Analogue/Digital Hybrid VLSI Synapses for Recall- and Learning Mode Neural Networks

Torsten Lehmann, **Erik Bruun** and **Casper Dietrich***

The Computational Neural Network Center,
Electronics Institute, Building 349,
Technical University of Denmark,
DK-2800 Lyngby, Denmark
Phone: +45 4593 4207, Fax: +45 4288 0117
Emails: `tlehmann@ei.dtu.dk`, `bruun@eiffel.ei.dtu.dk`

Abstract — *In this work we present a hardware efficient matrix-vector multiplier architecture for artificial neural networks with digitally stored synapse strengths. We present a novel technique for manipulating bipolar inputs based on an analogue two's compliments method and an accurate current rectifier/sign detector. Measurements on a CMOS test chip are presented and validates the techniques. Further, we propose to use an analogue adjustment, based on a simple capacitive storage, for enhancing weight resolution during learning. It is shown that the implementation of Hebbian learning and back-propagation learning in this system is possible using very little additional hardware compared to the recall mode system.*

1. INTRODUCTION

One of the major challenges in analogue VLSI implementations of artificial neural networks (ANNs) is the issue of weight storage [1]. Presently no good analogue memory exists and designers are usually obliged to discretize the synapse strengths (e.g. using digital storage) or use devices requiring non-standard processes, e.g. floating gate devices. Though digital storage has a severe area penalty and floating gate devices are tedious to program, successful system implementations of both types have been reported in the literature [2, 3]. Primarily recall mode systems, that is. For analogue systems with on-chip learning simple capacitive storage [4] seems to be the more popular choice because the value stored in such a memory is easy to adjust and is not subject to discretizing effects which usually degrades learning [5]. Weight deterioration in such systems is a serious problem, though.

In this paper we are concerned with the issue of teaching analogue VLSI neural networks with digitally stored synapse strengths: We have previously reported a hardware efficient architecture for the implementation of digital weight storage, which minimizes the area penalty of this — in other respects — attractive solution [6]. Below, we prove the validity of this multiplying DAC synapse (MDAC synapse) by measurements on an experimental matrix-vector multiplier chip. This chip includes a novel scheme for manipulating the bipolar neuron activations which are preferred in many learning

This work was supported by the Danish Technical Research Council and the Danish Natural Science Council.

* Now with GIGA A/S, NKT Allé 1, DK-2605 Brøndby.

systems. With respect to learning, we propose to use an "analogue adjustment" to the digital weight storage for enhancing the weight resolution temporary, during learning — a method compatible with "standard" analogue learning implementations. Several hardware implementations of learning schemes are outlined.

2. HARDWARE EFFICIENT MULTIPLYING DAC SYNAPSES

A commonly used model for artificial neural networks can be described by

$$\underline{y} = \underline{g}(\underline{s}), \qquad \underline{s} = \underline{\underline{w}} \begin{bmatrix} \underline{y} \\ \underline{x} \end{bmatrix} \equiv \underline{\underline{w}}\,\underline{z}\,,$$

where $\underline{y}$ is a vector of neuron activations, $\underline{g}$ is the activation function (applied by coordinates), $\underline{x}$ is the network inputs, $\underline{s}$ is called the net input vector and $\underline{\underline{w}}$ is the connection strength (or synapse) matrix. For a layered network many entries in the $\underline{\underline{w}}$ above is zero. VLSI implementations of ANNs usually reflect these equations, as being composed of a matrix vector multiplier followed by a non-linear activation function block. Often the multiplier and the activation function are placed on different chips to ensure cascadability and thus the ability of this chip set to implement arbitrary architectures [7]. The multiplier should then have current outputs and voltage inputs. It is an architecture like this we have in mind in the present text, though the considerations herein are not restricted to that. The most important subcircuit by far in these architectures is the store-multiply-accumulate synapses of the matrix-vector multiplier; we shall be concerned primarily with that in the present text. For complete chip set architectures refer to [8, 9].

