Implementation of a Muli-Layer Perceptron including Back Propagation training Algorithm

Petin, Y.A.

Award date:
1993

Implementation of a Multi-Layer Perceptron including Back Propagation Training Algorithm

Master thesis of Y.A. Pétin

Supervisor	: Prof.dr.ir. W.M.G. van Bokhoven
Coach	: Dr.ir. J.A. Hegt
Period	: Dec. 1 1992 - Aug. 23 1993
At	: Electronic circuit design group (EEB)
	: Faculty of electrical engineering
	: Eindhoven University of Technology

Abstract

This work concerns the implementation of the Back Propagation Algorithm (BPA) using a pulse stream technique.

Among the existing pulse stream techniques, the Coherent Pulse Width Modulation (CPWM) seems to be the best option. Using CPWM, the forward path has already been designed with lower chip area and power consumption compared to other analogue realizations.

Implementation of the BPA on the chip is desired to speed up its training. The existing analogue implementations of the BPA require a large chip area and have a high power consumption. One should expect that the CPWM technique could bring the same good results for the implementation of the backward path as for the forward path.

In this work, the structure of the MLP backward path has been investigated. A synapse unit, a neuron unit for the hidden layers and an output neuron unit have been studied. For each of them, circuits needed for the backward path are proposed with their characteristics. Simulations of the complete backward path are presented.

It appears that the BPA suffers from non idealities introduced by the hardware. The dynamic multipliers based on the CPWM technique introduce offsets which affect the performance of the training algorithm. Also a number of problems with respect to the interface between the synapse chip and the neuron chip still have to be solved.

Contents

Abstract i

Contents ii

Introduction 1

Chapter 1 Theory of the Multilayer Perceptron 2
1.1 Introduction 2
1.2 Structure of a MLP 3
1.3 Backpropagation Training Algorithm 4

Chapter 2 Electronic Implementation 8
2.1 Internal organisation of the synapse and the neuron chip 8
2.2 Consideration of the hardware implementation of the BPA 14
2.3 Cascadability 14

Chapter 3 Information Representation 17
3.1 Review of the different approaches and techniques 17
3.2 Pulse Modulation Arithmetic 19
3.3 Structure of the synapse and the Neuron with CPWM signals 20

Chapter 4 The Synapse Unit 22
4.1 Two-quadrant Multiplier S1 23
4.2 Four-quadrant Multiplier S2 25
4.2.1 Approach #1 25
4.2.2 Approach #2 27

Chapter 5 The Neuron Unit (Hidden Layer) 30
5.1 Sigmoid Unit 30
5.2 Feedback Path 33
5.2.1 Derivative of Sigmoid Function 33
5.2.1.1 Approach #1 33
5.2.1.2 Approach #2 36
5.2.2 Two-quadrant Multiplier N2 38
5.2.3 Two-quadrant Multiplier N1 44

Chapter 6 The Output Neuron Unit 46
6.1 Error Generation Circuit 46
6.2 Converters 51
6.2.1 Wave-Driven Converter 51
6.2.2 Capacitor-Charging Converter 52

Chapter 7 Backward Path and Simulation Results 54
7.1 Normal Neuron Unit with one Synapse Unit 54
7.2 Output Neuron Unit 57

Chapter 8 Conclusions and Recommendations 61

Bibliography 63

Introduction

Work on Artificial Neural Net (ANN) began more than 40 years ago, with the works of Mc Culloch and Pitts, Hebb, Rosenblatt, Widrow, and others.

More recent works of Hopfield, Rumelhart and Mc Clelland, Seijnowski, Feldena, Grossberg, and others have led to the resurgence of the field.
This new interest is due to the development of new net topologies and algorithms, as well as by a growing fascination with the functioning of the human brain. Recent interest has also been raised by the realization that human-like performance in the areas of speech and image recognition require enormous amounts of processing.

Since software simulations of neural networks are less attractive due to the limits of conventional machines, researchers have been working on implementations that are more adapted to inherent neural properties.

For hardware implementations, electronic and optical approaches have been proposed. Since the optical approaches suffer from the absence of flexible optical devices, electronic approaches have been more developed. In fact, several VLSI implementations can be found in literature, which use a number of different working principles, implementation technologies and interconnection architectures. These hardware implementations may use either digital, or analogue techniques or a mixture of the two. Among the last technique, the so-called "pulse-stream technique" has been steadily evolved and improved by several authors (Murray, Hinai and Meodor) since its inception in 1986 . These pulse-stream techniques have been inspired by biology and borne out of a desire to perform analogue computation with chips, produced by means of a digital fabrication process.

The most appealing feature of ANNs is their ability to learn. While it is trivial to implement learning into digital circuits, only a number of analogue implementations (not based on the pulse stream techniques), [11], [16], integrate learning into their systems. Without any doubt on chip learning is desirable for large ANNs, but for analogue ANNs, on chip learning involves an inevitable complexity of the hardware.

The different pulse-stream techniques have been analyzed in the work of T. Claasen-Vujčić [6], in which two approaches for the feedforward path have been proposed, using Coherent Pulse Width Modulation. She concludes that using this technique, the feedforward path can be implemented with less circuits compared to others pulse-stream techniques. One can assume that the implementation of the back propagation training algorithm can also be realized using the CPWM technique.

In chapter I, the background of the theory of the multi-layer perceptron is presented, while chapter II deals with the electronic implementation of the MLP and the effects of the implementation on the backpropagation learning algorithm. In chapter III the existing hardware implementation techniques are briefly reviewed and the possibilities to encode the information are also resumed. Chapter IV deals with the Synapse Unit. Since the normal Neuron Unit is different from the Output Neuron Unit, they will be presented and analyzed in details in chapter V and VI respectively. Finally in chapter VII a complete design for the backward path is proposed and PSPICE simulation results are presented.

1 Theory of the Multilayer Perceptron

This chapter presents the background of the theory of the Multi-Layer Perceptron (MLP) and of the backpropagation training algorithm necessary to the general comprehension of this work. In fact, much more detailed information concerning this theory can be found in literature [25] and [29].

1.1 Introduction

Neural Networks consist of processing units called neurons. These neurons can be connected to each other in different ways, by weighted connections. The weighting of these connections is performed by the synapses, which multiply the propagating signal with the weight's value (W). The value of these weights can be positive or negative. Weights which are adjustable are set to their definitive values after 'training'. The next figure shows a model of N synapses connected to a neuron.

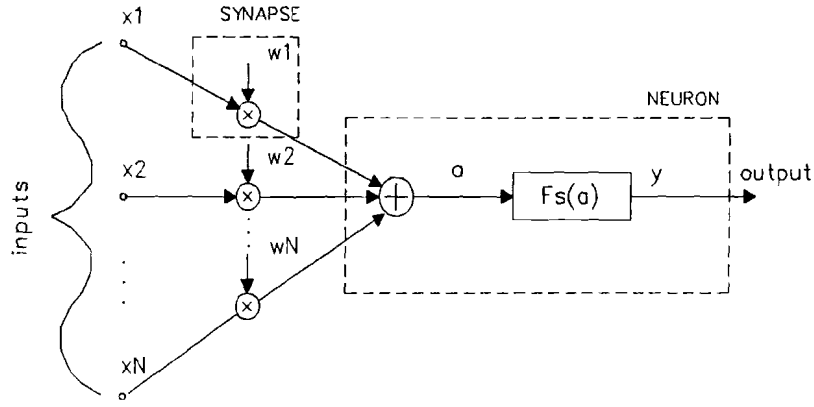


Fig. 1.1.1 Model of a neuron

The neuron adds the incoming signals from the synapses, after which the sum is fed to the a function with saturating characteristic (Fs()).
Three kinds of non linearities [17],[25], step threshold, ramp threshold and sigmoid are usually used, depending on the application of the neural networks.
From these three kinds of non linearities, a sigmoid or a sigmoid-like characteristic is the one which is most frequently implemented in the analogue multilayer perceptron.
The sigmoid can be expressed as follows:

$$Fs\ (x) = \frac{K}{1 + e^{-\tau\,(x\,-\,\alpha)}} \qquad (1.1.1)$$

where τ represents the temperature factor (steepness) and α is an offset.
A plot of a sigmoid function corresponding to the mathematical expression (1.1.1) is presented in fig.1.1.2.

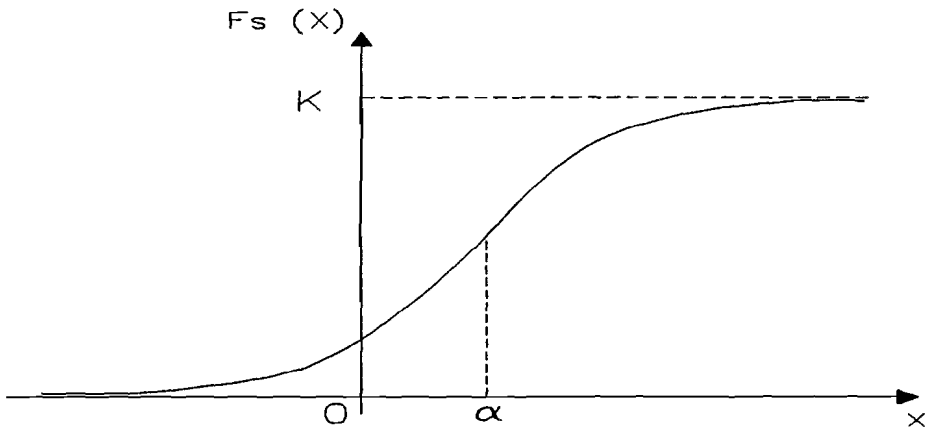


Fig. 1.1.2 Sigmoid function

The derivative of the sigmoid, which is needed in the backpropagation algorithm (see paragraph 1.3), is easy to calculate and can be expressed in the function itself:

$$F's\ (x) = \frac{\partial\ (Fs\ (x))}{\partial x} = \frac{\tau\ e^{-\tau\ (x\ -\ \alpha)}}{1\ +\ e^{-\tau\ (x\ -\ \alpha)}} = \tau\ Fs\ (x)\ (1\ -\ Fs\ (x)) \qquad (1.1.2)$$

Figure 1.1.3 presents a plot of the derivative of the sigmoid function as expressed above in (1.1.2).

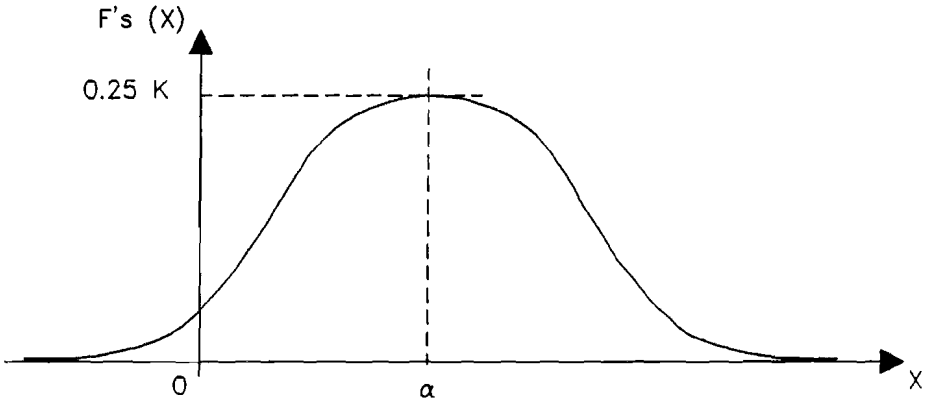


Fig. 1.1.3 Derivative of the sigmoid function

1.2 Structure of a MLP

Using the so-called neuron units previously presented, different neural network structures can be build. One of them is the multi-layer perceptron, in which the neurons are organized in layers.
The next figure represents the structure of a multi-layer perceptron with N inputs , M_L outputs

and L layers.

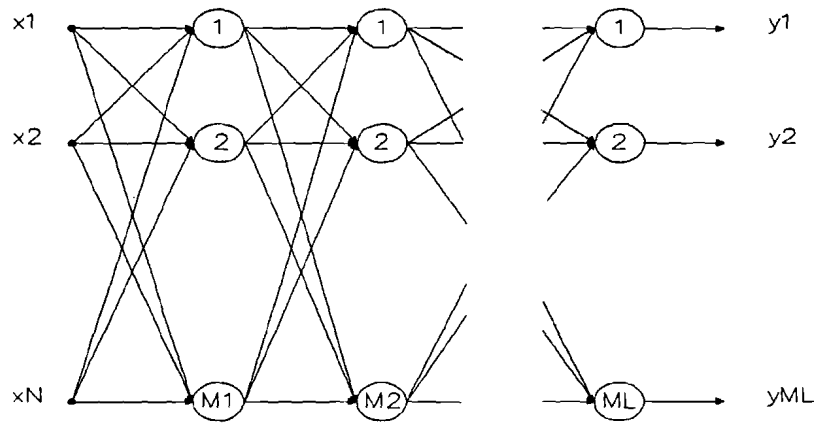


Fig. 1.2.1 Structure of a MLP

All the layers between the first layer (inputs) and the last layer (outputs) are called hidden layers. The number of neurons can be different for each layer.
During the normal use of the neural networks, the signals propagate from the first layer (inputs) to the last layer (output) through the so-called feedforward path. If the learning algorithm is implemented on chip, then during the learning phase, a feedback path is present.
The number of layers and the number of neurons per layers are two parameters depending on the kind of functions to be approximated. All the continuous functions can be approximated with a two layers net [29].

1.3 Back Propagation Algorithm

In order to be able to approximate functions the MLP has to be trained. The first training algorithm for Multilayer Perceptrons was the Backpropagation Algorithm which has a few drawbacks. Today, variants of the Backpropagation or faster algorithms are being proposed [21], but most of them are difficult to implement in hardware.
The Backpropagation Learning Algorithm is an iterative gradient algorithm designed to minimised the mean squared difference between the actual output of a MLP and the desired pattern.
All the variables used in the following backpropagation formulas are listed below:

N : number of inputs of a layer

M : number of neurons in a layer

x_i : input i of a layer

w_{ji} : weight from input i to neuron j

a_j : activity of neuron j

y_j : output from neuron j in a layer

d_j : desired output from neuron j

δ_j^L : error term for neuron j in output layer L

δ_j^l : error from neuron in layer l

η : learning rate

$S(.)$: sigmoid threshold function of neuron

$S'(.)$: derivative of the sigmoid threshold function

Next, the back propagation algorithm is described step by step:

Step 1. Initialize weights

All the weights of the net are set to small random values

Step 2. Present inputs and desired outputs

Present an input vector $x_0, x_1, \ldots x_N$, and specify the desired outputs $d_0, d_1, \ldots d_M$

Step 3. Calculate actual outputs

Starting with the first layer, the output of each layer is calculated until the actual output of the net is obtained

The output of the neuron j of layer l is:

$$y_j = S(a_j) = S\left(\sum_{i=1}^{N} w_{ji} x_i\right) \text{ , with } 1 \leq j \leq M \qquad (1.3.1)$$

Step 4 . Updating the weights

Once the error is calculated as the difference between the actual output and the desired output, it is propagated back through the neural net from the last layer to

the first. From this the name " backpropagation training algorithm " stems.

error term for the last layer: $$\delta_j^L = S'(a_j)(d_j - y_j) \quad (1.3.2)$$

error term for all the remaining layers: $$\delta_j^l = S'(a_j) \sum_{k=1}^{M} \delta_k^{l+1} w_{jk}^{l+1} \quad (1.3.3)$$

weight correction: $$w_{ji}^l(t+1) = w_{ji}^l(t) + \eta \delta_j^l x_i \quad (1.3.4)$$

where t is the iteration number

Step 5 . Repetition of the sequence

Steps 2 to 4 are repeated until the following condition is achieved,

$$\sum_{j=1}^{M} (d_j - y_j)^2 \leq \varepsilon \quad \text{where} \quad \varepsilon \quad \text{is the maximum error allowed.} \quad (1.3.5)$$

The implementation of a nonlinear function with a variable offset α can improve the performances of the net. This way the sigmoid function can be shifted over the total input range (see fig. 1.3.1). The offset can be realized by adding an extra input common for all the neurons and a bias weight per neuron. The input has a value equal to one while the bias weights are adjustable with the backpropagation algorithm.

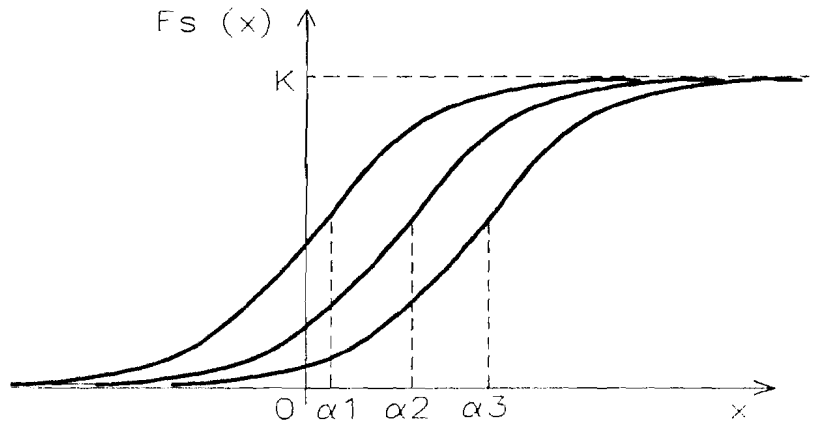


Fig. 1.3.1 Sigmoid function shiftable

In spite of its success and popularity, the implementation of the standard BP algorithm suffers from many well known caveats [22]:

- it is heavily influenced by the values of the learning rate parameter η
- it may converge very slowly
- it is heavily dependent on the size of the networks
- it is sensitive to some non idealities in hardware implementations
- it may require a high resolution for the weights

However, the on chip implementation is required for large neural nets with a high number of interconnected neurons, in which it will speed up the training velocity with respect to software implementations.

2 Electronic Implementation

In this second chapter, after a brief review of the requirements considering electronic implementation of MLP, the synapse and neuron unit, which make up the synapse and neuron chip respectively, are presented in detail. The electronic aspect of cascadability is discussed and some considerations concerning the hardware implementation of the backpropagation training algorithm are presented.

The following aspects are very important in view of any electronic implementation of a multi-layer-perceptron. The number of neurons and synapses which can be accommodated per chip depends on the following aspects:

- power dissipation (lower by using pulse-stream techniques)
- synapse and neuron implementation areas
- number of pins per neuron

The performance of the MLP depends on the following aspects:

- response time (this aspect plays a key role in the maximum speed of the total MLP)
- range of inputs, outputs and weights values
- flexibility and cascadability
- susceptibility to parameters variations and noise

2.1 Internal Organization of the Synapse and Neuron Chip

For the clarity of this report and for practical reasons, the mathematical terms used in paragraph 1.3 as well as the formulas of the training algorithm are translated in terms of electrical signals.

Forward Propagation Input : $FPI_i = x_i$ (2.1.1)

Forward Propagation Output : $FPO_j = y_j$ (2.1.2)

Backward Propagation Input : $BPI_j = d_j - y_j$, for the last layer (2.1.3)

$$BPI_j^l = \sum_{k=1}^{M} (\delta_k^{l+1} w_{jk}^{l+1}) \text{ , for the others layers} \quad (2.1.4)$$

Backward Propagation Output : $$BPO_i^l = \sum_{j=1}^{M} (\delta_j^l w_{ji}^l) \quad (2.1.5)$$

$$FPO_j = S(A_j) \tag{2.1.6}$$

$$A_j = \sum_{i=1}^{N} (w_{ji} \, FPI_i) \tag{2.1.7}$$

$$BPO_i = \sum_{j=1}^{M} (S'(A_j) \, w_{ji} \, FPI_i) \tag{2.1.8}$$

$$\Delta w_{ji} = \eta \, S'(A_j) \, BPI_j \, FPI_i \tag{2.1.9}$$

$$BPIS_j = S'(A_j) \, BPI_j \tag{2.1.10}$$

Depending on their future applications as well as the manner of training and weight storage, the electronic implementations of the MLP may differ. The idea developed in the previous works [6], [28] which try to obtain the most flexible structure as possible, is also maintained for the electronic implementation of the BPA. In order to be able to increase the number of neurons per layer, by paralleling layers blocks and without changing the number of synapses, one layer has to be split in a synapse and neuron chip [15]. The synapse and neuron part are shown in the next figure.

