information from the FTT and is tested by its ability to diagnoses multiple faults. Fig. 1 illustrates the flowchart of the proposed method. Test vectors are generated using an ATPG software package [9] while truth tables representing all possible input combinations and 50% of possible input combinations are generated using a C program. One of the above data sets is selected to train and test the ANN diagnostic system. The training data is generated by inserting one fault in the circuit followed by the construction of the corresponding FTT. The testing data is compiled by randomly inserting two faults in the circuit. The ANN diagnostic system is trained with single-fault data and tested with multiple-fault data.

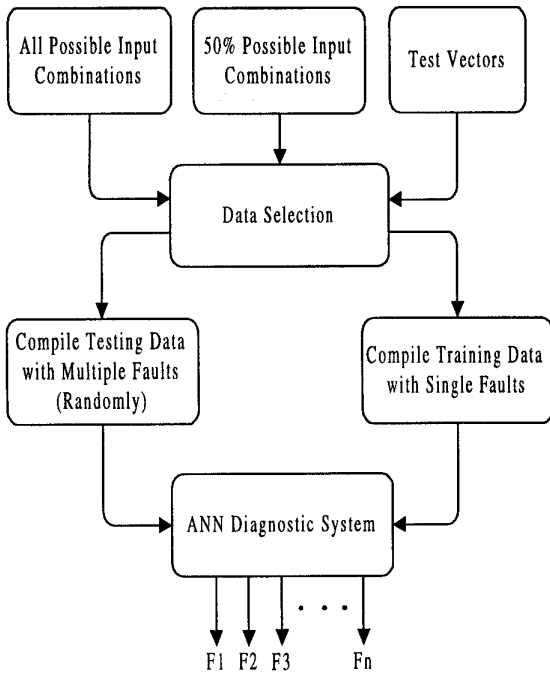


Fig. 1 Flowchart of the diagnostic system

3. DATA REPRESENTATION

The training data is formed using a single stuck-at fault, while the testing data is compiled randomly with multiple faults. Inputs to the ANN are consecutive sets of binary circuit inputs and corresponding outputs fed in a serial form for each set of input combinations in the FFT. The number of input sets fed into the ANN is dependent on the type of the data selected (all possible input combinations, 50% of possible input combinations or test vectors). In order to illustrate the data representation method adopted, the example in Fig. 2 is used. Table 1 represents the FFT of the circuit when the input $i1$ is stuck-at 0 ($i1/0$), input $i2$ is stuck-at 1 ($i2/1$) and the output $g1$ is stuck-at 0 ($g1/0$).

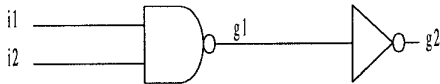


Fig. 2 Example circuit.

		f-free		i1/0		i2/1		g1/0	
i1	i2	g1	g2	g1	g2	g1	g2	g1	g2
0	0	1	0	1	0	1	0	0	1
0	1	1	0	1	0	1	0	0	1
1	0	1	0	1	0	0	1	0	1
1	1	0	1	1	0	0	1	0	1

Table 1 Fault truth table of the example.

All combinations of data presented to the ANN for the circuit in Fig. 2 is shown in Table 2. Each entry in the table is formed by sequences of the string 'i1 i2 g1 g2' repeated for all $i1$ and $i2$ combinations in table 1. This is repeated for each of the cases: fault-free, $i1$ stuck-at 0 ($i1/0$), $i2$ stuck-at 1 ($i2/1$) and $g1$ stuck-at 0 ($g1/0$). Each case (row) represents a separate input pattern to the ANN.

f-free	0	0	1	0	0	1	1	0	1	0	1	0	1
i1/0	0	0	1	0	0	1	1	0	0	0	1	0	0
i2/1	0	0	1	0	0	1	1	0	1	1	0	1	0
g1/0	0	0	0	1	0	1	0	1	1	0	0	1	1

Table 2 ANN inputs representation.