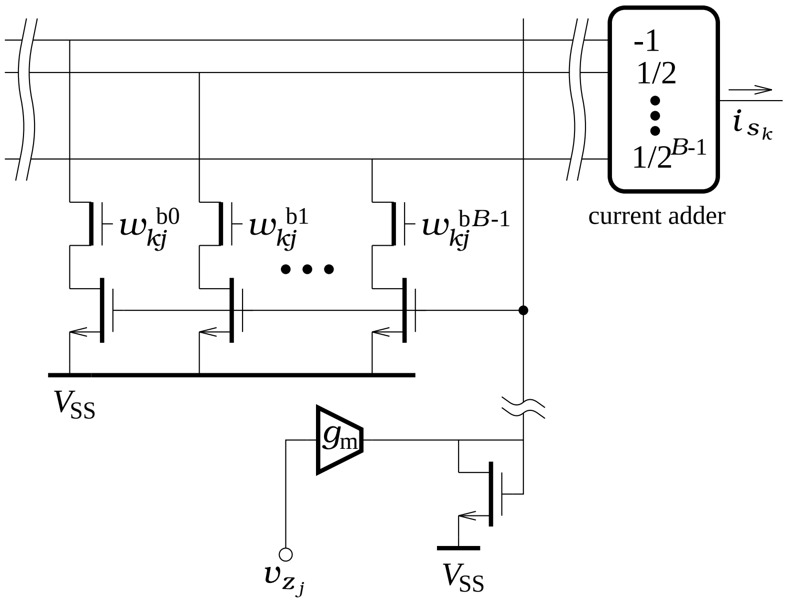

Figure 1. Basic hardware efficient analogue/digital hybrid synapse with digital storage.

The basic architecture for our hardware efficient synapse with digital weight storage (MDAC synapse) is seen in figure 1 [6]. The transconductor and diode coupled transistor transform the input voltage v_{z_j} to a voltage which, when applied to the gate of a transistor (current source) at a synapse site, causes a proportional drain current, $g_{\rm m} v_{z_j}$. This voltage is distributed to a column of synapses (or matrix-vector multiplier elements). The multiplication is performed by switching the current sources to or from the horizontal current accumulation lines (controlled by the locally stored synapse strength word) and subsequently doing a binary weighted sum of these partial products (using the row global current adder). The current output from a row of synapses is given by

$$i_{s_k} = -\sum_{j=1}^{M} w_{kj}^{\mathrm{b}0} g_{\mathrm{m}} v_{z_j} + \sum_{b=1}^{B-1} \frac{1}{2^b} \sum_{j=1}^{M} w_{kj}^{\mathrm{b}b} g_{\mathrm{m}} v_{z_j} = g_{\mathrm{m}} \sum_{j=1}^{M} w_{kj} v_{z_j} \; ,$$

where $w_{kj}^{\mathrm{b}b}$ is the bth bit of weight w_{kj} (stored as a two's compliment signed number), B is the number of bits in w_{kj} and M is the number of row inputs.

The input voltages obviously have to be non-negative using this architecture. This is unfortunate from a learning point of view, as ANNs are usually taught easier if the neuron activations are allowed to assume both positive and negative values [10]. Assuming now that we have access to the sign, $\mathrm{sgn}(v_{z_j})$†, and magnitude, $|v_{z_j}|$, of the inputs, we can apply a sort of *analogue two's compliment negation* of the weights if the input is negative. This requires only one extra current source at the synapse site and a couple of switch transistors, see figure 2, and thus does not compromise the hardware efficiency of our MDAC synapse.

