UNIVERSITY
OF OSLO

MES
Mikro Elektroniske Systemer

Neuromorphic Electronics

Lecture Notes

Fall Term 2006

Department of Informatics

University of Oslo

read by

Philipp D. Häfliger

Abstract

This is the script for an introductory course in neuromorphic electronic circuits. These are circuits inspired by the nervous system that either help verifying neuro-physiological models, or that are useful components in artificial perception/action systems. Research also aims at using them in implants. These circuits are computational devices and intelligent sensors that are very differently organized than digital processors. Their storage and processing capacity is distributed. They are asynchronous and use no clock signal. They are often purely analog and operate time continuous. They are adaptive or can even learn on a basic level instead of being programmed. A short introduction into the area of brain research is also included in the course.

The students will learn to exploit mechanisms employed by the nervous system for compact energy efficient analog integrated circuits. They will get insight into a multidisciplinary research area. The students will learn to analyze analog CMOS circuits and acquire basic knowledge in brain research methods.

Contents

List of Figures

Chapter 1

Introduction

The term 'neuromorphic' was introduced by Carver Mead around 1990. He defined neuromorphic systems as artificial systems that share organization principles with biological nervous system. So what are those organization principles?

A brain is fundamentally differently organized than a computer and science is still a long way from understanding how the whole thing works. A computer is really easy to understand by comparison. Features (or organization principles) that clearly distinguish a brain from a computer are massive parallelism, distributed storage, asynchronous processing, self organization. Neuromorphic circuits try to incorporate those principles. Whereas a computer by contrast is a synchronous serial machine, with centralized storage and it is programmed. Table 1.1 compares these different organizing principles.

Research in neuromorphic aVLSI can lead to computing machines that are fundamentally differently organized than computers. Machines that best computers at tasks that humans are better at than computers. This can be achieved by more closely copying the biological pendant. And although computers are general machines that can simulate any other digital machine and arbitrarily approximate analog processes, neuromorphic machines can outrank them in terms of speed, energy efficacy, and size.

1.1 Neuromorphic Circuits at Present

Our understanding of the nervous system can be considered good in the peripheral parts only: Many sensors and their low level processing and muscles and their enervation were exhaustively

Computer	Brain
Serial	Parallel
One powerful central CPU, memory	10^{11} simple distributed computational and memory units
Busses shared by several components	Dedicated local point to point connections
Not very power efficient (needs cooling)	Very power efficient (hair to keep it warm ;-))
Digital, time-discrete	Analog, continuous time
Programmed	Learning
Sensitive to errors	Robust to errors (using redundancy)

Table 1.1: Organizing principles of computers and the nervous system

explored. So it comes as no surprise that neuromorphic electronics that mimics those parts is the most successful. Especially visual and auditory 'intelligent' sensors came out of that line of research.

Chapter 2

Neurophysiology in a Nutshell

Understanding the brain is one of the biggest remaining challenges to science. Scientists are very busy scratching together bits and pieces of knowledge about the nervous system and in fact the data acquired is enormous. But how all the parts work together to solve the fuzzily defined tasks in our daily lives is still a mystery.

In this chapter we try to give an impression of today's brain research. To the neuromorphic engineer the most interesting subfield is neurophysiology, where the nervous system is observed in action. So at first an overview on the methods used for these observations is given, followed by some examples of the knowledge that has been gained by these methods.

2.1 Selected Methods in Neurophysiological Research

The different observation methods that are applied in neurophysiology do all have strengths and weaknesses. There is always a trade off between resolution and the total observed area/volume. Table 2.1 gives a short summary of the methods that are discussed some more in the following.

method	test subjects	observation area	order of temp. res.	order of spat. res.
Psycho Physical Exp.	alert humans			
EEG	alert humans	patches of brain surface	ms	cm^2
fMRI and PET	alert humans	brain cross-sections	40ms	5mm^3
Extra Cellular Electrodes	alert test animals	a neighbourhood of neurons	μs	$100^3\mu$m^3
Intra Cellular Electrodes	anesthetized test animals, slice preparations	one neuron	μs	$10^3\mu$m^3
Fluorescent Tracers	anesthetized test animals, slice preparations	a dendritic tree	?	$< \mu$m

Table 2.1: Summary on some methods in neurophysiological research

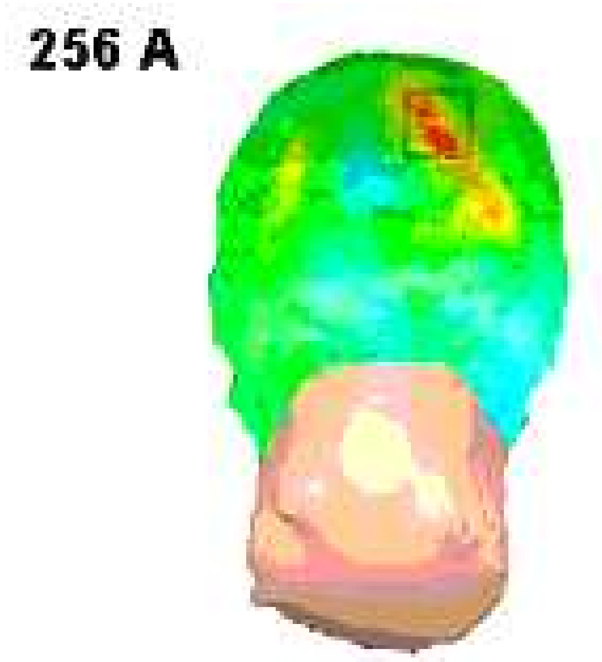


Figure 2.1: Reconstruction of correlated brain activity on the surface of the cortex with data from a 256 surface electrode array (EEG) [48]

2.1.1 Psychophysical Experiments

Psychophysical experiments constitute the most holistic approach to brain research. Human subjects are tested for reactions to all kinds of stimuli. For example reaction times and error rates in visual recognition tasks.

2.1.2 EEG

EEG applies surface electrodes to the human test subject's skin on the skull. Thus, minuscule changes in potential can be measured that are related to correlated neuron activity in the underlying cortical area. A certain spatial resolution can be achieved by placing multiple electrodes on the skull. In the example in figure 2.1 is a reconstruction of activity measured by a net of 256 electrodes.

Output of individual electrodes is very detailed with a good temporal resolution. Example traces are depicted in figure 2.2. These traces are the result of the averaged activity of a huge number of neurons though, and only correlated activity can be observed. (Uncorrelated activity will just be erased by the averaging.)

2.1.3 fMRI and PET

PET and fMRI are methods that allow to observe processes in the brain as a whole. They are also quite unobtrusive and are thus used on human test subjects. Both measure 'brain activity' indirectly, i.e. they measure observables that can be related to 'brain activity', such as oxygen or sugar levels. PET uses radioactive tracers, e.g. labeled oxygen or glucose. fMRI detects for

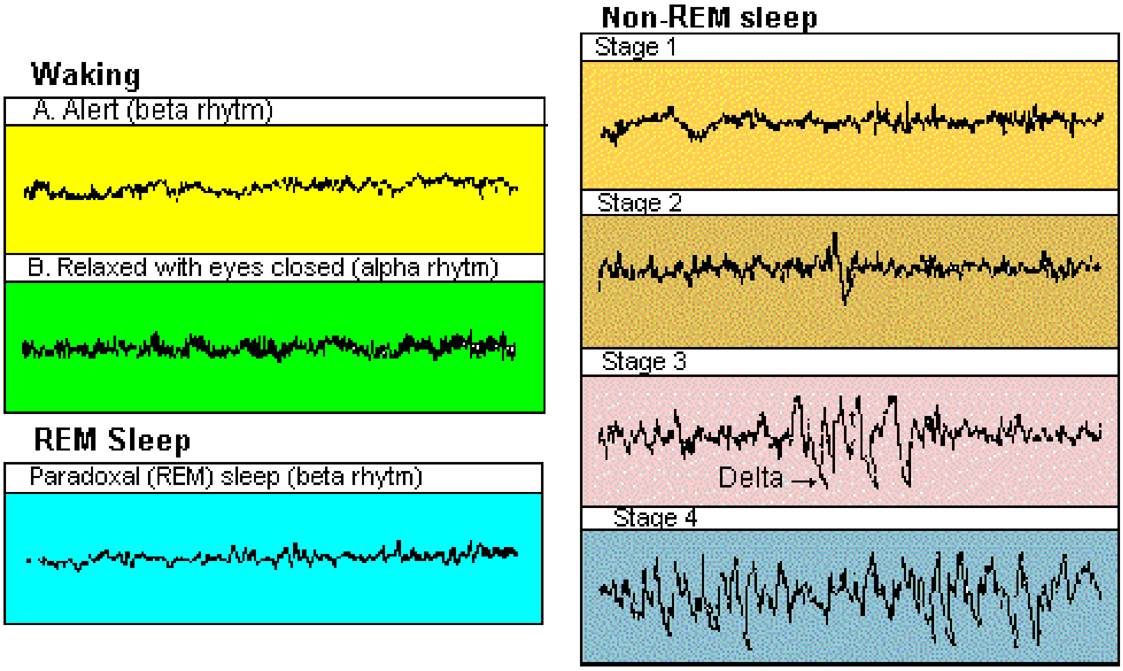


Figure 2.2: Brain EEG activity in different sleep stages [8]

example oxygen level in blood and/or blood-flow. An example of the kind of data gathered by fMRI is shown in figure 2.3.

2.1.4 Extracellular Electrodes

Much more detailed data can be gained by measuring intrusively in the brain tissue. Electrodes that lie inbetween the neurons pick up changes of potential that are caused by production of action potentials. With appropriate post-processing of the data, action potentials of single neurons can be isolated. That works especially well with placement of multiple electrodes. These kind of measurements require serious surgery and are so far only performed on lab animals (in vivo) or in brain tissue preparations (in vitro).

2.1.5 Intracellular Electrodes

Even more local details can be recorded by actually penetrating the cell membrane of neurons. This can be achieved by sharp electrodes, even in immobile anesthetized lab animals. An instrument that is less damaging to the cell is the patch clamp electrode, that can be patched upon the cell membrane and that cuts out a hole causing minimal leakage of extracellular liquid into the cell. This electrode however requires even more precise placing and has thus so far only been used in slice preparations. Also only a small number of those electrodes can be placed upon a cell simultaneously.

Figure 2.4 is an example of a phase contrast microscopy picture of two patch clamp electrodes that are placed on two cell bodies of interconnected cells. A trick that requires days or weeks of patience from the experimenter.

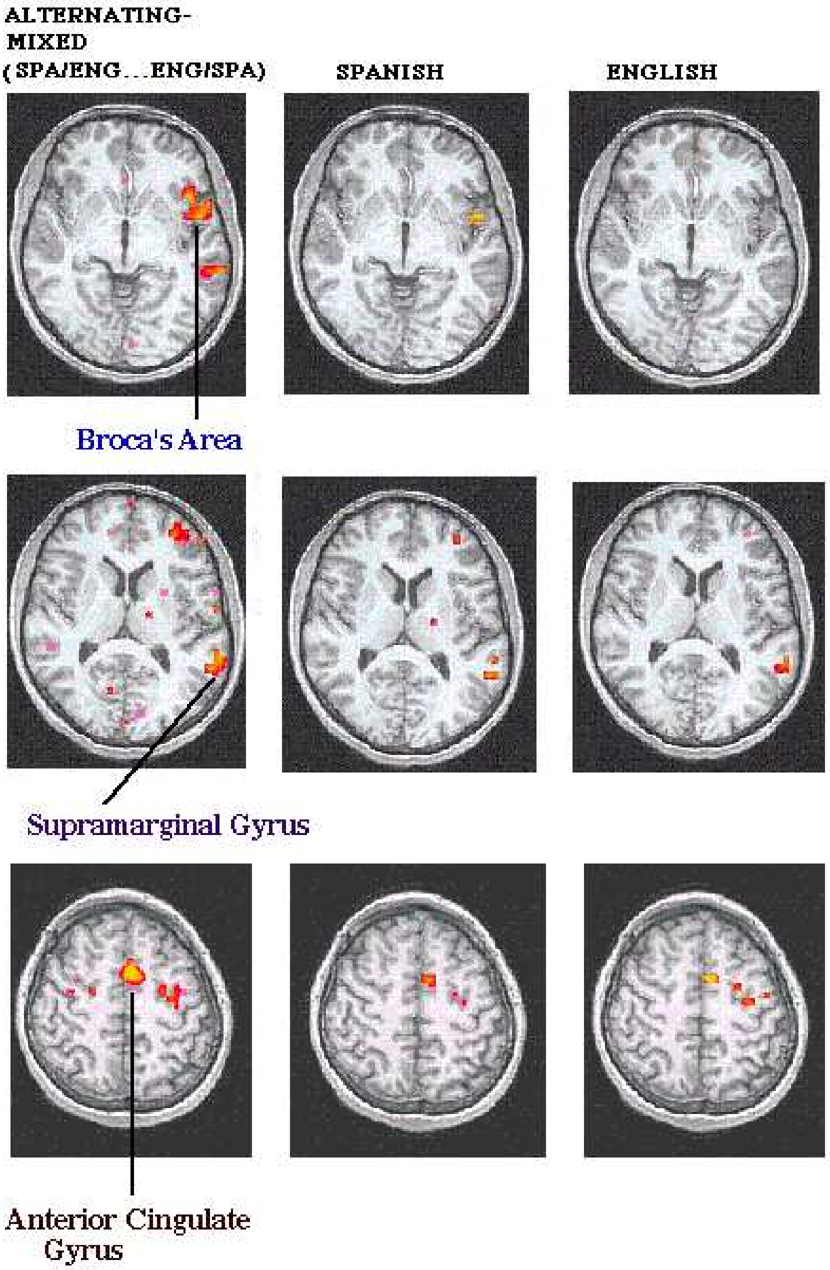


Figure 2.3: fMRI in bilingual task: picture naming [21]

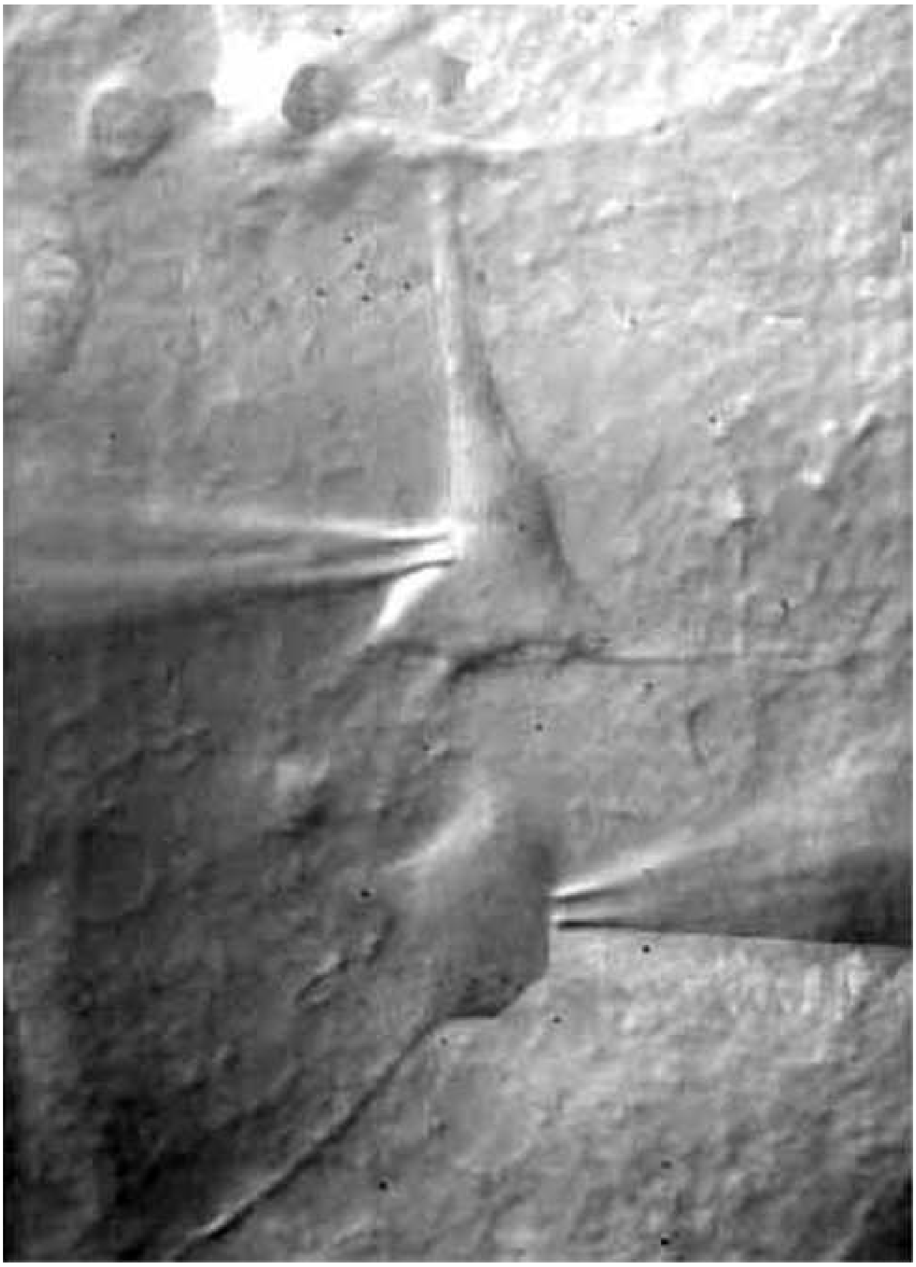


Figure 2.4: Patch clamp electrodes onto two neurons in in vitro preparation [53]

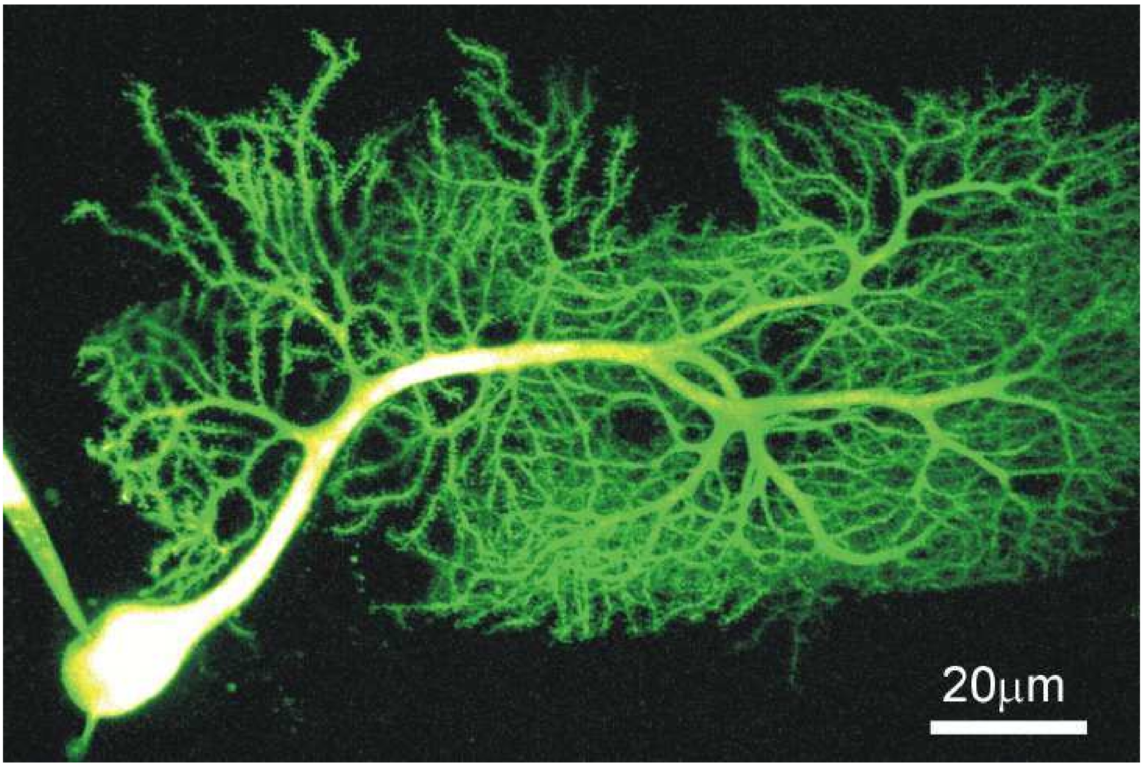


Figure 2.5: Purkinje cell with fluorescent tracer in two photon microscope [10]

2.1.6 Fluorescent Tracers and Imaging

Some dynamics in neurons can be observed with help of fluorescent tracers. Two photon microscopy has received some press in that context, e.g. in observing calcium dynamics inside a dendrite. Research goes in explores possibilities of multi-photon microscopy. This method offers excellent spatial resolution. It can be used in slice preparations and even on anesthetized animals. With the help of fluorescent tracers e.g. observations of calcium dynamics can be made, with excellent spatial resolution, very good temporal resolution, for an area of observation of up to a complete dendritic tree.

2.1.7 Briefly Mentioned: Methods in Neuroanatomy

We did not discuss methods that reveal purely anatomical information. There would still be a lot to be said about methods in microscopy in combination with various tracers and stainers that colour particular structures. Like antibodies that attach to particular ion channels, or chemicals that spread through a whole neuron cell body, including the complete dendritic and axonal tree.

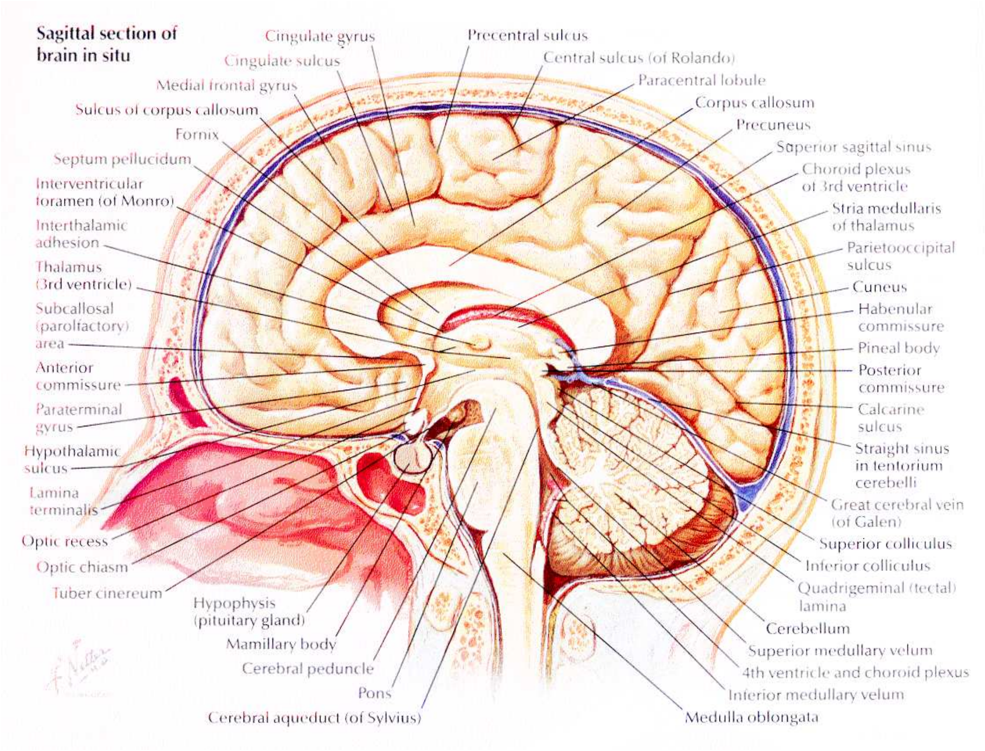


Figure 2.6: A drawing of a crossection of the brain according to [40]

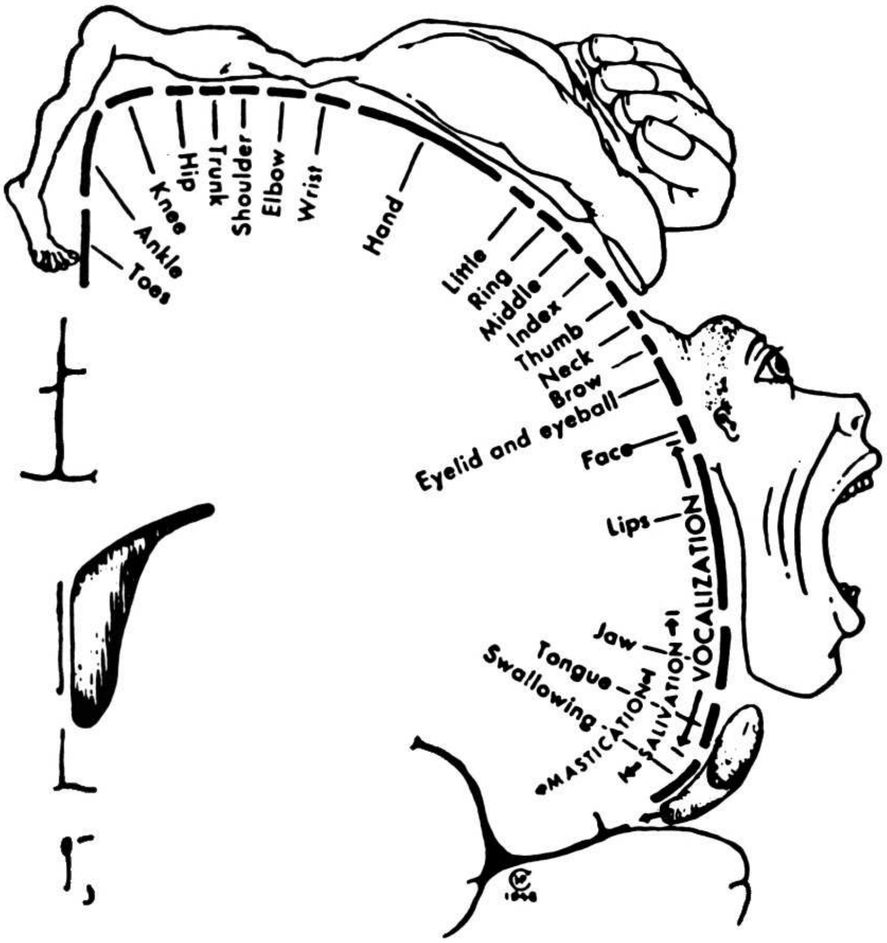


Figure 2.7: The famous Motor-Homunculus representing the body parts on motor cortex [43]

2.2 Selected Neurophysiological and Neuroanatomical Knowledge

2.2.1 Brain Anatomy

A good impression of the brains structure and its parts with their Latin names is shown in figure 2.6. Of particular interest to neurophysiology in the recent years is the cortex, the most 'modern' part of the brain, believed to host the 'higher' brain functions, such as conscious thought. It is thoroughly explored and divided into functional subregions as shall be further discussed in the following.

2.2.2 Cortical Regions

Activity in many cortical areas is known to be correlated with particular body functions or perceptions. Thus, one can identify for example a visual cortical region, a motor cortex, or sensory-motor cortex. It is also known in many instances that there is a topological mapping between sensors

"Whoa! That was a good one! Try it, Hobbs—just poke his brain right where my finger is."

Figure 2.8: An illustration that is not far from the truth by Gerry Larson

or actuators and subsequent brain areas, i.e. neighbourhood relations are preserved. The motor-cortex, for example, has been mapped early by stimulating regions of patients in brain surgery and observing the motor reactions. A famous illustration of these findings is the so called motor-homunculus (figure 2.7). An equivalent picture exists for sensory-motorcortex. An illustration that is not too far from the truth of the method applied to get these results can be found in figure 2.8.

The most explored cortical area is the visual cortex. In primates processing of visual data seems to involve a substantial part of the cortical area. Sub areas have been mapped out by presenting anesthetized lab animals with particular optical stimuli and measuring resulting activity with electrodes. Some of them are involved in perception of motion, of colour, of low level optical features, or of high level objects. Again topological mapping (this time of the visual field) is maintained from one region to the next. In a famous publication D. C. van Essen, C. H. Anderson and D. J. Felleman [56] summarized all known visual cortical regions for macaque monkeys (figure 2.9) and ordered them in a hierarchy (figure 2.10). The hierarchy is based on anatomical studies that explored which regions have direct connections to which other regions. This can for example by done by injecting colored tracer chemicals that spread through the axons into one region. Then it will be revealed where those axons go.

2.2.3 Organization within Cortical Regions

The use of staining tracers also gave insight into some of the intra regional organization of the cortex. For example in region V1 it has been shown that there is subregions that receive input dominantly from one eye. These connection patterns are known as ocular dominance patterns (figure 2.11 by [26] and 2.12 by [3]).

It seems to be a general rule for cortical cells that neighbouring cells within a cortical area are always involved in similar tasks. Thus a stimulus that activates one cell is very likely to elicit a reaction also in the surrounding cells. Or a cell in Motor cortex stimulating one muscle fiber is likely to be surrounded by cells activating neighbouring fibers. One expression of that property is the

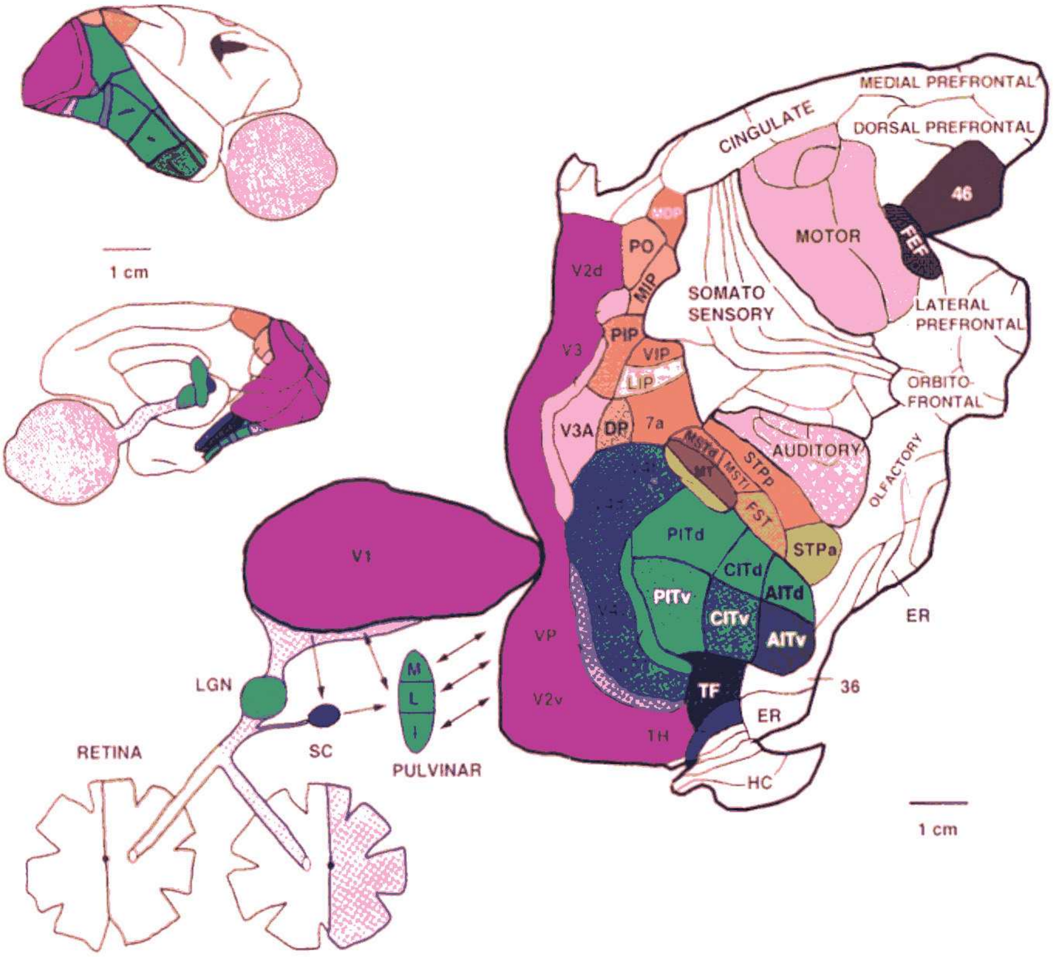


Figure 2.9: The most famous classification of cortical regions according to [56]

