A HARDWARE EFFICIENT CASCADABLE CHIP SET FOR ANN'S WITH ON-CHIP BACK-PROPAGATION

TORSTEN LEHMANN
The Computational Neural Network Center, Electronics Institute,
Technical University of Denmark, Building 349, DK-2800 Lyngby, Denmark.

In this paper, an analogue, cascadable, CMOS chip set for artificial neural networks is presented. The chip set (a synapse chip and a neuron chip) offer on-chip back-propagation learning in a fully parallel, layered, feed-forward network of arbitrary size and topology. The learning scheme is implemented with no extra circuits at the synapse sites (compared to the system without the learning scheme) and extra circuits of a complexity only about the same as the neurons at the neuron sites. Also, no additional wiring is required by the learning scheme. Measurements on an experimental chip set are presented.

1. Introduction

Analogue neural networks favour in speed over digital ones only in the case of massively parallel implementations. The slower, analogue processing elements can be much smaller than their digital equivalents — thus a larger number can be integrated on a chip. Also, the performance of a general purpose ANN (Artificial Neural Network) chip with many parallel processing elements is usually far from its peak performance, when it is applied to real-world applications[1], because the parallelism can not fully be exploited.

As a consequence of this, one could argue (on the extreme) that there are two ways of implementing dedicated hardware for ANN's:

- either a general purpose system with one powerful processor (ie. a DSP) to ensure efficient usage of silicon area at all times
- or an application specific, fully parallel system (ie. an analogue ANN) to exploit the inherent parallelism when the network architecture is known.

In this work we have leaned towards the second solution.

Using a host computer to train such a system has the advantage of easy changeable learning algorithms. There is, however, reasons for using dedicated hardware for the learning algorithm as well as for the ANN. One is that the learning algorithm typically is computationally a very heavy task: the back-propagation learning algorithm[2], for example, requires $O(N^4)$ multiplications in a system with N neurons, compared to the $O(N^2)$ multiplications required by the ANN itself. This, of course, calls for parallel computations. Another reason is that a host computer is not necessarily available: in small, battery powered, "intelligent" devices, for instance, using adaptive ANN's this would be the case.

As the back-propagation algorithm is widely used, this motivates the investigation of possible implementations of a parallel, analogue ANN with on-chip back-propagation.

Others have also looked at analogue ANN's with on-chip back-propagation[3,4,5], but the complexity of the hardware involved is a serious drawback[4,6,7]. A very complex piece of hardware for the learning scheme is clearly undesirable because it reduces the complexity of the ANN for a given silicon area and it is idle (and hence a "waste") after the learning phase of the ANN. For this and other reasons (eg. non-ideal components) perturbated and new algorithms have emerged[3,4,6,7].

In this paper, we show how it is possible to implement the back-propagation algorithm with no extra hardware at the synapse sites and hardware at the neuron sites of twice the complexity of a previously reported neuron[8]. Also, there is no wiring needed beyond what is required by the fully parallel feed-forward ANN. In section 2 the basic system architecture is described. In section 3 it it shown how this system can be expanded to include on-chip back-propagation. In section 4 results from an experimental chip set is presented. Conclusions are drawn in section 5.

2. Basic Architectures

The basic architectures of an analogue, scalable ANN have been reported elsewhere[8] but will be repeated here — in a slightly different version — to illuminate the final architectures including the learning extensions.

As an alternative to an analogue ANN chip dedicated to a specific application, we have chosen to develop a system that is scalable — that is, it can be expanded to an arbitrary size, fully parallel network[9] (in principle). The idea is to build a hardware system for a specific application with a number of "building block" components.

The usual way to make a scalable ANN is by splitting the synapses and the neurons on different chips[9]. The synapse chip (which is a matrix-vector multiplier) should then have voltage inputs and current outputs, which ensures that several synapse chips can be cascaded and thus implement a matrix-vector multiplier with more inputs and/or outputs than a single chip (cf. Fig. 4). The neuron chip (which consists of a vector of non-linear functions) should have current inputs and voltage outputs.

2.1. *The Synapse Chip*

The central circuit on the synapse chip is the four quadrant analogue multiplier[10]. This should have a current output, so the output from a row of synapses easily can be accumulated, and a voltage input, so the output from a neuron can be distributed to a column of synapses[9,10].

