Analog Electronic Neural Network Circuits

Hans P. Graf and Lawrence D. Jackel

Forward

The future status of artificial neural networks as an algorithmically and architecturally competitive computational technology will be based upon performance and implementation. In both regards, the artificial neural network must be shown to be superior to other, possibly more conventional, approaches.

***Circuits and Devices Magazine* is featuring three overview articles on the current status of artificial neural network implementation technology. The following paper by Graf and Jackel is the first of these papers and describes the current status of *analog electronic* implementation of artificial neural networks. Subsequent papers will describe photonic and digital implementation. The trilogy will provide a quality overview of current work in this exciting field of emerging technology**

*Robert J. Marks II**

Abstract

The large interconnectivity and the moderate precision required in neural network models present new opportunities for analog computing. Analog circuits for a wide variety of problems such as pattern matching, optimization and learning have been proposed and a few have been built. Most of the circuits built so far are relatively small, exploratory designs. The most mature circuits are those for template matching, and chips performing this function are now being applied to pattern recognition problems.

Introduction

The latest wave of interest in connectionist neural network models has been fueled by new theoretical results and by advances in computer technology that make it possible to simulate networks of much higher complexity than was possible before. Moreover, microelectronic technology has reached a stage where large neural networks can be integrated onto a single chip. As early as the 1960s analog network circuits were built that demonstrated collective computation and learning (see [1-2]). However, these networks had to use discrete components, and networks with just a few neurons resulted in very bulky circuits. This limited the size of networks that could be built, hence their computing power.

Today, a rapidly growing number of researchers are working on hardware implementation of neural network models. Four years ago, about five groups in the U.S. were building electronic neural networks and a similar number were implementing optical networks. In 1988, at several conferences devoted to neural networks, some 50 groups presented circuits or proposed designs, most of them from the U.S. but several from Europe and a few from Japan.

Electronic neural networks rely on strongly simplified models of neurons. It is generally assumed that the computing power of neural systems, electronic or biological, arise from the collective behavior of large, highly interconnected, fine-grained networks. An individual node, a neuron, does only very simple computations. Fig. 1 shows a simplified neural model consisting of the processing node (amplifier) interconnected to other neurons by resistors. The activity level of a neuron is its output voltage. The neuron i gets input from a neuron j through a resistor with the conductance T_{ij}. This conductance is referred to as the connection strength or the connection weight. If the voltage of the input wire is held at ground (e.g., in a virtual ground arrangement) then the signals coming from other neurons are currents with values of:

$$I_{ij} = Vout_j \, T_{ij} \tag{1}$$

I_{ij}: current flowing from neuron j to neuron i

$Vout_j$: output voltage of neuron j

T_{ij}: connection strength between neuron i and neuron j (conductance of the resistor)

All the currents coming from the other neurons are summed on the input wire and the output voltage of the neuron is a function of this total current. Typically, the amplifier has a nonlinear transfer characteristic; it can be a hard threshold or a smoother sigmoid. The output voltage of neuron i is given by:

$$Vout_i = f\left(\sum_{j=0}^{j=N} I_{ij}\right) = f\left(\sum_{j=0}^{j=N} Vout_j \, T_{ij}\right) \tag{2}$$

f: transfer function of the amplifier (neuron)

Equation (2) shows that computing sums of products is a key operation performed by the network and a hardware implementation has to focus on doing this efficiently. Very often only modest precision is required so that it is possible to use analog computation for this task. In an analog net-

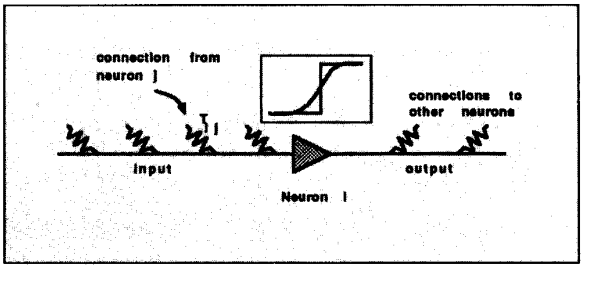

Fig. 1 A model neuron with the processing node (amplifier) and a few connections (resistors) to other neurons. The inset shows two possible transfer characteristics of the amplifier.

* R.J. Marks II is the Chairman of both the *IEEE Neural Networks Committee* (pro term) and the *IEEE Circuits and Systems Technical Committee on Neural Systems and Applications*. He is a Professor of Electrical Engineering at the University of Washington, Seattle. He was asked by *Circuits and Devices Magazine* to edit a series of three articles on neural networks.

8755-3996/89/0700-0044$1.00 © 1989 IEEE

work, a single resistor can perform a multiplication using Ohm's law, and summing of currents on a wire is provided by Kirchhoff's law. Therefore, an analog circuit that computes sums of products can be built much more compactly than a digital circuit.