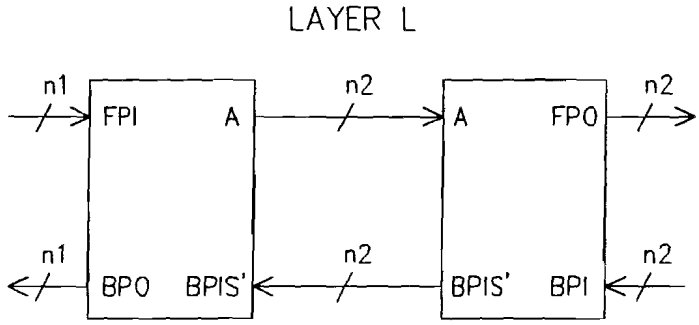


Fig. 2.1.1 Synapse and neuron part in layer L

Not only for the feedforward path but also for the feedbackward path, distributed signals are represented by voltages (see in fig. 2.1.1 FPI, FPO and BPIS'). Also for both paths, summed signals are represented by currents (see in fig. 2.1.1 A, FPI and BPO).
By cascading or paralleling synapse and neuron chips, it is possible to expand the number of outputs (see fig. 2.1.2.a) or inputs (see fig. 2.1.2.b) of an ANN.

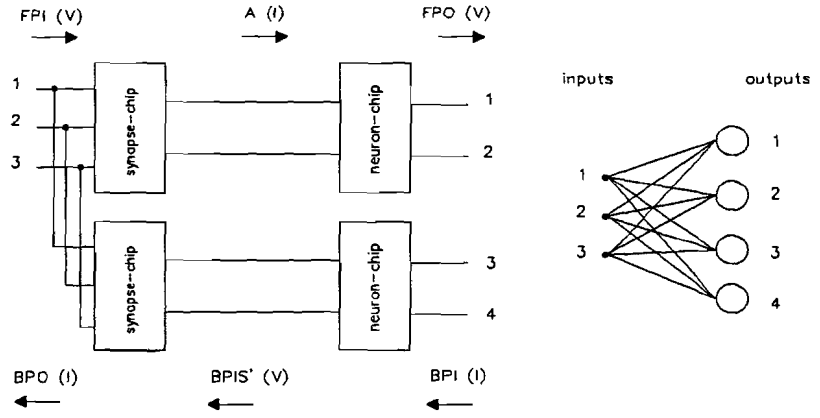


Fig. 2.1.2.a Expanding of the outputs

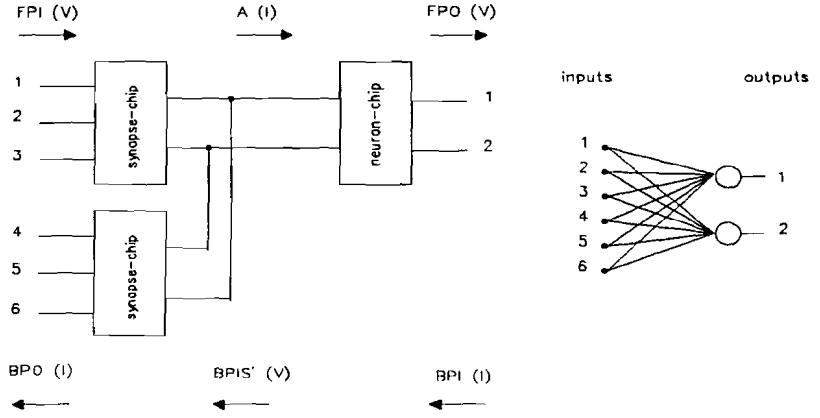


Fig. 2.1.2.b Expanding of the inputs

The two next figures (fig. 2.1.3 and fig. 2.1.4) show the global structure of the synapse and neuron unit; each of them is followed by a list of the operations and the range of the signals present in it. Subscripts i and j refer to the i^{th} input and j^{th} neuron. The letter V refers to voltage signals and the letter I to current signals. The terms positive and negative refer either to the direction in the case of the currents or to the values with respect to a reference voltage (Vref) in the case of the voltage.

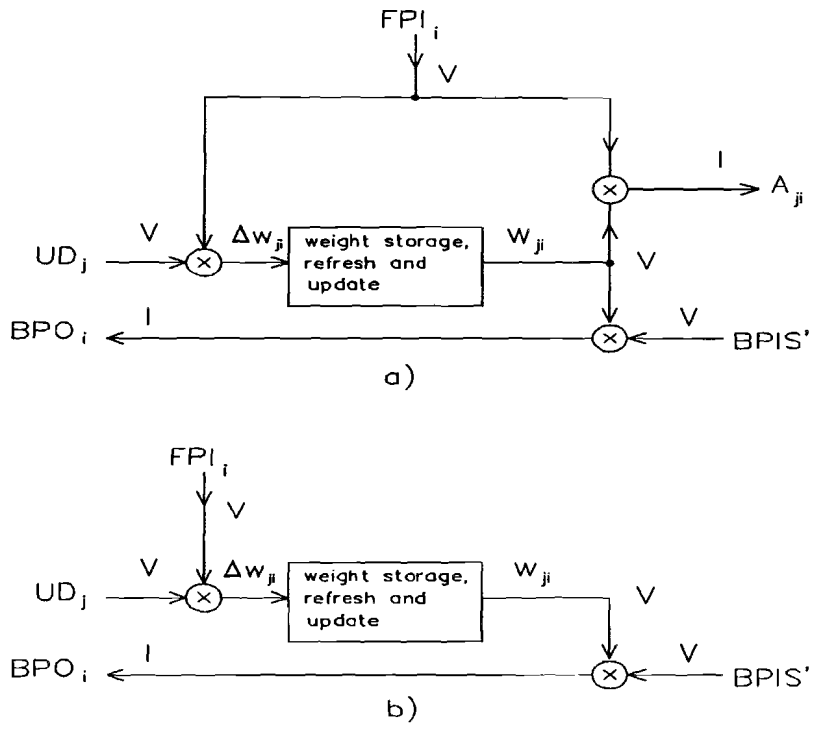


Fig. 2.1.3 a)Synapse unit, b)backward path

FPI_i	= positive signal
w_{ji}	= positive or negative signal
$A_{ji} = w_{ji}\ FPI_i$	= two-quadrant multiplication
UD_j	= positive or negative signal
$\Delta w_{ji} = UD_j\ FPI_i$	= two-quadrant multiplication
$BPIS'$	= positive or negative signal
$BPO_i = BPIS_j\ w_{ji}$	= four-quadrant multiplication

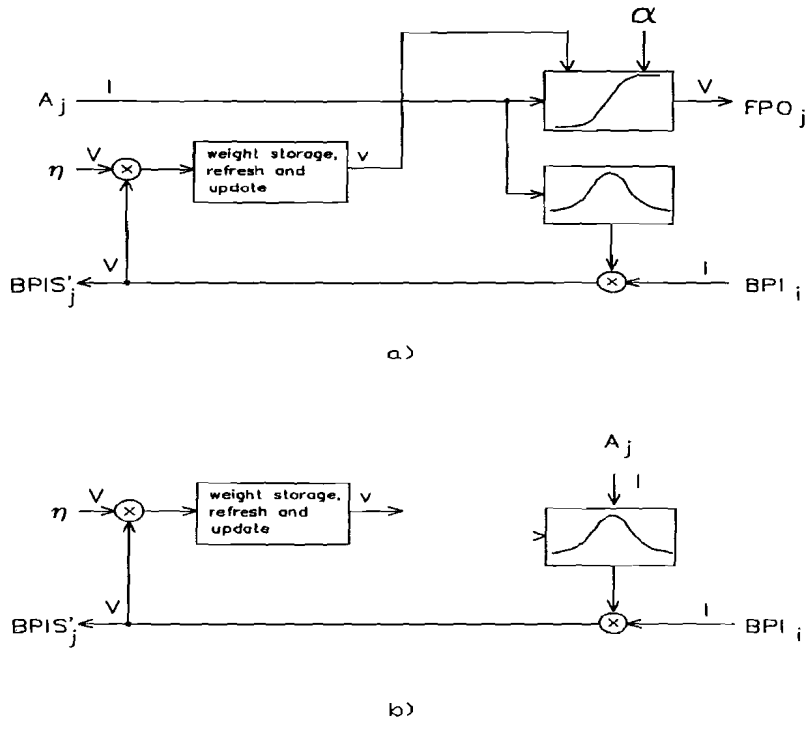


Fig. 2.1.4 a)Neuron unit, b)backward path

$A_j = \sum_{i=1}^{N} A_{ji}$ = positive or negative signal

$FPO_j = S(A_j)$ = positive or negative signal

$BPI_i = \sum_{j=1}^{M} FPO_{ji}$ = positive or negative signal

$BPIS_j = S'(A_j) \quad FPI_j$ = two-quadrant multiplication

$UD_j = \eta \; BPIS_j$ = two-quadrant multiplication

The synapse and neuron units described above are placed in a synapse and neuron chip respectively. The synapses are organized in array, see fig. 2.1.5.a, while the neurons are placed in layer, see fig. 2.1.5.b.

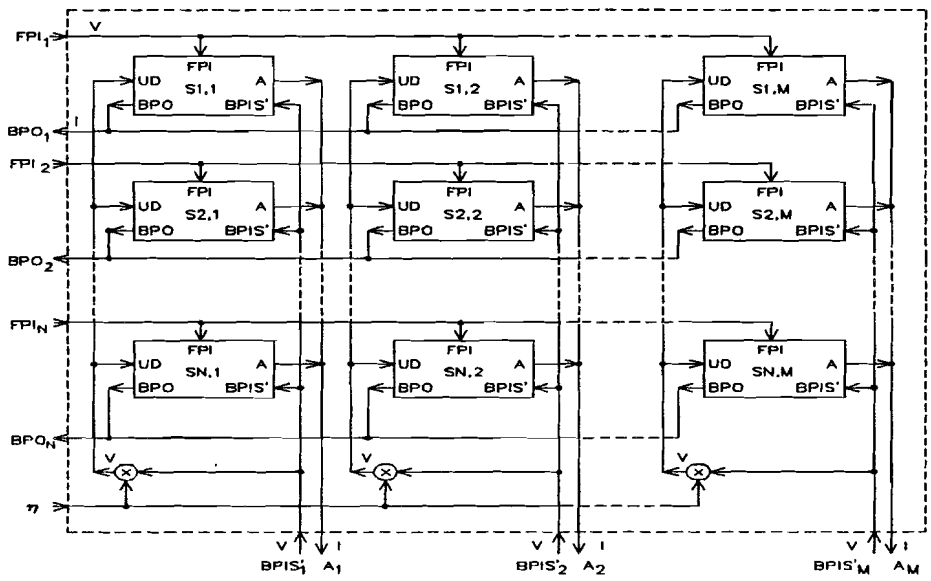


Fig. 2.1.5.a Synapse chip

The internal signal UD_j is calculated for each column of synapses which are connected to the same neuron in the next layer

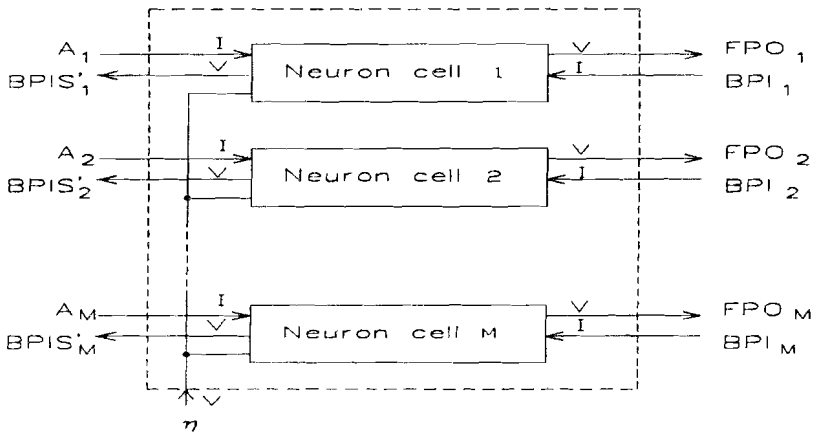


Fig. 2.1.5.b Neuron chip

2.2 Considerations of the hardware implementation of the BPA

While it is easy to incorporate learning into digital implementations, only a number of analogue implementations [11],[16],[17], integrate learning into their system.
In fact, in numeric implementations, the fine resolution is achieved using a very high number of digits, which also produces also very high precision, with relatively low additional cost.
It can be assumed that the rareness of analogue implementations of the BPA is due to the high accuracy and resolution demanded by this training algorithm. Analogue ANN circuits, generally suffer from inherent non-idealities, which are:

- non linear analogue multiplication and addition
- non ideal activation (threshold) function of neurons
- decay of weights, weight update and backpropagation error due to charge leakage when capacitors are being used for data storage.

This rareness of analogue implementations of the BPA is also due to a number of constraints compared with the off chip learning implementations which are slower, yet more flexible and accurate:

- resolution of mathematical calculation
- methods for modification and weight storage
- bounded weights with limited resolution (the resolution required depends on the desired output RMS error)

All these points lead to a limited calculation precision, which affects the quality and speed of training.
In literature [2], the influence of the cross talk effect is also discussed. The author concludes that the cross talk effects affect the learning speed, but not the performance of the net.
Use of continuous variables is essential in neural computation, since the convergence of learning depends on the ability to make very small changes on some variables.
However, the key question of how much precision is actually needed is still unsolved. The required precision will not be the same for a net used for the solution of a classifier problem as for the approximation of a sinus function.

2.3 Cascadibility

In theory, a fully flexible MLP topology is possible. In fact a flexible structure means that a layer should work properly with one input as well as with a large number of inputs. However, due to implementation dependant constraints, the maximum number of inputs is limited. As defined in paragraph 2.1, the output of the synapses are weighted currents. This implies that the resulting current for one input is different than for a hundred inputs. Assuming that the sigmoid function saturates for the input range corresponding one input, the region within the sigmoid is not saturated will represent less than 1% of the total input range in case of hundred inputs.
Therefore some authors [6],[20] and [28] have proposed to apply an automatic normalization to the sum.

The idea has also been developed in the work of T. Claasen-Vujčić and implies that the resulting sum always remains within a certain range and in fact results in a linear scaling (1/ N). For more detailed information about the techniques used, see paragraph 2.1.2 in [6].
After simulations realized with the neural net simulator developed in the same group, it can be assumed that it would be better to use the scaling function $1/\sqrt{N}$. Such a scaling function seems difficult to implement. But the problem can be overcome by implementing only one capacitor per neuron and adjusting the steepness of the sigmoid function with a factor $1/\sqrt{N}$.
In the feedback path, the output currents from the synapses are also summed. One could expect that this sum should be also scaled according to $1/\sqrt{N}$. It can be assumed that the error signal in the feedback path is automatically scaled by tuning the steepness of the sigmoid function since the steepness factor is also present in the derivative formula of the sigmoid function (1.1.2).

At the interface of the synapse chip and a neuron chip there will be a parasitic capacitor. This parasistic capacitor is in parallel with the capacitor implemented in the synapse. It will affect the value of the resulting capacitor voltage. This problem exists as well as for the feedforward path as for the feedback path. A way to resolve it can be to use the circuit shown in fig. 2.3.1.

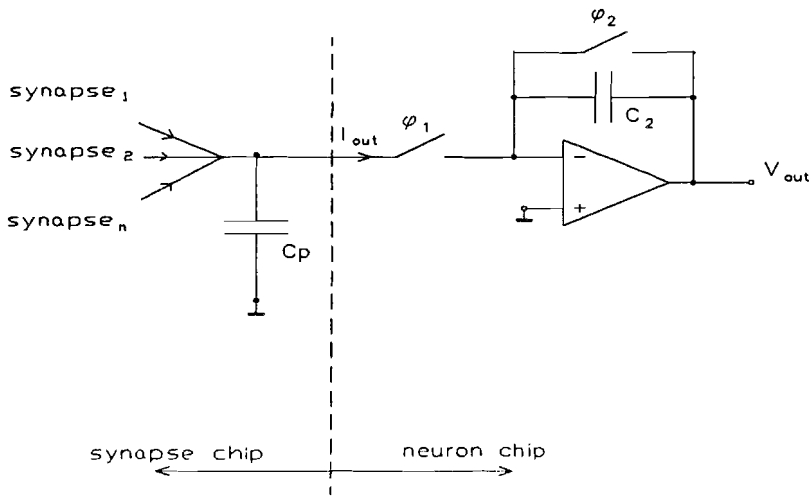


Fig. 2.3.1 Input satge buffer of the neuron unit

The input range of the sigmoid is limited (0 - 3V). Due to the statical distribution of the synaptic weights the chance that all the outputs of the synapses are maximum is very low. But since the capacitor voltage boundaries are those of the power supply (0 - 5V), the capacitor voltage has to be limited to 3V. This may be done by using an integrator clipping at 3V. However, this approach implies that two large sized capacitors (forward and backward path) have to be integrated per neuron unit.

An another possible approach should be to use the auto-scaling function proposed in [6]. In that case a multiplication by $\sqrt{N}$ (N = number of synapses connected) has to be performed in the neuron unit. The appropriate analog multiplier should be implemented just before the sigmoid function. The problem of the connection of the sysnapses to the neuron unit is a complex problem because scaling has to be carried out, the influence of the parasitic capacitor has to be reduced and the size

of the implementable capacitor is relatively small. However this problem has to be solved before any implementation of the feedforward path can be done.
In this work the first approach is considered. However the problem of the size of the capacitor and the way to limit its voltage have not been investigated.

3 Information Representation

3.1 Review of the different techniques and circuits

The basic functions of computation, storage and communication usually present in Neural Networks can use different representations of information, which are presented and detailed below.

Time and Amplitude continuous:
Information is carried by voltage (or current) levels; there is no time sampling or quantization, both for processing and communication. These are the fully analogue circuits.

Discrete time, Discrete amplitude:
Information is carried, stored and processed in binary form, using gates, registers and other logic devices. These are fully digital circuits; In literature several diverse fully digital Neural Systems can be found [1].

Discrete time, Continuous amplitude:
These are the so-called mixed circuits, or more precisely sampled data circuits. Information is carried by voltage (currents) which are only valid on specific points of time.

For each of the three kinds of approaches, either positive or negative aspects are highlighted. This work deals with the last approach, thus attention will be paid to this latter class of circuits in particular.

Digital ANN's:

The strenghts of the digital ANNs are almost evident:

. digital ANNs are flexible
. digital design techniques are automated and well known
. noise immunity is high
. resolution and precision can be high
. memory uses standard devices
. communication and computation can easily be multiplexed
. digital ANN can be directly integrated in a digital processing environment (e.g. a workstation)

However, for neural networks, they have some unattractive features:

. digital circuits of this complexity work mostly synchronuously while real neural nets are asynchronuous
. all states, activities, etc are quantizied
. digital multipliers occupy large silicon area
. large power dissipation

Analogue ANN's:

The benefits of analogue networks are more subtle

. asynchronuous behavior is obtained automaticly
. smooth neural activities is obtained automaticly
. circuit elements can be small

While on the debit side

. noise immunity is low
. arbitrary high precision is not possible (offset, gain, non-linearity bandwidth and slew rate)

Mixed ANN's:

The term mixed ANN is used when the neural system has internal information represented in analog and/or numeric form. The system then contains a combination of analogue and digital circuits with appropriate A/D/A converters. Another class of ANN hardware which is referred to as mixed are pulse stream circuit which will be worked out in more detail below.

A pulse waveform is defined by three parameters: amplitude, width, rate and phase. Several forms of pulsed modulations act on different parameters:

. Pulse Amplitude (PAM)
. Pulse Rate (PRM)
. Pulse Edge (PEM)
. Pulse Width (PWM)

PWM use binary signals and encode information in the time domain. A convenient variation of PWM is Coherent Pulse Width Modulation (CPWM) [24], with which no edge is constrained to a fixed position, but there is a global time reference which defines a fixed frame of evaluation cycles.

For the CPWM, activation pulses have a constant frequency $f_0 = \frac{1}{T_0}$, while their width is proportional to the activation value:

$$T_i = T_{max} \cdot X_i \text{ , where } X_i \in [0..1] \text{ and } T_{max} \leq T_0$$

(see fig. 3.1.1.a below)
A reference clock (CCK) defines two phases: the CPWM signal is allowed to be "1" only during the active phase, whereas it must always be "0" during the idle phase. An interesting advantage of the CPWM signals is that they can be multiplexed more easily than analogue signals, due to their digital nature and coherence (see fig. 3.1.1.b)

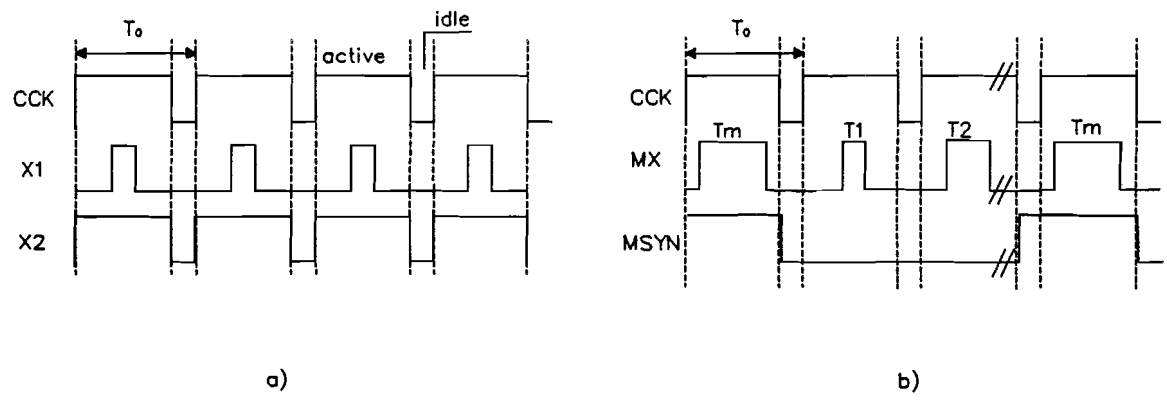


Fig. 3.1.1 Timing diagram of a CPWM modulation

Coherent Pulse Width Modulation incorporates the main benefit of synchronous digital circuits; new information can be transmitted or processed in each cycle. A benefit with respect to synchronous circuits is that ground and power supply current spikes caused by state switching are spread over the evaluation cycle, thus reducing crosstalk noise toward the analogue parts. CPWM outperforms the other Pulse Stream techniques since it presents the lowest computation energy and response time. Further, CPWM is not affected by any frequency error, nor any phase uncertainty.