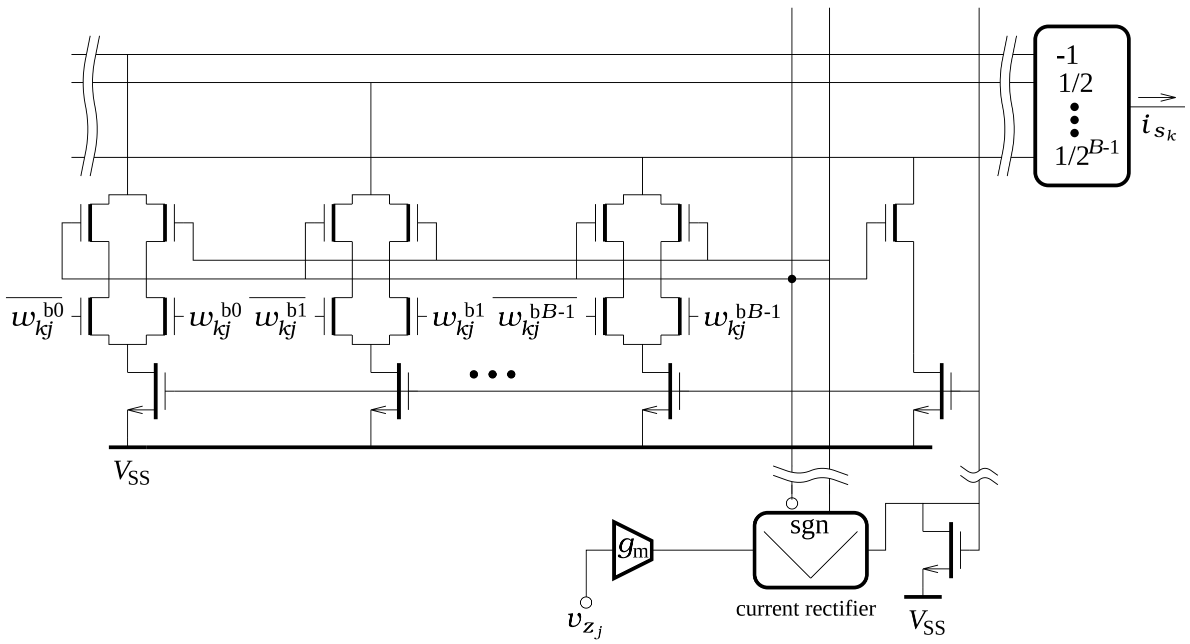

Figure 2. Synapse handling bipolar neuron activation inputs.

For $v_{z_j} \geq 0$ the synapse schematic is equivalent to that in figure 1. For $v_{z_j} < 0$

† We regard this as a binary value: $\mathrm{sgn}(v_{z_j}) = 1$ for $v_{z_j} \geq 0$ and $\mathrm{sgn}(v_{z_j}) = 0$ otherwise. This implies, for instance, that $\overline{\mathrm{sgn}(v_{z_j})} \equiv 1 - \mathrm{sgn}(v_{z_j})$.

the synapse $\{k, j\}$ contribution to the output current, is given by:

$$\begin{aligned} i_{s_{kj}} &= -\overline{w_{kj}^{\mathrm{b0}}} g_{\mathrm{m}} |v_{z_j}| + \sum_{b=1}^{B-1} \frac{1}{2^b} \overline{w_{kj}^{\mathrm{b0}}} g_{\mathrm{m}} |v_{z_j}| + \frac{1}{2^{B-1}} g_{\mathrm{m}} |v_{z_j}| \\ &= (1 - w_{kj}^{\mathrm{b0}}) g_{\mathrm{m}} v_{z_j} - \sum_{b=1}^{B-1} \frac{1}{2^b} (1 - w_{kj}^{\mathrm{b0}}) g_{\mathrm{m}} v_{z_j} - \frac{1}{2^{B-1}} g_{\mathrm{m}} v_{z_j} \\ &= -w_{kj}^{\mathrm{b0}} g_{\mathrm{m}} v_{z_j} + \sum_{b=1}^{B-1} \frac{1}{2^b} w_{kj}^{\mathrm{b0}} g_{\mathrm{m}} v_{z_j} = g_{\mathrm{m}} w_{kj} v_{z_j} , \end{aligned}$$

as desired.

