Use of Neural Networks in Testing Analog to Digital Converters

K. MOHAMMADI, S. J. SEYYED MAHDAVI
Department of Electrical Engineering
Iran University of Science and Technology
Narmak, 16844,
Tehran, Iran

Abstract: - In the past two decades, the techniques of artificial neural networks are growing mature, as a data-driven method, which provides a totally new perspective to fault diagnosis. Testing issues are becoming more and more important with the quick development of both digital and analog circuit industry. Analog-to-digital converters (ADCs) are becoming more and more widespread owing to their fundamental capacity of interfacing analog physical world to digital processing systems. In this paper, we study the use of neural networks in fault diagnosis of ADCs and compare the results with other ADC testing approaches such as histogram, FFT and sine fit test techniques. In this paper, we study the use of neural networks in fault diagnosis of ADCs and compare the results with other ADC testing approaches such as histogram, FFT and sine fit test techniques. In this paper, we introduce two ideas to improve the training phase time. They are separation and indexing of neural network outputs. Finally, we concluded that neural network approach is a robust way for fault diagnosis of ADCs and also other mixed signal circuits.

Key-Words: fault, ADC, neural networks, test, histogram, FFT, sine fit, training time, output separation.

1 Introduction

With the continuing advances in digital technology, now with deep-submicron devices and wires switching at such high frequencies that RF effects appear, it becomes increasingly hard to maintain a clear separation between the digital abstraction from 1's and 0's upward and the underlying physical world which is characterized by analog behaviors [1].

While digital test and diagnosis techniques have reached a fair level of maturity, analog test strategies are still evolving. Analog circuit testing is made particularly difficult by the following reasons:

- The simulation complexity of analog circuits is significantly greater than the complexity of simulating comparable digital circuits. This problem is aggravated by the fact that there do not exist, at the current time, commercial concurrent fault simulators for analog circuits. Hence, analog faults must be simulated sequentially, leading to impractical overall simulation times.
- Analog circuit's parameters can assume any value, under fault, on a continuous scale from 0 to ∞. In practice, it is not possible to simulate the circuit under test (CUT) over the entire range of possible component values. Moreover, the relationships between the component deviations under fault and the circuit specifications are often highly non-linear. This makes fault diagnosis over all fault conditions, very difficult [2].

Analog-to-digital converters (ADCs) are becoming more and more widespread owing to their fundamental capacity of interfacing analog physical world to digital processing systems. In this development, a pre-eminent role has been played by testing activity, because the actual ADC metrological behavior is defined and verified by setting up specific figures of merit and suitable experimental techniques.

In the past two decades, the techniques of artificial neural networks are growing mature, as a data-driven method, which provides a totally new perspective to fault diagnosis [3]. Neural networks have been trained to perform complex functions in various fields of application including pattern recognition, identification, classification, speech, vision and control systems. In this paper, we study the use of neural networks in fault diagnosis of ADCs

2 Neural networks

The field of neural networks has a history of some five decades, but has found solid application only in the past fifteen years and the field is still developing rapidly. Thus, it is distinctly different from the fields

of control systems or optimization where the terminology, basic mathematics, and design procedures have been firmly established and applied for many years.

Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. So such an element is called a 'neuron'. Fig. 1 shows a simple neuron. As in nature, the network function is determined largely by the connections between elements. We can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements. Commonly neural networks are adjusted, or trained, so that a particular input leads to a specific target output. Such a situation is shown in Fig. 2.

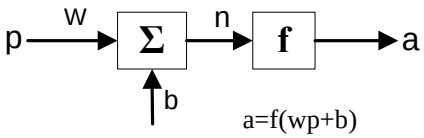

Fig. 1: Model of a simple neuron

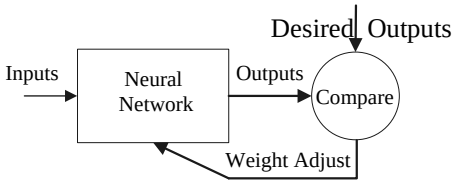

Fig. 2: Model of a neural network

The network is adjusted, based on a comparison of the output and the target, until the network output matches the target. Typically many such input/target pairs are used, in this supervised learning, to train a network. There are several supervised learning algorithms which 2 important ones are back propagation algorithm and radial basis function network. In this paper, we study the use of these two algorithms in fault diagnosis [4, 5].