3.2 Pulse Modulation Arithmetic

Multiplication, summation and other operations required in neural systems should be performed preferably by simple circuits. In most cases, the circuits used to implement PRM, PWM, CPWM and PEM are very similar and some of these functions can also be realized directly in the time domain, using standard logic gates.

- Summation

  - Digital summation of pulses can be performed using OR gates, but overlapping cause a saturation effect.

  - Analogue summation can be accomplished by feeding current pulses on a capacitor, or by averaging the voltage on the current load.

- Multiplication

  The multiplication circuits operate on a pair of pulse parameters.

  - Rate * width circuits usually encode one input in the pulses repetition rate and stretch the width according to the value of the other input.

- Width * height circuits encode one input in the width of single pulse and modify the amplitude according to the other input. They can use any type of analogue multiplier, where one input is restricted to binary variables. This technique provides higher speed than rate * width circuits.

3.3 Structure of the synapse and the neuron unit with CPWM signals

Several authors in Europe [19],[23], have begun to confirm that Coherent Pulse Width Modulation is the most adequate among all the different kinds of pulse modulation.
In fact this technique allows a fairly high integration density even though some compromises have to be made in the network's performances: either in speed, flexibility, computation accuracy or weight resolution.

In figure 3.3.1.a and figure 3.3.1.b the general structure of the synapse and the neuron units respectively, adjusted to the CPWM signals are shown. The pulsed signals are referred to with the index p. On several places in both the Synapse and the Neuron unit, pulsed signals have to be converted into continuous signals and vice-versa.
The two existing kinds of conversions are:

- pulsed current into continuous voltage (type 1)
- continuous voltage into pulsed voltage (type 2)

The circuits realizing these conversions will be elaborated in more detail in chapter 6.

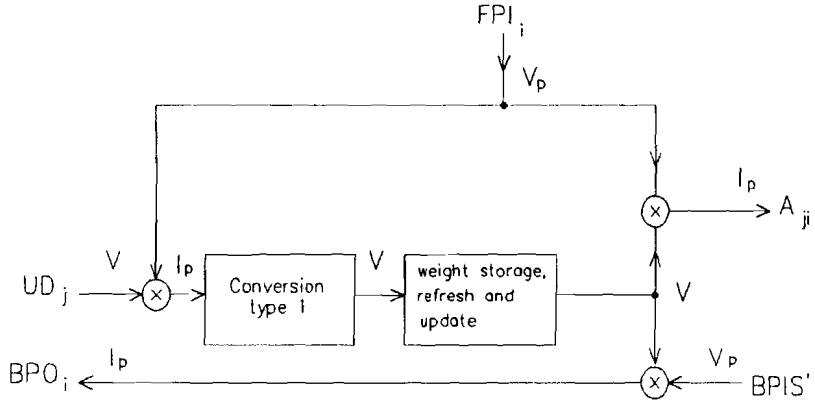


Fig. 3.3.1.a Synapse unit with CPWM signals

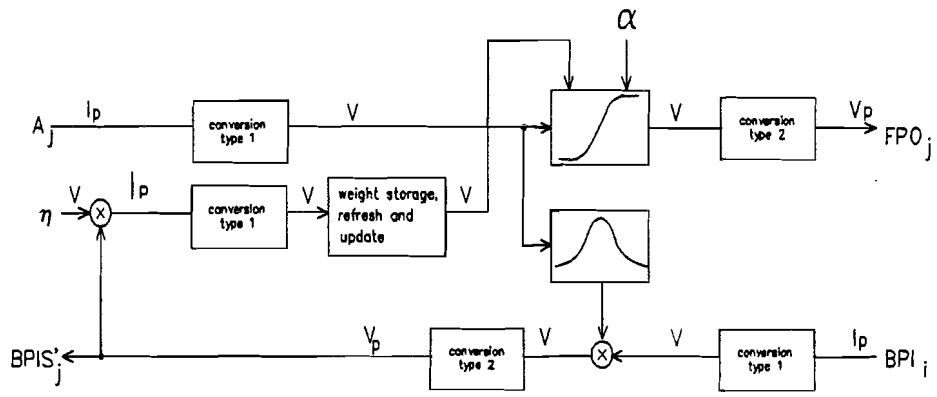


Fig. 3.3.1.b Neuron Unit with CPWM signals

In the backpropagation algorithm (see paragraph 1.3) step 4, the difference between the actual output and the desired output is calculated for each neuron in the output layer before the weights can be updated. In contrast to the idea of a fully flexible structure, a separate neuron chip has been adopted for the output layer of the MLP. In this chip all the Neuron units will perform the calculation of the error. These neuron units are almost similar to the ones shown in figure 3.3.1.b. The conversion type 2 placed after the sigmoid function is suppressed. The output of this neuron unit is thus represented by a continuous voltage, simplifying the interface (also developed in this group) between a PC and the ANN. The circuit performing the generation of the error preceeds the conversion type 1 placed before the four-quadrant multiplier. This circuit has two inputs. One is the output of the circuit realizing the sigmoid function and the other corresponds to the desired pattern, which is also represented by a continuous voltage. Figure 3.3.1.c shows the output neuron unit with CPWM signals.

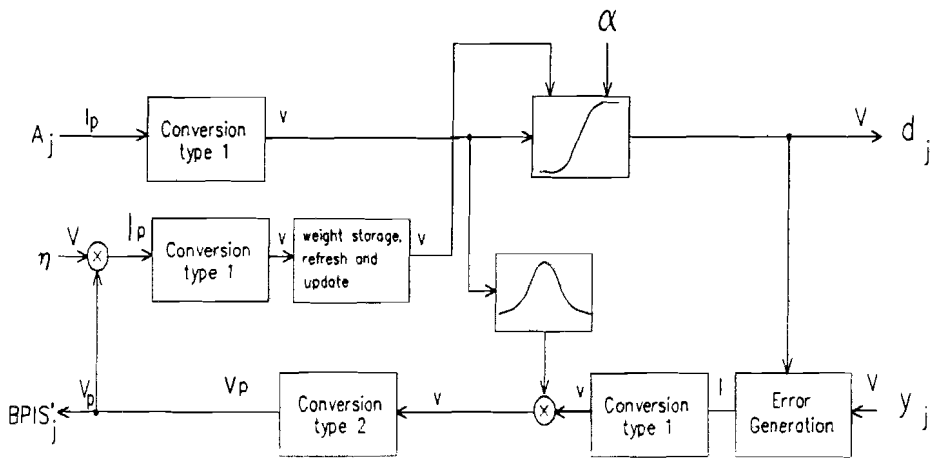


Fig. 3.3.1.c Output neuron unit with CPWM signals

4 The Synapse Unit

The different parts consisting of the feedback path of the synapse unit are presented in this chapter. When designing the synapse unit, several aspects such as the power dissipation and the implementation area have to be considered. The number of synapses which has to be accommodated per Synapse chip is very large. In a MLP, the number of Synapses in one layer is N times (N = number of the layer inputs) bigger than the number of neurons in the layer.
The feedback path of the synapse unit is presented in Figure 4.1

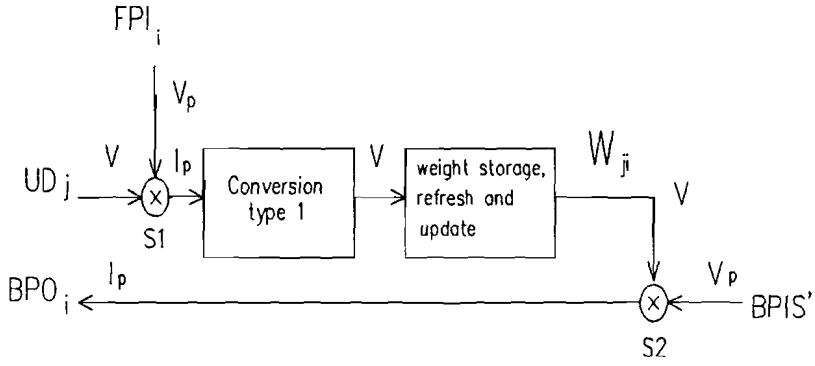


Fig. 4.1 Feedback path of the synapse unit

Once the weights are determined by learning, they have to be used by the network to transform the input vectors to the outputs. During this process the weights have to be stored in the corresponding synapses and neurons. Several weight storage mechanisms exist:

- Digital (the weight is stored as a few bits in a register)
- analog EEPROM (EEPROM is used to store the weights of an analog synapse)
- Capacitive storage (the weight value is stored on a MOS capacitor)

In [15] an analogue weight storage circuit has been developed using the last technique. Since it suffers from weight decay, a refresh circuit has also been proposed. But the weight storage circuit should be redesigned since it has some drawbacks. One can expect that during the learning phase the weights should be also refreshed, but it can be assumed that the refresh is not necessary if the learning speed is high enough. One of the caveats of the hardware implementation of the back-propagation algorithm is the limited dynamic weight range. Since the weight range adopted in [6] of about 2V, corresponds to the maximum possible range with respect to simple circuits, it is thus maintained in this work as well.
The feedback path of the synapse unit consists of two multipliers: multiplier S2 (see fig. 4.1), a four-quadrant multiplier which affects the error signal in function of the weight values and multiplier S1 (see fig. 4.1), a two-quadrant multiplier which multiplies the synapse input FPI_i by the UD_j value (see fig. 4.1). The output of multiplier S1 corresponds to the updated weight value.

4.1 Two-quadrant Multiplier S1

In [6], many different kinds of dynamic multipliers have been proposed. Among all those, the class of the dynamic transconductance multipliers seems to be the most adequate for the CPWM technique. The multipliers of this class have good multiplying linearity and a low power consumption. They function according to the following principle: a voltage-to-current converter is made active or inactive by a CPWM voltage signal.
The functional specifications of the multiplier S1 are listed below:

- two-quadrant multiplier
- input #1: UD_j = continuous voltage signal, positive / negative values
- input #2: FPI_i= pulsed voltage signal, positive values
- output : pulsed current signal, positive / negative values

A structure similar to the one of the dynamic transconductance multiplier implemented in the feedforward path, is applied in the feedback path to perform the multiplication described above. In fig. 4.1.1 the transconductance multiplier S1 is shown with the size of the transistors.

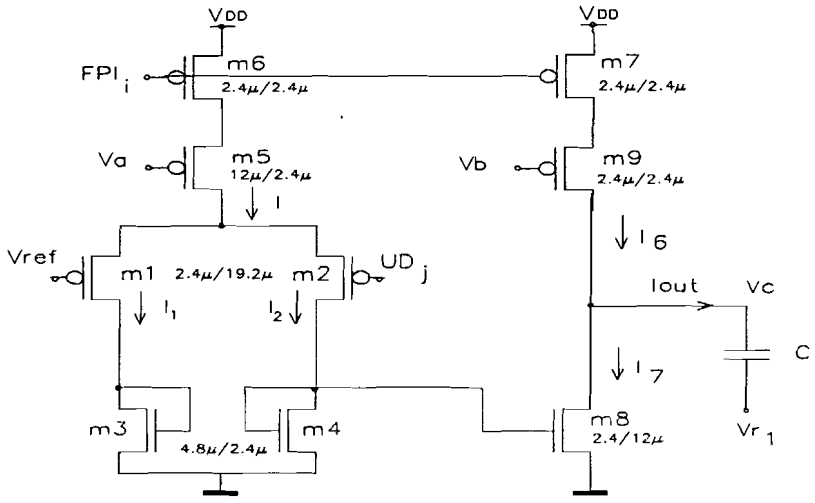


Fig. 4.1.1 Transconductance multiplier S1

The bias voltages Va and Vb are already distributed per synapse chip for the multiplier of the feedforward path. The circuits used to obtain the bias voltages Va and Vb, are described in [6], which is why they are not presented here.
The resulting output current Iout corresponding to the difference between the bias current I6 and the mirrored and scaled current I7 is fed into the capacitor C. This capacitor C is the input capacitor of the conversion stage type 1 (see paragraph 6.2.1) which samples and holds the capacitor voltage V_C before the analogue weight storage stage (see fig. 4.1).
The bias current I6 is determined based on the fact that the output current Iout is equal to zero when the voltage difference between the input voltages UD_j and Vref is equal to zero.

The transfer characteristic of the VCC is shown in figure 4.1.2

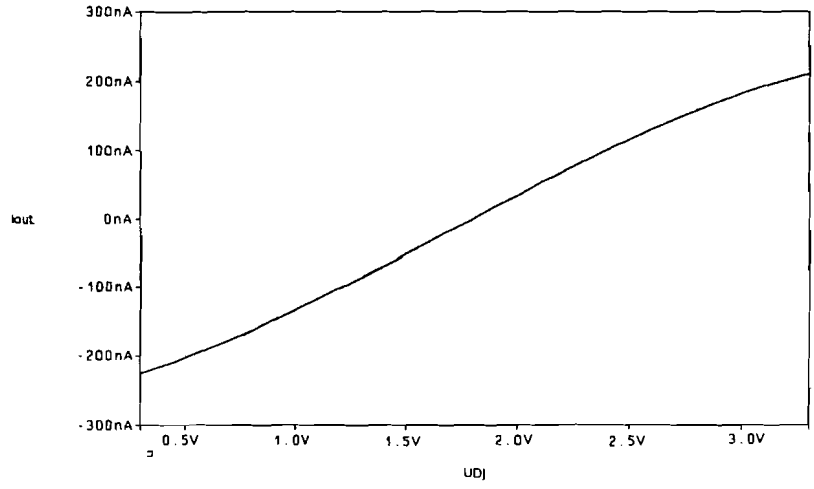


Fig. 4.1.2 Transfer characteristic of the VCC

The sensitivity of the previous circuit to variations in threshold voltage V_T has been checked with the Monte-Carlo analysis in [6]. Variations of $\pm$ 50mv for the same type of transistors between different chips, and $\pm$ 10mv between all transistors inside a same chip have been considered.
The multiplication graphs produced by multiplier S1 are printed in fig. 4.1.3. It shows that this multiplier has an output voltage offset which represent 1% of the total output voltage range.

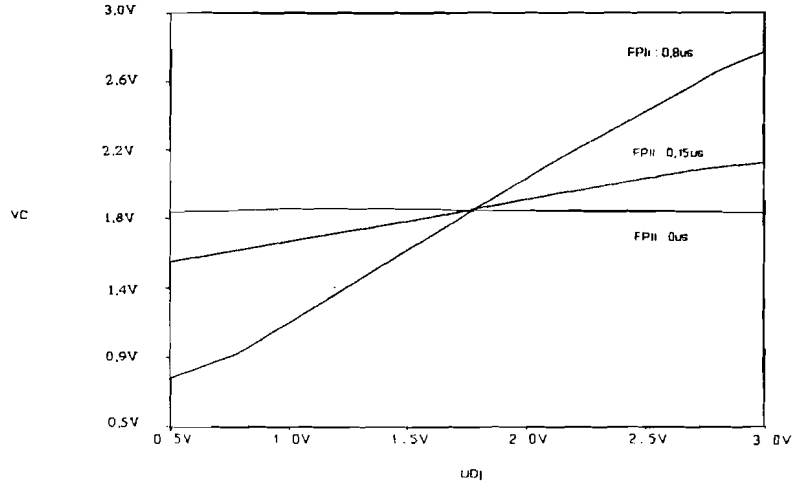


Fig. 4.1.3 The multiplication graphs produced by multiplier S1

The previous multiplier has a good linear range (< 5% of non linearity) of multiplying, for a UD_j voltage range between 0.85 and 2.75 volts with the V_{inzero}=1.8V. The power consumption of this multiplier is very low, since it fluctuates between 4uW and 20uW.

4.2 Four-quadrant Multiplier S2

Another important part of the feedbackward path of the Synapse unit (fig. 4.1) is the four-quadrant multiplier S2. The multiplier performs the multiplication between the back propagated error signal (FPIS') and the weight values (w_{ji}). This multiplier also belongs to the class of the dynamic multipliers, since one of its input is encoded by CPWM voltage (FPIS'). So it has the same working principle as the one described previously in paragraph 4.1.
The functional specifications of the multiplier S2 are listed below:

- four-quadrant multiplier
- input #1: w_{ji} = continuous voltage signals, positive / negative values
- input #2: BPIS' = pulsed voltage signal, positive / negative values
- output = pulsed output current, positive / negative values

Two different possible approaches for the dynamic multiplier S2 are proposed below, followed by a summary of their respective characteristics.

4.2.1 Approach #1

This first approach consist of using the two-quadrant dynamic transconductance multiplier presented in the previous paragraph 4.1 twice to realize a four-quadrant dynamic multiplier. The principle of the four-quadrant multiplier is presented in figure 4.2.1.1

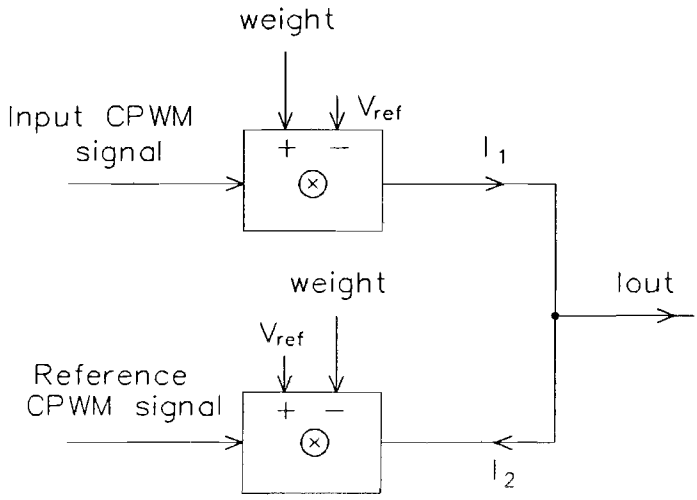


Fig. 4.2.1.1. Four quadrant multiplier S2

The output current I1 of the first two-quadrant multiplier is switched by the normal input CPWM signal, whereas the output current I2 of the second two-quadrant multiplier is switched by a reference CPWM signal. By inverting the reference voltage Vref and the weight voltage w_{ji} of the second multiplier, its VCC characteristic is inverted compared to this of the first one. Since their outputs are connected together, the resulting output current Iout is equal to the current difference between the current I1 and the current I2. In figure 4.2.1.2 the two characteristics of the two

VCCs are shown.

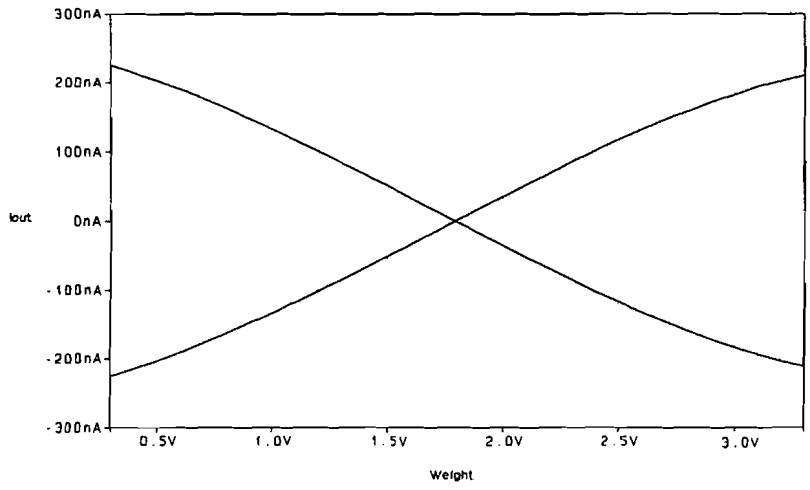


Fig. 4.2.1.2. The two VCC characteristics

When the output weight voltage corresponds to the reference voltage Vref, the absolute value of both currents I1 and I2 are equal to zero (fig. 4.2.1.2) and regardless of the active period of the input signal (BPIS'), the final output current (Iout) is equal to zero. If the active period of the input CPWM signal is the same as the on of the reference CPWM signal, the resulting output current (Iout) is also equal to zero regardless of the voltage difference between Vin and Vref. If two identical two-quadrant multiplier as shown in fig. 4.1.1, are used to obtain the multiplier S2, the latter has an offset voltage of 7% of the total output voltage range. This offset is caused by a delay between the switched currents I7 and I6 of both two-quadrant multipliers. A way to reduce this offset is switching the current I7 of both stages and not the bias current of each transconductance.