The operation performed by a whole network is determined by the connection weights T_{ij}. The large computational power of a whole network results from the parallel operation of a large number of these model neurons. A major difference between a neuron and a digital gate is the high fan-in and fan-out of the neuron. A biological neuron is typically connected to several thousand other neurons. Such a high interconnectivity is very difficult to achieve in an electronic circuit since a huge number of connections and wires are required and all the wiring has to be placed on the two-dimensional surface of a chip. However, electronic networks do exist that interconnect a few hundred neurons.

What Are Neural Networks Good For?

Neural networks are of particular interest for cognitive tasks or control problems. Most of the problems neural networks have been applied to lie in one of the following areas:

- Machine vision
- Speech recognition
- Robotics, Control
- Expert systems

These are a few of the areas where conventional computers perform very poorly compared to our brains.

In many cognitive tasks such as vision, large amounts of data with a low information content have to be processed. For example, consider the task of identifying an object in an image of several hundred thousand pixels. The object's position and its orientation is information that can be encoded in just a few bits. Reducing the data in the image down to the relevant part is a problem that is not well suited for standard computers. The processor has to plow through all the pixel data, performing operations with a very high precision on pixels that are mostly meaningless. Highly interconnected neural networks, on the other hand, provide an architecture that is very effective in extracting correlations among image pixels.

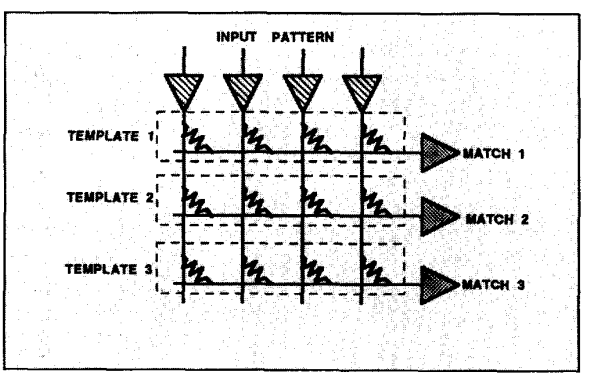

Fig. 2 A network architecture for template matching. A template is stored along the input wire of a neuron. If the transfer characteristics of an amplifier is linear, the output voltage is proportional to the inner product of the stored template and the input pattern. If the characteristic is a threshold function, the output will be high if there is a high "similarity" between the input pattern and the template. This operation can be used for feature extraction in pattern recognition applications.

All the problems neural networks solve can also be solved with alternative methods, and many algorithms have been developed for the tasks mentioned above. But evidence is mounting that neural networks can provide the most efficient solution for some classes of problems (see reports on speech recognition and vision in [3]).

The effectiveness of a neural network algorithm strongly depends on the hardware that executes it. In simulations on a computer one has to step time-sequentially through each interconnection to update the state of a neuron, a process that is painfully slow when the number of interconnections is large. Only with special purpose hardware can one hope to exploit the parallelism inherent in neural network models. So far, most applications of neural networks have been simulations on standard computers. Most analog hardware implementations are still in the research stage and only a few designs have been applied to "real world" applications.

Computing with Analog Networks

In the following paragraphs circuits implementing several different "neural" algorithms are described.

Template Matching

A very efficient use of the circuit shown in Fig. 1 is template matching. In this application a pattern is compared with a list of templates stored in a network organized as shown in Fig. 2, and the similarities between the input pattern and the stored templates are computed. Equation (2) shows that the model neuron can be used to compute inner products of vectors. If the connections T_{ij} along the input wire of a neuron represent the components of one vector and the inputs represent the components of the other vector, then the current flowing into a neuron is proportional to the inner product of the two vectors:

$$I_i = \sum_{j=0}^{N} Vout_j \, T_{ij} \propto \langle a,b \rangle \qquad (3)$$

- a — vector (pattern) with components represented by the inputs
- b — vector (template) with components represented by the connections

This is a very useful operation with many applications in pattern recognition. The network can compute a large number of inner products in parallel which makes it a very powerful processor. A microelectronic neural network performing this operation has been used with good success as a coprocessor of a workstation in pattern recognition experiments.[4]

Associative Memory

In a conventional memory each stored word is retrieved by providing its address. In an associative memory there is no address per se; a memory word is retrieved by providing part of the word itself, possibly with some errors. If the given key is a reasonable match to the corresponding