The ANN output representation is as follows: a '1' output indicates a fault in a corresponding gate whereas a '0' indicates a fault free gate. In this study 0.1 is used instead of binary '0' and 0.9 instead of binary '1', In order to overcome the boundary effects of the sigmoid activation function.

4. ANN ARCHITECTURE

Initial investigations showed that an ANN using a Multi-layer Perceptron with backpropagation algorithm is suitable for this work. The ANN consists of three layers, the number of neurons in the input layer is dependent on the type of data selected. In this study, when C17 is used for example, the number of input neurons with all possible input combinations, 50% of possible input combinations and test vectors are 352, 176 and 44 respectively. The number of neurons in the hidden layer are 95, 65 and 35 respectively. The number of neurons in the output layer equals the number of all possible single faults added to the fault free case in the circuit, which in this case is 23 neurons. The learning rate and momentum term are 0.4 and 0.09 respectively. Our investigation also revealed that the use of Gaussian initialisation function [10] provides a better diagnosis performance. Since it introduces random noise into the ANN training set producing a structure which is more robust and flexible to cope with a wide variety of circuits.

5. RESULTS

Three circuits are used to demonstrate the performance of the ANN diagnostic system, C17 ISCAS benchmark circuit, the circuit C1 [5] shown in Fig. 3 and the circuit C2 [11] shown in Fig. 4.

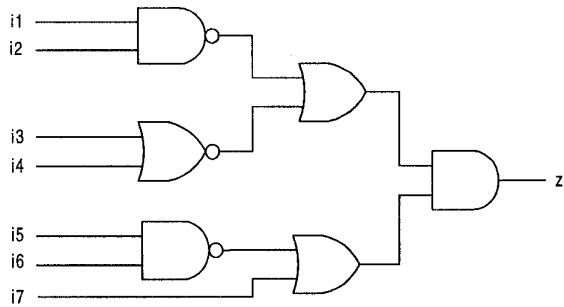


Fig. 3 Example circuit C1.

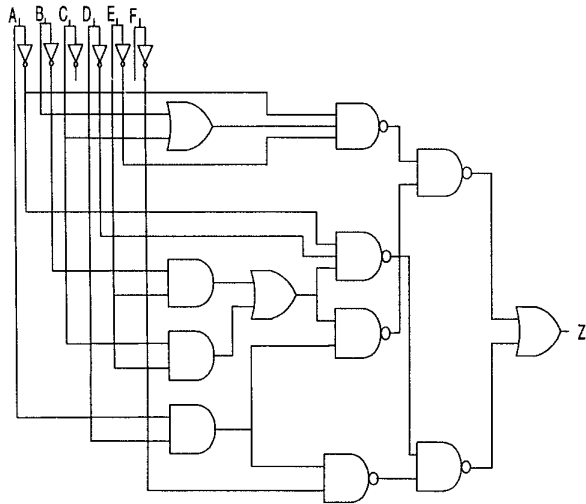


Fig. 4 Example circuit C2.

Table 3 illustrates the results obtained when random multiple faults are inserted into the circuit of C17. The results reflect the diagnostic performance for one fault detection, two faults detection and the number of incorrectly detected faults. Each combination of random faults are tested using the three data selection types mentioned above. For each data selection type, twenty different examples of testing data with randomly selected multiple faults are used. The results in Table 3 represent the average. Considering the case of single fault detection, the diagnostic performance of the ANN is 100% in both cases when all possible input combinations and the test vectors are used. When 50% of the possible input combinations are used the diagnostic performance is 95.65%. For the two fault detection case the diagnostic performance of the ANN is 88.8% when all possible input combinations are used, 67.3% when 50% of the possible input combinations are used, and 89.3% when test vectors are used. This is a significant improvement on the initial results published in [1].

