

A Flexible and Expendable Neuroimage Processor Architecture

Gunhee Han, *Member, IEEE*, and Edgar Sánchez-Sinencio, *Fellow, IEEE*

Abstract—An analog versatile neuroimage processor (VNIP) architecture is proposed here. VNIP can process various types of neural network and image processing structures, without any hardware modification. The structure allows unlimited expansion of network size and the compensation of process variation. The proof-of-concept chip is implemented, using a combination of continuous-time multiplier and switched-capacitor techniques. The throughput is 12×10^6 synapses/s \cdot mm² and the energy consumption is 10^{-9} J/synapse. A test chip was fabricated, using a 1.2- μ m double-poly CMOS process and tested, verifying the flexibility and expandability of the architecture.

Index Terms—.

I. INTRODUCTION

NEURAL networks are structural approaches to solve complex problems that are difficult to solve analytically. The paradigm of a neural network is often inspired from a biological system. The common aspect of neural networks and image processing is the fact that both require massive parallel computation of matrix multiplication. Although the recent developments of high-performance digital signal processors have speeded-up the computation and have achieved, in practice, almost real-time processing, their high cost, large physical size, and power consumption often limit the usage of such systems in many every-day applications and, in particular, in portable applications.

Several analog implementations of parallel systems have been reported [1]. The analog implementation has the advantage over the digital system in the physical size and in the power consumption. Unfortunately, the application-specific structure of many analog implementations limits the applications range of the developed analog system. Several analog general-purpose processors are reported in [2]–[7]. However, they still have a limited application range and network size. In practice, most applications require the use of a large network size that cannot be implemented as a fully parallel structure on a single chip, even with an analog implementation. The modular approach [8]–[12], implementing a large system with the array of small subsystems, has a limit due to the high cost of reliable analog interchip communication.

Manuscript received December 15, 1997; revised October 15, 1998. This paper was recommended by Associate Editor T. Roska

G. Han is with Yonsei University Seoul 120-749, Korea.

E. Sánchez-Sinencio is with the Department of Electrical Engineering, Texas A&M University, College Station, TX 77843-3128 USA (e-mail: sanchez@ee.tamu.edu).

Publisher Item Identifier S 1057-7122(99)08065-4.

It is very difficult and involves a high cost to build a practical fully parallel analog system, without any significant breakthrough in fabrication technology. Therefore, an analog versatile neuroimage processor (VNIP) is proposed. It should be designed to be:

- expandable without requiring high-cost nor complex interchip connections;
- capable of process variation compensation;
- Flexible to implement a number of paradigm architectures;

It is an analog pseudo-parallel system with maximum flexibility, expandability, and computation efficiency. The proposed VNIP allows correction of process variation as well.

The structures of neuroimage processing networks are categorized in Section II. In Section III, the VNIP architecture is proposed. Section IV presents several application examples. The circuits and chip-test results are provided in Sections V, and VI, respectively. Section VII provides the conclusions.

II. NEUROIMAGE PROCESSING NETWORKS

The major computing element of a neural network is called a neuron [1]. It performs a sum-of-product computation and a nonlinear mapping expressed as

$$\begin{aligned} y &= f(\text{net}) \\ \text{net} &= \sum_i w_i x_i \end{aligned} \quad (1)$$

where y is the output of a neuron, x_i is the input, w_i is the weight associated to the x_i , and f is an application-dependent nonlinear function. The structure of a network is determined by the nature of the problem to be solved.

The networks can be categorized by the dimension of the data. The one-dimensional (1-D) network is expressed as

$$y_j = f\left(\sum_i w_{i;j} x_i\right) = f(\text{net}_j) \quad (2)$$

where y_j is the output of the j th neuron and $w_{i;j}$ is the weight from the input x_i to the j th neuron, as shown in Fig. 1. The 2-D (2-D) network is expressed as

$$y_{lm} = f\left(\sum_i \sum_j w_{i;j;lm} x_{ij}\right) \quad (3)$$

where y_{lm} is the output of the neuron _{lm} , located at the l th row and the m th column, x_{ij} is the input located at the i th

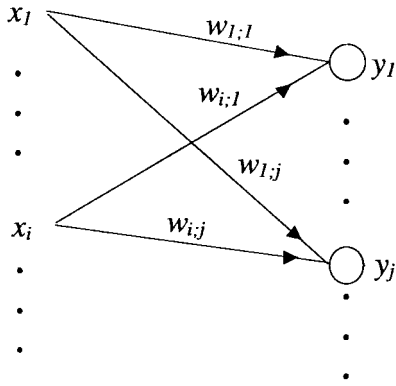


Fig. 1. 1-D network.

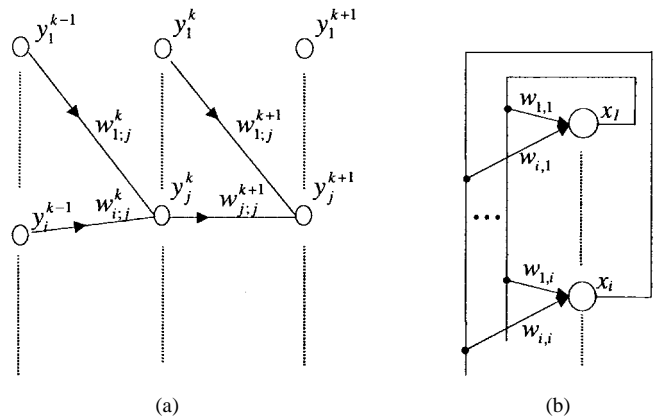


Fig. 3. (a) Feed-forward multilayer networks. (b) Recurrent network.