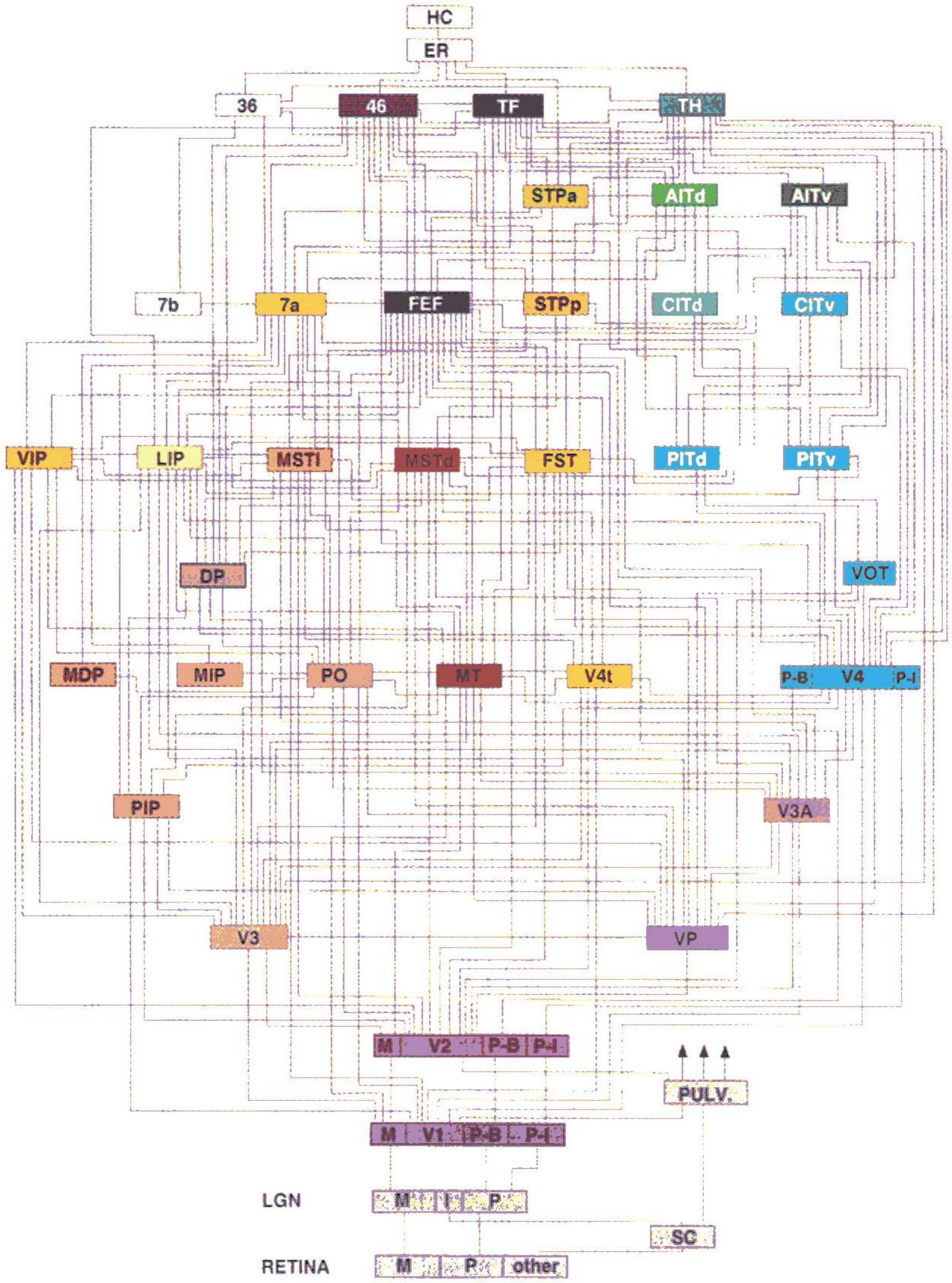
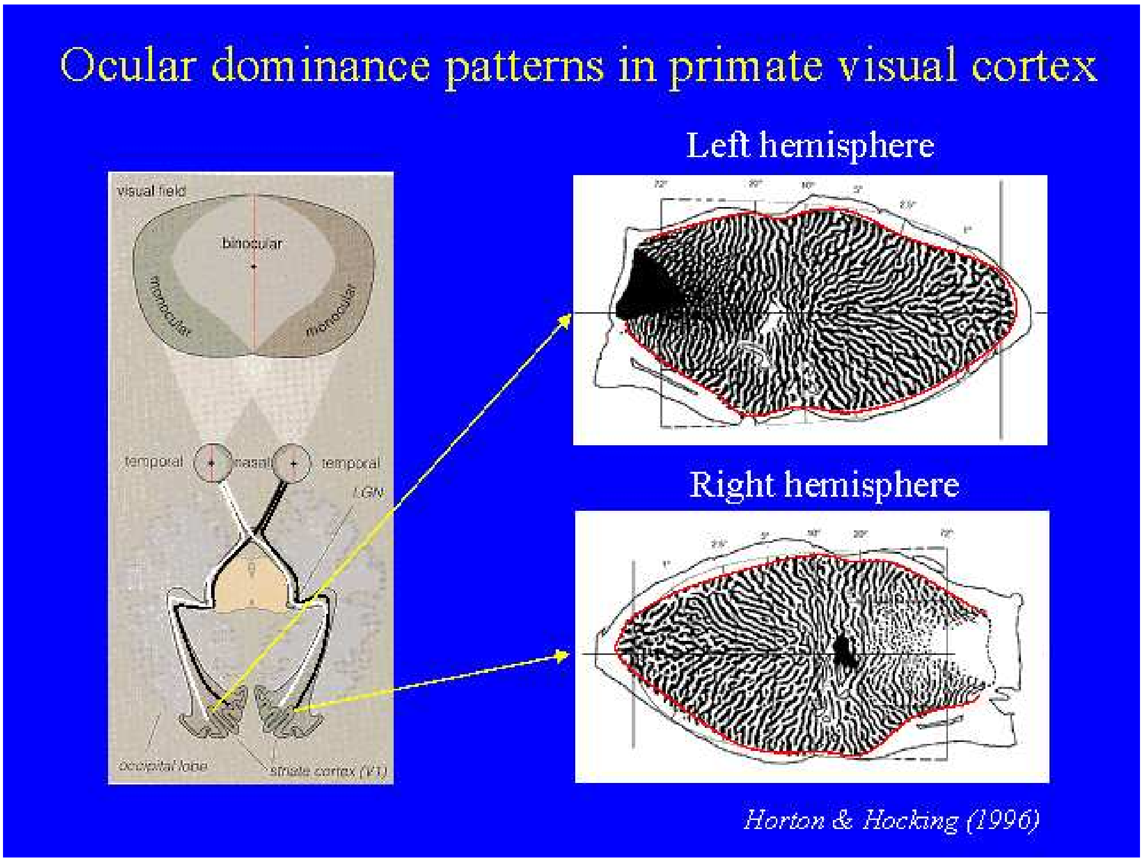


Figure 2.10: The most famous connection hierarchy of cortical regions according to [56]



Ocular dominance patterns in primate visual cortex

Left hemisphere

Right hemisphere

Horton & Hocking (1996)

Figure 2.11: Ocular dominance patterns according to [26]

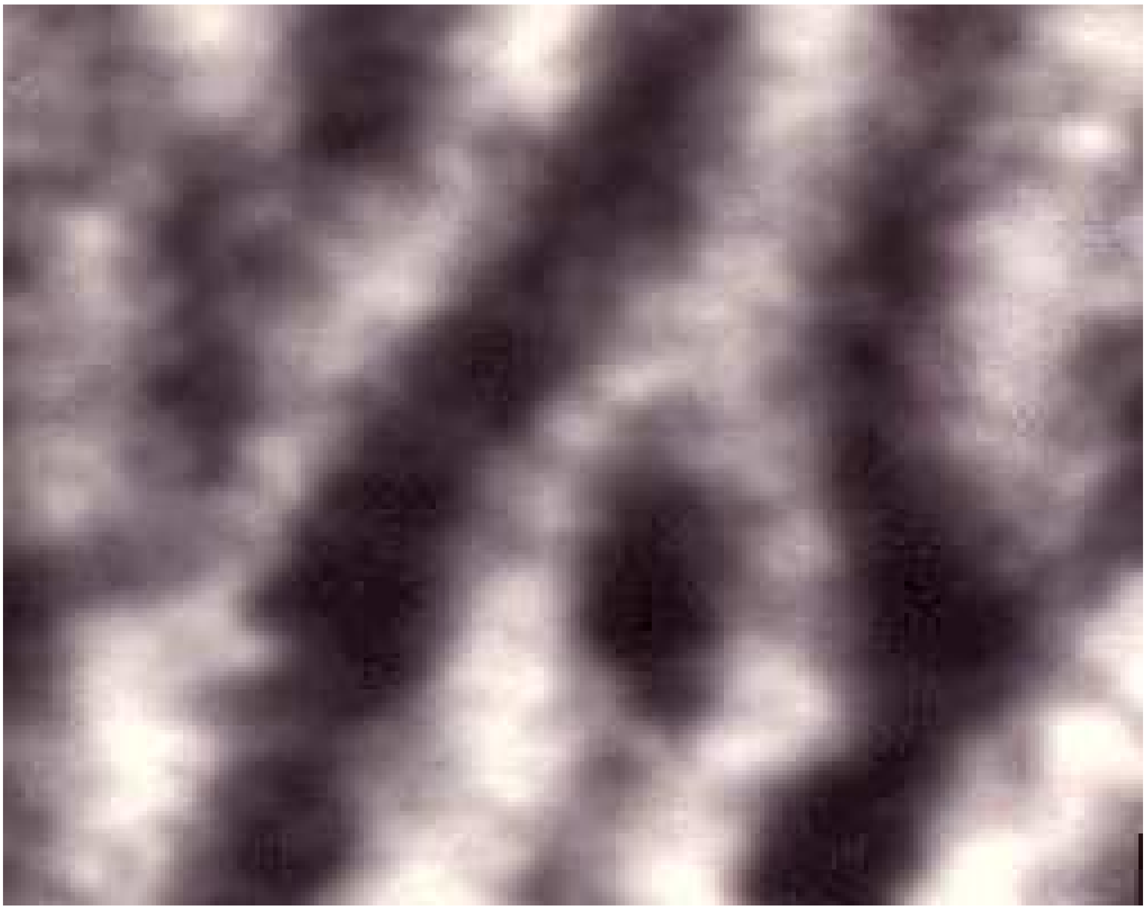


Figure 2.12: A close-up of ocular dominance patterns by [3]

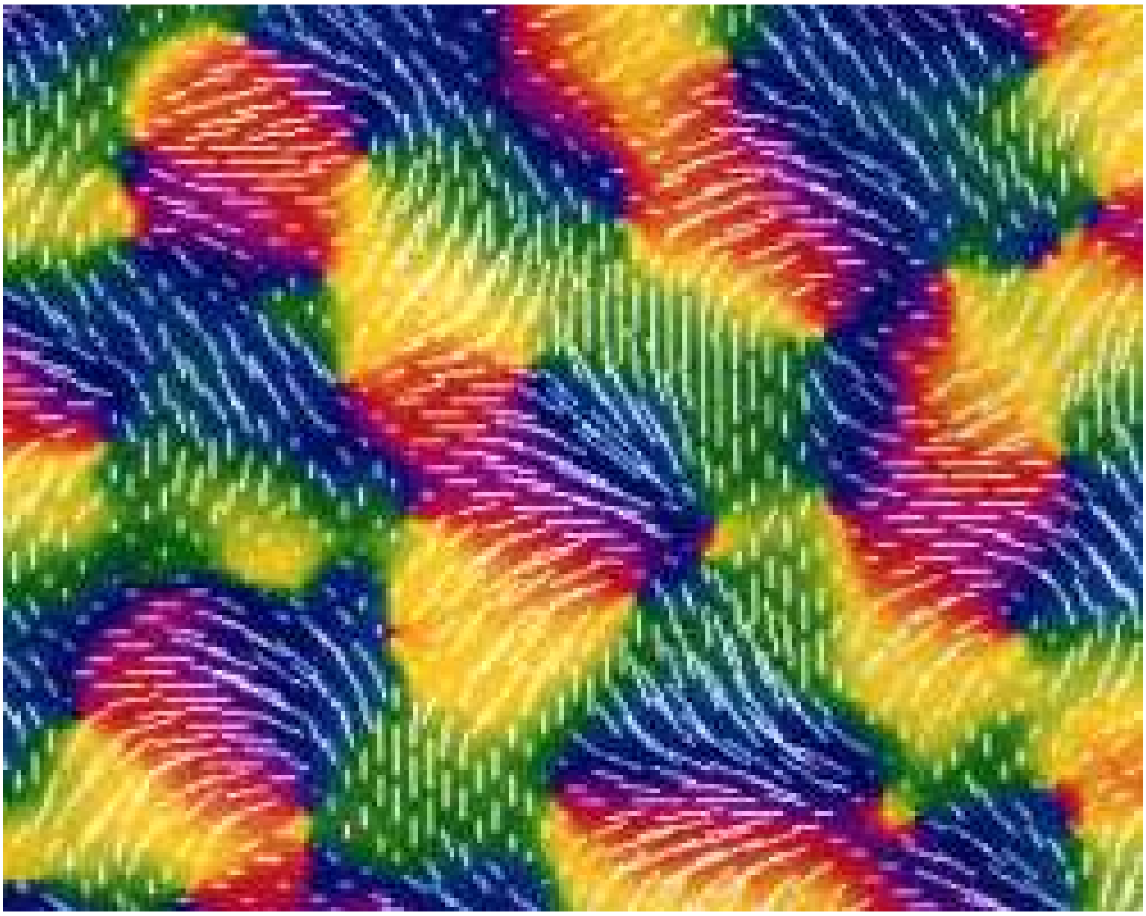


Figure 2.13: Orientation selectivity patterns [4]. The classic article of this author is [5].

previously described topological mapping between brain parts. But also other properties than its location in space of an optimal stimulus are shared among neighbouring cells. The optimal stimulus for cells shifts gradually as one moves on over the cortical surface. Such stimulus properties in V1, for instance, are position in the visual field, ocular dominance, and also orientation selectivity. That last property of V1 cells is that they are not optimally stimulated by a point in the visual field but rather by a bar of a particular orientation. Figure 2.13 shows regions on the V1 surface where the cells respond optimally to bars of a particular orientation. The orientation preference is indicated by colour and little bars.

2.2.4 Microcolumns and Cortical Layers

If the anatomy of cortex is examined it appears to be quite uniform across its surface. Researchers thus have still reason to hope that they can model cortex as an assembly of identical computational units and that only the connectivity pattern between those units defines the system behaviour. A name that has been given to those units is ‘cortical microcircuit’ or ‘microcolumn’.

As one examines the structure through the cortex, i.e. vertical to its surface, a layered organization is observed. Anatomists distinguish 6 principal layers and some sublayers. The layers can be identified by their mixture of cell types and by their connectivity pattern: Where their input comes from and where they project to. It is for example known that layer 4 and to a lesser degree layer 6 in V1 receive the bulk of the direct sensory input from the Thalamus.

The layers can be made visible with different kinds of stainings. Figure 2.15 shows a light microscope picture of a vertical cut of V1 and figure 2.16 is a drawn interpretation of such a microscope picture that emphasises some relevant structures.

2.2.5 Neurons and Synapses

A big variety of neuron types can be found in the brain. Many anatomical studies are occupied with them and their connection patterns. One major distinction criteria is that of excitatory and inhibitory neurons. But also their form and area of occurrence identify them. Some basic insight on the operation principal of neurons have been gained by experiments conducted on the so called ‘Giant Squid Axon’. In particular the spike generating mechanism has been studied on this special neuron. The role of sodium (Na+) and potassium (K+) ions have been studied by Hodgkin and Huxley [24]. More detail on their model will be discussed later in this script in a separate chapter (chapter 4). But in short: neurons receive charge packages as inputs triggered by voltage spikes (action potentials) that they receive from other neurons. Once the accumulated charge exceeds a threshold, they in turn produce an action potential that they send to other neurons through a narrow ‘cable’ called the axon.

The connection sites between neurons are called synapses (compare figure 2.17). In most cortical synapses the presynaptic terminal (part of the axon of the ‘sending’ cell) and the postsynaptic terminal (part of a dendrite or cell body of the ‘receiving’ cell) are actually physically separated by a narrow gap, the synaptic cleft. The transmission of the signal between the two terminals is conducted chemically. Vesicles with those chemicals (transmitters) are released from the presynaptic terminal when triggered by an action potential. They cross the synaptic cleft and attach to ion channels on the post synaptic terminal which are thus opened. They then let positively or negatively charged ions inside the cell, in effect adding or removing charge. Note that a sending neuron is exclusively excitatory or inhibitory, i.e. releasing the same type of transmitter on all its terminals. The synapses are also believed to be the primary means of storing learnt information in the brain. Their strength, i.e. the amount of charge transmitted to the post synaptic cell per input spike, can change and, thus, change the behaviour of the network.

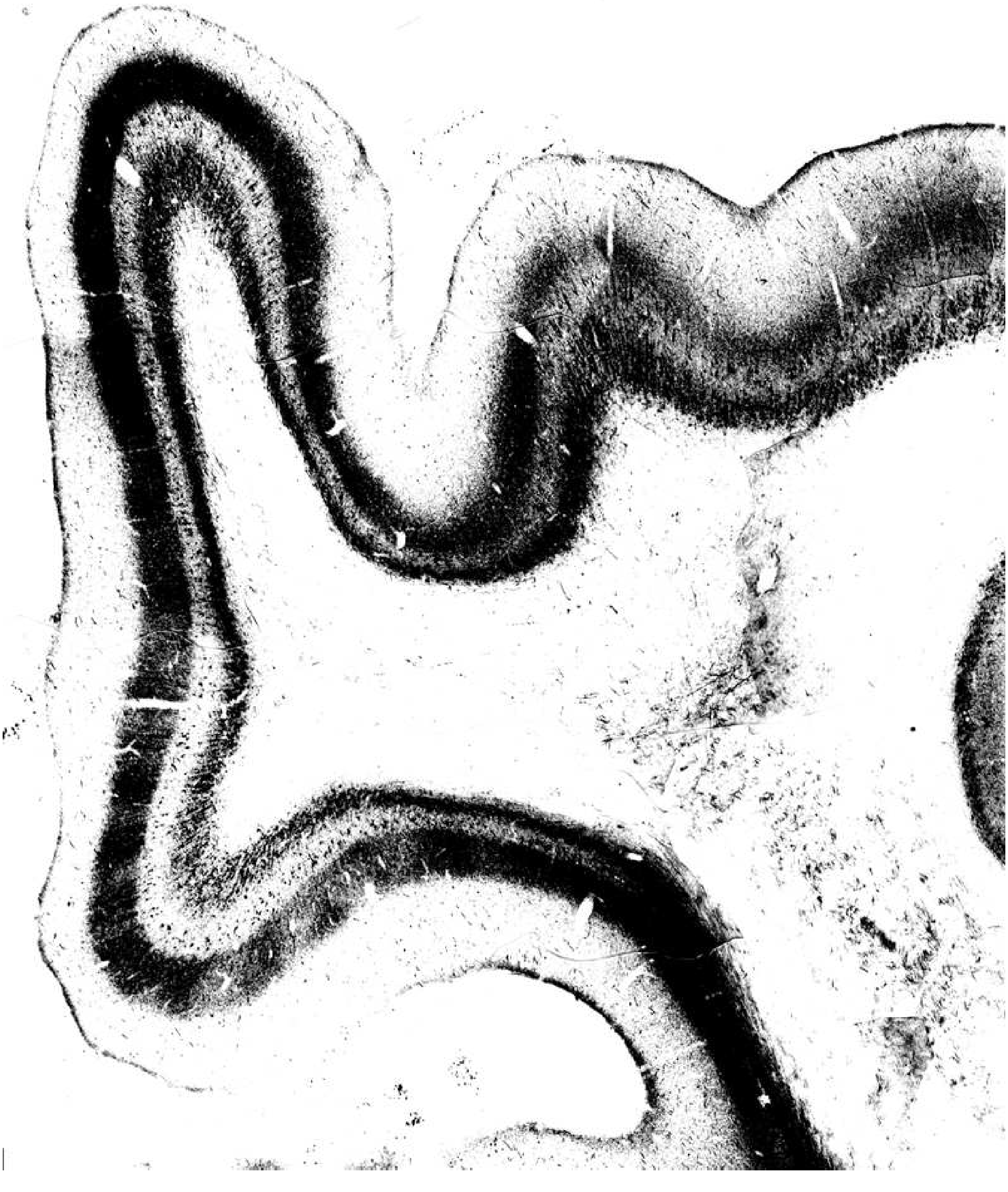


Figure 2.14: Staining of cortical layers in cat cortex [37]

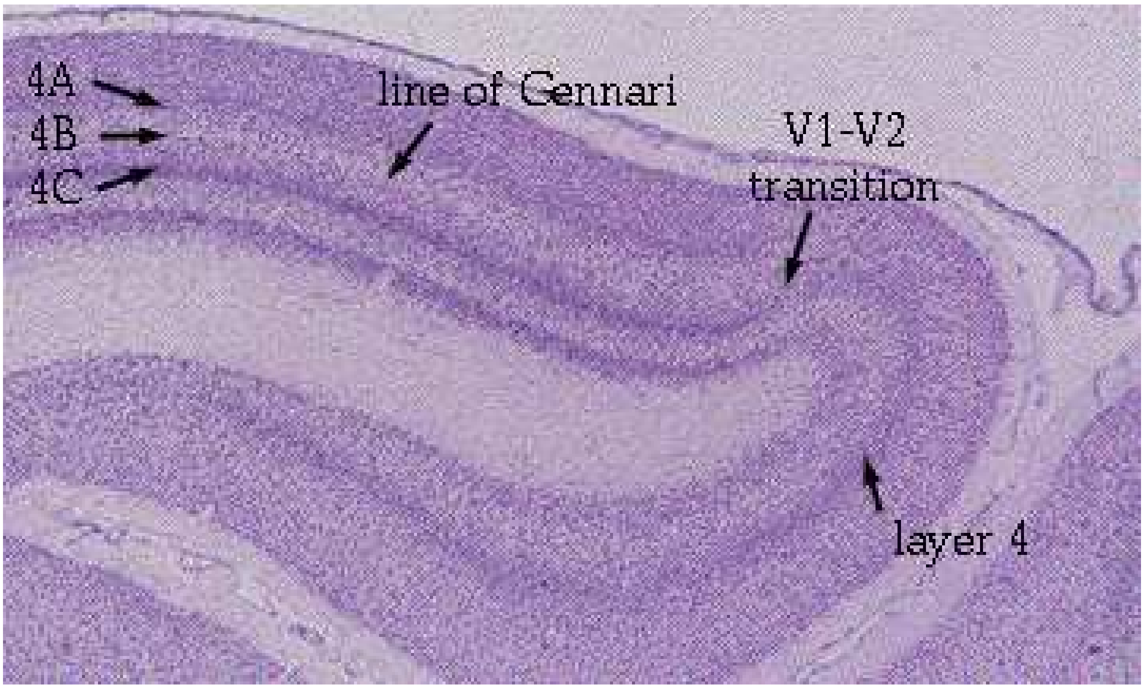


Figure 2.15: Staining of cortical layers in cat cortex [37]

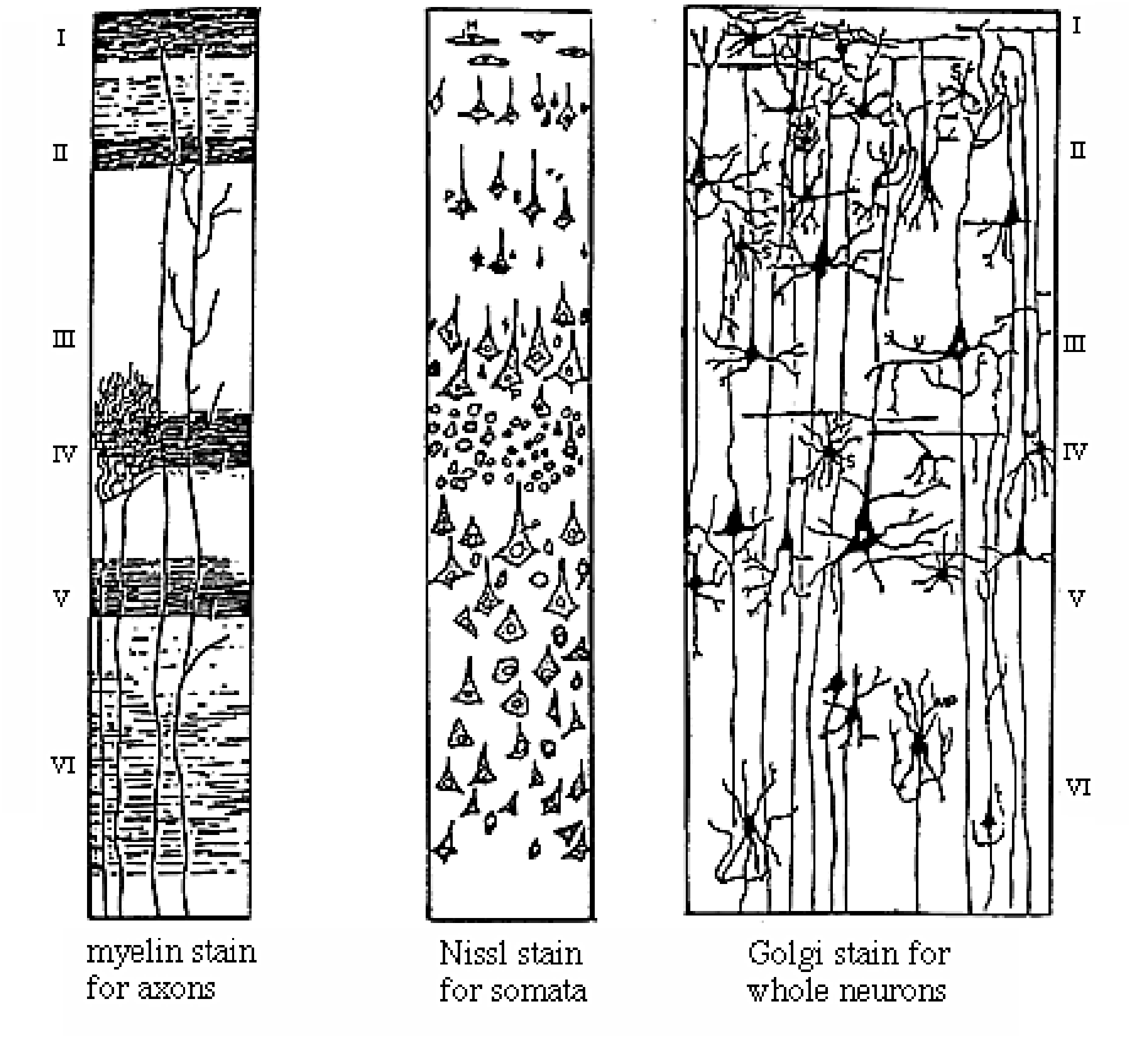


Figure 2.16: Different staining techniques revealing different features of cortical layers [41]

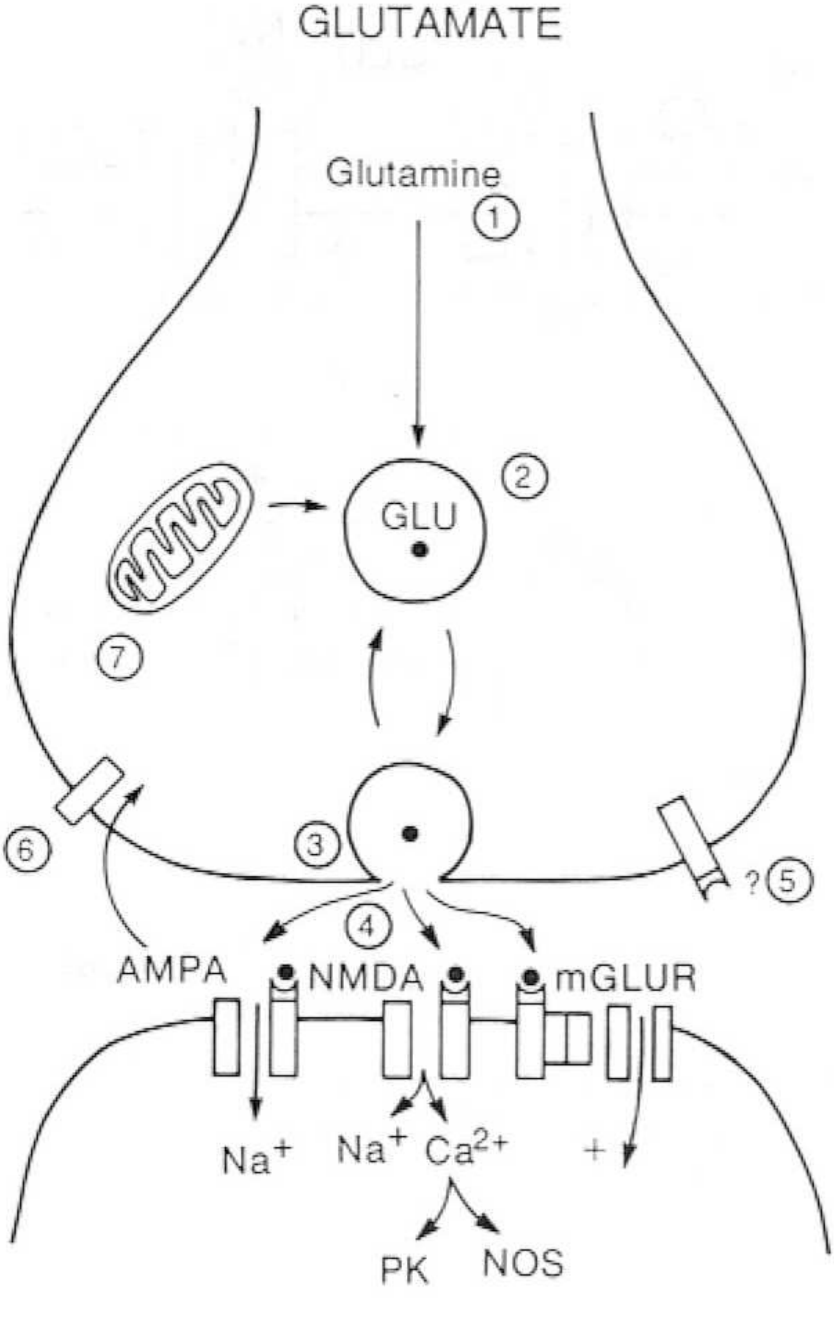


Figure 2.17: Exitatory glutamate synapse and some of its processes [51]

Chapter 3

Basic Analog CMOS

3.1 Field Effect Transistors

THE basic electronic element used on digital microelectronic chips is the transistor. It comes in different variants. We will only discuss FETs (field effect transistors) here (in particular CMOS-FETs (complementary metal oxide silicon field effect transistors)), which is the dominant transistor type nowadays. It is a four terminal device but often the 'Bulk' terminal is not explicitly represented and just connected to a supply voltage (Gnd for NFETs and Vdd for PFETs). Digital electronics uses them as a voltage controlled switch. The Gate voltage (terminal G) turns on or of the Drain (terminal D) current. However, in neuromorphic circuit the analog properties of this device are often used explicitly, according to the formulae given below. The subthreshold variant is most often used, because of its exponential behaviour that appears so often in nature too and because of the low current consumption in this mode of operation. The drawback is increased sensitivity to noise.

The CMOS-FET is a symmetric electronic device. The current from drain to source can be described as the difference between a forward current I_F and a reverse current I_R, both of which are dependent on the gate voltage and the voltage on either source or drain as given by the same equation (according to [57]):

$$I_{F(R)} = I_S ln^2 \left[1 + e^{\frac{V_G - V_{T0} - nV_{S(D)}}{2nU_T}} \right] \tag{3.1}$$

If the reverse current is so small as to be negligible, the transistor is said to be in *saturation.*

If $I_F << I_S$ ($V_G < V_{T0} + nV_S$) the transistor is said to be in *weak inversion* or in *subthreshold* and (3.1) can be simplified to:

$$I_F = I_S e^{\frac{V_G - V_{T0} - nV_S}{nU_T}} \tag{3.2}$$

If $I_F >> I_S$ ($V_G > V_{T0} + nV_S$) the transistor is said to be in *strong inversion* or *above threshold* and (3.1) can be simplified to:

$$I_{F(R)} = \frac{I_S}{4} \left(\frac{V_G - V_{T0} - nV_{S(D)}}{nU_T} \right)^2 \tag{3.3}$$

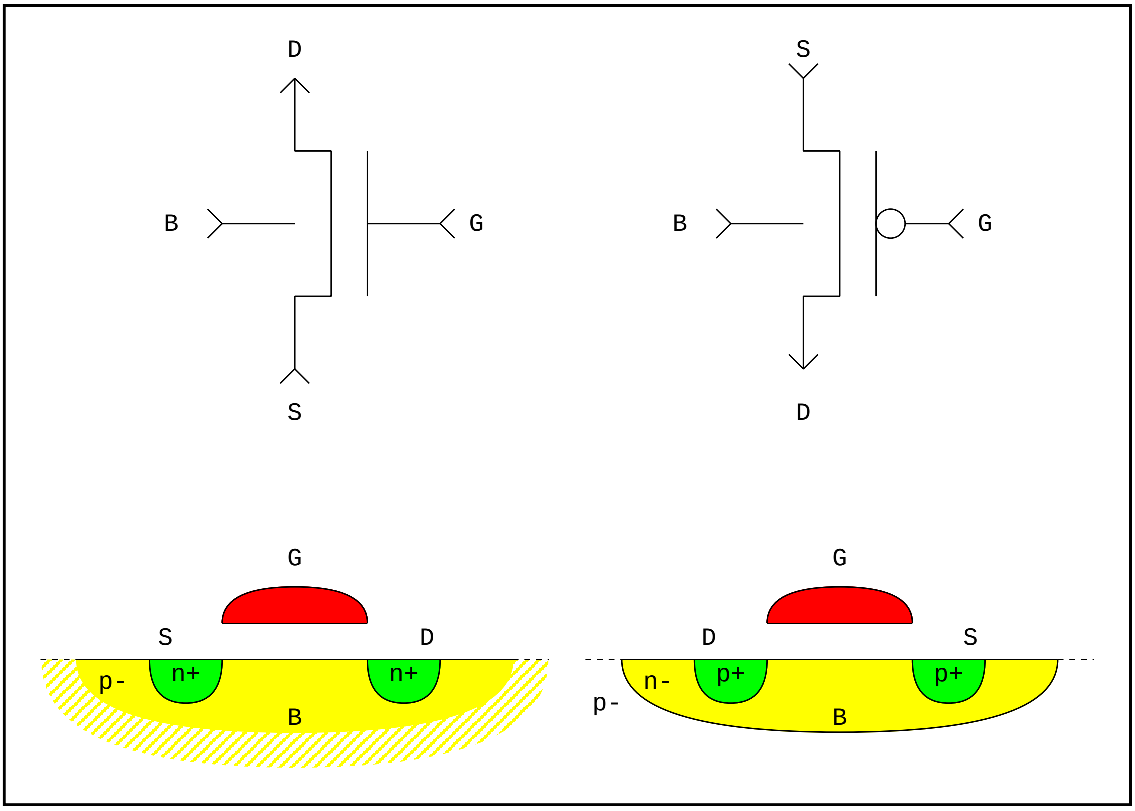


Figure 3.1: nFET and pFET in an nwell CMOS technology, symbol and cross section

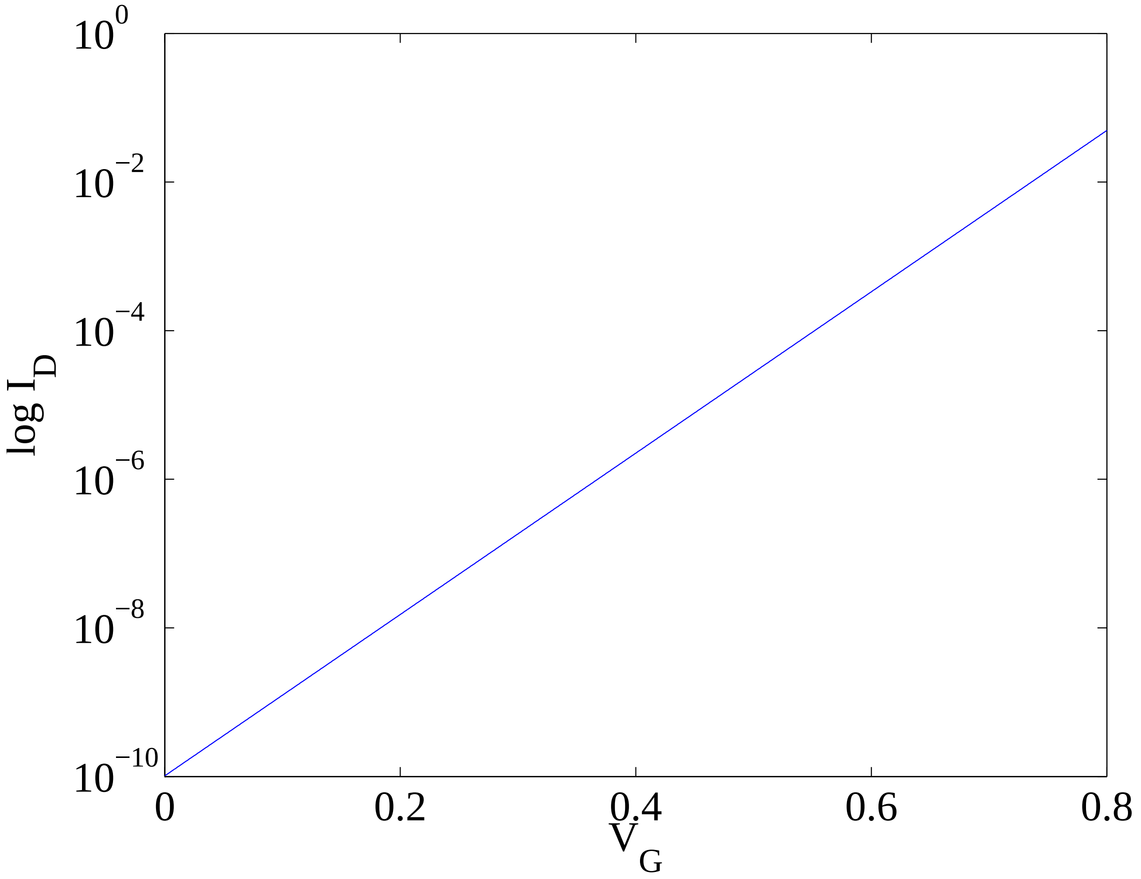


Figure 3.2: Transistor current IDS in dependency of gate voltage V_G in subthreshold on a logarithmic scale.

