Modular Analog Continuous-Time VLSI Neural Networks with On Chip Hebbian Learning and Analog Storage

B. Linares-Barranco[1], E. Sánchez-Sinencio[2], A. Rodríguez-Vázquez[1], and J. L. Huertas[1]
[1]Centro Nacional de Microelectrónica (CNM), Ed. CICA, Avda. Reina Mercedes s/n, 41012 Sevilla, SPAIN
[2]Dept. of Electrical Engineering Texas A&M University, College-Station, TX77843, U.S.A.

Abstract

A modular analog circuit design approach for hardware implementations of neural networks is presented. This approach is based on the use of small transconductance multipliers as the main component, and is therefore called the T-mode (Transconductance-mode) approach. This circuit design technique will be used to design a set of modular chips, which will be assembled to build either BAM networks, Hopfield networks, Winner-Take-All networks, or simplified ART1 networks. The approach will be extended afterwards in order to include a hebbian learning rule into each synapse. As an example, a learning BAM network system will be shown. The experimental results given were obtained from 2μm CMOS double-metal double-polysilicon (MOSIS) prototypes.

I. Introduction

There are many neural network algorithms available in the computer science related literature [1]-[14], most of which have been studied and implemented in software environments. However, for applications where real time processing is necessary and/or the size of the complete computing system needs to be reduced, some type of special purpose hardware implementation technique needs to be devised. In particular, analog circuits capability for intrinsic high-speed operation with moderate area and power consumption [15] make these techniques worth to be explored in connection to neural networks.

One of the main drawbacks of analog IC design is its high dependence on components and process parameters, and the consequent need for calibration, on chip tuning and/or temperature variations compensations circuitries. However, in adaptive neural networks those imperfections can be largely tolerated because as the system learns to perform a certain task it implicitly compensates for nonidealities present in the physical components the whole system is made of [16].

Several analog [17]-[18] or mixed analog-digital [19] hardware techniques have already been proposed, some of them including learning capability [20]. In the approach we present [21], [22] modularity is a property (which also happens in [20]), as well as the possibility of reconfiguring the network to realize different neural network systems.

In the next Section we will describe the principles of the proposed T-mode circuit design technique. Then we will describe how to extend this technique to include hebbian learning and analog storage capability. Finally we will give experimental results of programmable and learning prototypes. Using the programmable modular chips we have assembled a BAM network, a Hopfield network, a Winner-Take-All network, and a simplified ART1 network. Using the learning modular chips we will give results of an adaptive BAM network. All these prototypes were obtained from a standard 2μm double-metal, double polysilicon CMOS process (MOSIS).

II. The T-Mode Neural Circuit Design Technique

Most of the Neural Network algorithms available in the literature have a short term memory (STM) whose continuous-time version[1] operation can be described by the following set of nonlinear first order differential equations,

$$C\dot{x}_i = -\alpha x_i + \sum_{j=1}^{N} w_{ji} f(x_j) + I_i \qquad i = 1, \ldots N \tag{1}$$

where x_i is the activity of neuron i, w_{ji} is the weight of the synaptic interconnection from neuron j to neuron i, I_i is the external input to neuron i, α and C are positive constants, and $f(\bullet)$ is a nonlinear, monotonically increasing function with a maximum and a minimum saturation values. Fig. 1 shows how to implement the set of equations (1) using transconductance amplifiers of transconductance gain w_{ji} for the synaptic interconnections, and using a nonlinear voltage amplifier to provide the neuron output $y_i = f(x_i)$. In the steady state, the association of resistor R and voltage amplifier $f(\bullet)$ acts as a nonlinear resistor with driving point characteristics described by

$$i = g(v) = \frac{1}{R} f^{-1}(v) \tag{2}$$

Extending this to the nonsteady state case, the circuit of Fig. 2 is obtained, which is described by the following set of first order nonlinear differential equations

$$C\dot{y}_i = -g(y_i) + \sum_{j=1}^{N} w_{ji} y_j + I_i \qquad i = 1, \ldots N \tag{3}$$

1. Grossberg provides a method [23] to map a discrete-time description of a neural network into a continuous-time one, and vice versa. Therefore, the neural network algorithms reported with discrete-time dynamics can also be represented by equation (1).

