Proceedings of 1993 International Joint Conference on Neural Networks

An Analog Neural Network Chip with Random Weight Change Learning Algorithm

Kenichi Hirotsu*, Martin A. Brooke
School of Electrical Engineering
Georgia Institute of Technology, Atlanta, Georgia 30332

Abstract

Although researchers have been engaged in fabrication of neural network hardware, only a few networks implemented with a learning algorithm have been reported. A learning algorithm is required to be implemented on a VLSI chip because off-chip learning with a digital computer consumes too much time to be applied to many practical problems. The main obstacle to implement a learning algorithm is the complexity of the proposed algorithms. Algorithms like Back Propagation include complex multiplication, summation and derivatives, which are very difficult to implement with VLSI circuits. The authors propose a new learning algorithm, which is suitable for analog implementation , and implemented it on a 2.2 mm x 2.2 mm neural network chip with 100 weights, using the standard 2.0 μm MOSIS process. The chips have successfully learned the XOR Gate problem.

I. Introduction

A number of researches have been engaged in application of neural networks with digital computer simulation in various areas. However, the serial operation of digital computers causes slow processing speed, making neural networks impractical for some real world problems. Neural networks have been implemented with analog or digital integrated circuits [1]-[4] to improve the signal processing speed. This kind of hardware normally has adjustable weight storage circuits, in which weight values are loaded after the values are determined by calculation with an external digital computer. Although this off-chip learning type hardware improved the processing speed significantly, the very long learning period with a digital computer has limited its application. Recently, some researchers have fabricated on-chip learning neural networks. Arima et al. implemented a revised Boltzmann Machine learning algorithm with analog/digital hybrid circuitry[5],. Morie et al. fabricated an analog neural network chip with Back Propagation algorithm[6]. Yasunaga et al. and Shima et al. [7],[8] employed digital circuitry for implementation of Back Propagation or Hebbian learning. These traditional learning algorithms are expressed with complex equations and require complicated multiplication, derivatives or simulated annealing, which increase the complexity of the circuits, consume large areas with digital circuitry, and are significantly degraded by the non-idealities of analog circuitry.

The authors proposed a simple learning algorithm for analog implementation, called Random Weight Change algorithm, which is immune to the non-idealities of analog circuits[9]. They fabricated 100 weights with on-chip learning and the performance was tested.

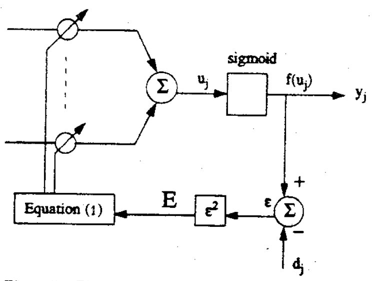
II. Random Weight Change Learning Algorithm

Although Back Propagation is commonly adopted for the learning of feedforward neural networks, it is not suitable for analog implementation because of its vulnerability to the non-idealities of analog circuits, especially offset. To avoid these problems, a new learning algorithm with no multiplication, which is usually the origin of the non-idealities, was proposed[9].

The new learning algorithm is described in Fig. 1 and Equation (1). Weights are changed randomly from the initial state with a small increment of + δ or - δ. If the summation of the errors at the outputs from the desired values decreases by the weight change, the same weight change is iterated until the error increases. If the summation of the errors increases, weights are updated randomly again. The summation of the output errors can be calculated off the chip with a digital computer. Because a digital computer is fast enough for simple summation only at the output layer, this summation is not necessary on chip.

* On educational leave from Sumitomo Electric Industries, Ltd., Osaka, Japan

This work is supported in part by Analog Devices Inc.



$$\left.\begin{array}{ll} W_{ij}(n+1) = W_{ij}(n) + \Delta W_{ij}(n+1) & \\ \Delta W_{ij}(n+1) = \Delta W_{ij}(n) & : \text{if } E(n+1) < E(n) \\ \Delta W_{ij}(n+1) = \delta * \text{Rand}(n) & : \text{if } E(n+1) \geq E(n) \end{array}\right\} \text{Equation (1)}$$

(Rand(n) : Random function which has only two states ± 1)

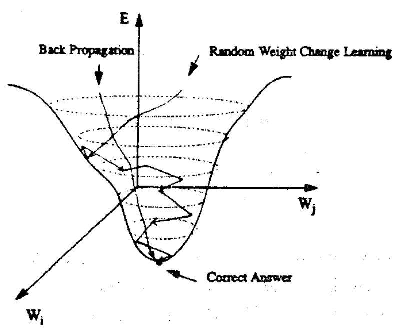


Fig. 1. Random Weight Change Learning **Fig. 2. Comparison with Back Propagation**

Fig. 2 makes a comparison of the algorithm with Back Propagation. During Back Propagation learning, the operating point goes down along the steepest slope of the energy curve of the network. For Random Weight Change, the operating point goes up and down on the energy curve rather than descends straight along the steepest slope, however, statistically descends and finally reaches the correct answer.