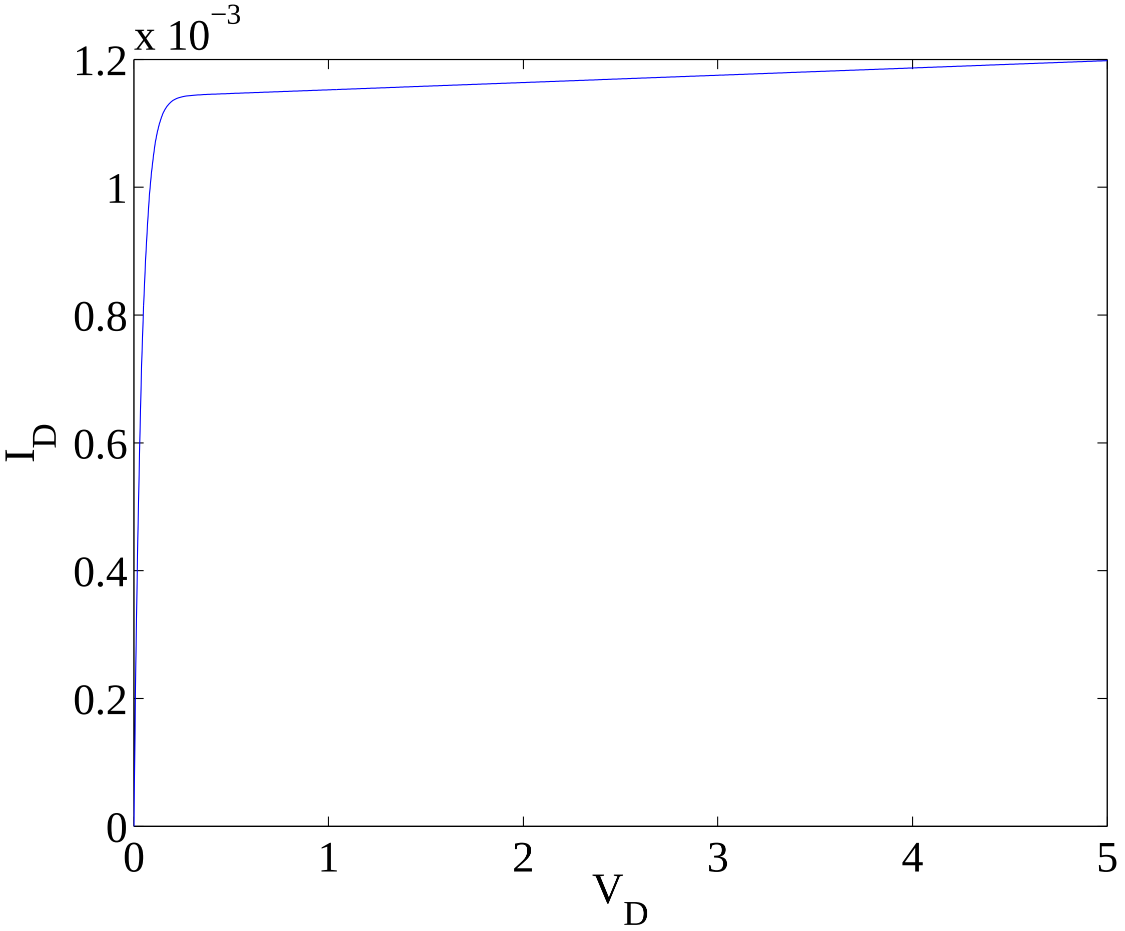


Figure 3.3: Transistor current IDS in dependency of drain to source voltage V_{DS} including Early effect.

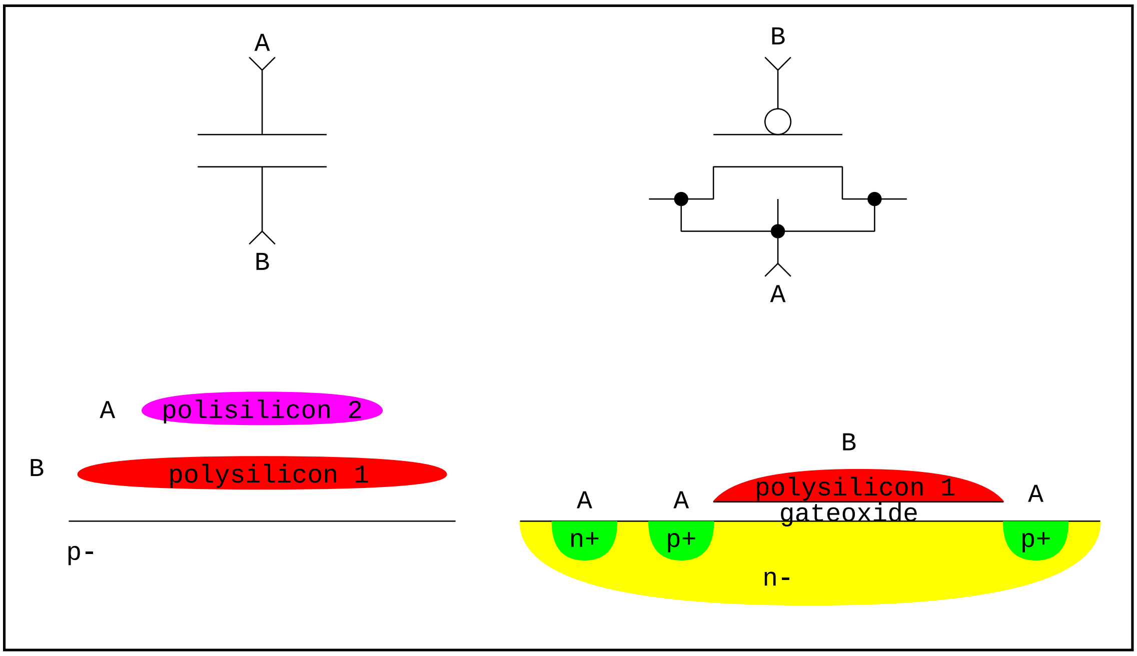


Figure 3.4: Capacitances in CMOS, symbol and cross section

A secondary effect that is neglected in these formulae, but never the less often important is the Early effect. It expresses a slight linear dependency of the saturated transistor current I_{DS} on the drain voltage V_D.

$$I_F = \frac{V_D + V_{Early}}{V_{Early}} I_F \tag{3.4}$$

3.2 Capacitors

An omnipresent device, intentionally or not. On microchips there are many unintentional 'parasitic' capacitances that influence the dynamic behaviour. Also intentionally placed capacitances influence the circuit dynamics. The behaviour can be described with a simple equation:

$$V = \frac{1}{C} Q \tag{3.5}$$

or as a differential equation if we differentiate (3.5) and substitute $\frac{\delta Q}{\delta t}$ with I.

$$\frac{\delta V}{\delta t} = \frac{1}{C} I \tag{3.6}$$

In neuromorphic and other analog designs they are often defining circuit time-constants or they are used to control floating nodes. In the later case one can actually move the voltage V_f on a floating node with a controlling capacitance C and a controlling voltage V_c applied to the other terminal of the capacitance according to:

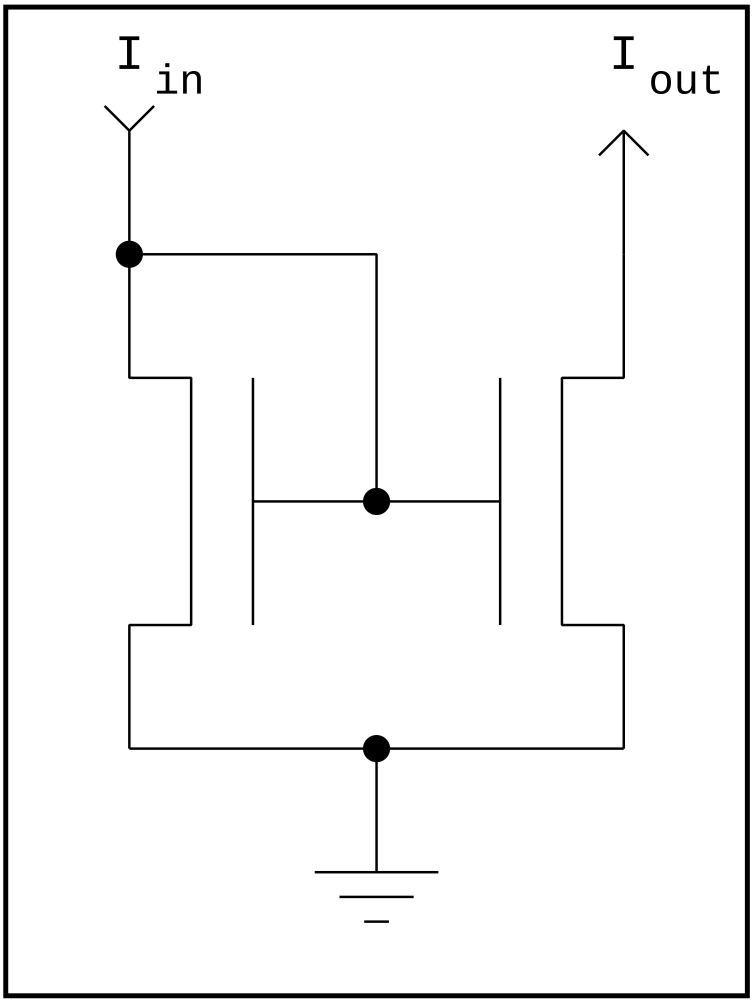


Figure 3.5: Schematics of a basic CMOS current mirror.

$$\frac{\delta V_f}{\delta t} = \frac{\delta V_c}{\delta t}\frac{C}{C_{tot}} \tag{3.7}$$

where C_{tot} is the total capacitance of the node, parasitic or intended, assuming constant voltages on all other capacitor terminals.

In CMOS intentional capacitors can be constructed between two polysilicon layers or as MOS-capacitors, using the gate oxide (figure 3.4). The later are unfortunately not linear, as the capacitance changes dependent if the 'channel' is 'depleted' or even 'inverted'.

3.3 Current Mirror

The basic current mirror (figure 3.5) creates a 'copy' of a current. It is especially useful if the way a current is used could potentially influence the current source. In that case it is better to work on a copy without interfering with the original current.

3.4 Differential Pair

A classic circuit (figure 3.6) that is the basis of many more complex circuits. It comes into play whenever voltages or currents are compared. Ina first approximation, the bias transistor can be seen as a current source supplying a bias current I_b. According to Kirchhoff's law, that bias current needs to be supplied by the two branches of the differential pair. And according to the transistor characteristics in subthreshold and in saturation those two currents are given by the gate voltages V_1 and V_2 and the source voltage V_C that is common to those two transistors. These facts lead to the formula:

$$I_b = I_1 + I_2 = I_S e^{\frac{-V_{T0}-V_C}{nU_T}} \left(e^{\frac{V_1}{nU_T}} + e^{\frac{V_2}{nU_T}} \right) \tag{3.8}$$

One thing one can see from that formula is that the ratio of the two currents will be exponentially dependent on the difference of the input voltages:

$$\frac{I_1}{I_2} = \frac{I_1}{I_b - I_1} = e^{\frac{V_1 - V_2}{nU_T}} \tag{3.9}$$

So for just a small difference in voltage, the bigger of the two currents will rapidly approach I_b. In other words: the circuit can almost be seen as a element with a binary output, where all the bias current is supplied only by the branch with the bigger gate voltage as input.

3.5 Transconductance Amplifier

The transconductance amplifier in its basic form is the father of all CMOS amplifiers (figure 3.7). Based on a differential pair it amplifies the difference of two input voltages in a range of up to a few 100mV. Beyond that range it works as a comparator, identifying the bigger input, since the output saturates. Its characteristics in the subthreshold can be deduced to follow this equation:

$$I_{out} = I_b \frac{e^{\frac{V_+}{nU_T}} - e^{\frac{V_-}{nU_T}}}{e^{\frac{V_+}{nU_T}} + e^{\frac{V_-}{nU_T}}} = I_b \tanh \frac{V_+ - V_-}{2nU_T} \tag{3.10}$$

3.6 Follower

A very nice circuit that allows to observe an analog voltage with minimal interference. It is just an amplifier with its output fed back onto the minus input (figure 3.8).

3.7 Resistor

A very useful element that is unfortunately hard to come by in any usefully big strength on a CMOS chip. It is often just a cable of polysilicon with a resistance in the order of a few Ohms per square (figure 3.9). In many models intended for CMOS implementation resistors are, thus, replaced by a followers which has similar but not quite the same properties as long as it remains in its linear range of operation. Some CMOS processes, however, offer a special polysilicon treatment to make it highly resistive.

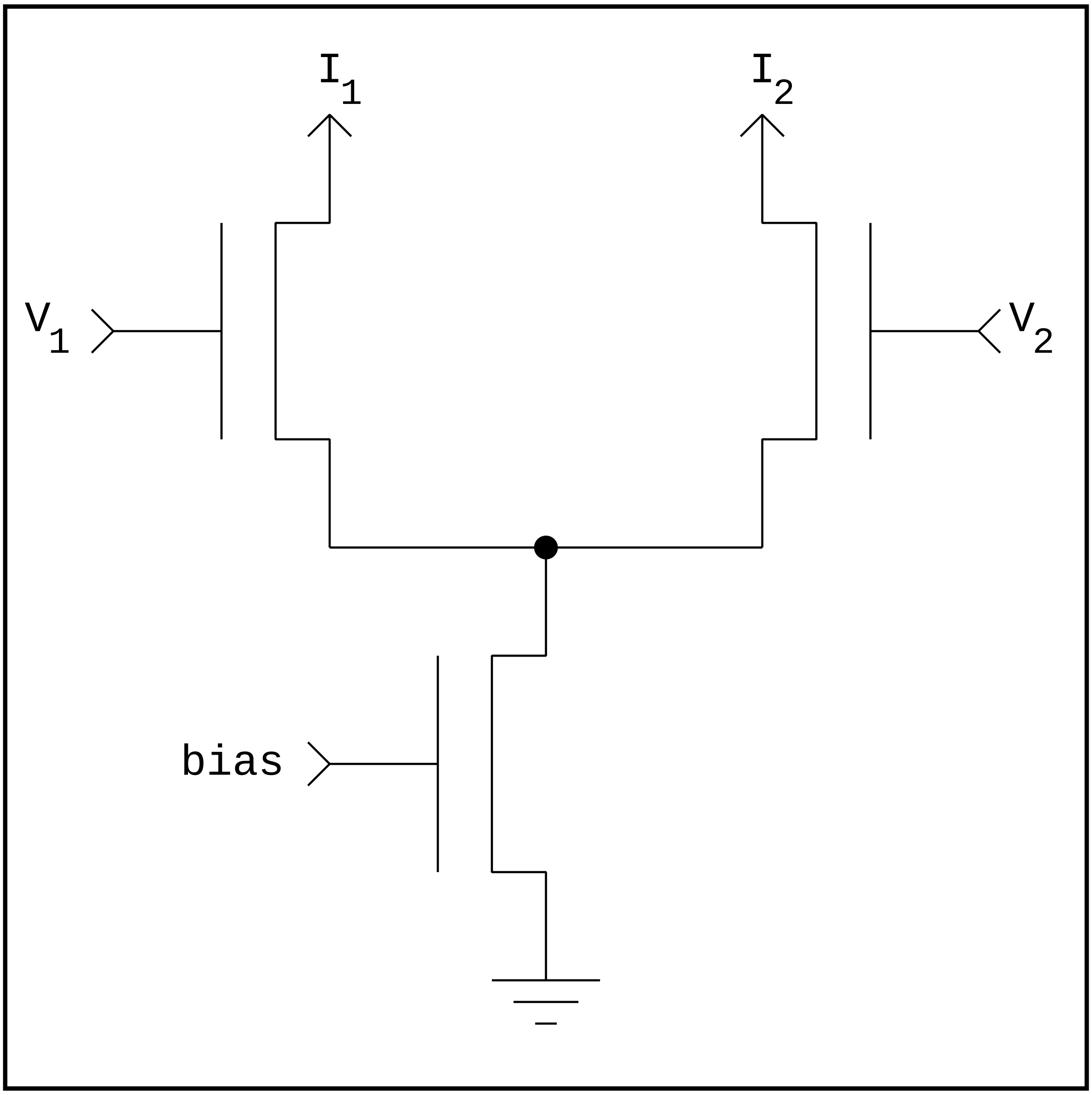


Figure 3.6: A differential pair

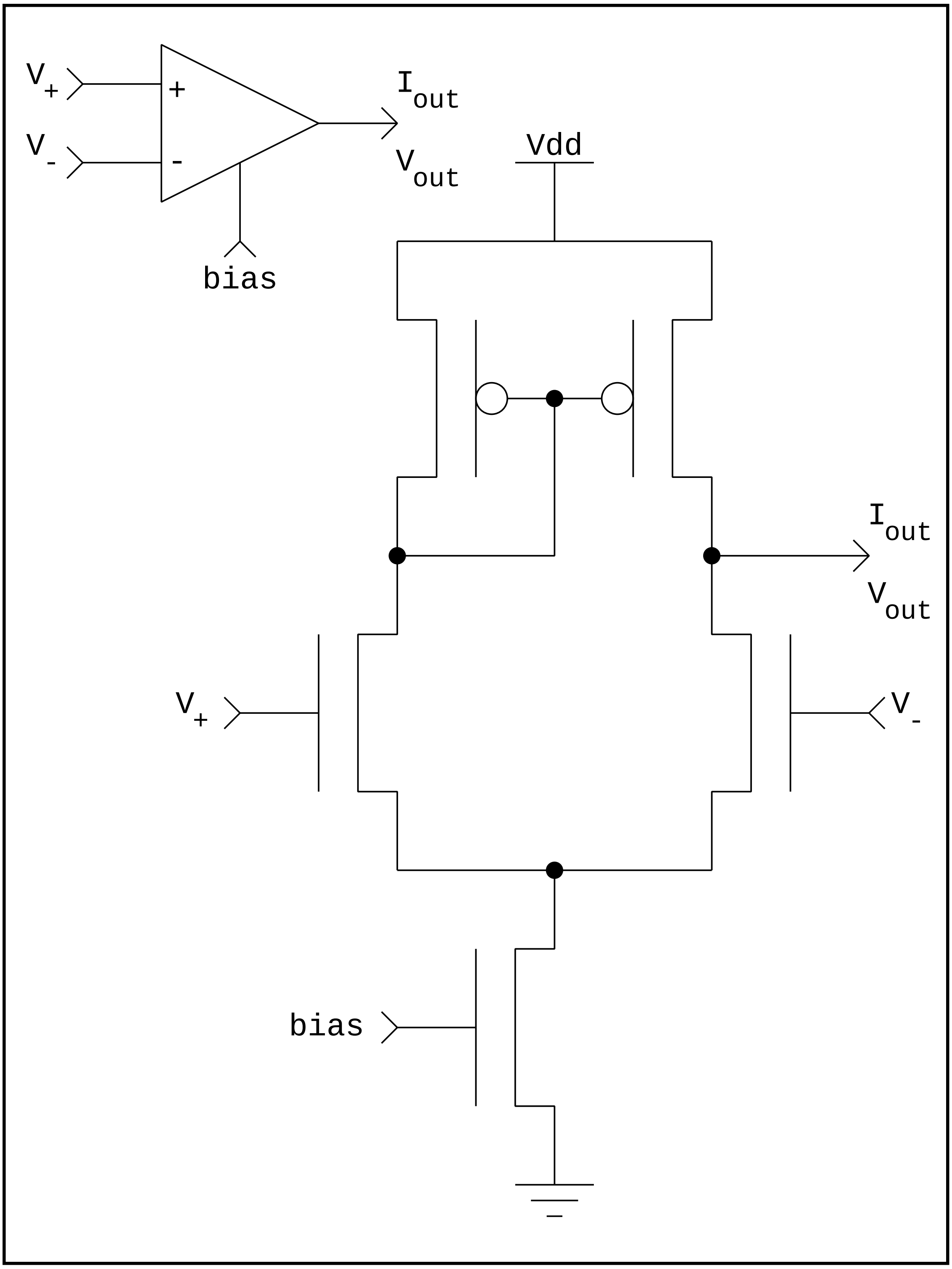


Figure 3.7: The basic transconductance amplifier

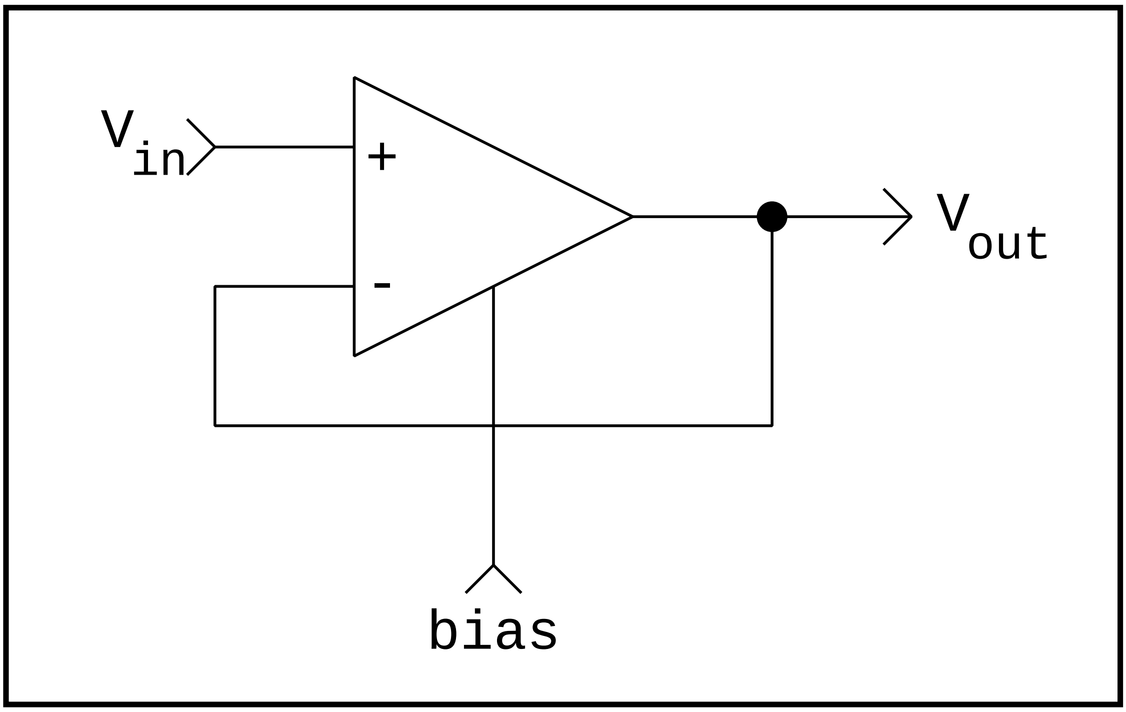


Figure 3.8: A follower, or analog voltage buffer

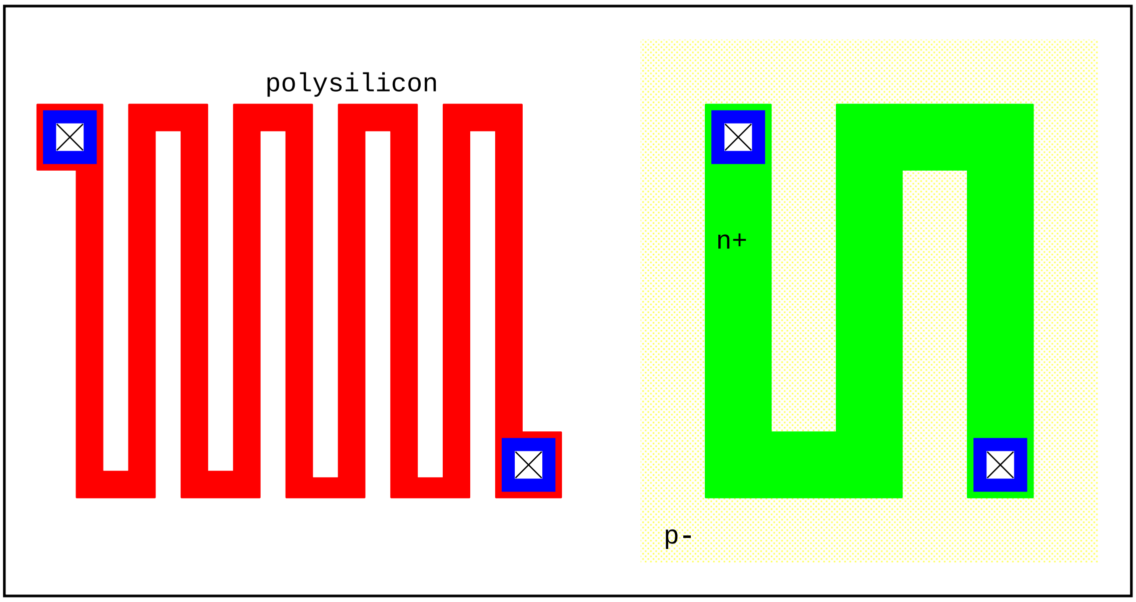


Figure 3.9: Possible resistor implementations in CMOS

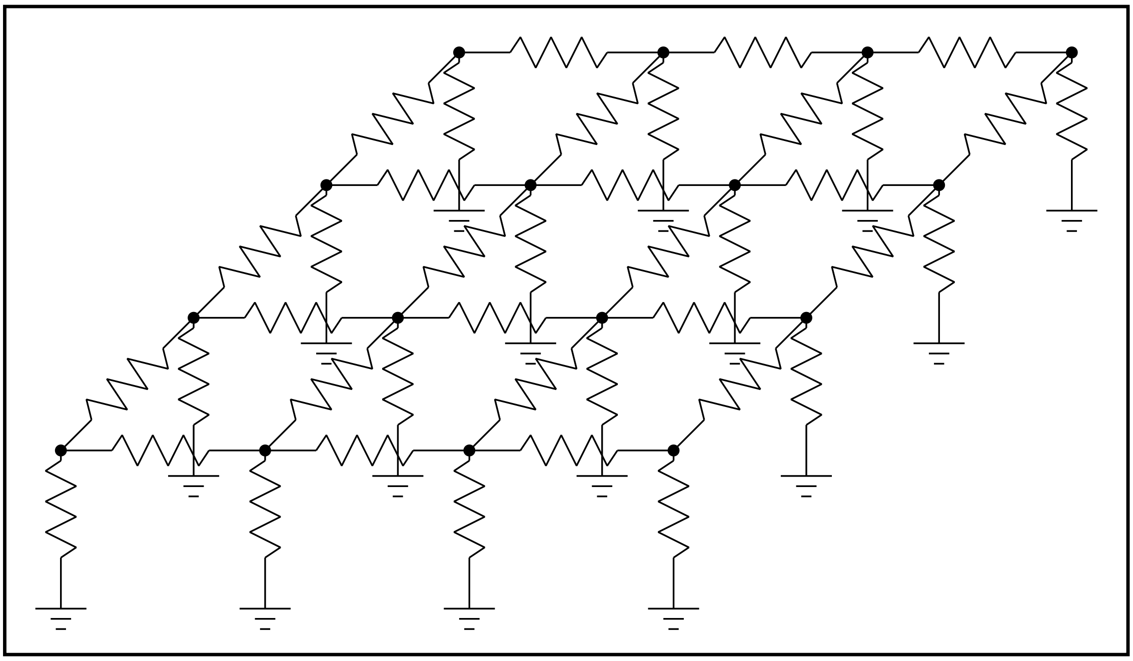


Figure 3.10: A resistive net, often used for spatial smoothing of an array of analog inputs.

3.8 Resistive Nets

A circuit often used to compute local averages (figure 3.10): When injecting a current into a node of the net it will rise the voltage of that node and to an exponentially declining degree the voltage of the neighbouring nodes. Thus it is for example used in the ’silicon retina’.

$$\frac{V}{R_V} = \frac{\delta^2}{\delta x^2 \delta y^2} \frac{V}{R_H} \tag{3.11}$$

As mentioned before, it is unfortunately not easily possible to obtain big resistors in CMOS. Thus, diffuser nets instead of resistive nets are often used. They replace the resistors with transistors with fixed gate voltage. This network behaves the same as a resistive network if one only observes the currents. This is nicely deduced in [57] for subthreshold by defining a pseudo voltage V^* that is exponentially dependent on the voltage, and a pseudo resistance R_H^* and R_V^* respectively.

$$\begin{aligned} \frac{V^*}{R_V^*} &= \frac{\delta^2}{\delta x^2 \delta y^2} \frac{V^*}{R_H^*} \\ V^* &= -e^{\frac{-V}{U_T}} \\ \frac{1}{R^*} = g^* &= I_S e^{\frac{V_G - V_{T0}}{nU_T}} \end{aligned} \tag{3.12}$$

3.9 The Winner Take All Circuit

The winner take all (WTA) circuit (figure 3.13) is certainly not a standard circuits in standard analog CMOS electronics books. But it has become one for the neuromorphic engineers. A

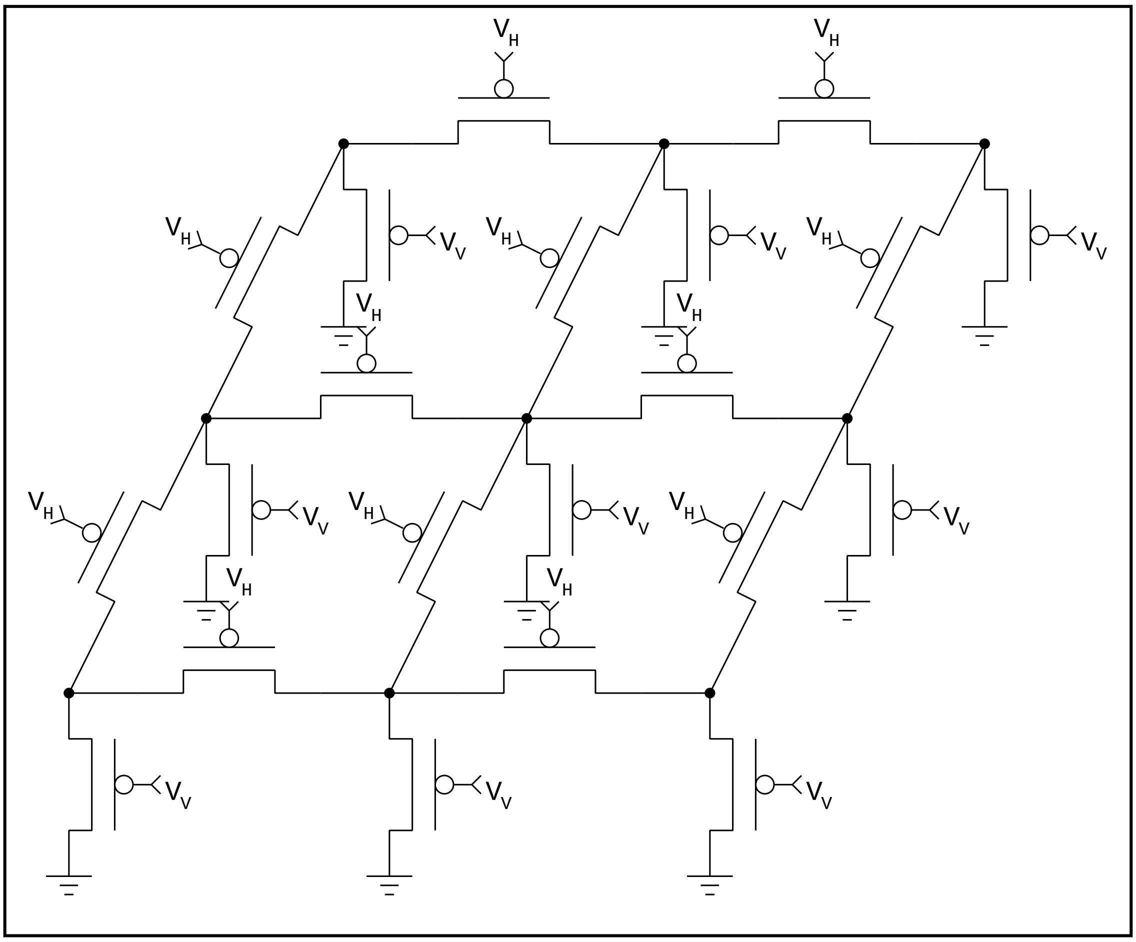


Figure 3.11: A diffuser net, linear in current mode.

