Low Power Analog Implementation of Multilayer Neural Networks

Masters thesis of A.G. Snijders

Supervisor: Prof. dr. ir. W.M.G. van Bokhoven
Coach: Dr. ir. J.A. Hegt
At: Electronic Circuit Design Group
Faculty of Electrical Engineering
Eindhoven University of Technology

The Eindhoven University of Technology accepts no responsibility for the contents of theses and reports written by students.
Acknowledgements

I wish to thank all who contributed to my graduation, especially Hans Hegt for coaching.

I would like to thank Tanja Claasen and the whole EEB group for the pleasant social atmosphere.
Abstract

Most of the used neural networks are simulated on a conventional serial computer. Because of the very time consuming training algorithms, it is fruitful to implement neural networks including their training algorithm on-chip. The structure of a feedforward multilayer neural network has already been derived with the possibility of integrating the training algorithm.

One component of the synapse and neuron, the Weights Storage, has been designed yet. But all other components should be designed with a low power dissipation and small area. A comparison has been made between two signal representations (Pulse Stream and Continuous signals), and the continuous representation with differential signals is chosen because of the smaller area and lower crosstalk.

An approach for minimizing the power dissipation was the design in Weak Inversion (or Subthreshold) of the most used component in the neural network: a multiplier. The multiplier, which must have a large dynamic input range, can be implemented in Weak Inversion but the nonlinearity appears to be large. Linearizing circuits can be used but this results in a lot of area. Therefore a multiplier is implemented which works in Strong Inversion, but with relative small currents and smaller area.

The estimated area of the implemented synapse and neuron without implementation of the training algorithm is respectively: 3428μm² and 14000μm². The power dissipation is 3μW for the synapse and 7μW for the neuron.

The synapse and neuron are implemented with differential signals but the number of input/output pins determines the number of synapses or neurons that can be integrated on-chip. This number of synapses or neurons can be doubled by using single signals in stead of differential signals, but this is extra noise sensitive. Multiplexing is an option for decreasing the number of pins.

The designed multiplier is nonlinear with a nonlinearity of 20% FS. System simulation of a neural network showed that this nonlinear multiplier in the forward path affects the convergence speed of the Back Propagation training algorithm but hardly affects the output error after training. However a combination of this nonlinear multiplier and quantized Weights results in a larger output error. The multiplier nonlinearity can be decreased to the disadvantage of more area or power dissipation.

An extendable neural network needs a kind of normalization for the incoming signal of a neuron. The summarized signal of connected synapses must have a certain range independent to the number of synapses.

The used normalization technique is very sensitive to variation in threshold voltage of the MOST so other techniques must be investigated further.
Contents

1 Introduction .................................................. 1

2 Theory Analog Multilayer Perceptron .......................... 2
   2.1 Back Propagation Training Algorithm .................... 4
   2.2 Overview Neural Network with Back Propagation algorithm 7

3 Signal Representation ........................................... 13
   3.1 Continuous signals ..................................... 13
      3.1.1 Continuous Single signals ......................... 14
      3.1.2 Continuous Differential signals ................... 15
   3.2 Pulse-Stream signals .................................. 17
      3.2.1 Pulse Frequency Modulation multiplier ............. 17
      3.2.2 Pulse Width Modulation multiplier ................ 19
   3.3 Comparison Continuous versus Pulse Stream Signals ....... 21
   3.4 Discussion ............................................. 22

4 Low Power Design of Forward Path ............................ 24
   4.1 Structure Synapse and Neuron ........................... 24
   4.2 Subthreshold Operation ................................ 25
   4.3 Multiplier .............................................. 26
      4.3.1 Voltage-to-Current Converter ....................... 31
      4.3.2 Analysis ........................................... 31
      4.3.3 VCC with 3 stages ................................ 34
      4.3.4 PSPICE simulations ................................ 35
      4.3.5 VCC with Non ideal Current sources .............. 38
      4.3.6 Gilbert multiplier with VCC ....................... 41
   4.4 Comparison Gilbert multiplier with and without VCC ....... 46
   4.5 Normalizing: approach for hundreds of synapses .......... 47
      4.5.1 Analysis Normalizing cell .......................... 49
      4.5.2 Implementation Normalizing cell with Non ideal Current Source ................. 50
      4.5.3 Pspice simulations ................................ 51
   4.6 Structure Synapse and Neuron with Normalization .......... 55
   4.7 Sigmoid .................................................. 57
      4.7.1 Implementation ...................................... 57
   4.8 Optimum Transistor Matching ............................ 62

5 Implementation Synapse and Neuron .......................... 63
   5.1 Synapse Implementation .................................. 63
   5.2 Neuron Implementation .................................. 64
   5.3 Two Synapses connected with a Neuron ................... 66
      5.3.1 Static Analysis ..................................... 66
      5.3.2 Transient Analysis ................................ 69
1 Introduction

Neural Networks are used for problem solving with complicated problems which are difficult to program on a computer. For example handwriting recognition and speech-processing.

In a software environment, the training of a neural network is very slow. Therefore a neural network, including the training algorithm, should be integrated on-chip. The proposed neural network is a feedforward multilayer neural network which is built with synapse-layers and neuron-layers. The network topology could be chosen freely by paralleling or cascading synapse-layers and neuron-layers. When a synapse-layer and a neuron-layer are integrated on respectively a synapse-chip and a neuron-chip, then the neural network is extendable.

The signals within synapse and neuron can be represented with Pulse Stream signals or Continuous signals. The representation determines the required area, power dissipation and noise so it is fruitful to investigate these representations further. When the most advantageous representation is chosen, then all components, except for the Weight Storage, have to be designed.
2 Theory Analog Multilayer Perceptron

Neural networks are built with synapses and neurons. The neurons are connected with each other by links with variable weights (synapse). The weight determines the strength of the signal that passes through the link. This structure is derived from the biological nerve-cell. The neurons collect the signals from the incoming links and when the summation of all signals exceeds a threshold, the neuron will generate a signal that is sent to all connected neurons.

Synapses and neurons are built with electronic components to create a similar electrical network. Neural Networks have proven to be successful, for instance in speech- and character recognition. These human-like performances require an enormous amount of processing and Neural nets provide the processing capacity by the large numbers of simple computational elements that work parallel [11].

In figure 2.1 a neuron with some synapses is shown.

![fig. 2.1: Model of neuron with three synapses](image)

The synapse multiplies the input with a weight and the neuron summarises all input signals and passes the result through a non-linearity.

![fig. 2.2: Models of synapse and neuron](image)

In literature [11],[15] three kind of non-linearities are used: step threshold, ramp threshold and a sigmoid (fig. 2.3). For more details about these nonlinearities see [15, chapter 9].
When the Back Propagation algorithm is used for training, a sigmoid function is very convenient (paragraph 2.1). The slope of the sigmoid is not very important: it must saturate for small and for large signals and it must be continuous. With these components different neural network structures can be built. For the on-chip implementation, Multilayer Perceptrons are chosen. These networks are feedforward networks: the output of a neuron is only transported to the next layer. In the next figure a Perceptron with one input layer, one output layer and two hidden layers is shown.

Every layer consist of one or more neurons, and the number of neurons can be different for each layer.
2.1 Back Propagation Training Algorithm

The first training algorithm for Multilayer Perceptrons was the Back Propagation algorithm. Nowadays variants of Back Propagation or faster algorithms are suitable. It is possible to implement the components of the Back Propagation algorithm on-chip, but other training algorithms can also be used if they have proven to be successful and if all used variables are available on-chip.

The neural network must be trained by adjusting the weights within the synapses. First the weights are set at random values and then an input pattern is presented. The calculated output pattern is compared with the desired output pattern, and the difference of these two patterns will be used to readjust all the weights in the whole network. The error propagates through the network from the last layer back to the first. This procedure is followed for every pattern that has to be learned. Finally the goal of the Back Propagation algorithm is to minimize the mean square error between the calculated- and the desired output patterns.

Used variables:
- \( N \): number of inputs of a layer
- \( M \): number of neurons in a layer
- \( x_i \): input \( i \) of a layer
- \( w_{ij} \): 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 \( j \)
- \( \delta_j \): error for neuron \( j \) in output layer
- \( \delta_j^L \): error for neuron in layer \( L \)
- \( \eta \): learning rate
- \( S(\cdot) \): sigmoid threshold function of neuron
- \( S'(\cdot) \): derivative of sigmoid threshold function

The Back Propagation algorithm is described step by step.

1 Initialize weights
Set all weights to small random values.

2 Present inputs and desired outputs
An input vector is presented to the first layer and the desired output vector is presented to the output of the last layer.

3 Calculate outputs
For each layer the output vector is calculated. Start with the first layer. The output of the first layer will propagate to the second layer etc.

Forward Propagation formulas for one layer:

\[
y_j = S(a_j) = S\left(\sum_{i=1}^{N} w_{ij} x_i\right), \quad 1 \leq j \leq M
\]  
\[\text{(eq. 2.1)}\]
4 Adapt weights

The errors are calculated, starting at the output layer. The error of the output layer depends on the difference between the calculated and the desired output vector.

$$\delta_j = S'(a_j)(d_j - y_j)$$ \hspace{1cm} (eq. 2.2)

The errors of all other layers depend on their contribution to the error of the next layer.

$$\delta_j = S'(a_j) \sum_{k=1}^{M} (\delta_{k}^{L+1} w_{jk}^{L+1})$$ \hspace{1cm} (eq. 2.3)

The weights are readjusted according to:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x_i$$ \hspace{1cm} (eq. 2.4)

5 Repeat the procedure by going to step 2

Steps 2 to 4 are repeated until the error \( \sum_{j=1}^{M} (d_j - y_j)^2 \) for all outputs and for each input/output pair is below a threshold [15, chapter 8].

A differential continuous nonlinearity is required for the threshold function. A sigmoid function has the advantage that the derivative is easy to calculate and can be expressed in the function itself.

**fig. 2.5: Sigmoid function**

Mathematic equation for sigmoid and derivative are:

$$y = \frac{1}{1 + e^{-\tau(x-a)}} \hspace{1cm} \tau: \text{temperature factor,} \quad a: \text{offset}$$ \hspace{1cm} (eq. 2.5)

$$\frac{dy}{dx} = \frac{\tau e^{-\tau(x-a)}}{1 + e^{-\tau(x-a)}} = \tau y(1-y)$$ \hspace{1cm} (eq. 2.6)

In figure 2.5 and 2.6 the temperature term \( \tau=1 \). This term determines the maximum steepness of the sigmoid and the value is not very important: the Back Propagation algorithm can compensate this value by scaling the Weights.

The offset term \( a \) must also be adapted by the Back Propagation algorithm. This offset can be implemented by adding an input at each neuron. The input is connected to a synapse...
with a weight equal to the offset \( a \). The synapse input is regarded as an extra neuron with output equal to 1 (fig. 2.7).

Each neuron layer has one extra neuron with output equal to 1. The synapse weights between this extra neuron and the neurons in the layer represent the offsets. Only these weights are shown in the figure.

*fig. 2.7: Offset of sigmoid and derivative is implemented with extra neurons*
2.2 Overview Neural Network with Back Propagation algorithm

The implementation of the Back Propagation algorithm will speed up the training velocity with respect to software implementations. So a backward path is added to the network model in order to adjust all weights [5].

The formulas, used in the Back Propagation algorithm (eq. 2.1 to 2.4), can be translated to signals for the electronic implementation of the network.

The following signals are used:

FPI: Forward Propagating Input \( FPI_i = x_i \)

FPO: Forward Propagating Output \( FPO_j = y_j \)

BPI: Backward Propagating Input \( BPI_j = d_j - y_j \) last layer

\[
BPI_j^L = \sum_{k=1}^{M} (\delta_k^{L+1} w_{jk}^{L+1}) \quad \text{all other layers}
\]

*fig. 2.8: Signals between layers*
BPO: Back Propagation Output

\[ BPO_i^L = \sum_{j=1}^{M} (\delta_j^L w_{ij}) \]

The Back Propagation formulas are rewritten as:

\[ A_j = \sum_{i=1}^{N} w_{ij} FPI_i \]

\[ FPO_j = S(A_j) \]

\[ BPO_i = \sum_{j=1}^{M} (w_{ij} S'(A_j) \cdot BPI_j) \]

\[ \Delta w_{ij} = \eta \cdot S'(A_j) \cdot BPI_j \cdot FPI_i \]

The number of neurons per layer can be increased by paralleling layer-blocks. However the number of synapses can't be changed.

This can be solved by splitting each layer-block in a synapse part and a neuron part [9]. Figure 2.9 and 2.10 show that many network structures can be made by cascading or paralleling synapse- and neuron sections.

\[ \text{fig. 2.9: Expanding the number of inputs} \]
fig. 2.10: Expanding the number of outputs

The Forward Propagation inputs to the synapses are represented by voltages because they must be distributed to other synapse sections. The inputs to the neurons are currents because currents can easily be summed. The summation occurs automatically outside the neuron. These representations are also chosen for the Backward signals.

The activity \( A \) of the synapse is represented by a current and the modified error term (BPIS' = \( S'(A) \cdot BPI \)) is represented by a voltage. The used signals with their range are summarised. The internal signal \( UD_j \) for adjusting the weight during the learning algorithm is distributed to several synapses and is therefore represented by a voltage (fig. 2.13).

\[
A_{ij} = w_{ij} \cdot FPI_i \quad \text{2 quadrants multiplication, result negative/positive}
\]

\[
A_j = \sum_{i=1}^{N} A_{ij} \quad \text{negative/positive signal}
\]

\[
FPO_j = S(A_j) \quad \text{positive signal}
\]

\[
BPIS_j' = S'(A_j) \cdot BPI_j \quad \text{2 quadrants multiplication, result negative/positive}
\]

\[
UD_j = \eta \cdot BPIS_j' \quad \text{2 quadrants multiplication, result negative/positive}
\]

\[
BPO_{ij} = w_{ij} \cdot BPIS'_j \quad \text{4 quadrants multiplication, result negative/positive}
\]

\[
BPO_i = \sum_{j=1}^{M} BPO_{ij} \quad \text{negative/positive signal}
\]

\[
\Delta w_{ij} = UD_j \cdot FPI_i \quad \text{2 quadrants multiplication, result negative/positive}
\]

In the next figures the internal organization of the synapse- and neuron section are shown [18].
The dotted parts of the figures represent the components, used by the Back Propagation algorithm. If another training algorithm is used, these components may change. All other components are located in the forward path.