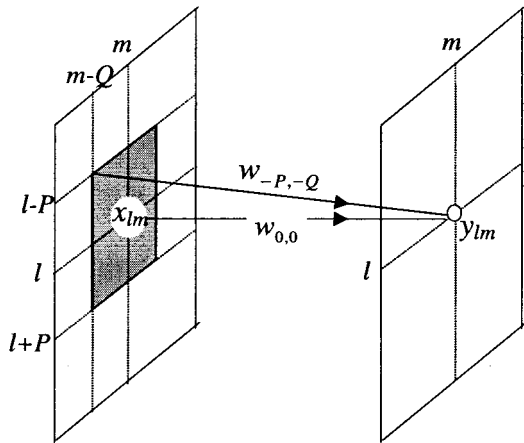


Fig. 2. 2-D network with shift-invariant weight.

row and the j th column, and $w_{ij;lm}$ is the weight from the input x_{ij} to the neuron l_m . In many applications, the weight is shift invariant and (3) can be expressed as

$$y_{lm} = f \left(\sum_{p=-P}^P \sum_{q=-Q}^Q w_{pq} x_{(l+p)(m+q)} \right) \quad (4)$$

where p and q represent the vertical and horizontal distances from the neuron l_m to an input node, respectively. The P and Q are vertical and horizontal neighborhood sizes, respectively. A pictorial description is given in Fig. 2.

The neuroimage processing structure can be also grouped into a feed-forward multilayer network or a recurrent network, as shown in Fig. 3. The discrete-time recursive network description is equivalent to a multilayer structure, from the computational point of view. The k th iteration corresponds to the k th layer.

III. PROPOSED PROCESSOR ARCHITECTURE

The basic architecture is a matrix-vector multiplication processor. The additional signal routing provides flexibility and expandability. The architecture consists of an array of rows, as shown in Fig. 4. Each row consists of main memory, a multiplier, an accumulator (Σ), an output buffer memory

(OBM), and a multiplexer. At row j , the multiplier $_j$ receives one input (w_{ij}) from a main-memory cell. In the main memory, the same vertical cells, the i th column on all rows are selected at the same time. All multipliers share the other input, the x terminal. At this node the sequence of inputs $\{x_i\}$ is applied, while the corresponding column, the i th column, is simultaneously selected in the main memory. The output of the multiplier $_j$ is accumulated in the accumulator $_j$. At the end of the sequence, $\{x_i\}$, the accumulator on the j th row contains the result of

$$net_j = \sum_i w_{ij} x_i. \quad (5)$$

A similar computing structure for the optimization problem is reported in [12]. This result can be sent to three different directions. It can be sent to the OBM or one of the cells in the main memory on the same row. Or, it can be shifted down simultaneously. The multiplexer selects one element of the OBM at a time to read out, sequentially, its contents. The output of the multiplexer is applied to the nonlinear block $f(\cdot)$, which performs additional operations such as addition, subtraction, multiplication, division, nonlinear mapping, and/or logical operations. This block can have an external input u . The output of the nonlinear block y can be fed back to the x -terminal. Since only one nonlinear block is required, a high-performance programmable nonlinear block can be implemented for a wide range of applications. This feature is usually not acceptable in conventional fully parallel implementations.

The content of the OBM is accessible, independent of the accumulator operation. This feature allows the use of pipelining, which means that the computation is performed by multipliers and accumulators while the I/O data is performed at the OBM and the multiplexer.

Since only one interrow connection is required, the vertical expansion of network is unlimited. The horizontal expansion is achieved by increasing the horizontal memory size. However, if the main memory is implemented on chip, then the horizontal network size is limited by the silicon size. If it is implemented off chip, then the limited number of pins limits the vertical network size.

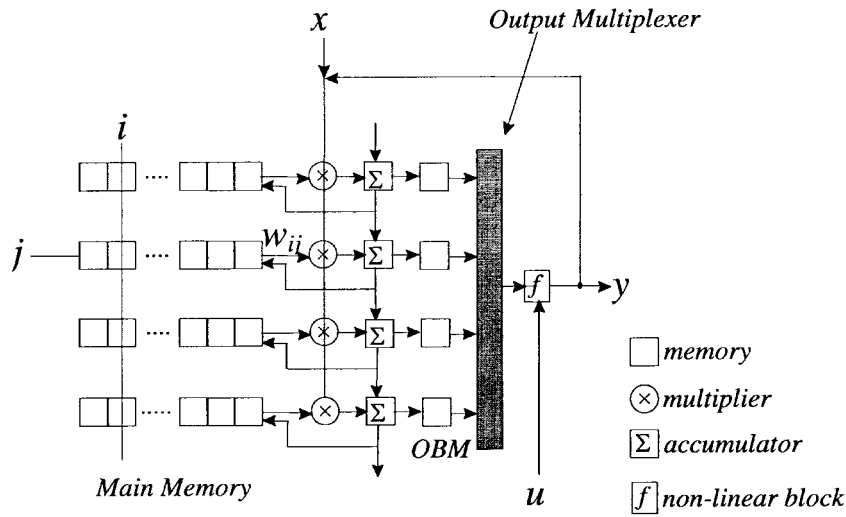


Fig. 4. Architecture of proposed processor.

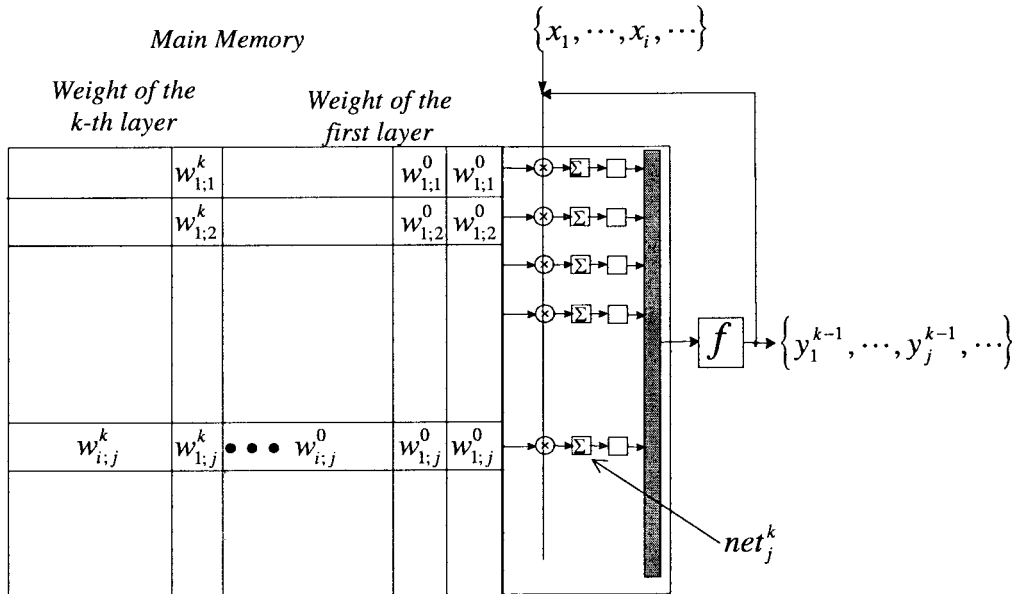


Fig. 5. Application in a 1-D network.