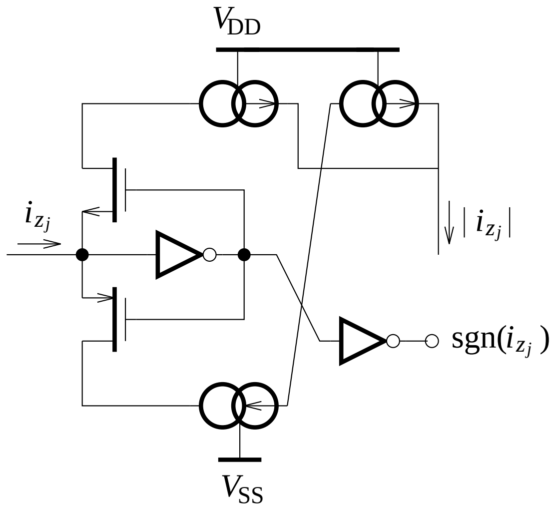

Figure 3. Schematic of accurate current rectifier/sign detector.

For the applicability of the bipolar input synapse circuit, it is crucial that an *accurate current rectifier/sign detector* can be constructed. We base this on a fast current sign detector found in [11]. It turns out that this circuit can be used as the inherently low offset core of a current rectifier. The principal schematic of the current rectifier/sign detector is seen in figure 3. It is possible for only one of the transistors to be turned on at any one time; thus a current flowing into the input terminal will inevitably be sunk by the p-channel transistor, whereas a current flowing out of the input terminal will be sourced by the n-channel transistor. Adding a couple of current mirrors gives the desired rectifier. The accuracy is determined by transistor matching in these current mirrors. The sign of the current can be detected at the inverter output. In our actual implementation small bias currents were added to the current mirror inputs for the sake of speed.

3. EXPERIMENTAL VALIDATION

A 4×4 elements matrix-vector multiplier embracing the principles above has been fabricated in a standard $1.5\,\mu$m, 5 V, double metal, digital CMOS process. The input transconductors were based on cross coupled differential pairs for linearity [12]. The output scaling current adder was based on scaling high-swing cascode current mirrors [13]; small bias currents were added to the mirrors for the sake of speed. Finally, standard six-transistor static RAM cells were used as connection strength memories. The 8 bit resolution synapse consumes an area of $335\,\mu\text{m} \times 123\,\mu\text{m}$. Approximately one third is occupied by the digital memory, one third by the current sources and switch transistors and one third is used for routing.

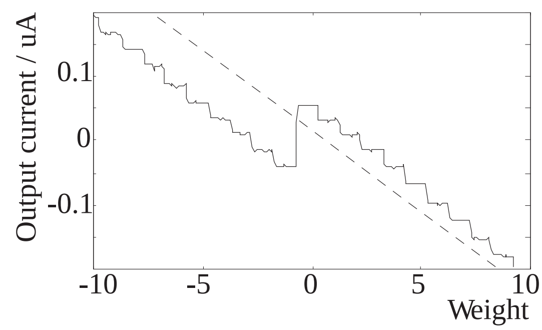

Figure 4. Measured synapse DAC characteristic; zero value transition.

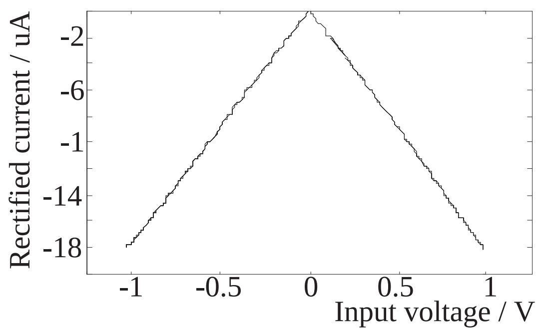

Figure 5. Measured characteristic for on-chip test current rectifier (scaled output, preceded by transconductor).

Figure 4 shows a close-up around the critical $-1(11111111_{\mathrm{b}}) \rightarrow 0(00000000_{\mathrm{b}})$ transition on the synapse DAC characteristic for constant v_{z_j} input. The large step ($\lesssim$ 4 LSB) is caused by mismatch in the scaling current adder and in the current sources. The resulting non-linearity is $D_w \lesssim 1.5\,\%$ of the synapse output current range, corresponding to 6 bit accuracy. The v_z to i_s non-linearity (for constant weight) is $D_z \lesssim 1.5\,\%$ and is primarily caused by the transconductor: the current rectifier and associated circuits results in only a small kink (below 0.4 %) in the transfer function. The transfer characteristic of an on-chip test rectifier can be seen in figure 5. The chip output offset current is $i_{s\mathrm{ofs}} \lesssim 0.44\,\mu$A or $\lesssim 13\,\%$ of the synapse output current range. These figures are compatible with typical neural network applications [14]. The input to output propagation delay is somewhat large, $t_{\mathrm{pd}\,zs} \lesssim 5\,\mu$s, and is caused by the large capacitive load on the diode coupled transistor in figure 2. For a scaled version of the chip, a buffer should be inserted.