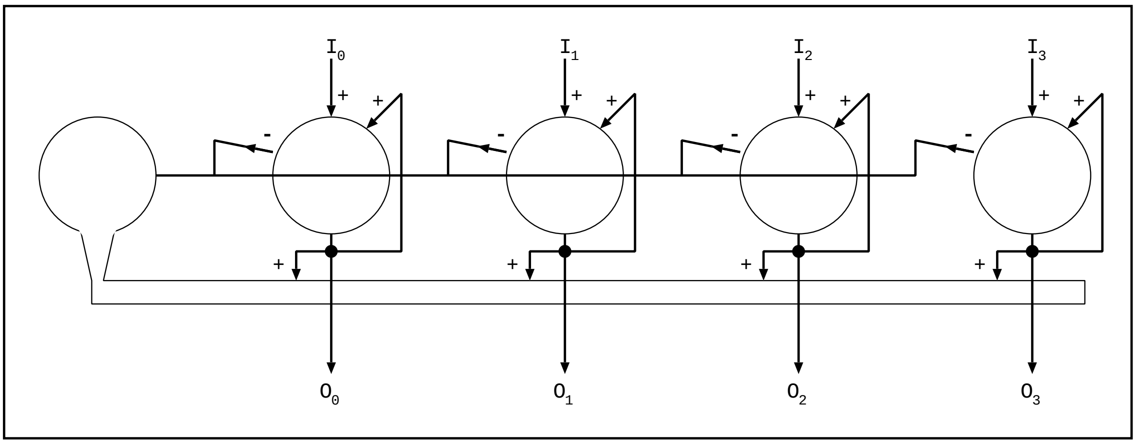


Figure 3.12: Neuronal winner take all principle.

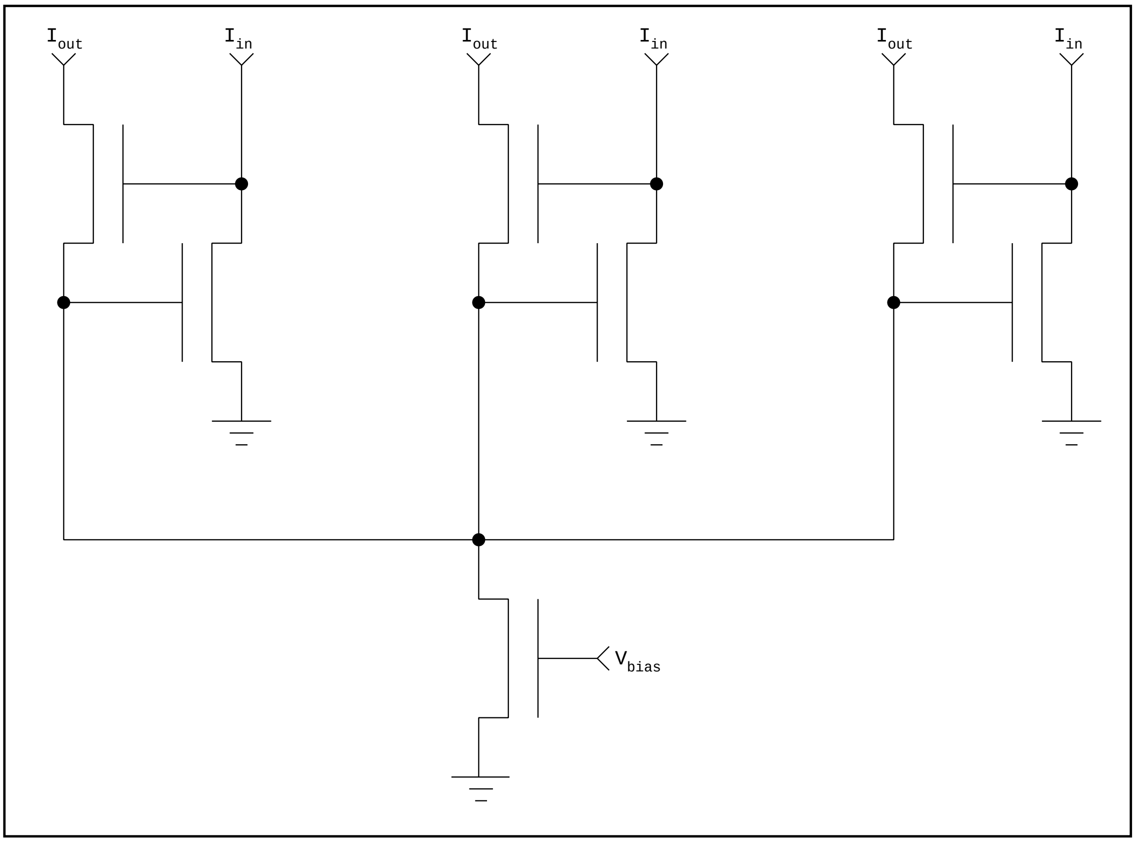


Figure 3.13: Basic analog CMOS WTA

WTA circuit/network filters out the strongest of a number of inputs, namely the 'winner'. In mathematical terms it is a function $\overrightarrow{\mathrm{wta}}(\vec{x}) : R^n \rightarrow R^n$ with

$$\overrightarrow{\mathrm{wta}}(\vec{x}) = \sum_{i \in S} \vec{e}_i, \; S = \left\{ s : s \in [1, n], x_s = \max_{j \in [1,n]} x_j \right\} \tag{3.13}$$

But actually also variants of that function count as WTA networks. In particular if the output of the winner is not just 1 but actually the original input to that node.

A neural network implementation of that function is sketched in figure 3.12. Global inhibition exerts inhibition on all neurons that is equal to the sum of the activity of all neurons. All neurons receive one excitatory external input and feedback from their own output. When running this setup recursively, the only stable state of that network is that in which only the neuron with the strongest input remains active.

The WTA circuit in figure 3.13 is actually based on a differential pair, only that we don't just talk of a pair here but possibly of several more than two branches. In figure 3.13 there are three branches shown. The basic differential pair really is a WTA circuit already: with quite a sharp transition, all the bias current flows through that branch with the biggest gate voltage. In the WTA circuit the differential pair structure can be seen when considering only the output current branches. The gate voltage of the transistor in this differential pair structure is set by a feedback loop however. The inputs are currents. The feedback loop drives the voltages even further apart, for a small difference in input currents.

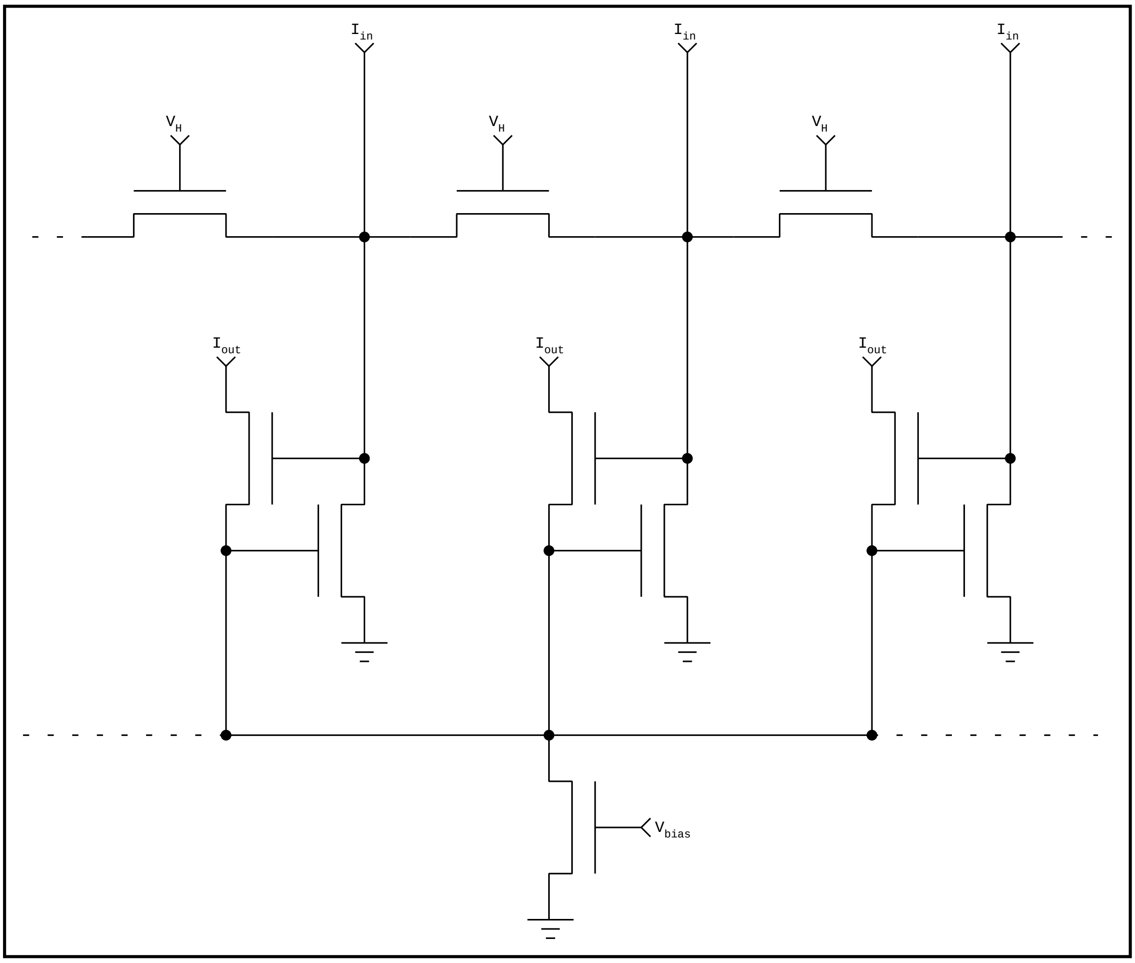


Figure 3.14: WTA with spatial smoothing/ cross excitation.

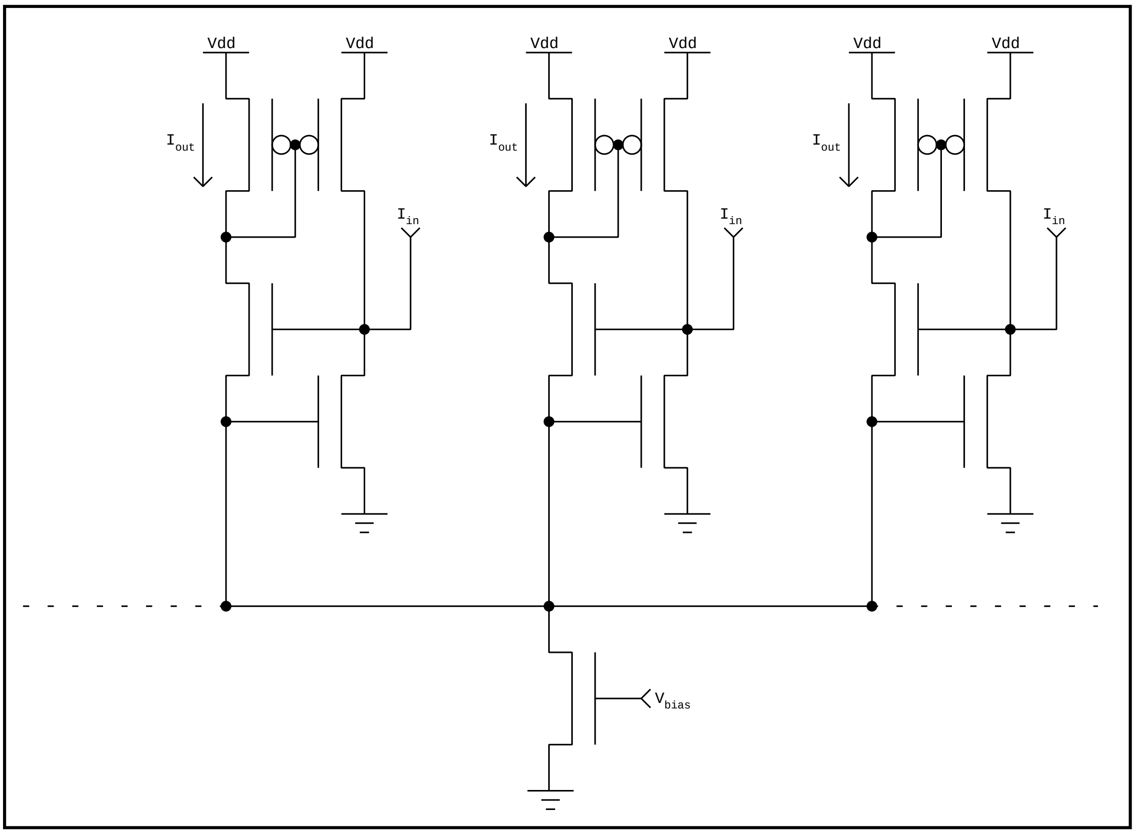


Figure 3.15: WTA with hysteresis

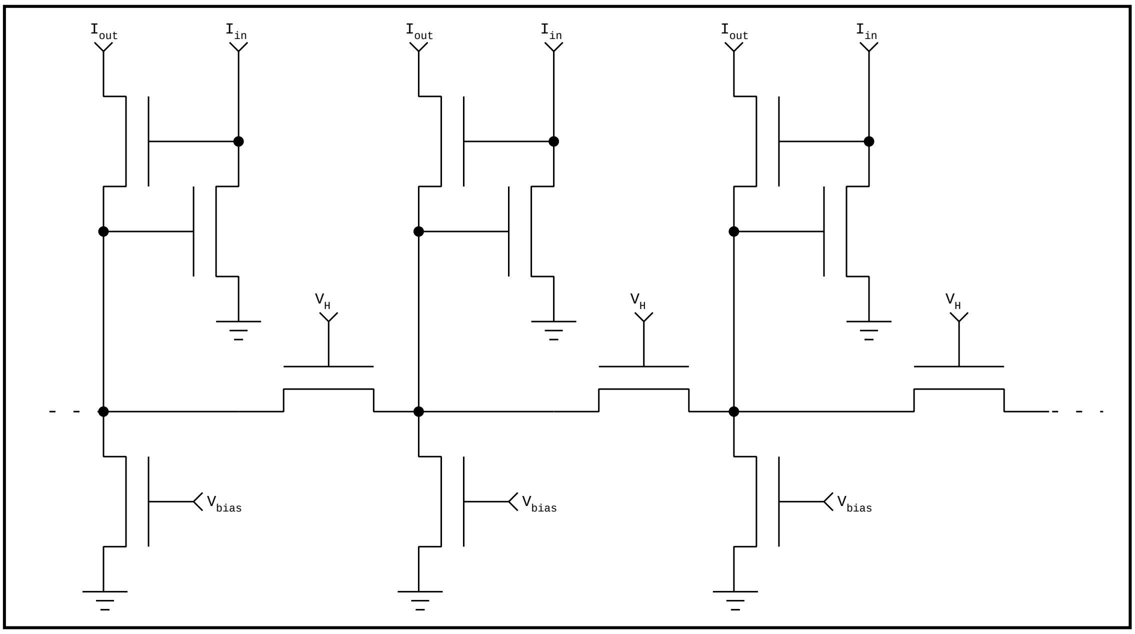


Figure 3.16: WTA with local winners/local cross inhibition

Chapter 4

Real and Silicon Neurons

4.1 Real Neurons

The neuron can be seen as the basic information processing unit of the brain of which it has around 10^{11} each with about 10^4 connections to others of its kind. 'Neuron' is the name for a biological brain cell that is an infinitly complicated organism and the arena for many chemical and electric processes. It is the way of science to try to understand such complicated systems by formulating simplified models thereof. A crude anatomical model of a neuron is shown in the figure 4.1. Of course, there are much more detailed models. Be aware that a lot of research has gone into describing different classes of neurons in great detail. Synapses are the connection sites between neurons. They usually connect axons and dendrites. The axon carries the neuron's output voltage pulse, the so called action potential. Dendrites are the input sites receiving current input that flows to the soma and is integrated there. The axon hillock is the site where an action potential is initiated as the integrated input current amounts to a voltage higher than some threshold.

Figure 4.2 gives an impression of how a real neuron looks like. It is a light microscope photograph kindly provided by John Anderson, Institute for Neuroinformatics, Zürich Switzerland. He also provided the reconstruction of the dendritic tree of a neuron (a cortical pyramidal cell) shown in figure 4.3, the result of many days of work. The bar at the bottom represents 100 micro meters.

4.2 aVLSI Models of Neurons

Neuromorphic engineers are more interested in the physiolocical rather than the anatomical model of a neuron though, which is concerned with the funcionality rather than only classifying its parts. And their preference lies with models that can be realized in aVLSI circuits. Luckily many of the models of neurons have always been formulated as electronic circuits since many of the varyiing observables in biological neurons are voltages and currents. So it was relatively straight forward to implement them in VLSI electronic circuits.

There exist now many aVLSI models of neurons which can be classified by their level of detail that is represented in them. A summary can be found in table 4.1. The most detailed ones are known as 'silicon neurons'. A bit cruder on the level of detail are 'integrate and fire neurons' and even more simplifying are 'Perceptrons' also known as 'Mc Culloch Pitts neurons'. The simplest way however of representing a neuron in electronics is to represent neurons as electrical nodes.

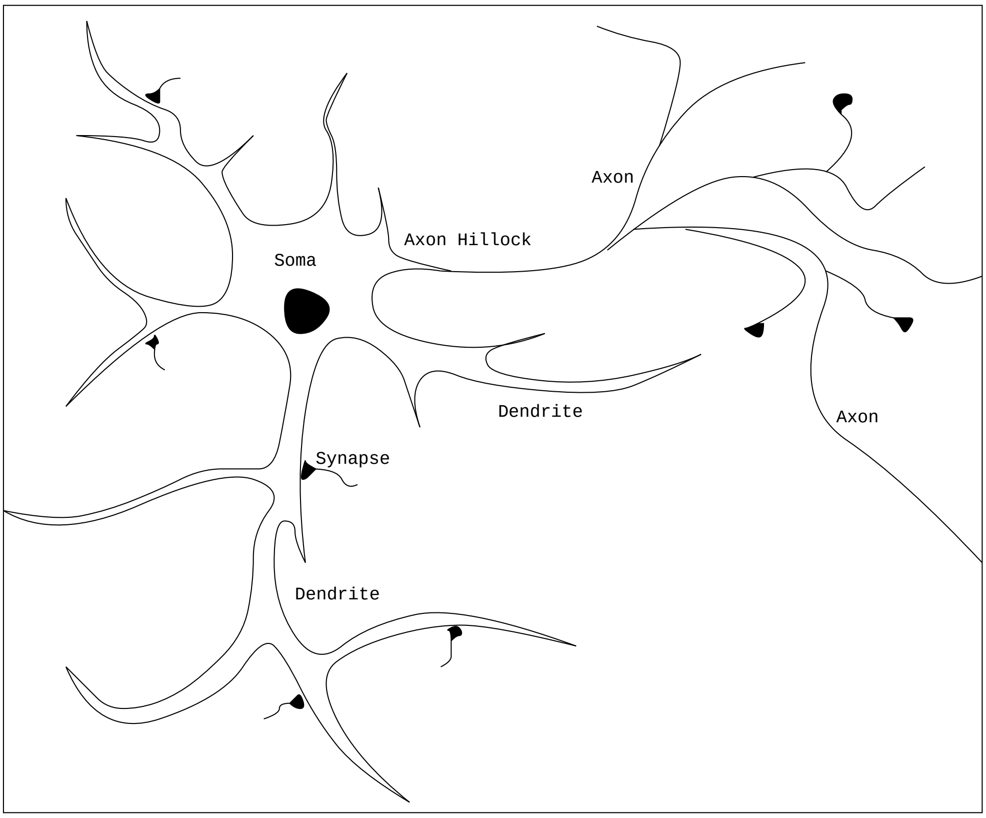


Figure 4.1: Basic anatomical parts of a Neuron

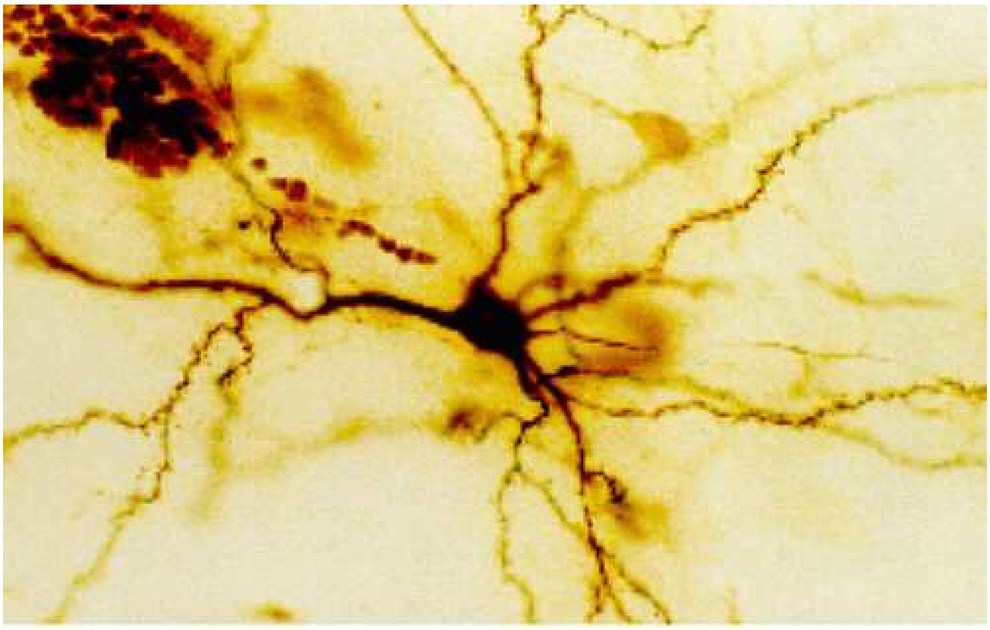


Figure 4.2: Light microscope photograph of a stained neuron

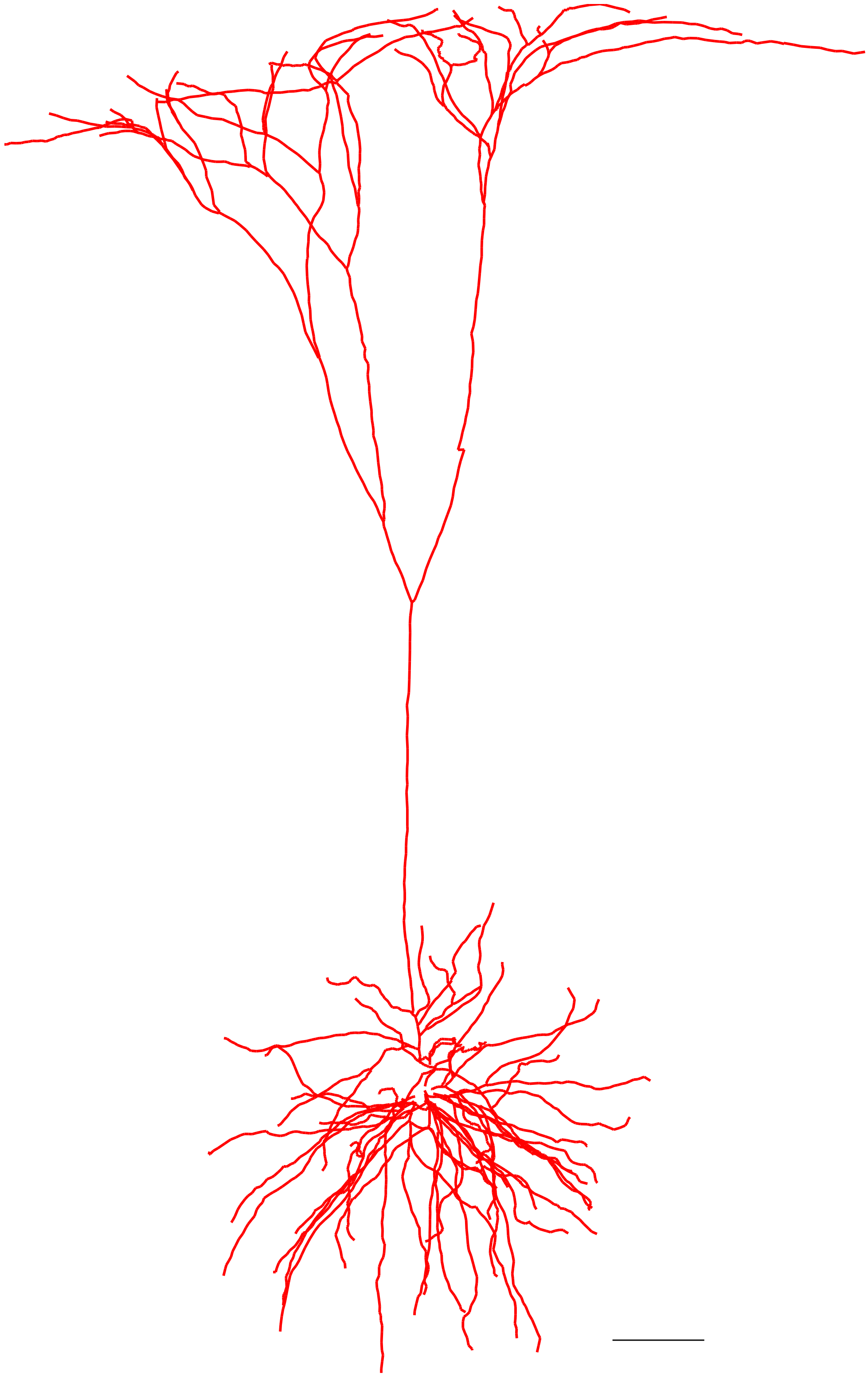


Figure 4.3: The 3D reconstruction of a cortical layer 5 pyramidal cell.

electrical nodes	most simple, big networks implementable
perceptrons	mathematically simple, but complicated in aVLSI
integrate and fire neurons	mathematically complex, but simple in a VLSI
compartemental models	complex, simulation of big networks are very slow

Table 4.1: aVLSI models of neurons

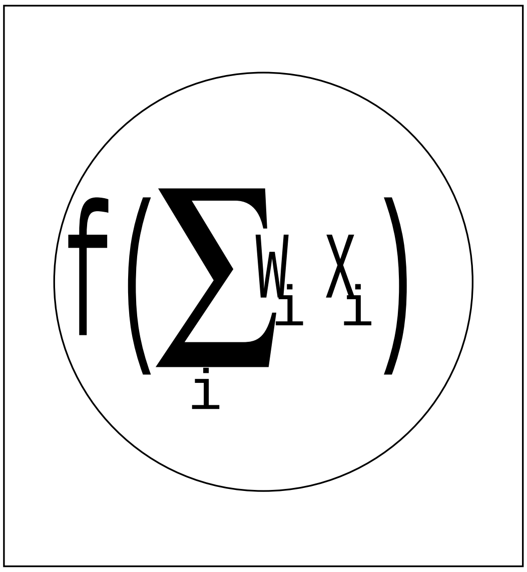


Figure 4.4: The mathematical model of a Perceptron

4.2.1 Simple Electrical Nodes as Neurons

The most simple of all neuronal models is to just represent a neuron's activity by a voltage or a current in an electrical circuit, and input and output are identical, with no transfer function inbetween. If a voltage node represents a neuron, excitatory bidirectional connections can be realized simply by resistive elements between the neurons. If you want to add the possibility for inhibitory and monodirectional connections, followers can be used instead of resistors. Or if a current represents neuronal activity then a simple current mirror can implement a synapse. Many useful processing networks can be implemented in this manner or in similar ways. For example a resistive network can compute local averages of current inputs.

4.2.2 Perceptrons (Mc Culloch Pitts neurons)

A perceptron is a simple mathematical model of a neuron. As real neurons it is an entity that is connected to others of it's kind by one output and several inputs. Simple signals pass through these connections. In the case of the perceptron these signals are not action potentials but real numbers. To draw the analogy to real neurons these numbers may represent average frequencies of action potentials. The output of a perceptron is a monotonic function (refered to as activation function) of the weighted sum of its inputs (see figure 4.4).

Perceptrons are not so much implemented in analog hardware. They have originaly been formulated as a mathematical rather than an electronic model and traditional computers are good at those whereas it is not so straight forward to implement simple maths into aVLSI. Still there do

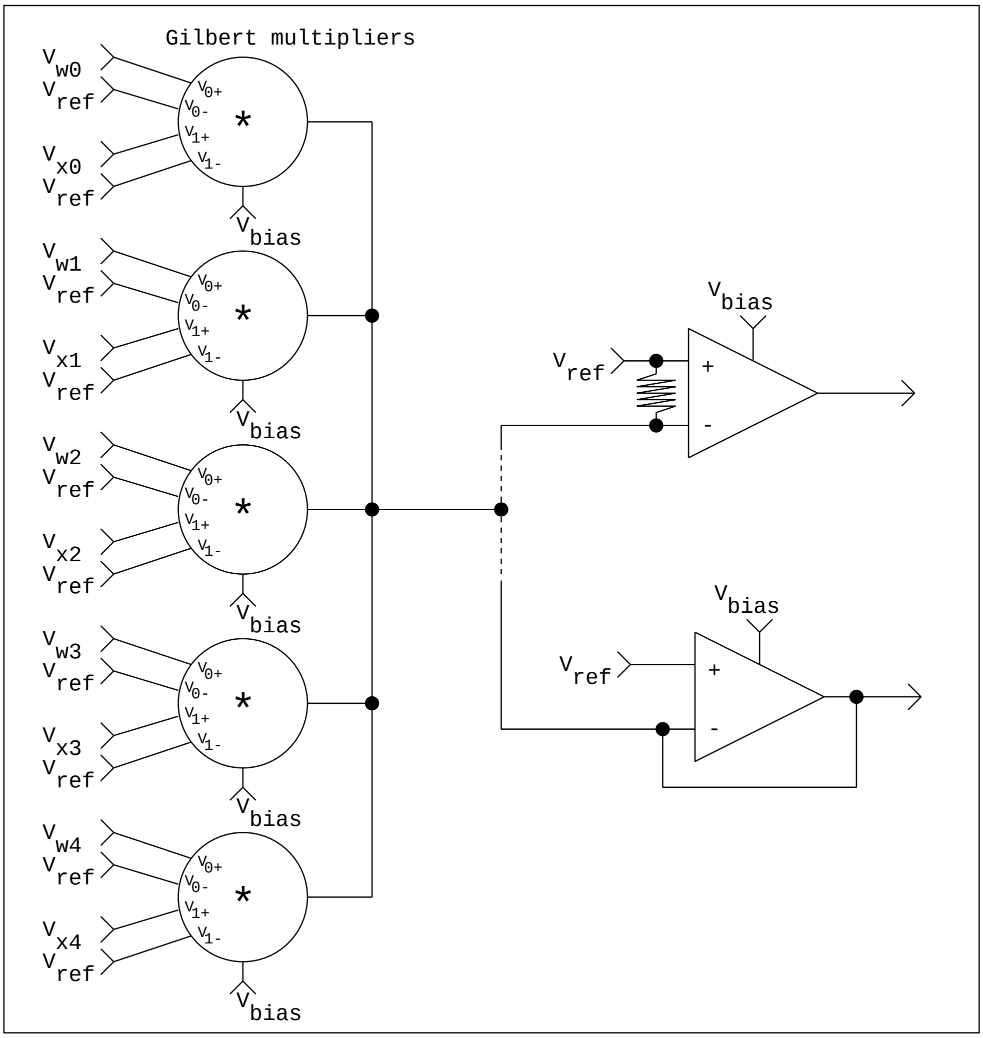


Figure 4.5: A possible implementation of a Perceptron

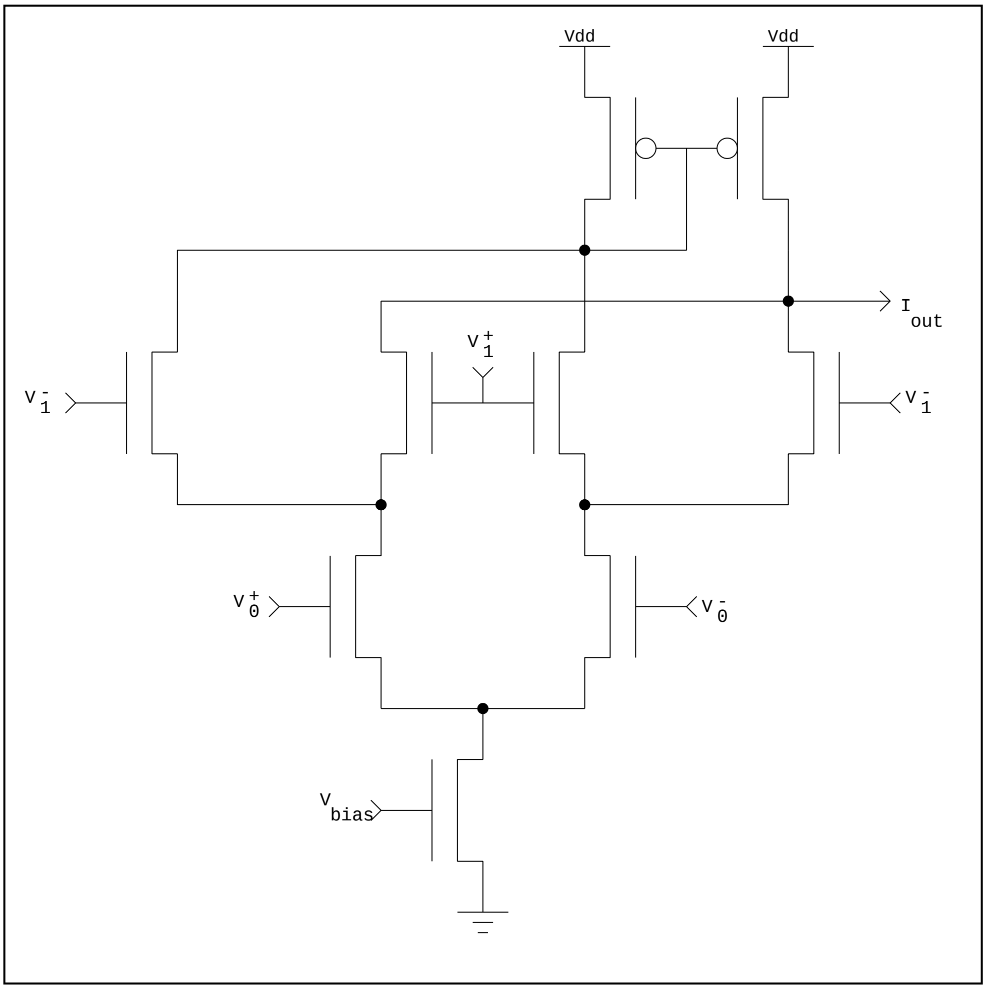


Figure 4.6: The Gilbert multiplier, multiplying differential input voltages, resulting in an output current

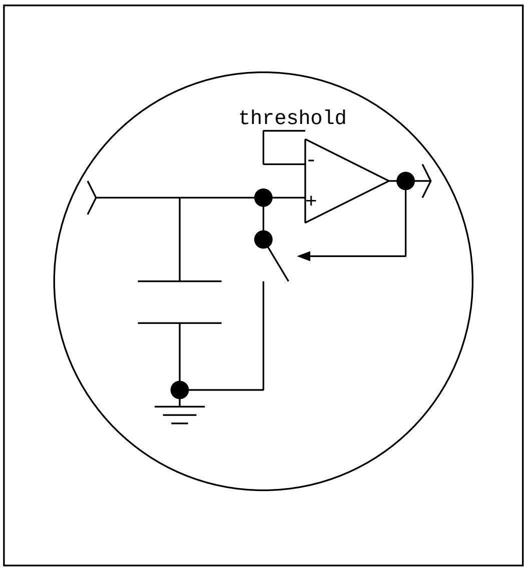


Figure 4.7: Concept of the Integrate-and-Fire neuronal model

exist aVLSI implementations of perceptrons since they still promise the advantage of a real fully parallel, energy and space conservative implementation.