It can be shown [21] that, although the dynamics of descriptions (1) and (3) are slightly different, given the same initial conditions both descriptions will reach the same final steady state.

Using the technique of Fig. 2 to build a BAM network [5]-[7] results in the 2-layer circuit shown in Fig. 3, where a_i are the layer 1 neurons outputs, b_j the layer 2 neurons outputs, I_i the layer 1 external inputs, J_j the layer 2 external inputs, and w_{ji} the weight of the bidirectional synaptic connection between neuron j in layer 2 and neuron i in layer 1. Note that this circuit is able to be partitioned into several subcircuits, each of which inside a different modular chip [21],[22].

For the synaptic interconnections the transconductance amplifier shown in Fig. 4 was used, while for the nonlinear resistor the circuit is given in Fig. 5. The integrating capacitors were not physically implemented. Instead the parasitic capacitance of the interchip connection was used.

III. The Learning and Weight Storage Circuitry

An adaptive BAM [5] includes an additional first order differential equation for each bidirectional synapse in order to perform a hebbian learning rule

$$\dot{w}_{ji} = -w_{ji} + a_i b_j \tag{4}$$

Using the same transconductance amplifiers of the synaptic interconnections and, since the voltages a_i and b_j are locally available in each synapse, the circuit of Fig. 6 will add the learning capability to each synapse.

Fig. 7 shows a simplified diagram of the analog storage circuitry. The A/D converter and D/A converter pair is shared by all the synapses in the same chip. Once learning is finished, the integrating capacitor is disconnected from the learning circuit and used as a storage capacitor. The voltage drift due to parasitic leakage currents is compensated through periodic refreshing of its value by the A/D-D/A pair, so that the stored voltage remains within a finite interval [24].

IV. Experimental Results

In this Section we will give experimental results obtained from programmable and learning T-mode implementations. In the programmable case, the synaptic array chips have the bias input w_{ji} of the synaptic multipliers connected directly to an external pin, so that the value w_{ji} can be externally programmed. In the learning case the synaptic array chips have this bias internally connected as shown in Fig. 6.

A. Programmable Synaptic Array Chips

a) 9×9 BAM Network

A modular BAM network was assembled. Three pair of patterns were programmed into this two layer (9 neurons per layer) network (see Fig. 8). All three patterns could be successfully retrieved by giving partial images of the stored patterns. Fig. 9 shows the convergence to pattern *A* in one of the cases.

b) Hopfield Network

By connecting the modular chips in the way shown in Fig. 10, a 5-neuron Hopfield network is obtained. The pattern '10101' was programmed into the synaptic array chip. Fig. 11 shows the convergence to either the stored pattern '10101' or its complementary '01010' depending on the hamming distance between the external input and the stored pattern.

c) Winner-Take-All Network

A Winner-Take-All network is a particular case of Hopfield network in which all self-connections are made excitatory and the other connections are made inhibitory. Using the previous Hopfield network and reprogramming all the synapses a Winner-Take-All network was set up. Fig. 12 shows the transient of two neuron outputs with very similar inputs.

d) Simplified ART1 Network

A simple way to visualize the STM of an ART1 network [8] is as a BAM whose top layer is a Winner-Take-All network. This is shown in Fig. 13 for a 5×5 simplified ART1 T-mode network. Up to five patterns can be programmed into this network (see Fig. 14). The output converges to the stored pattern with the smallest hamming distance to the input pattern. If there are more than one stored patterns with the same minimum hamming distance, then all these patterns become active.

B. Learning 5×5 BAM

In order to train the learning BAM all patterns to be stored have to be presented sequentially at the inputs. We have fabricated a 5×5 learning synapses array chip which we used to assemble an adaptive BAM network.

Fig. 15 shows three patterns to be used for the training process. All three patterns could be successfully retrieved by giving partial images of the stored patterns as inputs. Fig. 16, for example, shows the convergence to pattern *C*.