In the Forward path only one multiplier is required, but this multiplier can be replaced with a Voltage-to-Current converter.
In figure 2.13 and 2.14 the organization of a synapse chip and a neuron chip are shown. The synapse chip contains a matrix of synapses. The internal signal $U_{D_j}$ is calculated for each column of synapses that is connected to the same neuron in the next layer.

*fig. 2.13: Synapse chip*

In these figures, only the signal lines between synapses and neurons are shown; internal signals for timing and control are not given. In chapter 4 the organization will be further analyzed.
In the next chapter, the signal representation will be discussed.
3 Signal Representation

In the previous chapter, the network building blocks have been explained with the extension of Back Propagation in mind. The building blocks will be designed with analog circuits because of their relative small area [18]. The synapse and the neuron communicate with each other via currents and voltages. The reason for this option is that currents can easily be summed and voltages can easily be distributed. However, there are several ways to transport the information from synapses to neurons and from neurons to the next synapse layer. Each representation influences the building blocks of the synapse and the neuron.

The design criteria (in hierarchial order) are:
- power dissipation
- chip area
- noise, crosstalk
- channel capacitance

In literature, many signal representations are used which are based on pulse stream signals. The information is stored in the modulation form of the signal [13],[14]. The following arguments are used for supporting this representation: small area and low power dissipation. Another argument is based on the working of the human brain itself; the neurons in the human brain communicate by firing pulses to other neurons. Other articles deal with analog continuous signals, but good arguments are not given [8],[16].

In this chapter, the signal representations, which are used in literature, are described and the advantages and disadvantages are examined. In the end the best representation with respect to the criteria will be chosen to implement. A synapse and neuron circuit will be presented for each representation. The components of the Back Propagation path are given in the dotted area.

3.1 Continuous signals

The most important building block that has to be designed is the multiplier. This is the main element of the synapse and plays an important role in the chip area and the power dissipation. In a network with n neurons, the amount of synapses is of order n² so the synapse-area must be minimized. With continuous signals, the multiplier can be designed with a transconductance multiplier [18], (figure 3.1). It’s a very simple element with only two transistors. The resulting differential current can also be converted to a single current with a current mirror. The voltage V1 can be implemented single or differentially. Both approaches are described below.
3.1.1 Continuous Single signals

Figure 3.2 shows the elements used for the realization of the synapse. The multiplier that is used is a transconductance multiplier (figure 3.1). The gate of the second transistor is connected to a reference voltage. The Back Propagation algorithm is also implemented. The components used by the Back Propagation algorithm are located in the dotted area. These components are only required when the Back Propagation algorithm is implemented on-chip.

The multiplier that is connected to the Weight is the transconductance multiplier (fig. 3.1). One gate is connected to a reference voltage to obtain a differential voltage. The neuron is shown in the next figure.
fig. 3.3: Neuron with single continuous signals

Area
In the forward and in the backward path, a differential-to-single ended converter must be used. This converter can be implemented with a current mirror, consisting two transistors. In the Back Propagation path of the neuron a converter will be used for converting a differential current to a differential voltage [18, converter nr.1]. Also a differential current-to-single voltage converter must be used. Many converters must be used in this representation to switch from differential to single signals.

Power Dissipation
The multiplier works in saturation with large currents (\( \geq 50 \mu A \)). The input voltage range is large (1-5V). The currents are large and continuous so the power dissipation is also large.

Noise
Noise on the supply voltage can influence the signals. However, it is difficult to predict the effect of noise on the total network. The noise power is larger in baseband signals than in modulated signals, except for Amplitude Modulated signals [4, pp.362-369],[13].

3.1.2 Continuous Differential signals

Fewer converters are used in this approach because all signals are differential. It's an efficient design because the multipliers also work on differential input voltages, and produce differential output currents (fig. 3.4 and 3.5).
In the Backward path a 4 quadrant multiplier is required. This multiplier could be a Gilbert multiplier. Only one converter is required.

In figure 3.5, the neuron is shown.

In these configurations, the transconductance multiplier is used. The multiplier in the forward path works like a voltage-to-current converter.

**Area**
Less converters are used, but all signals are transported through double signal lines. It is hard to predict the total amount of chip area occupied by the double signal lines.

**Power Dissipation**
The power dissipation will be comparable with the previous approach.

**Noise**
Because of the differential signals, common signal disturbances and jitter on the supply voltage will have little effect on the information that is transported [23],[24]. Fewer converters in the synapse also means less additional noise and fewer conversion errors.
3.2 Pulse-Stream signals

There are several modulation forms:
- Pulse Amplitude Modulation
- Pulse Phase (or Position) Modulation
- Pulse Frequency (or Rate) Modulation
- Pulse Width Modulation, or mixtures of these modulation techniques.

Pulse Amplitude Modulation is very sensitive to noise [4] and is not very satisfactory in Neural Networks [13].

Pulse Phase- and Pulse Width Modulation encode information in the time domain, and are equivalent with Pulse Frequency Modulation. These modulation forms use the benefits of a binary coded signal with respect to noise.

With Pulse Phase Modulation two signals are used. The information is stored in the phase difference between two pulsed waveforms [13]. Two signal lines must be used and one can be used as a global reference signal. A drawback of this method is the phase skew, especially when long signal lines are used to distribute the (reference) signals over the chip. Because of the double signal lines and the sensitivity to phase skew, this method is not searched further.

Two of these techniques (Pulse Frequency- and Pulse Width Modulation) are compared. In the next paragraphs, the occupied area and the power dissipation of the synapse and the neuron are tried to estimate. Therefore the transistors are assumed to have minimum sizes, and the area is determined by counting the number of transistors. Although it is hard to predict the required area and the power dissipation in this premature stage, its useful to look deeper at the several signal representations.

3.2.1 Pulse Frequency Modulation multiplier

In this approach, Switched Capacitor multipliers can be used [9]. These multipliers are very small and very linear. The pulsed output current depends on the input voltage and the frequency of the pulsed voltage.

\[
\begin{array}{c}
\text{Vin1} \\
\text{Vin2} \\
\text{Vin2} \\
\text{-Vin2} \\
\end{array}
\]

\[
\begin{array}{c}
\text{Vout} \\
\text{lout} \\
\end{array}
\]

\[\text{fig. 3.6: Switched Capacitor multiplier}\]

The neuron must convert this pulsed current into a constant voltage. To obtain a constant voltage the current pulses must be averaged in time and be converted to a voltage. This
can be done by integrating the currents. However the integrating time must be longer than the longest period time of the pulsed output current, and can't be too long otherwise the output voltage will saturate to the supply voltage. Another method to obtain a voltage is to load a capacitor. The time constant of this low pass filter is also critical.

Area
The SC-multiplier is very small. The synapse can be very small too. Two pulsed signals must be transported to the SC-multiplier which adds extra area.

The neuron contains large area consuming components like an integrator and a voltage-to-frequency converter.

Power Dissipation
The power dissipation in the synapse is very small, but the neuron consumes more power when an integrator and a VCO are used.

Noise
Because of the binary character of the signals, noise will have less influence in comparison to continuous signals. However more crosstalk can be expected between neighbour signal lines because of the high frequency signals with steep rising- and falling edges.

In the next figures, the synapse and neuron based on Pulse Frequency Modulation are visualized.

The SC-multiplier has two pulsed voltage inputs. The phase shift between these two signals is 180 degrees. In figure 3.7 a local inverter is used to obtain the inverted signal.
3.2.2 Pulse Width Modulation multiplier

This modulation type is used by Murray [13],[14]. The used multiplier is a Pulse Width modulation multiplier (figure 3.9). The input of the multiplier is a signal with a variable frequency and a fixed Pulse Width. The Weight modulates the width, so a signal is generated with input frequency and variable Pulse Width.

fig. 3.8: Neuron with two SC-multipliers

fig. 3.9: Multiplier, based on Pulse Width Modulation

In this figure, the multiplier is shown with a part of the integrating circuit, which converts the excitatory- and inhibitory voltage pulses to a single voltage.

Area
The multiplier looks complicated but when all transistors are minimum sized, then the required area is not large. It is hard to estimate the required area on forehand. The neuron has to contain a frequency-to-voltage converter.

Power Dissipation
The power dissipation of the synapses is very small, because the signals are small voltages and small current pulses. The neuron however contains a VCO. This component dissipates more power.

Noise
The information is stored in the pulse width, so the Signal-to-Noise ratio is higher than in baseband signals or in Amplitude modulated signals [4]. The synapse and neuron are shown in figure 3.10 and 3.11.

**fig. 3.10: Two Width Modulation multipliers in synapse**

One multiplier with converter is used in the Forward path. The converter contains two inverters and two transistors. In the Backward Path a Width Modulation multiplier with converter is used and also a 4 quadrant multiplier must be used.

**fig. 3.11: Width Modulation multipliers in neuron**

A Voltage Controlled Oscillator is required both in the Forward and in the Backward path. This component contains at least 6 transistors and two capacitors [9, par.5.3], but at this point, it's not clear if these transistors are minimum size transistors. It is not clear what kind of 4 quadrant multiplier must be used. The converter in the Backward path of the neuron, for converting the differential current BPI to a single...
current, could be implemented with a current mirror. It is premature at this moment to forecast the area of each building block because the architecture is not clear. In order to get an idea, the area is estimated simply by counting all transistors and converters.

3.3 Comparison Continuous versus Pulse Stream Signals

In table 3.1 all components are grouped together with the estimated area and power consumption. Minimum sized transistors are used.

For the continuous signals the differential approach is preferred above the single approach:

+ Less area because of fewer converters, but double signal lines
+ Better noise behaviour and less crosstalk

The Pulse Frequency approach is preferred with respect to those with a Pulse Width Modulation multiplier:

+ Very small SC-multiplier
+ Power dissipation, noise and crosstalk are hard to predict but shall have the same impact on Pulse Frequency and Pulse Width Modulated signals

- A main drawback of these signals is the high bandwidth of the signal lines. The steeper the edges, the better the Signal-to-Noise ratio [4], but the worse the crosstalk [24]
- The neurons contain a relative large VCO
### table 3.1: Overview Signal Representations

<table>
<thead>
<tr>
<th>Signal Representation</th>
<th>Continuous Signals</th>
<th>Pulse Stream Signals</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Area Synapse</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single</td>
<td>3 mult: 6 transistors</td>
<td>3 mult: 6 transistors</td>
</tr>
<tr>
<td></td>
<td>1 converter (diff. I→V)</td>
<td>1 converter (diff. I→V)</td>
</tr>
<tr>
<td></td>
<td>2 converters (diff. I→I)</td>
<td></td>
</tr>
<tr>
<td>Differential</td>
<td>3 mult: 6 transistors</td>
<td>3 mult: 6 transistors + 3 capacitors</td>
</tr>
<tr>
<td></td>
<td>1 converter (diff. I→V)</td>
<td>3 capacitors</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 converters (diff. I→V)</td>
</tr>
<tr>
<td>Power Dissipation</td>
<td>50µA x 5V</td>
<td>5µA x 5V</td>
</tr>
<tr>
<td>Low power in Subthreshold</td>
<td></td>
<td>Averaged in time</td>
</tr>
<tr>
<td>Noise</td>
<td>Coupled undesired signals will have no effect</td>
<td>Better Signal-to-Noise ratio than continuous signals</td>
</tr>
<tr>
<td>Crosstalk</td>
<td>Information is stored in difference between the signals so coupled signals have no effect in the information</td>
<td>Because of steep pulse edges, crosstalk can occur. Asynchronous pulses will diminish this effect</td>
</tr>
<tr>
<td>Channel Capacitance</td>
<td>For relative low frequencies</td>
<td>For high frequencies (1 MHz)</td>
</tr>
</tbody>
</table>

**Remark:** only components are summarized which are different for each signal representation.

### 3.4 Discussion

Two approaches are in my opinion the best candidates to be designed: Continuous differential signals and Pulse Frequency modulated signals with SC-multipliers.

**Continuous Differential Signals:**
- Relative small area in comparison to single continuous signals.
- Less sensitive to noise.
Signal Representation

Pulse Frequency (SC-multiplier): Small area and very low power dissipation in comparison to Pulse Width Modulation.

In the next table, advantages and disadvantages of the Continuous Differential approach and the Frequency Modulation approach with SC-multiplier are summarised.

<table>
<thead>
<tr>
<th>Continuous Differential signals</th>
<th>Pulse Stream with SC-multipliers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small area, but double signal lines</td>
<td>Very small synapse but very large neuron</td>
</tr>
<tr>
<td>Large dissipation</td>
<td>Low dissipation of multiplier (5μA averaged in time)</td>
</tr>
<tr>
<td>Low crosstalk</td>
<td>Steep pulse causes crosstalk</td>
</tr>
<tr>
<td>No noise reduction but external distortion will have no effect on information</td>
<td>Better Signal-to-Noise ratio, the steeper the pulses, the better the Signal-to-Noise ratio</td>
</tr>
<tr>
<td>Low bandwidth</td>
<td>High bandwidth because of steep pulse edges</td>
</tr>
</tbody>
</table>

The main drawback of the continuous differential approach is the large power dissipation which is caused by the transconductance multiplier. The area of the synapses are large but the neurons will be small in the differential approach.

The most required building block is the multiplier. If a multiplier can be found which consumes equal or less power than the SC-multiplier, the differential approach seems advantageous. If no multiplier can be found that consumes little power, there is no argument for using circuits based on continuous differential signals.