4. INCORPORATING LEARNING HARDWARE

The above matrix-vector multiplier architecture is well suited for recall-mode systems. It can be trained using chip-in-the-loop training residing in a host computer that have stored the high precision representation of the weights which are needed during learning [15]. In adaptive stand-alone systems or when speed requirements demand utilization of the parallelism inherent in the learning algorithm we must consider on-chip learning. The first concern would be: how can we enhance the weight resolution sufficiently for the learning procedure?

We propose to combine the digital memory with a simple capacitive one for the accumulation of small weight changes; an "*analogue adjustment*". This approach is particularly attractive because the weight changes can be governed by simple analogue circuitry, thereby minimizing the hardware cost of the learning algorithm.

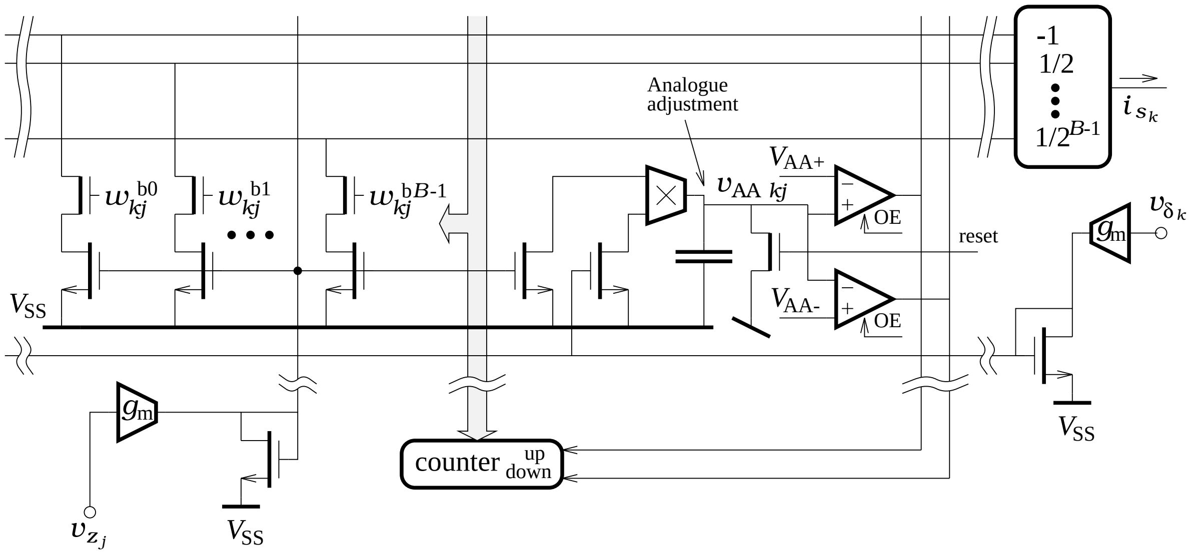

Figure 6. Learning algorithm controlled analogue adjustment and exhaustion detection.