This learning algorithm is less efficient than Back Propagation on digital computer simulation, however, it is good for analog implementation, because it needs no multipliers which cause offsets or other non-idealities, and very simple circuits for weight updates are available, which have only two states of weight change, $+\delta$, $-\delta$. Moreover, the algorithm does not require any specific network structure. It can be applied to other than feedforward networks, although fully connected feedback networks do suffer due to local minimum problems.

III. Design of the network

Weight and non-linear (neuron) circuits were designed with analog integrated circuits. Very simple and small circuits were adopted because Random Weight Change learning allows large non-idealities of analog circuits.

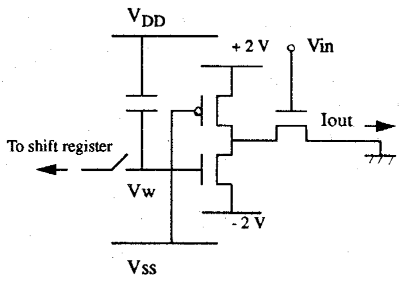
(a) Weight Circuit

A two quadrant multiplier with an analog memory was designed as a weight circuit. A weight value is stored on a capacitor in analog way as shown in Fig. 3. The equation for the circuit is expressed in Equation (2) when all the MOS transistors are in non-saturation region.

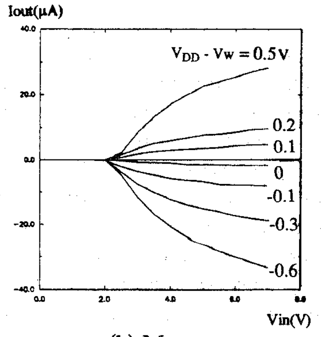
$$I_{out} = \beta\,(V_{in} - V_{tn})(V_{DD} - V_w) \qquad \text{- - - - - Equation (2)}$$

Vin : Input	Vtn : Threshold voltage of NMOS
Vw : Weight value	β : Constant

Measurement results are shown in Fig. 3 (a), (b). Although the characteristic is not linear, Iout is monotone increasing for both Vin and (V_{DD} - Vw).



(a) Circuit

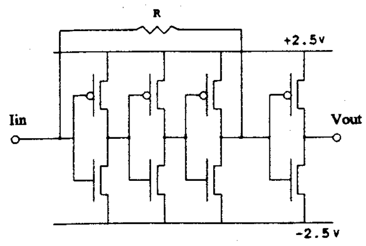


(b) Measurement

Fig. 3. Weight Circuit

(b) Non-linear (neuron) Circuit

A non-linear circuit is shown in Fig. 4(a). The output is bounded and represents a monotone increasing function like sigmoid. The measured characteristic is shown in Fig. 4(b).



(a) Circuit

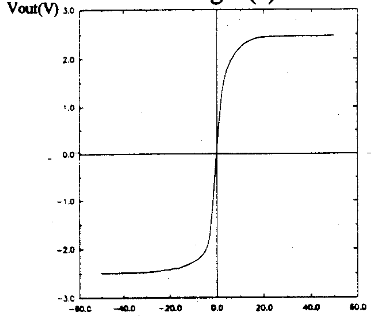
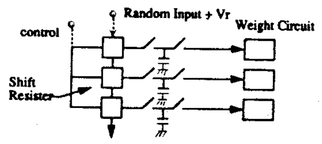


Fig. 4. Non-linear(neuron) Circuit (b) Measurement

(c) Random Function Generator

Random value $\pm\delta$ is generated with a shift register. A random pulse Vr or -Vr is applied at an edge of the shift register and the pulse propagates toward the other edge as shown in Fig. 5. After the first pulse reached the edge, no arbitrary two terminals are logically related each other. Random weight change $\pm\delta$ is given to each weight circuit through a switched capacitor.



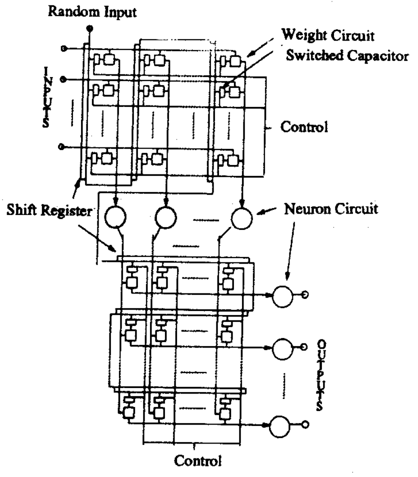


Fig. 5. Random Weight Change Generation Fig. 6. Network Structure

The whole network is explained in Fig. 6. Weight chips and neuron chips are fabricated separately and are assembled as described in the figure. Any network structure can be realized by assembling these two types of the chip and larger networks are available by cascading them. The specification of the two chips is described in Table. 1, 2.