ANN size In:Mid:Out	1 Fault detected (%)	2 Faults detected (%)	Incorrectly detected
352:95:23	100	88.8	0
176:65:23	95.65	67.3	0
44:35:23	100	89.3	0

Table 3 Fault diagnosis performance of ANN in C17

Table 4 shows the diagnosis performance with one fault and two faults for circuits C1, C2 and C17 in the case of using test vectors. It is clear from the table that the diagnostic performance of one fault is 100% in all circuits and the diagnostic performance of two faults are 87.9%, 76.8% and 89.3% for C1, C2 and C17 respectively. At the same time the number of incorrectly detected faults in all circuits is zero.

Fault detection	C1	C2	C17
1 fault detected (%)	100	100	100
2 faults detected (%)	87.9	76.8	89.3
Incorrectly detected	0	0	0

Table 4 ANN diagnostic performance using test vectors

Table 5 shows the desired (Des) and actual (Act) outputs of the ANN diagnostic system for the circuit C17 when using test vectors to construct the FTT with groups of two randomly selected faults. A comparison of the desired and actual outputs for each inserted fault (indicated in bold text) is performed. If the desired output for a given fault, is greater than a threshold value (in our case chosen to be 0.4) then the fault is correctly diagnosed. The diagnosis statistics are summarised towards the bottom of table 5. These show the fact that at least one fault from the two randomly inserted faults in the circuit is always detected, while in 72.73% of cases both faults are correctly diagnosed.

6. CONCLUSION

A technique is proposed for the use of ANNs for single and multiple stuck at fault diagnosis. When the ANN is trained with data derived from circuit test vectors, a compact, flexible and efficient ANN architecture is produced which provides significant improvement in multiple fault diagnosis. The technique is demonstrated with a number of example circuits.

6. REFERENCES

- [1] T. Arslan, and A. AL-Jumah, "A compact artificial neural network approach for multiple fault location in digital circuits", *Electronics Letters*, 33, 1801-1803, 1997.
- [2] S. Stephen, and Y. Shih-chien, "Multiple fault detection of combinational logic circuits", *IEEE Trans. on Computers*, c-24, 233-242, 1975.

- [3] T. Kuo, J. Wang, and J. Lee, "Multiple fault detection using single-fault tests", *Electronic Letters*, 27, 1329-1330, 1991.
- [4] J. Wen-ben, and H. Patrick, "Multiple fault testing using minimal single fault test set for fanout-free circuits", *IEEE Trans. Computer-Aided Design*, 12, 149-157, 1993.
- [5] K. Lai, and P. Lala, "Multiple fault detection in fan-out free circuits using minimal single fault test set", *IEEE Transaction on Computers*, 45, 763-765, 1996.
- [6] T. Arslan, A. AL-Jumah, and H. Alkadim, "Functional fault diagnosis of mixed analogue/digital circuit boards using artificial neural networks", *Control '97*, Mexico, 1997.
- [7] B. Wilkins, *Testing digital circuits*, U.K, Van Nostrand Reinhold co. ltd, 1986.
- [8] B. Kagle, J. Murphy, L. Koos, and J. Reeder, "Multi-fault diagnosis of electronic circuit boards using neural networks", *IJCNN*, Washington, II-197-II-20, 1991.
- [9] H. Van Der Linden, "Automatic test pattern generation for three-state circuits", *PhD thesis*, Delft university of Technology, 1996.
- [10] *Neuralworks professional II/PLUS user's guide*, Neural Ware Inc., USA, 1993.
- [11] F. Hill, and G. Peterson, *Computer aided logical design with emphasis on VLSI*, Canada, John Wiley & Sons, 1993.