In the next chapter a low power multiplier will be designed and implemented. Because the Pulse Stream version is not very bad in comparison to the Continuous Differential version this Pulsed Stream version is also worked out [25]. The next chapters are focused on the design of the synapse and the neuron, working with continuous differential signals. In the end the most fruitful implementation will be used for the Neural Network.
4 Low Power Design of Forward Path

In this chapter, all elements of the synapse and neuron for the forward direction will be designed and simulated individually. Starting with the multiplier, all components are described. The used PSPICE parameters can be found in appendix I. For nominal analysis (not Monte Carlo analysis), the nominal threshold voltages are used. In chapter 6 the elements for the implementation of the Back Propagation algorithm are described.

4.1 Structure Synapse and Neuron

The signals are continuous differential currents and voltages. In figure 4.1 and 4.2, the forward path of a synapse and a neuron are shown. All multipliers are 2 quadrant and the differential input of the multiplier is suited for the Weights. The Weights represent a negative or a positive value by referring the Weight output voltage to a reference voltage. The range of the output current (A) must be equal to the input range of the sigmoid in the neuron.

![Diagram of Forward Propagation path in synapse](image)

*fig. 4.1: Forward Propagation path in synapse*

The update signal UD is now used to load and change the Weight from outside the synapse. In paragraph 5.5 the interface with the outside world is described. The sigmoid inside the neuron must convert the current to a voltage with a sigmoidal characteristic. This output voltage represents a positive value and must be compatible with the input voltage range of the multiplier inside the synapse.
The input eta (learning rate) will be used for loading and changing the Weight. The multiplier in the neuron is used as a voltage-to-current converter. The implementation of these circuits will be adapted and extended in the next paragraphs.

4.2 Subthreshold Operation

Small currents and low voltages will be used in circuits with low power dissipation. Especially for medical applications, low power dissipation is an important design constraint. In this field of design, scientists have already been developing low power circuits for many years, which are used in pacemakers and hearing-aids. The MOS transistors in most of these circuits are working in Subthreshold Operation (Weak Inversion) [3],[17].

The transistor may be characterized by the next parameters [22]:

\[ V_T = \frac{kT}{q} \]

Temperature voltage

\[ V_{th} \]

Threshold voltage

\[ n \]

Subthreshold slope

\[ I_0 = 2 \cdot n \cdot K_p \cdot V_T^2 \cdot \frac{W}{L} \]

Specific current

All parameters, except the width W and the length L, are imposed by the fabrication process. The transistor works in Subthreshold for \( V_{gs} < V_{th} \).

The drain current for a MOS transistor which is working in Subthreshold, is exponential (eq. 4.1). MOST circuits working in Subthreshold are equivalent with bipolar circuits with the advantage of having no gate current. The model that is represented by equation 4.1 is adequate for quick design calculations.

\[ I_d = I_0 \cdot e^{\frac{V_{gs}}{n \cdot V_T}} \quad V_{gs} < V_{th} \]

The currents ranges from pA to a few µA and the gate source voltages are hundreds of millivolts. Because of this small voltage range, most Subthreshold circuits are current-circuits [1]. Due to the fact that the gate source voltages are below the threshold voltage,
and thus the Id-Vgs characteristic is exponential, these circuits are very sensitive to variations in the threshold voltage. This has great influence on the behaviour of the circuit. The variations of the threshold voltage measured by IMEC are presented in appendix II. For further details about Subthreshold see the next articles [1],[21].

4.3 Multiplier

The synapse consist of a Weight Storage and a multiplier. The Weight Storage has already been designed [9] and works with an in- and output range from 1V to 4V. The reference voltage Vref equals 2.5V, so the Weight represents a value that can be positive and negative. Multipliers are searched which have two voltage inputs and a current output. One input must have a voltage range from 1V to 4V, and is connected to the Weight Storage.

A multiplier was searched that works in Subthreshold. Callias [3] uses multipliers which multiply one current with another current: \( I = I_1 \times I_2 \times C \). Other Subthreshold multipliers are found which are based on the bipolar Gilbert multiplier. The nonlinear multiplier characteristics (tanh) are linearized with tanh\(^1\) converters [7, pp.595-600]. These converters need a linear Voltage-to-Current converter with a large input voltage range from 1-4V. The VCC is tried to implement in Subthreshold but it did not succeed. But it could be implemented in Strong Inversion [19]. When a Gilbert core with two tanh\(^1\) converters are used to obtain a large voltage range (larger than 50mV in subthreshold), then a large area is required. But when this Gilbert multiplier is designed in Strong Inversion, with low currents (200nA), then these two converters are not necessary and area can be spared.

Remark: In literature [26] a suggestion has been made to combine the nonlinear tanh characteristic of the multiplier in a synapse with the sigmoid output of the neuron in the previous layer. The neuron only has to summarize all input signals, and the multiplier contains the non-linearity. In the last layer non-linearities must be added. This must be checked with system simulation of a neural network.
First a transconductance multiplier will be examined, figure 4.3.

\[ I_m = S \left( \frac{W}{L} \right) \]

**fig. 4.3: Transconductance multiplier**

The tail current \( I_b \) will be chosen large enough such that transistors \( m_1 \) and \( m_2 \) are working in Strong Inversion, and small enough for a low power dissipation. The simple quadratic equations are used for the calculations. Both transistors are working in saturation.

Used: \( S_x = \frac{K_{p_x}}{2} \left( \frac{W}{L} \right)_x \) \( S_x \): parameter for transistor pair \( x \)

The currents are:

\[ I_1 = S \left( V_{g_{s1}} - V_{th} \right)^2 \]
\[ I_2 = S \left( V_{g_{s2}} - V_{th} \right)^2 \]

\[ V = V_{g_{s1}} - V_{g_{s2}} = \sqrt{\frac{I_1}{S}} - \sqrt{\frac{I_2}{S}} \]

Use: \( I_b = I_1 + I_2 \)

\[ V \geq 0 \rightarrow I_1 = \frac{1}{2} I_b + \frac{1}{2} I_b \cdot \sqrt{\frac{2 \cdot S \cdot V^2}{I_b} - \frac{S^2 \cdot V^4}{I_b^2}} \]
\[ I_2 = \frac{1}{2} I_b - \frac{1}{2} I_b \cdot \sqrt{\frac{2 \cdot S \cdot V^2}{I_b} - \frac{S^2 \cdot V^4}{I_b^2}} \]
Low Power Design of Forward Path

\[ V \leq 0 \Rightarrow I_1 = \frac{1}{2} I_b - \frac{1}{2} I_b \cdot \sqrt{\frac{2 \cdot S \cdot V^2}{I_b} - \frac{S^2 \cdot V^4}{I_b^2}} \]
\[ I_2 = \frac{1}{2} I_b + \frac{1}{2} I_b \cdot \sqrt{\frac{2 \cdot S \cdot V^2}{I_b} - \frac{S^2 \cdot V^4}{I_b^2}} \]

Physical limitation: \( I_1 \geq 0 \text{A} \) and \( I_2 \geq 0 \text{A} \).

The currents clip for \( |V| > \frac{I_b}{\sqrt{S}} \):

\[ V \leq -\frac{I_b}{\sqrt{S}} \Rightarrow I_1 = 0 \text{A} \land I_2 = I_b \]
\[ V \geq +\frac{I_b}{\sqrt{S}} \Rightarrow I_1 = I_b \land I_2 = 0 \text{A} \]

\[ I_1 - I_2 = I_b \cdot \sqrt{\frac{2 \cdot S \cdot V^2}{I_b} - \frac{S^2 \cdot V^4}{I_b^2}} = V \cdot \sqrt{2 \cdot S \cdot I_b} \cdot \sqrt{1 - \frac{V^2 \cdot S}{2 \cdot I_b}} \quad (eq. 4.1) \]

When the current \( I_b \) is related with the square of another variable, \( V_2 \), then a good multiplication can be made. As will be explained in paragraph 4.5, the used multiplier is extended to a 4 quadrant Gilbert multiplier, fig. 4.4.

\[ \text{fig. 4.4: Gilbert multiplier with MOS transistors} \]
The quadratic equations are used and all transistors are working in saturation.

\[
V_{1} \geq 0 \rightarrow I_1 = \frac{1}{2} I_5 + \frac{1}{2} I_5 \cdot \sqrt{\frac{2S \cdot V_1^2 - S^2 \cdot V_1^4}{I_5}}
\]

\[
I_2 = \frac{1}{2} I_5 - \frac{1}{2} I_5 \cdot \sqrt{\frac{2S \cdot V_1^2 - S^2 \cdot V_1^4}{I_5}}
\]

\[
I_3 = \frac{1}{2} I_6 - \frac{1}{2} I_6 \cdot \sqrt{\frac{2S \cdot V_1^2 - S^2 \cdot V_1^4}{I_6}}
\]

\[
I_4 = \frac{1}{2} I_6 + \frac{1}{2} I_6 \cdot \sqrt{\frac{2S \cdot V_1^2 - S^2 \cdot V_1^4}{I_6}}
\]

\[
V_{1} \leq 0 \rightarrow I_1 = \frac{1}{2} I_5 - \frac{1}{2} I_5 \cdot \sqrt{\frac{2S \cdot V_1^2 - S^2 \cdot V_1^4}{I_5}}
\]

\[
I_2 = \frac{1}{2} I_5 + \frac{1}{2} I_5 \cdot \sqrt{\frac{2S \cdot V_1^2 - S^2 \cdot V_1^4}{I_5}}
\]

\[
I_3 = \frac{1}{2} I_6 + \frac{1}{2} I_6 \cdot \sqrt{\frac{2S \cdot V_1^2 - S^2 \cdot V_1^4}{I_6}}
\]

\[
I_4 = \frac{1}{2} I_6 - \frac{1}{2} I_6 \cdot \sqrt{\frac{2S \cdot V_1^2 - S^2 \cdot V_1^4}{I_6}}
\]

Physical limitation: \( I_1 \geq 0 \), \( I_2 \geq 0 \), \( I_3 \geq 0 \) and \( I_4 \geq 0 \)

\[
I_1 - I_2 = V_1 \cdot S \cdot \sqrt{\frac{2 \cdot I_5}{S} - V_1^2} \quad |V_1| \leq \sqrt{\frac{I_5}{S}}
\]

\[
I_4 - I_3 = V_1 \cdot S \cdot \sqrt{\frac{2 \cdot I_6}{S} - V_1^2} \quad |V_1| \leq \sqrt{\frac{I_6}{S}}
\]

\[
I_{out} = (I_1 - I_2) - (I_4 - I_3) = V_1 \cdot S \cdot (\sqrt{\frac{2 \cdot I_5}{S} - V_1^2} - \sqrt{\frac{2 \cdot I_6}{S} - V_1^2}) \quad \text{(eq. 4.2)}
\]

\( I_5 \) and \( I_6 \) are derived from eq. 4.1.

\[
I_5 - I_6 = V_2 \cdot S \cdot \sqrt{\frac{2 \cdot I_b}{S} - V_2^2}
\]

\( I_5 \) and \( I_6 \) can be written in the next forms:
Substitution of eq. 4.3 in eq. 4.2 results in:

\[
I_\text{out} = V_1 \cdot S^+ \left( \frac{S^+}{s} \left( \frac{I_b}{S^+} - \frac{V_2^2}{2} + \frac{V_2}{\sqrt{2}} \right)^2 - V_1^2 \right)
\]

Assume \( V_1 \) and \( V_2 \) are small:

\[
I_\text{out} \approx V_1 \cdot V_2 \cdot 2 \cdot S^+ \cdot S
\]

The tail current \( I_b \) does not influence the product. This is used for the normalization circuit, described in paragraph 4.5.

Formula 4.4 is only valid for \( |V_2| < \sqrt{\frac{I_b}{S^+}} \). Above this value, one transistor of transistor pair m5/m6 is closed and the other transistor is fully conducting.

In the calculation, the input voltages are assumed small, but the multiplier must have a large dynamic range for at least one input. The Weight voltage ranges from 1V to 4V, so one input must be adapted for this large voltage range.

The dynamic range can be enlarged by:
- using larger tail current \( I_b \)
- longer gate lengths \( \rightarrow \) smaller transistor ratios

Some methods for linearizing a Gilbert multiplier are given by [2]. Lima [10] uses linear Voltage-to-Current Converters for linearizing the nonlinear characteristic of transistor pair m5/m6. In the following, such way of linearizing will be used, but it is possible that other linearizing techniques are better. This is not investigated further. In the next figure 4.5 such circuit, a Voltage-to-Current Converter (VCC), is used for creating a larger dynamic range than with m5 and m6.

The input \( V_1 \) of the multiplier (in the synapse) is connected to the sigmoid output of a previous neuron (fig. 4.1 and 4.2). When this input voltage \( V_1 \) is not too large, the nonlinear behaviour of the Gilbert core should not be a problem. It is also easier to design a sigmoid with a small output voltage range because the currents are very small (low power dissipation), see paragraph 4.7.

The resulting multiplier is shown below.
V1: input voltage with small range
V2: input voltage with large range

In the next paragraph, the VCC is designed according to the criteria:
- input range 1-4V
- small power dissipation
- small area

4.3.1 Voltage-to-Current Converter

The reason for applying a VCC is the use of large voltages for the Weight Storage. MOS transistors used as resistors can’t work in the linear range with such small currents and large voltages, and high precision resistors with small area are difficult to design, so the transconductance of the MOS transistors is varied in order to create a Voltage-to-Current Converter with a large input voltage range [19].

4.3.2 Analysis

The VCC is based on differential transistor pairs from which the transconductance will be controlled by a voltage. In order to decrease the conductance, several transistor pairs are coupled. In the next figure, two coupled transistor pairs are shown.
All transistor pairs are identical and matched, and all transistors work in saturation. The MOS transistors will be operating in Strong Inversion and the calculations are done with the quadratic MOST equations.