2.1 Back Propagation Algorithm (BP)

Back propagation was created by generalizing the Widrow-Hoff learning rule to multiple-layer networks (MLP) and nonlinear differentiable transfer functions. Standard back propagation is a gradient descent algorithm, as is the Widrow-Hoff learning rule, in which the network weights are moved along the negative of the gradient of the performance function. The term *back propagation* refers to the manner in which the gradient is computed for nonlinear multilayer networks. Standard algorithm for adjusting weights and biases uses the following equations:

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu\mathbf{e}(k)\mathbf{p}^T(k) \quad (1)$$

$$\mathbf{b}(k+1) = \mathbf{b}(k) + 2\mu\mathbf{e}(k) \quad (2)$$

where **w**, **b**, **e**, **p** and μ represent weight matrix, bias matrix, error, input matrix and learning rate.

2.2 Radial Basis Function Networks (RBF)

The idea of radial basis function (RBF) networks derives from the theory of function approximation. RBF networks take a slightly different approach from BP. They are two-layer feed-forward networks. The hidden nodes implement a set of radial basis functions (e.g. gaussian functions). The output nodes implement linear summation functions as in MLP. The network training is divided into two stages, first the weights from the input to hidden layer are determined, and then the weights from the hidden to output layer. RBF training/learning is very fast. The RBF network overcomes drawbacks of the MLP network by using non-monotonic transfer function based on the gaussian density function. It is the radial basis function that is a set of non-linear functions built into one function. The radial basis function uses hyper ellipsoids to partition the pattern space. The function "s" in a "k" dimensional space has elements "s_k" which partition the space [4, 5, 6, 7]:

$$s_k = \sum_{j=1}^{m} \lambda_{jk} \Phi(\| x - y_j \|) \quad (3)$$

where λ_{jk}, Φ, $\|x - y_j\|$, x and y_j represent strength of connection, radial basis function, distance measure, input of the network and centre of the hyper ellipse.

3 ADC fault diagnosis using neural networks

In this paper, we use a 3 bit flash ADC [8]. Fig. 3 shows a schematic of flash ADC. The fault models that were applied to the circuit consist of single stuck at fault (stuck at 0 and stuck at 1), open, bridge [9] and tolerance fault [7, 10]. In single stuck at faults, a node of the circuit sticks to the voltage of ground or voltage of source. The former is called stuck at 0 (SA0) and the latter is called stuck at 1 (SA1). In bridge fault model, two nodes of circuit are shorted together. In tolerance fault model, according to the test engineer opinion, for a specific parameter of a device (such as resistance), some amount of tolerance would be accepted and deviation more than this tolerance would be fault. . 18 places for stuck at, 32 places for open, 56 places for bridge, and 8 places for tolerance fault were identified in the 3 bit flash ADC

circuit. We specified a number for each of these fault places. Table 1 shows the numbers that indicate each place of occurring stuck at fault in the circuit.