Figure 4.2.1.3 shows the improved circuit of the multiplier S2.

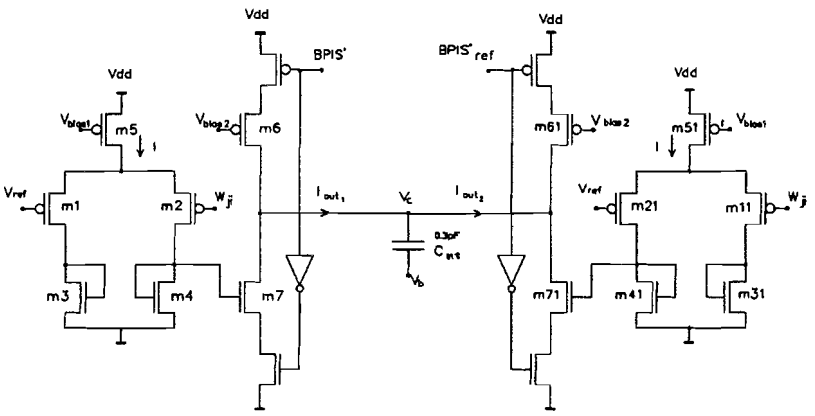


Fig. 4.2.1.3 Improved multiplier S2

In the case of the previous circuit the voltage offset can be reduced to 5 % of the total output voltage range. This multiplier has a low power consumption about 100uW, and requires a large number of transistors. A reference CPWM is needed and since the switches are pmos and nmos transistors the inverted signals of both CPWM signals are required as well. Mismatching between the transitors and treshold voltage variations increase the initial offset. However, a different approach described below, provides a four dynamic multiplier with less transistors and less output voltage offset.

4.2.2 Approach #2

With this second approach, a reference CPWM signal is not required any more, just the input CPWM signal and its inverse. Figure 4.2.2.1.a shows the circuit of the multiplier S2 based on this other principle.

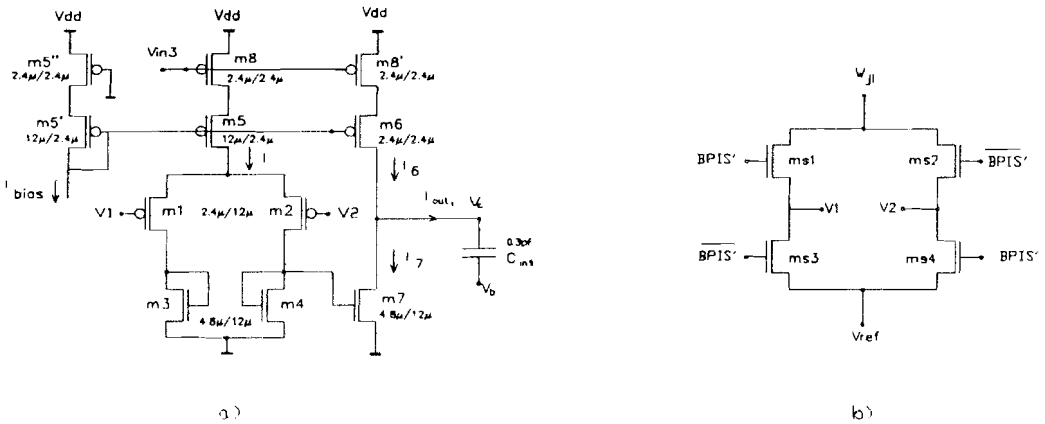


Fig. 4.2.2.1 Multiplier S2

Compared to the circuit in figure 4.2.1.3, only one voltage-to-current converter is used here. The input voltages V1 and V2 of the circuit in fig. 4.2.2.1.a are switched between the volatges UD_j and Vref (see fig. 4.2.2.1.b) in function of the input CPWM voltage FPI_i and its inverse. The circuit has the same VCC characteristic as the one shown in fig. 4.1.2. Input voltages of the tranconductance stage are switched according the CPWM signal and its inverse. The bias current I6 and I are switched by an external signal (Vin3) synchronized with the CPWM signal. This external signal has an active period of 0.8us, corresponding to the maximum active period possible.
The value of the current I6 is determined based on the fact that the output current Iout is equal to zero when the gate voltages of the transistor m1 and m2 are continuous and equal to the reference voltage Vref. The logic circuitry required for controlling the four switches is presented in figure 4.2.2.1.b. When the active period of the input CWPM signal is 0.4us, the output current integrated on the capacitor provides no voltage changes with respect to the reference voltage Vb.

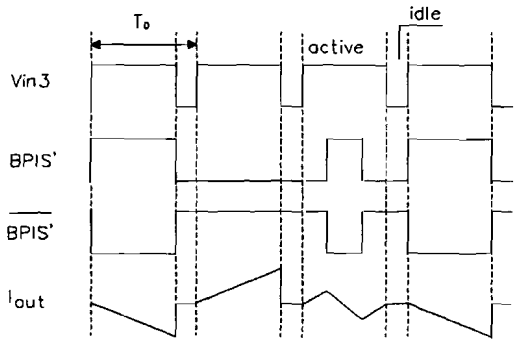


Fig. 4.2.2.2 Timing of the CPWM signals

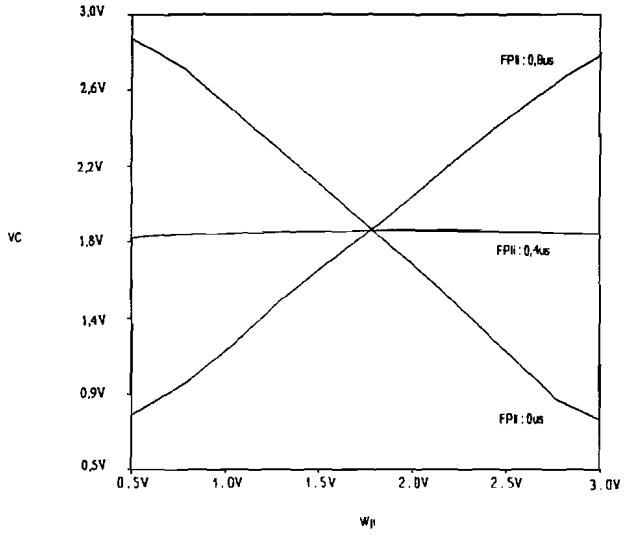


Fig. 4.2.2.3 The multiplication graphs produced by multiplier S2

Figure 4.2.2.2 shows the timing of the different signals used in the latter approach. Figure 4.2.2.3 shows the graphs of the multiplication of the multiplier S2. In this figure, one can see that the multiplier S2 has an voltage offset equal to approx. 2 to 3 % of the total output voltage range. The power consumption is approx. 50uW.
Instead to switch the bias currents I and I6, two nmos transistors can be used to set the voltages V1 and V2 during the idle period 0.45us. Simulations have shown that this solution is more sensitive to mismatching between transistors than the previous soltution.

Conclusions

From the two previous approaches proposed above, the dynamic multiplier of the second approach seems to be the most performant. It has a lower offset and lower implementation area compared to the first approach. It suffers less from mismatching between transistors. The power consumption is also reduced considering the second approach. The results of Monte-Carlo analysis for both solutions show that the second circuit is also less sensitive to variation in the threshold voltage. One drawback of the second technique is that it requires an extra pin per chip for the external signal (Vin3). For the first circuit, the reference CPWM signal needed in the forward path can be used. The offset of both dynamic multipliers increases with the variations in the treshold voltage. After simulations it can be assumed that offsets higher than 3% pertube the backpropagation training algorithm. Further simulations should be done to see if the influence of these offsets on the training algorithm can be decreased.

5 The Neuron Unit

This chapter deals with the feedback path of the neuron unit for the hidden layers of the MLP. This neuron unit is different from the output neuron unit presented in the following chapter. The output neuron incorporates an extra circuit performing the error calculation. The feedforward path of the neuron unit has been slightly modified. Therefore a sigmoid function with a variable steepness is proposed below.

5.1 Sigmoid Function

It has been proved that the linear auto scaling function proposed in [6],[28], is not satisfying and the forward path has been slightly modified to perform the new requirements. Figure 5.1.1 shows the new feedforward path of the neuron unit.

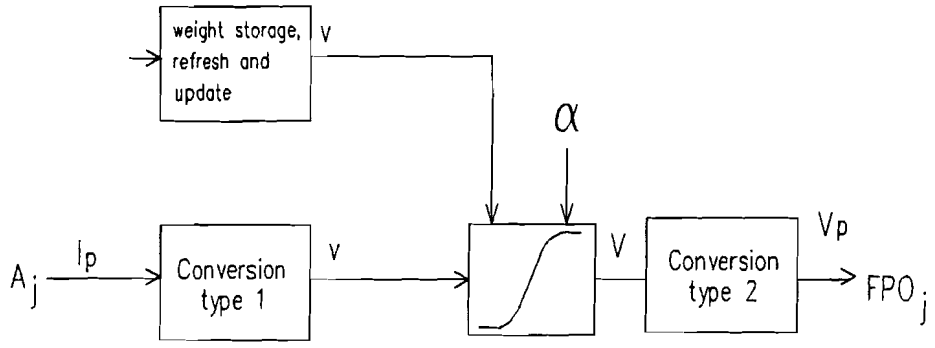


Fig. 5.1.1 Feedforward path of the neuron unit

In [6], two types of circuits performing the sigmoid function, have been proposed. The first one has a fixed sigmoid function. The bias weight output voltage is converted into a current which can shift the input range of the sigmoid function. The second circuit incorporates the shift into the sigmoid function. In this circuit the bias weight output voltage can directly shift the sigmoid function. The new scaling function depends on the square root of the number of synapses connected to one neuron (see paragraph 2.3), and seems very difficult to implement. It seems therefor easier to implement a sigmoid function with a variable steepness. Assuming that all the neurons from one layer have the same number of inputs, the slope of the sigmoid function of all the neurons placed in the same layer can be tuned by means of one voltage. This voltage is set to a value proportional to the inverse of the square root of the number of synapses connected to the neurons. In literature [12], several circuits performing the sigmoid function with a variable slope have been proposed. They have all the same disadvantage: their large implementation area and their complexity. The second sigmoid circuit designed in [6] has a low power consumption and a

small implementation area, it seems therefore better to adapt this circuit to the new requirements. Figure 5.1.2 shows the sigmoid circuit used in [6] with the dimensions of the transistors.

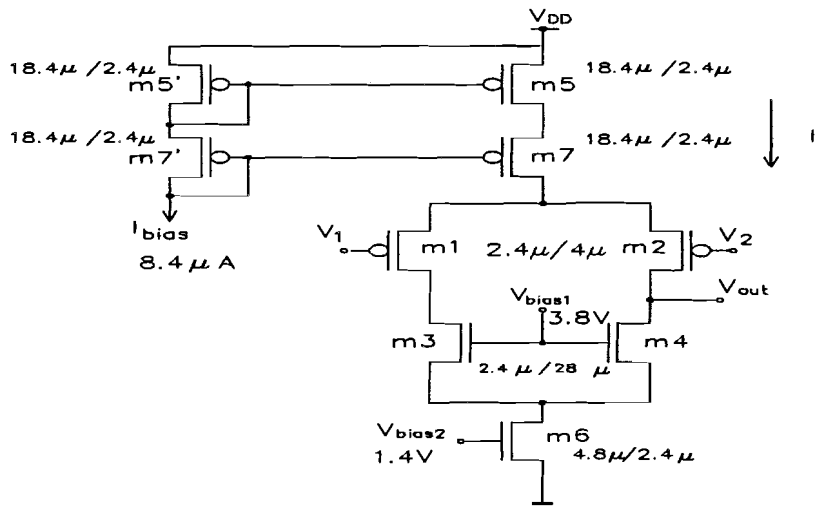


Fig. 5.1.2 Sigmoid circuit

In the previous circuit, one can easily see that the slope of the sigmoid transfer function can be changed by controlling the gain in the voltage difference Vd=V1-V2. A way to control the gain in the differential input voltage of the sigmoid circuit is applying a transconductance stage with a transistor connected between its two loads. This transistor works in the triode region and its gate voltage allows the gain of differential output voltage of the stage to be controlled. Figure 5.1.3 shows the circuit used for controlling the gain in voltage difference V1-V2.

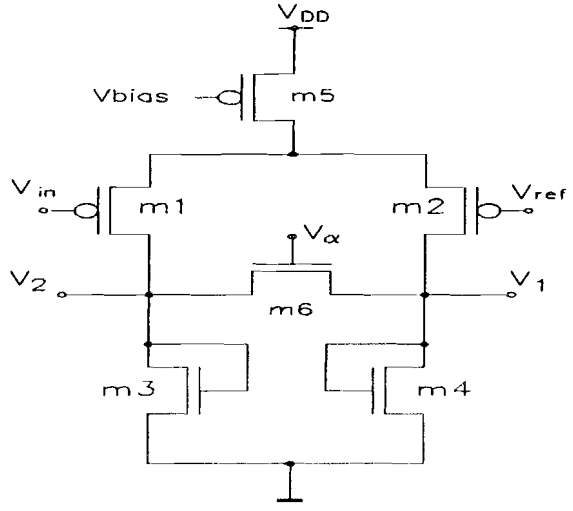


Fig. 5.1.3 Transconductance amplifier with variable gain

The gate voltage V_α of the NMOS transistor m1 is tunable between 2V and 5V and the gain in voltage difference V1-V2 can be changed within a factor 7. At least 64 synapses could be connected to one neuron. It implies that the slope of the sigmoid function has to be tunable at

minimum within a factor $\sqrt{64}=8$, [6]. Therefore the voltage gain of the circuit of the figure 5.1.3 has to be increased. This can be achieved by adding four extra transistors working in saturation. Figure 5.1.4 shows the improved circuit.

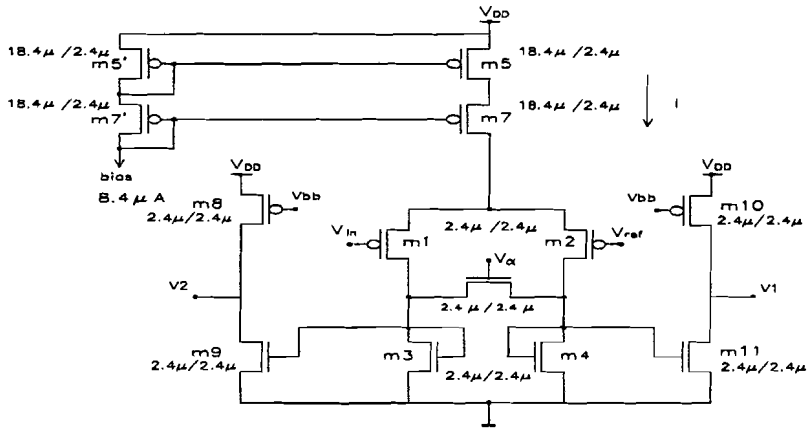


Fig. 5.1.4 Improved transconductance amplifier with variable gain

In this stage, the voltage gain is tunable within a factor 11. Thus the slope of the sigmoid function can be changed within a factor 11 as well. This means that a maximum of 121 synapses can be connected to one neuron. In connecting the circuit in figure 5.1.4 to the one in figure 5.1.1, a sigmoid unit incorporating a variable slope and a shift of the sigmoid function is obtained. Sigmoid transfer functions with different slopes for a weight input voltage Vref=1.5V are printed in figure 5.1.5. Different shifted sigmoid transfer characteristics for an input voltage V_α of 3V are shown in figure 5.1.6. The nominal input range of this circuit is between 0 and 3V.

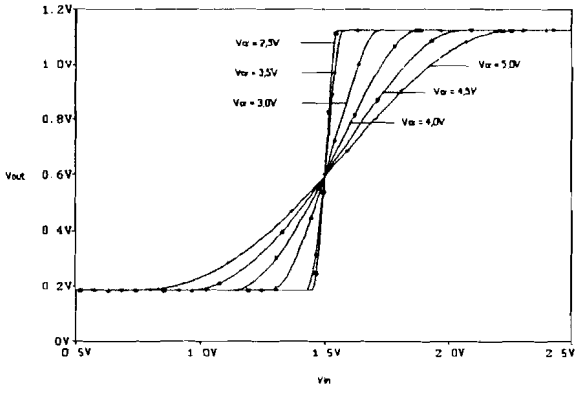


Fig. 5.1.5 Different slope of the sigmoid

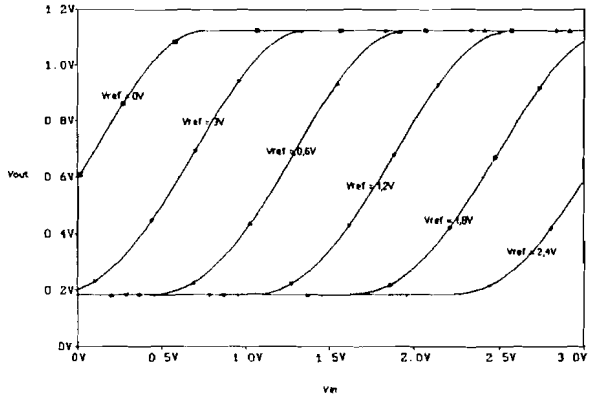


Fig. 5.1.6 Shift of the sigmoid function

Monte-Carlo analysis has shown that the previous circuit is too sensitive to variation in the threshold voltage V_T. The output voltage range of the sigmoid is identical to that used in [6], therefore the rest of the feedforward path is not modified.

5.2 Feedback path

The feedback path of the neuron unit for the hidden layers is shown in fig 5.2.1

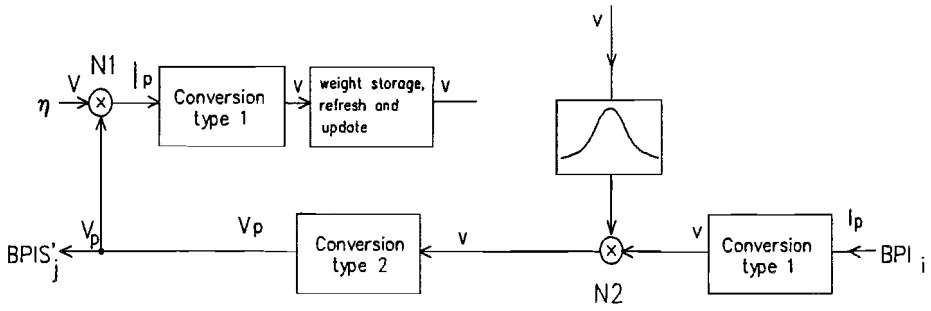


Fig. 5.2.1 Feedback path

The implementation of the backpropagation learning algorithm has many advantages, see paragraph 1.3, but it increases the implementation area of the neuron unit considerably. This training algorithm requires the derivative of the sigmoid function and two multipliers. Two types of converters are present in different places in the path, which will be discussed briefly in paragraph 6.2. The remark made in the previous chapter concerning the weight output voltage range remains valuable for the bias weight of the neuron. One very important part of the feedback path is the derivative of the sigmoid function.

5.2.1 Derivative of the Sigmoid Function

In the literature [11],[16],[27], only few implementations of the backpropagation learning algorithm exist. Most of them have a large implementation area and a large power consumption. Two different approaches for implementing the derivative of sigmoid function are proposed in the following paragraphs.

5.2.1.1 Approach #1

One way of obtaining the derivative of the sigmoid without performing the multiplication of the terms S() and 1-S(), is to apply part of the sigmoid transfer characteristics S() and 1-S(). According to the weight output voltage is lower or larger than the Vref voltage either the sigmoid characteristic S() or 1-S() is used. This approach provides a circuit with low power consumption and a small implementation area, since the circuit of the sigmoid is implemented in the feedforward path. Figure 5.2.1.1.1 shows the circuit used to approximate the derivative of the sigmoid .