The multiplier chosen is the *MOS Resistive cell* with four transistors shown in Fig. 1a. It can be shown[11] that the *output current*, $i_{wx+} - i_{wx-}$ is given by

$$i_{wx+} - i_{wx-} = \beta V_w v_x = \frac{1}{R_{\text{diff}}} v_x \quad , \qquad (1)$$

where $\beta = W/L \cdot \mu_n C_{\text{ox}}$ is the *transconductance parameter* for the MOS transistors and R_{diff} is the differential resistance seen from the input terminals. It should be noted that the transistors are operated in the triode region and that the output terminals must be at the same potential.

In addition to the multiplier, the synapse circuit consists of an analogue memory[10]: the weight, w_{ij} (cf. below) is stored as the differential voltage V_w on the gate capacitances of the transistors in the MOS Resistive cell[12]. To access the weight, a nand gate and two MOS switches are required at each synapse site. The synapse weights can be accessed randomly, one at a time.

The schematic of the circuit on the synapse chip that participates in the matrix-vector multiplication is shown in Fig. 2. The current conveyors[13] subtract the accumulated synapse output currents, while ensuring virtual short-circuits of the synapse outputs (as required):

$$i_{s_i} = \sum_j i_{w_{ij}x_j+} - \sum_j i_{w_{ij}x_j-} = \beta \sum_j v_{x_j} V_{w_{ij}} \quad . \qquad (2)$$

The output potentials of the current conveyors (ie. the input potentials of the following neuron chip) should be close to V_{ref} to avoid DC common mode currents in the synapses.

2.2. *The Neuron Chip*

The principal schematic of a neuron on the neuron chip is shown in Fig. 3. The resistors are implemented as MOS Resistive cells and the bipolar transistors are the parasitic devices inherent in a usual CMOS process[14].

The *neuron transfer function* is given by[8]

$$v_{y_i} = R_{\text{or}} \alpha I_{\text{b}} \tanh\left(\frac{R_{\text{is}} i_{s_i}}{2V_t}\right) \quad , \qquad (3)$$

where $\alpha \approx 0.5$ and $V_t = kT/q$ is the *thermal voltage*.

3. Extensions for On-Chip Back-Propagation

The back-propagation learning algorithm for a layered feed-forward neural network can be described as follows[2]. The *neuron i activation*, y_i^l in *layer l* is

$$y_i^l = g(s_i^l) = g\left(\sum_j w_{ij}^l x_j^l\right) = g\left(\sum_j w_{ij}^l y_j^{l-1}\right) \qquad (4)$$

where $g(.)$ is the *squashing function* (in our case $g(.) \equiv \tanh(.)$), s_i^l is the *net input* to neuron i, x_j^l is *input j* to layer l and w_{ij}^l is the *connection strength*. The *threshold* is implicitly given as the strength from a constant input, $x_0^l = -1$.

For a given *input pattern*, $\{x_1^1, x_2^1, \ldots, x_{N_1}^1\}$ and *target values*, d_i for the neurons in the *output layer*, L, we have the neuron *error*

$$\epsilon_i^l = \begin{cases} d_i - y_i^l \quad , & \text{for } l = L \\ \sum_j w_{ji}^{l+1} \delta_j^{l+1} = \sum_j w_{ji}^{l+1} g'(s_j^{l+1}) \epsilon_j^{l+1} \quad , & \text{for } 1 \le l < L \end{cases} \quad . \qquad (5)$$

The connection strengths should then be updated as

$$\left(w_{ij}^l\right)_{\text{new}} = \left[\left(w_{ij}^l\right)_{\text{old}} + \eta \delta_i^l y_j^{l-1}\right](1 - \epsilon_{\text{dec}}) \quad , \qquad (6)$$

where η is the *learning rate*, δ_i^l is defined implicitly by (5) and ϵ_{dec} is an optional *weight decay* parameter.

3.1. *The Back-Propagation Synapse Chip*

Studying the equations above, we notice that the matrix, $\underline{\underline{w}}^l$ used in the forward propagation (4) (we have $\underline{s}^l = \underline{\underline{w}}^l \underline{x}^l$) is the transposed of the one used in the backwards propagation (5) (can be written as $\underline{\epsilon}^{l-1} = \left(\underline{\underline{w}}^l\right)^T \underline{\delta}^l$). In terms of the synapse chip, the transposition is equivalent to *exchanging the inputs and the outputs.* This also satisfies another requirement of a hardware implementation of the back-propagation algorithm: the signal flow is in the opposite direction when $\underline{\epsilon}^{l-1}$ is calculated compared to when $\underline{s}^l$ is calculated.