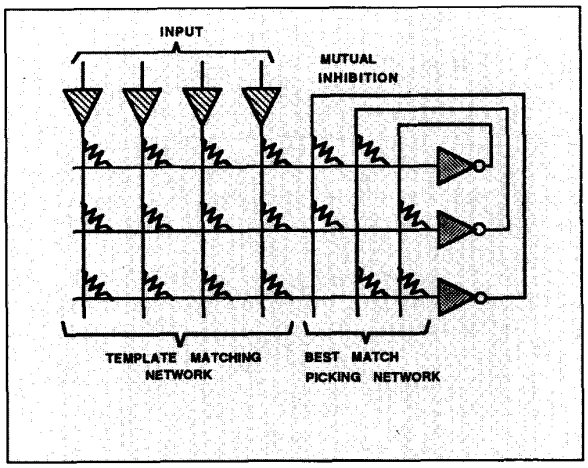

Fig. 3 An associative memory network. The left part of the network compares the input with all the stored templates, and the right part chooses the best match.

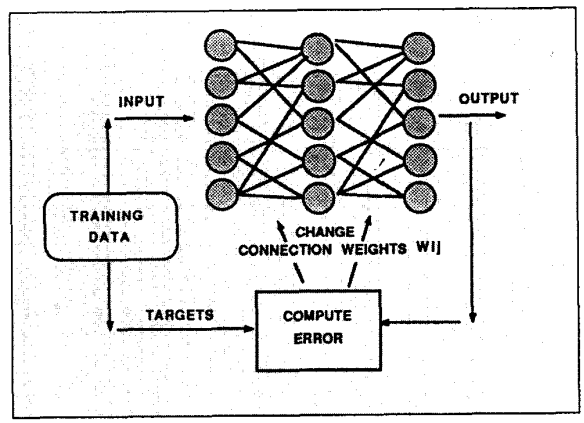

Fig. 4 Schematic of a circuit for supervised learning. A supervisor outside the network compares the actual outputs of the network with the desired outputs and makes adjustments to the connections in the network.

part of the stored word, the entire corrected word will appear at the memory output. This is reminiscent of the way the human memory seems to behave—one aspect of a memory evokes many other associated ideas. The neural network performs this function by coding the stored memories in the resistive interconnections. There are a number of ways of coding the interconnections to act as a content-addressable memory.[5-7] From a practical point of view it has become apparent that a simple template matching network followed by a maximum selector is the most efficient way of implementing this function in hardware. Fig. 3 shows the schematic of such a network. The left part of the network is a template matching network as described above and the right hand side consists of inhibitory interconnections among the output neurons. If these inhibitory connections are much stronger than the excitatory connections in the template matching part, then only one output neuron will be high in a stable state. If several neurons are on they will inhibit each other and only one of them will survive the fight while all the others are turned off. The output neuron that is getting the strongest input from the template matching network is the one that is turned on. Therefore, the one neuron that is high indicates which stored template best matches the input pattern. Several analog microelectronic circuits performing the associative memory function have been built. [8-10]

Learning

One of the most interesting aspects of neural networks is their learning capability. A wide variety of learning algorithms have been developed.[6, 11-12]. In a neural network learning is done by adaptively changing the interconnection strengths between the neurons. In this way, for example, a classifier can be built, not by programming the network, but by presenting it with a number of training examples and allowing the network to build up the discriminant function automatically. The learning capability of multilayered networks is one of the most active areas of neural network research right now.

Fig. 4 shows a schematic arrangement for supervised learning. The network is presented with a set of training examples. For each example, the output of the network is compared with the desired output and slight adjustments to the interconnection strengths in the network are made. With a proper weight-adjustment algorithm, e.g., the back-propagation algorithm [11], numerous presentations of the training data can produce a network that gives the correct input-output relations for the training data. If the network has the proper architecture and if there are sufficient training data [13], the network may be able to generalize, i.e., it will also give correct outputs for input data it has never seen before.

From a hardware point of view the most important aspect is that most learning techniques require interconnection weights that are adjustable in small steps. Such an interconnection requires considerable circuitry and is difficult to build in a small area. Various approaches to build networks with a high resolution in the weights are being explored; a few of them are described in the next section.

For many learning schemes an analog implementation may not be suitable. For example, it seems that during learning, the back-propagation algorithm requires a resolution of more than 8 bits in the weights in order to learn a problem large enough to be of practical interest. Analog circuits with such a high precision can be built but the advantage of a smaller area compared with a digital circuit is lost when the precision has to be too high. Therefore, analog circuits are of greatest interest where only moderate precision is required.