A simple aVLSI implementation of a perceptron is given in the schematics in figure 4.5. This particular implementation works well enough in theory, in praxis however it is on one hand not flexible enough (particularly the activation function), on the other already difficult to tune by its bias voltages and prone to noise on the a chip. Circuits that have really been used are based on this one but were more extensive to deal with the problems.

The basic Gilbert multiplier schematics that is used in the proposed perceptron is shown in figure 4.6.

4.2.3 Integrate and Fire Neurons

This model of a neuron sticks closer to the original in terms of its signals. Its output and its inputs are pulse signals. In terms of frequencies it actually can be modelled by a perceptron and vice versa. It is however much better suited to be implemented in aVLSI. And the spike communication also has distinct advantages in noise robustness. That is also thought to be a reason, why the nervous system uses that kind of communication.

An integrate and fire neuron integrates weighted charge inputs triggered by presynaptic action potentials. If the integrated voltage reaches a threshold, the neuron fires a short output pulse and the integrator is reset. These basic properties are depicted in figure 4.7.

The most popular and very simple and elegant aVLSI implementation of an integrate and fire neuron of figure 4.8 has been proposed by Carver Mead [38]:

A neuronal property that is highly valued in some applications is that it attenuates its output when the input is constant. The neuron is said to adapt and is therefore more sensitive to transients (derivative) than to sustained activity. One possible extension of the above integrate and fire neuron to make it adaptive could be the one in figure 4.9.

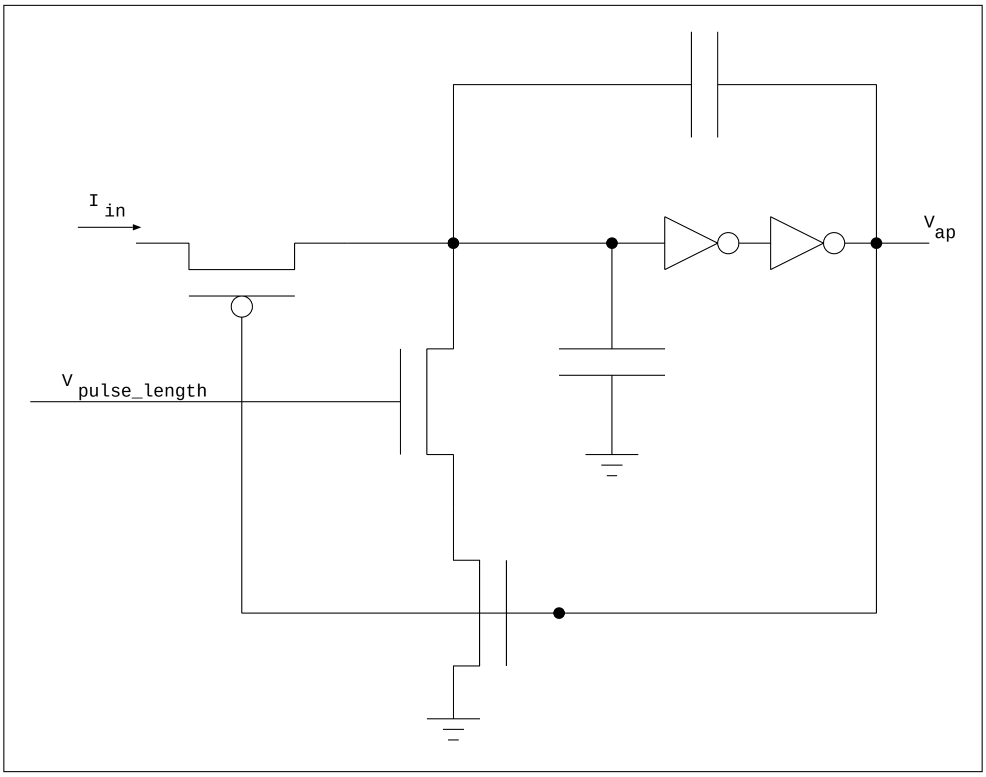


Figure 4.8: A implementation of an Integrate-and-Fire neuron according to C. Mead [38].

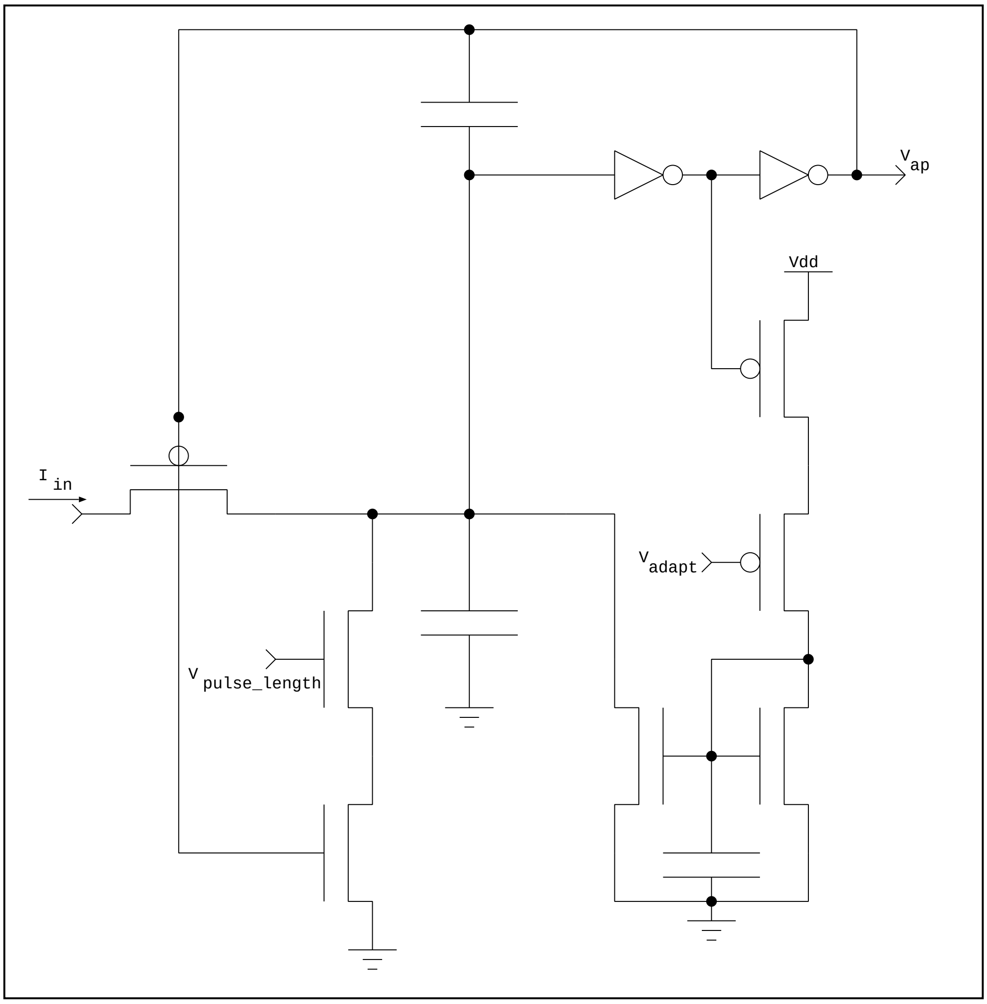


Figure 4.9: A possible implementation of an adaptive Integrate-and-Fire neuron

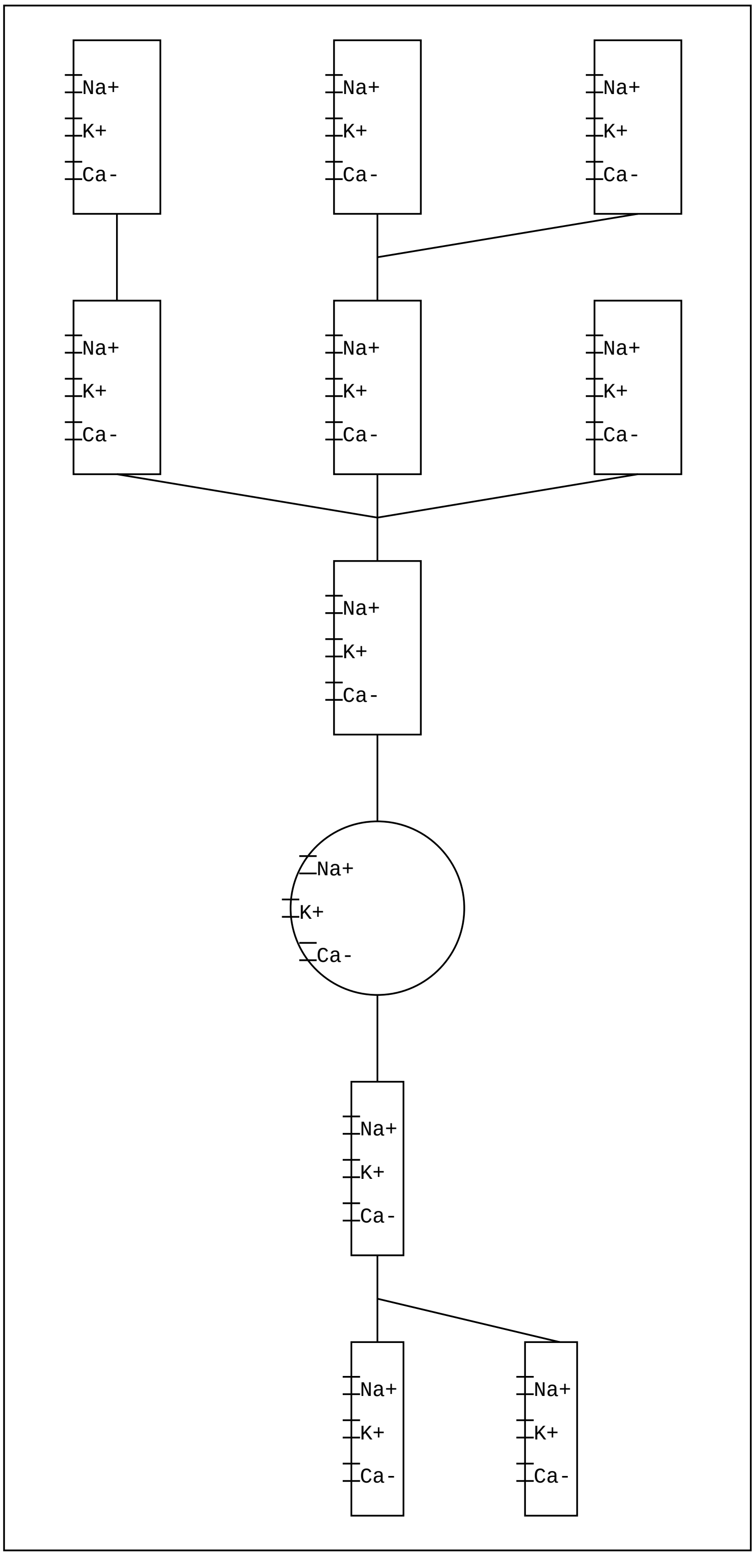


Figure 4.10: Concept of a compartemental model of a neuron, its dendrites and its axon.

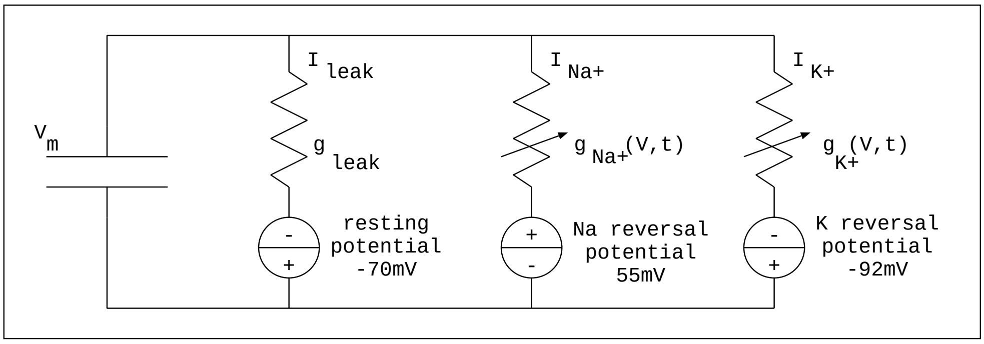


Figure 4.11: The Hudgkin Huxley model of action potential production, i.e. the active channel properties in the axon

4.2.4 Compartemental Neuronal Models (Silicon Neurons)

So called 'silicon neurons' [34, 45] take the analogy to real neurons even further. They take into account that a neuron is not uniform and has varying properties and states along its dendrites and the axon. Signals internal to the neuron do not spread instantly. These facts are modelled in 'silicon neurons' by assembling the cell out of connected compartments. Within one compartement different ionic currents that flow through ion-specific channels in real neurons are represented. These channels open or close dependent on voltage or concentrations (represented as voltages) of other ions or chemicals. Action potentials are not mere digital pulses but resemble more closely the real ones with sharp voltage increas as the membrane reaches threshold and undershooting as the action potential is turned of.

Such 'compartemental models' are also popular in software, but only aVLSI implementations make real time simulations possible.

On that level of detail the behaviour of neurons has been described as electric circuits from the start. Maybe the most popular example of this is the model of the spike generating circuit in the axon by Hudgkin and Huxley [24, 23] (figure 4.11)

This same model has been translated to CMOS aVLSI like depicted in figure 4.12 [45].

The connection between compartments is merly resistive. The passive behaviour of a chain of such compartments can be modelled by a cable equation (figure 4.13):

$$\frac{d}{dt}CV + \frac{V}{R_V} = \frac{d^2}{dx^2}\frac{V}{R_H} \tag{4.1}$$

If one node voltage in an infinite cable is caused to change, the neighbours will follow the change with a delay and with attenuated amplitude and speed. The effects of more complicated stimulations is more difficult to analyze and is therfore usually observed in numerical simulations or in a hardware model.

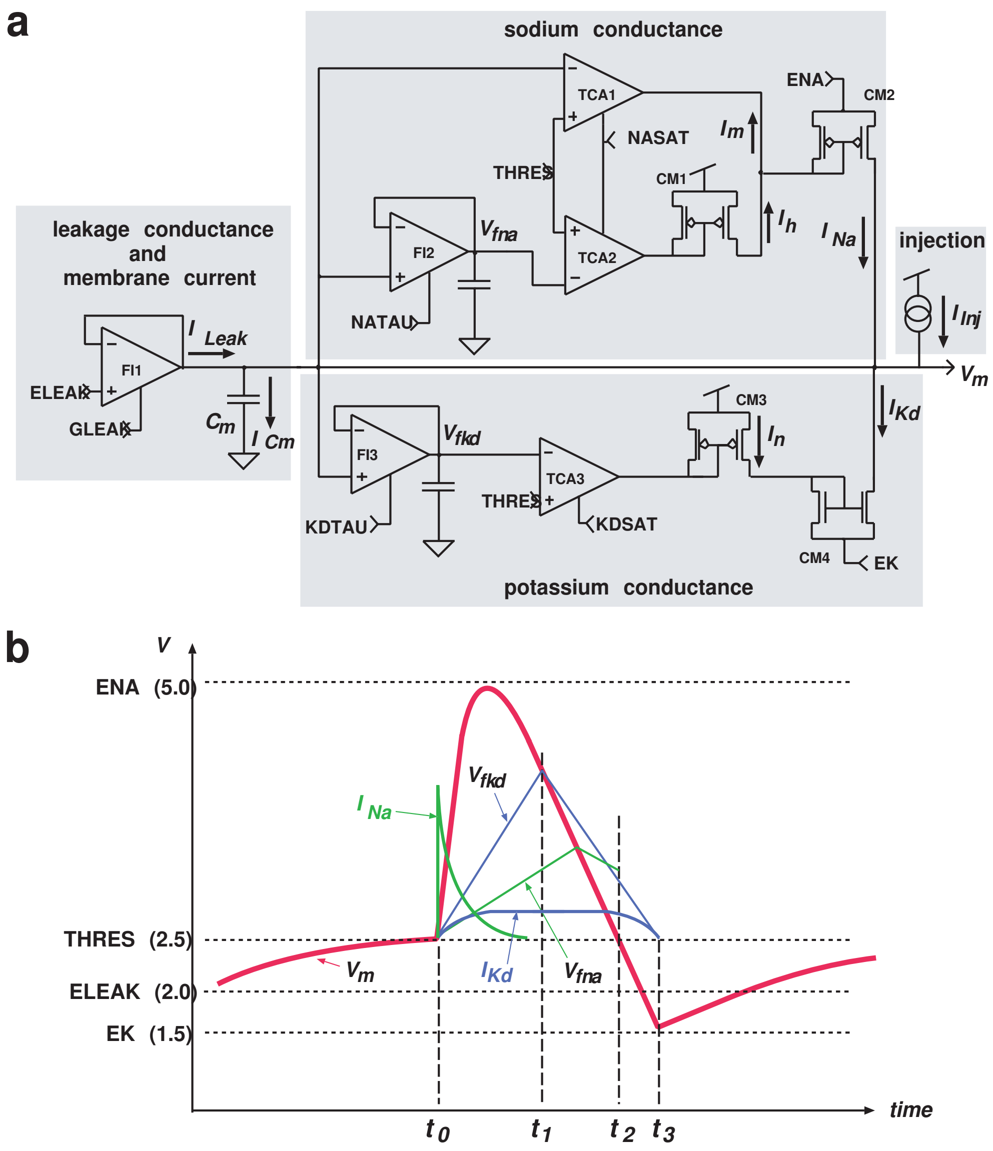


Figure 4.12: A CMOS Implementation of a HH-soma, a) schematics, b) typical signal time course

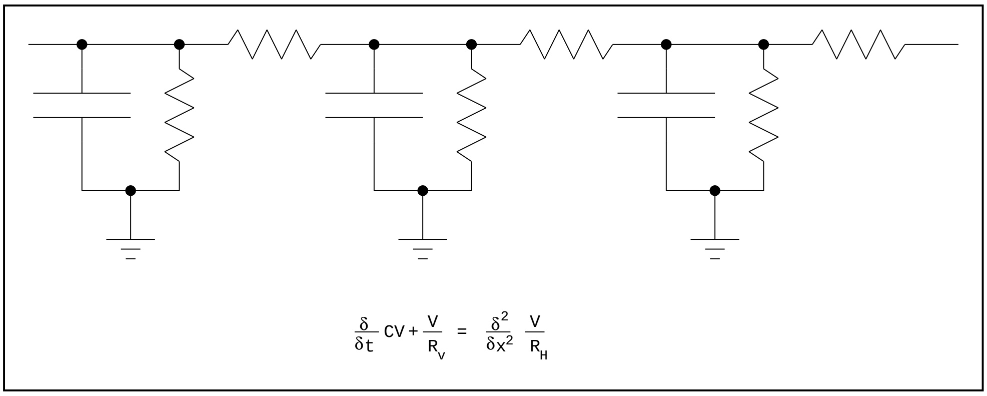

$\frac{\delta}{\delta t} CV + \frac{V}{R_v} = \frac{\delta^2}{\delta x^2} \frac{V}{R_H}$

Figure 4.13: Model of a cable

Chapter 5

Coding in the Nervous System

5.1 The action potential

The action potential (voltage pulse, spike) is the main form of information transmission in the nervous system. But also analog electrical signals are sometimes used and a lot of chemical processes are involved, for example at the synapse, in conveying the AP to the postsynaptic cell, and in initiating changes in the synapse that result in learning. Especially the exact workings of the later are still unknown. APs are considered to be digital in amplitude (ca. -70mV, 30mV), and fixed in duration (ca. 1ms). They are issued by most neurons with a minimal inter-spike interval of ca. 5ms.

How information is encoded with those spikes in the huge network that is the nervous system is not yet completely known. In the periphery of the nervous system we do have some knowledge however: We know a lot about how to stimulate optical, auditory and touch sensor-cells and about their responses. We also know some about how motor neurons stimulate muscles. But the waters get more murky in the 'deeper' parts of the nervous system.

The interesting questions in this context are 'Do we know how to reconstruct a stimulus from observing the activity in the nervous system?' and 'Do we know how to predict an action from observing activity in the nervous system?'. As mentioned above, the answer to those questions is yes, if we observe the peripheries of the nervous system, and if the stimuli and actions are simple (like a single flashing light point at a particular location in the visual field or the contraction of a single muscle). But for more complex stimuli (seeing the face of someone we know or experiencing a sunny day in autumn in the forest, with all the rich information that such an experience comprises) and actions (pushing the green button or going down to Giusepe's to buy a pizza) the answer to those questions is rapidly approaching 'absolutely not'.

5.2 Hints in Experiments

5.2.1 Classical experiments based on observing spike rates

Already for a long time researchers have tried to gather knowledge about the connection of brain activity and sensing and acting. They have been trying to find correlations between any kind of activity in the nervous system and stimuli or motor responses.

Twitching Frog Legs Experimenters: Galvani and Volta (1700)

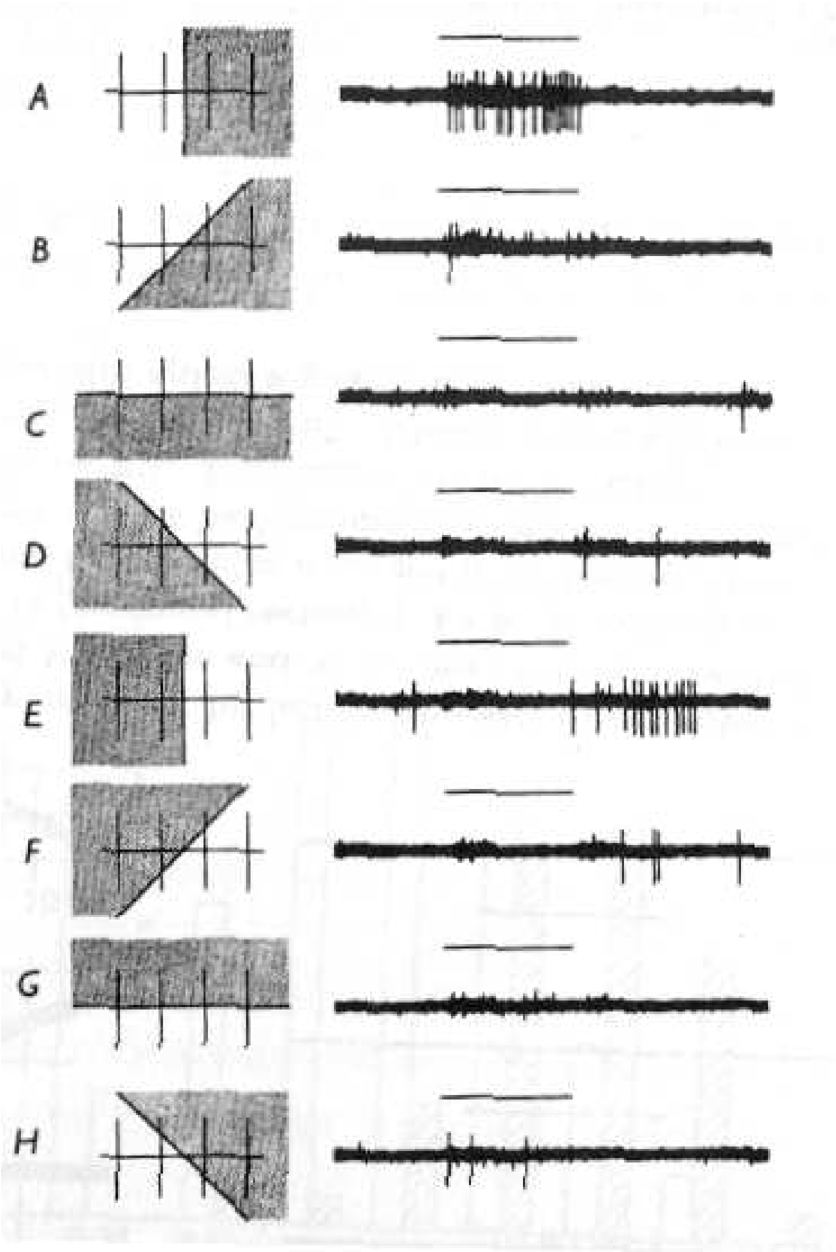


Figure 5.1: Orientation selective cells in cat visual cortex, stimuli and responses of one cell [27]

The first documented artificial stimulation of muscles. By touching the muscles in a frog leg with a metal, it could be made to twitch.

Simple and Complex Cells in Visual Cortex Experimenters: Hubel and Wiesel [27]

A real classic. Hubel and Wiesel were the first to document correlations between firing rates of cortical neurons in visual cortex and bar-stimuli that are characterized by an orientation and sometimes a direction of motion (Example in figure 5.1). Since then many people have documented 'receptive fields' and 'optimal stimuli' for cells in visual cortex, by looking for maximal response in terms of firing rate.

Motorneurons Muscle fibers react with a contracting force to an action-potential from a motorneuron. Steady force can be achieved by sending spikes with a steady frequency to the fiber.

Segmentation by synchronization Regular oscillations in population activity in neurons is a widely spread phenomenon in the brain. Some researchers suggest that observed features that are represented by oscillatory activity of particular populations will synchronize the oscillations for features that belong to the same object.

5.2.2 Classical Experiments observing temporal spike codes

There are however experiments that suggest that there are more details about some stimuli encoded in the precise firing patterns of individual neurons. A provocating thought along this line is that it is not necessary to change the average activity of a neuron to transmit information. So maybe some of the information about stimuli is encoded in places where no-one did bother to look yet.

Highly consistent firing patterns for high entropy stimuli Experiment by: Bair and Koch [2] (figure 5.2)

One particular discovery was made by accident, so the rumors say. During a recording from a neuron (in the MT cortical area) a lab-animal was repeatedly presented with a moving random dot pattern. Surprisingly the cell responded very consistently with almost the same spike train (see figure 5.2). It turned out that due to a programming bug, the random pattern was always created with the same random-generator-seed and thus it was the exact same pattern in every trial. So it seems that if the stimulus has high entropy (in the information theoretical sense) a lot of detailed information is contained in a single cell's spike pattern. And it also seems that a neuron is quite an exact device. By contrast, if the stimuli are simpler, e.g. moving bars, then the detailed pattern looks random and different from trial to trial. Maybe such 'simple' stimuli need not be represented by the full capacity of the neuron.

Synfire Chains Experiment by: Moshe Abeles [1]

Some researchers noticed a strange thing when recording from multiple units in the cortex. A specific while after a stimulus had been presented, a particular group of neurons consistently tended to fire an AP simultaneously. Other than that the firing patterns looked more or less random. The researchers explained that phenomenon by a 'synchrony code' model known as synfire chain (figure 5.3). They propose that groups of neurons fire a spike in synchrony (triggered by some kind of meaningful information event) and that this synchronous activity propagates from group to group, establishing a temporal relation to the triggering event. One neuron may participate in several of these groups. Thus, observing only one neuron, this will very much look like Poisson distributed random activity.

Place Cells in Hippocampus Experiment by: O'Keefe and Recce [42]

Place cells are cells in Hippocampus, often observed in Rats. They become active if the rat is in a particular place within an environment. This place can be reconstructed by looking at

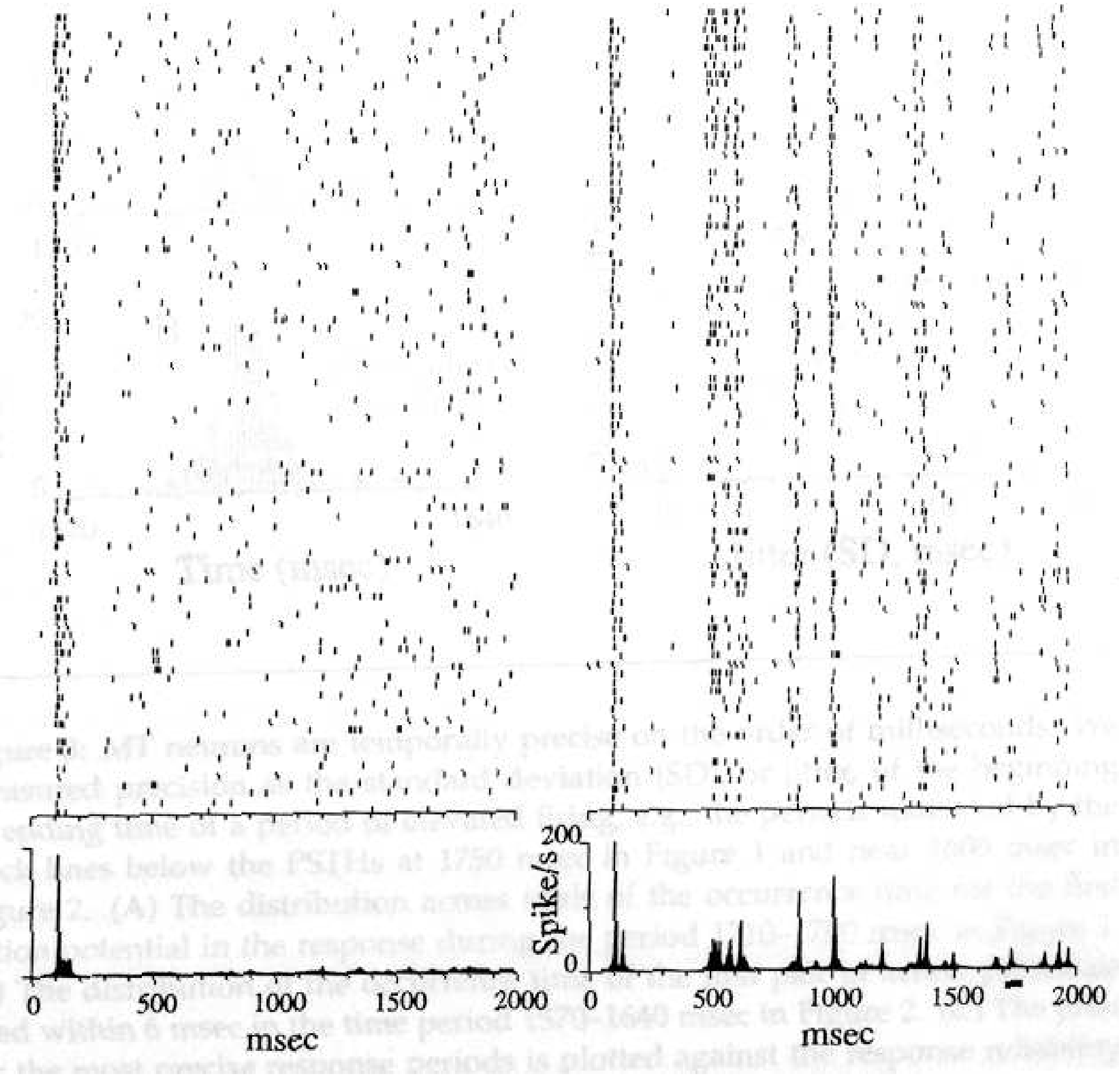


Figure 5.2: Neuron response to a moving random dot pattern, when repeatedly presented the very same random dot pattern (right) and different random dot patterns (left) [2]

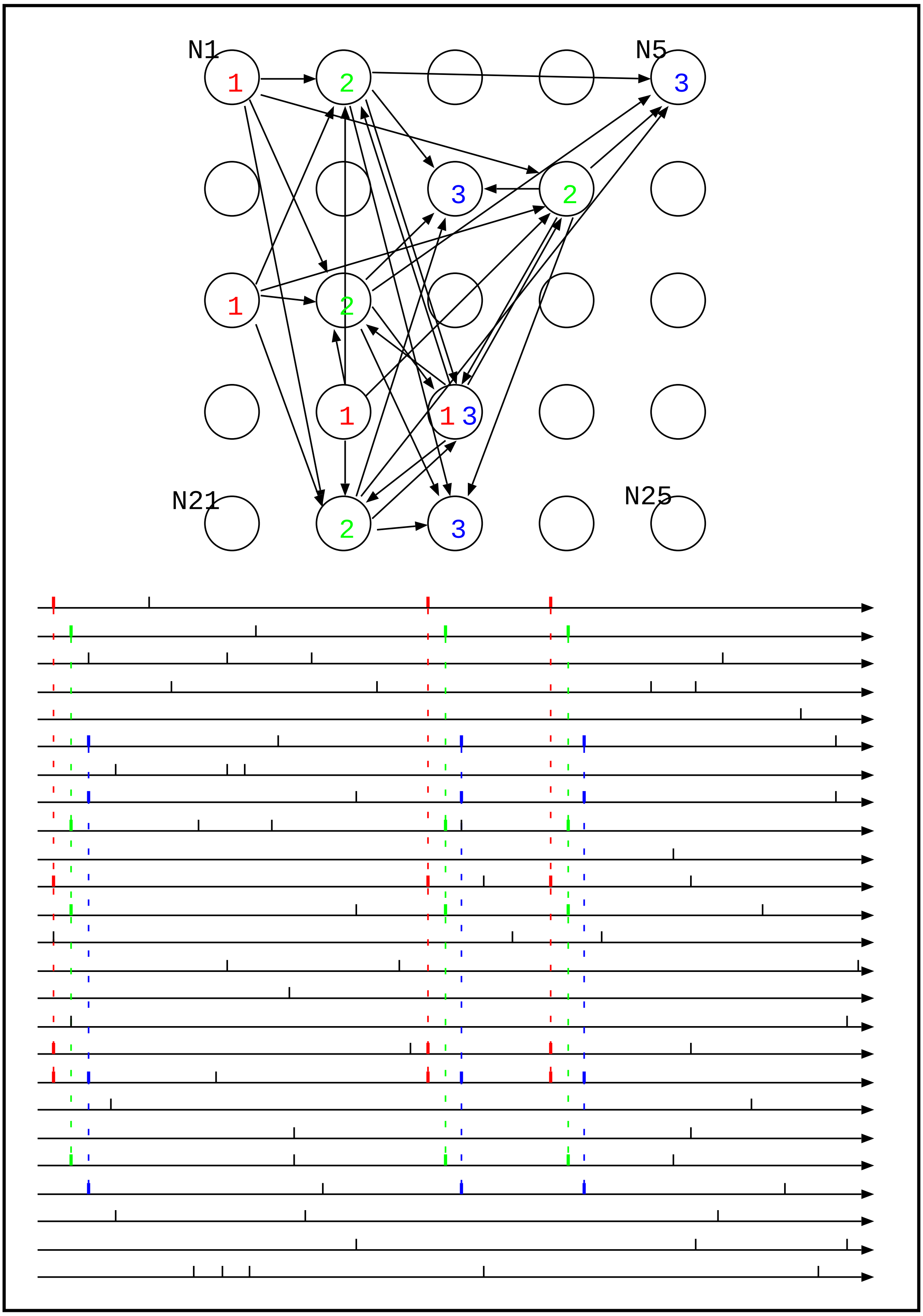


Figure 5.3: The concept of Synfire Chains originally proposed by M. Abeles [1]

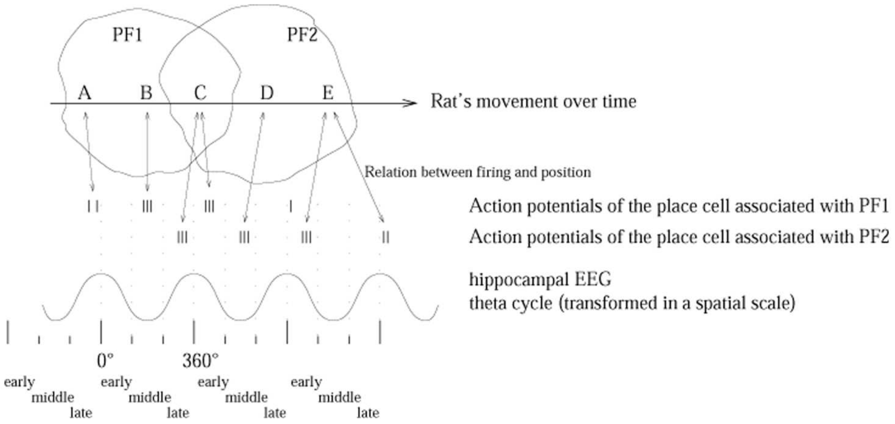

PF1
PF2
A
B
C
D
E
Rat's movement over time
Relation between firing and position
Action potentials of the place cell associated with PF1
Action potentials of the place cell associated with PF2
hippocampal EEG
theta cycle (transformed in a spatial scale)
0°
360°
early
middle
late

Figure 5.4: Phase relation of place cells' activity to synchronous background activity indicating more detailed position within the receptive field of the cell [42]

the rate of the neuron. But it has turned out that by looking at the temporal firing pattern of the cell in relation to the entire population theta rhythm activity (observed by EEG) one can even reconstruct the rat's position within that place more precisely (figure 5.4). That more detailed position information is encoded in the phase relationship between the cells burst activity and the population theta rhythm. (theta rhythm: 5-8Hz)

Spike Based Learning Experiments by: Larry Abbot etal. or Henry Markram etal. [36]

Experiments that do not directly allow to make a connection between a stimulus and a temporal firing pattern, but that show the interest of the nervous system in particular temporal patterns, are the experiments that revealed changes in synaptic strength caused by particular firing sequences. This is explained in more detail in chapter 9 about learning.