Now, looking at the synapse multiplier, Fig. 1 we see that it is perfectly symmetric — that is, if we apply a differential voltage, v_δ at the former output nodes and ensure virtual short-circuit between the former input nodes, the difference current, $i_{w\delta+} - i_{w\delta-}$ injected to the former input nodes would be

$$i_{w\delta+} - i_{w\delta-} = \beta V_w v_\delta = \frac{1}{R_{\text{diff}}} v_\delta \quad . \tag{7}$$

This is illustrated in Fig. 1b.

This implies that we can enable the multiplier in Fig. 2 to perform the multiplication with the transposed matrix on the appropriate signals, simply by exchanging the current conveyors and the voltage buffers in Fig. 2.

Further, when several modified synapse chips are cascaded, it is still valid that multiplication with the transposed matrix is performed when inputs and outputs are exchanged.

Finally, (6) says that we must be able to multiply δ_i^l by y_j^{l-1}. The back-propagation synapse chip must thus be able to route the input signal $v_{y_j^{l-1}}$ directly to the output pins (cf. next section).

If the current-conveyors in Fig. 2 are implemented as supply-current sensed op-amps[13], the components needed for each row/column on the chip are: two op-amps, two current mirrors and 11 switch transistors (plus row/column decoders, synapses etc.). Care must be taken on the placement of the switch transistors to ensure virtual short-circuit of the synapse outputs.

We have now seen that *the extension to bidirectional performance suitable for back-propagation is possible with virtually no extra components for the cascadable synapse chip.* Also, *no routing is needed beyond what is necessary for the implementation of a fully parallel feed-forward system,* which is not necessarily the case in other on-chip learning implementations[3,5,7].

Hardware responsible for the weight update (basically two multipliers and an adder) is often placed at the synapse sites[3,5,7]. As this enables all synapse weights to be updated in parallel, this gives an order $O(N^2)$ improvement in speed compared to a serial implementation of the algorithm. When placing the weight updating hardware outside the synapse chip core, only weights from one neuron in layer $l-1$ to the neurons in layer l can be updated simultaneously (cf. below). Often a serial access to the weights is used which further limits the weight updating to one weight at a time. In other words: our proposed synapse chip architecture gives at most an order $O(N)$ improvement in speed compared to a serial approach. This is the price for not using extra synapse hardware.

The operation of the back-propagation system proposed is illustrated in Fig. 4. During normal operation all chips are in *forward* mode and the response to an input pattern is propagated to the output after a certain delay. When a synapse weight, w_{ij}^l in a layer l is to be updated, all chips in the previous layers operate in *forward* mode and produce the y_k^{l-1}'s. All the chips in the layers following layer l operate in *reverse* mode and produce the ϵ_i^l's. The synapse chips in layer l operate in *route* mode and route y_j^{l-1} to the inputs of the neuron chips in layer l. These in turn operate in *learn* mode and calculate $(w_{ij}^l)_{\text{new}}$.

3.2. *The Back-Propagation Neuron Chip*

The calculation of $(w_{ij}^l)_{\text{new}}$ for $i \in \{1, 2, \ldots, N_l\}$ (cf. above) can be performed in parallel or sequentially. As the access to the weights on the synapse chips are "one at a time" (as well as the access to the weights in a RAM backup that would typically be used for the analogue on-chip storage[10]), we have chosen the sequential approach. A block diagram of the neurons on such a back-propagation neuron chip is seen in Fig. 5. (Note that the derivative $g'(.)$ is calculated as $1 - g^2(.)$ as $g(.) \equiv \tanh(.)$.)

The multipliers in Fig. 5 are based on the MOS Resistive cell and a total of six op-amps per neuron is needed. The area of the back-propagation neuron is approximately twice the area of the neuron previously reported[8].

It should be noted that the back-propagation system architecture as described is compatible with several improvements to the algorithm as the eta finder algorithm[15], the Fahlman cost function[2] and weight decay.