IV. APPLICATIONS OF VNIP

A. 1-D Networks

In the case of the 1-D networks, the weights are stored in the main memory, as shown in Fig. 5. $\{y_j^k\}$ represents the output of j th neuron on k th layer. To process the first layer, the input is applied as a sequence to the x terminal while a corresponding column is selected in the main memory. At the end of the sequence $\{x_i\}$, the accumulators contain the nets of first layer net^1 . The contents of accumulators are transferred to the OBM and then the accumulators are reset. The contents of the OBM are read out one by one through the nonlinear block, using the multiplexer. The outputs of the nonlinear block are the output sequence of the first layer. The second layer is processed by applying this output to the x terminal, while corresponding weights for the second layer are selected in the

main memory. Once all the output buffer memories are read out, the accumulators have net^2 . Then this data is sent to the OBM and then read out through the nonlinear block. The output is the sequence of the second layer. A multilayer can be processed by repeating the above procedure.

A discrete-time recurrent network is processed as a multi-layers structure, whose weights are identical for all layers.

A feed-forward network can have crossover connections from a nonadjacent layer, as shown in Fig. 6(a). The weight $w_{i;j}^{l;k}$ represents the weight from i th neuron on the l th layer to the j th neuron on k th layer. This network can be processed using two processors and a two-input multiplexer, as shown in Fig. 6(b). Each processor generates the nets for odd and even layers, alternately. Similarly, when there are crossover connections from the L th lower layers, then L processors and an L -inputs multiplexer are required.

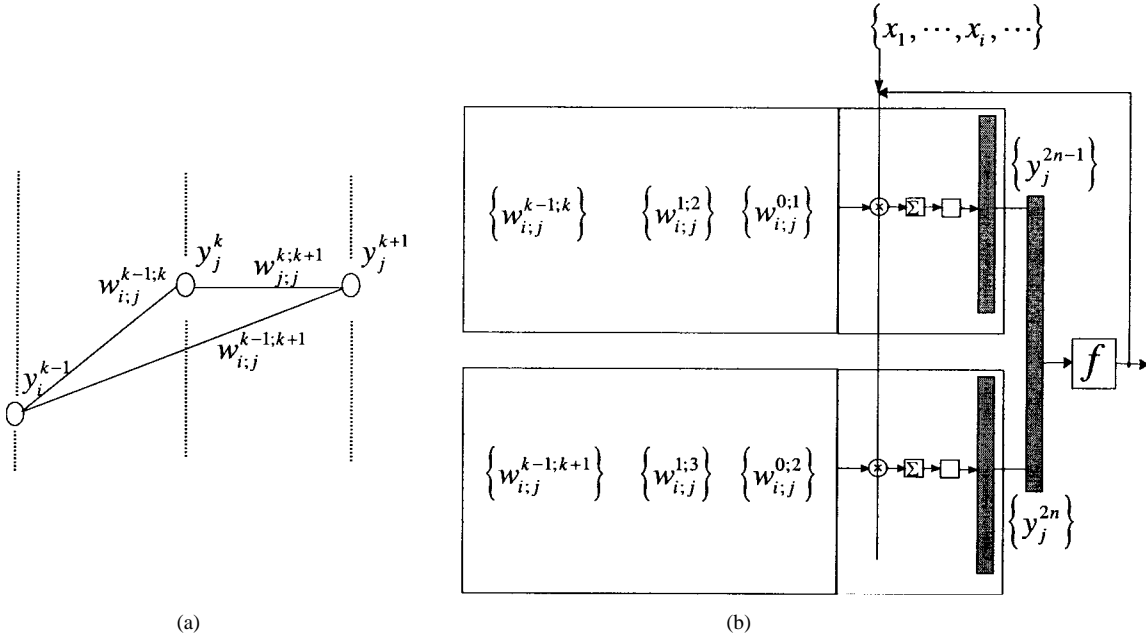


Fig. 6. (a) Application in a 1-D multilayer network with crossover connections. (b) Its implementation using VNIP.