In the evaluation phase the network is very tolerant to low precision in the weights as well as in the neuron states. Typically, there are large numbers of inputs contributing to one result and random errors are reduced due to averaging. In one example, in a network with 60,000 weights that was trained to recognize hand-written digits, the resolution in the weights was reduced to five bits and the neuron states were quantized to just three levels throughout most of the network. Despite this reduction in resolution the performance of the network remained unchanged compared with the network that had the full precision of

*private communication by Y. LeCun

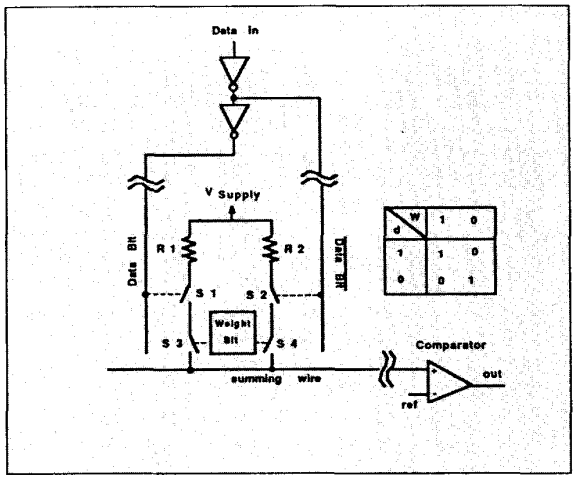

Fig. 5 A binary interconnection where the multiplication of the weight value with the input signal is provided by a XOR gate.

32 bits in the weights and the neuron states.* For the training with the back-propagation algorithm, however, the full precision was required.

This does not mean that analog circuits are limited to the evaluation phase and are of no interest for learning. But learning algorithms have to be chosen that are tolerant to imperfections of the hardware such as low precision.

Two-dimensional Resistor Networks

Circuits inspired by the architecture of the retina have been built by designers at Caltech. [14-15] These networks consist of grids of locally connected resistors plus photosensitive transistors. The circuits work directly with light input and can execute low-level vision functions such as computing spatial and temporal gradients of the light intensity.

Examples of Analog Implementations

The most important element in a network is the interconnection. There are large numbers of interconnections in the network and typically the number of neurons that can be integrated on a chip is limited by the area required for the interconnections. In Fig. 1 the interconnections are drawn as simple resistors. Depending on the function of the network, the interconnections may have to be programmable and may require several bits of analog depth.

Networks with fixed value resistors are of interest for applications where the function the network has to execute is known in advance and no changes will be needed during operation. The advantage of fixed-value resistors is their small size. Resistors made of αSi were built as small as 0.25μm × 25μm.[16] With a density of four resistors per square μm, 4×10^8 resistors could be packed into 1 cm^2. Various other materials beside αSi have been tested such as Ge:Cu, Ge:Al, or cermets [17].

A network is more flexible if the interconnections are programmable. To achieve this, a storage cell for the weight is needed plus the connecting element, e.g., a resistor or a current source, controlled by the weight. Various ways of implementing these two elements have been explored.

If the input signals as well as the weights are binary, the multiplication between the neuron signal and the weight value reduces to a simple logic function (AND, XOR). An interconnection element with this function can be built in a small area. Fig. 5 shows an example of such a connection.[18] A static memory cell stores the weight bit and an XOR gate controlled by the weight and a bit of the input signal executes the multiplication. If switches S1 and S3 or S2 and S4 are enabled, current flows through resistor R1 or R2 into the summing wire where all the contributions from the interconnections are added. The total current is then compared with a reference current in a comparator. A photomicrograph of the whole circuit is shown in Fig. 6. It essentially implements the circuit shown in Fig. 2. The network stores 46 templates, each 96 bits long, hence there are 4,416 connections in the circuit. It has been designed for machine vision applications where it can execute tasks such as feature extraction. Analog computation is used only internally; the input as well as the output data are digital which makes integration in a digital system straightforward. This circuit does a computation every 100ns which corresponds to an evaluation of 44 billion connections per second. An older version of this circuit has been used extensively for two years in machine vision experiments. A recognizer for handwritten digits achieving state of the art recognition rates was developed using this chip as a coprocessor on a workstation. In this application the network performs the computationally intensive tasks of line thinning and feature extraction.[19]

For some applications, analog depth in the interconnections is required. If the weights are stored digitally, then some sort of digital-to-analog converter is needed at each interconnection. Fig. 7 shows an example of a multiplying D/A converter.[10] The transistors controlled by the input voltage work as current sources and their widths are ratioed to deliver a current of 1, 2, 4 and 8 times the basic current. Bits B0 to B3 control switches that connect these