4. Experimental Results

An experimental chip set, using the principles just described, has been fabricated in a standard $2.4\,\mu$m CMOS process (4 neurons and 8×4 synapses per chip). A summary of the chip characteristics is given in Table 1. Fig. 6 shows measured synapse characteristics in forward mode and Fig. 7 shows the characteristics in reverse mode for the same synapse. The non-linearity[a] is less than 3.5%. The output current (i_s) offset error can be quite large, however. Of the magnitude of a single synapse maximum output current (caused by the current conveyors). This is not necessarily a problem in recall mode but might be in learning mode, in which case auto offset cancelling techniques must be employed. The synapse weight resolution[b] is at least 10 bit which should be sufficient for a range of applications. (Note that a smaller weight change is possible during learning, thought it can not necessarily be distinguished at the output.)

Fig. 8 shows measured neuron transfer characteristics. The non-linearity is less than 2% and the offset errors observed are probably of less importance to the recall mode performance. Fig. 9 shows the derivative of the neuron transfer function as calculated by the neuron chip (as $dy/ds = 1 - y^2$). The asymmetry is caused by the output offset of the neuron transfer function and the input offset of the "$1 - y^2$" block. The non-linearity is less than 6%. The offset errors are quite large and whether they are compatible with the back-propagation algorithm has yet to be experimentally proven. In particularly poor specimens of the neurons, the measured derivative can be negative which is destructive for the learning process. There are two possible solutions to this problem: (i) to use a Fahlman cost function (where $g'(s)+\gamma$ substitutes $g'(s)$ in the weight updating rule) with a sufficiently large γ or if this fails (ii) to employ auto offset cancelling techniques. Alternatively, the derivative can be calculated using a difference quotient approach[16].

The variations on transconductances, gains, offset errors (hopefully), etc. should be cancelled during the learning process.

The synapse- and neuron chip propagation delays give a layer propagation delay of $t_{\text{layer pd}} < 2.5\,\mu$s which gives 12.8 MCPS (Mega Connections Per Second) per synapse chip in a layer (or 4 GCPS if a full-size 100×100 synapse chip is used). The neuron chip takes $t_{x\Delta w\ \text{pd}} \approx 3.6\,\mu$s to calculate a new weight giving approximately 0.25 MCUPS (Mega Connection Updates Per Second) for the back-propagation system. Given the droop rate of the neuron activation sampler, this will limit the system size to about $2 \cdot 10^6$ connections for an 8 bit accuracy of the neuron activations. (Applications using $0.3 \cdot 10^6$ connections are known[17].)

5. Conclusions

In this paper an analogue, cascadable CMOS chip set for artificial neural networks with on-chip back-propagation learning has been presented.

Firstly, the basic architectures of the chip set (a synapse chip and a neuron chip) were presented. Secondly, the necessary circuit extensions to this chip set for a on-chip back-propagation implementation was described. Finally measurements from an experimental chip set was presented.

As the synapse multiplier on the synapse chip, the four transistor MOS Resistive cell was chosen. Because of the symmetry of this cell, the chip could perform a matrix-vector multiplication using the transposed matrix if the inputs and outputs of the chip were exchanged. When several chips were cascaded to implement a matrix-vector multiplier with more inputs and/or outputs than a single chip, it was still true that exchanging the inputs and outputs resulted in a multiplier using the transposed matrix. Thus (a part of) the back-propagation algorithm can integrated on the synapse chip with virtually no extra components, potentially giving an order $O(N)$ improvement in speed compared to a serial solution.

The transfer function of the neurons on the neuron chip was chosen to be the hyperbolic tangent (implemented with a differential pair of "parasitic" lateral bipolar transistors). The derivative of this function is easily calculated and as a consequence of this, the back-propagation neuron has only approximately twice the size of the original neuron. The back-propagation neuron chip performs the required computations for the learning algorithm, not performed by the back-propagation synapse chip.

The learning algorithm thus implemented did not require much hardware overhead and no wiring apart from what was needed by the ANN itself.

Such a low hardware cost chip set has been fabricated and is currently under test. Whether the non-idealities measured (especially offset errors) are compatible with the back-propagation algorithm has yet to be experimentally proven. But as the algorithm has some requirements on the accuracy on certain signals, the measurements suggest that auto offset cancelling techniques possibly have to be employed

[a] We define the *non-linearity*, D_ξ of a quantity ξ as $D_\xi \stackrel{\text{def}}{=} \max_\xi |f(\xi) - \xi|/(\xi_{\max} - \xi_{\min})$ where $f(.)$ is a non-linear function.