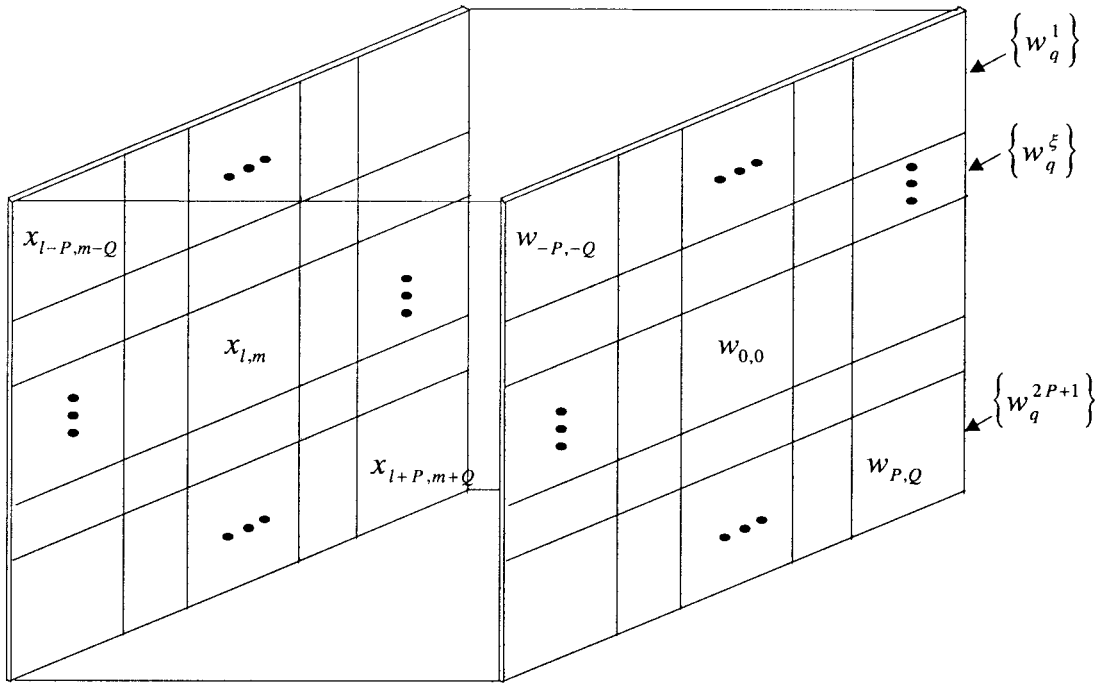


Fig. 7. Indexing of neighborhood and weights in a 2-D network with shift-invariant weights.

B. 2-D Networks

Based on (4), define

$$\text{net}_{lm}(\xi) = \sum_{p=-P}^{\xi-1-P} \sum_{q=-Q}^Q w_{pq} x_{(l+p)(m+q)},$$

$$\xi = 1, \dots, 2P + 1. \tag{6}$$

Then the following recursive equation can be formulated as:

$$\text{net}_{lm}(\xi) = \text{net}_{lm}(\xi - 1) + \sum_{q=-Q}^Q w_q^\xi x_{(l+\xi-1-P)(m+q)},$$

$$\text{net}_{lm}(0) = 0 \tag{7}$$

where w^ξ is the ξ th row vector in the matrix $\{w_{pq}\}$. Fig. 7 shows these indexing terms in 2-D networks. The m and l in (7) are the column and the row numbers on the 2-D data, respectively. q is the column index within the neighborhood and the weight matrix. Then net_{lm} is given by

$$\text{net}_{lm} = \text{net}_{lm}(2P + 1). \tag{8}$$

In (7), the summation is equivalent to that of (5), from the computational point of view. The only difference is that in (5), the weight is a 2-D array and the signal is a 1-D vector, while in (7), the signal is a 2-D array and the weight is a 1-D vector. This means that a fixed hardware structure can process

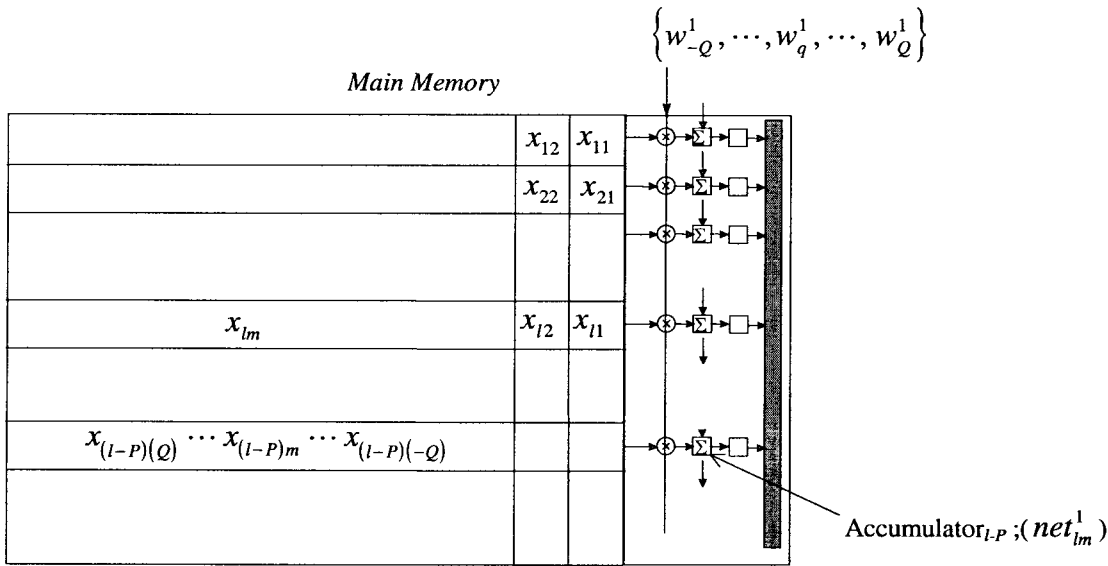


Fig. 8. Application in 2-D networks.

both 1-D and 2-D networks by interchanging the signal and the weight.

The 2-D propagating data, x , are stored in the main memory and the weight, w , is applied to the x terminal as a sequence, as shown in Fig. 8. Since the proposed structure is a linear array of processing elements, a 2-D signal is processed column by column.

For the given column m , at the first stage $\xi = 1$. The weights $\{w_q^1\}$ are applied to the x terminal, while corresponding columns $m - q$ are selected from the main memory for $q = \{-Q, \dots, Q\}$. Once this procedure is finished, the accumulator $_{l-p}$ contains the $net_{lm}^1(1)$. The contents of the accumulators are shifted down and the above procedure is repeated with the weight $\{w_q^2\}$. After $(2P + 1)$ iterations $\xi = (2P + 1)$, the accumulator $_{l-p}$ contains $net_{lm}^1(2P + 1)$, which is the end of the net computations for a given m . Then the contents of the accumulators are transferred to the OBM and read out through the nonlinear block. The output sequence corresponds to the result for the m th column. Repeat the above procedure for next column.

A 2-D multilayer network can be realized in a way similar to the 1-D case. The output sequence from the nonlinear block can be sent back to the main memory as a sequence, or the nets can be directly sent back to the main memory in parallel.