Reaction time in object recognition Experimenter: Simon Thorpe

These experiments are an intriguingly simple and a convincing argument for the relevance of temporal encoding in the nervous system. Monkeys and human test subjects were presented pictures and had to decide if there was food (or an animal in another series of experiments) in the picture or not. they had to push one of two buttons accordingly. The reaction time for this task was about 200ms-500ms. The experimenters argue that the neurons along the processing path through the nervous system, from the optical input to the motor reaction, do only have time to fire one to two spikes during that time. So there is no time to compute a temporal average. The information is conveyed with only a few spikes per neuron. Candidate encoding schemes are population or purely temporal.

5.3 Candidate Codes

Based on all of these experiments theoreticians propose many different ways of extracting information from neuronal activity or to encode information in artificial neural systems. These encoding schemes can be divided into several, not mutually exclusive classes. All of them look at patterns of neuronal activity over time and space, some of them look at average activity (over time (and

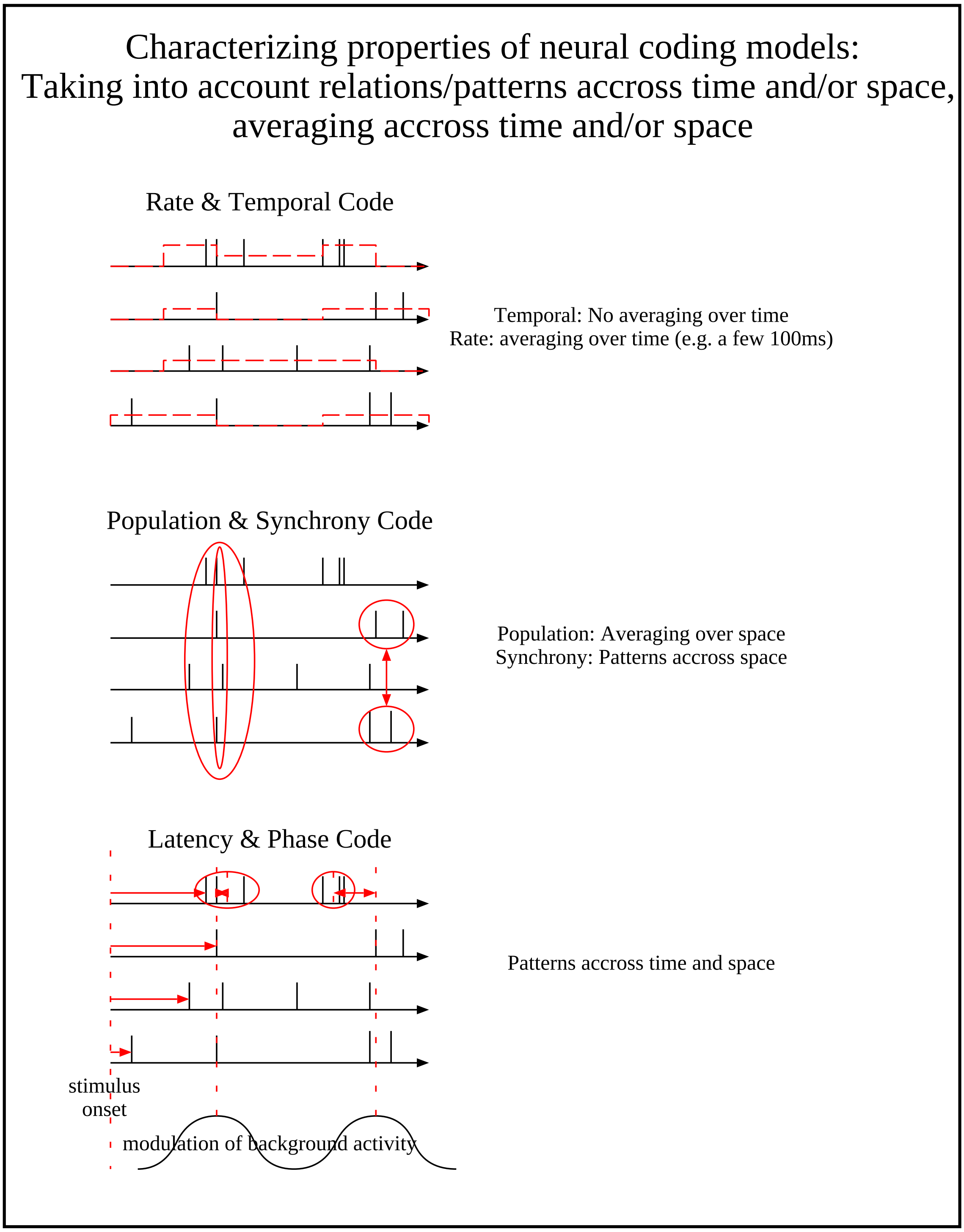

Characterizing properties of neural coding models:
Taking into account relations/patterns accross time and/or space,
averaging accross time and/or space
Rate & Temporal Code
Temporal: No averaging over time
Rate: averaging over time (e.g. a few 100ms)
Population & Synchrony Code
Population: Averaging over space
Synchrony: Patterns accross space
Latency & Phase Code
Patterns accross time and space
stimulus
onset
modulation of background activity

Figure 5.5: AN illustration of the coding schemes mentioned in the text

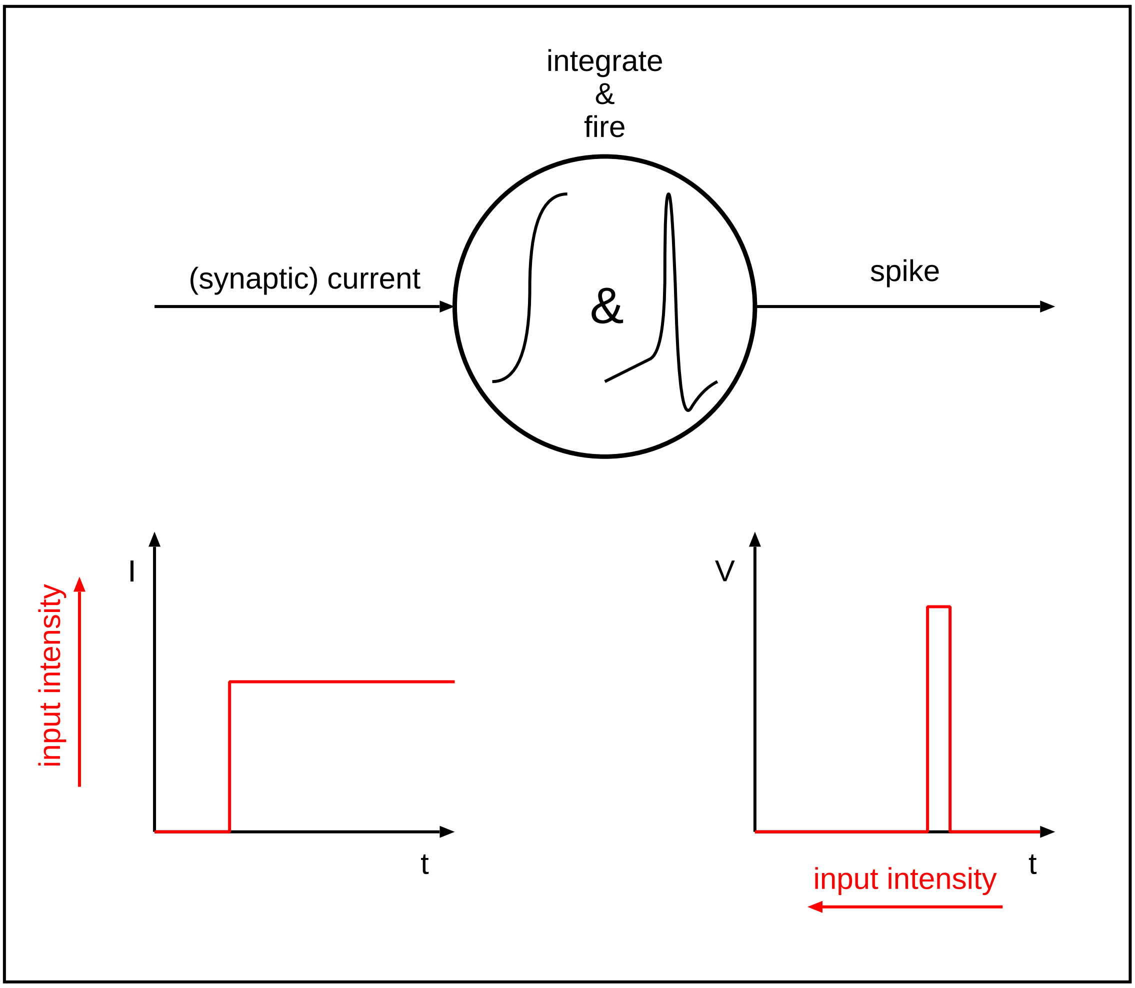


Figure 5.6: One popular temporal coding scheme that looks at spatio temporal patterns in a group of neurons is latency encoding, or rank-order encoding. It looks at the first spike of each neuron after stimulus onset. An I&F neuron naturally encodes the inverse of stimulus strength in the latency of this first spike.

space): rate coding; over space population coding) some of them rather at spike patterns (over time (and space): temporal coding; over space: synchrony coding). Figure 5.5 illustrates some of these coding concepts.

Some properties of some of these coding schemes are listed in the following:

Rate Codes Information (about stimulus or action or thoughts) contained in: Spike activity averaged over time. Properties: Robust against noise, good spatial resolution, but slow

Population Codes Information contained in: Spike activity averaged over space Properties: Robust against noise, fast, but worse spatial resolution

Temporal Codes Information contained in: Spatio-temporal spike patterns Properties: Fast with a good spatial resolution but sensitive to noise

example: Latency Code, time to first spike (see figure 5.6)

example: Phase Code: Single spikes in relation to population activity oscillations.

example: Synchrony codes (mixture with population code)

Chapter 6

Neuromorphic Communication: the AER Protocol

The way of communicating information within the nervous system is rather different from the methods used in most artificial electronic systems. Most neurons communicate by way of nerve pulses (action potentials) via dedicated point-to-point connections (axons). On the other hand the communication channels between computers or inside computers (such as busses) transmit more complex signals at higher rates than an axon. The physical channels are mostly shared and time multiplexed by several components and their density can therefore be kept lower.

As neuromorphic engineers try to implement systems that are organized more like the nervous system, the brain's way of communicating internal signals becomes a major obstacle to electronics. The sheer numbers are staggering: The human brain contains about 10^{11} neurons each of which has 1000 to 10000 synapses. All those synapses receive input via a dedicated output line of the sending neuron. The so called white matter underneath the folded surface of the cortex (the most recently developed part of the brain in evolution, believed to hold conscious functions) is densely packed with such connections between cortical areas and other brain parts. It holds much more volume than the gray matter, the surface layer of the cortex which holds the neurons. The most progressed digital computer chips cannot by far provide the same density of wires. That is mainly because computer chips are confined to two dimensions in their layout whereas the nervous system makes full use of all the volume it occupies.

6.1 The Basic Idea of Address Event Representation (AER)

The most prominent idea of how to come closer to the capacity of point-to-point in the nervous system is the one of address event representation (AER) [33, 39, 44, 6]. It uses the advantage of speed in digital systems to make up for the disadvantage in density and emulates point-to-point connections rather than physically implement them.

The principle is intriguingly simple: Each neuron in a network is assigned an address. When a neuron wishes to send a nerve pulse (action potential) it places its address on a digital bus via an encoder. Synapses that are supposed to receive that action potential are connected to the same bus via a decoder and get stimulated when their address occurs. The bus can unfortunately only transmit APs serially but it can do it much faster than an axon. It can transmit APs so tightly spaced in time that for a network of neurons that operate on a natural time-scale these pulses are virtually simultaneous.

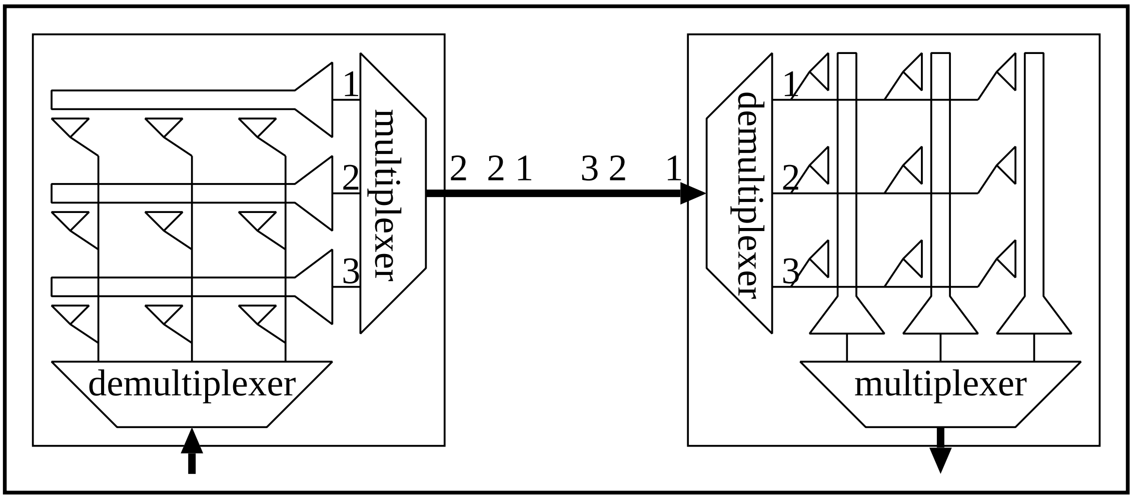


Figure 6.1: The basic idea of Address Event Representation (AER)

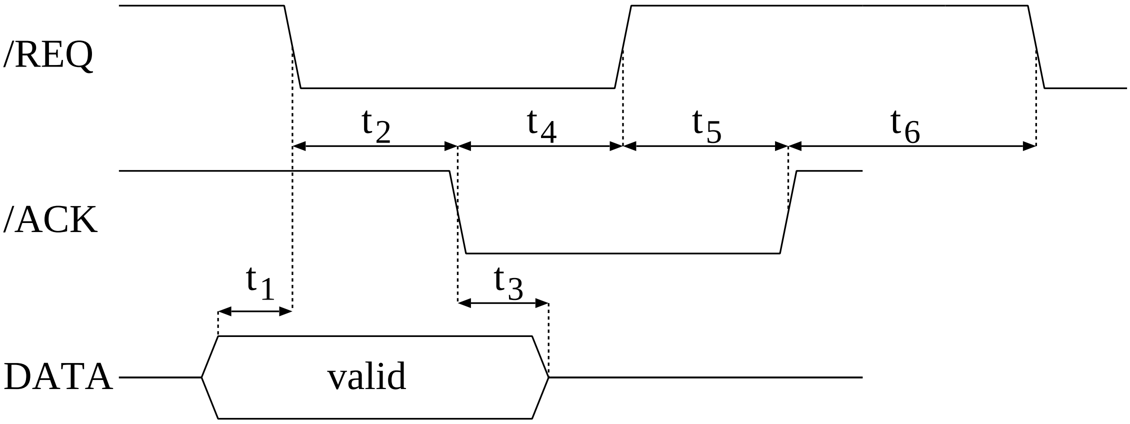


Figure 6.2: Asynchronous control signals for a four phase handshake communication of one data package

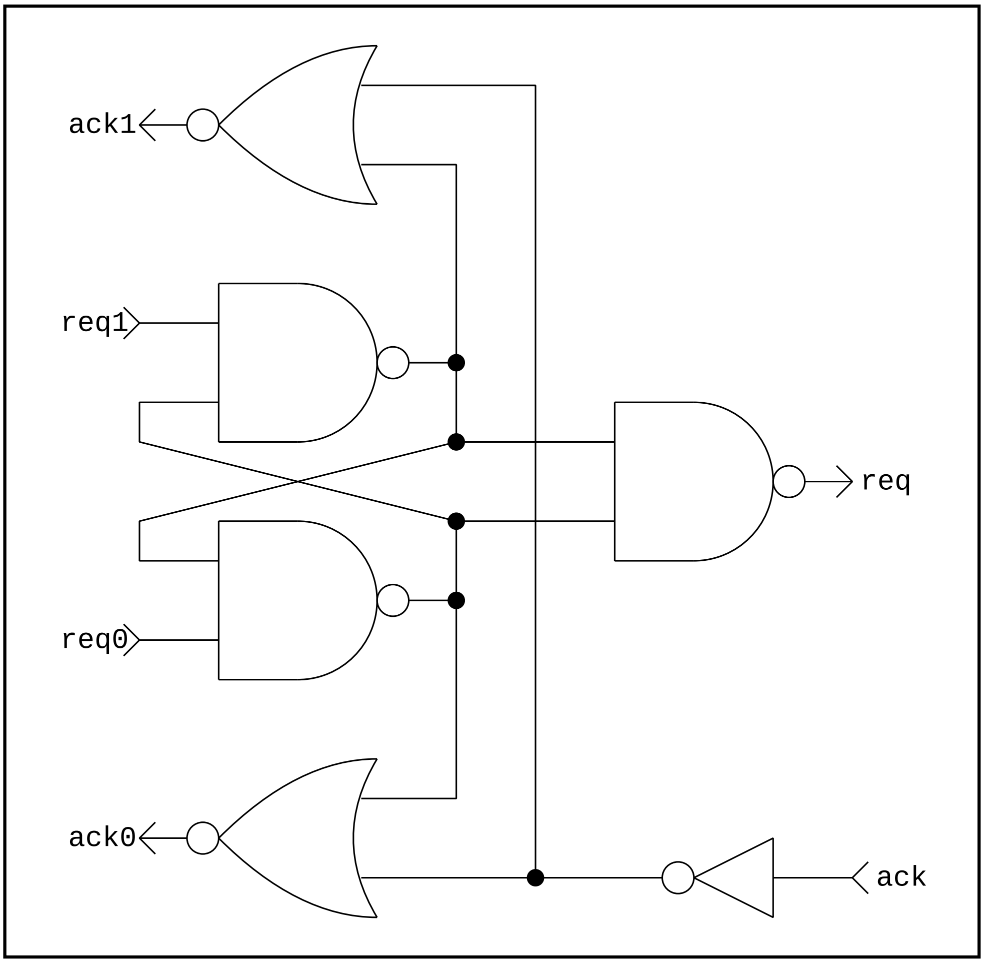


Figure 6.3: A two-input 'greedy' arbiter

6.2 Collision Handling

Since multiple point-to-point connections share the same bus there is need for a bus control mechanism or a collision handling. Different research groups have developed different approaches to that problem. The main principles are:

- full arbitration (The Caltech solution [33, 6] and WTA arbitration [29])
- discarding (The Swiss solution [39])
- aging versus loss trade off (The Oslo solution [35])

6.2.1 Full Arbitration

In full arbitration [33, 6] the workload lies with the sender. Collisions are resolved before sending an APs address via the bus. Neurons request bus access from an arbiter if they want to transmit a spike. The arbiter can for example be implemented as a binary tree of elements that handle two input requests and issue a request of their own to the next layer in the tree-structure. Such a binary arbiter cell can be implemented like shown in figure 6.3:

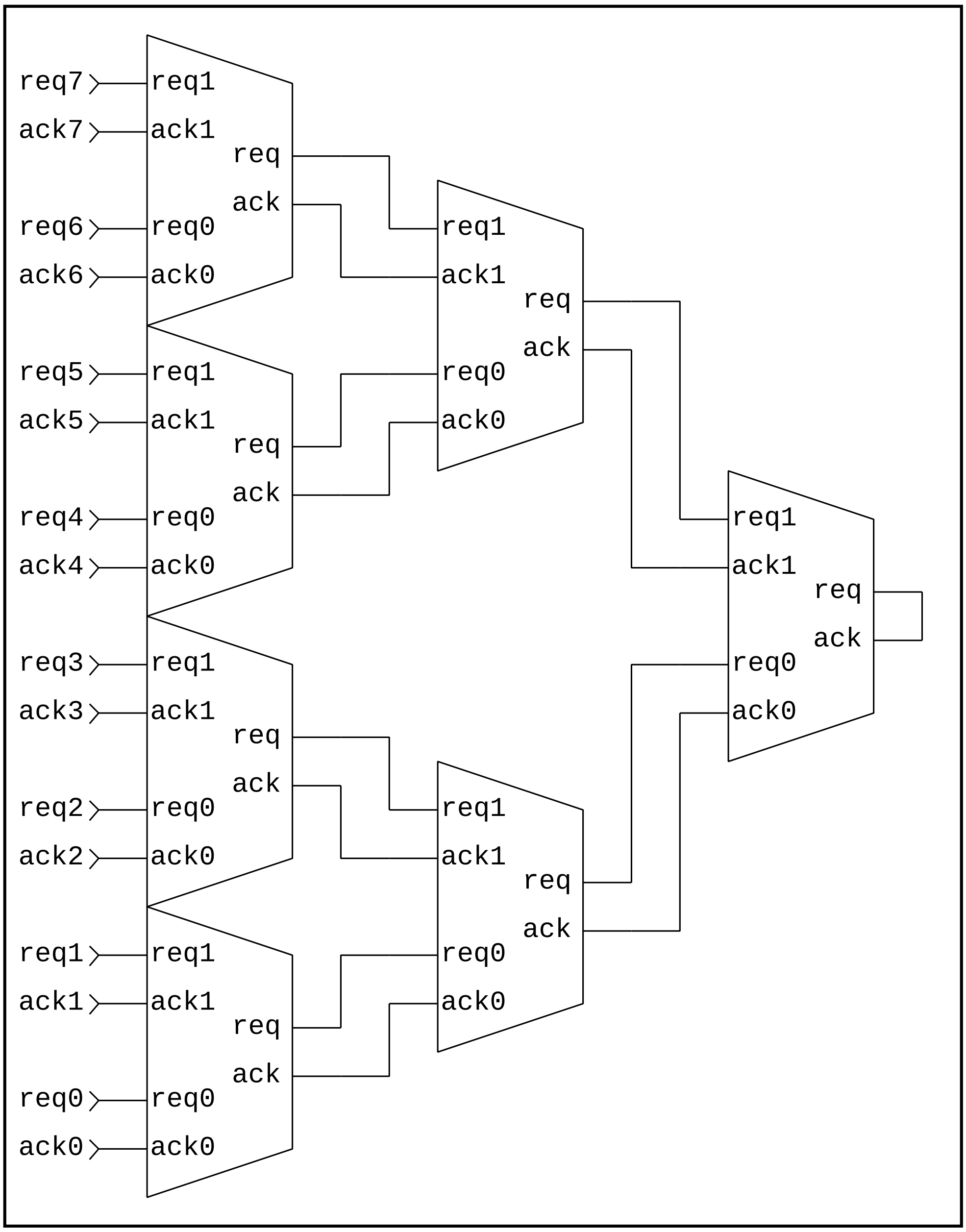


Figure 6.4: A binary tree of 'greedy' arbiters

The two coupled NAND gates can be looked at as a RS flipflop with active low inputs. Normally if there is no request active this RS flipflop is in its 'illegal' state when R and S are set. Both outputs will be forced to 1. Which state the flipflop will fall into is determined by which of the two inputs, S or R, is withdrawn first. Or in other words which active high request ('req0' or 'req1') comes in first. If req0 is the winner, then the output of the lower NAND gate will become 0 and the signal 'req' will be propagated to the next layer of the tree. If that 'req' is granted by an incoming 'ack', 'ack0' is set. Following a four phase handshake protocol the requesting instance will thereupon withdraw its 'req0'. If in the meantime 'req1' has been set, that request gets also granted without releasing 'req'. That principle of serving all local requests first before giving control back to the higher level in the hierarchy is called greedy arbitration. Only if there are no more request from the left ('req0' and 'req1') is 'req' withdrawn and other arbiter cells on the same level get a chance of being acknowledged.

These arbiter cells can be assembled into a binary tree (figure 6.4). The top most 'ack' and 'req' need to be shorted together.

pro

- fast

  Since this method of arbitration is among the fastest methods it can reach a much faster throughput than other methods before the probability of collisions becomes significant.

- No loss of APs

  In case of a small number of collisions, no spikes will be lost. Only in the case of heavy overload on the bus it might happen that neurons are delayed for so long that they would like to spike again before the first AP could be transmitted. In that case the second spike is lost.

contra

If the load of request begins to exceed the throughput of the bus, some non idealities start to show their influence.

- unfair arbitration

  Because of the greedy arbitration neighbours in the binary tree structure of the transmitting neuron get priority over more distant ones. That can lead to a total monopolizing of sub regions of neurons of the communication resources in case of continuous collisions. But one might say that in that case the heavy overload on the bus renders the transmitted data meaningless anyway.

- uncontrolled delays

  In applications where a high temporal precision is needed already the jitter in the spike timing caused by a few collisions might be a problem.

Another variant of full arbitration is WTA arbitration [29]. This variant is somewhat slower, though, than the greedy arbitration. And dependent on the actual implementation, it might even be more unfair than the greedy/binary-tree arbitration (e.g. if there is a default winner due to circuit mismatch). The 'aging versus loss' concept described later in this chapter also uses a WTA arbitration and does actually achieve fairer arbitration.

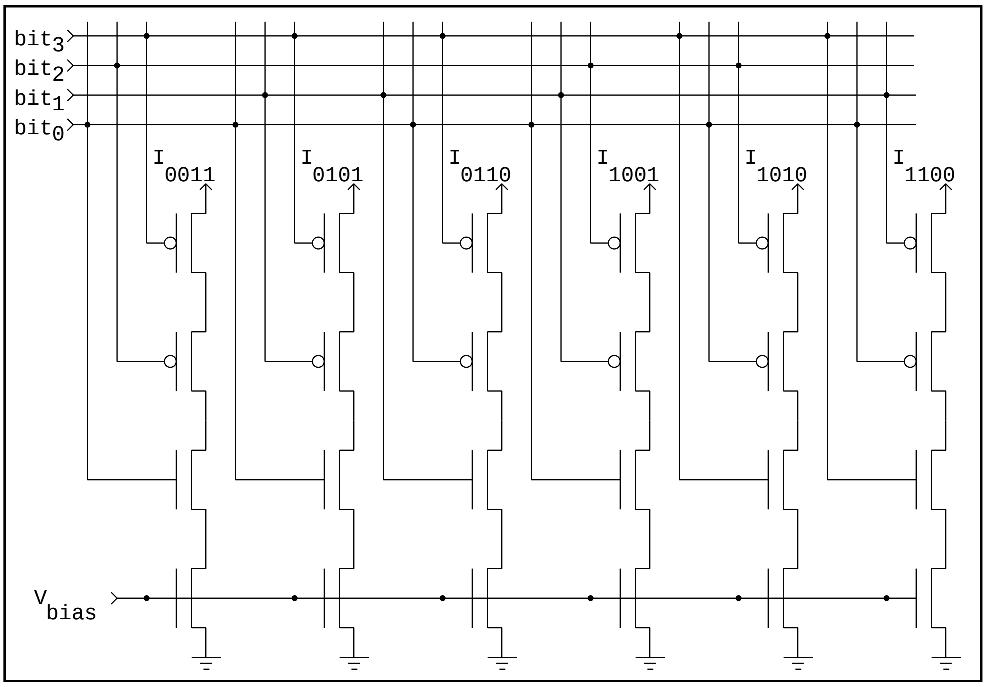


Figure 6.5: An AER receiver that discards addresses that are rendered invalid by a collision

6.2.2 Collision Discarding

This strategy [39] resolves collisions on the receiver side. Any sender can anytime pull the high bits of its address up. There is some redundancy in the address codes such that collisions can be detected. One coding that has been proposed was to only use codes with a fixed number of 1s. It has been claimed that with a code length of n bits and allowing only codes of n/2 1s one needs only two bits more than a normal binary number encoding, for $n < 20$. In other words one needs n bits to encode 2^{n-2} addresses. Events on the receiver side can be decoded as depicted in figure 6.5

The addresses are decoded by their zeros. Only one of the ones needs to be monitored to detect an event. It is assumed that the output currents get integrated and that a pulse is much longer than any glitches that might occur due to switching. In that way only an intentional pulse contributes significantly to the integrated current.

pro

- easy circuitry (e.g. no handshake)
- no distortion in the timing of transmitted pulses

contra

- loss of pulses
- pulses need certain width
- specific requirements of the receiver

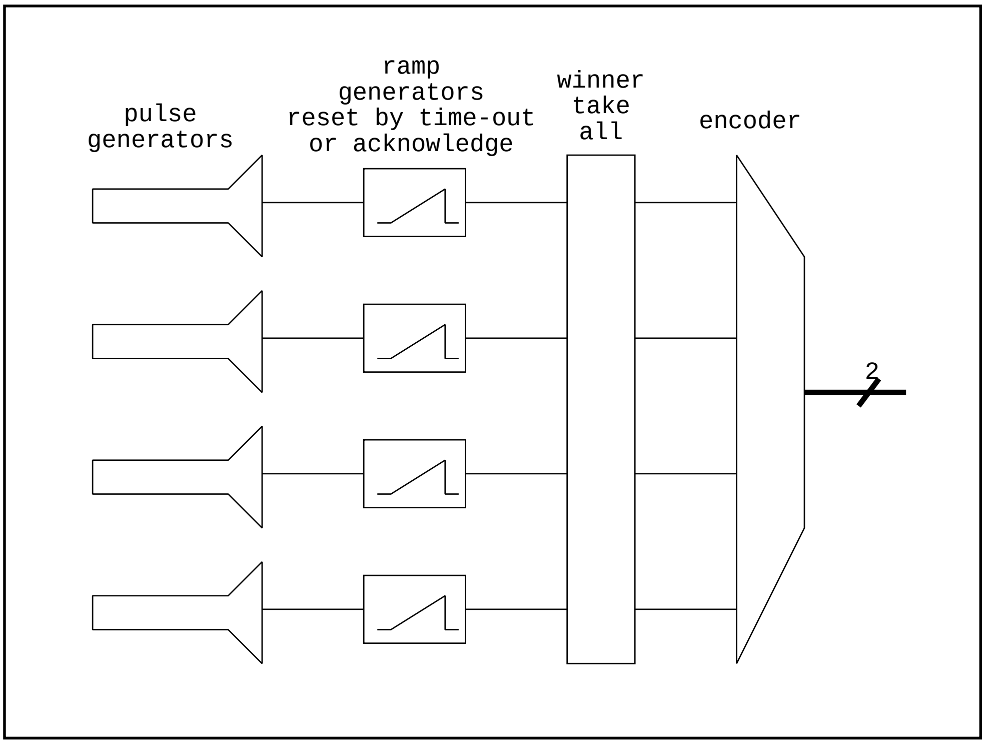


Figure 6.6: The principle of 'Aging versus Loss trade Off' AER communication

6.2.3 Aging versus Loss Trade Off

This concept [35] is a compromise between the other two. The idea is that full arbitration is a fine thing usually, but if one starts to delay pulses too much, one is better off discarding them. In addition the suggested implementation offers fair arbitration.