Table 1: Weight Chip

Size	2.2 x2.2 mm
Number of weights	100 (10 x 10)
Operation speed	< 100 ns
Time for weight change	< 100 ns
Supply voltage	$\pm$5 V
Size of a weight circuit	150 x 150 um

Table 2: Neuron Chip

Size	2.2 x 2.2 mm
Number of neurons	18
Operation speed	< 100 ns
Size of a neuron circuit	60 x 80 um

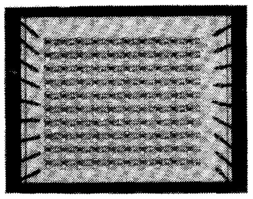


Photo 1. Weight Chip

IV. Results

The weight and neuron chips were fabricated at MOSIS with 2.0 μm CMOS double poly process. The picture of the weight chip is shown in Photo. 1. Those chips were tested and the performance was evaluated. The operating speed, the weight update time and the weight retention time were measured and the XOR problem was learned on the chip. A pulse voltage was applied to an input of the two layered feedforward network and the operating speed of 100 ns was obtained by measuring the output response. The weight update time and weight retention time are 100ns and 2s (time for losing 1% of the weight value), respectively. Assuming a network of 10 inputs x 10 hidden layer neurons x 10 outputs is constructed, 2 GCPS is achieved.

XOR gate problem was learned successfully on the network of 2 inputs x 2 hiddens x 1 output. A typical example of the output error decrease during the learning is described in Fig. 7. The iteration times for reaching the output error of 30 % (maximum error for the whole input patterns) was 520 to 1130 for nine tests according to the initial weights, while simulation results showed 560 to 870. Larger problems are being tested now.

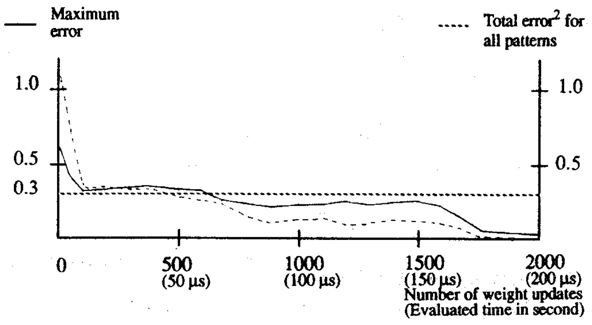


Fig. 7. Error during Learning

V. Conclusion

A neural network with a new learning algorithm, Random Weight Change learning, was successfully implemented with analog integrated circuits. 2.0 µm CMOS double poly process was employed for fabrication at MOSIS and operating speed of 100 ns, weight update time of 100 ns, and weight retention time of 2 s were achieved. XOR gate problem was presented to the network and the learning was performed successfully, demonstrating the validity of the learning algorithm for analog implementation.

References

[1] M. Holler et al., "An Electrically Trainable Artificial Neural Networks (ETANN) with 1024 'Floating Gate Synapses'", International Joint Conference on Neural Networks, vol. II, pp. 191, 1989.

[2] M. Yasunaga et al., " Design, Fabrication and Evaluation of a 5-inch Wafer Scale Neural Network LSI composed of 576 Digital Neurons", International Joint Conference on Neural Networks, vol. II, pp. 527, 1990.

[3] A. F. Marray, " Pulse-Stream VLSI Neural Networks, Mixing Analog and Digital Techniques", IEEE Transactions on Neural Networks, vol. 2, No. 2, pp. 193, 1991.

[4] S. Satyanarayana, et al., " A Reconfigurable VLSI Neural Network", IEEE Journal of Solid-State Circuits, vol. 27, No. 1, pp. 67, 1992.

[5] Y. Arima et al., " A self-learning Neural Network Chip with 125 Neurons and 10K Self-organization Synapses", IEEE Journal of Slid-State Circuits, vol. 26, no. 4, pp. 607, 1991.

[6] T. Morie et al., " Analog VLSI Implementation of adaptive algorithms by an extended Hebbian synapse circuit", IEICE Transactions on Electronics, vol. E-75, no. 3, pp. 303, 1992.

[7] T. Shima et al., " Neuro Chips with On-Chip Back-Propagation and/or Hebbian Learning", IEEE Journal of Solid-State Circuits, vol. 27, no. 12, 1992.

[8] M. Yasunaga et al., " A Self-Learning Digital Neural Network Using Wafer-Scale LSI", IEEE Journal of Solid-State Circuits, vol. 28, no. 2, pp. 106, 1993.

[9] K. Hirotsu and M. Brooke, " A New Learning Algorithm for Analog Neural Network Implementation", Neural Networks, submitted.