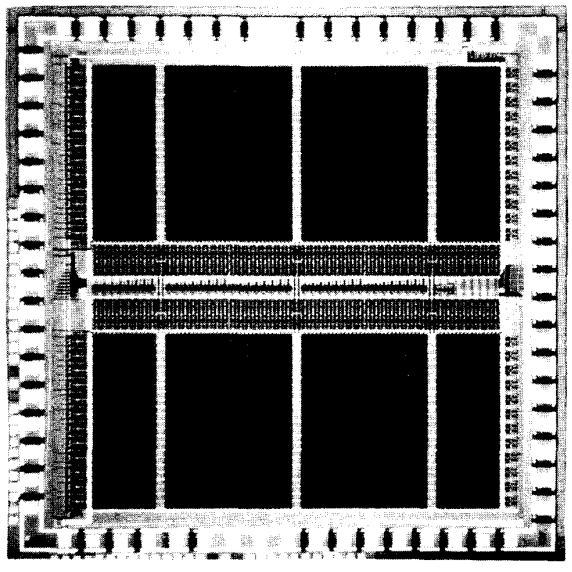

Fig. 6 A photomicrograph of the neural network chip using the interconnections shown in Fig. 5. The size of this circuit is 6.7mm × 6.7mm. It is fabricated in 2.5μm CMOS technology and contains about 70,000 transistors.

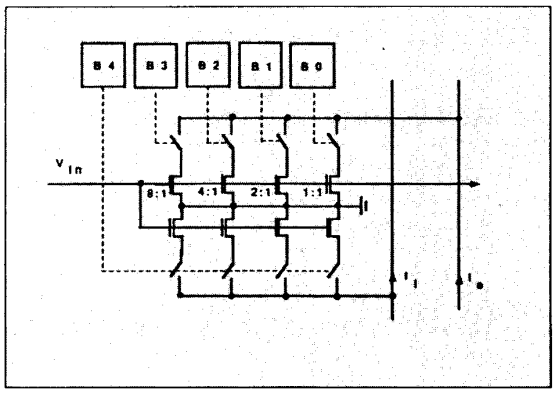

Fig. 7 Schematic of a digital interconnection with a resolution of four bits plus sign. The circuit does a D/A conversion of the weight value and an analog multiplication of the input signal with the weight (after [10]).

current sources to the positive summing wire. Bit B4 is the sign. If it is high, all the bottom switches are enabled, drawing a current from the negative summing wire. In this way positive or negative currents can be produced to give the weight a resolution of four bits plus sign. The contributions from all the interconnections are summed on the two wires. The two currents are subtracted from each other in a current mirror and the result is the input for the neuron. A matrix with 1024 such multiplying D/A converters has been built in CMOS technology. The circuit has been connected to external amplifiers and has been tested as an associative memory. [10]

The analog connection strength can also be stored as a charge package on a capacitor. Several groups are working on this concept [20-21] which has the potential for variable weight values with a high resolution and relatively small cell size. However, this dynamic storage technique requires refreshing since the charge on the capacitor leaks away. Refreshing analog values requires considerable overhead that may offset a lot of the advantage gained in smaller interconnection size. Fig. 8 shows an interconnection using two capacitors to store the weight. The difference in the voltages on the two capacitors provides the value of the weight. An analog multiplier multiplies the input voltage with the weight value and the output current is proportional to this product. A test matrix with 1020 such interconnections has been built.[20]

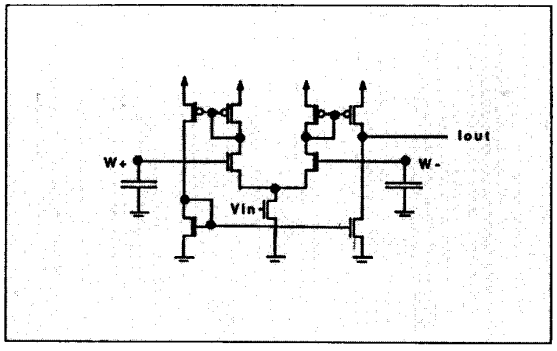

Fig. 8 An interconnection where the weight value is stored in analog form as charge packages on the two capacitors. The circuit multiplies the voltage difference of the two capacitors with the input voltage.

The circuits mentioned above have programmable interconnections but the weight values have to be computed externally and are then loaded onto the chip. Simple on-chip learning has been demonstrated in one design where digital circuitry was added to the interconnections to update the weights automatically based on local information. A small network with 6 neurons and 15 interconnections of this type has been built.[22]

Another learning chip implements an algorithm developed by Kohonen.[6] It contains 16 nodes and 112 interconnections.[23]

The designs described so far are all fabricated in standard CMOS technology and use current summing to compute sums of products. An alternative approach is to sum charge packages instead of currents. CCD technology is ideally suited for this type of computation. An exploratory design combining MNOS devices for the storage of the weights with CCD technology has been built with 26 neurons and 169 synapses.[24]

Several research teams are investigating devices that combine the storage and the multiplying function to build smaller interconnections. One potential device is the floating gate MOS transistor. This device combines nonvolatile storage (possibly with analog depth) and the connecting element in one device. The channel conductance hence the connection strength is determined by the charge stored on the floating gate.