V. CIRCUIT DESIGN

The major building block in the VNIP is the weighted integrator that consists of a multiplier and an integrator. If the multiplier offset is not canceled, then this offset is undesirably accumulated in the integrator. A switched capacitor weighted integrator with offset cancellation is proposed to tackle this problem. Fig. 9 shows its schematics. The multiplier, including the offset, can be modeled as

$$z = K(x - x_o)(y - y_o) + z_o \quad (9)$$

where K is a multiplication constant and x_o , y_o , and z_o are the offsets. These offsets can be canceled, using four combinations

of input signal polarity, $\{(x, y), (-x, y), (-x, -y), (x, -y)\}$ as follows:

$$\begin{cases} z_1 = K(x - x_o)(y - y_o) + z_o \\ z_2 = K(-x - x_o)(y - y_o) + z_o \\ z_3 = K(-x - x_o)(-y - y_o) + z_o \\ z_4 = K(x - x_o)(-y - y_o) + z_o \end{cases} \quad (10)$$

Then

$$(z_1 - z_2) + (z_3 - z_4) = 4Kxy. \quad (11)$$

In Fig. 9, ϕ_1 and ϕ_2 are nonoverlapping clocks. The charge injected into the integrator at ϕ_2 corresponds to $(z_{\phi_2} - z_{\phi_1})$. Using clocks ϕ_A and ϕ_B , the switches at the multiplier input interchange the differential input line and the actual input to the multiplier becomes $\{(x, y), (-x, y), (-x, -y), (x, -y)\}$. At the end of the fourth phase, the integrator contains the offset canceled multiplication, as in (11). A folded CMOS Gilbert multiplier [13] is used.

The other key analog block is the output multiplexer. This block reads out the content of the OBM without destroying its content. The simple buffer, as shown in Fig. 10, is used for simplicity. A pair of these circuits is used for the differential structure. A two-stage buffer is used to minimize the clock feedthrough from the selection switch to the OBM capacitor. The sources of the source followers are connected together and form a data bus line that is biased by a current source (I_S). Since only one of the selection switches in the OBM is turned on at a time, only one source follower loads the data on the data bus line.

Fig. 11 shows the block diagram of the VNIP. The gray rectangular box represents the pad. All the signals and building blocks are fully differential. The w inputs are coming from the memory and are held at sample-and-hold (S/H) circuit. One S/H circuit is used for the x input terminal. The clocks ϕ_1 , ϕ_2 , ϕ_A , and ϕ_B are generated by an on-chip integrator clock. The signal in the integrator can be copied to the OBM by turning on the transfer switch. The content of the OBM can

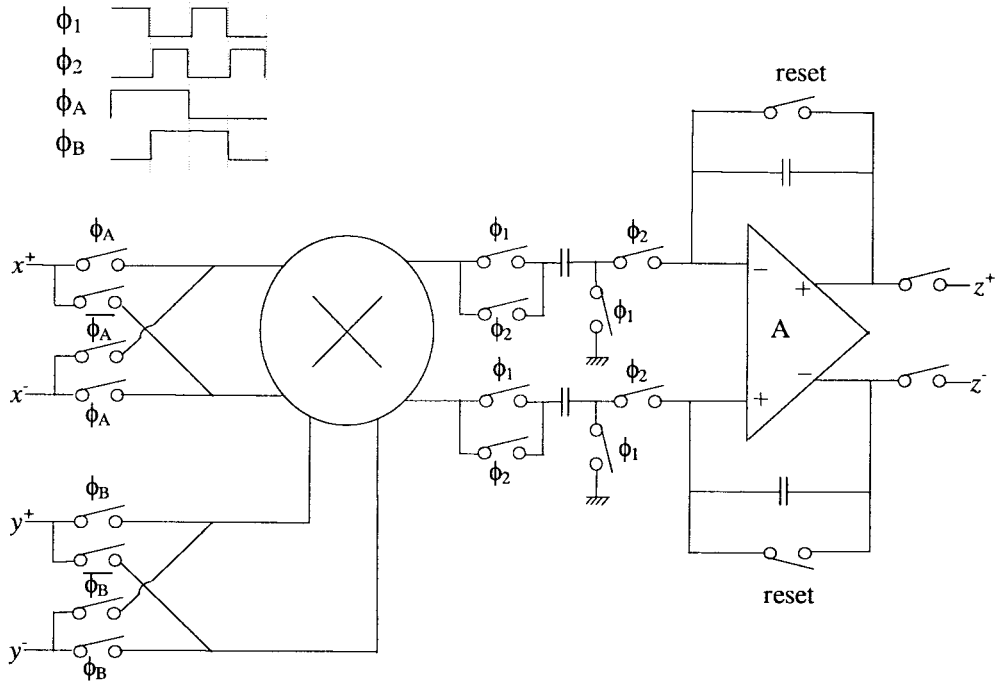


Fig. 9. A switched-capacitor weighted integrator with offset cancellation.