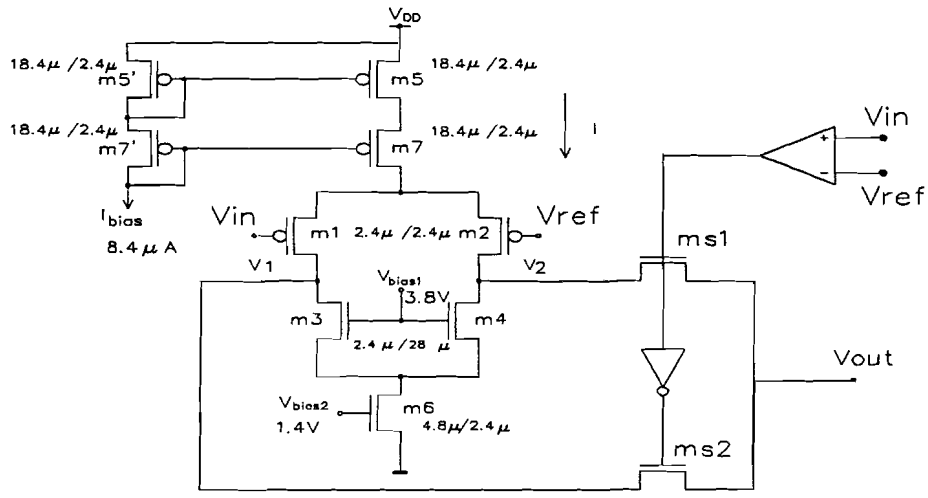


Fig. 5.2.1.1.1 Derivative circuit #1

The input voltage of the sigmoid unit Vin is compared with the weight output value Vref. The output voltage of the comparator affects the position of the switches (ms1 and ms2). The NMOS transistors (ms1, ms2) used as switches, provide an output voltage equal to either V1 or V2. The output voltage of the derivative circuit is plotted in figure 5.2.1.1.2 for a set of weight output voltage Vref (1.1V, 1.5V, 1.9V). The nominal input voltage range Vin is between 0 and 3V.

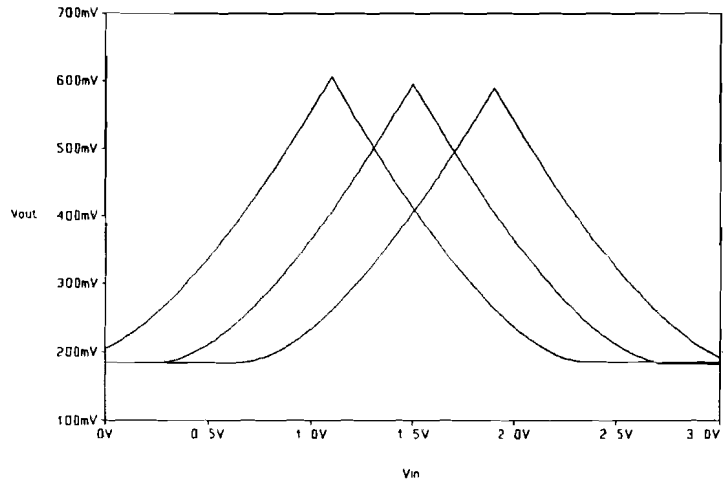


Fig. 5.2.1.1.2 Derivative transfer characteristic

The shape of the transfer function is similar to the one suggested in [29]. Simulations have to be done on a neural net simulator to check if this approximated shape of the derivative of the sigmoid can be used. A disadvantage of this circuit is its relatively small output voltage range of 0.8V.
Based on the same technique, the circuit of the figure 5.2.1.1.3 has a transfer function much closer to the ideal derivative function and a larger output voltage range. In this circuit, two sigmoid stages are used instead of one. They have two different reference voltages Vref1 and Vref2 but have the same input voltage Vin. A part of their respective transfer functions are used used, by switching transistors ms1 or ms2, i.e. S() of the sigmoid stage having Vref1 as reference

voltage and 1-S() of the other sigmoid stage stage (Vref2).

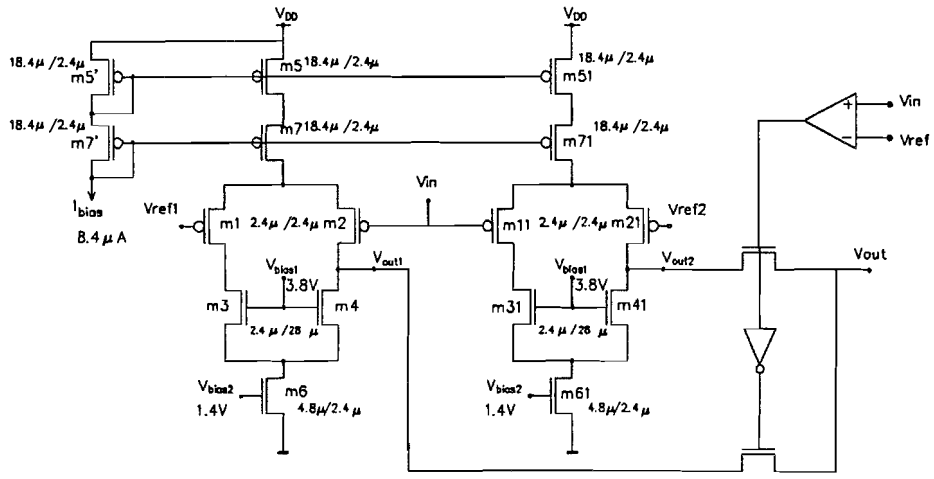


Fig. 5.2.1.1.3 Improved derivative circuit #1

The transfer function of this circuit shown in figure 5.2.1.1.4. has been obtained for an input voltage range between 0 and 3V and two refrence voltages Vref1 and Vref2 equal to 1V and 2V respectively.

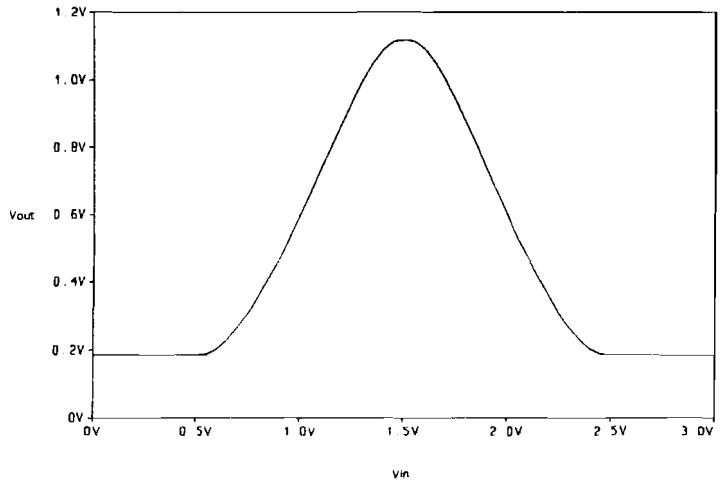


Fig. 5.2.1.1.4 Improved derivative transfer characteristic

This circuit has a large implementation area. It requires an extra circuit to keep Vref1 and Vref2 at $\pm$ 2V of the reference voltage Vref over the total range of this voltage. In all the hardware implementations of the derivative of the sigmoid function presented in the literature [11],[16] an analogue multiplier is used. Therefore, the second approach presented below is based on this technique.

5.2.1.2 Approach #2

The mathematical formula of the derivative of the sigmoid function can be expressed as follows:

$$F'_S(x) = \frac{\partial (Fs(x))}{\partial x} = \frac{\tau e^{-\tau (x-\alpha)}}{1 + e^{-\tau (x-\alpha)}} = \tau Fs(x)(1 - Fs(x)) \qquad (5.2.1.2.1)$$

where τ is the steepness and α an offset. One can see in (5.2.1.2.1) that the derivative can be expressed as a product of the sigmoid function itself.
This implies that the derivative of the sigmoid can be obtained by multiplying the two terms Fs and (1 - Fs). The steepness factor τ present in (5.2.1.2.1) has to be also multiplied with the product of the two terms S() and 1-S(). This can be done without implementing an extra multiplier, just by adapting the learning rate η in function of τ before the learning is applied to the neuron unit. This is possible because it affects the error signal as well (see the backward path in figure 5.2.1). The two terms Fs and (1-Fs) are represented by the two output voltages at the two nodes of the loads of the differential pair of transistors used in the sigmoid unit (see figure 5.1.1). In the literature, an improved wide range Gilbert multiplier is often used to perform the multiplication. This circuit has an output current and required large bias currents implying a larger power consumption. In figure 5.2.1, one can see that the output of the derivative stage is the input of the analogue multiplier N2. Multiplier N2 has to have an output represented by a voltage (see chapter 6.2.1). Four conversions take place in the feedbackward path, therefore it is better to investigate an analogue multiplier with an output voltage in order to limit the number of conversions as much as possible.
Several authors [14],[18],[26], have proposed a multiplication cell using four transistors working in their triode regions. The two-quadrant multiplier N2 is also based on the same multiplication cell, which is detailed in paragraph 5.2.3. Figure 5.2.1.2.1 shows the complete circuit performing the derivative of the sigmoid function.

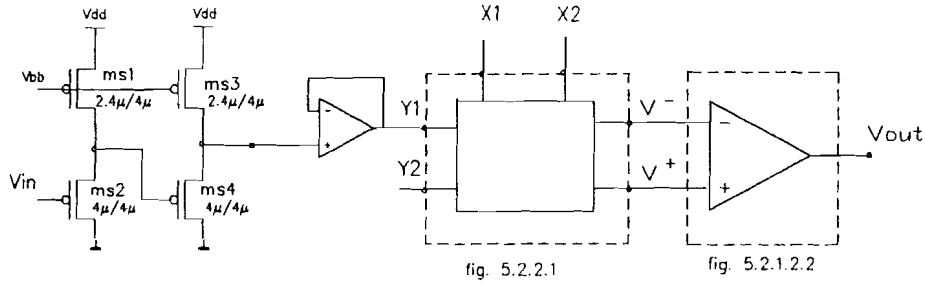


Fig. 5.2.1.2.1 The global derivative circuit

The differential voltage between the drain voltages of transistors m4 and m6 of the circuit of the sigmoid (fig. 5.1.2) is connected to the differential input X1 and X2 of the multiplication cell. The input voltage Y2 is connected to a bias voltage, whereas Y1 corresponds with the shifted voltage V_{in}, the latter is equal to the drain voltage of the transistor m3 of the sigmoid circuit (fig. 5.1.2).

The bias voltage Y2 is equal to the minimum value of the voltage Y1.The four transistors ms1 to ms4 introduce a voltage shift of 2V. An operational amplifier configured as a source follower, plays the role of input buffer for the input Y1. The multiplier cell provides a differential output voltage (V+ - V-). A transconductance amplifier stage (see figure 5.2.1.2.2) is used to obtain a single output voltage. The output voltage can be expressed as follows:

$$V_{out} = K\,(V^{+}-V^{-}) \quad \text{where} \quad K = 2\frac{\sqrt{\mu_1 C_{ox_1}\frac{W_1}{L_1}}}{\sqrt{I_{D5}\,(\lambda_1+\lambda_3)}}\,. \qquad (5.2.1.2.2)$$

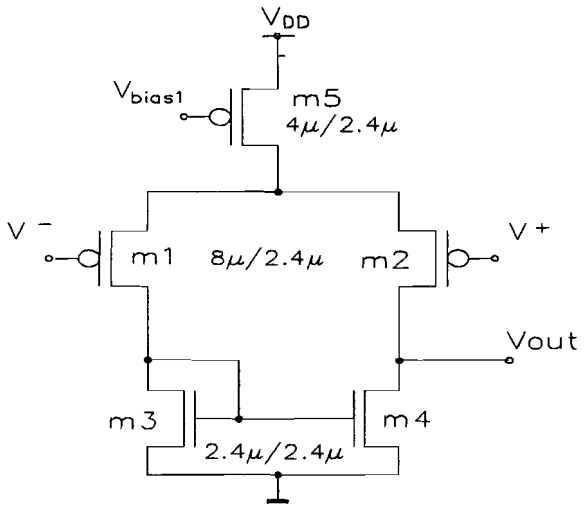


Fig. 5.2.1.2.2 Gain Stage

By taking all the transistors of minimum length and a bias current I_{D5}=10uA, a voltage gain of 30 can be achieved. PSPICE simulations have been done for different bias weight voltages, see figure 5.2.1.2.3. The results of the Monte-Carlo analysis presented in figure 5.2.1.2.4 shows that this circuit is considerably less sensitive to variation into the threshold voltage V_T ($\pm$ 50mV between chips and $\pm$ 10mV within one chip). Although long transistors are used in the multiplier cell (24um), the settling time of an impulse response is short i.e. 100ns.

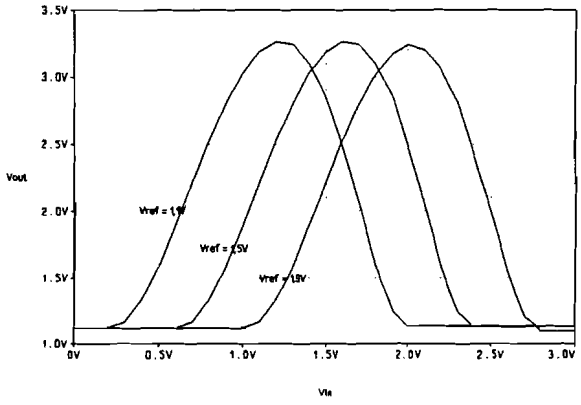


Fig. 5.2.1.2.3 Derivative transfer characteristic

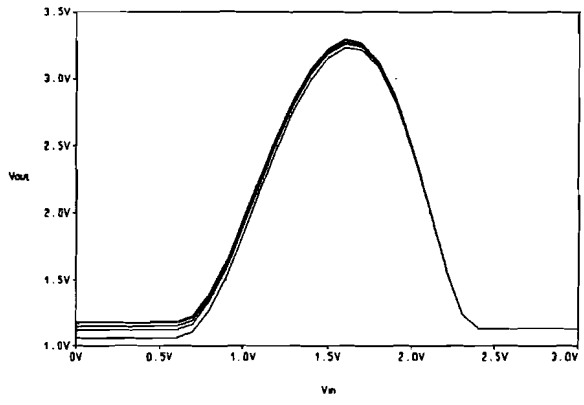


Fig. 5.2.1.2.4 Monte-Carlo Analysis

The shape of the derivative is not completely symmetrical and it is positively shifted of 0,1V. This shift is due to the non-linearities introduced by the multiplier. Simulation of a MLP with such derivative characteristic has shown that the performances of the neural net are not strongly affected by the non ideal shape of this derivative function.

Conclusion:

Although the last circuit has a larger implementation area than the circuit proposed in figure 5.2.1.1.1, it provides better results. The two circuits proposed in the first approach (fig. 5.2.1.1.1 and 5.2.1.1.3) suffer from many drawbacks and the circuit in fig. 5.2.1.2.1 seems to be the best option.

5.2.2 Two-quadrant Multiplier N2

In figure 5.2.1, the error incoming signals BPI_i of the synapses are pulsed currents. A conversion stage type 1 performs the conversion into a continuous voltage. Multiplier N2 multiplies this voltage by the output voltage of the derivative circuit. The functional specifications of this multiplier are listed below:

- two-quadrant multiplier
- input #1 BPIS':continuous voltage signal positive / negative value
- input #2 S' :continuous voltage signal positive value
- output :continuous voltage signal positive / negative value

In literature [3],[5],[18] many analogue multipliers are presented. Most of them have a current output and require large bias currents. A one-quadrant analogue multiplier is also used for the derivative of the sigmoid and therefore the investigation of a small analogue multiplier having a

voltage output is emphasized. In recent publications [14],[16], a four transistor continuous time MOS transconductor has been proposed. Furthermore in [14], an analogue MOS implementation of synapses has been developed using this circuit. The MOS transconductor shown in figure 5.2.2.1 can be made with four identical NMOS or PMOS transistors. These transistors function in their triode region. Further the circuit of the figure 5.2.2.1 will be connected to an op-amp, see figure 5.2.2.2.

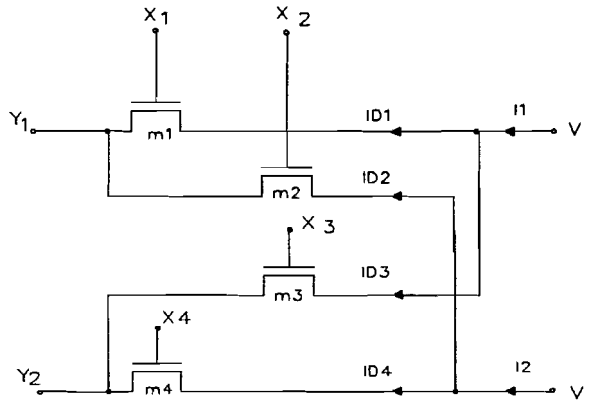


Fig. 5.2.2.1 Four-Transistors MOS Transconductance

According to the current drain of a transistor working in triode region

$$I_D = \mu C_{OX}\frac{W}{L}[(V_{GS}-V_T)V_{DS}-\frac{V_{DS}^2}{2}] \qquad (5.2.2.1)$$

each current drain can be expressed as:

$$I_D = I_{LD} - I_{ND} \qquad (5.2.2.2)$$

where I_{ND} is a nonlinear term $\mu C_{OX}\frac{W}{L}\frac{V_{DS}^2}{2}$ and I_{LD} is a linear term $\mu C_{OX}\frac{W}{L}(V_{GS}-V_T)V_{DS}$ in V_{DS} .

One can write the following formula for the nonlinear term of the current I1

$$-I_{NI} = I_{ND1} + I_{ND3} = \frac{\mu\ C_{OX}}{2}\frac{W}{L}[(V-Y_1)^2 + (V-Y_2)^2] \qquad (5.2.2.3)$$

Since the non linear component IN2 of the current I2 is exactly equal to IN1 , a complete cancellation of those non linear terms in the current difference I1-I2 is realized. The linear component of I1 is given by:

$$I_{LI} = I_{LD1} + I_{LD3} = \mu C_{OX}\frac{W}{L}[(X_1-Y_1-V_T)(V-Y_1) + (X_3-Y_2-V_T)(V-Y_2)] \qquad (5.2.2.4)$$

whereas that of I2 is given by

$$I_{L2} = I_{LD2} + I_{LD4} = \mu C_{OX}\frac{W}{L}[(X_2-Y_1-V_T)(V-Y_1) + (X_4-Y_2-V_T)(V-Y_2)] \quad (5.2.2.5)$$

By making X1=X4 and X2=X3, the difference between the two output currents results in:

$$I_2 - I_1 = \mu C_{OX}\frac{W}{L}(Y_1-Y_2)(X_1-X_2) \quad \text{where} \quad X_1, X_2 \geq \max [\ Y_1+V_T\ ,\ Y_2+V_T\ ] \quad (5.2.2.6)$$

to ensure the linear operation for all MOS transistors. Figure 5.2.2.2 shows the complete multiplier N2 and figure 5.2.2.3 shows the op-amp used in the previous multiplier.

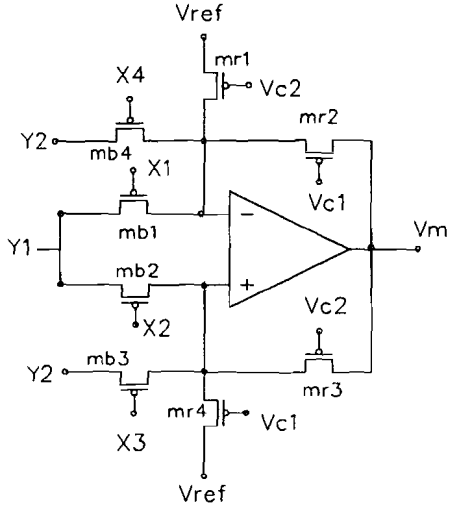


Fig. 5.2.2.2 Multiplier #1

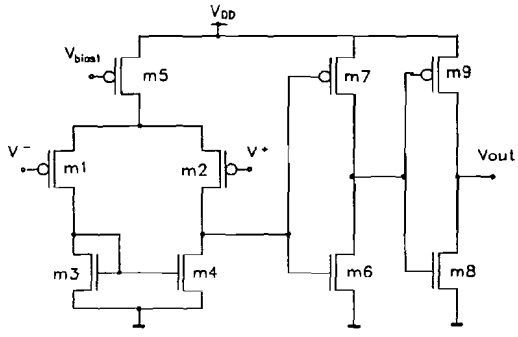


Fig. 5.2.2.3 Op-Amp

The op-amp has a feedback composed of the same type of transistors as those used in the circuit of the figure 5.2.2.1. They work also in their triode region. The following general formula can be derived from the application of the Kirchoff current law to the two nodes of the input terminal of the op-amp:

$$V_m = \frac{K_i\ (\ Y_1-Y_2\ )(\ X_1-X_2\ )}{K_r\ (\ V_{C1}-V_{C2}\ )} + V_{ref} \quad \text{where} \quad K_{b_i} = \mu C_{OX}\ (\frac{W}{L})_{b_i} \quad \text{and} \quad K_{r_i} = \mu C_{OX}\ (\frac{W}{L})_{r_i} \quad (5.2.2.7)$$

By adjusting the voltage VC1 and VC2 the dynamic output voltage range of Vm can be changed. The four transistors used in the feedback of the op-amp work in the triode region under the following conditions.

$$X_i-Y_i \leq V_T < 0 \quad (5.2.2.8)$$

$$V_{ci}-V_T \leq V_{ref} \quad \text{and} \quad V_{ci}-V_T \leq V_m \quad (5.2.2.9)$$

The reference voltage Vref is 2.5V. The previous circuit can be used to realize either a one or a two or four-quadrant multiplier. Figure 5.2.2.4 shows the multiplication graphs produced by the multiplier #1.