V. Conclusions

We have developed a circuit design technique (called 'T-mode') for analog, continuous-time VLSI neural network circuit implementations. A great variety of neural systems can be implemented in hardware using this technique. We have built several of these systems and have tested them experimentally. These systems were a Hopfield Network, a BAM Network, a Winner-Take-All Network, and a simplified ART1 Network. We have also added a Hebbian learning scheme to one them (the BAM) as well as a dynamic analog memory to keep the learned weights. All these systems have been implemented on Silicon and successfully tested.

References

[1] J. J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities,", *Proc. Natl. Acad. Sci. USA*, vol. 79, pp. 2554-2558, April 1982.

[2] J. J. Hopfield, "Neurons with Graded Response Have Collective Computational Properties Like those of Two-State Neurons," *Proc. Natl. Acad. Sci. USA*, vol. 81, pp. 3088-3092, May 1984.

[3] J. J. Hopfield and D. W. Tank, "Neural Computation of Decisions in Optimization Problems," *Biol. Cybern.*, vol. 52, pp. 141-152, 1985.

[4] D. W. Tank and J. J. Hopfield, "Simple 'Neural' Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit," *IEEE Trans. Circuits and Systems*, vol. 33, pp. 533-541, May 1986.

[5] B. Kosko, "Adaptive Bidirectional Associative Memories," *Applied Optics*, vol. 26, pp. 4947-4960, 1 December, 1987.

[6] B. Kosko, "Bidirectional Associative Memories," *IEEE Trans. Systems, Man and Cybernetics*, vol. 18, pp. 49-60, January/February 1988.

[7] B. Kosko, *Neural Networks and Fuzzy Systems*. Englewood, NJ: Prentice Hall, 1992.

[8] G. A. Carpenter and S. Grossberg, "A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine," *Computer Vision, Graphics, and Image Processing*, vol. 37, pp. 54-115, 1987.

[9] T. Kohonen, "The 'Neural' Phonetic Typewriter," *Computer*, pp. 11-22, vol. 21, March 1988.
[10] T. Kohonen, *Self-Organization and Associative Memory*, 2nd ed. New York: Springer Verlag, 1988.
[11] R. P. Lippmann, "An Introduction to Computing with Neural Nets," *IEEE ASSP Mag.*, April 1987.
[12] J. A. Anderson, "A Simple Neural Network Generating Interactive Memory," *Math. Biosciences*, vol. 14, pp. 197-220, 1972.
[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-Propagating Errors," *Nature 323*, pp. 533-536, 1986.
[14] G. E. Hinton and R. J. Sejnovski, "Learning and Relearning in Boltzmann Machines," in *Parallel Distributed Processing*, vol. 1, Ch7, D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA, M.I.T. Press, 1986.
[15] E. A. Vittoz, "Future of Analog in the VLSI Environment," *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS90), New Orleans*, pp. 1372-1375, 1990.
[16] C. Mead, "Neuromorphic Electronic Systems," *Proceedings of the IEEE*, vol. 78, No. 10, pp. 1629-1636, October 1990.
[17] K. A. Boahen, P. O. Pouliquen, A. G. Andreou, and R. E. Jenkins, "A Heteroassociative Memory Using Current-Mode MOS Analog VLSI Circuits," *IEEE Trans. Circuits and Systems*, vol. 36, pp. 747-755, May 1989.
[18] S. W. Tsay and R. W. Newcomb, "VLSI Implementation of ART1 Memories," *IEEE Trans. Neural Networks*, vol. 2, No. 2, pp. 214-221, March 1991.
[19] A. F. Murray, D. Del Corso, and L. Tarassenko, "Pulse-Stream VLSI Neural Networks Mixing Analog and Digital Techniques," *IEEE Trans. Circuits and Systems*, vol. 36, pp. 193-204, May 1989.
[20] Y. Arima, K. Mashiko, K. Okada, T. Yamada, A. Maeda, H. Notani, H. Kondoh, and S. Kayano, "A 336-Neuron, 28K-Synapse, Self-Learning Neural Network Chip with Branch-Neuron-Unit Architecture," *IEEE J. Solid-State Circuits*, vol. 26, No. 11, pp. 1637-1644, November 1991.
[21] B. Linares-Barranco, E. Sánchez-Sinencio, A. Rodríguez-Vázquez, and J. L. Huertas, "A Modular T-Mode Design Approach for Analog Neural Network Hardware Implementations," *IEEE J. Solid-State Circuits*, (to be published).
[22] B. Linares-Barranco, *Analog Neural Network VLSI Implementations*, PhD Dissertation, Texas A&M University, College-Station, Texas, December 1991.
[23] S. Grossberg, "Nonlinear Neural Networks: Principles, Mechanisms, and Architectures," *Neural Networks*, vol. 1, No. 1, pp. 17-61, 1988.
[24] P. B. Brown, R. Millecchia, and M. Stinely, "Analog Memory for Continuous-Voltage, Discrete-Time Implementation of Neural Networks," *Proceedings of the IEEE ICNN'87*, vol. 3, pp. 523-530, 1987.