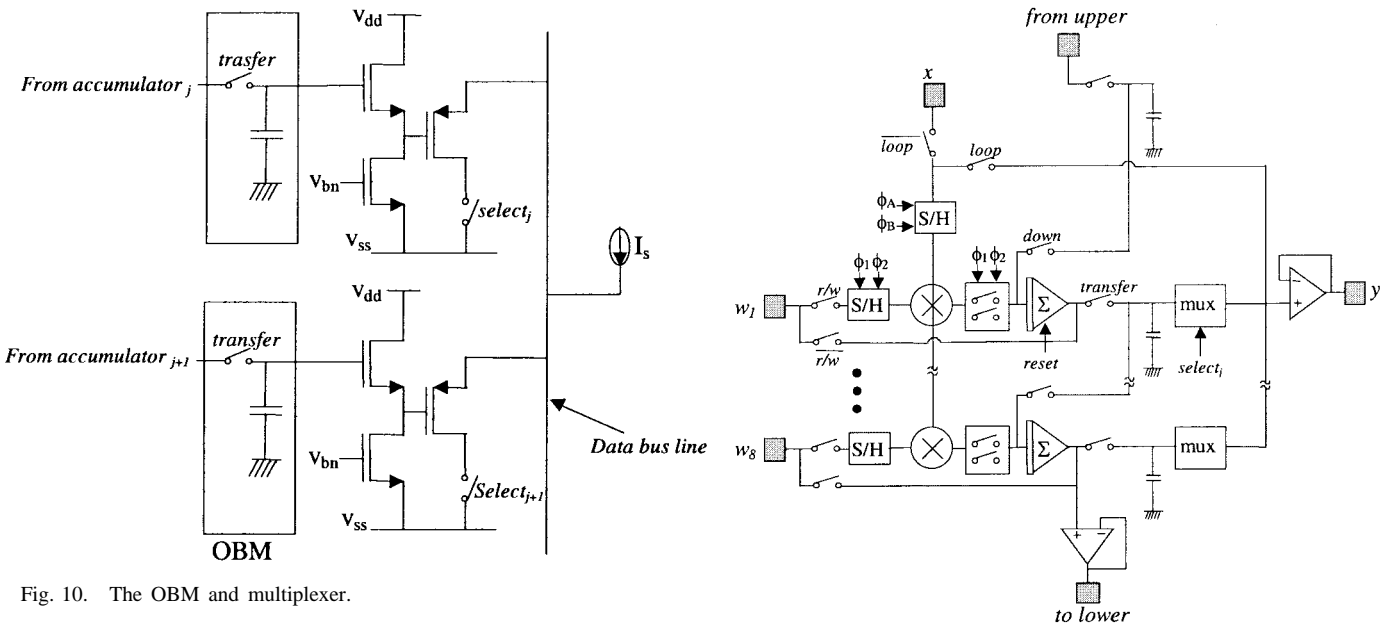


Fig. 10. The OBM and multiplexer.

be shifted down to the integrator on one row below, using the down switch. The bottom row has output buffer to drive the pad, *to lower*. This pad is connected the *from-upper* pad on the other chip for expansion. This expansion is independent from the large capacitance of the expansion pin because it is driven by a relatively large buffer. This structure allows unlimited network size expansion. The down operation in 2-D networks is achieved by a transfer-reset-down clock sequence. The signal on the data bus line can be sent to the pad through the output buffer. It also can be sent to the multiplier's *x* terminal by turning on the loop switch. The contents of the accumulators can be accessed in parallel, through the same pad for *x* input signal that is multiplexed by the *r/w* switch.

Fig. 11. Block diagram of the VNIP and control of switches.

The main memory can be a capacitor memory array as a short-time memory. In the case of the 1-D network, the main memory should be a permanent memory that can be implemented, using a floating gate array. In the case of an image processor, the main memory is replaced with a photosensor array as an input device.

One effect of process variation is the gain of mismatch of multipliers. The gain of each row can be obtained by measuring the output of a row with test input signal. For the 1-D network, the weight on the same row should be scaled according to the corresponding measured row gain. This

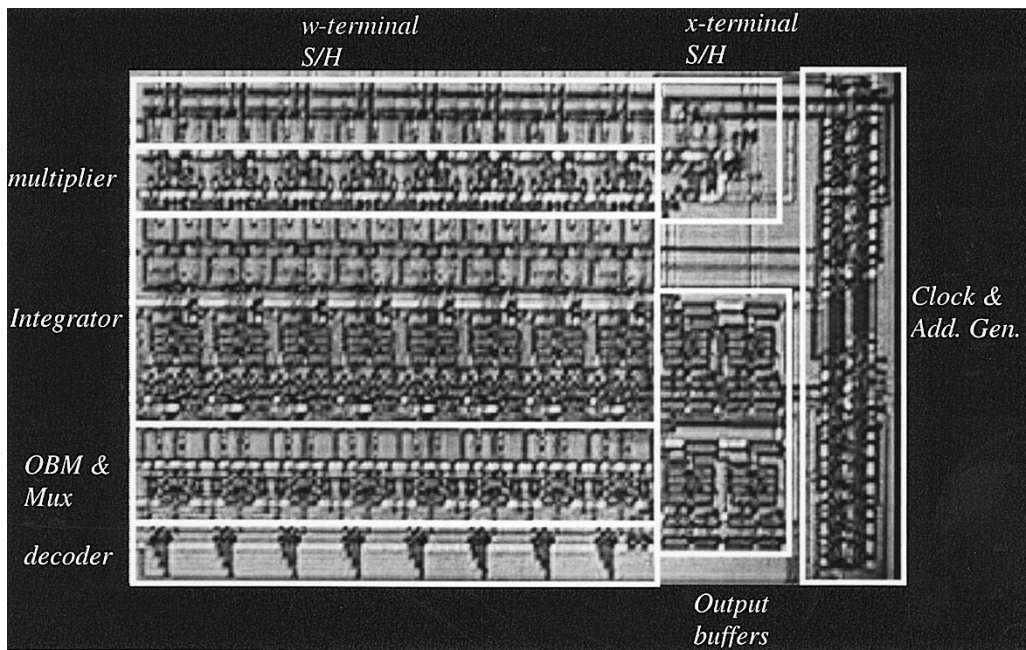


Fig. 12. Microphotograph of fabricated VNIP.

procedure can be implemented in the training. In the case of the 2-D network, the gain of each row is stored in an external permanent memory and used for compensation at the output.

The proof-of-concept chip is fabricated using a double-poly 1.2- μm CMOS process, to demonstrate the flexibility of the proposed architecture. Fig. 12 shows a microphotograph of fabricated chip. This chip contains eight rows of processing elements. The maximum clock frequency of the test chip is 4 Mhz. The silicon area is $100\lambda \times 840\lambda/\text{row}$. The throughput is 12×10^6 synapses/s \cdot mm² and the energy consumption is 10^{-9} J/synapse, where one synapse corresponds to one multiplication and addition. Note that these data do not include the main memory, whose implementation is application dependent. The accuracy of the computation is dependent on the choice of circuit, power supply voltage, application, and network size.