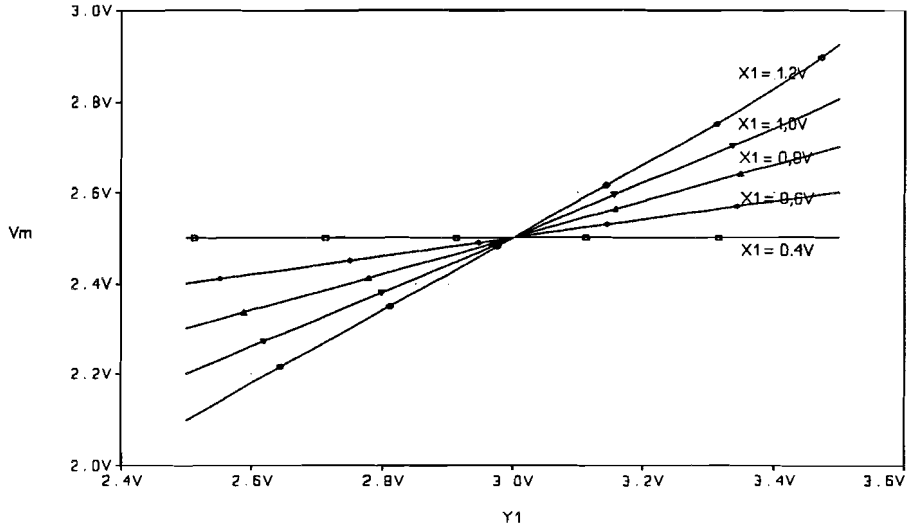


Fig. 5.2.2.4 Multiplication graphs produced by the multiplier #1

The input range of X1 is between 0.V and 1.V, while this of the second input Y1 is between 2.5V and 3.5V. The output voltage range is between 2.1V and 2.9V. Larger the open gain loop of the op-amp, smaller the offest introduced by the multiplier. One major drawback of the previous circuit is that it requires a large number of transistors and that it has a non negligible power consumption. Further the complexity of the op-amp increases if multiplier has to receive high frequency signals. Therefore this multiplier is not further investigated.

When the accuracy requirements are modest for the multiplication, an alternative circuit has been proposed in [20]. Here, a simple transconductance operational amplifier without feedback is used. The complete improved circuit is shown in figure 5.2.2.5. The transconductance amplifier is presented in detail in figure 5.2.2.6.

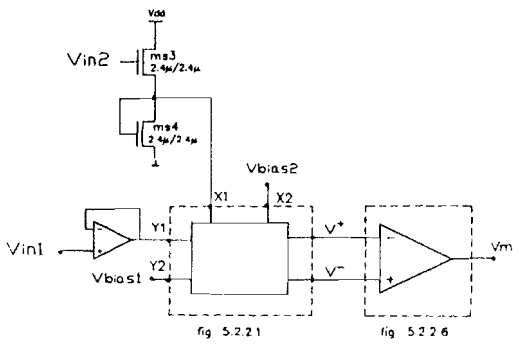


Fig. 5.2.2.5 Improved multiplier #2

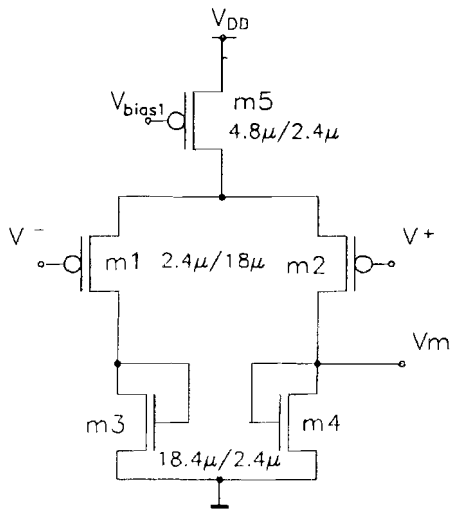


Fig. 5.2.2.6 Transconductance amplifier

In figure 5.2.2.5, the output of the multiplier cell is connected to the input terminals of the op-amp. This implies that the currents I1 and I2 of the multiplication cell shown in fig. 5.2.2.1 are equal to zero. The currents I1 and I2 can be expressed as follows:

$$I_1 = I_{D1}+I_{D3} = I_{LD1}+I_{LD3}+I_{ND1}+I_{ND3} = 0 \qquad (5.2.2.10)$$

$$I_2 = I_{D2}+I_{D4} = I_{LD2}+I_{LD4}+I_{ND2}+I_{ND4} = 0 \qquad (5.2.2.11)$$

where I_{LDi} are the linear terms and I_{NDi} are the non linear terms

$$I_{LDi} = \mu C_{OX}\frac{W}{L}(V_{GS_i}-V_T)V_{DS_i} \qquad (5.2.2.12)$$

and

$$I_{NDi} = \mu\frac{C_{OX}}{2}\frac{W}{L}V_{DS_i}^2 \qquad (5.2.2.13)$$

where $1 \leq i \leq 4$

By replacing the terms (5.2.2.12) and (5.2.2.13) in the current I1 and I2, the following general formula can be obtained:

$$V^+-V^- = \frac{(X_1-X_2)(Y_1-Y_2)}{(V^++V^-)-(X_1+X_2)+V_T} \qquad (5.2.2.14)$$

where V_T is the threshold voltage. Finally

$$V_m = K\,(V^+-V^-) \qquad (5.2.2.15)$$

where $K \simeq \sqrt{\frac{W_1}{L_1}\frac{L_3}{W_3}}$

For a small dynamic input voltage range (Y1-Y2), the two terms (X1+X2) and (V+ - V-) in the denominator of (5.2.2.14) are negligeable which results in:

$$V_m = K\,\frac{(X_1-X_2)(Y_1-Y_2)}{(V^++V^-)-(X_1+X_2)+V_T} \qquad (5.2.2.16)$$

By using four PMOS transistors with the bulk connected to the source instead of four NMOS transistors, the linearity of the multiplier can be increased.
The multiplication graphs produced by the improved multiplier #2 are printed in figure 5.2.2.7. They have been obtained for

$-0.6V \leq Y_1-Y_2 \leq +0.6V$ where $Y_2=3V$ and

$-1.2V \leq X_1-X_2 \leq 0V$ where $X_1=0.4V$

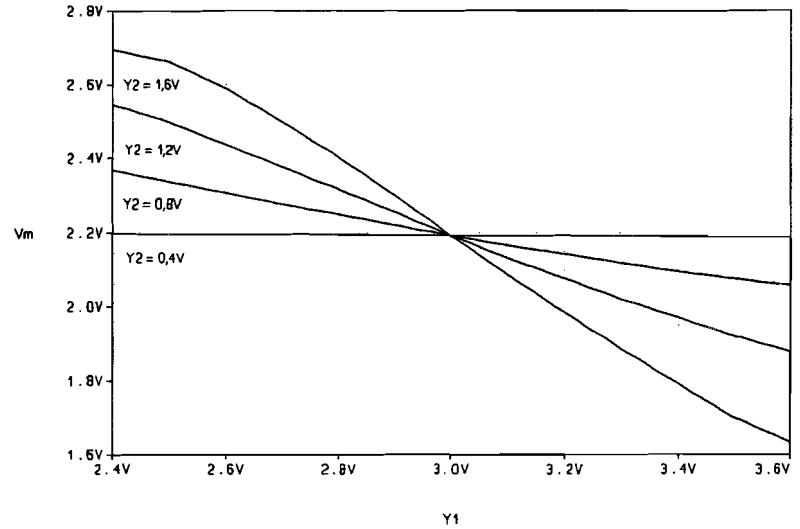


Fig. 5.2.2.7 The multiplication graphs produced by multiplier #2

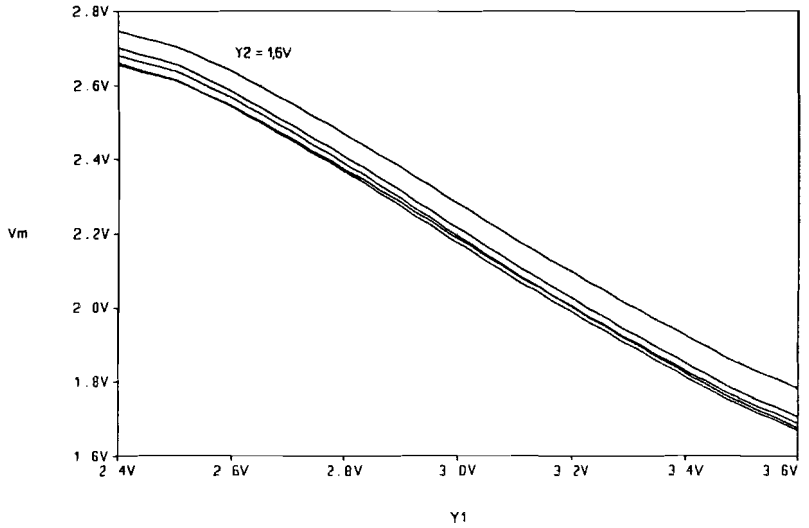


Fig. 5.2.2.8 Monte-Carlo Results

The results of the Monte-Carlo analysis are printed in figure 5.2.2.8. They show that the circuit in figure 5.2.2.5 is less sensitive to threshold voltage variation. By using long transistors (L = 24um) for the multiplier cell, the currents can be decreased. Simulations for input impulses have shown a settling time of approx. 150ns.

5.2.3 Two-quadrant Multiplier N1

In figure 5.2.1, multiplier N1 performs the multiplication of the learning rate η by the error signal BPIS'. Its result represent the updtate value of the weight. The functional specifications of the multiplication are listed below.

- two-quadrant multiplication
- input #1: η continuous voltage positive value
- input #2: BPIS' pulsed voltage positive / negative values
- output: pulsed output current positive / negative values

This multiplier is based on the same principle as the multiplier S2, only the characteristic of the VCC is different. Therefore its functional behaviour will not be detailed here. A very important aspect of this multiplier is its dynamic range of multiplying. In fact the learning rate may vary in a range of 1 to 100. Figure 5.2.3.1 shows the circuit of the multiplier N1.

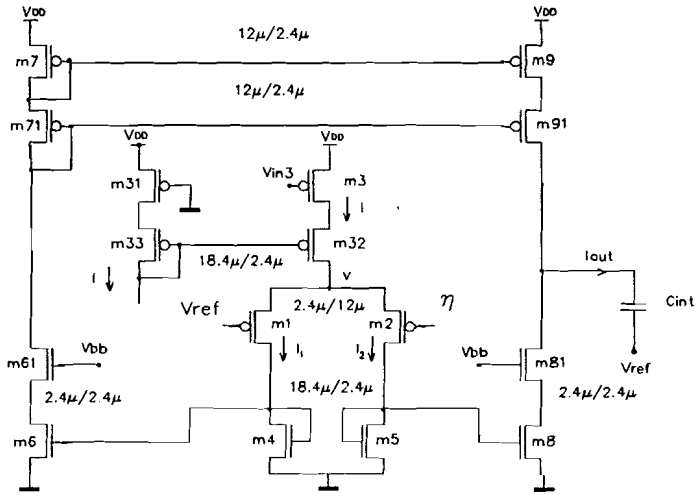


Fig. 5.2.3.1 Multiplier N1

The VCC characteristic of the multiplier N1 is shown in figure 5.2.3.2. The continuous input voltage is between 1.5V an 3V. Figure 5.2.3.3 shows the multiplication graphs produced by multiplier N1.

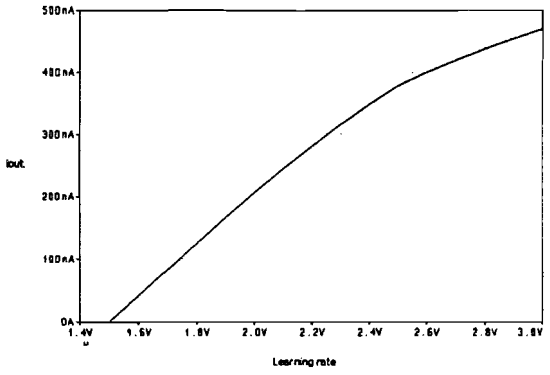


Fig. 5.2.3.2 VCC characteristic

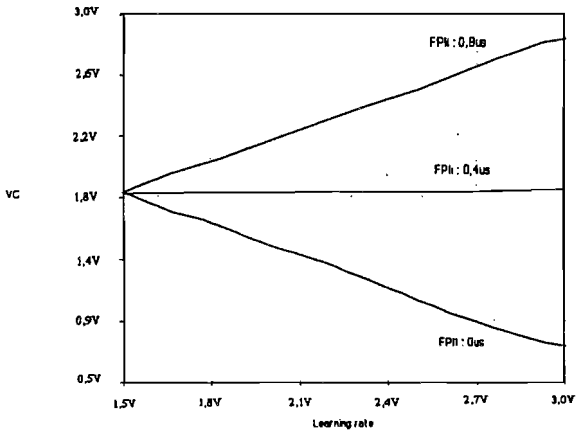


Fig. 5.2.3.3 The multiplication graphs produced by multiplier N1

Similar to the multiplier S2, this circuit introduces an offset of 2 to 3% and has a low power consumption. This multiplier has a good linearity.
The two kinds of converters present in the neuron unit for hidden layers are similar to the ones present in the output neuron unit, which is why they will be briefly reviewed in the next chapter.

6 The Output Neuron Unit

In the backpropagation training algorithm presented in detail in paragraph 1.3, the error between the actual output and the desired output is calculated in step 4. In the hardware implementations of the backpropagation algorithm proposed in the literature , the error can be calculated either on chip or off chip. On chip, this can be achieved by a logic gate (digital ANN), whereas with off chip this can be achieved by a set of resistances (Analog ANN). The idea is to incorporate the computation of the error part to the neuron chip. A special output neuron unit is therefore developed. This output neuron unit differs from the neuron unit placed in the hidden layers since it incorporates the error generation circuit and has different output signals representations. By doing it this way the implementation area and power consumption of the output neuron unit is similar to the ones of the normal neuron unit. Figure 6.1 shows the internal organization of the output neuron unit.

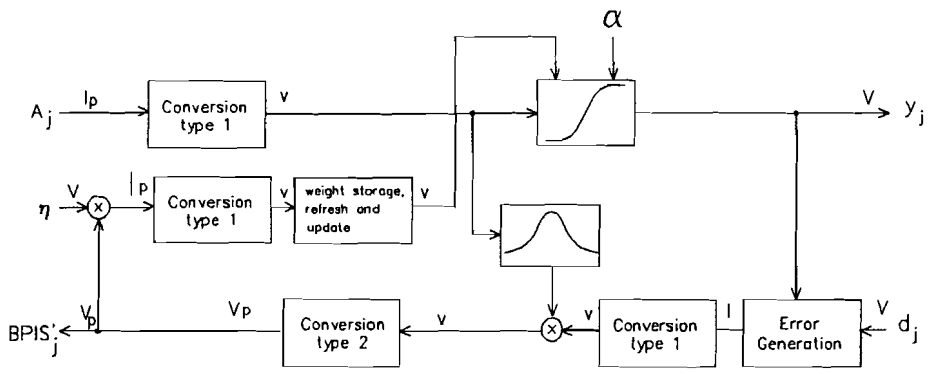


Fig. 6.1 Output neuron unit

In this case, the continuous output voltage of the sigmoid unit can be connected to the error calculation stage. The interfacing board developed in the same department [30] has several continuous inputs and therefore the output signal (d_j)is not pulsed any more. So the converter type 2 taking place in the normal neuron unit (fig.5.1.1), can be suppressed.

6.1 Error Generation Stage

The functional specifications of this stage are listed below

- input #1: y_j continuous voltage, positive value
- input #2: d_j continuous voltage, positive value
- output : continuous output current, positive / negative values

The working principle of this stage is to perform the current difference Iout of two currents I1 and I2. The two inputs voltage V1 and V2 can be converted into the two currents I1 and I2,

respectively, by means of two VCCs. This can easily be implemented by using a differential pair of transistors presented in figure 6.1.1.

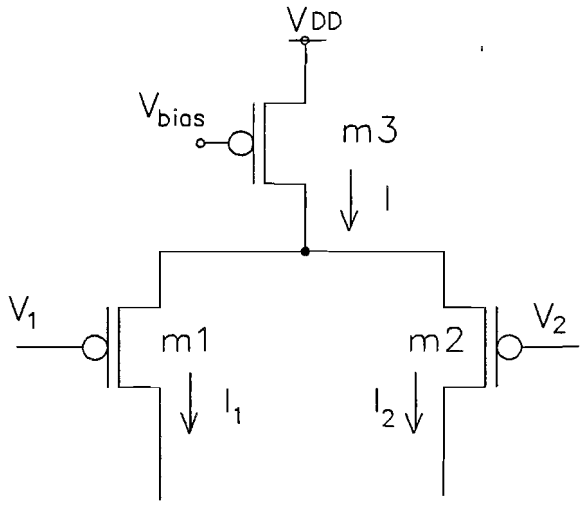


Fig. 6.1.1. Principle of error generation

All the transistors work in saturation region and therefore the following (simplified) relation occurs between the current Iout and the voltage difference VD=V1-V2

$$I_{out} = I_2 - I_1 = \mu C_{ox}\frac{W}{L}V_D\left(\frac{4I}{\beta} - V_D^2\right) \quad \text{where} \quad \beta = \mu_0 C_{ox}\frac{W}{L} \tag{6.1.1}$$

From equation (6.1.1), a linear relation can exist between the output current and the voltage difference VD, if the following condition is taken into consideration:

$$V_D \leq \sqrt{\frac{4I}{\beta}} \tag{6.1.2}$$

By connecting a current mirror as load to the previous differential pair of transistors (see fig. 6.1.2), the current difference can be performed. With a large bias current I, the condition (6.1.2) is met but in that case a large capacitor C_{int} is required. By decreasing the bias current, a smaller size capacitor can be used, but then the transistors m1 and m2 have to be very long, see condition (6.1.2).

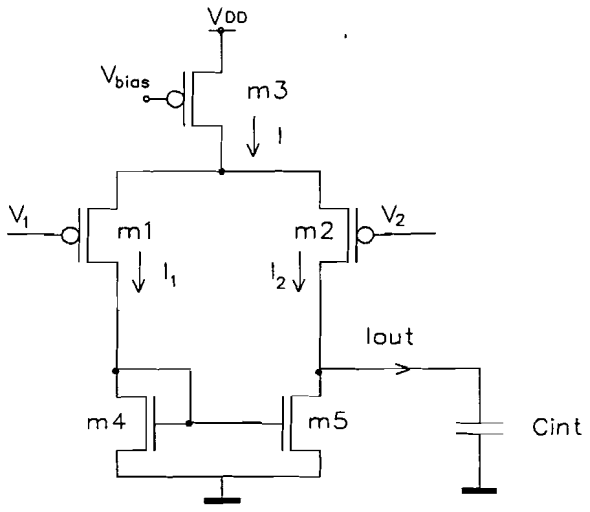


Fig. 6.1.2 error generation circuit

By scaling the currents I1 and I2, the resulting output current Iout can be lowered and a smaller capacitor can be used. Figure 6.1.3 shows the improved previous circuit.