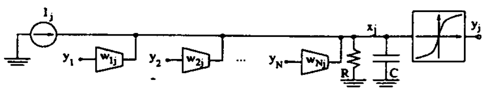

Fig. 1. A T-Mode Implementation of Neural Network

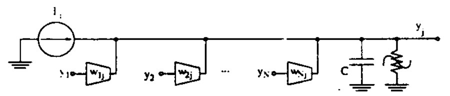

Fig. 2. Modified T-Mode Neural Network

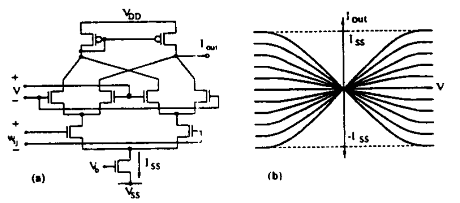

Fig. 4. (a) Circuit Implementation of Transconductance Multiplier, (b) DC Transfer Curves

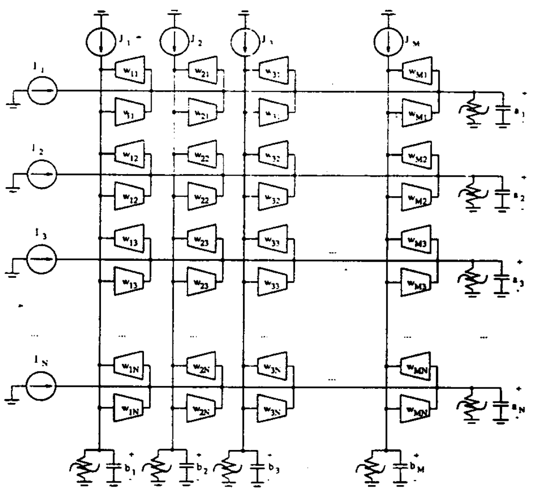

Fig. 3. T-Mode Circuit Implementation of BAM Algorithm

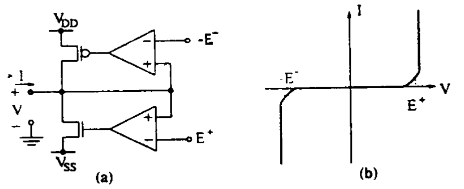

Fig. 5. (a) Nonlinear Resistor Circuit Implementation, (b) Transfer Curve

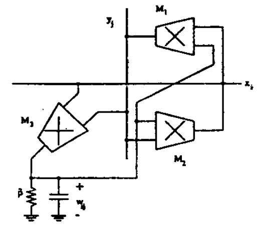

Fig. 6. Learning BAM Synaptic T-Mode Circuit

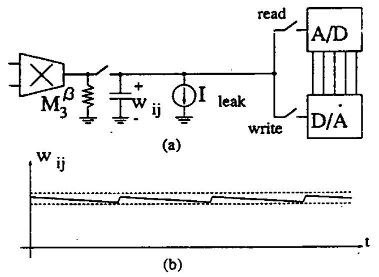

Fig. 7. Simplified Schematic of Weight Refreshing Circuit

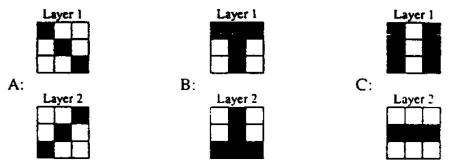

Fig. 8. Patterns Stored in 9x9 BAM

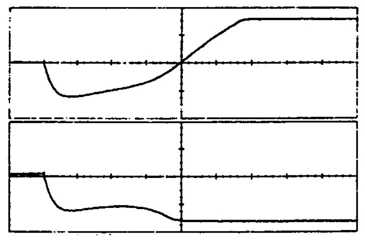

Fig. 12. T-Mode Winner-Take-All Circuit Transient

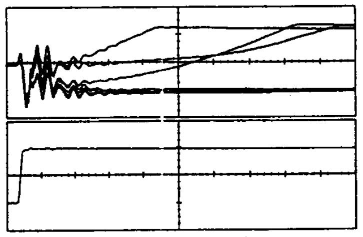

Fig. 9. Convergence to Pattern A in 9x9 BAM

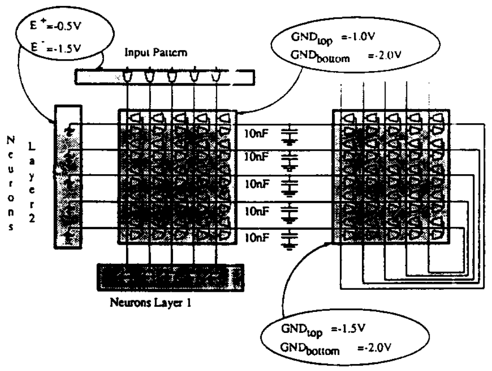

Fig. 13. Interconnection Topology for Simplified ART1 Network Using T-Mode Modular Chips

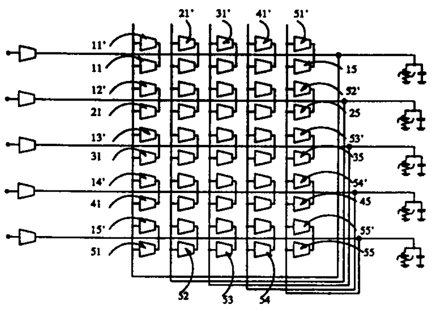