A sample utilization of the analogue adjustment is shown in figure 6. For simplicity we have shown a synapse with a positive input requirement; hardware handling signed v_{z_j} and v_{δ_k} is included as above. The value of the analogue adjustment, $v_{\mathrm{AA}kj}$ is changed by the current input, current output multiplier which is a part of the the learning algorithm hardware. The accumulated weight change on the capacitor must ultimately be reflected in the digital register, of course: Two comparators are placed at the synapse site checking if the range of $v_{\mathrm{AA}kj}$ ($V_{\mathrm{AA}-} < v_{\mathrm{AA}kj} < V_{\mathrm{AA}+}$) is exceeded. Sequentially, within each column the comparator outputs are enabled (by the signal OE), and if the analogue adjustment is outside the range $[V_{\mathrm{AA}-}, V_{\mathrm{AA}+}]$ the corresponding digital weight is increased/decreased accordingly by the up/down counter shown in figure 6. Also, the analogue adjustment is reset to zero. The up/down counter and associated logic is shared by all synapses in a column. One could also place a voltage buffer at the synapse site and let a column share the comparators. The necessary resetting of the analogue adjustment will introduce noise in the learning scheme. This will most probably degrade learning but not prohibit it. One could be tempted to place a multiplier that accumulated $v_{z_j} \cdot v_{\mathrm{AA}kj}$ to the "$1/2^{B-1}$" output line such that the chip output would reflect the actual weight. However, improved learning results if the errors are computed on the basis of the truncated weights rather than the ones with "artificially" high resolution [16].

4.1. The learning scheme

The learning algorithm for our ANN could be Hebbian learning [17] or non-linear backpropagation learning [18, 9], for instance (for complete architectures see these references).

Using Hebbian learning, the architecture in figure 6 is sufficient. v_{δ_k} must then be the post synaptic neuron (k) activation.

Using (non-linear) back-propagation, the circuit must be slightly expanded: This algorithm requires a column, j, of synapses to compute the back-propagated error $g_\mathrm{m}\sum_k w_{kj}v_{\delta_k}$, where v_{δ_k} is the "deltas" computed at the neuron sites. If discrete time is acceptable the neuron k activation can be sampled; in this case we can use the very same multiplier for the $w_{kj}\cdot v_{\delta_k}$ product as we used to compute the $w_{kj}\cdot v_{z_j}$ product [9]. We need only $2B+4$ additional switches for redirecting the multiplier input and outputs. We would also need a scaling current adder for each column, of course. A rough estimate of the synapse hardware overhead for back-propagation is 50 %. (In our particular implementation of the MDAC synapse, there is plenty of room for the necessary B extra signal lines per column; if this is was not the case we could use the register access lines for the reverse mode synapse output.)

5. CONCLUSIONS

In this paper we have addressed the issue of weight storage in (analogue) neural networks. Weight discretization is the most reliable way of eliminating leakage problems associated with, for instance, capacitive analogue storage. Using static digital storage is a convenient solution; it is, however, quite area consuming. We showed how to implement a matrix-vector multiplier with digital weight storage in a hardware efficient way. This was based on a previously reported current scaling/summing technique and a novel technique for manipulating bipolar inputs: we extracted the sign information via an accurate current rectifier/sign detector and used an analogue two's compliment negating scheme at the synapse sites. The total synapse transistor count (B bits) is $B+1$ identical current sources and $4B+1$ switch transistors; plus the digital memory. Measurements on a test chip showed non-linearities below 1.5 % of the synapse output current range.

Using digital weight storage is usually a problem during learning because the required weight resolution in this phase is much higher than what is conveniently implemented. We have proposed to add to the digital weight an analogue storage (a capacitor) which would accumulate weight changes during learning. This is a very attractive approach as the weight change then can be computed by simple analogue hardware; i.e. the hardware overhead for the learning algorithm would be very small. The implementation of Hebbian learning and back-propagation learning was outlined showing a very modest area overhead indeed.

Combining the virtues of both digital and analogue storage, using our area efficient approach to digital weight storage and our analogue adjustment temporary resolution enhancement, we believe is a very capable architecture.

REFERENCES

[1] E. Vittoz, H. Oguey, M. A. Maher, O. Nys, E. Dukstra and M. Chevroulet, "Analog Storage of Adjustable Synaptic Weights," in *VLSI Design of Neural Networks*, Ulrich Ramacher and Ulrich Rückert, Eds., Norwell: Kluwer Academic Publishers, 1991, pp. 48–63.