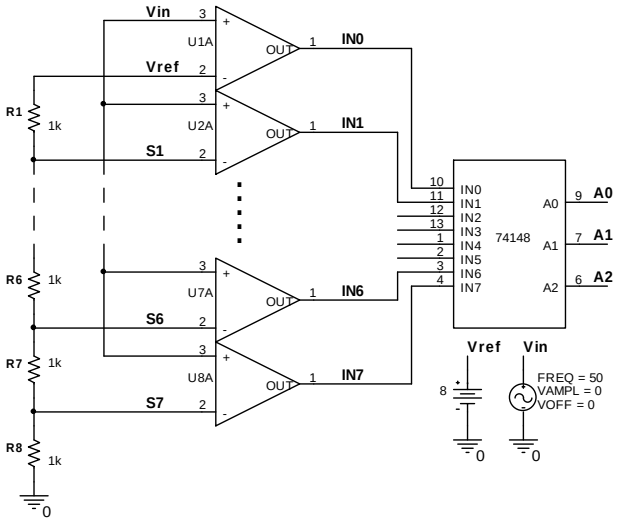

Fig. 3: A simple flash ADC

For each of 5 fault models, fault data bank was created by changing the input voltage from zero to 8.1V (little more than full scale voltage), using Orcad 9.2. Totally, 5454 samples for all of fault models were obtained. The voltage of 19 nodes of circuit and the 5 fault model were considered as neural network inputs and outputs.

Table 1: Numbers for each Place of Occurring Stuck At Fault in the Circuit

Node Name	S1	S2	S3	S4	S5	S6
Fault No.	1	2	3	4	5	6
Node Name	S7	IN0	IN1	IN2	IN3	IN4
Fault No.	7	8	9	10	11	12
Node Name	IN5	IN6	IN7	A0	A1	A2
Fault No.	13	14	15	16	17	18

In training phase, we observed that applying all the samples together to the network is an impractical work. Long training time, possible stoppage of training in an inappropriate SSE (sum square error) in MLP networks and insufficient neuron number for reaching desirable SSE are some reasons of this. So for improving the training time of the neural network, we used the ideas of outputs separation and indexing.

3.1 Separation of neural network outputs

Fig. 4 shows the flowchart of this technique. For example, SA0 fault can occur in 18 nodes of the ADC circuit. In the case of outputs separation, we divide the desired outputs of SA0 fault model into 18 classes. In class number 1, we convert all of outputs which were not equal to 1, into zero. In class number 2, we convert all the outputs that are not equal to 2 into zero and others into 1. We continued this work for all other classes. Then, the outputs of each class that belong to bound of [0 1], were rounded to their nearest integer value. Now, outputs can be only 0 or 1. Then, we multiplied outputs of each class by its class number (For example, outputs of class number 2, were multiplied by 2) .Then, we summed all 18 class outputs and we had one total network.

Experiment show that this method decreases the training time to less than 1 percent for MLP networks and less than 50 percents for RBF networks beside the primary case. Furthermore, when we tested the networks, we observed that the error of neural network in finding the location of the fault reaches zero. Whereas, error of neural network without output separation never reaches zero. So, we see that this method increases the network generalization, even. Table 2 compares the results of RBF and MLP network in the case of using separation of outputs. Fig. 5 shows the structure of the MLP neural network that was used for training each separated network. We used a network with a five neurons hidden layer.

Table 2: RBF and MLP networks results with separation of outputs

	MLP Network	RBF Network
Network Error ($E_{separation}/E_{normal}$)	0.004 %	0.058 %
Training Time ($t_{separation}/t_{normal}$)	0.68 %	50 %
No. of Neurons ($n_{separation}/n_{normal}$)	28 %	126 %