Several material systems that change their resistivity when an electrical programming pulse is applied have been used as variable connections. After the programming pulse is removed, the resistor value is constant while the device is used for the computation. Interconnections that can be written once have been built using αSi:H.[25] A material that can be programmed repeatedly is even more desirable. WO_3 has shown such behavior in a series of tests.[26] The programming speed is slow, on the order of seconds, but this may be improved. A key issue for such a material is that it is compatible with VLSI processing technologies.

Discussion

So far, hardware implementations of neural networks are primarily explorations of various design possibilities. A comparison of the number of interconnections in the various circuits mentioned above indicates the tradeoff between the complexity of the interconnections and their size. If more functionality such as high resolution or learning capability is put into an interconnection, fewer can fit onto a chip. The optimal solution depends on the application and on the system in which the network is integrated.

The computational speed of analog networks built so far lies typically between 10^9 and 10^{11} interconnections per second [27], a much higher rate than digital circuits can achieve. Board level emulators have been built with a speed of 10^6 to 10^7 interconnections per second. An emulator, on the other hand, can have much higher resolution in the interconnections and neuron outputs and it is much more flexible than a hardware network. Most of the analog neural network circuits have not yet been integrated into systems and therefore it is difficult to estimate their true performance in applications.

The size and the speed of the neural networks will increase as designers gain more experience with such circuits. Most of the networks described here are built in CMOS

technology with 2μm to 3μm design rules; a considerable increase of the size of the circuits can be attained by switching to 1μm or submicron technologies. A network designed recently at AT&T Bell Labs in 0.9μm technology contains 32,000 interconnections.

In addition to advances in technology, we expect a substantial improvement of the computational power from a collaboration of theorists and hardware designers. A lot of problems, e.g., how much precision is required in the interconnections to solve a task, have not yet been studied thoroughly. So far, hardware designers have primarily been trying to build circuits based on theorists' models that were in turn inspired by neurobiology. It is crucial for this field that theorists and hardware developers work closely together and that theoretical models are not only inspired by biological wetware but also take into account the limitations of the electronic hardware.

Lawrence D. Jackel received the B.A. degree in physics from Brandeis University, Waltham, MA, in 1969, and the PhD degree from Cornell University, Ithaca, NY, in 1976. His thesis research was in the physics and applications of superconducting weak links.

He has been at AT&T Bell Laboratories in Holmdel, NJ since 1975 and he is now head of the Device Structures Research Department. His current research interests include electronic neural networks, physics, fabrication, and application of devices with nanometer scale features.

He is a member of the IEEE and the American Physical Society.

Hans Peter Graf, a member of the technical staff at AT&T Laboratories in Holmdel, NJ, is conducting research on collective computing systems. He designed several micro-electronic circuits implementing connectionist neural network models. These chips are being used in machine vision experiments and he is developing algorithms for these applications. He has also been involved in the development of resistive connections with sub-micron dimensions for neural network circuits.

Mr. Graf received a Diploma in physics in 1976 and a PhD in physics in 1981, both from the Swiss Federal Institute of Technology in Zurich, Switzerland.

References

[1] B. Widrow and M. E. Hoff, "Adaptive Switching Circuits," *IRE WESCON Convention Record*, New York, 1960, pp. 96–104; also in *Neurocomputing*, J. A. Anderson and E. Rosenfeld (eds.), Cambridge: MIT Press, pp. 126–134, 1988.

[2] P. Mueller, T. Martin, and F. Putzrath, "General Principles of Operations in Neuron Nets with Application to Acoustical Pattern Recognition," in *Biological Prototypes and Synthetic Systems*, Vol. 1, E. E. Bernard and M. R. Kare (eds.), New York: Plenum, pp. 192–212, 1962.

[3] *Proc. Conf. Neural Information Processing Systems*, Denver, Colorado, 1988, to be published 1989.

[4] H. P. Graf, L. D. Jackel, and W. E. Hubbard, "VLSI Implementation of a Neural Network Model," *Computer*, vol. 21(3), pp. 41–49, 1988.

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

[6] T. Kohonen, *Self-Organization and Associative Memory*, Springer, 1984.

[7] E. B. Baum, J. Moody, and F. Wilczek, "Internal Representations for Associative Memory," *Biol. Cybernetics*, vol 59, pp. 217–228, 1988.

[8] M. Sivilotti, M. R. Emerling, and C. A. Mead, "VLSI Architectures for Implementation of Neural Networks," in *Proc. Conf. Neural Networks for Computing, American Institute of Physics Conf. Proc.*, vol. 151, J. S. Denker (ed.), pp. 408–413, 1986.