	ff	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	f20	f21	f22	
Des	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Des	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0.1	0.1	0.1	0.8	0.1	0	0.1	0	0.1	0.1	0.8	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.1	0.1	
Des	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0.1	0.9	0.1	0.2	0	0	0.1	0.3	0	0	0	0.1	0.1	0.2	0.2	0.2	0	0	0.1	0	0.1	0.1	0.1	
Des	0.1	0.1	0.1	0.9	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0	0.1	0.1	0.8	0.6	0	0.1	0	0.1	0.2	0.2	0.1	0	0.1	0.1	0.1	0.2	0.1	0	0.2	0.2	0.2	0.2	
Des	0.1	0.1	0.9	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0.1	0.1	0.8	0	0.7	0.1	0.1	0	0	0.1	0.1	0.1	0.1	0.1	0.1	0	0.2	0.2	0.3	0.3	0.1	0.2	0.1	
Des	0.1	0.1	0.1	0.9	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0	0.1	0.1	0.9	0.1	0.1	0.5	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.1	0.2	0.1	0.1	0.2	0	0.1	
Des	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0.1	0.8	0.2	0.1	0.1	0.1	0	0.1	0.1	0.8	0.1	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0	0.1	0	0.1
Des	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0	0.1	0.7	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.8	0.1	0.1	0.1	0.3	0.1	0.1	0	0	0.1	0.1	0	0	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.8	0.1	0	0.1	0.2	0.1	0.2	0.1	0	0.1	0	0.1	0
Des	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0.1	0.1	0.1	0.8	0.1	0.1	0.2	0.1	0.2	0.4	0	0	0.1	0	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.2	
Des	0.1	0.9	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0	0.3	0.1	0.8	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.2	0.1	0.1	0.1	0	0.2	0.1	0.2	0	0	0.1	0.2	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.9	0.1	
Act	0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0	0.2	0	0.2	0.6	0	0.1	0.1	0	0.1	0	0.6	0.2	0.2	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.9	0.1	0.1	0.1	
Act	0	0.1	0.1	0.1	0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.7	0.1	0.2	0.1	0.5	0.1	0.2	0.1	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.9	
Act	0.1	0.1	0.1	0.1	0.1	0.1	0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.1	0.1	0.8	0.8	0.1	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	
Act	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	
Act	0.1	0.1	0.2	0.1	0.1	0.2	0.1	0.2	0.2	0.1	0.2	0.1	0.1	0.2	0.2	0.1	0.1	0.2	0.1	0.5	0.1	0.9	0.1	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.9	0.1	0.1	0.1	0.1	0.1	
Act	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.2	0.1	0.2	0	0.1	0.9	0.2	0.7	0.2	0.1	0.2	0.1	0.1	0.1	0.1	0.1	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.9	0.1	
Act	0.1	0.1	0.1	0.1	0.1	0	0.2	0.1	0.2	0.1	0.1	0.2	0.1	0.2	0.1	0.7	0.1	0.2	0.1	0.1	0.1	0.9	0.1	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.1	0.1	
Act	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.2	0.2	0.1	0.8	0.1	0.1	0.2	0.1	0.1	0.1	0.1	0	0.1	0.1	0.1	0.1	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	
Act	0.1	0.1	0.1	0.1	0.1	0.1	0	0.1	0.1	0.1	0.1	0.7	0.1	0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.8	0.2	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.1	0.1	0.1	0.9	0.1	
Act	0	0.1	0.1	0.1	0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.7	0.1	0.2	0.1	0.3	0.1	0.2	0.1	
Des	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.1	0.1	0.9	0.1	0.1	
Act	0.1	0.2	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0	0	0.1	0	0.7	0.1	0	0.7	0.2	0.2	

number of diagnoses where at least one fault is correctly diagnosed = 22
 number of diagnoses where two faults are correctly diagnosed = 16
 percentage of diagnoses where at least one fault is correctly diagnosed = 100%
 percentage of diagnoses where two faults are correctly diagnosed = 72.73%
 number of diagnoses where faults are incorrectly diagnosed = 0
 percentage number of diagnoses where faults are incorrectly diagnosed = 0.0

Table 5 The ANN performance with C17.