Used: $S_x = \frac{Kp_x \cdot (W/L)_x}{2}$, $S_x$: parameter for transistor pair $x$

Currents for the first (left) pair:

$I_1 = I + i = S_0(v_1 - V_{th})^2 \geq 0A$  \hspace{1cm} (eq. 4.5)

$I_2 = I - i = S_0(v_2 - V_{th})^2 \geq 0A$  \hspace{1cm} (eq. 4.6)

$v_0 = v_1 - v_2 = \sqrt{\frac{I_1}{S_0}} - \sqrt{\frac{I_2}{S_0}}$  \hspace{1cm} (eq. 4.7)

The information $\Delta I (=2i)$ can be extracted from (eq. 4.5, 4.6 and 4.7).

$$\Delta I = 2 \cdot v_0 \cdot S_0 \cdot \sqrt{\frac{I}{S_0} - \frac{v_0^2}{4}} = 2 \cdot v_0 \cdot \sqrt{I \cdot S_0} \cdot \sqrt{1 - \frac{v_0^2 \cdot S_0}{4 \cdot I}}$$  \hspace{1cm} (eq. 4.8)

The first part of the equation represents the transconductance of the first transistor pair. In a similar way for the second pair:
The input voltage is divided over two transistor pairs:

\[ v_i = v_\theta + v_\varphi \]

Relation between \( v_\varphi \) and \( v_\theta \):

\[ v_\theta^2 \cdot S_\theta = v_\varphi^2 \cdot S_\varphi \]

\[ v_\varphi = v_\theta A, \quad A = \frac{S_\theta}{S_\varphi} \]

\[ v_i = v_\theta (1+A) \rightarrow v_\theta = \frac{v_i}{(1+A)} \quad (eq. 4.10) \]

Substitution of 4.10 in 4.8 results in:

\[ \Delta I = 2 \cdot \frac{v_i}{(1+A)} \cdot \sqrt{1 \cdot S_\theta} \cdot \sqrt{1 - \frac{v_\theta^2 \cdot S_\theta}{4 \cdot I \cdot (1+A)^2}} \quad (eq. 4.11) \]

The negative feedback decrements the transconductance with (1+A) and the distortion factor with (1+A)^2. When the number of stages is extended, the input voltage \( v_i \) is divided over more stages and the transconductance and distortion will decrease: (1+A+B+C+...) and (1+A+B+C+...)^2. The transconductance of all transistor pairs are expressed in the transconductance of the left input pair.

Response for \( m \) transistor pairs (\( m \) stages):

\[ A = B = \ldots = 1, \quad equal \ S \]

\[ \Delta I = 2 \cdot \frac{v_i}{m} \cdot \sqrt{1 \cdot S_\theta} \cdot \sqrt{1 - \frac{v_i^2 \cdot S_\theta}{4 \cdot I \cdot m^2}} = 2 \cdot \frac{v_i}{m} \cdot \sqrt{1 \cdot S_\theta} \cdot D \quad (eq. 4.12) \]

\[ D = \sqrt{1 - \frac{v_i^2 \cdot S_\theta}{4 \cdot I \cdot m^2}} = \sqrt{1 - \alpha} \quad (eq. 4.13) \]

Physical limitation: \( I_1 \geq 0 A, I_2 \geq 0 A, I_3 \geq 0 A \) and \( I_4 \geq 0 A \)

Equation 4.12 is only valid for \( |v_i| \leq m \cdot \frac{2I}{S_\theta} \).

When this voltage is exceeded, the currents will clip.
\[ v_i \leq -m \cdot \sqrt{\frac{2I}{S_0}} \rightarrow I+i=0A \land I-i=2I \rightarrow \Delta I = -2I \]

\[ v_i \geq +m \cdot \sqrt{\frac{2I}{S_0}} \rightarrow I+i=2I \land I-i=0A \rightarrow \Delta I = 2I \]

This converter can be extended with more transconductance pairs. The input voltage is then divided over more pairs and the converter will be more linear. A drawback is the required area and the extra current sources that must be implemented.

The distortion \( \alpha \), eq. 4.13, can be decreased simply by taking transistors with large gate lengths. Another method is to extend the converter with extra transistor pairs (m). The distortion will decrease quadratic. When the current is increased, the distortion decreases but the power dissipation increases.

4.3.3 VCC with 3 stages

This converter can be enlarged by adding PMOS or NMOS pairs with current sources. Least current sources are required with odd transistor pairs. For example if the VCC is extended to 4 pairs, by adding 2 PMOS pairs between the two NMOS pairs, 5 current sources are required. But if this VCC is extended to 5 pairs, by adding 2 PMOS and 1 NMOS pair between the two NMOS pairs, also 5 current sources must be implemented [19]. With respect to the number of transistor pairs and the number of current sources, it is favourable to use a VCC with an odd number of transistor pairs. A VCC is built with 3 transistor pairs (fig. 4.7).

\[ \text{fig. 4.7: 3 stage Voltage-to-Current Converter} \]
Suppose: \( \frac{W}{L}_\theta = \frac{W}{L}_\varphi = \frac{W}{L}_\gamma = \frac{W}{L} \)

Look only at conductance: \( \frac{\sqrt{IS_\theta}}{(1+A+B)} \)

\[
A = \sqrt{\frac{S_\theta}{S_\varphi}} = \sqrt{\frac{Kp_\theta}{Kp_\varphi}} = \sqrt{\frac{57 \cdot 10^{-6}}{17 \cdot 10^{-6}}} = 1.831 \quad (\text{eq. 4.14})
\]

\[
B = \sqrt{\frac{S_\gamma}{S_\theta}} = 1 \quad (\text{eq. 4.15})
\]

Combining equations 4.14 and 4.15:

\[ m = (1 + A + B) = 3.831 \quad (\text{eq. 4.16}) \]

### 4.3.4 PSPICE simulations

When the output currents of the converter are chosen, the dissipation of the multiplier is fixed as well. A 3 stage VCC is examined because it contains an odd number of stages, and needs relatively few current sources. When a 4 stage VCC was chosen, relatively more current sources with large gate lengths must be used (next paragraph), and then it's more fruitful to extend the VCC to 5 stages. Also more transistors means more signal lines between them.

The total current flowing through the 3 stage VCC is 6I. The current I=100nA is chosen large enough for the transistors to work in Strong Inversion, and the total current is 600nA per VCC. This current may be increased to create a more linear behaviour (eq. 4.12 but the power dissipation also increases. When the W/L ratio of the transistors in the VCC is decreased to 2.4/48, then the characteristic has a full scale linearity error of 20% for an input voltage range from -1.5V to +1.5V. Despite this large error, this VCC with 3 stages will be used. When this non linearity proofs to be too large (chapter 6), the VCC can always be adapted. The reference voltage Vref=2.5V is connected to V'.

In the PSPICE simulations the next parameters are used:
- I=100nA
- (W/L) = 2.4/48
- m=3.831

**Remark:** The supply voltage Vdd=5V in all PSPICE simulations. The theoretical characteristic of this 3 stage VCC can be derived from eq. 4.12.
\[ \Delta I_{vcc} = V_2 \cdot 2 \cdot \frac{\sqrt{I_{S_{vcc}}}}{m} \cdot D_{vcc} = V_2 \cdot 197 \cdot 10^{-9} \cdot D_{vcc} \]

This characteristic is shown in the next figure:

**fig. 4.8: Theoretical differential output current of 3 stage VCC**

A PSPICE simulation for these VCC with the already mentioned parameters is shown in figure 4.9.
Low Power Design of Forward Path

Used values: \( V_{in} = V^+ - V^- \)
\( V = 2.5V \)

![Graph showing differential output current and linearity error for 3 stage VCC](image)

fig. 4.9: Differential output current and linearity error for 3 stage VCC

The theoretical transconductance (197nA/V) does not differ very much from the simulated transconductance: 182nA/V.
In paragraph 4.3.6, this VCC will be combined with the multiplier core and the response of the multiplier with VCC will be derived.
4.3.5 VCC with Non ideal Current sources

In figure 4.10, a VCC is shown with MOS transistors as current sources.

![Diagram of VCC with NMOS input transistors and non ideal current sources](image)

The positive input $V^+$ for the Weight varies from 1V to 4V, and the negative input $V^-$ is 2.5V. The voltage drop $V_{ds}$ over the transistor $m_8$ diminishes the linear conversion range of the VCC. The current $2I=200nA$. When the transistors $m_7$, $m_8$ and $m_9$ are minimum sized, then they work in Subthreshold, and a variation in threshold voltage of 20mV (appendix II) results in a current deviation of 25%. The transistors can work in Strong Inversion (less sensitive to variation in threshold voltage) when the gate length is increased, but then the voltage drop $V_{ds}$ will grow. This voltage drop results in a saturation effect for small voltages at the positive input of the VCC. When the voltage $V^+-V_{ds_{m8}}$ is below the threshold voltage of transistor $m_1$, this transistor works in Subthreshold and the drain current through this transistor is relative small in comparison to the drain current flowing through transistor $m_6$. The equation 4.12 is not valid any more and further decrease of $V^+$ has no effect on the differential output current (fig. 4.11).

A compromise must be made: a large input voltage and a large sensitivity to spread in threshold voltage, or a smaller input voltage range and less sensitivity.

For the current source transistors ($m_7$, $m_8$ and $m_9$) a ratio of 2.4/24 is chosen.
With PSPICE nominal and Monte Carlo analysis is done with the circuit in figure 4.10. With Monte carlo analysis a maximum threshold voltage variation of 20mV between neighbour transistors is possible.

Used values: \( V_2 = V^+ - V^- \)
\[ V^- = 2.5V \]

\[ \square \text{ nominal characteristic} \]

\[ \text{fig. 4.11: Nominal and Monte Carlo Simulation NMOS VCC} \]

In this characteristic, the left current end saturates. This occurs when the voltage \( V_{gs} \) of transistor \( m_1 \) is below the threshold voltage. In this area the drain current through \( m_1 \) is very sensitive to threshold variation.

In the neuron, a VCC is used to implement the multiplication (paragraph 4.1). This VCC has output currents which must flow to ground. This is realized by exchanging all NMOS transistors for PMOS transistors and vice versa. The output current varies from -400nA to +400nA (will be explained in paragraph 5.2).
**VCC with PMOS input transistors and non ideal current sources**

In the next figure, one nominal and 4 Monte Carlo simulations are shown for the same circuit.

- **nominal characteristic**

```
Date/Time run: 11/10/92 09:54:29  Temperature: 27.0
```

**fig. 4.13: Monte Carlo Simulation PMOS VCC**
This VCC has a larger output current swing (400nA) than the NMOS VCC (fig. 4.10) and therefore, the current transistors deliver a larger current than those for the NMOS VCC. For this reason, the ratio of the current source transistors in the PMOS VCC is larger. Thus for the transistors m7, m8 and m9 a ratio of 2.4/8 is chosen.

Only the characteristic-ends change a lot due to variation in threshold voltage. The reason for this effect is transistor m1 (fig. 4.10 and 4.12), which is working in Subthreshold. When this effect is not tolerable, the voltages Vpmos and Vnmos must be changed in order to increase the gate source voltages. This diminishes the negative effect due to variation in the threshold voltage but requires larger gate lengths for creating small currents. This consumes more area. In paragraph 4.8 some suggestions are made for decreasing threshold mismatching.

4.3.6 Gilbert multiplier with VCC

The proposed multiplier, consisting of a Gilbert core and a 3 stage VCC will be used. The resulting multiplier with Gilbert core and VCC is once more shown in the next figure.

The output of the Gilbert multiplier core has already been derived, eq. 4.2, and now the transistors m5/m6 are replaced with a 3 stage VCC. The calculation is almost the same as for the Gilbert multiplier because the upper part (m1/m2/m3/m4) stays the same. From eq. 4.12 it can be derived that:

\[ I5 - I6 = \Delta I_{vcc} = \frac{2 \cdot V2}{m} \cdot \sqrt{I \cdot S_{vcc}} \cdot \sqrt{1 - \frac{V2^2 \cdot S_{vcc}}{4 \cdot I \cdot m^2}} \]  

(eq. 4.17)

I5 and I6 can be written in the next forms:
$I_5 = \frac{1}{4}S_{VCC}\cdot\left(\frac{4\cdot I}{S_{VCC}} - \frac{V_2^2}{m^2} + \frac{V_2}{m}\right)^2 \land I_6 = \frac{1}{4}S_{VCC}\cdot\left(\frac{4\cdot I}{S_{VCC}} - \frac{V_2^2}{m^2} - \frac{V_2}{m}\right)$

Substitution of $I_5$ and $I_6$ in eq. 4.2 results in:

$$I_{out} = V_1\cdot S_{Gu}\cdot\left(\frac{S_{VCC}}{S_{Gu}}\cdot\left(\frac{2\cdot I}{S_{VCC}} - \frac{V_2^2}{2\cdot m^2} + \frac{V_2}{\sqrt{2}\cdot m}\right)^2 - V_1^2 + \right)$$

$$- \frac{S_{VCC}}{S_{Gu}}\cdot\left(\frac{2\cdot I}{S_{VCC}} - \frac{V_2^2}{2\cdot m^2} - \frac{V_2}{\sqrt{2}\cdot m}\right)^2 - V_1^2$$

(eq. 4.18)

Assume $V_1$ and $V_2$ small:

$$I_{out} = V_1\cdot S_{Gu}\cdot\frac{S_{VCC}}{\sqrt{2}\cdot m} = \frac{V_1\cdot V_2\cdot \sqrt{2}\cdot S_{Gu}\cdot S_{VCC}}{m}$$

Equation 4.18 is only valid for $|V_2| < m\cdot\frac{2\cdot I}{S_{VCC}}$, and

$$V_1 \leq \frac{S_{VCC}}{S_{Gu}}\cdot\left(\frac{2\cdot I}{S_{VCC}} - \frac{V_2^2}{2\cdot m^2} - \frac{V_2}{\sqrt{2}\cdot m}\right)$$

In this formula, all transistors are assumed to work in Strong Inversion and are saturated. The dynamic range is now decreased with $m$ (compare eq. 4.4 and eq. 4.18). PSPICE simulations have shown that when the input voltage $V_1$ is kept below 0.2V, the linearizing circuit is not necessary for a good multiplication.
The valid part of this formula is plotted in the next figure.