VI. EXPERIMENTAL TEST-CHIP RESULTS

The main memory is emulated using a PC and data-acquisition boards. A software program is written for the user interface. For illustration purposes, three different types of neural networks (a two-layer feed-forward network, a fully connected recurrent network, and a 2-D network) are implemented without any hardware modification. Though the network size expansion is not limited, small-sized examples are presented in this paper.

The XOR problem, depicted in Fig. 13(a), is implemented to demonstrate the application for the two-layer feed-forward network. Figure 13(b) shows its truth table. The weights are properly chosen and then a test input sequence $\{(1,1), (1,-1), (-1,1), (-1,-1)\}$ is applied to the input of the network. Fig. 13(c) shows the states of accumulators measured from the chips, using an oscilloscope. The expected

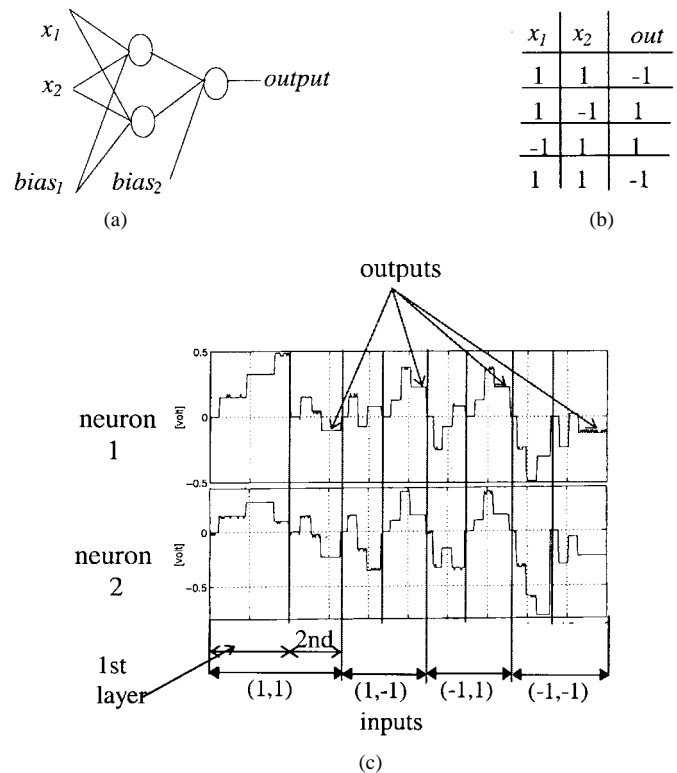


Fig. 13. Test result of 1-D multilayers networks exclusive-OR (XOR) problem. (a) Two-layer feed-forward network for the XOR problem. (b) Its desired output. (c) The state of the accumulators.

signs of the neuron's state, $\{-, +, +, -\}$, for the above test inputs sequence are obtained.

The winner-takes-all (WTA) problem is implemented to demonstrate the application for a 1-D recurrent network, as shown in Fig. 14(a). The initial conditions are set to 1 and 0.8

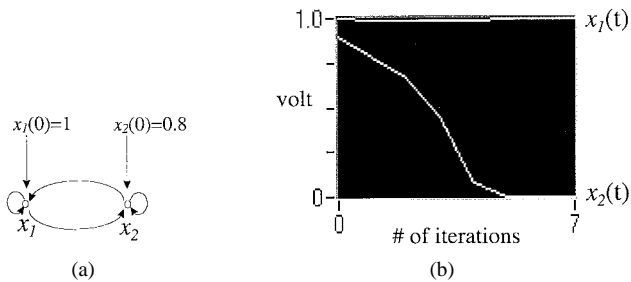


Fig. 14. Test results of 1-D recurrent networks (WTA problem). (a) Fully connected network for the WTA problem. (b) WTA experimental with two components.

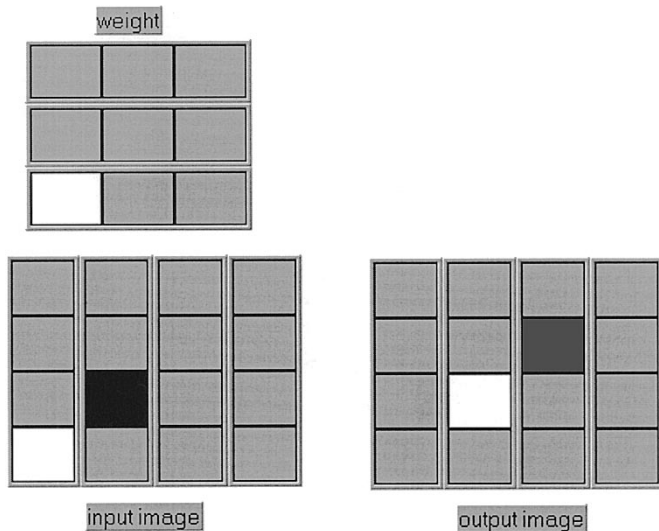


Fig. 15. Test result of 2-D networks (propagation).

V, respectively. The weights between neurons are set to -0.2 V and self loop weights are set to 1 V. Fig. 14(b) shows the output of each neuron with respect to the iteration number. The graph is taken from the user interface of the software. The winner which was set to 1 V stays at 1 volt, but the loser which was set to 0.8 V goes to 0 V.

A 3×3 neighborhood is implemented to demonstrate the application for a 2-D network. The white pixel in Fig. 15 represents 4 V, the black represents -4 V, and the gray represents zero. A propagation test is performed for the functionality test [14]. Fig. 15 shows one example of a propagation test. Since only the lower left weight is white, the image should be shifted toward the upper right direction, without changing the polarity. The output image shows the expected correct result.

VII. CONCLUSION

A tradeoff of versatility versus circuit complexity has been implemented in the proposed NIP. The proposed neuroimage processor provides a flexible and expandable architecture that is capable of processing a number of neural networks or image processing structures, without any hardware modifications, and a wide range of applications can be expected from this processor. The structure allows unlimited expansion of network size and the compensation of process variation. The

use of multiplexing does not cause time overhead, with the use of pipelining scheme. This architecture requires a reliable analog memory just as any other analog neuroimage processor does.