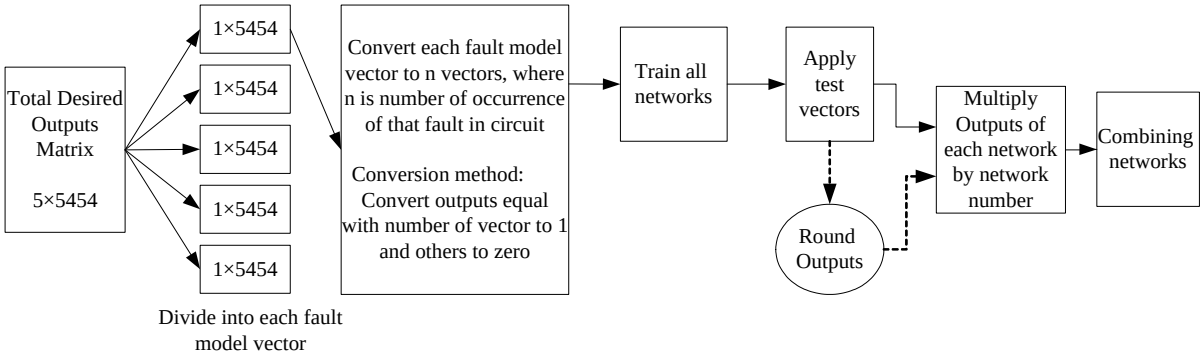

Fig. 4: Separation technique flowchart.

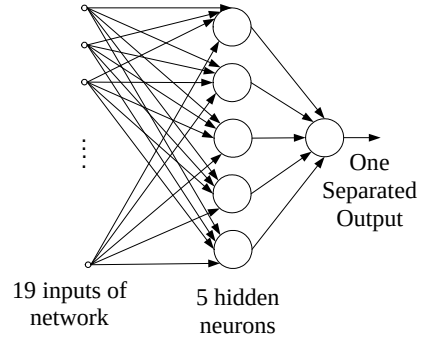

Fig. 5: Structure of the MLP neural network that was used for training each separated network.

For testing the network, we chose 40 test vectors that consist of 10 test vectors for each of SA0, SA1, open and tolerance, at random and applied them to the network. Fig. 6 shows the error of MLP network of each of 5 fault models after applying the 40 test vectors. We see that the network of SA0 fault model, has correctly anticipated the location of fault for all test vectors. But the network of SA1 fault model has error (-1) for test vector number 36. This test vector is obtained by applying tolerance fault to resistance R1 in the circuit. The interesting point is that this error tells us, the test vector number 36 is similar to input patterns for SA1 fault in node S1 (the inferior node of R1) in the circuit. So, neural network can tell us that the test vector may belong to which other input pattern of other fault models. Of course, we should indicate that the neural network has not correctly anticipated the location of fault in 3 test vectors (about 7.5 percents of all cases). In 2 of this 3 test vectors, the neural network anticipated the circuit is fault free.

3.2 Indexing of neural network outputs

To perform the indexing of neural network outputs, we should convert each number in desired output vector to its index. For better understanding this concept, pay attention to the following example. If vector x is in form of x = [0, 2, 1], then its index matrix will be like this:

$$x = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad (4)$$

For example, SA0 fault can occur in 18 nodes of the ADC circuit. So, indexing of neural network outputs, converts the desired output vector of SA0 fault model to 5454×18 matrix. The first advantage of indexing is that amount of error of all samples in calculating SSE, would be equal. We anticipate that indexing would have the following two more advantages:

- Using indexing, neural network can identify the occurrence of 2 or more faults in the circuit.
- Because of enlargement of neural network output layer and also because most of elements of desired output matrix are zero (the maximum number of ones is equal to the number of columns of desired output matrix, so always more than 94.45 percents of desired output elements are zero.), the network would need less neurons in middle layers and it would converge faster.

Experiment show that this method decreases the training time to less than 1 percent for MLP networks. But this method can't improve the training phase time for RBF networks. Table 3 compares the results of RBF and MLP network in the case of using indexing of outputs.

Table 3: RBF and MLP networks results with indexing of outputs

	MLP Network	RBF Network
Network Error ($E_{separation}/E_{normal}$)	0.3 %	8.3 %
Training Time ($t_{separation}/t_{normal}$)	0.8 %	103.8 %

In the case of existence of 2 faults in the circuit, this method can correctly identify one of fault locations in all input test vectors. But without indexing, network can correctly identify one of fault locations only in 30 percents of cases. Indexing method can correctly identify both fault locations in 20 percents of cases. So we see that indexing method can improve the network generalization, too. Table 4