The currents will clip in the area in which formula 4.18 is not valid. The clipped output currents are determined by V1.

The W/L ratios of all transistors m1 to m4 are 2.4/24.

PSPICE simulation shows that the multiplier works with only one VCC but the linearity is not very good within the large voltage range from -1.5V to +1.5V. When the non-linearity is too large, the VCC can be extended. The non-linearity is determined by the VCC characteristic.
Five curves are plotted for $V_1=0\text{mv}-50\text{mV}-100\text{mV}-150\text{mV}-200\text{mV}$.

**fig. 4.16: PSPICE simulation multiplier**

The left ends are curved and saturated due to the VCC with MOS current sources.
Monte Carlo simulations (fig. 4.17) show a horizontal and a vertical shift of the characteristic which can be modeled as offset. The left ends vary a lot: this is caused by the VCC. One plot (V1=0V) even has a negative gradient for negative V2. Only the worst characteristics are plotted in the next figure. Five curves are plotted for V1=0mV-50mV-100mV-150mV-200mV.

The results from Monte Carlo analysis will be used for System Simulation (chapter 6) with non ideal multipliers. This simulation must decide whether or not the multiplier must be adapted.
4.4 Comparison Gilbert multiplier with and without VCC

The VCC was used for linearizing the lower part (m5/m6) of the Gilbert multiplier. Transistors m5 and m6 (fig. 4.4) can be seen as a one-stage VCC and the output current can be derived with equation 4.12. The distortion $\alpha$ can be decreased (paragraph 4.3.1) and it is obvious that the extension with other transistor pairs has more influence on $\alpha$ than increasing the current $I_b$.

The multiplier with 3 stage VCC has a current flow of 3x200nA=600nA.
Transistors: Gilbert core 4x2.4/24
                 VCC 6x2.4/48
Total estimated area: 2900$\mu$m$^2$
Characteristic: figure 4.16

Now a Gilbert multiplier without VCC is simulated with a tail current $I_b=600nA$. In order to obtain the same dynamic range, the gate lengths were increased to extreme lengths.
Transistors: Gilbert multiplier 6x2.4x180
Total estimated area: 5500$\mu$m$^2$
Characteristic: see next figure. Five curves are plotted for $V1=0$mV-50mV-100mV-150mV-200mV.

$\text{Gilbert multiplier with large tail current (800nA)}$

Date/Time run: 11/10/92 10:30:43
Temperature: 27.0

According to the required area, it is more fruitful to extend the number of transistor pairs in the lower part of the Gilbert multiplier, than increasing the gate lengths and tail current. However these characteristics are better than those achieved with a Gilbert multiplier with VCC (fig. 4.16).
4.5 Normalizing: approach for hundreds of synapses

The structure of the neural network is variable and depends on the users' problem. Several synapses may be connected per neuron input but to handle a difficult pattern recognition problem, hundreds of synapses per neuron may be preferred. Because of the limited working range of the sigmoid, the Back Propagation algorithm will adjust the Weights in the synapses in such way that the total summarized signals from the synapses will stay within a certain input range of the sigmoid. No matter if there are twelve or at least twelve hundred synapses per neuron.

In a software environment, the Weights are usually implemented as floating point values with a very large range. The accuracy of the Weights is related with the machine accuracy and is very high, and very small values can be used. The Back Propagation algorithm is able to adjust the Weights to very small values in order to work with hundreds of synapses.

In hardware however, the Weights are limited to the supply voltage, and are quantized. When hundred synapses are connected to the neuron, the back Propagation algorithm may adjust the Weights to small values. But the smallest value, nonequal to zero, is represented by the first quantization level. Weights can skip between zero and this level during training. If the smallest value is not equal to zero, the summarized output of all synapses may saturate with hundreds of synapses.

Three solutions to this problem are mentioned below.

1) The neuron "knows" how many synapses are connected and will adjust the gradient of the sigmoid; more synapses correspond with a smaller gradient. This method will only work when the summation of the differential output currents is fixed. When the resulting currents of the Gilbert multiplier are summed, the tail current will appear (fig. 4.4). The tail current is independent of the input signals of the multiplier. The neuron can identify the number of synapses and can adjust the sigmoid. The information is obtained simply by subtracting the two input currents of the neuron.

Advantage: no extra signals between synapse and neuron.

A drawback of this method is the large current that flows into the neuron if there are many synapses. The adjusting circuit must work with a very large current input range which is unbounded.
2) The second method is based on normalization of the output currents of the synapses. The total current will be normalized to a current that lies within a fixed current range. The sigmoid input range is designed for this current range. When more synapses are connected, the individual output currents of the normalizing cells will decrease, but the total output current will be the same. An advantage of this method is that the total input current of the neuron is always the same. Because the normalizing circuit is distributed over the number of synapses, the total current of the normalizing cell is also distributed over all synapses. A drawback is the extra connection at every synapse and neuron.

3) A third method is normalization via impedances. The output current of a synapse is generated with current sources with a finite output impedance. When synapse output currents are summarized, the current will grow but the impedance will decrease due to paralleling of impedances. The voltage at the summation point of all synapses is the average of all synapse outputs. An advantage of this method is that no extra pins are required. This method is not worked out however it is useful to investigate this method further.

The second and third solution are more elegant than the first one because the normalization occurs automatically. Theoretically, the number of synapses is unbounded. In the next paragraph the second method is analyzed.
4.5.1 Analysis Normalizing cell

The next circuit is used for normalization [6],[22].

\[
\text{fig. 4.20: Normalization circuit with bipolar transistors}
\]

The voltage Vs is equal for all connected transistor pairs. Suppose all transistor pairs are identical and matched. With the exponential bipolar equations, the next formula can be derived:

\[
V_s = V_T \cdot \ln\left(\frac{I_{in_1}}{I_{out_1}}\right) = V_T \cdot \ln\left(\frac{I_{in_2}}{I_{out_2}}\right) = \ldots = V_T \cdot \ln\left(\frac{I_{in_n}}{I_{out_n}}\right)
\]

\[
I_{out_i} = k \cdot I_{in_i}, \quad \forall i \quad k=\text{normalize constant}
\]

\[
\sum_{i=1}^{n} I_{out_i} = I_t
\]

This circuit is extended for differential currents (fig. 4.21). Now the normalizing cell handles two input and two output currents. The synapse outputs (multiplier) will be connected to the inputs of the normalizing cell. The total input current off all cells together depends on the number of synapses. The summed output current of all cells is restricted to 2I\(_t\).

The relation between differential input current I\(_{in_n}\) and differential output current I\(_{out_n}\) for cell \(n\) is:

\[
I_{out_n} = k \cdot I_{in_n}, \quad \forall n \quad k=\text{normalize constant}
\]
\[ \sum_{i=1}^{n} I_{out_i} = 2I_t \]

Figure 4.21 shows the extension with other synapses.

The summed output current of a 2 quadrant Gilbert multiplier is not fixed, but dependent to one input and varies. The summed output current of a 4 quadrant Gilbert multiplier is fixed to the tail current. This tail current is independent on the inputs of the multiplier. The normalization circuit normalizes the total incoming current in relation to the total output current 2I_t. When the summarized input current is not fixed (4 quadrant Gilbert) but variable (2 quadrant Gilbert), then the normalization does not work.

This normalization circuit has proven to be successful for several applications in which it works with bipolar transistors instead of MOS transistors.

4.5.2 Implementation Normalizing cell with Non ideal Current Source

The normalization circuit is based on bipolar transistor equations, and it is essential to work with exponential equations. In order to retain exponential equations, the MOS transistors have to work in Subthreshold. The drain currents will vary from 0 to 200nA (output current range multiplier). The transistor W/L ratio must be large in order to get a Vgs below the Subthreshold voltage. PSPICE simulations show that a W/L ratio of 28/2.4 is sufficient to steer the transistors in Subthreshold with a current below 200nA.
The current source for the generation of the current $2I_t$ is realized with a PMOS load transistor that is working in saturation. To diminish the effect of variation in the threshold voltage, this transistor must not work in Subthreshold Operation.

The drain source voltage of the load transistor must be small. In the next figure one part of a normalizing cell is plotted with a PMOS transistor used as current source. The voltage $V_{dd}$ (in the ideal version) is replaced with the voltage $V_{norm}$.

\[ \text{fig. 4.22: Current source with PMOS load transistor} \]

Suppose $V_{norm} \neq V_{dd}$ and a current $I_{in}$ flows through transistor $m_1$. When $V_{norm}$ decreases, $V_g$ of $m_1$ will also decrease retaining the same current $I_{in}$. For retaining the same current $I_{out}$, the source voltage of $m_2$ ($V_d$) also decreases. But when $V_{norm} = V_{dd}$, the voltage $V_d$ will be near the supply voltage. In this case $V_{ds}$ of $m_3$ is too small for $m_3$ to work in saturation. PSPICE simulations have confirmed this behaviour.

The voltage $V_{norm}$ must be lower than $V_{dd}$ but this means a smaller voltage range for the multiplier and VCC which are connected "between" the normalization cell and ground. A drawback is that this voltage $V_{norm}$ must be distributed over the whole chip. For the PMOS load transistor, a $W/L$ ratio of 2.4/28 is chosen. When $V_{pmos} = 3.6V$ ($V_{gs} = -1.4V$), the output current $2I_t$ will be about 195nA and the voltage drop over transistor $m_3$ is 0.4V. $V_{norm}$ is chosen lower: 4.5V. With nominal threshold voltages the normalization cell works very well (next paragraph), but a variation in threshold voltages destroys the nice linear behaviour (Monte Carlo analysis).

**4.5.3 Pspice simulations**

The total current $2I_t$ is chosen to be 200nA. Simulations are done with 1, 5, 10 and 21 connected normalize cells. The normalization works well and the results with 21 cells are
shown below. In the next PSPICE simulation, the outputs of 21 normalizing cells are connected and different currents are applied on the 21 differential current inputs. The theoretical and simulated currents are given in table 4.1. In the simulation, a PMOS transistor is used as current source with an output current of 195nA.

<table>
<thead>
<tr>
<th>cell number</th>
<th>Input currents (nA)</th>
<th>Differential input current (nA)</th>
<th>Theoretical output current (nA)</th>
<th>Simulated output current (nA)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;left&quot;  &quot;right&quot;</td>
<td></td>
<td>Nominal            Monte Carlo</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>200 0</td>
<td>200</td>
<td>9.286              8.929        13.091</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>190 10</td>
<td>180</td>
<td>8.357              8.233        7.815</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>180 20</td>
<td>160</td>
<td>7.429              7.531        12.189</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>170 30</td>
<td>140</td>
<td>6.500              6.631        8.984</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>160 40</td>
<td>120</td>
<td>5.571              5.685        4.448</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>150 50</td>
<td>100</td>
<td>4.643              4.739        7.142</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>140 60</td>
<td>80</td>
<td>3.714              3.792        3.888</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>130 70</td>
<td>60</td>
<td>2.786              2.844        4.269</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>120 80</td>
<td>40</td>
<td>1.857              1.896        2.370</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>110 90</td>
<td>20</td>
<td>0.929              0.948        2.587</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>100 100</td>
<td>0</td>
<td>0.00               0.000        0.949</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>90 110</td>
<td>-20</td>
<td>-0.929             -0.948       2.308</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>80 120</td>
<td>-40</td>
<td>-1.857             -1.896       0.857</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>70 130</td>
<td>-60</td>
<td>-2.786             -2.844       -1.818</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>60 140</td>
<td>-80</td>
<td>-3.714             -3.792       -2.948</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>50 150</td>
<td>-100</td>
<td>-4.643             -4.739       -4.756</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>40 160</td>
<td>-120</td>
<td>-5.571             -5.685       -5.398</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>30 170</td>
<td>-140</td>
<td>-6.500             -6.631       -7.659</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>20 180</td>
<td>-160</td>
<td>-7.429             -7.531       -7.626</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>0 200</td>
<td>-200</td>
<td>-9.286             -8.929       -12.703</td>
<td></td>
</tr>
</tbody>
</table>

The results are plotted in the next figure. The theoretical output currents are visualized with a continuous line, but only integer cell numbers can be used! With Monte Carlo analysis, 10 simulations are done and the simulation with the largest absolute error is shown.
Low Power Design of Forward Path

x : nominal simulation
□ : Monte Carlo simulation

Output current
[nA]

fig. 4.23: Theoretical and simulated output currents with 21 normalizing cells

The absolute error is plotted in the next figure.
Absolute output error: \[ e(k) = \frac{I(k)_{\text{theoretic}} - I(k)_{\text{simulation}}}{I_{\text{max, theoretical}}} \times 100\% \]

The error from Monte Carlo analysis (next figure) must be multiplied with 10 to obtain the real error.

**fig. 4.24: Absolute conversion error (%)**

The error between theoretical and simulated output current never exceeds 4%.

The absolute error for the nominal simulation is large at high currents because one transistor is getting out of the Subthreshold region. This effect can be avoided by enlarging the W/L ratio of all transistors within the normalizing cell. However, this "large" error takes place at the saturation area of the multiplier and the sigmoid, and is negligible in comparison to the saturation effect.

When threshold mismatch of 20mV is realistic then this normalize circuit can not be used because of the large absolute output error. Then other methods for some kind of normalization must be searched. Some design rules for minimizing the threshold mismatch are given in paragraph 4.8.
4.6 Structure Synapse and Neuron with Normalization

The structure of the synapse and the neuron are changed with respect to those described in paragraph 4.1. The voltages FPI and FPO are now differential, and can be connected to the multiplier in the synapse. Drawback is the extra input per synapse and extra output per neuron. But the used currents and voltages are small so a differential approach is favourable with respect to noise. Also every synapse is extended with one normalizing cell (4 transistors), and has one extra input for the current $I_n$. In figure 4.25 the synapse is shown.