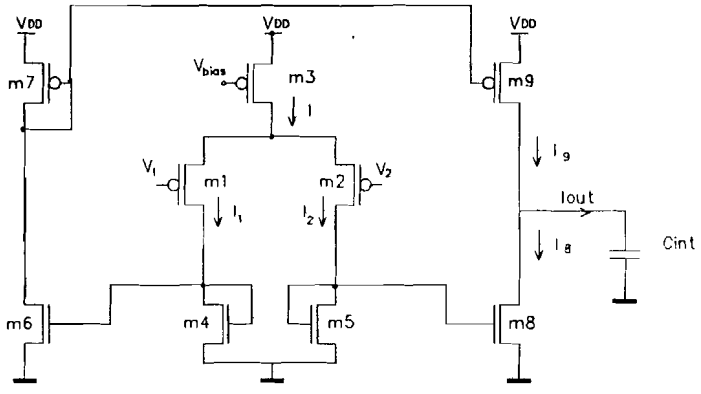


Fig. 6.1.3 Improved circuit

The resulting output current is the difference of the scaled and mirrored currents I7 and I6. An output voltage range of 1V with a reference voltage of 2.5V can be obtained for a capacitor C_{int}=0.6pF and a bias current I=5.5uA. The relation between the input and output current of a current mirror shows that variation of the drain-source voltages of both transistors affect the scaling factor.

$$\frac{I_1}{I_2} = \frac{W_1}{L_1}\frac{L_2}{W_2}\frac{(1+\lambda_1 V_{DS_1})}{(1+\lambda_2 V_{DS_2})} \qquad (6.1.3)$$

where V_{DSi} is the drain source voltage and λ_i is the channel length modulation
To prevent this variation, the current mirror (m7, m9) in the circuit of the figure 6.1.3 is replaced by a cascode current mirror m7, m71, m9 and m91. Two other cascode transistors (m61 and m81) are also added. The improved version of the error generation circuit is printed in figure 6.1.4

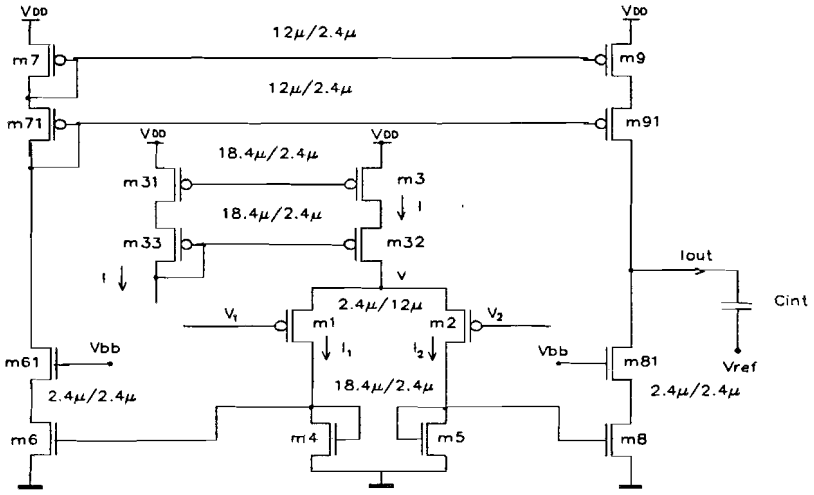


Fig. 6.1.4 Error generation circuit

Figure 6.1.5 shows the variation of the output current Iout for variations of the input voltage V1 (output sigmoid) between 0.2 and 1V and variations of the input voltage V2 (desired output) in the same range. The results of the Monte-Carlo analysis, which was conducted for variation of the threshold voltage, are shown figure 6.1.6

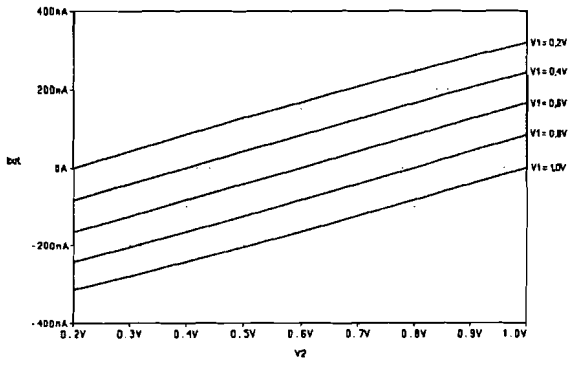


Fig. 6.1.5 Variations of the output current

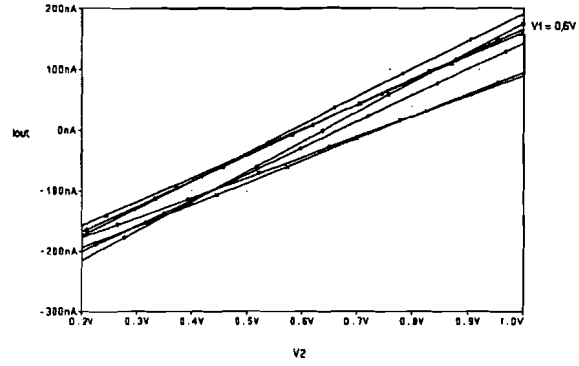


Fig. 6.1.6. Monte-Carlo results

The error calculation stage proposed above has a low power consumption (200uW). The desired output (see the specifications of the interfacing board in [30]) has a dynamic range of minimum 2.5V. The two inputs of the error calculation have to have the same dynamic range, therefore a buffer with a gain equal to 0.3 is used to reduce the input voltage range of the desired output of 2.5V to 0.8V. Figure 6.1.7.a shows the buffer circuit with the size of the transistors. Its transfer characteristic is printed in figure 6.1.7.b. The Monte-Carlo analysis has shown that this circuit is less sensitive to voltage threshold variations.

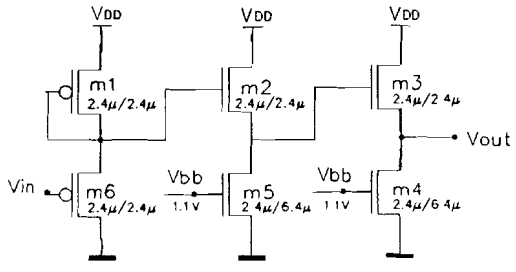


Fig. 6.1.7.a Buffer

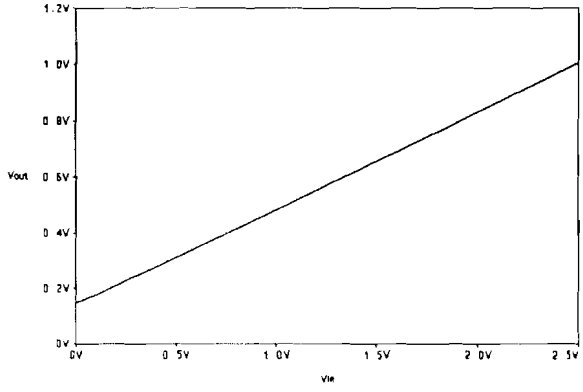


Fig. 6.1.7.b Transfer Characteristic

The error generation circuit has an offset of about 3%, when there is mismatching between the transistors. It decreases the performances of the training algorithm because the error calculated can not be lower than a minimum. A way to reduce the influence of this offset will be to increase

the gain of the error generation circuit. In that case the relation between the voltage difference (V1-V2) and the output current Iout will be not linear any more. This solution has to be investigated. It may be also a way to reduce the influence of all the offsets introduced by the different dynamic multipliers present in the backward path.

6.2 Converters

Two kind of converters are present in the feedback path of the synapse and neuron unit. They also introduce some non idealities which can affect the performances of the training algorithm.

6.2.1 Wave form driven Conversion

This converter type 2, used in the feedback path of both neuron units is based on the binary comparison between an analogue input signal Vi and a triangular wave Vr synchronous with a master clock CCK. The width of the output pulse is proportional to the input amplitude:

$$T_i = T_{max} \frac{V_i - V_{min}}{V_{max} - V_{min}} \qquad (6.2.1.1)$$

where Vmin and Vmax are the lowest as well as the highest values of the voltage Vr, within the duration of the active phase. The input dynamic range can easily be adapted to the requirements by tuning the amplitude of the triangular wave. The only limitations are due to input offset, hysteresis and a conversion delay, which do not otherwise affect the converter transfer function. Figure 6.2.1.a shows the circuit of this converter, while figure 6.2.1.b shows the timing of the different signals.

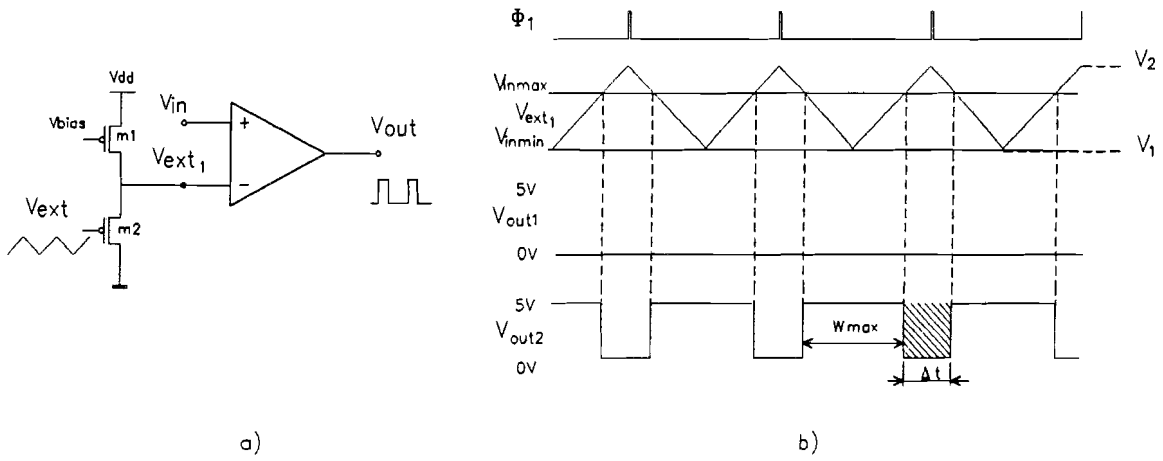


Fig. 6.2.1.1 a) Converter Type 2 b) Timing of the signals

The triangular signal used in [12] has a Vmax=1.5 and Vmin=0. In order to do not increase the number of pins per chip, the same triangular wave is just shifted to be used in the feedbackward

path. The two minimum sized transitors used to realise this level shifter are shown in figure 6.2.1.1.a. The triangular signal is shifted by 1.6V. In this way a minimum value of the input voltage Vin corresponds to a minimum pulse width of 0us for the output pulsed voltage Vout. The second approach of dynamic multiplier S2 (paragraph 4.2.2) and N1 (paragraph 5.2.3) is based on the fact that the minimum width of the CPWM signals is 0us. The period of the master clock is 1.25us, but the maximum active period of the CPWM signals is 0.8us with an idle time of 0.45us. This time is required to control all the switches in the sample and hold circuits and to synchronize all the signals. Further this converter is the same as the one designed for the feedforward path and the characteristics of the op-amp used in the converter can be found in paragraph 5.2.5 in [6].

6.2.2 Capacitor Charging Conversion

The converter type 1 is present on different places in the feedbackward path. A capacitor Ct is charged with a constant current Iz for the entire duration of the input pulse (for CPWM). At the end of the active phase, the capacitor voltage is sampled on the capacitor Cs and transferred to the output. Ct is shunted during the idle phase, and loaded in the active phase. In practice, to ensure a good conversion, sampling and discharging are two consecutive and non overlapping subphases of the idle phase (see figure 6.2.2.1.b). The transfer function of the converter can be expressed as follows:

$$V_0 = V_{ref} + \frac{I_Z T_i}{C_T} \qquad (6.2.2.1)$$

where T_i is the time of integration and V_{ref} is the reference voltage usually equal to 2.5V.
Figure 6.2.2.1.a shows the converter type 1 and figure 6.2.2.1.b. shows the timing of the signals.

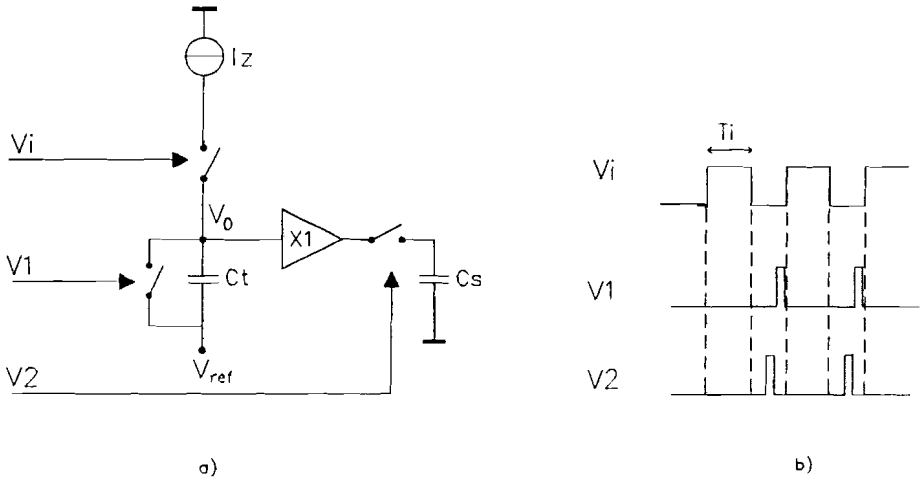


Fig. 6.2.2.1 a) Converter type 1 b) Timing diagrams

Depending of the place where converter type 1 is placed, the current Iz can be:

- a continuous current (error generation circuit)
- a single pulsed current (dynamic multipliers S1 and N1)
- a sum of pulsed currents (input of the neuron unit (BPI_i))

This type of conversion suffers from several problems. The converter's sensitivity depends heavily on matching between a current and a capacitor which is poor for typical silicon VLSI technology.

An operational amplifier configurated as a voltage follower is used as a buffer in the circuit of the figure 6.2.2.1.a. The characteristics of this circuit can be found in paragraph 5.2.3 of [12]. One attractive aspect of this conversion is its very low power consumption. The signals of sampling (V2) and discharging (V1) are synchronized with master clock and are active during the idle phase (0.45us) of the CPWM signal (Vi).

7 Backward Path and Simulation Results

The different circuits consisting of the synapse unit and the two types of neuron units have been presented in detail in the three previous chapters. The implementation of the backpropagation algorithm has been done by using one of the pulse stream techniques - the Coherent Pulse Width Modulation. The complete feedback path counts a lot of circuits (see fig. 4.1, 5.1 and 6.1). Therefore two different circuits have been simulated. The first one corresponds to a neuron connected with one synape, while the second circuit corresponds to the output neuron unit.

7.1 Normal Neuron Unit with one synapse Unit

One synapse has been connected to a normal neuron. Figure 7.1.1 shows the block scheme of the backward path. Figure 7.1.2 shows the simulation of multiplier S1. The input pulsed voltage FPI_i is active at low level and has a constant pulse width of 0.8us while the continuous input voltage UD_j has three different values. The voltage $\Delta_S w_{ji}$ follows the variations of the voltage UD_j with a delay of 1,25us. This delay corresponds to the conversion time of converter type 1. Figure 7.1.3 shows the simulation of multiplier N1. In that case the input pulsed voltage $BPIS'_j$ is active at high level and has a constant pulse width of 0.8us, while the continuous input voltage η changes from its minimum to its maximum. It results that the voltage $\Delta_N w_{ji}$ changes also from its minimum to its maximum with a delay of 1,25us (converter type 1).

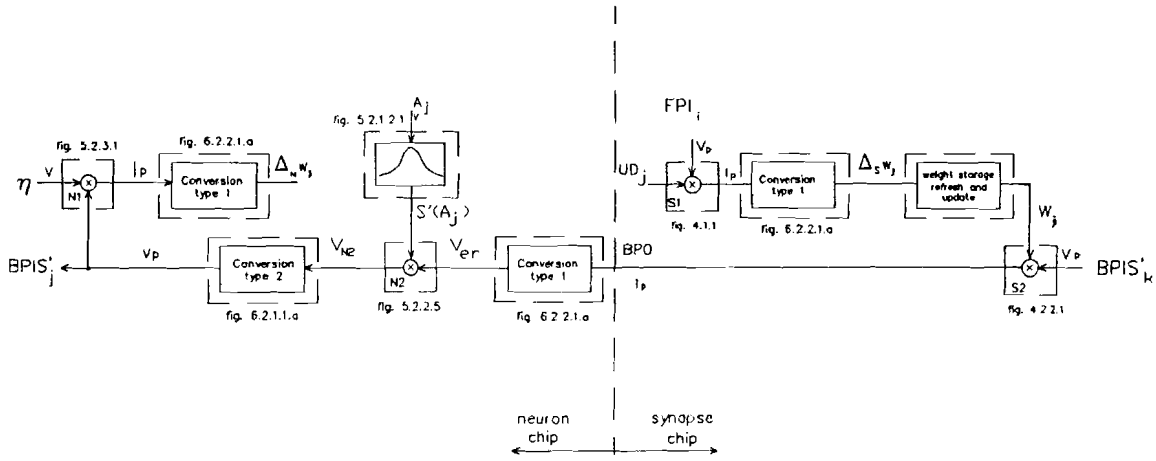
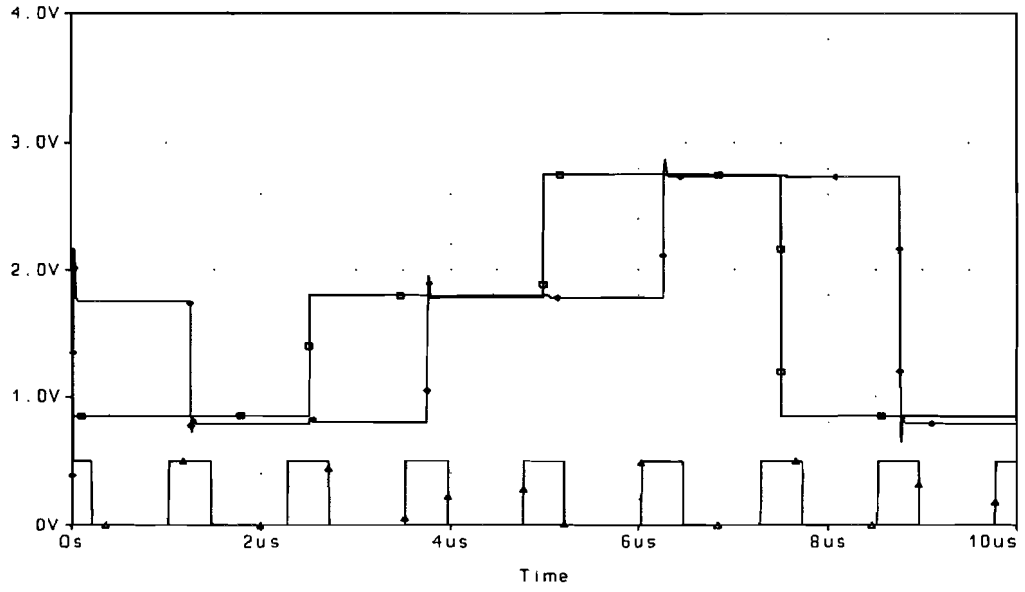
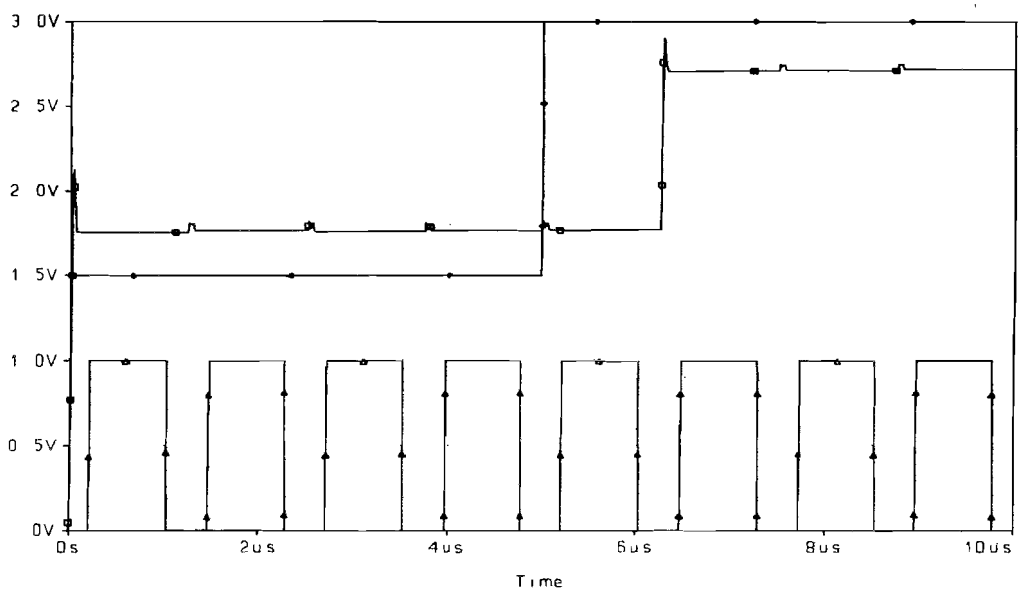


Fig. 7.1.1 Block scheme of the backward path



$\square = \mathrm{UD_j}$ $\quad \diamond = \Delta_s w_{ji}$ $\quad \triangle = \dfrac{FPI_i}{10}$

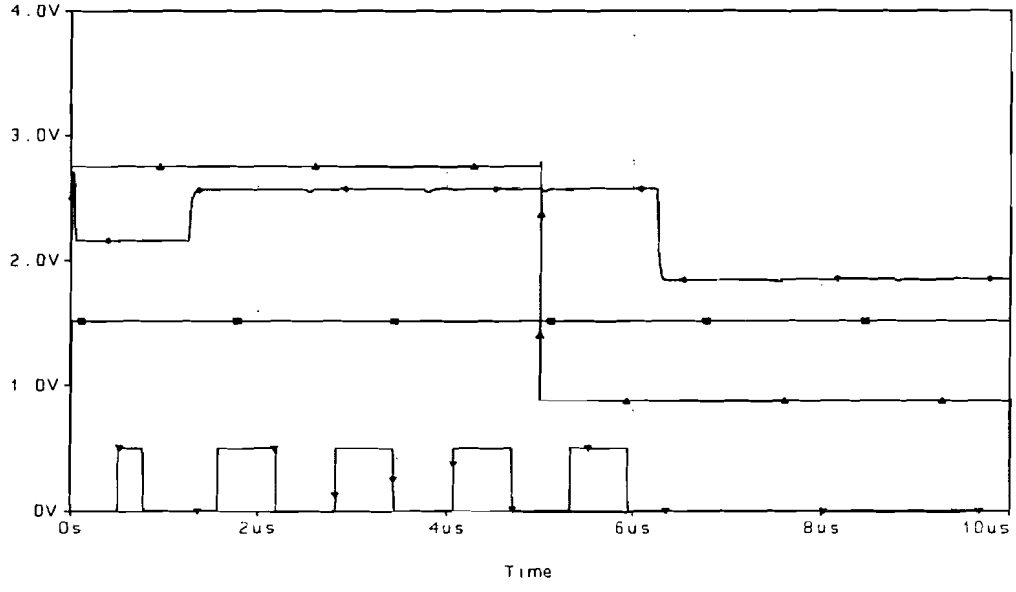
Fig. 7.1.2 PSPICE simulation of the multiplier S1