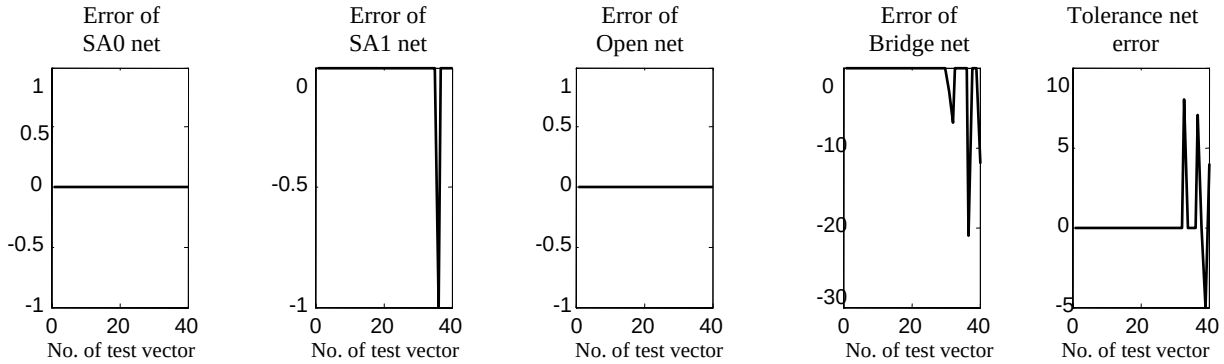

Fig. 6: Separation technique. Error of applying test vectors to MLP networks of fault models a) SA0 b) SA1 c) Open d) Bridge e) Tolerance

compares the results of RBF and MLP network in the case of occurrence of 2 faults in the circuit.

Table 4: RBF and MLP Networks Results with 2 Fault Occurrence in the Network

		Locate 1 of 2 faults correct	Locate both faults correct
MLP	Normal	30 %	0 %
	Separation	50 %	0 %
	Indexing	100 %	20 %
RBF	Normal	30 %	0 %
	Separation	30 %	0 %
	Indexing	50 %	0 %

Fig. 7 shows the error of MLP network of each of 5 fault models after applying the 40 test vectors. We see that like separation method, this technique has the ability of finding similarity between the test vector and patterns of other fault models. We also see that this method has not correctly anticipated the location of fault in 1 test vector (about 2.5 percents of all cases). So, in this method, network has less error than separation technique.

In general, we can conclude that besides neural network can locate the fault with high reliability; it identifies other possible places that the test vector may belong to. So we can say this method is a robust way for fault diagnosis in testing mixed signal circuits.

4 Conclusion

In this paper, we studied different ADC test approaches. Stuck at, open, bridge and tolerance faults were applied to a 3 bit flash ADC. For each of fault model, a data bank was created. Using these fault data banks, neural network was trained. In this paper, we introduced the ideas of separation and indexing of neural network outputs to improve the training phase time. Experiment show that these methods decrease the training time to less than 1 percent for MLP networks beside the primary case. Furthermore, when we tested the networks, we observed that the error of neural network in finding the location of the faults reaches zero. Whereas, error of neural network without output separation never reaches zero. We observe that these methods increase the network generalization.

Table 5: Generalization and Equivalent Fault Detection of RBF and MLP Networks

		generalization	Equivalent fault
MLP	Separation	92.5 %	Yes
	Separation & Rounding	92.5 %	Yes
	Separation and Indexing	97.5 %	Yes
RBF	Separation	60 %	No
	Separation & Rounding	80 %	No
	Separation and Indexing	92.5 %	Yes

Table 5, compares RBF and MLP networks generalization capability. We see that when indexing and separation of outputs are used together, generalization reaches 97.5 %.

We also simulated other ADC testing approaches such as histogram, FFT or sine fit. 500 samples were taken using PSPICE in Orcad 9.2 for each case of applying each fault model to possible places in the circuit. We observed that these methods have the advantage of being simple and fast. These methods are successful in finding out if the circuit is faulty or not. But they weren't useful methods for fault location analysis, while neural network approach is able to determine the location of fault, too.