[b] We define the *resolution* as the smallest weight change that can be distinguished at the output of the synapse chip.

in large analogue systems. The small size test chips have a performance of 12.8 MCPS per synapse chip in a layer and 0.25 MCUPS.

Acknowledgments

This work was performed as part of a Ph.D. study under the supervision of Professor Erik Bruun. It was supported by the *Danish Technical Research Council* and the *Danish Natural Science Council*.

References

1. L. D. Jackel, "Practical Issues for Electronic Neural-Net Hardware," tutorial notes at the *4'th Neural Information Processing Systems Conf.*, 1991.
2. J. Hertz, A. Krogh and R. G. Palmer, *Introduction to the Theory of Neural Computation*, Redwood City: Addison-Wesley Publishing Company, 1991.
3. M. Valle, D. D. Caviglia and G. M. Bisio, "An Experimental Analog VLSI Neural Chip with On-Chip Back-Propagation Learning," in *Proc. 18'th European Solid-State Circuits Conf.*, 1992, pp. 203–206.
4. P. W. Hollis, J. J. Paulos and C. J. D'Costa, "An Optimized Learning Algorithm for VLSI Implementation," *Proc. 2'nd Int. Conf. Microelectronics for Neural Networks*, 1991, pp. 121–126.
5. Yiwen Wang, "A Modular Analog CMOS LSI for Feedforward Neural Networks with On-Chip BEP Learning," in *Proc. IEEE Int. Symp. Circuits and Systems '93*, vol. 4, 1993, pp. 2744–2747.
6. M. Jabri and B. Flower, "Weight Perturbation: An Optimal Architecture and Learning Technique for Analog VLSI Feedforward and Recurrent Multilayer Networks," *IEEE Trans. Neural Networks*, vol. NN-3, no. 1, pp. 154–157, 1992.
7. A. F. Murray, "Multilayer Perceptron Learning Optimized for On-Chip Implementation: A Noise-Robust System," *Neural Computation*, vol. **4**, no. 3, pp. 366–381, 1992.
8. J. A. Lansner and T. Lehmann, "An Analog CMOS Chip Set for Neural Networks with Arbitrary Topologies," *IEEE Trans. Neural Networks*, vol. NN-4, no. 3, pp. 441–444, May 1993.
9. S. Eberhardt, T. Duong and A. Thakoor, "Design of Parallel Hardware Neural Network Systems from Custom Analog VLSI 'Building Block' Chips," *Proc. IEEE Int. Joint Conf. Neural Networks*, 1989, pp. II-183–II-190.
10. E. Sánchez-Sinencio and C. Lau, *Artificial Neural Networks*. New York: IEEE Press, 1992.
11. Z. Czarnul, "Novel MOS Resistive Circuit for Synthesis of Fully Integrated Continuous-Time Filters," *IEEE Trans. Circuits and Systems*, vol. CAS-33, no. 7, pp. 718–721, Jul. 1986.
12. Y. Tsividis and S. Satyanarayana, "Analogue Circuits for Variable-Synapse Electronic Neural Networks," *Electronics Letters*, vol. **23**, no. 24, pp. 1313–1314, 1987.
13. C. Toumazou, F. J. Lidgey and C. A. Makris, "Extending Voltage-mode Op Amps to Current-mode Performance," *IEE Proceedings*, vol. **137**, Pt. G., no. 2, pp. 116–130, 1990.
14. E. A. Vittoz, "MOS Transistors Operated in the Lateral Bipolar Mode and Their Application CMOS Technology," *IEEE J. Solid-State Circuits*, vol. SC-18, no. 3, pp. 273–279, 1983.
15. L. M. Reyneri and E. Filippi, "An Analysis on the Performance of Silicon Implementations of Backpropagation Algorithms for Artificial Neural Networks," *IEEE Trans. Computers*, vol. C-40, no. 12, pp. 1380–1389, Dec. 1991.
16. G. Bogason, "Generation of a Neuron Transfer Function and it's Derivative," submitted to *Electronics Letters*.
17. S. Brunak, J. Engelbrecht and S. Knudsen, "Prediction of Human mRNA Doner and Acceptor Sites from the DNA Sequence," *J. Mol. Biology*, no. 220, pp. 49–65, 1991.