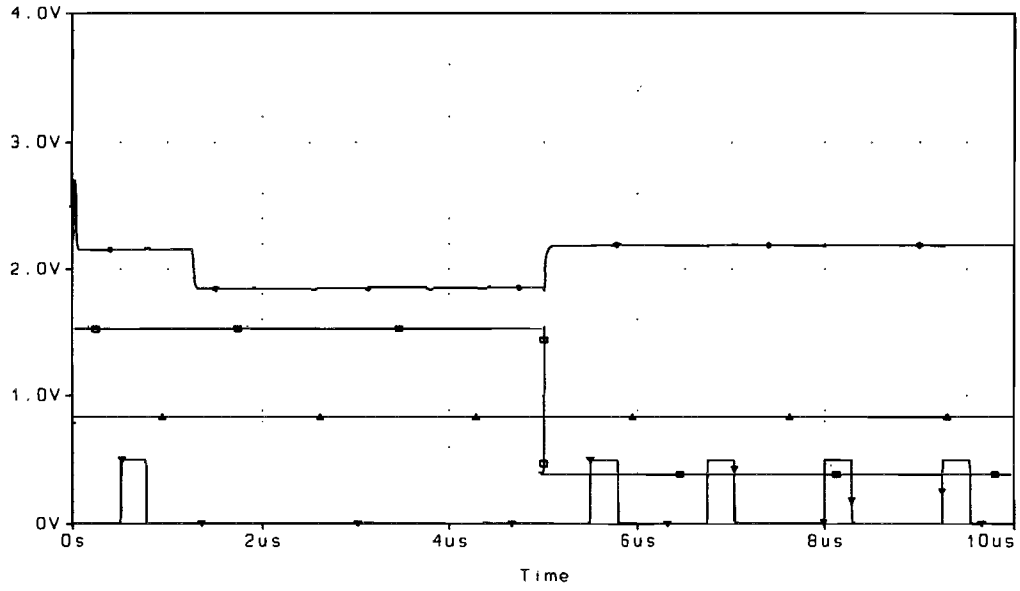
$\square = \Delta_N w_{ji}$ $\quad \diamond = \eta$ learning rate $\quad \triangle = \dfrac{BPIS_j'}{5}$

Fig. 7.1.3 PSPICE simulation of the multiplier N1

Figure 7.1.4.a and 7.1.4.b show the simulations of the complete backward path. In figure 7.1.4.a, the synaptic weight value w_{ji} changes from its maximum to its minimum; the width of pulses of signal $BPIS'_k$ is 0.8us. The derivative of the sigmoid $S'(A_j)$ is at its maximum level. The voltage V_{er} follows the variations of w_{ji} with a delay of 1.25us (conversion type 1). The width of pulses of signal $BPIS'_j$ is 0.8us when the voltage w_{ji} is maximum. When the voltage w_{ji} is minimum then the width of the pulses of the signal $BPIS'_j$ is 0us.
In figure 7.1.4.b, the synaptic weight value w_{ji} is at its minimum while the width of the pulses of the signal BPIS' is 0.8us. The derivative of the sigmoid changes from its maximum to its minimum. The voltage V_{er} follows the variations of those of the derivative of the sigmoid. The width of pulses of the signal $BPIS'_j$ is 0us when the voltage $S'(A_j)$ is maximum. When the voltage $S'(A_j)$ is minimum then the width of the pulses of the signal $BPIS'_j$ is 0.4us, which is the reference pulse width.



a) $\square = S'(A_j)$ $\diamond = V_{er}$ $\Delta = w_{ji}$ $\nabla = \frac{BPIS'_j}{10}$



b) □ = $S'(A_j)$ ◇ = V_{er} △ = w_{ji} ▽ = $\frac{BPIS'_j}{10}$

Fig. 7.1.4.a,b PSPICE simulations of the complete backward path

7.2 Output Neuron Unit

Figure 7.2.1 shows the block scheme of the backward path of the output neuron unit. PSPICE simulation are printed in figure 7.2.2.a and 7.2.2.b.
In figure 7.2.2.a, the desired output d_j is at its minimum and the derivative of the sigmoid function S'(A_j) is at its maximum level. When the actual output y_j is at its minimum level then the width of the pulses of the signal BPIS'$_j$ is 0.4us. The width of the pulses of BPIS'$_j$ is 0us when the actual output y_j is at its maximum level.
In figure 7.2.2.b, the actual output y_j is at its average level while the desired output d_j changes from its minimum to its maximum. The derivative of the sigmoid function is at its maximum.
The width of the pulses of the signal BPIS'$_j$ is either small or large considering that the desired output d_j is at its minimum or maximum level.

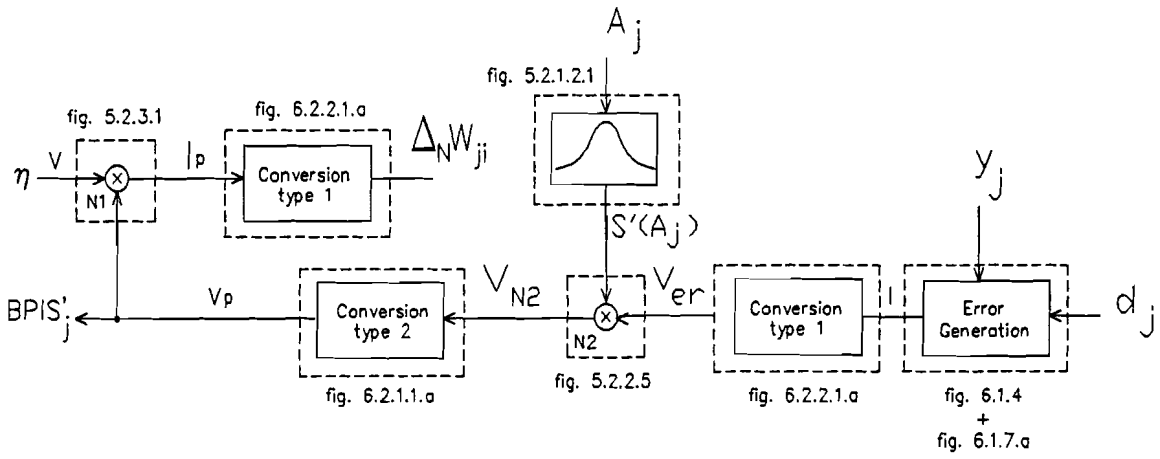
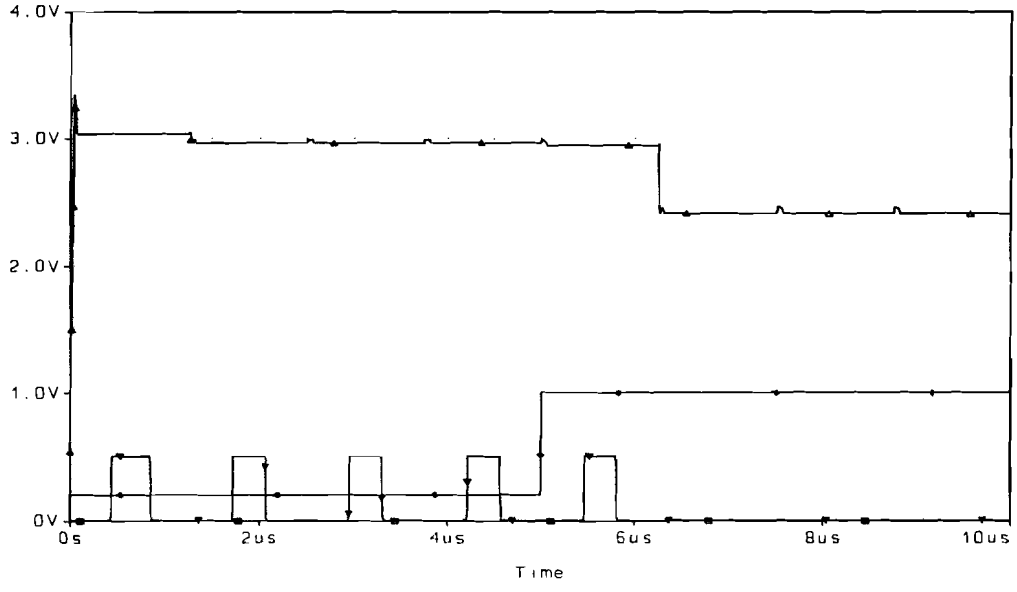
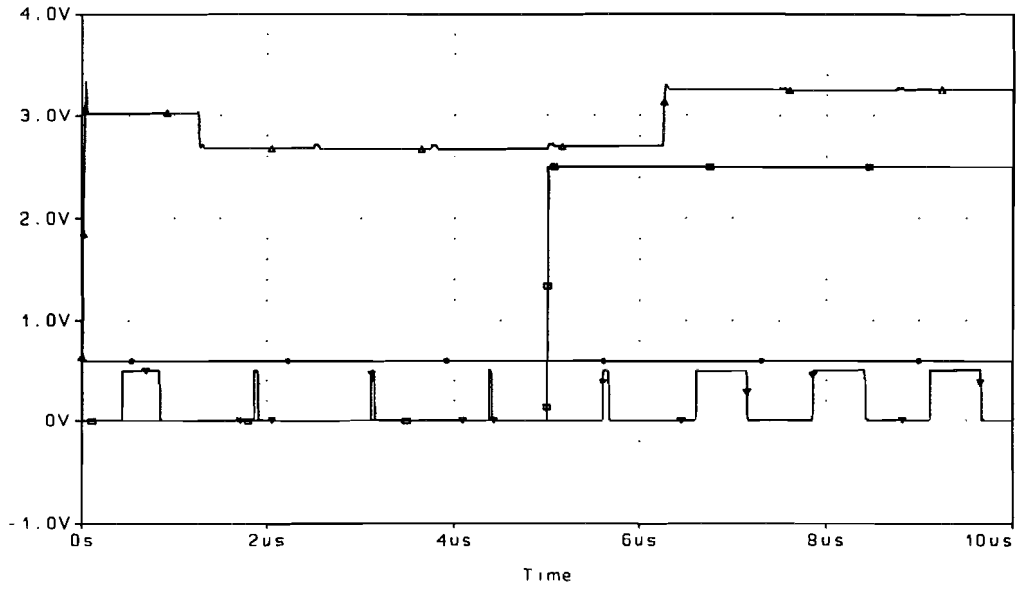


Fig. 7.2.1 Block scheme of the backward path of the output neuron unit



a) □ = d_j desired output ◇ = y_j actual output △ = V_{er} ▽ = $\frac{BPIS_j'}{10}$



b) □ = d_j desired output ◇ = y_j actual output △ = V_{er} ▽ = $\frac{BPIS_j'}{10}$

Fig. 7.2.2.a,b PSPICE simulations of the backward path

The circuit used to obtain the derivative of the sigmoid function is presented in paragraph 5.1 with its PSPICE simulations. To keep the figures above simple, the signals of the circuit of the derivative of the sigmoid have not been printed.

The characteristics of both synapse and neuron unit are listed below.

Synapse:

- Weight 0.85V - 2.75V
- UD_j 0.85V - 2.75V
- FPI_i 0V - 5V / 0.15us - 0.8us
- BPIS' 0V - 5V / 0us - 0.8us
- BPO -200nA..+200nA / 0us - 0.8us

- power dissipation 150uW
- response time min 1.25us
  max 2.5us

Normal Neuron Unit:

- Weight	0.85V - 2.75V
- BPI_i	0V - 5V / 0us - 0.8us
- $BPIS'_j$	0V - 5V / 0us - 0.8us
- FPO_j	0V - 5V / 0.15us - 0.8us
- η	1.5V - 3V
- power dissipation	758uW

Output Neuron Unit:

- Weight	0.85V - 2.75V
- d_j	0V - 2.5V
- $BPIS'_j$	0V - 5V / 0us - 0.8us
- y_j	0.2V - 0.8V
- η	1.5V - 3V
- power dissipation	865uW

For both neuron units the response time is:

min 2.5us
max 3.75us

8 Conclusions and Recommendations

Although it seems that the CPWM technique offers acceptable results considering the power dissipation and chip area, it offers a limited precision. Especially the dynamic multipliers have a limited precision due to some offsets.

It appears that the BPA is very sensitive to a number of non idealities introduced by the hardware, the convergence speed and the miminum obtainable error are affected.

Mismatching between the transistors increases mainly the offsets introduced by the dynamic multipliers. These dynamic multipliers use the difference of two currents to achieve positive and negative values. Application of a symmetrical power supply might be a way to reduce this mismatch. Simulations done on the ANN simulator of the departement have shown that these offsets have to be (extremelly) small.

The rest of the feedback path introduces also some non negligible offsets. The influence of those offsets on the BPA should be reduced by investigating one of the ideas proposed below:

- implementing an improved version of the BPA using less multipliers.

- controlling the gain of the error signal in each neuron chip to keep the error amplitude larger than the offsets introduced by the hardware. (it leads to strong requirements concerning the multiplier N1 and the weight update part).

- using the backward path to propagate back an error signal equal to zero which can be used as a reference. This can be done during the time that the signal is propagating through the network in the forward path. The corresponding offset can be saved into a capacitor and then be substracted from the normal error signal (with offset). This approach leads to a serious increase of complexity of the hardware.

In this work, it has been considered that the output currents of the synapses are integrated on a single capacitor. This is an ideal case. Also the influence of parasitic capacitors has to be considered. Therefore investigations should be done to resolve this important problem. Furthermore the limits of the capacitor voltage are those of the power supply i.e. 0-5V. Therefore it should be kept within a certain range (minimum - maximum) for the proper functioning of the sigmoid circuit.

The auto-scaling function (1/ N, N=number of synapses connected to a neuron) proposed in [6] could be used if a multiplication by $\sqrt{N}$ is insert in the forward path to performe the required scaling function $1/\sqrt{N}$.

It can be concluded that the CPWM technique could be used to implement the backpropagation learning algorithm. But since it has a limited precision, simulations should be done to determine which precision is required.

No real comparison with implementations on chip of the BPA using the CPWM technique is possible. Although several groups in Europe are working on this subject, no results of this topic have yet been published in the literature.

Bibliography

[1] **Alippi C., Nigri M.E.**
'Hardware Requirements for Digital VLSI Implementation of Neural Networks'
IEEE Int. Joint Conf. on Neural Networks, Singapore, Nov. 1991, pp. 1873-1878

[2] **Annema A.J. et al.**
'Cross Talk Effects in Neural Networks'
Proc. 2nd Int. Conf. on Microelectronics for Neural Networks, 1990, pp. 35-45

[3] **Bibyk S. et al.**
'Current-Mode Neural Network Building Blocks for Analog MOS VLSI'
IEEE Int. Symposium on Circuit and System, May 1990, pp. 3283-3285

[4] **Botha T.**
'CMOS Analogue Current-Steering Multiplier'
Electronics Letters, Vol. 28, No. 28, 1992, pp. 525-526

[5] **Bult K., Wallinga H.**
'A CMOS Four quadrant Analog Multiplier'
ESSCIRC 1985, 11th Solid-State Circuits Conf., Toulouse, Vol. 21, No. 3, pp. 430-435

[6] **Claasen-Vujčić T.**
'Implementation of a Multi-Layer Perceptron Using Pulse Stream Techniques'
Master Thesis, TU Eindhoven, Febr. 1993

[7] **Del Corso D., Reyneri L.M.**
'Mixing Analog and Digital Techniques for Silicon Neural Networks'
ISCAS, 1990, pp. 2446-2449

[8] **Del Corso D.**
'Hardware Implementations of Artificial Neural Networks'
'New Trends in Neural Computation', IWANN 1993, Springer-Verlag, pp. 405-417

[9] **Eguchi H. et al.**
'Neural Network Hardware with Learning Function Using Pulse-Density Modulation'
Electronics an Communications in Japan 1991, Part 2, Vol. 74, No. 11, pp. 66-74

[10] **Frye R.C. et Al.**
'Back-Propagation Learning and non idealities in Analog Neural Network Hardware.
IEEE Trans. on Neural Networks, 1991, Vol. 2, No. 1, pp. 110-117

[11] **Furman B., Abidi A.A.**
'An Analog CMOS Backward Error-Propagation LSI'
22nd Asilomar Conf. on Signals, Systems and Computers, Vol. 2, 1989, pp. 645-648

[12] **Herman A.**
'Implementation and Performance of an Analog Non-Volatile Neural Network'
Intel Report

[13] **Hollis P.W. et al.**
'The Effects of Precision Constraints in a Backpropagation Learning Network'
Neural Computation 1990, Vol. 2, pp. 363-373

[14] **Ismail I.**
'Four-Transistor Continuous -Time MOS Transconductor'
Electronics Letters, Vol. 23, No. 20, 1987, pp. 1099-1100

[15] **Kuipers M.**
'Electronic Implementation of Multilayer Neural Networks Including Back Propagation Training Algorithm'
Master Thesis, TU Eindhoven, April 1992

[16] **Lehman T.**
'A Cascadable Chip Set for ANNs with On-Chip Back-Propagation'
Proc. 3rd Int. Conf. on Microelectronics for Neural Networks, Edinburgh 1933, pp. 149-158

[17] **Lippman R.P.**
'An Introduction to Computing with Neural Nets'
IEEE ASSP-Magazine, 1987, pp. 4-22

[18] **Khachab N.I. et al.**
'A simple all MOS continuous-time multiplier/divider cell and its applications in VLSI signal processing'
Electronics Letters, Vol. 25, No.23 , Nov. 1989, pp. 1001-1004

[19] **Murray A.F., Smith A.V.W.**
'Asynchronous VLSI Neural Networks Using Pulse-Stream Arithmetic'
IEEE Journal of Solid-State Circuits, 1988, Vol. 23, No. 3, pp. 688-697

[20] **Ngolediage J.E, et al.**
'CMOS Phase Detector and Four Quadrant Multiplier for Implementation in Analogue Neural Networks'
Electronics Letters, 1992, Vol. 28, No. 12, pp. 1142-1143

[21] **Reyneri L.M., Filippi E.**
'Modified Backpropagation Algorithm for Fast Learning in Neural Networks'
Electronics Letters 1990, Vol. 26, No. 19, pp. 1564-1566

[22] **Reyneri L.M., et al.**
'An Analysis on the Performance of Silicon Implementations of Backpropagation Algorithms for Artificial Neural Networks'
IEEE Trans. on Computers 1991, Vol. 40, No. 12, pp. 1380-1389

[23] **Reyneri L.M., Sartori M.**
'A Neural Vector Matrix Multiplier Using Pulse Width Modulation Techniques'
Proc. 2nd Int. Conf. on Microelectronics for Neural Networks, Munich, Oct. 1991, pp. 269-272

[24] **Reyneri L.M.**
'Interfacing Sensors and Actuators To CPWM and CPEM Neural Networks'
Proc. 3rd Int. Conf. on Microelectronics for Neural Networks, Edinburgh 1993, pp. 11-19

[25] **Rumelhart D.E., McCelland J.L.**
'Parallel Distributed Processing: Exploration in the Microstructure of Cognition'
Foundations Vol. 1, Cambridge MA:MIT Press 1986

[26] **Salam F.M.A. et al.**
'An Analog MOS Implmentation of the Synaptic Weights for Feeforward/Feedback Neural Nets'
Proc. 32nd Midwest Symposium on Circuits and Systems, Vol. 2, Aug. 1990, pp. 1016-1019

[27] **Shima T. et al.**
'Neuro Chips with On-Chip Back-Propagation and/or Hebbian Learning'
IEEE Journal of Solid-State Circuits, Vol. 27, No. 12, Dec. 1992

[28] **Snijders A.G.**
'Low Power Implementation of Multilayer Neural Networks'
Master Thesis, TU Eindhoven, Dec. 1992

[29] **Teulings P.M.W.**
'Analog Electronic Implementation of Multilayer Neural Networks Including Back Propagation Training Algorithm'
Master Thesis, TU Eindhoven, Aug. 1991

[30] **Van Teeffelen J.J.M.**
'Interfacing Neural Chips with a Personal Computer'
Master Thesis, TU Eindhoven, Aug. 1993