![Synapse with normalizing cell](image)

*fig. 4.25: Synapse with normalizing cell*

The current source $2I_t$ must be implemented within each neuron, and distributes the current to all connected synapses (fig. 4.26, 4.27 and 4.28).

![Neuron with differential signals and current source](image)

*fig. 4.26: Neuron with differential signals and current source*

In figures 4.27 and 4.28 the synapse chip and neuron chip are shown.
fig. 4.27: Synapse chip (without Back Propagation) with normalize current

fig. 4.28: Neuron chip (without Back Propagation) with current source

When it is able to select one neuron Weight in this chip, only one pin for all eta-signals is required.

Remark: the voltages Vref, Vnorm, Vmos and Vpmos (not shown in the figures above), must be transported over the whole chip.
The differential input and output signals are illustrated with the ellipses near the signal lines. The synapse chip contains NxM synapses: M neurons can be connected with N synapses per neuron. For the synapse chip Mx4+Nx2 output pins are required for the Forward path.

The neuron chip requires Mx6 pins for the Forward path. Pins for control voltages, supply voltage and ground are not included.

4.7 Sigmoid

The nonlinear characteristic of a transistor pair is used for the implementation of a sigmoid like non-linearity. The characteristic of a transistor pair working in Subthreshold is given by equation 4.23

\[ \Delta I = I_{bias} \cdot \tanh \left( \frac{V}{2 \cdot n \cdot V_T} \right) \]  

(eq. 4.23)

This function is plotted in the next figure.

*fig 4.29: Tangent hyperbolical*

Used values:  
\[ I_{bias}=500 \text{pA} \]  
\[ n=1.5 \]  
\[ V_T=26 \text{mV} \]

4.7.1 Implementation

The incoming currents from the normalizing cell (I1 and I2) vary from 0nA to 200nA and must be converted to voltages, in order to drive the transistor pair m3/m4 (fig. 4.30). With two load transistors m5/m6, the differential output current of pair m3/m4 is converted to voltages. This differential voltage Vout must satisfy with the voltage range of the multiplier of the synapse.
All transistors have minimum sizes, except for m1 and m2. These transistors convert the incoming currents (0-200nA) to a voltage within the range of the transistor pair. This voltage range is -70mV to 70mV. The next ratios are chosen:
m1: $W/L = 2.4/41.6$
m2: $W/L = 2.4/24$
The load transistors m5/m6 work in saturation and must convert a current to a voltage. The differential output voltage $V_{out}$ varies from 0mV to 200mV. When another synapse (next layer) is connected, this voltage (FPO) is equal to the input voltage (FPI) range of the used multiplier. Simulations have proven that the tail current Ibias must be small for getting a non-linear behaviour. This current must be about 400pA (400$x10^{-12}$A). When the PMOS loads are working in saturation, a quite linear current-to-voltage conversion is possible with small voltage swings of several hundreds of millivolts. The drain source voltage of the PMOS loads range from 0V to approximate 1V for currents between 0 and 400pA. This means that the two output voltages $V_{out^+}$ and $V_{out^-}$ are about 4V. This is high enough for a proper functioning of the multiplier. The differential voltage swing is about 200mV.

The differential output voltage of this circuit should be positive. Therefore an extra current source is added with a current equal to the tail current Ibias. In figure 4.31 this addition is shown with MOS transistors used as current sources.
Vd of m8 is larger than Vd of m7. Because of this difference and the Early effect of the transistors, the drain current of m8 will be larger too. Therefore, the W/L ratio of m8 must be changed to compensate this effect. The W/L ratio of m8 must be decreased to obtain equal currents.

PSPICE simulations have shown (Vbias=0.8V):

m7: Vds=0.5V    Id=387pA
m8: Vds=4V     Id=493pA

\[
\left( \frac{W}{L} \right)_{m7} = \frac{493}{387} \left( \frac{W}{L} \right)_{m8} \wedge W_{m7} = W_{m8} \Rightarrow L_{m8} = 1.27 \cdot L_{m7}
\]

L_{m7}=2.4\mu m, L_{m8}=3.2\mu m

In figure 4.32 a PSPICE simulation is shown of this circuit.
Used values:
I_{in} = I_1 - I_2
V_{bias} = 0.8V

fig. 4.32: PSPICE simulation non-linearity

This circuit is also analyzed for variations in the threshold voltage. The current sources m7 and m8 are very sensitive to variations in the threshold voltage, because they are working in Subthreshold. This effect can be diminished by increasing the voltage V_{bias} above the threshold voltage. But this results in very long gate lengths in order to get a small current of 400pA.

Example: 
V_{bias}=0.8V \rightarrow L = 2.4\mu m \rightarrow \text{Id}_{m7} \approx 380\text{pA}
V_{bias}=0.95V \rightarrow L = 160\mu m \rightarrow \text{Id}_{m7} \approx 300\text{pA}
In figure 4.33, Monte Carlo simulation is plotted.

**Monte Carlo analysis. Sigmoid load**

The horizontal shift is not very critical because the Back Propagation algorithm is able to compensate this shift. The right end varies a lot from 160mV to 260mV for varying threshold voltages. This effect is not damaging for the network: it only means a different working range.

The negative outputs at the left end may cause problems because the multiplier in the synapse is a 4 quadrant multiplier. Therefore the current sources m7,m8 should be less sensitive to variation in threshold voltage. This results in large gate lengths, so other techniques for shifting the characteristic should be examined.

Although this drawback, all components have been designed yet, and in chapter 5, a complete synapse and neuron are simulated.
4.8 Optimum Transistor Matching

Mismatching is the major limitation to accuracy of analog circuits [Vittoz, 20]. The following design rules are a recipe for minimizing the threshold mismatch as far as possible for transistor pairs.

+ same temperature (no problem with low power)
+ same length and width
+ minimum distance between two transistors
+ same orientation to eliminate dissymmetries in process ↦ source to drain current flows in matched transistors should be parallel
+ no minimum sizes to reduce the effect of edge fluctuations and to obtain averaging of fluctuating parameters
5 Implementation Synapse and Neuron

With all components, earlier described, a synapse and a neuron are built. With PSPICE static and transient simulations are done with the subcircuits, given in appendix III.

5.1 Synapse Implementation

The complete synapse is shown in the next figure.

A Monte Carlo simulation with this synapse is shown in fig. 5.2.
A nominal and the worst Monte Carlo simulation are shown. Used values:

- \( V_{\text{ref}} = 2.5 \text{V} \)
- \( V_{\text{norm}} = 4.5 \text{V} \)
- \( V_{\text{pmos}} = 3.6 \text{V} \)
- \( V_{\text{nmos}} = 1.25 \text{V} \)
- \( V_{\text{w_synapse}} = -1.5 \text{V} \ldots +1.5 \text{V} \)

In each simulation three values for FPI are used: 0mV-100mV-200mV.

\[ \text{fig. 5.2: Nominal and Monte Carlo analysis of single Synapse} \]

The offset is very large which is caused by the normalizing cell.

### 5.2 Neuron Implementation

The neuron contains a VCC, a sigmoid load and a current source. The VCC is a PMOS version, and the output currents of the VCC are connected with the input of the sigmoid load (fig. 5.3).
fig. 5.3: Neuron

Used value: $V_{\text{bias}}=0.8\text{V}$.

The input current of the sigmoid ranges from -200nA to +200nA, and the Weight in the neuron must be able to "shift" the sigmoid so that the sigmoid output can be maximal when the differential input (A) of the neuron is minimal. Therefore the output current of the VCC must range from -400nA to +400nA.

The added VCC in the neuron contributes a total summed output current of 400nA. In paragraph 4.7 the sigmoid has been designed for a summed input current of 200nA. Therefore the converter in the sigmoid (fig. 4.30, m1/m2) are replaced with two PMOS versions with other ratios:

- m1: $W/L=2.4/48$
- m2: $W/L=2.4/41.6$

The bias voltage for the sigmoid is still 0.8V.
In the next figure, the neuron input current $I_{in}$ (A) is varied from -200nA to +200nA and the Weight voltage of the neuron ranges from 1V to 4V ($V_{w,neuron}$ ranges from -1.5 to +1.5V with steps of 0.25V). This results in a "shift" of the sigmoid shape.

\[ \text{fig. 5.4: Neuron response to varying input current (A) and internal Weight} \]

The Weight is able to create a maximum output with a maximum negative input current. When the Weight voltage equals the reference voltage ($V_{w,neuron}=0$V), then the nominal sigmoidal characteristic is obtained. The sigmoidal characteristic is very sensitive to threshold mismatch (paragraph 4.8). This can be suppressed by using current sources which are not working in Subthreshold, but this results in very large gate lengths.

In the next paragraphs, synapses are connected to a neuron. Simulations of this small network are compared with the theoretical expected characteristics, based on the PSPICE simulations, described in chapter 4. With Transient analysis, the network speed will be investigated.

5.3 Two Synapses connected with a Neuron

5.3.1 Static Analysis

Static and Transient analysis is done on a network consisting two synapses (fig. 5.5).
The three weight voltages are fixed and both inputs are varied.

Used values: $V_{w_{\text{synapse1}}} = 0.5\, \text{V}$
$V_{w_{\text{synapse2}}} = -1\, \text{V}$
$V_{w_{\text{neuron}}} = 0\, \text{V}$

Input FPI1 is varied from 0mV to 200mV for three values of FPI2 (0mV - 100mV - 200mV). The expected neuron output voltage $FPO$ can be derived from the multiplication characteristic (fig. 4.16) and the neuron characteristic (fig. 5.4).

Remark: the outgoing currents from both synapses must be normalized.

The simulation is shown in the figure below.
The simulated characteristics are a little curved because of the curved multiplier characteristic of synapse S1 (fig. 5.2). The difference between all curves is also not equal due to the multiplier characteristic in the synapse. The simulated values does not differ very much from the expected output values.
5.3.2 Transient Analysis

The speed of this "network" can be investigated with Transient analysis. It is predictable that the speed will not be very high because all parasitic capacitances must be charged and discharged with small currents. Both inputs FPI1 and FPI2 are varied simultaneously in order to investigate the propagation time of the circuit. The output voltage FPO can be checked with the static voltage in fig. 5.7.

![Transient analysis: 2 synapses to 1 neuron](image)

**fig. 5.8: Transient analysis**

The propagation time is large, 40μs, due to the parasitic capacitances which must be charged and discharged with the small currents.
5.4 Extension with Back Propagation Algorithm

The Back Propagation algorithm can be implemented on-chip because all signals for this training algorithm are locally available (chapter 2). The extra components are given in the dotted area in figure 5.9 and 5.10 (compare with figures 3.4 and 3.5).

**Synapse**

The 4 quadrant multiplier in the synapse is implemented with a Gilbert multiplier.

*fig. 5.9: Synapse with Backward path*
The following components must be added:
- 2 Gilbert multipliers (including VCC)
- 1 normalizing cell
- 1 converter: differential current (0-200nA) to single voltage (1-4V)

This large output voltage requires large gate lengths because the multiplier uses small currents (low power).

**Neuron**

The neuron input (BPI) is a summation of all incoming currents from all synapses in the next layer. This incoming current must be normalized. The sigmoid-derivative must also be designed. But it is not clear if this function must be used. Maybe it's possible to use other functions than this derivative for the Back Propagation algorithm. Therefore system simulation must be done with other kind of functions in the Backward path to get an idea of the effect of this function.

**fig. 5.10: Neuron with Backward path**

The following components must be added:
- 1 Gilbert multiplier, including VCC
The multiplier in the Backward path must multiply a differential current BPI with a voltage or current (depends on derivative), and generates a differential output voltage. This can be done with two converters and a Gilbert multiplier. When the two input voltages have a small range, no VCC is required in the multiplier. The output voltage BPIS' must have a small voltage range (≈200mV) because it is connected with a Gilbert multiplier in the synapse.

The implementation requires 5 extra signal lines per synapse and 5 extra signal lines per neuron. The synapse contains two normalizing circuits, and the neuron contains two current sources (two PMOS transistors).
5.5 Interface Synapse-chip and Neuron-chip

The structure of the synapse and neuron without Back Propagation are already given in figures 4.25 to 4.28. The individual Weights in the synapse and the neuron must be selected during training in order to change that individual Weight. The synapse Weight can be selected by using the FPI signal and the UD signal.

The new Weight voltage is presented at UD, and the FPI signal selects the row of synapses (fig. 4.27). When FPI is "high", the Switch closes and the voltage will be presented to the Weight Storage.

The neuron Weight is loaded with the signal line eta (fig. 4.26). After training, the voltage at the input of the Weight Storage will have no effect on the Weight. The control of the network, built with synapse-chips and neuron-chips, is described by Kuijpers [9, chapter 6.3].
5.6 Specifications

Design aspects for the implementation are:
- low area
- low power dissipation
- low number of pins

The components were designed with low power dissipation (3μW per synapse and 7μW per neuron) which consumes more area than was expected in advance.

Suppose no Back Propagation on-chip.

Total estimated area synapse: (without Weight Storage) = 3430μm²
Total estimated area neuron: (without Weight Storage) = 14000μm²

The number of pins is the main bottleneck. The synapse chip with MxN synapses requires 4xM+2xN pins. When all differential signals are converted to single signals, then 3xM+N pins are required. The signals on-chip are differential.

The neuron chip with differential signals requires 5xM+1 pins.

Suppose the number of pins=140, and M=N.
This means only 22 synapses or 22 neurons per chip.

When synapses and neurons are integrated together on the same chip, then the chip density can be upgraded, because less input/output pins are required and the chip area can fully be used. The low power dissipation is more important. However, the network topology is restricted (chapter 2).