Table 1. Measured characteristics of the back-propagation synapse- and neuron chips. The column "Bits" is the equivalent in least significant bits of the property value given an 8 bit resolution.

Property	Value	Bits	Notes
Synapse non-linearity	$D_{wx} \lesssim 3.5\,\%$	$9\,\mathrm{LSB}_8$	
Synapse transconductance variation[c]	$\Delta G_{\mathrm{syn}} \lesssim 6\,\%$	$15\,\mathrm{LSB}_8$	
Synapse chip input offset	$\lvert V_{x\,\mathrm{ofs}}\rvert \lesssim 12\,\mathrm{mV}$	$2\,\mathrm{LSB}_8$	
Synapse chip output offset	$\lvert I_{sO\,\mathrm{ofs}}\rvert \lesssim 3\,\mu\mathrm{A}$	$192\,\mathrm{LSB}_8$[d]	
Synapse weight offset	$\lvert V_{w\,\mathrm{ofs}}\rvert \lesssim 55\,\mathrm{mV}$	$7\,\mathrm{LSB}_8$	
Synapse weight resolution	$V_{w\,\mathrm{res}} \lesssim 2\,\mathrm{mV}$	$1/4\,\mathrm{LSB}_8$	
Neuron (tanh) non-linearity	$D_g \lesssim 2\,\%$	$5\,\mathrm{LSB}_8$	
Neuron input transresistance variation[c]	$\Delta R_{\mathrm{is}} \lesssim 2\,\%$	$5\,\mathrm{LSB}_8$	
Neuron output transresistance variation[c]	$\Delta R_{\mathrm{or}} \lesssim 2\,\%$	$5\,\mathrm{LSB}_8$	
Neuron input offset	$\lvert I_{sI\,\mathrm{ofs}}\rvert \lesssim 0.8\,\mu\mathrm{A}$	$51\,\mathrm{LSB}_8$	
Neuron output offset	$\lvert V_{yO\,\mathrm{ofs}}\rvert \lesssim 41\,\mathrm{mV}$	$5\,\mathrm{LSB}_8$	
Neuron derivative non-linearity	$D_{g'} \lesssim 6\,\%$	$15\,\mathrm{LSB}_8$	
Neuron derivative input offset	$\lvert V_{g'I\,\mathrm{ofs}}\rvert \lesssim 68\,\mathrm{mV}$	$9\,\mathrm{LSB}_8$	
Neuron derivative output offset	$\lvert V_{g'O\,\mathrm{ofs}}\rvert \lesssim 244\,\mathrm{mV}$	$31\,\mathrm{LSB}_8$	
Synapse chip propagation delay[e]	$t_{xs\,\mathrm{pd}} \lesssim 0.9\,\mu\mathrm{s}$	$1/2\,\mathrm{LSB}_8$	$C_L \gtrsim 16\,\mathrm{pF}$, $R_L \approx 3\,\mathrm{k}\Omega$
Neuron squashing propagation delay[e]	$t_{sy\,\mathrm{pd}} \lesssim 1.6\,\mu\mathrm{s}$	$1/2\,\mathrm{LSB}_8$	$C_L \gtrsim 16\,\mathrm{pF}$
Neuron chip weight calculation time[e]	$t_{x\Delta w\,\mathrm{pd}} \lesssim 3.6\,\mu\mathrm{s}$	$1/2\,\mathrm{LSB}_8$	$C_L \gtrsim 16\,\mathrm{pF}$
Synapse weight drift	$\delta_w \lesssim 0.2\,\mathrm{mV/s}$	$0.03\,\mathrm{LSB}_8/\mathrm{s}$	$C_{\mathrm{hold}} \approx 2 \times 1.6\,\mathrm{pF}$
Neuron activation drift	$\delta_y \lesssim 0.5\,\mathrm{mV/s}$	$0.06\,\mathrm{LSB}_8/\mathrm{s}$	$C_{\mathrm{hold}} \approx 2 \times 2\,\mathrm{pF}$

[c] The variations are for a single chip.
[d] Equivalent LSB of a single synapse.
[e] The delays are for the signals to settle within the given precision.

Figure Captions

Figure 1. The MOS Resistive (four transistor) cell used as synapse multiplier. The transistors are operated in the triode region. (**a**) operation in forward mode (**b**) operation in reverse mode.