[9] H. P Graf and P. deVegvar, "A CMOS Associative Memory Chip Based on Neural Networks," *Digest IEEE Int. Solid State Circ. Conf.*, L. Winner (ed.), IEEE Cat. No:87CH2367-1, pp. 304–305, 1987.

[10] J. Raffel, J. Mann, R. Berger, A. Soares, and S. Gilbert, "A Generic Architecture for Wafer-Scale Neuromorphic Systems," *Proc. IEEE First Int. Conf. Neural Networks*, IEEE Cat. No:87TH0191-7, San Diego, Vol. IV, pp. 485–493, 1987.

[11] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing*, Cambridge: MIT Press, 1986.

[12] G. A. Carpenter and S. Grossberg, "The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network," *Computer*, vol. 21(3), pp. 77–88, 1988.

[13] J. S. Denker, D. Schwartz, B. Wittner, S. Solla, R. Howard, L. Jackel, and J. Hopfield, "Large Automatic Learning, Rule Extraction and Generalization," *Complex Systems*, vol. 1, pp. 877–922, 1987.

[14] M. Sivilotti, M. Mahowald, and C. Mead, "Real Time Visual Computations Using Analog CMOS Processing Arrays," in *Advanced Research in VLSI, Proc. Stanford Conf.* 1987, P. Losleben, ed., Cambridge, MIT Press, pp. 295–311.

[15] J. Hutchinson, C. Koch, J. Luo, and C. Mead, "Computing Motion Using Analog and Binary Resistive Networks," *Computer*, vol. 21, no. 3, pp. 52–63, 1988.

[16] L. D. Jackel, R. E. Howard, H. P. Graf, B. Straughn, and J. S. Denker, "Artificial Neural Networks for Computing," *J. Vac. Sci. Technol.*, vol. B61, p. 61, 1986.

[17] A. P. Thakoor, A. Moopen, J. L. Lamb, and S. K. Kahanna, "Electronic Hardware Implementations of Neural Networks," *Applied Optics*, vol. 26, no. 3, pp. 5085–5092, 1987.

[18] H. P. Graf and L. D. Jackel, "VLSI Implementations of Neural Network Models," in *Concurrent Computing*, S. K. Tewksbury et al. (ed.), New York: Plenum, pp. 33–46, 1988.

[19] J. S. Denker, W. R. Garner, H. P. Graf, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, H. S. Baird, and I. Guyon, "Neural Network Recognizer for Hand-Written Zip Code Digits: Representation, Algorithms, and Hardware," to appear in [3].

[20] D. B. Schwartz and R. E. Howard, "A Programmable Analog Neural Network Chip," *Proc. IEEE 1988 Custom Integrated Circuits Conf.*, IEEE Cat. No.:88CH2584-1, pp. 10.2.1–10.2.4.

[21] Y. Tsividis and S. Satyanarayana, "Analog Circuits for Variable-Synapse Electronic Neural Networks," *Electronics Letters*, vol. 23, pp. 1312–1313, 1987.

[22] J. Alspector and R. Allen, "A Neuromorphic VLSI Learning System," in *Advanced Research in VLSI, Proc. Stanford Conf.* 1987, P. Losleben, ed., Cambridge: MIT Press, pp. 351–367.

Continued on page 55

-DB(VM(2))	Loop Gain
VP(2)	Loop Phase

These additional analyses are really only approximations that hold for F << Fc (the zero dB frequency of the open loop gain) and open loop gain >> loop gain because they are dependent upon the feedback network, so use them with caution.

Even this simple example shows the wealth of feedback loop performance data available when waveform mathematics is applied to SPICE.

Loop Gain Analysis

The classical method of determining the loop gain of a feedback circuit in SPICE is shown in Fig. 2A. The large inductor and capacitor allow SPICE to find the DC operating point for the analysis, but open the feedback loop for the AC analysis and Vin is injected directly into the feedback path. The loop gain is determined from analysis of node 2 and is only accurate if the impedance looking into node 2 is much less than the impedance looking into node 3 or:

Z2 << Z3 Impedance Inequality

The drawback to this classical method is that the impedance balance in the loop has been disturbed by the inductor capacitor combination. If the loop gain was in any way dependent on the OPAMP's output impedance or the feedback network's impedance, the analysis would be in error.