[2] Phillip H. W. Leong and Marwan A. Jabri, "Kakadu — A Low Power Analogue Neural Network," in *Proc. 3'rd International Conference on Microelectronics for Neural Networks*, pp. 207–216, 1993.

[3] Hernan A. Castro, Simon M. Tam and Mark A. Holler, "Implementation and Per-

formance on an Analog Nonvolatiole Neural Network," *Analog Integrated Circuits and Signal Processing*, vol. 4, no. -, pp. 97–113, 1993.

[4] Christian Schneider and Howard Card, "Analog CMOS Synaptic Learning Circuits Adapted from Invertebrate Biology," *IEEE Transactions on Circuits and Systems*, vol. CAS-38, no. 12, pp. 1430–1438, 1991.

[5] Lionel Tarassenko, Jon Tombs and Graham Cairns, "On-chip Learning with Analogue VLSI Neural Networks," *International Journal of Neural Systems*, vol. 4, no. 4, pp. 419–426, 1993.

[6] Erik Bruun, John A. Lansner and Torsten Lehmann, "Analog VLSI Architectures for Computational Neural Networks," in *Proc. 10'th NORCHIP Seminar*, pp. 59–68, 1992.

[7] Silvio Eberhardt, Tuan Duong and Anil Thakoor, "Design of Parallel Hardware Neural Network Systems from Custom Analog VLSI 'Building Block' Chips," in *Proc. IEEE International Joint Conference on Neural Networks*, pp. II-183–II-190, 1989.

[8] John A. Lansner and Torsten Lehmann, "An Analog CMOS Chip Set for Neural Networks with Arbitrary Topologies," *IEEE Transaction on Neural Networks*, vol. 4, no. 3, pp. 441–444, May 1993.

[9] Torsten Lehmann, "A Hardware Efficient Cascadable Chip Set for ANN's with On-chip Back-propagation," *International Journal of Neural Systems*, vol. 4, no. 4, pp. 351–358, 1993.

[10] Simon Haykin, *Neural Networks: A Comprehensive Foundation*, New York: Macmillan Collage Publishing Company, Inc., 1994.

[11] R. Domínguez-Castro, A. Rodríguez-Vázquez, F. Medeiro and J. L. Huertas, "High Resolution CMOS Current Comparators," in *Proc. 18'th European Solid State Circuits Conference*, pp. 242–245, 1992.

[12] Scott T. Dupuie and Mohammed Ismail, "High Frequency CMOS Transconductors," in *Analogue IC design: the current-mode approach*, C. Toumazou, F. J. Lidgey & D. G. Haigh, Eds., IEE Circuits and Systems (2) Series, London: Peter Peregrinus Ltd., 1990, pp. 181–238.

[13] P. J. Crawley and G. W. Roberts, "High-Swing MOS Current Mirror with Arbitrarily High Output Resistance," *Electronics Letters*, vol. 28, no. 4, pp. 361–363, 1992.

[14] Antonio J. Montalvo, Paul W. Hollis and John J. Paulos, "On-Chip Learning in the Analog Domain with Limited Precision Circuits," in *Proc. International Symposium on Circuits and Systems*, pp. I-196–I-201, 1992.

[15] Hollis, Harper and Paulos, "The Effects of Precision Constraints in a Backpropagation Learning Network," *Neural Computation*, vol. 2, no. 3, pp. 363–373, 1990.

[16] John A. Lansner, "Analogue VLSI Implementation of Artificial Neural Networks," Ph.D.-thesis, Electronics Institute, Technical University of Denmark, 1994.

[17] Takeshi Shima, Tomohisa Kimura, Yukio Kamatani, Tetsuro Itakura, Yasuhiko Fujita and Tetsuya Iida, "Neuro Chips with On-chip Back-propagation and/or Hebbian Learning," *IEEE Journal of Solid-State Circuits*, vol. SC-27, no. 12, pp. 1868–1876, 1992.

[18] John Hertz, Anders Krogh, Benny Lautrup and Torsten Lehmann, "Non-linear Back-propagation: Doing Back-propagation without Derivatives of the Activation Function," Neuroprose preprint, Niels Bohr Institute, Copenhagen, 1994.