The principle is that of a fifo queue with a time out. The implementation is depicted in the blockdiagram in figure 6.6: A spike triggers the linear rise of a voltage. A WTA circuit determines the highest voltage among those ramped up voltages of all neurons. The neuron with the highest voltage is granted bus access and the voltage is reset to its idle state. Thus, something similar to a FIFO is implemented. If the ramped voltage reaches a threshold without getting access to the bus, it is reset anyway. Like this, pulse transmissions that are delayed for too long, get discarded.

pro

- possibility of trading loss versus maximal delays

contra

- slow
- extensive circuitry

Chapter 7

Retinomorphic Circuits

7.1 The Retina

The retina is the photo sensitive layer at the back of our eyeball (figure 7.1).

It comes as quite some surprise to most people, how much image processing in the human visual system is done before we consciously perceive an image. Taking one glance at a scene, we seem to see the whole picture as a camera would see it. But that is not really the case: The information that really reaches the higher areas of our brain is highly condensed. Just the relevant features of the picture are present here and a lot of details have been discarded.

Already the eye is more specialized than a camera. The retina which is the photo-sensitive part of the eye, what would be the film in a photo camera, is not uniform. We only perceive a clear colour camera-like picture in the very center of our visual field. With distance from the center the resolution and the colour perception decreases and the light and motion sensitivity increases (Example: when moving a hand forward from out of sight to the side of ones head, we are able to spot it sooner if the fingers are wiggled). The image is further processed before it is sent on to the Thalamus and the brain. There is some horizontal interactions going on between the nerve cells on top of the photo receptors that help adapt cells to the average light level (Example: looking out a window we can see features outside and inside at the same time, although the illumination levels are some orders of magnitude different. A camera can only adjust its shutter to either the inside light level (with the window appearing one brilliantly white spot) or to the outside light level (with the inside as a uniform dark frame).) and to enhance edges (Example: if you look at walls in a white room that meet in a corner, one wall will be more bright and the other more dark because of the angle of the main source of illumination. And very close to the corner, you will perceive the brighter wall becoming brighter still, and the darker more dark. Thus, the contrast between them is enhanced in your eye, but physically the brightness is not really changing towards that corner.).

A conceptual model that shows some of the nerve cells that lie on top of the actual photo receptors could look something like figure 7.2, or even more schematized and simplified in figure 7.3. The photo receptors (rods (peripheral vision, not colour- but very illumination-sensitive: better for night vision) and cones(colour sensitive, concentrated in the center of the visual field)) are excited by light that is focused on them through the eye lens. They adapt or tire when stimulated and the signal they are sending out is thus attenuated when presented with a constant stimulus. The photo receptors in turn excite so called horizontal cells that actually collect input from a group of photo receptors and from their colleagues. Such their own activity reflects the collective average light level of a certain neighbourhood. The difference between that collective light level and the local light level is computed in the bipolar cells. Some of them get excited (arrow synapses in figure 7.3) by the direct input from the photo receptor and subtract (bubble

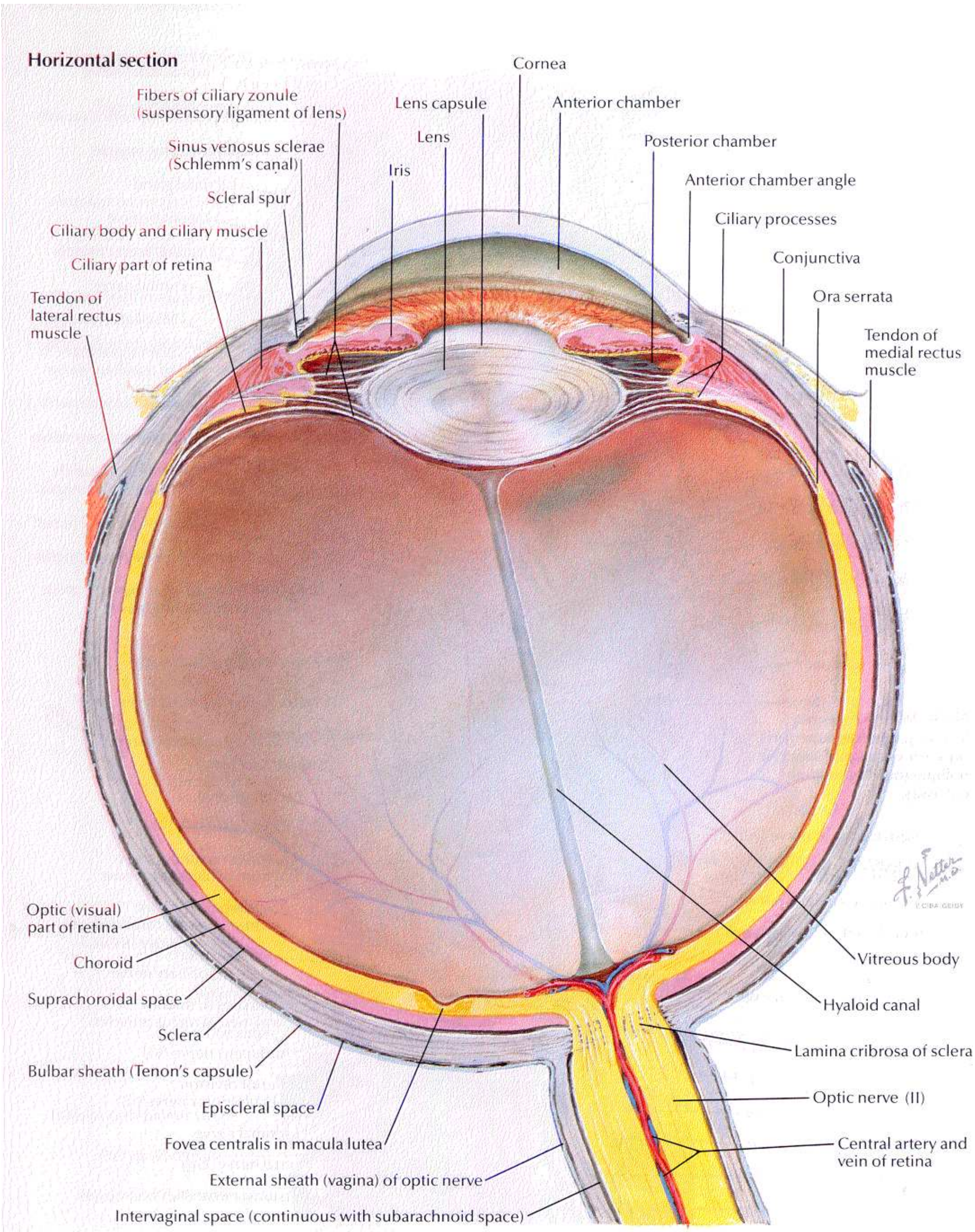

Horizontal section
Cornea
Fibers of ciliary zonule (suspensory ligament of lens)
Lens capsule
Anterior chamber
Lens
Posterior chamber
Sinus venosus sclerae (Schlemm's canal)
Iris
Anterior chamber angle
Scleral spur
Ciliary processes
Ciliary body and ciliary muscle
Conjunctiva
Ciliary part of retina
Ora serrata
Tendon of lateral rectus muscle
Tendon of medial rectus muscle
Optic (visual) part of retina
Choroid
Vitreous body
Suprachoroidal space
Hyaloid canal
Sclera
Lamina cribrosa of sclera
Bulbar sheath (Tenon's capsule)
Optic nerve (II)
Episcleral space
Fovea centralis in macula lutea
Central artery and vein of retina
External sheath (vagina) of optic nerve
Intervaginal space (continuous with subarachnoid space)

Figure 7.1: Cross section of the eyball according to [40]

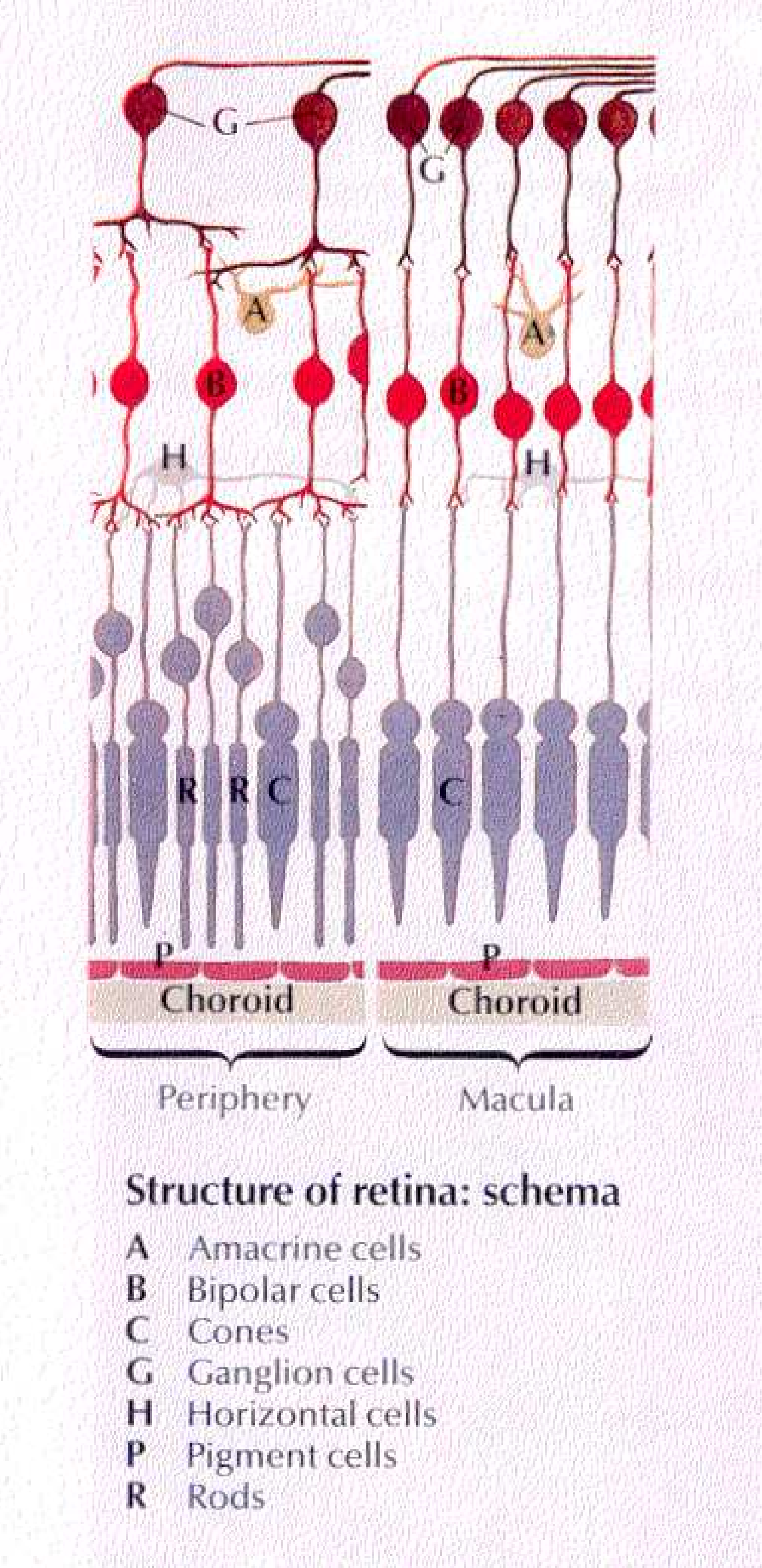


Figure 7.2: Retinal Cells according to [40]

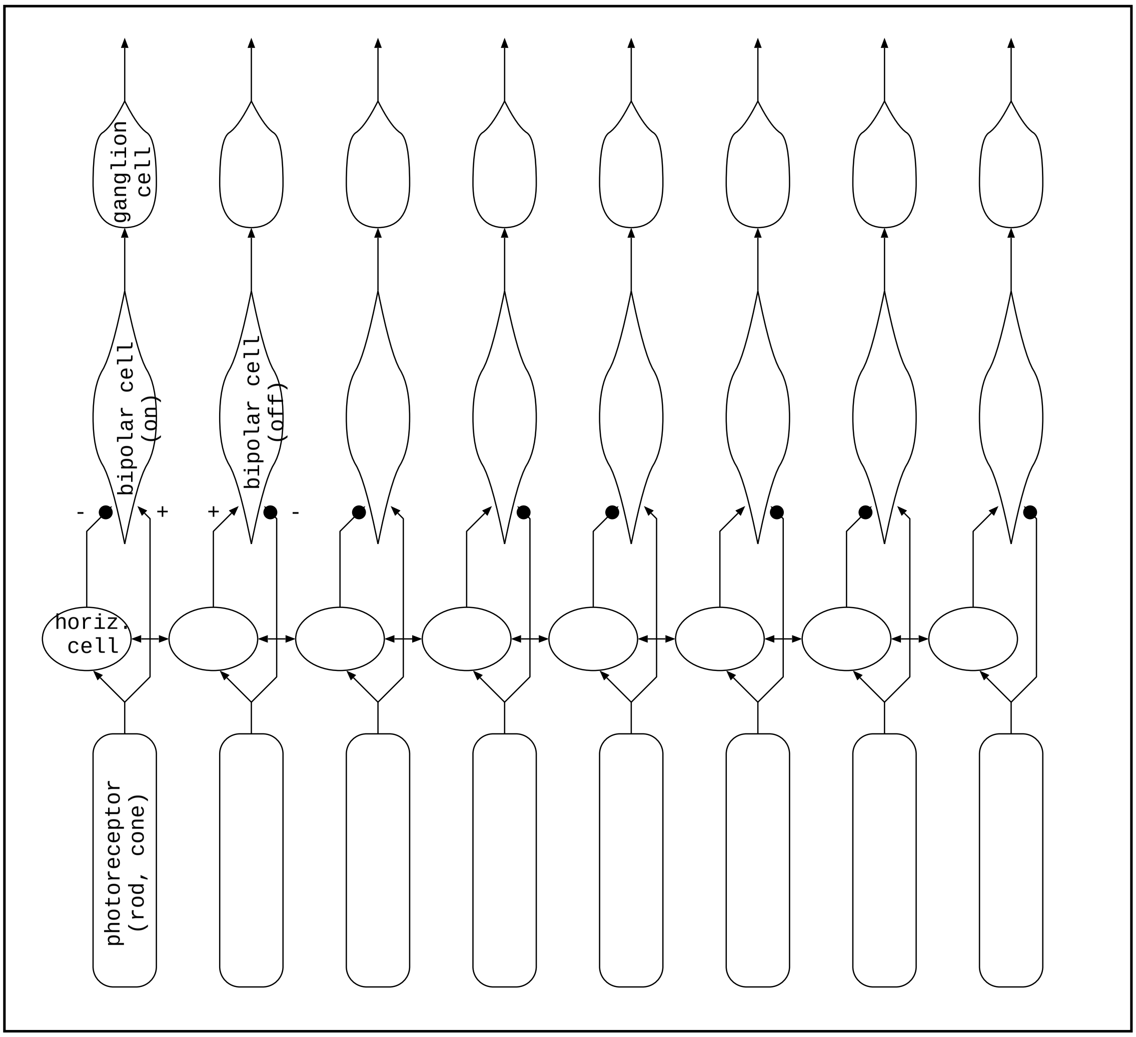


Figure 7.3: A schematic impression of retinal cells

synapses in figure 7.3) the collective average from it. They react to bright spots. Other Bipolar cells get inhibited by the local photo receptor and excited by the neighbourhood average. Those get excited by dark spots.

7.2 CMOS photo sensors

Neuromorphic imagers or simply retinomorphic circuits make use of the fact that CMOS photo sensors can be co-located on the same substrate as processing circuitry. Thus, they are perfectly suited to mimic those local computations of the retina.

Photo active structures in semiconductors make use of the fact that photons lead to impact ionizations in the material. Thus, a electron is set free from an atom in the conductor and a electron-hole pair is created: a negative and a positive carrier charge. Normally those two charges recombine immediately again, unless there is an electric field within the material separating them for good, in which case there will be a current of these charges flowing along that electric field.

Some definitions:

Intensity radiation energy per time unit per area

Contrast ratio of highest and lowest intensity in a picture

Some criterions applicable to photo cells are:

- gain
- speed
- signal noise (temporal)
- mismatch noise (spatial)
- linear, logarithmic etc.
- output resistance
- fill factor

7.2.1 Photo diodes

There is an electric field in the depletion region of PN junction that can be used to separate spontaneous electron hole pairs caused by photon impact (figure 7.4, layout in figure 7.5). A steady current starts to flow across the junction. The structure can be approximated as a current source that is linear with light intensity. Photo diodes are fast, low gain and low noise photo active structures.

7.2.2 Photo transistors

In the layout in figure figure 7.6 a PNP photo diode is depicted. The photo current through the lower junction acts as a current flowing out of the collector an is thus amplified through the transistor in the emitter current. Photo transistors are slower than photo diodes but have bigger gain. On the other hand the signal is more noisy too.

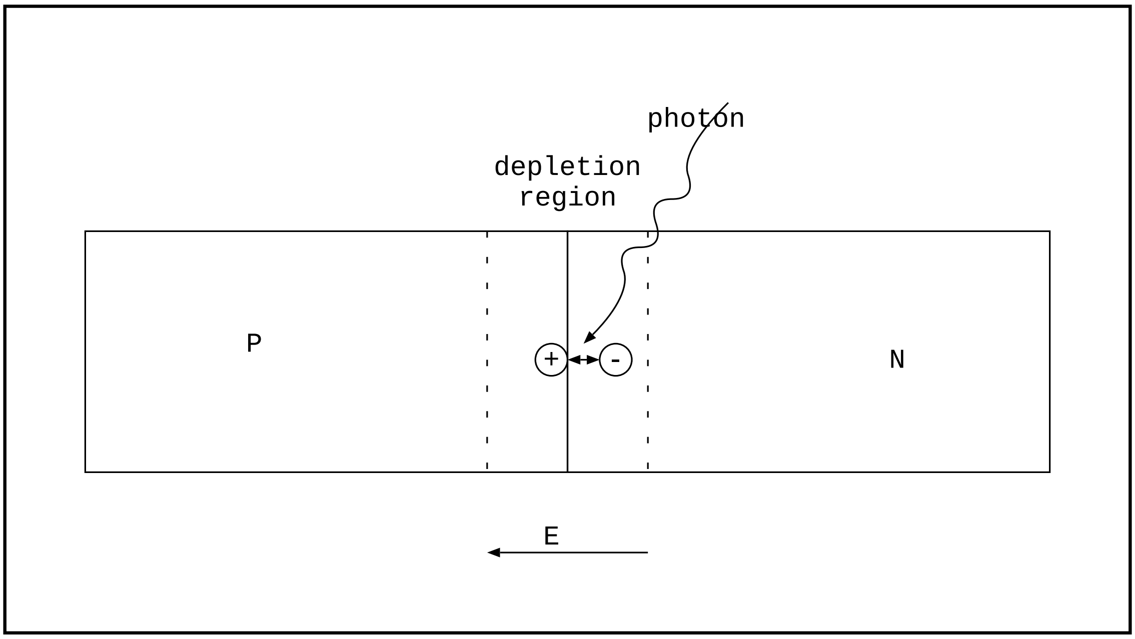


Figure 7.4: Separating light impact electron-hole pairs in the depletion region of a PN junction (photo diode)

7.2.3 Photo gates

Transistor gates in CMOS technology can also be used to make photo active devices (figure 7.7). Above a substrate they can create a depletion region that also separates light generated electron-hole pairs. They are for instance used in charge coupled device (CCD) cameras.

7.3 Photo Current Amplification

7.3.1 Linear by Early effect

Usually photo currents are very small. Thus, it is often desirable to amplify them already in the pixel, to allow more reliable read out or (as in neuromorphic circuits) to use them for further computations.

One way of achieving high linear voltage gain is shown in figure 7.8. The Early effect supplies the gain and therefore it is not easily controllable. Also, the output resistance is too high to drive a read out line, so an extra driver stage (e.g. follower) needs to be added to the pixel. The high gain allows the cell to be very sensitive within a very small range of intensity. But outside that range the cell simply saturates. The range can be set by the bias voltage.

7.3.2 Logarithmic by gate to source voltage

Logarithmic amplification (as provided by the two circuits in figure 7.9) is highly desirable because it allows the pixel to operate on a big range of intensities with constant gain proportional to contrast. The output voltage can be found analytically by solving the normal subthreshold transistor equation for the source voltage (for the NFET) or the gate voltage (for the PFET).

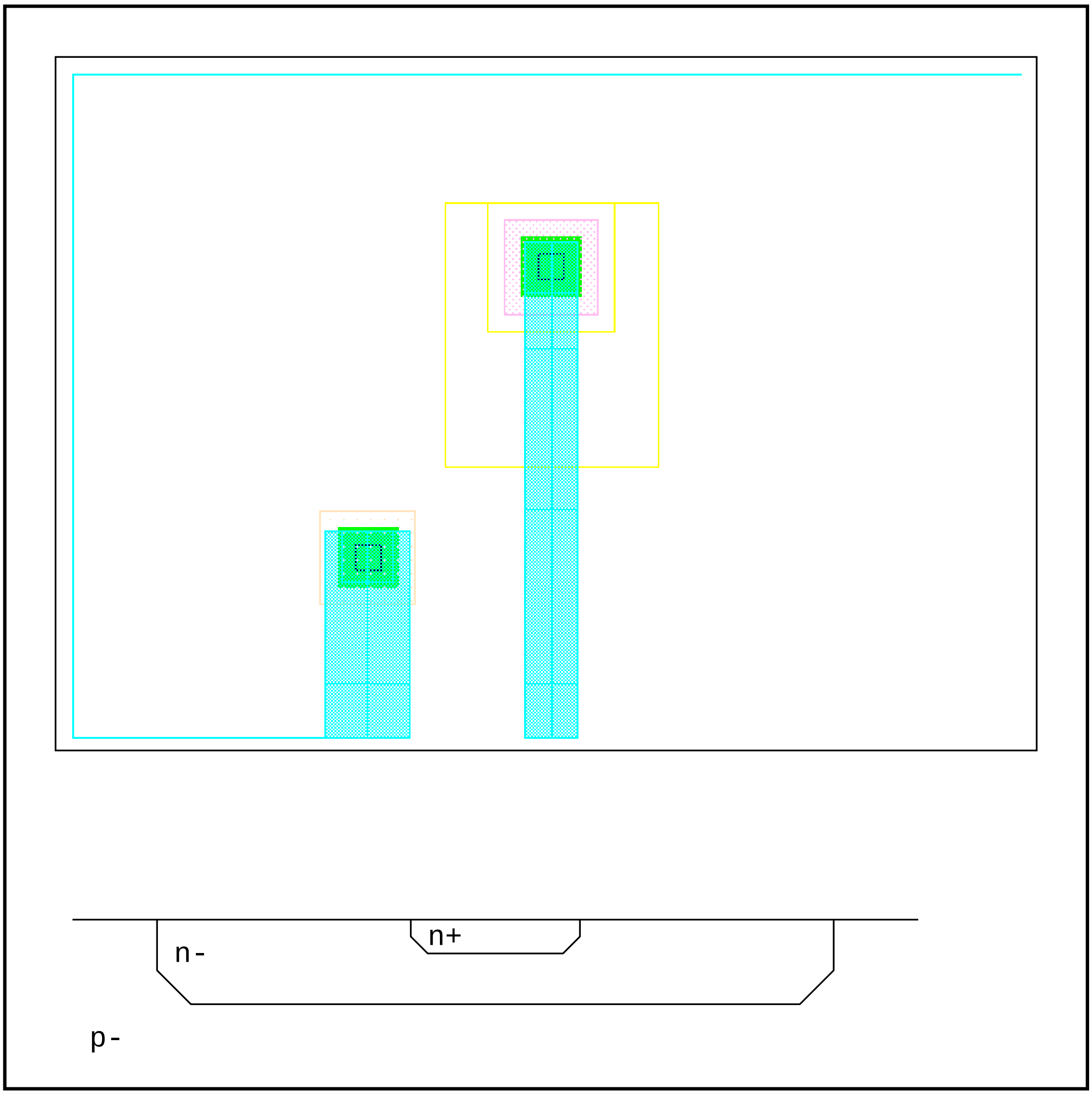


Figure 7.5: A possible layout for a photo diode

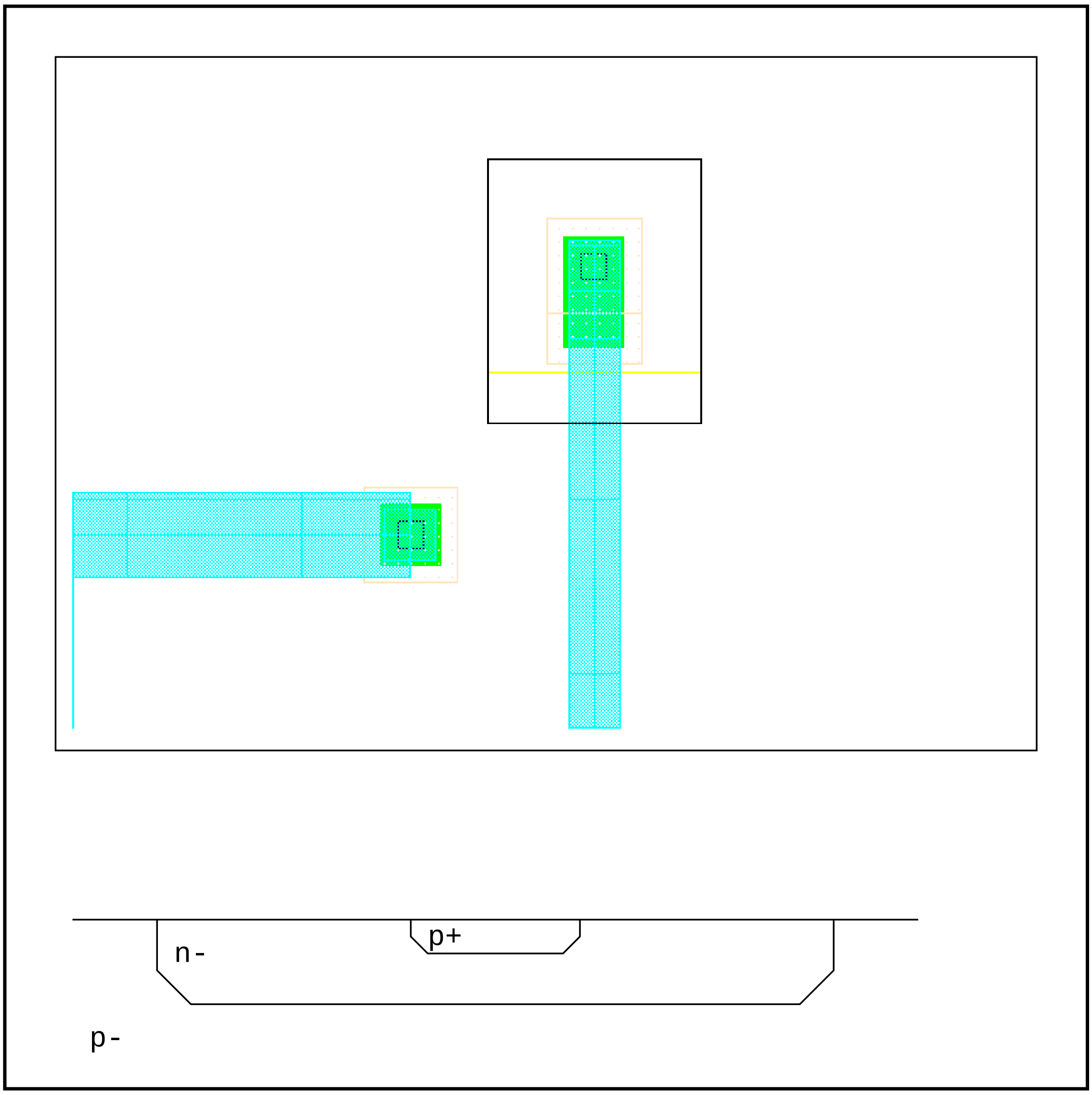


Figure 7.6: PNP photo transistor layout

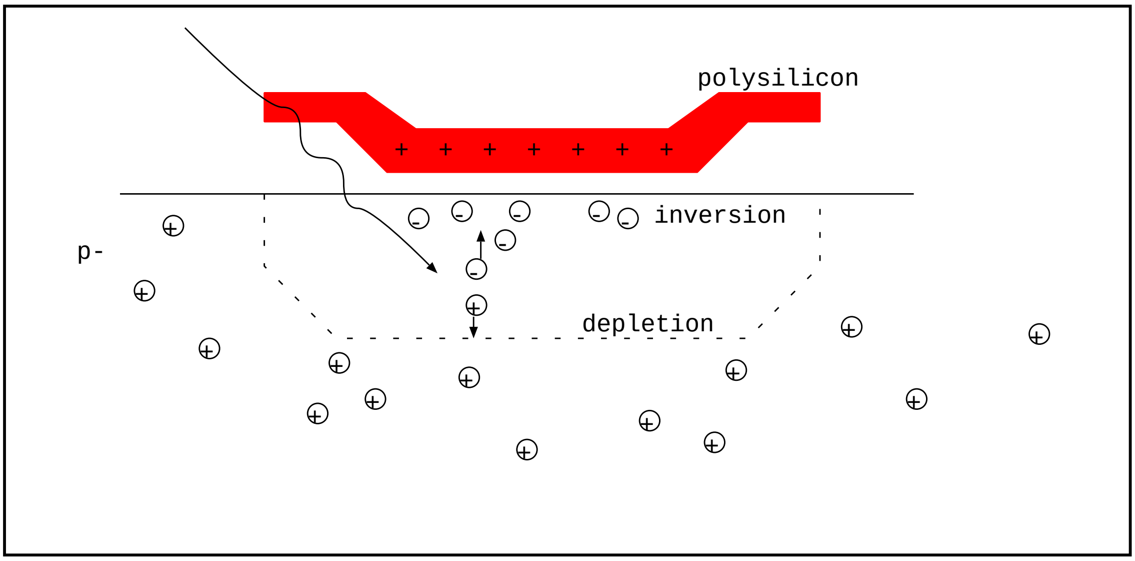


Figure 7.7: Charge separation in a photo gate

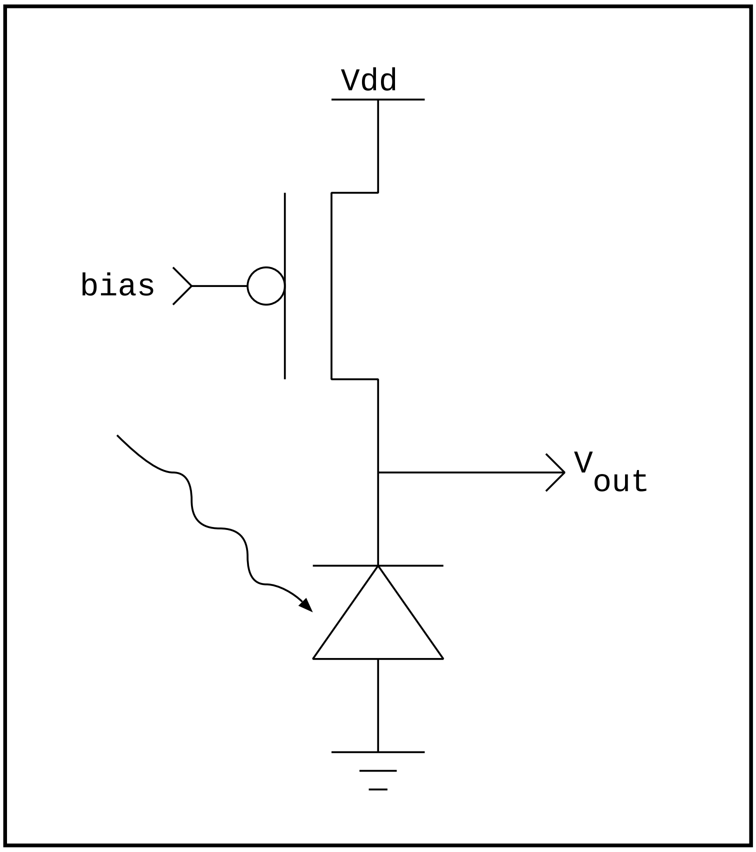


Figure 7.8: Linear amplification due to drain resistance/Early effect

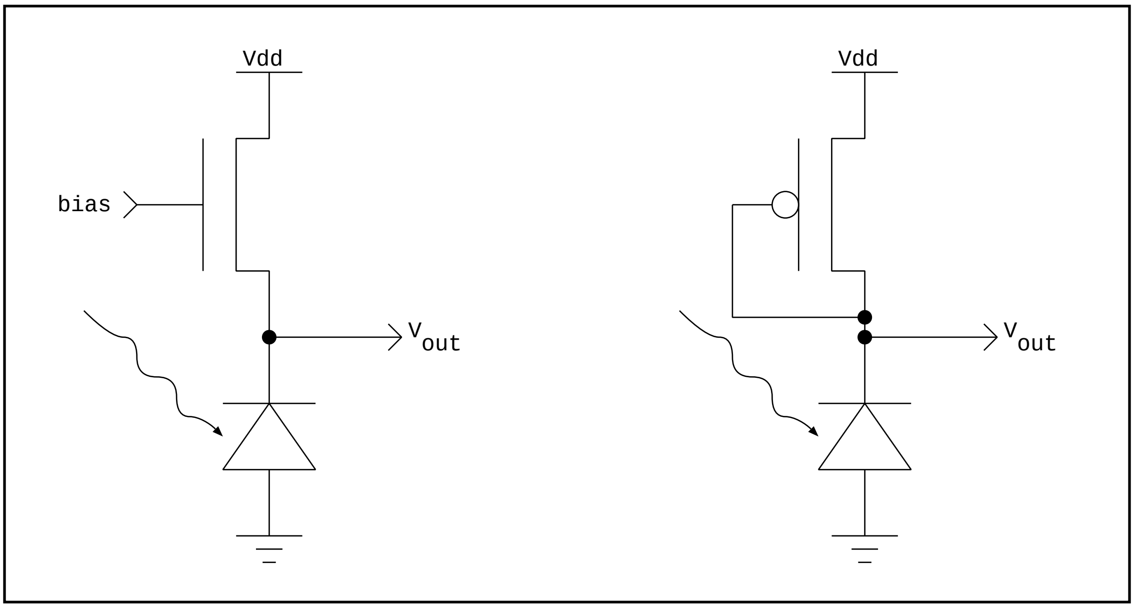


Figure 7.9: Two ways of achieving logarithmic amplification by the 'exponential source conductance' of an NFET (left) and by the 'exponential conductance' of a diode connected PFET (right).