In general, we can conclude that beside neural network can locate the fault with high reliability; it identifies other possible places that the test vector may belong to. So we can say this method is a robust way for fault diagnosis in testing mixed signal circuits.

References

[1] Peter B.L. Meijer, “Neural Networks for Device

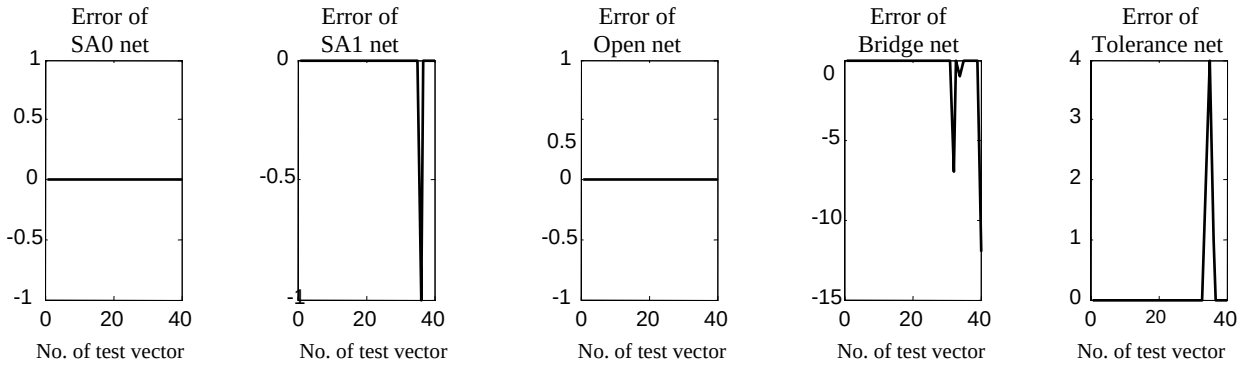

Fig. 7: Indexing technique. Error of applying test vectors to MLP networks of fault models a) SA0 b) SA1 c) Open d) Bridge e) Tolerance

and Circuit Modeling", *3rd International Workshop Scientific Computing in Electrical Engineering,* 2000

[2] S. Chakrabarti, S. Cherubal, A. Chatterjee, "Fault diagnosis for mixed signal electronic systems", *Proceedings of IEEE Aerospace Conference,* 1999, 169-179 vol.3

[3] P. Arpaia, F. Cennamo, P. Daponte, "Metrological characterization of analog-to-digital converters-a state of the art", *3rd Advanced A/D and D/A Conversion Techniques and Their Applications, IEE ADDA-99*, Glasgow (UK), 26-28 July 1999, pp. 134-144

[4] Howard Demuth, Mark Beale, Neural Network Toolbox for Use with MATLAB, MATLAB User's Guide Version 4

[5] Simon Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall, 1999

[6] I. Dalmi, L. Kovacs, I. Lorant, G. Terstranszky, "Adaptive learning and neural networks in fault diagnosis", *UKACC International Conference on Control '98,* Conference Publication No.455, IEE 1998

[7] Mohammadi. K., Mohseni. A. R., "Fault diagnosis of analog circuits with tolerances by using RBF and BP neural networks", *2002 Student Conference on Research and Development Proceedings,* Shah Alam, Malaysia

[8] Mandeep Singh, Israel Koren, "Fault Sensitivity Analysis and Reliability Enhancement of Analog-to-Digital Converters", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* Volume 11, Issue 5 (October 2003), Pages: 839 - 852

[9] Samiha Mourad, Yervant Zorian, *Principles of Testing Electronic Systems,* John Wiley & Sons, Ltd, 2000

[10] Ying Deng; Yigang He; Yichuang Sun, "Fault diagnosis of analog circuits with tolerances using artificial neural networks", *The 2000 IEEE Asia-Pacific Conference on Circuits and Systems,* 2000, IEEE APCCAS 2000