REFERENCES

- [1] E. Sánchez-Sinencio and C. Lau, *Artificial Neural Networks*. New York: IEEE, 1992.
- [2] S. Satyanarayana, Y. P. Tsividis, and H. P. Garf, "A reconfigurable VLSI neural network," *IEEE J. Solid-State Circuits*, vol. 27, pp. 67–81, Jan. 1992.
- [3] W. A. Fisher, R. J. Fujimoto, and R. C. Smithson, "A programmable analog neural network processor," *IEEE Trans. Neural Networks*, vol. 2, pp. 222–229, Mar. 1991.
- [4] M. Holler, S. Tam, H. Castro, and R. Benson, "An electrically trainable artificial neural network (ETANN) with 10240 floating gate synapses," in *Proc. IEEE Int. Conf. Neural Networks*, June 1989, vol. 2, pp. 191–196.
- [5] P. Muller, J. V. der Spiegel, D. Blackman, T. Chiu, T. Clare, J. Dao, C. Donham, T. Hsieh, and M. Loinaz, "A general purpose analog neural computer," in *Proc. IEEE Int. Conf. Neural Networks*, June 1989, vol. 2, pp. 177–182.
- [6] N. Mauduit, M. Duranton, J. Gobert, and J. Sirat, "Lneuro 1.0: A piece of hardware LEGO for building neural network systems," *IEEE Trans. Neural Networks*, vol. 3, no. 3, pp. 414–422, May 1992.
- [7] Y. Arima, M. Murasaki, T. Yamada, A. Maeda, and H. Shinohara, "A refreshable analog VLSI neural network chip with 400 neurons and 40 K synapses," *IEEE J. Solid-State Circuits*, vol. 27, pp. 1854–1861, Dec. 1994.
- [8] 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*, vol. 27, pp. 701–713, May 1992.
- [9] M. E. Robinson, H. Yoneda, and E. Sánchez-Sinencio, "A modular CMOS design of hamming network," *IEEE Trans. Neural Networks*, vol. 3 pp. 444–456, May 1992.
- [10] J. V. der Spiegel, R. Etienne-Cummings, and P. Kinget, "An analog neural computer with modular architecture for real-time dynamic computation," *IEEE J. Solid-State Circuits*, vol. 27, pp. 82–91, Jan. 1992.
- [11] R. Mason, W. Robertson, and D. Pincock, "An hierarchical VLSI neural network architecture," *IEEE J. Solid-State Circuits*, vol. 27, pp. 106–108, May 1992.
- [12] R. Domínguez-Castro, A. Rodríguez-Vázquez, J. L. Huertas, and E. Sánchez-Sinencio, "Analog neural programmable optimizers in CMOS VLSI technologies," *IEEE J. Solid-State Circuits*, vol. 27, pp. 1110–1115, July 1992.
- [13] J. Ramírez and S. Ming-Shen, "The folded Gilbert cell: A low voltage high performance CMOS multiplier," in *Proc. IEEE Midwest Symp. Circuits & Systems*, Aug. 1992, pp. 20–23.
- [14] M. Grimaila, J. Pineda, and G. Han, "Robust functional testing for VLSI cellular neural network implementations," *IEEE Trans. Circuits Syst. I*, vol. 44, pp. 161–166, Feb. 1997.



Gunhee Han (M'97) received the B.S. degree from Yonsei University, Seoul, South Korea, in 1990 and the Ph.D. degree from Texas A&M University, College Station, in 1997.

He was a Visiting Assistant Professor at Texas A&M University during the 1997–1998 academic year. He has been an Assistant Professor in the Department of Electronic Engineering, Yonsei University, since August 1998. His research interests include mixed-signal circuit design, neural networks, and digital signal recovery.



Edgar Sánchez-Sinencio (S'72–M'74–SM'83–F'92) was born in Mexico City, Mexico, on October 27, 1944. He received the Professional degree in communications and electronic engineering from the National Polytechnic Institute of Mexico, Mexico City, Mexico, the M.S.E.E. degree from Stanford University, Stanford, CA, and the Ph.D. degree from the University of Illinois at Champaign-Urbana, in 1966, 1970, and 1973, respectively.

From January 1965 to March 1967 he worked with the Mexican Atomic Energy Commission as a Design Engineer. In April 1967 he joined the Petroleum Institute of Mexico, where he was associated with the design of instrumentation equipment until August 1967. He worked as a Research Assistant at the Coordinated Science Laboratory, University of Illinois, from September 1971 to August 1973. In 1974 he held an industrial Post-Doctoral position with the Central Research Laboratories, Nippon Electric Company, Ltd., Kawasaki, Japan. From 1976 to 1983 he was the Head of the Department of Electronics at the Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Puebla, Mexico. He was a Visiting Professor in the Department of Electrical Engineering at Texas A&M University, College Station, during the academic years of 1979–1980 and 1983–1984. He is currently the TI Analog Engineering Chair Professor at Texas A&M University. His research interests are in the area of active filter design, RF-Communication circuits, and analog and mixed-mode circuit design.

Dr. Sánchez-Sinencio was the General Chairman of the 1983 26th Midwest Symposium on Circuits and Systems. From 1985 to 1988 he was an Associate Editor of News and Events for the *IEEE CIRCUITS AND DEVICES MAGAZINE* and from 1985 to 1987, an Associate Editor for *IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS*, (1985–1987). He was an Associate Editor for the *IEEE TRANSACTIONS ON NEURAL NETWORKS* and from 1997 to 1999 the Editor-in-Chief of the *IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II*. He is a coauthor of the book *Switched Capacitor Circuits* (New York: Van Nostrand-Reinhold, 1984) and a coeditor of the book *Low-Voltage/Low-Power Integrated Circuits and Systems* (Piscataway, NJ: IEEE, 1999). He is a former President of the IEEE Circuits and Systems Technical Committee on Neural Systems and Applications and the CAS Technical Committee on Analog Signal Processing. He received the 1995 Guillemin–Cauer Award for his work on cellular networks. He is a former IEEE CAS Vice President-Publications. He was also the Corecipient of the 1997 Darlington Award for his work on high-frequency filters.