When only synapses or neurons are implemented on-chip, then the number of pins determines the number of synapses or neurons, and not the chip-area. Single signals (differential intern) could reduce the number of pins. Also multiplexing is a possible option but then the parallel behaviour is affected.
6 System Simulations with Nonlinear Components

All components of the synapse and neuron have been designed and with Monte Carlo analysis the spread in threshold voltage has been simulated. The main disturbance is caused by the normalization circuit which has to work in Subthreshold. All disturbances of the components in the synapse can be modelled by a nonlinear multiplier with offset. In figure 4.17 two bad characteristics are shown and these two plots are used for the nonlinear multiplier characteristic.

The quantized Weight is also simulated for 8 bits quantization. In order to get information in which way one of these two restrictions effects the convergence speed and the error, a problem is simulated with the following restrictions:

1) Ideal components, floating point Weights
2) Quantized Weights (8 bits)
3) Nonlinear multiplier, only in forward path, with offset
4) Nonlinear multiplier, only in forward path, with offset, and quantized Weights (8 bits)

The simulations are done with a Pascal program which is able to train all kind of multilayer perceptrons with several training algorithms. In these simulations, the Back Propagation algorithm is used. 
Remark: Only the components in the Forward path are nonlinear. The calculations for the Weight adaptation during training are ideal (floating-point, ideal multiplication).

6.1 Network problem

Only one problem is simulated to get an idea of the behaviour of a Neural Network with non ideal components. The results are dependant of the problem and the network topology. Therefore the results can't be generalized for all kind of problems.

A problem is chosen with continuous inputs and outputs and the following function must be learned:

\[ Y = 0.5 + 0.3 \cdot \sin(6 \cdot X_1) \cdot \arctan(5 \cdot (X_2 - 0.5)) \]

This 2 dimensional function has inputs X1 and X2 which vary from 0 to 1, and output Y between 0 and 1. In the next figure the "landscape" according to this function is shown.
The next table gives the output Y (in the rectangle) for several values of X1 and X2.

<table>
<thead>
<tr>
<th>input X1</th>
<th>0.0</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.5000</td>
<td>0.5000</td>
<td>0.5000</td>
<td>0.5000</td>
<td>0.5000</td>
<td>0.5000</td>
</tr>
<tr>
<td>0.2</td>
<td>0.1672</td>
<td>0.2252</td>
<td>0.3704</td>
<td>0.6296</td>
<td>0.7748</td>
<td>0.8328</td>
</tr>
<tr>
<td>0.4</td>
<td>0.2588</td>
<td>0.3008</td>
<td>0.4060</td>
<td>0.5940</td>
<td>0.6992</td>
<td>0.7412</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6588</td>
<td>0.6305</td>
<td>0.5616</td>
<td>0.4384</td>
<td>0.3695</td>
<td>0.3420</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8557</td>
<td>0.7937</td>
<td>0.6386</td>
<td>0.3614</td>
<td>0.2063</td>
<td>0.1443</td>
</tr>
<tr>
<td>1.0</td>
<td>0.5998</td>
<td>0.5824</td>
<td>0.5389</td>
<td>0.4611</td>
<td>0.4176</td>
<td>0.4002</td>
</tr>
</tbody>
</table>

The next figure shows the "landscape" with these 36 values that is learned on the networks.
Function which has to be learned

For the simulation, two Neural Networks are chosen with one hidden layer consisting of 4 and 5 neurons (fig. 6.3 and 6.4).

The quantization will have effect on the behaviour of the network during the training. Suppose there is a path through the Weight-space from the initial Weights to the Weights in a local or global convergence minima. All Weights follow a path through the Weight-space to their final Weight position. This path can be different for other initial Weights. When the Weights are quantized, not all positions in path are available for Weights. They skip from one quantized position to another. This may effect the convergence speed and finally the output error.

The calculation of the Weight adaptation $\Delta W$ during the training algorithm must be very...
accurate (many bits for quantization), but the forward path, after training, doesn't have to be very accurate, and in literature 8 bits of quantization seems to be adequate [13].

In this simulation the Weights are quantized with 8 bits (256 levels), and the range is chosen from -10 to +10. This range is chosen after a simulation with ideal components. That simulation shows that all Weights doesn't exceed this range.

6.3 Model of Nonlinear Multiplier

In paragraph 4.5 the characteristic of the multiplier has been derived. This formula is used in this simulation. With Monte Carlo analysis, the offset has been derived and this is also taken into account. The input ranges and the output range of the multiplier must be adapted to the nonlinearity, used in the simulation program.

The output range of the sigmoid must fit with an input range of the multiplier. The other input of the multiplier must fit with the Weight range.

In figure 6.5 the used sigmoid (by Pascal program) is shown.

![Sigmoid nonlinearity used in Pascal Simulation](image)

The sigmoid input is chosen from -8 to +8 but can also chosen larger. This decision is disputable, but it is convenient for getting an idea about the behaviour of nonlinear components in a neural network.

The electronic multiplier has two inputs:

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Represents Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1: 0mV ... 200mV</td>
<td>0 ... 1</td>
</tr>
<tr>
<td>V2: -1.5V ... +1.5V</td>
<td>-8 ... +8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
<th>Represents Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iout: -200nA ... +200nA</td>
<td>-8 ... +8</td>
</tr>
</tbody>
</table>
Weight range:
-1.5V ... +1.5V represents value: -10 ... +10

The sigmoid has the following input and output:
input
A: -200nA ... +200nA represents value: -8 ... +8
output
FPO: 0mV ... 200mV represents value: 0 ... 1

The input ranges and the output range of the modeled multiplier correspond with the range of the Weight and the sigmoid. The next equation is used for the multiplier model.

\[ p(V_1, V_2) = V_1 \cdot V_2 \cdot k_1 \cdot \sqrt{1 - k_2 \cdot V_2^2} \]  

(eq. 6.1)

Find the extremes and calculate constants for the specified output range.

\[ \frac{\partial p(V_1, V_2)}{\partial V_2} = 0 \Rightarrow \text{find max} \Rightarrow V_{2_{\text{max}}} = \sqrt{\frac{1}{2 \cdot k_2}} \Delta 8 \Rightarrow k_2 = 0.0078 \]

Substitution: \[ p_{\text{max}}(V_{1_{\text{max}}}, V_{2_{\text{max}}}) = \frac{V_{1_{\text{max}}} \cdot k_1}{2 \cdot \sqrt{k_2}} \Delta 8 \]

\[ V_{1_{\text{max}}} = 1 \Rightarrow k_1 = 1.41 \]

The characteristic is plotted in figure 6.6.
fig. 6.6: Modelled nonlinear multiplier without offset

There are five curves plotted for different values of $V_1$: 0.00, 0.25, 0.50, 0.75 and 1.00. The voltage $V_2$ varies from -8 to +8 and the output $p(V_1,V_2)$ has a range from -8 to +8.

The $V_1$-offset is about 1/8 of the input range $V_1$ and the output-offset is about 1/20 of the whole output range (fig. 4.17). In the next figure the offset is modelled.

fig. 6.7: Modelled nonlinear multiplier with offset
The multiplier in the Pascal program is modelled with this multiplier with offset. This multiplier model is not an exact transformation of the implemented multiplier. The output current of the designed multiplier can’t exceed the value 200nA, delivered by the current sources within the VCC.

The used model (eq. 6.1) doesn’t deal with this restricted output current. When the output-offset is chosen very large, the output of this model will exceed the allowed upper limit of +8. But this effect will not occur with a small output-offset.

### 6.4 Simulation results

In the next table, the mean square error is presented after 500 train cycles (1 train cycle: all 36 patterns have been presented in strict order and the Weights are adjusted after every pattern), for a 2-4-1 and a 2-5-1 network.

This is repeated 5 times (run A to run E).

<table>
<thead>
<tr>
<th></th>
<th>2-4-1 Network</th>
<th></th>
<th>2-5-1 Network</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>run A</td>
<td>run B</td>
<td>run C</td>
</tr>
<tr>
<td>qnl</td>
<td>-10.63</td>
<td>-12.49</td>
<td>-12.06</td>
</tr>
</tbody>
</table>

i: Ideal components
q: Weights quantized, 8 bits
nl: Nonlinear multiplier in Forward Path
qnl: Weights quantized (8 bits), and nonlinear multiplier in Forward Path

In the next figure, the output error against the batch number is plotted for Run A on a 2-4-1 network.
6.5 Conclusions System Simulations

The Back Propagation Algorithm is simulated with ideal components and floating-point calculations. So only the components in the Forward Path are non-linear. When the Weights are quantized, the output error will change a little (0.3 dB) in comparison with the ideal simulation. Also when only a nonlinear multiplier is used in the Forward path, the final error is about the same as in the ideal case. The convergence speed is slower by all runs. When the quantization is combined with the non ideal multiplier, the error will be larger (fig. 6.8 lower right plot).

More simulations with various problems and different kind of networks should be done to obtain a better overview of the behaviour of the train algorithm, due to nonlinear components.
Conclusions

Some implementations with Pulse Stream signals and Continuous signals are compared, although it is difficult to predict the required area for each representation. The continuous approach with differential signals is worked out in this thesis. The Pulsed Stream version is worked in another thesis [25]. Advantages in comparison to Pulsed signals are a small neuron area and low crosstalk. The power dissipation is decreased by using small currents.

The main component of the synapse and neuron, the multiplier, should be designed with a low power dissipation. There is tried to design a multiplier in Subthreshold. Because of the small dynamic input ranges of such multiplier (+/- 100mV) two linearizing circuits are required for enlarging this range. These circuits can not be designed in Subthreshold. When this multiplier works with larger currents in Strong Inversion, then the power dissipation could still be smaller than with pulsed signals and a larger dynamic range can be achieved. This means a smaller linearize circuit and smaller area.

The multiplier is based on a Gilbert multiplier with MOS transistors, and works in Strong Inversion. Because of the small currents (600nA), flowing through the multiplier, the power dissipation of this multiplier is very low: ±3μW. The dynamic range of the multiplier is enlarged by linearizing the nonlinear characteristic of the Gilbert multiplier. This is done with a Voltage-to-Current Converter. The linear range of the VCC can be enlarged by:
- extending with more transistor pairs (distortion decreases quadratic)
- enlarging gate length transistors
- using larger currents

The multiplier (Full Scale) nonlinearity is 20% in the voltage range from -1.5V to +1.5V and 5% in the range from -1.0V to +1.0V. Simulations have shown that this nonlinearity of the multiplier in the forward path affects the convergence speed of the network in comparison with an ideal multiplication. But the output error is hardly affected. Quantized Weights (8 bits) have little effect on convergence speed. But a combination of these two non linearities lead to a larger output error (-10dB instead of -15dB for ideal network). More investigation on nonlinear components in the Back Propagation algorithm is necessary. In advance it is not clear if a strict linear multiplication is necessary for a good working of the neural network, so the multiplier nonlinearity is not decreased further.

The signals, which are transported differential, are less sensitive to noise than single signals, but they consume more area, and they require twice as many input/output pins. The main bottleneck seems to be the number of IC-pins, which is limited. When only synapses or neurons are integrated on a chip, all kind of network structures can be created. Drawback of this approach is that all inputs and outputs of synapses and neurons must be available at the outside of the synapse/neuron chip. When the continuous signals between synapse and neuron are multiplexed, then many pins can be saved, but this affects the parallel behaviour of the network, and requires multiplexing/demultiplexing techniques, circuits and timing signals.
Conclusions

The simulated mismatch in threshold voltage of 20mV affects the output behaviour of the synapse and neuron. The major disturbance is caused by the normalization circuit which works in Subthreshold Operation. When no variation is assumed, then the normalization works very good but a little threshold mismatch destroys the nice behaviour. The mismatch can be minimized by a smart design: transistors must be placed very close so the fluctuating parameters are almost the same for both transistors.

The forward propagation time of a synapse, connected to a neuron, is about 40µs. This long time is the result of the small currents which are used in order to get a low power dissipation. These small currents have to charge and discharge all parasitic capacitances. The speed can be upgraded by using larger currents, but also more power will be dissipated.

The synapse and neuron are designed without Back Propagation algorithm on-chip, but it is possible to add the sufficient components which are not designed yet. The estimated area of the synapse and neuron is respectively: 3428µm² and 14000µm². The synapse dissipates 3μW and the neuron dissipates 7μW. It is possible to create a working neural network with differential continuous signals but this approach has some limitations.
8 Recommendations

★ The Back Propagation algorithm is not implemented further and it is useful to investigate whether nonlinear components affect the training algorithm. It is also fruitful to look deeper at the sigmoid derivative, which determines the Weight change during training. When other training algorithms are used, then it is important that all used signals must be available locally on-chip.

★ Differential continuous signals consume a lot of pins so when all signals are converted to single signals, twice as many synapses or neurons can be integrated on-chip. Also multiplexing techniques could be examined further in order to increase the number of synapses or neurons that on-chip. Other signal types, like pulsed modulated signals may give better behaviour than the continuous differential approach, and it is fruitful to investigate this approach deeper.

★ The nonlinearity of the multiplier is large and can be made smaller. When the currents are increased, then all components must be redesigned for the larger current range. When only the VCC is extended, the nonlinearity decreases but the required area will increase. Maybe it is possible to combine the nonlinear characteristic of the neuron with the synapse-multiplier in the following layer. Then the multiplier should not be linearized.

★ Because normalization is necessary when an extendable network is made, further investigation to other normalization techniques is recommended.

★ It seems that increasing of the current, flowing through the multiplier, is a good option in order to obtain a global better behaviour of all components: smaller nonlinearity of the multiplier, less sensitivity due to threshold mismatch and a smaller forward propagation time. However more power will be dissipated.
Literature

[1] Andreou, A.G.

A 20-V Four-Quadrant CMOS Analog Multiplier, IEEE Journal of Solid-State Circuits, SC-20, no.6, dec. 1985, pp.1159-1168


[4] Carlson, A. Bruce


[6] Gilbert, B.