Fig. 10. Hopfield Network T-Mode Implementation

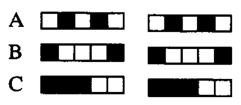

Fig. 14. Five Patterns Stored in the Simpified ART1 Network

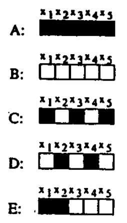

Fig. 15. Three Patterns Stored in the Learning BAM

Input		Stable Pattern	
(0)	00000	10101	(21)
(1)	00001	10101	(21)
(2)	00010	01010	(10)
(3)	00011	10101	(21)
(4)	00100	10101	(21)
(5)	00101	10101	(21)
(6)	00110	10101	(21)
(7)	00111	10101	(21)
(8)	01000	01010	(10)
(9)	01001	10101	(21)
(10)	01010	01010	(10)
(11)	01011	01010	(10)
(12)	01100	10101	(21)
(13)	01101	10101	(21)
(14)	01110	01010	(10)
(15)	01111	10101	(21)
(16)	10000	10101	(21)
(17)	10001	10101	(21)
(18)	10010	10101	(21)
(19)	10011	10101	(21)
(20)	10100	10101	(21)
(21)	10101	10101	(21)
(22)	10110	10101	(21)
(23)	10111	10101	(21)
(24)	11000	10101	(21)
(25)	11001	10101	(21)
(26)	11010	01010	(10)
(27)	11011	10101	(21)
(28)	11100	10101	(21)
(29)	11101	10101	(21)
(30)	11110	10101	(21)
(31)	11111	10101	(21)

Fig. 11. Measured Stable States for Hopfield Circuit Loaded with Pattern '10101'

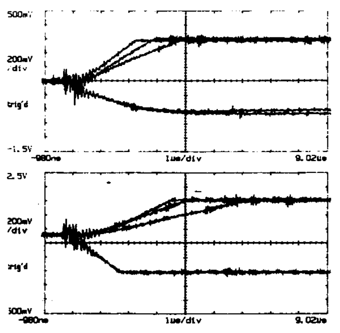

Fig. 16. Convergence to Pattern C in Learning BAM