Figure 2. Schematics of the computing elements on the synapse chip (in forward mode).

Figure 3. Principal schematics of a sigmoid neuron.

Figure 4. A three layer perceptron implemented using the chip set. The different operating modes of the various synapse- and neuron chips are indicated in the situation where the weights in the second layer are updated.

Figure 5. Block diagram of a back-propagation neuron. The elements below the dashed line are common for all neurons.

Figure 6. Measurement of synapse characteristic in forward mode.

Figure 7. Measurement of synapse characteristic in reverse mode.

Figure 8. Measurement of neuron transfer function.

Figure 9. Measurement of neuron transfer function derivative.

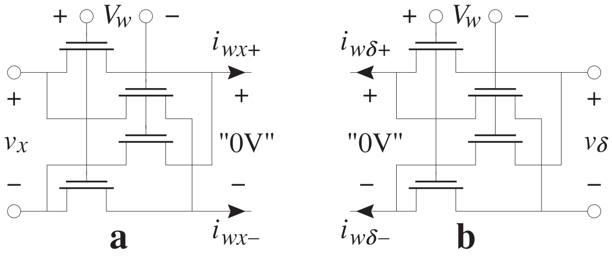

Figure 1. T. Lehmann

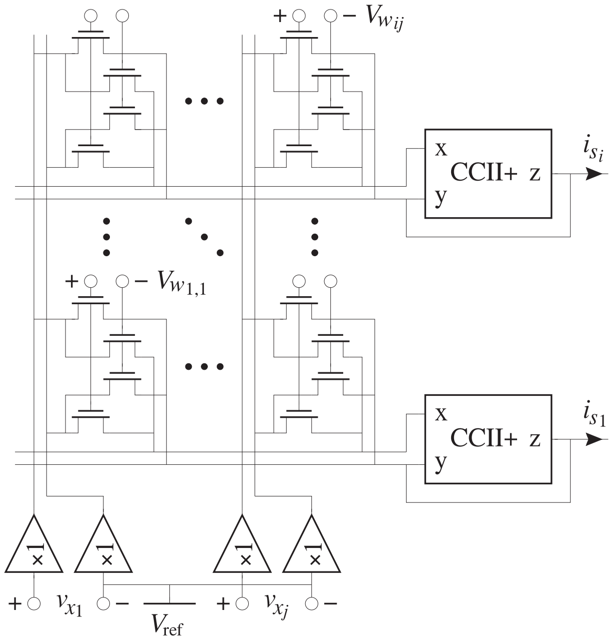

Figure 2. T. Lehmann

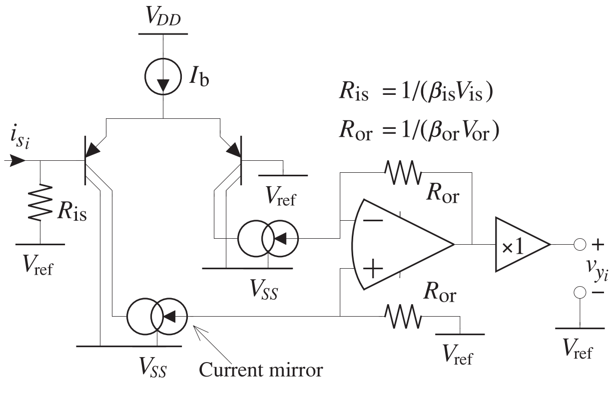

Figure 3. T. Lehmann

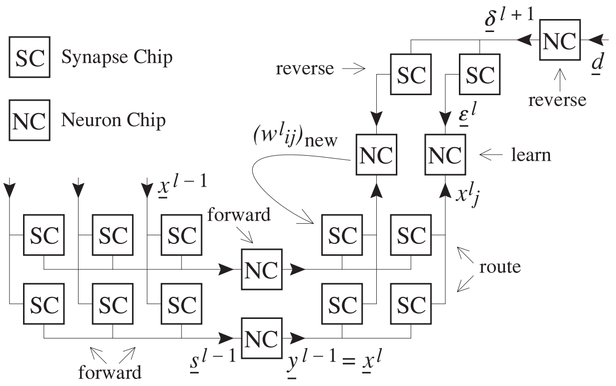

Figure 4. T. Lehmann

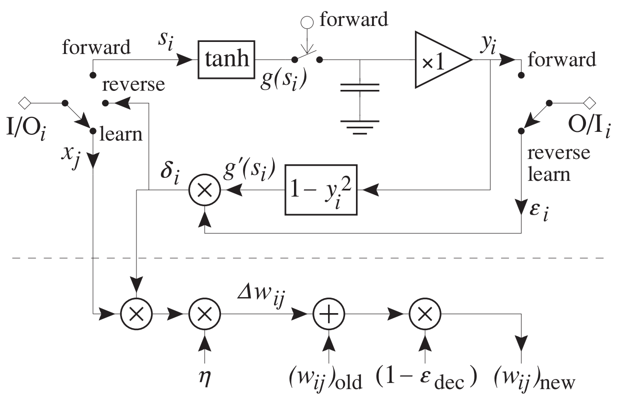

Figure 5. T. Lehmann

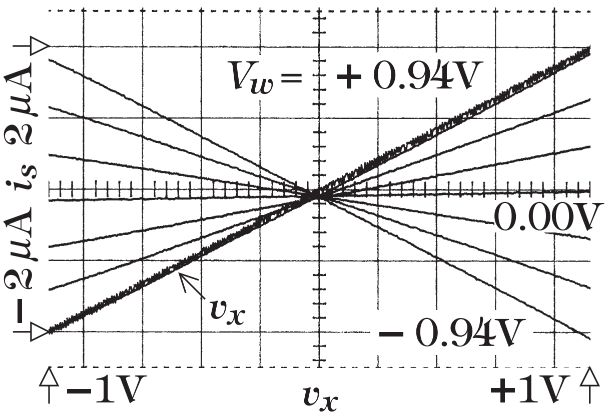

Figure 6. T. Lehmann

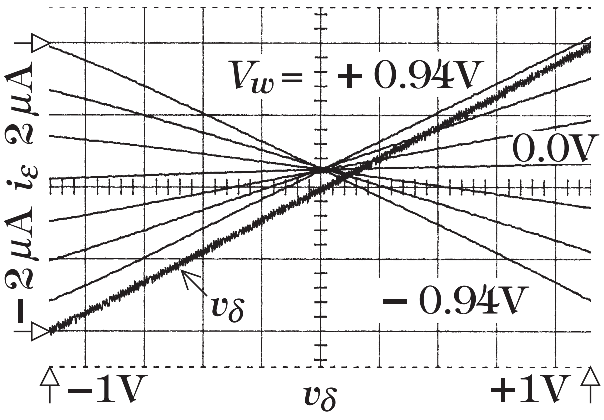

Figure 7. T. Lehmann

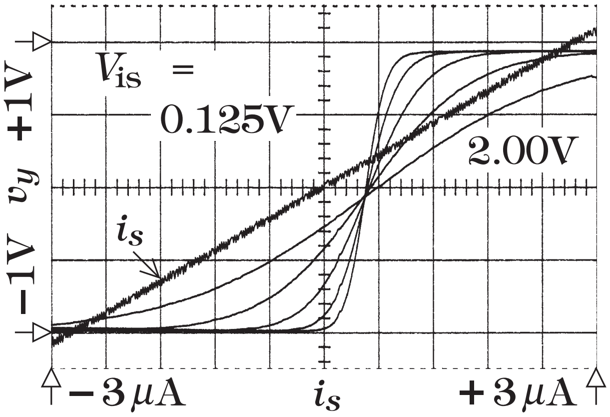

Figure 8. T. Lehmann

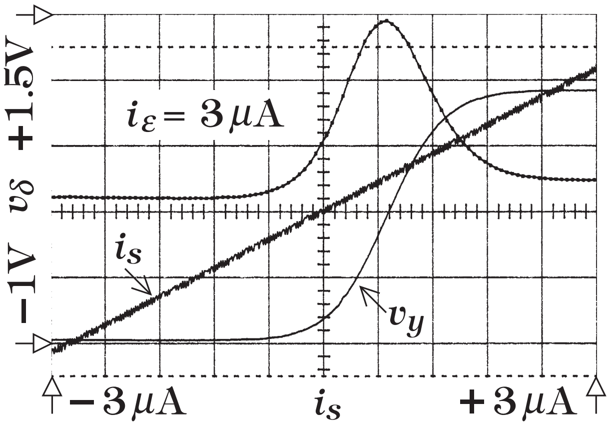

Figure 9. T. Lehmann