Electronic Implementation of Multilayer Neural Networks including Back Propagation Training Algorithm, Masters thesis Eindhoven, april 1992

[10] Lima, J.A. de
An Introduction to Computing with Neural nets, IEEE ASSP-Magazine, april 1987, p.4-22


[16] Schwartz, D.B. and Samalam, V.K.

[17] Stotts, L.J.

[18] Teulings, P.M.W.
Analog Electronic Implementation of Multilayer Neural Networks including Back Propagation Training Algorithm, Masters thesis Eindhoven, aug 1991

CMOS Voltage to Current Transducers, IEEE Trans. on Circuits and Systems vol.32, nov 1985, pp.1097-1104


[21] Vittoz, E.A.

[22] Vittoz, E.A.
[23] Warren, G.H. and Jungo, C.


[25] Claasen-Vujicic, T.
Masters thesis Eindhoven, to be written
Appendix I. PSPICE parameters

PSPICE parameters used in the simulations in accordance to the information from IMEC.

******************************************************************************

** SPICE parameters 2/26/92
* Leff = Ldesign -2x(LD + DEL)
* SPICE and PSPICE correct for LD, use only:
* Leff = Ldesign -2xDEL
* Remark: LD must be given in the model parameters!
*
* Weff = Wdesign -2xWD
* PSPICE correct for WD, SPICE APOLLO does not correct for WD
* Use for PSPICE:
* Weff = Wdesign
*
*
* NMOS: Vearly = 6.6V/um, DEL = -0.11
* PMOS: Vearly = 11V/um, DEL = 0.05
*
* lambda = 1/(Leff*Vearly)
* pd = ps = 2*Wdesign + 18um
* ad = as = Wdesign * 9um
******************************************************************************

Conductance parameter: $S = \frac{C_{ox}\mu W}{2L} = \frac{K_p W}{2 \frac{W}{L}}$

NMOS: $K_p = 57 \cdot 10^{-6} \text{ A/V}^2$
PMOS: $K_p = 17 \cdot 10^{-6} \text{ A/V}^2$
Appendix II. Variations in Threshold Voltages

With Monte Carlo analysis, threshold voltages are changed within a certain range. In order to get realistic simulations, a variation in threshold voltages is simulated.

The measured threshold voltages from IMEC over the years 1986 till 1989 results in the next values:

PMOS: \( V_{th} = -0.80 \text{V} \pm 60 \text{mV} \)
NMOS: \( V_{th} = 0.90 \text{V} \pm 60 \text{mV} \)

This voltage variation can be divided in two forms. The first is a global spread in threshold voltage: every chip, consisting of thousands of transistors, has a different mean threshold voltage. The threshold voltages of two (the same type) transistors on the same chip are in this case the same. This kind of variation hardly effects the final behaviour of the circuits. This variation is about 50mV.

The second spread can be disastrous for the behaviour of the circuits. This is the variation in threshold voltage between transistors on one chip. IMEC doesn’t give this variance but a (maybe pessimistic) guess is 20mV. In worst case the threshold voltages of two neighbour transistors differ 20mV. The total variation is 60mV.

In PSpice Monte Carlo analysis the following parameters are used:

PMOS: \( V_{th} = -0.80 \text{V} \) LOT/0=50mV DEV=10mV
NMOS: \( V_{th} = 0.90 \text{V} \) LOT/0=50mV DEV=10mV
Appendix III. PSPICE Subcircuits

* Subcircuit Gilbert multiplier including VCC

```
*.inc vcc3.sub

.SUBCKT GILBERT_MULTIPLIER 1 2 9 8 6 5 10 11
* call: V1+, V1-, V2+, V2-, Iout+, Iout-, Vpmos, Vnmos

.model n2.4/24 nmos(level=2 vto=0.9 lot/0=50mV dev=10mV kp=17u gamma=0.5 phi=0.69
+w=2.4u l=24.22u lambda=0.006 cgso=280p cgdo=280p rsh=45 cj=330u
+mj=0.5 cjsw=440p mjsw=0.33 js=1m tox=42.5n nfs=1e11 ld=0.35u
+xj=0.5 ucrit=1e4 delta=0.6 uo=234)

* 4 quadrant Gilbert multiplier, including VCC

** circuit **

m1 6 1 3 0 n2.4/24 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m2 5 2 3 0 n2.4/24 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m3 6 2 4 0 n2.4/24 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m4 5 1 4 0 n2.4/24 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u

** VCC **

X1 9 8 3 4 10 11 VCC

.ENDS GILBERT_MULTIPLIER
```

---

**Diagram:**

- A 4-quadrant Gilbert multiplier including VCC.
- Connections labeled: V1+, V1-, V2+, V2-, Iout+, Iout-, Vpmos, Vnmos.
- Circuit components with specific values.

---

* Appendix III. PSPICE Subcircuits*
* Subcircuit of 3 stage VCC

** Linear Voltage-to-Current Converter with 3 stages

** converter **

<table>
<thead>
<tr>
<th>m1</th>
<th>8</th>
<th>1</th>
<th>5</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>m2</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>m3</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>m4</td>
<td>4</td>
<td>4</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>m5</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>m6</td>
<td>9</td>
<td>2</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

** current sources **

<table>
<thead>
<tr>
<th>m7</th>
<th>7</th>
<th>1</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>m8</td>
<td>5</td>
<td>12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>m9</td>
<td>6</td>
<td>12</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Vdd 10 0 dc 5V

.ENDS VCC
* Subcircuit of 3 stage VCC with PMOS input stages

```
.SUBCKT VCCP 1 2 9 8 11 12
* call: V+, V-, I+i, I-i, Vpmos, Vnmos

.model n2.4/24 nmos(levell=2 vto=0.9 lot/0=50mV dev=10mV kp=57u gamma=0.3 phi=0.7
+w=2.4u l=24.22u lambda= 0.006 cgso=180p cgdo=180p rsh=25 cj=70u
+mj=0.5 cjsw=390p mjsw=0.33 js=1m tox=42.5u nfs=1e11 ld=0.22u
+xj=0.3u ucrit=1e4 delta=1 uo=614)

.model n2.4/48 nmos(levell=2 vto=0.9 lot/0=50mV dev=10mV kp=57u gamma=0.3 phi=0.7
+w=2.4u l=48.22u lambda=0.003 cgso=180p cgdo=180p rsh=25 cj=70u
+mj=0.5 cjsw=390p mjsw=0.33 js=1m tox=42.5u nfs=1e11 ld=0.22u
+xj=0.3u ucrit=1e4 delta=1 uo=614)

.model p2.4/24 pmos(levell=2 vto=0.8 lot/0=50mV dev=10mV kp=17u gamma=0.5 phi=0.69
+w=2.4u l=23.9u lambda=0.004 cgso=280p cgdo=280p rsh=45 cj=330u
+mj=0.5 cjsw=440p mjsw=0.33 js=1m tox=42.5u nfs=1e11 ld=0.35u
+xj=0.5u ucrit=1e4 delta=0.6 uo=234)

.model p2.4/48 pmos(levell=2 vto=0.8 lot/0=50mV dev=10mV kp=17u gamma=0.5 phi=0.69
+w=2.4u l=47.9u lambda=0.0019 cgso=280p cgdo=280p rsh=45 cj=330u
+mj=0.5 cjsw=440p mjsw=0.33 js=1m tox=42.5u nfs=1e11 ld=0.35u
+xj=0.5u ucrit=1e4 delta=0.6 uo=234)

*---------------------------------------------------------------*
* Linear Voltage-to-Current Converter with 3 stages
*---------------------------------------------------------------*

** converter **
ml 8 1 3 10 p2.4/48 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m2 6 6 3 10 p2.4/48 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m3 6 6 5 0 n2.4/48 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m4 7 7 5 0 n2.4/48 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m5 7 7 4 10 p2.4/48 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m6 9 2 4 10 p2.4/48 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u

** current sources **
m7 3 11 10 10 p2.4/24 l=8u as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m8 4 11 10 10 p2.4/24 l=8u as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m9 5 12 0 0 n2.4/24 l=8u as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u

Vdd 10 0 dc 5V
.ENDS VCCP
```

---

** Appendixes **
Subcircuit Sigmoid load

.SUBCKT SIGMOID 2 3 5 6 7
* call: lin+, lin-, Vout+, Vout-, Vbias
.model n2.4/24 nmos(level=2 vto=0.9 lot/0=50mV dev=10mV kp=57u gamma=0.3 phi=0.7
+w=2.4u l=2.62u lambda= 0.058 cgso=180p cgdo=180p rsh=25 cj=70u
+mj=0.5 cjsw=390p mjsw=0.33 j=1m tox=42.5n nfs=1e11 ld=0.22u
+xj=0.3u ucrit=1e4 delta=1 uo=614)
.model n2.4/24/24 nmos(level=2 vto=0.9 lot/0=50mV dev=10mV kp=57u gamma=0.3 phi=0.7
+w=2.4u l=24.22u lambda= 0.006 cgso=180p cgdo=180p rsh=25 cj=70u
+mj=0.5 cjsw=390p mjsw=0.33 j=1m tox=42.5n nfs=1e11 ld=0.22u
+xj=0.3u ucrit=1e4 delta=1 uo=614)
.model n2.4/48 nmos(level=2 vto=0.9 lot/0=50mV dev=10mV kp=57u gamma=0.3 phi=0.7
+w=2.4u l=48.22u lambda= 0.003 cgso=180p cgdo=180p rsh=25 cj=70u
+mj=0.5 cjsw=390p mjsw=0.33 j=1m tox=42.5n nfs=1e11 ld=0.22u
+xj=0.3u ucrit=1e4 delta=1 uo=614)
.model p2.4/2.4 pmos(level=2 vto=0.8 lot/0=50mV dev=10nV kp=17u gamma=0.5 phi=0.69
+w=2.4u l=2.3u lambda= 0.040 cgso=280p cgdo=280p rsh=45 cj=330u
+mj=0.5 cjsw=440p mjsw=0.33 j=1m tox=42.5n nfs=1e11 ld=0.35u
+xj=0.5u ucrit=1e4 delta=0.6 uo=234)

**********************************************************************
* Load with "Sigmoid" characteristic

** circuit **
m1 2 2 0 0 n2.4/48 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m2 3 3 0 0 n2.4/48 l=41.6u as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m3 5 3 4 0 n2.4/2.4 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m4 6 2 4 0 n2.4/2.4 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m5 5 5 1 1 n2.4/2.4 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m6 6 6 1 1 n2.4/2.4 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m7 4 7 0 0 n2.4/2.4 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
m8 6 7 0 0 n2.4/2.4 l=3.2u as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u

Vdd 1 0 dc 5V
.ENDS SIGMOID
* Subcircuit Normalizing Cell

```
..SUBCKT NORMCEL 1 2 3 4 5 6
* call: I+i, I-i, (I+i)out, (I-i)out, In, Vnorm
.model p2.4/2.4 pmos(level=2 vto=-0.8 lot/0=50mV dev=10mV kp=17u gamma=0.5 phi=0.69
+w=2.4u l=2.3u lambda=0.040 cgso=280p cgdo=280p rsh=45 cj=330u
+mj=0.5 cjsw=440p mjsw=0.33 js=1m tox=42.5n nfs=1e11 ld=0.35u
+xj=0.5u ucrit=1e4 delta=0.6 wo=234)

* One normalizing cell

** circuit **
m1 1 1 6 6 p2.4/2.4 w=28u
m2 2 2 6 6 p2.4/2.4 w=28u
m3 3 1 5 6 p2.4/2.4 w=28u
m4 4 2 5 6 p2.4/2.4 w=28u

.ENDS NORMCEL
```

---

** Circuit Diagram **

```
+-----------------+------------------+
|                  |                  |
|                  |                  |
|                  |                  |
|                  |                  |
|                  |                  |
|                  |                  |
+-----------------+------------------+
```

---

** Normalizing Cell Diagram **

```
6 Normalizing cell

1 2 3 4

I+i I-i I+i I-i

Vnorm In

5

page 96
```
* Subcircuit SYNAPSE

```
.SUBCKT SYNAPSE 1 2 3 4 5 6 7 12 10 11
* call: V+ (FPI), V- (FPI), Vweight+, Vweight-, In, Iout+ (A), Iout- (A),
* Vnorm, Vpmos, Vnmos

* One synapse WITHOUT Back Propagation
```

** circuit **

X1 1 2 3 4 8 9 10 11 GILBERT_MULTIPLIER
X2 8 9 6 7 5 12 NORMCEL

.ENDS SYNAPSE
* Subcircuit NEURON

```plaintext
*.inc vcc3p.sub
.inc sigfd.sub

.SUBCKT NEURON 1 2 3 4 5 6 7 8 9 10
* call: Iin+, Iin-, Vweight+, Vweight-, V+ (FPO), V- (FPO),
* Vbias, Vpmos, Vnmos, 2lt
.model p2.4/24 pmos(level=2 vto=-0.8 lot/0=50mV dev=10mV kp=17u gamma=0.5 phi=0.69
+w=2.4u l=23.9u lambda=0.004 cgso=280p cgdo=280p rsh=45 cj=330u
+mj=0.5 cjsw=440p mjsw=0.33 js=1m tox=42.5n nfs=1e11 ld=0.35u
+xj=0.5u ucrit=1e4 (delta=0.6 uo=234)

One neuron WITHOUT Back Propagation

** circuit **
m1 10 8 11 11 p2.4/24 as=2.16e-11 ad=2.16e-11 ps=22.8u pd=22.8u
X1 3 4 1 2 8 9 VCC3P
X2 1 2 5 6 7 SIGMOID
Vdd 11 0 dc 5V

.ENDS NEURON
```

---

**Diagram:**

- **Neuron:**
  - Inputs: I+I, I-I
  - Outputs: V+ (FPO), V- (FPO), Vbias
  - Inhibit: Vweight±, Vbias
  - Voltage Levels: VCC3P, VDD, GND, SIGMOID

---

---