The new method as shown in Fig. 2B uses the waveform mathematics capability of PSPICE along with a standard SPICE independent voltage source. The independent voltage source is an ideal entity in SPICE with zero series impedance and infinite parallel impedance. It is truly unrealizable in actual circuit design, but it can be put to good use with SPICE simulations. In Fig. 2B the voltage source itself breaks the feedback loop and does not change the impedance levels of the circuit at all. This preserves the integrity of the feedback loop while allowing a simple loop gain analysis. The loop gain in Fig. 2B is determined by:

DB(VM(2)/VM(3))	Loop Gain
VP(2)-VP(3)	Loop Phase

As with the transfer function analysis above, the open loop gain can also be found (with the same restrictions).

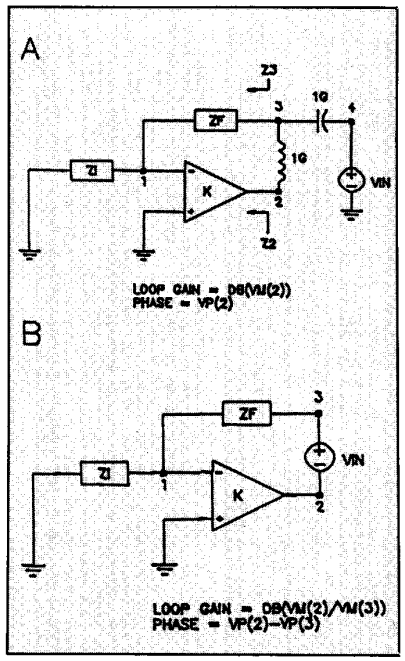

Fig. 2 A) Old method of determining loop gain. B) New method, which does not disturb circuit impedances.

The All in One Analysis

The All in One analysis method shown in Fig. 3 allows almost all loop parameters to be determined from a single run. The ideal SPICE voltage source is inserted in the loop between nodes 1 and 2, allowing the following loop parameters to be determined:

DB(VM(3)/VM(2))	Open Loop Gain
VP(3)-VP(2)	Open Loop Phase
DB(VM(1)/VM(2))	Loop Gain
(VP(1)-VP(2)	Loop Phase

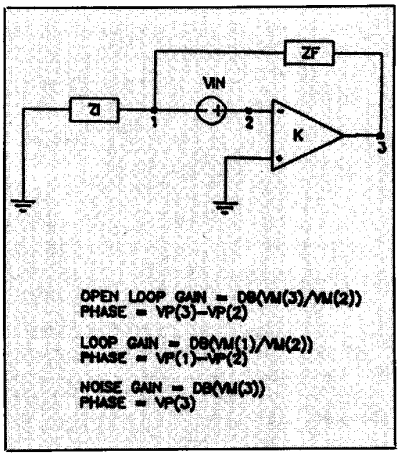

Fig. 3 The "Big Gun" analysis. Almost all loop parameters are available with one PSPICE run.

DB(VM(3))	Noise Gain
VP(3)	Noise Phase

All of these important loop parameters can be determined with a single PSPICE run with no penalty on run time since PSPICE stores the results of each node in the analysis anyway. With a circuit setup as in Fig. 3, important loop parameters like gain and phase margins can be determined along with how the circuit will respond to amplifier noise.

Putting It All Together

Each of these methods may be used separately or any and all source combinations can be written into a SPICE netlist at the same time. Simply set the desired source on with a "AC 1" statement or off with a "AC 0" statement for the particular analysis to be run. For example, the All In One analysis can be used with the Transfer Function analysis and by setting the desired sources either on or off, the proper SPICE run can be made. This is because an independent source in SPICE when turned off displays ideal passive characteristics and will not disturb the circuit's impedance levels.

Analog Electronic Neural Network Circuits—Continued from page 49

[23] J. Mann, R. Lippmann, R. Berger, and J. Raffel, "A Self-Organizing Neural Net Chip," *Proc. IEEE 1988 Custom Integrated Circuits Conf.*, IEEE Cat. No.:88CH2584-1, pp. 10.3.1–10.3.5.

[24] J. P. Sage, K. Thompson, and R. S. Withers, "An Artificial Neural Network Integrated Circuit Based on MNOS/CCD Principles," in: *Neural Networks for Computing, American Institute of Physics Conf. Proc.*, vol. 151, J. S. Denker (ed.), pp 381–385.

[25] A. P. Thakoor, J. L. Lamb, A. Moopen, and J. Lambe, "Binary Synaptic Connections Based on Memory Switching in a-Si:H," in *Proc. Conf. Neural Networks for Computing*, Snowbird, Utah, 1986, J. S. Denker (ed.), *American Institute of Physics Conf. Proc.*, vol. 151, pp. 426–431.

[26] "JPL Thin-Film Solid-State Memistor," in *DARPA Neural Network Study*, AFCEA Int. Press, p. 613, 1988.

[27] *DARPA Neural Network Study*, AFCEA Int. Press, 1988.