Still these pixels have too high output resistance to drive a read out line.

7.3.3 Common source amplification

All of the above photo cells would be suited for read out by an address event or another integrating strategy. To read out a photo cell by addressing or scanning, however, it must be able to drive a common read out line, i.e. a low output resistance is needed. E.g. a most simple two transistor inverting amplification stage can achieve that (A common source amplifier, figure 7.10). This solution is more space preserving than for example a full blown follower (which is also possible to use).

To keep the amplification limited, negative feedback can be applied e.g. in the manner depicted on the top of figure 7.11. Capacitive division can be used to implement different strength of negative feedback (bottom of figure 7.11). If one assumes infinite gain in the inverter, then the system gain is given by the ratio of the capacitances and the parameters of the transistor only: One can assume that whatever the change in photo current, the feedback transistor will compensate it 100%. Thus, one can assume that the current through the transistor, as given by the gate to source voltage, is equal to the photo current. Hence, the following formula can be derived:

$$V_{out} = \frac{C_{tot}}{C_{fb}} nU_T \log I_{photo} - nU_T \log I_S + V_{T0} + nV_S \tag{7.1}$$

Also, see figure 7.15 later in this text for a most clever extension of that circuit.

7.3.4 Source follower

The 'active pixel' (so named by the producer Micron), employs a source follower for read out amplification (figure 7.12). The charge accumulates under the photo gate. The control signal TX

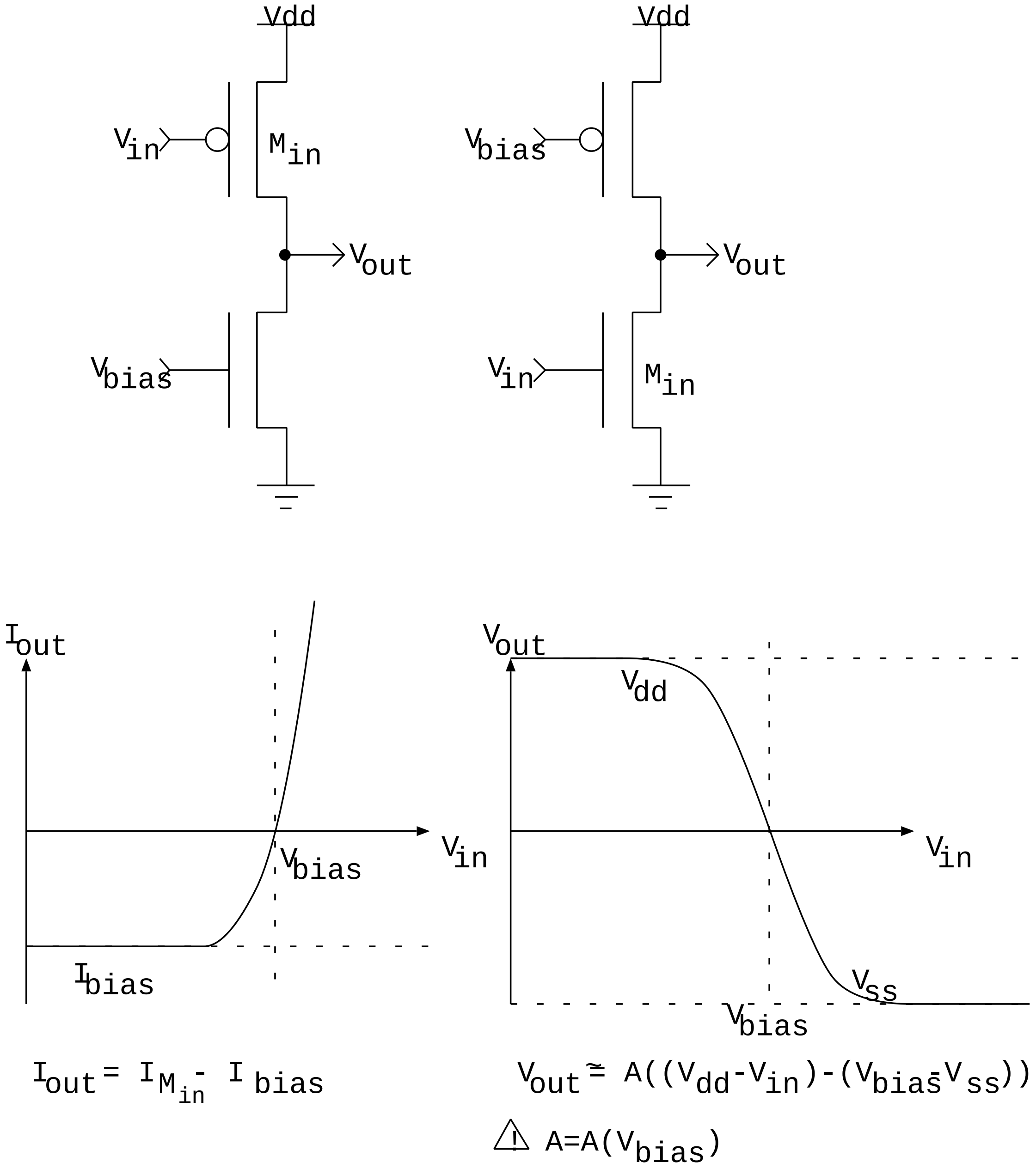


Figure 7.10: A variant of an inverter with a bias voltage that allows to set the gain and the driving current

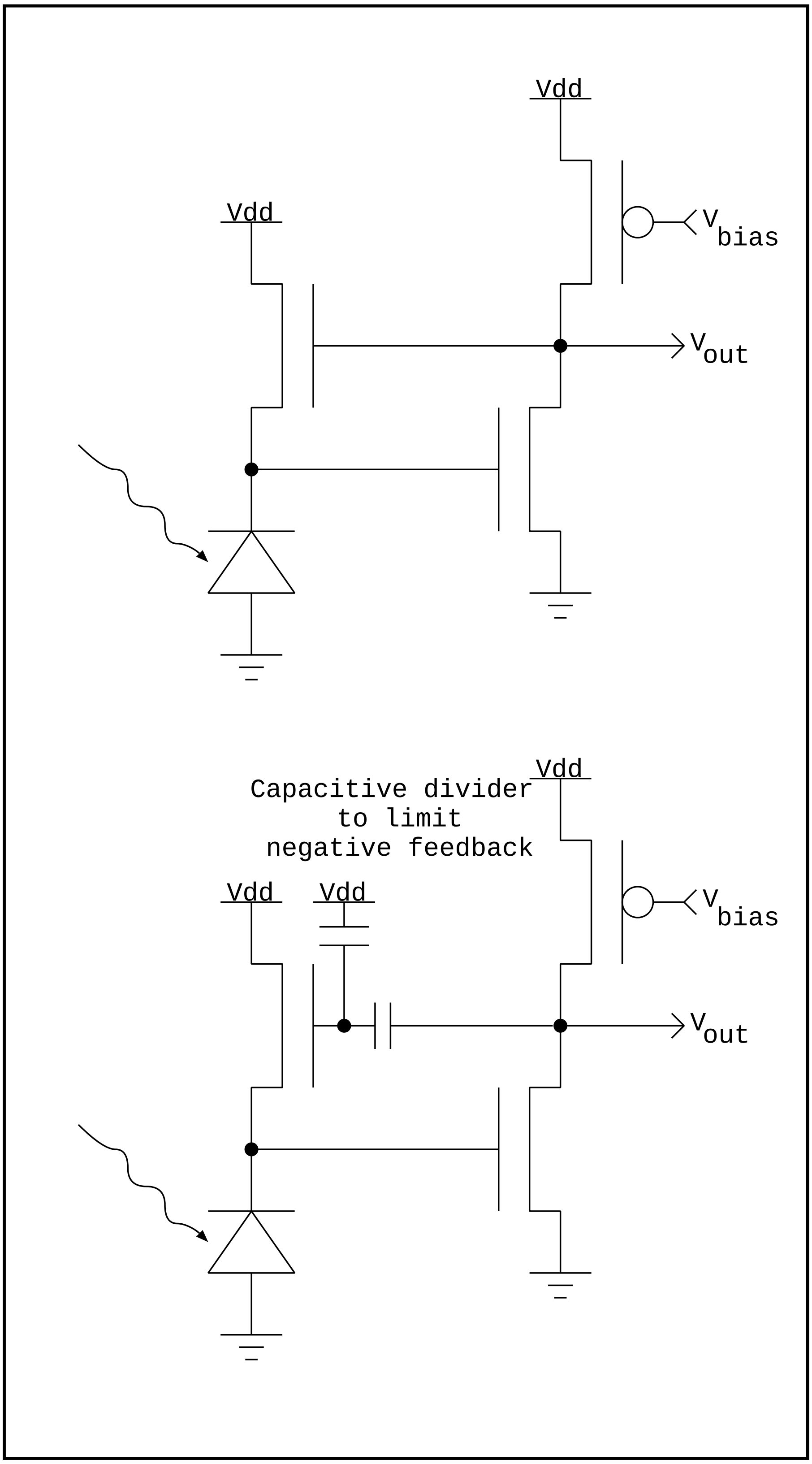


Figure 7.11: Amplified negative feedback almost nullifies the gain (top), actually if one assumes infinite gain in the inverter, one can use capacitive voltage division to control the system gain (bottom).

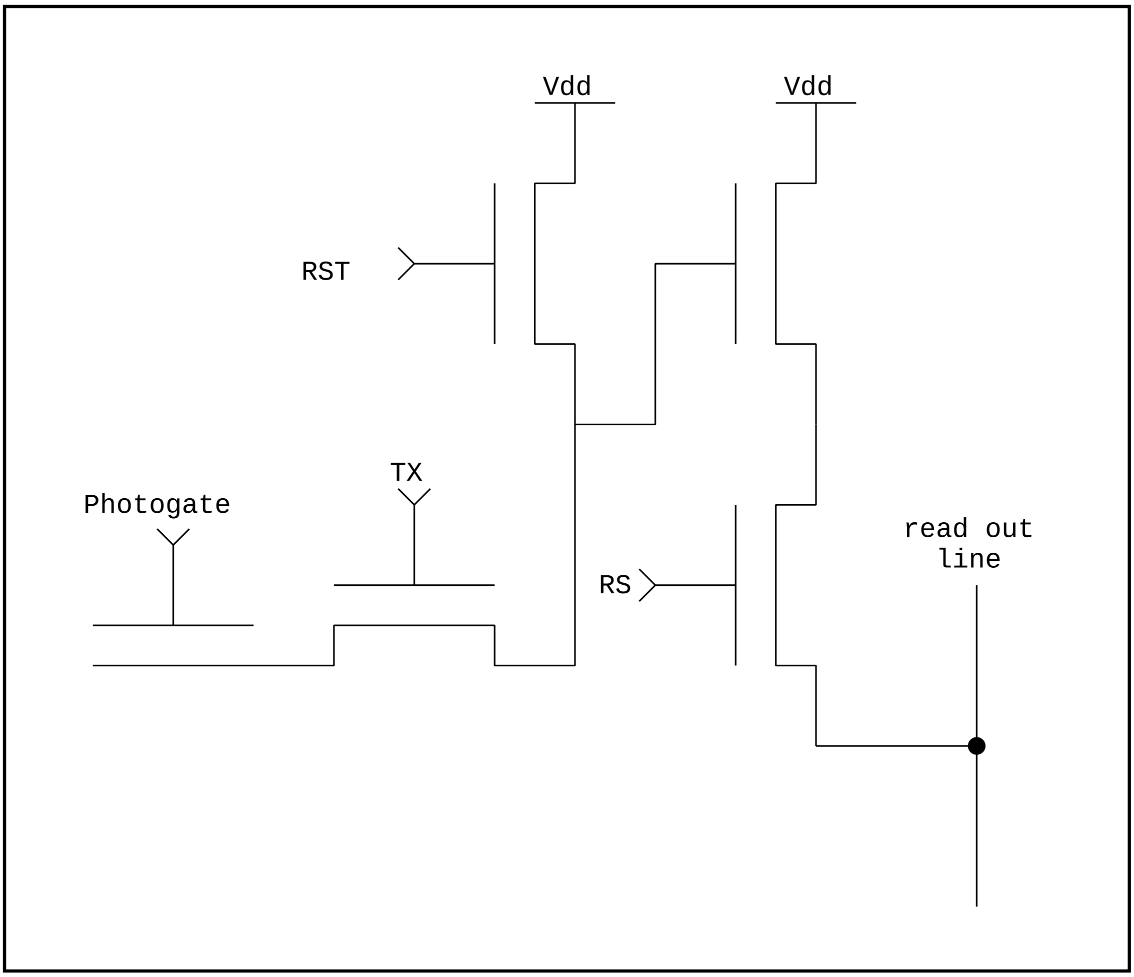


Figure 7.12: The ‘active pixel’ is so named by the patent holder ‘Micron’. It is ‘active’ in so far as an amplification is performed in the pixel

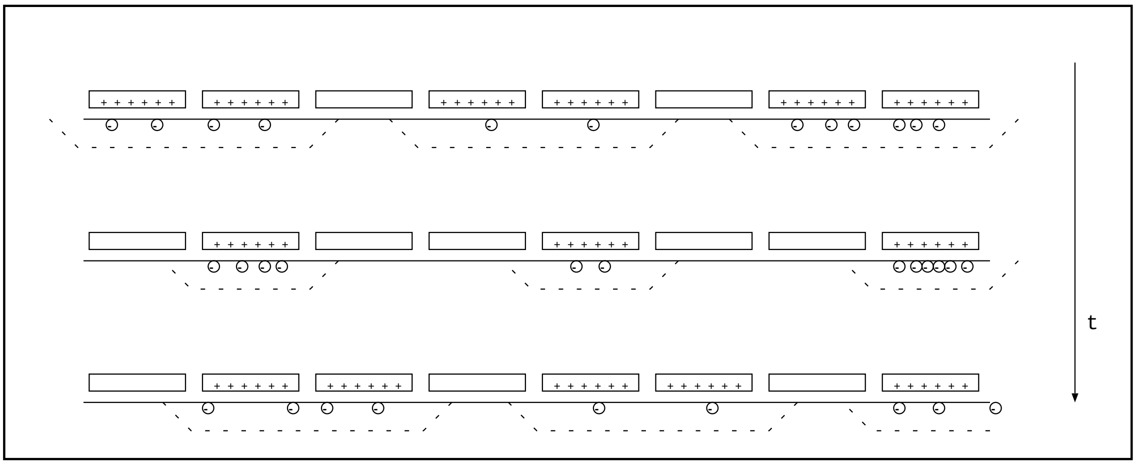


Figure 7.13: Read out in a charge coupled device: charges are accumulated under photo gates and then shifted in the array by appropriate switching among overlapping groups of gates

connects the photo gate to the gate of the transistor driving the read out line. A current is drawn from outside on the read out line and as the read out transistor's source is connected to it, the voltage on that line will be driven to the transistors gate voltage minus the threshold voltage.

7.4 Read Out Strategies

7.4.1 Addressing and scanning

Addressing is the most classic read out strategy used for many two dimensional structures, like for example memories. Individual cells are addressed from two sides of the two dimensional array by demultiplexers or scanners. The selected cell is granted access to a common read out line. The cells need to be able to drive that line.

7.4.2 Charge coupled devices (CCD)

CCD is the dominating technique in todays imager market allowing for the highest pixel density. The key lies in the read out technique of the pixels (see figure 7.13). All pixels are connected via silicon of the same doping. (Gate-)electrodes on top of that substrate can create electrically isolated regions that can be charged individually by the photo gate principle. By letting a 'wave' travel through the electrodes, these charge pockets can be shifted from one electrode to the next to the border of the array where it can be read out. Very dense structures can be achieved like this.

7.4.3 Address event representation

An asynchronous integrating read out strategy is AER (also see chapter 6). How it could be used with a photo cell is depicted in figure 7.14. Much like in an integrate-and-fire neuron, a node in the photo cell gets charged by a photo current until it reaches a threshold. Then a signal is sent out along both x and y axis to the periphery where an address event is generated. The light intensity is so coded in the frequency of AEs. The example in the figure shows the principle of an arbitered

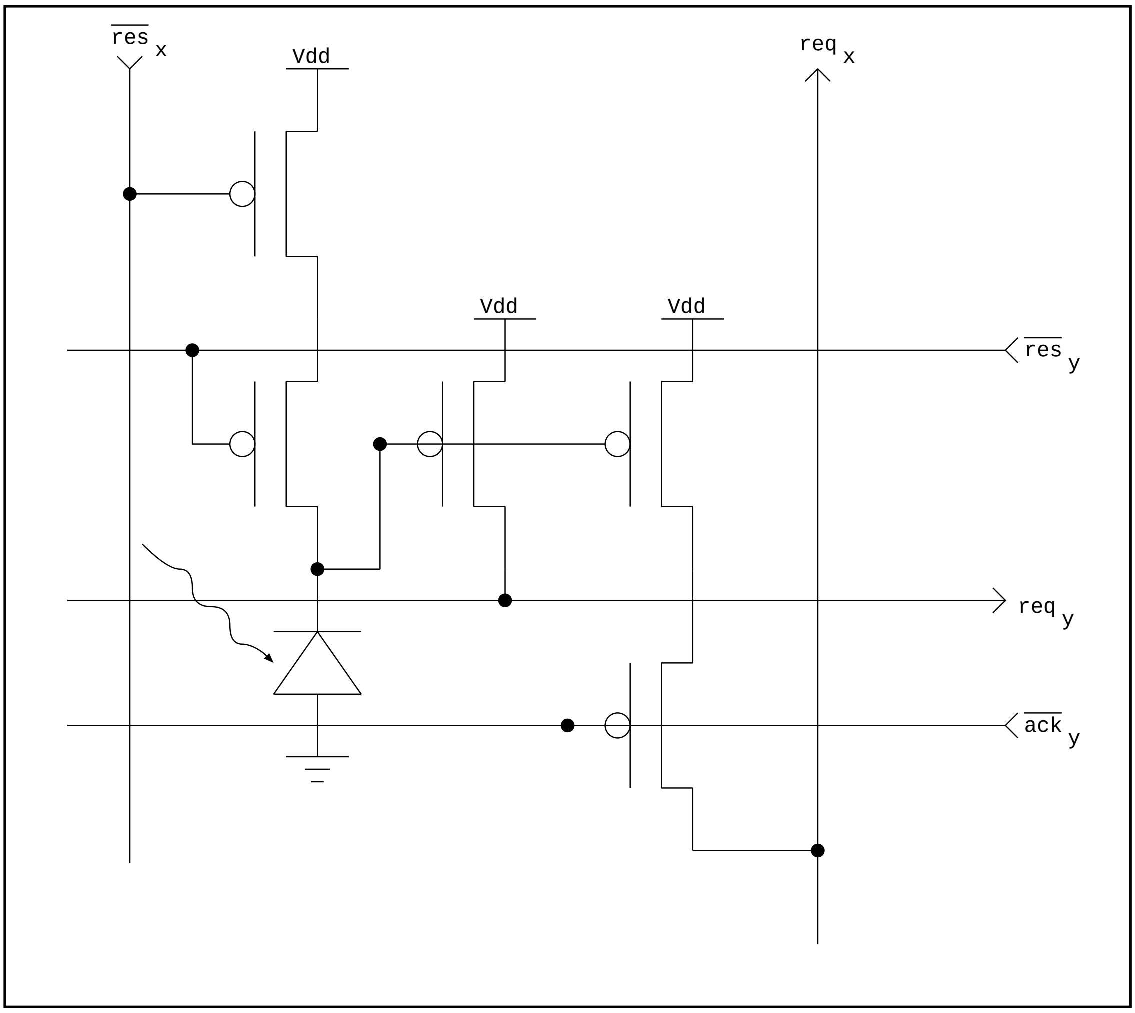

$\overline{\text{res}}_x$
Vdd
req_x
Vdd
Vdd
$\overline{\text{res}}_y$
req_y
$\overline{\text{ack}}_y$

Figure 7.14: A (too) simple photo pixel for AER read out

AER, where the pixel has to apply for an AE to be generated, first with the y-arbiter on the right of the two dimensional array, and then (if acknowledged) with the x-arbiter on the top. A real implementation will need to address some non ideal properties of that circuit, though, and will look somewhat more complex.

7.5 A Silicon retina

One rather famous adaptive photo cell, as introduced in its original form by Tobi Delbrück [9] (figure 7.15), uses capacitive negative feedback in a most clever manner, achieving little negative feedback and therefore big gain for small signals, and much negative feedback and therefore little gain for large signals. In addition, this solution has high pass properties, such that it decreases DC offsets. It thus, shows some of the adaptive properties that are present in real retinal photo receptors, as described earlier in this text. The non-linear element in that schematics can for example be implemented compactly like in figure 7.16

An analog circuit (by Misha Mahowald [33]) that also implements the horizontal cells' functionality of contrast enhancement is shown in figure 7.17. The direct photo cell input is compared to a local average through a resistive net. The resistive network is layed out to connect a hexagonal grid. In this implementation there went a bit of an effort in implementing the resistances with transistors as quasi linear circuits. The photo cells were the adaptive photo cells.

An alternative solution [7] in 'current mode' that has no need of liner resistances but replaces them with a current mode diffuser network, and that is therefore better suited for direct CMOS implementation is in figure 7.18. This implementation does not use the adaptive pixel but the current output (EPSC) of the above photo and diffuser cell goes to an adaptive integrate and fire neuron, which performs a similar function.

7.6 Further Image Processing

Neuroscientists have long ago discovered neurons beyond the retina (LGN/visual cortex) that are responsive to quite distinct properties of visual stimuli. Those properties become more and more complex, from simple retinal on/off cells to center surround in the LGN, 'simple' and 'complex' cells in V1 that are responsive to bars of a particular orientation and sometimes to direction of motion, to all kind of motion sensitive cells in MT, and finally cells that are responsive for example to faces. But in contrast to the retina, the architecture giving rise to these particular responses of cells is not clearly known and cause for debates. Most models are quite content in reproducing the properties without a strong claim of being faithful to the underlying neuroanatomy.

7.6.1 Motion

Token Based

These are algorithms that rely on a previous image processing stage that is able to identify and localize a particular token (feature/object) in the visual field. The classic example is the Reichardt detector (figure 7.19). A later example is found in [31]. As may be intuitively clear from the way the Reichardt detector is depicted, these algorithms can be implemented relatively easily with plausible neural building blocks. Delays can be achieved in dendrites and axons. Coincidence detection can be performed by integrate and fire neurons with a rather high leakage. So it is well possible, although not proven, that the nervous system performs a kind of token based motion detection.

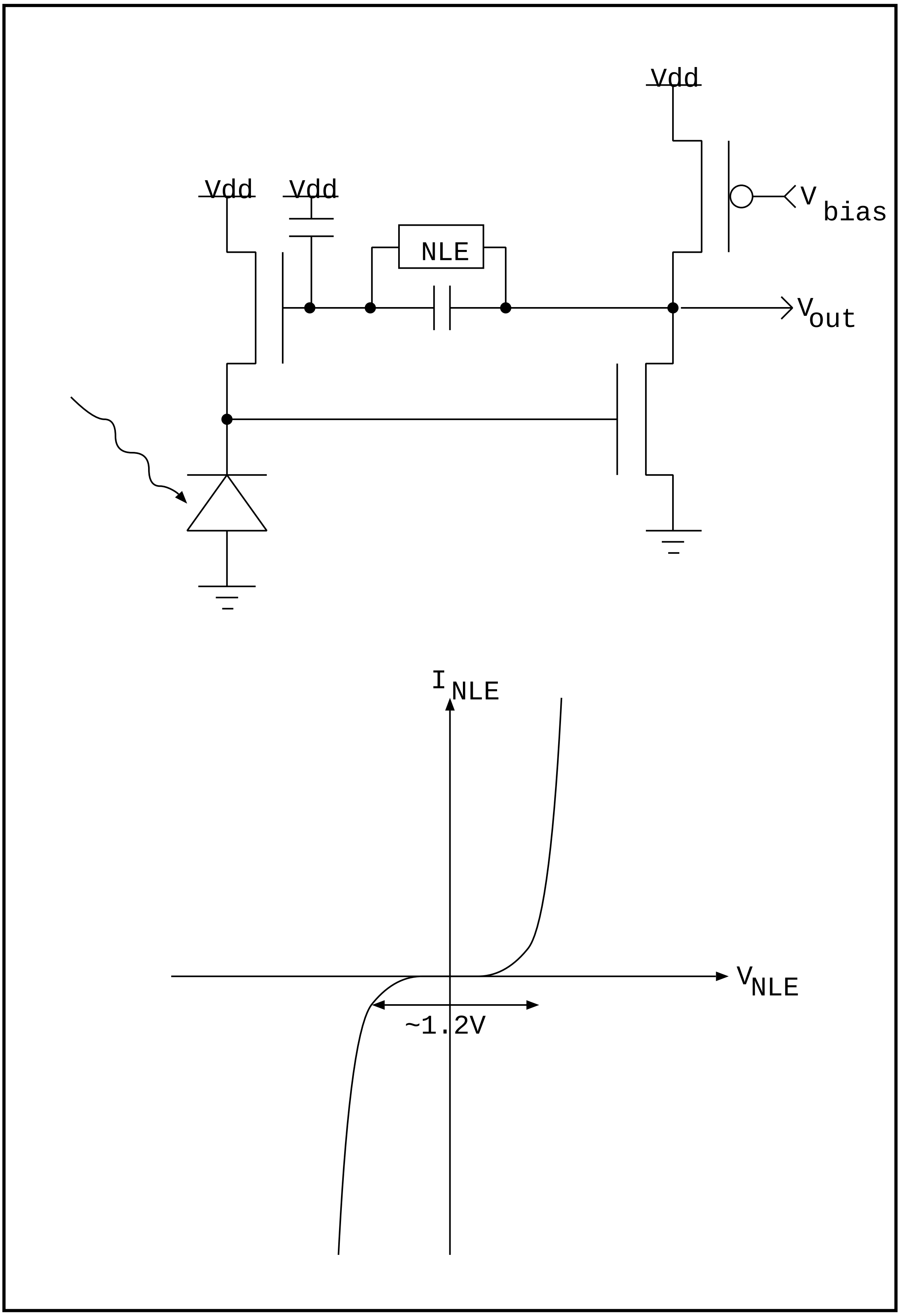


Figure 7.15: Adaptive photo cell as proposed in [9]

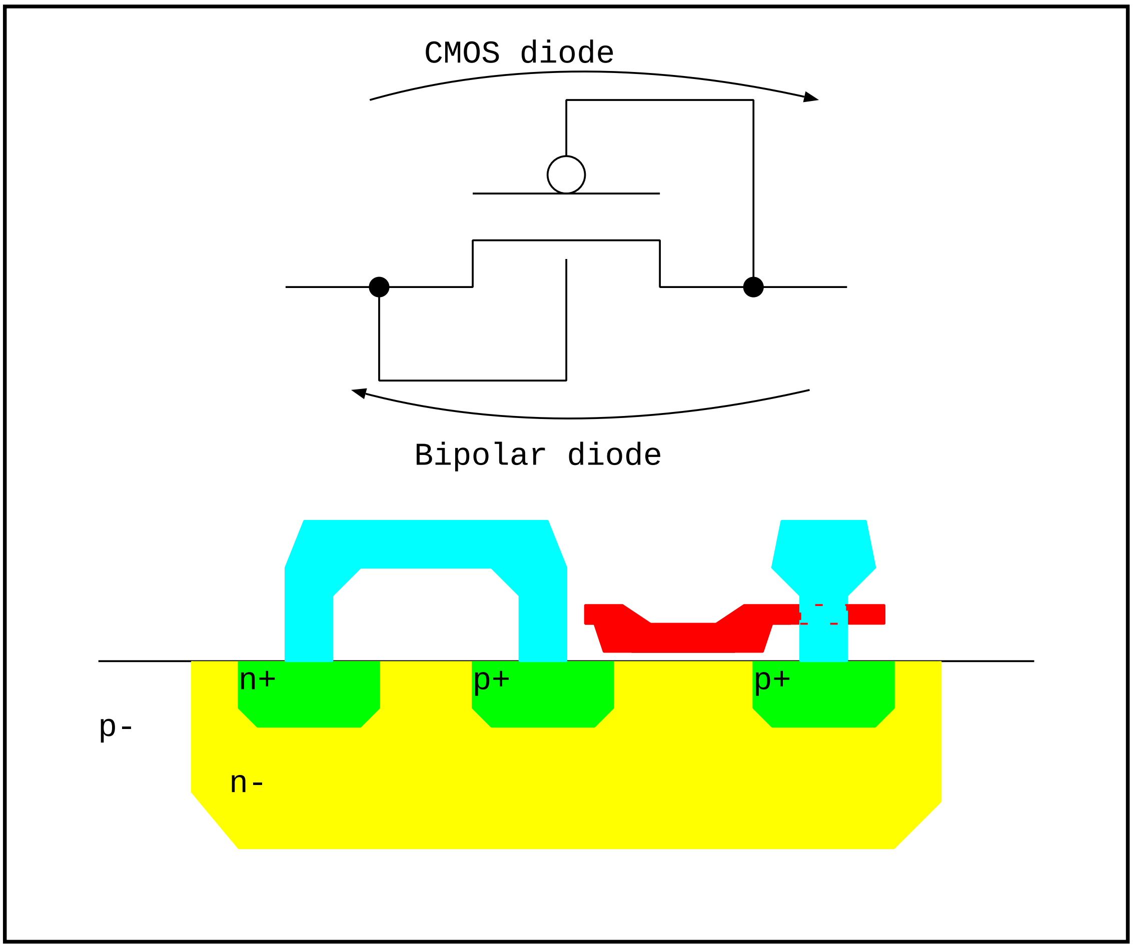


Figure 7.16: A non linear element, in effect consisting of two diodes in parallel, a CMOS diode connected transistor and a PN bipolar diode

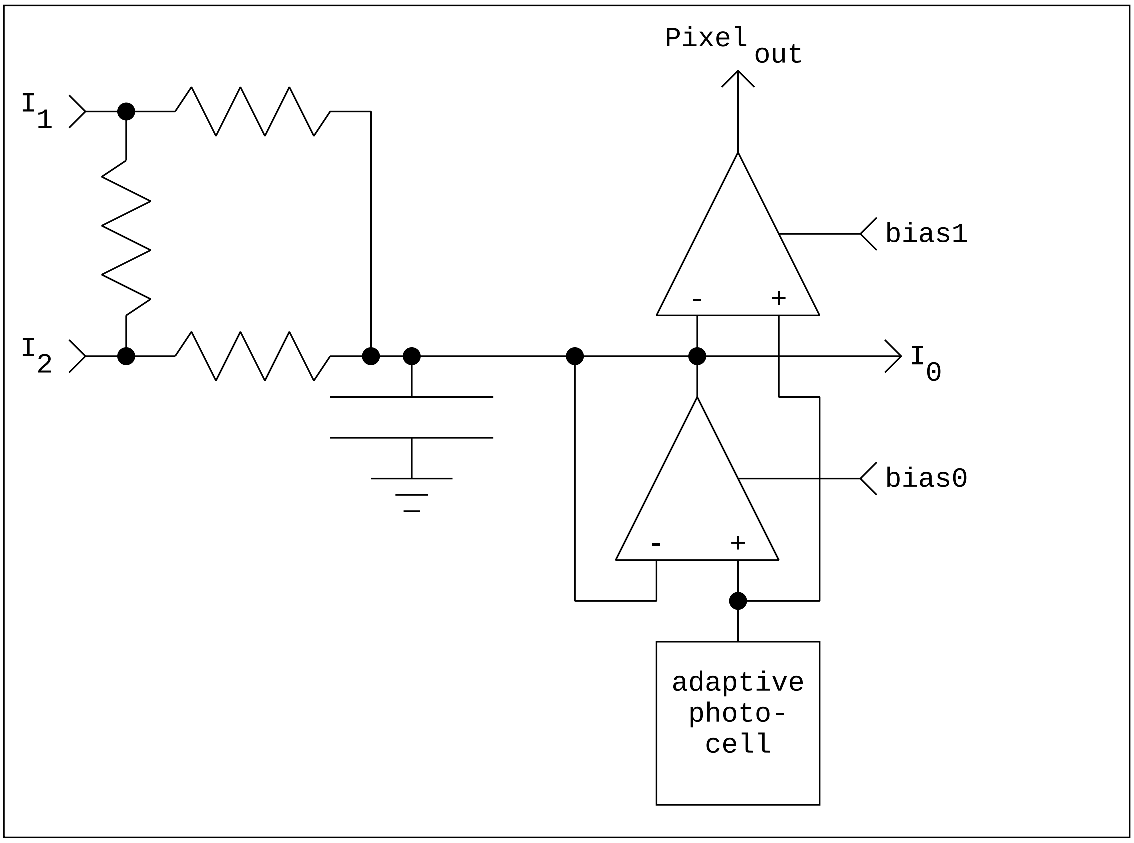


Figure 7.17: Silicon retina pixel according to [33]

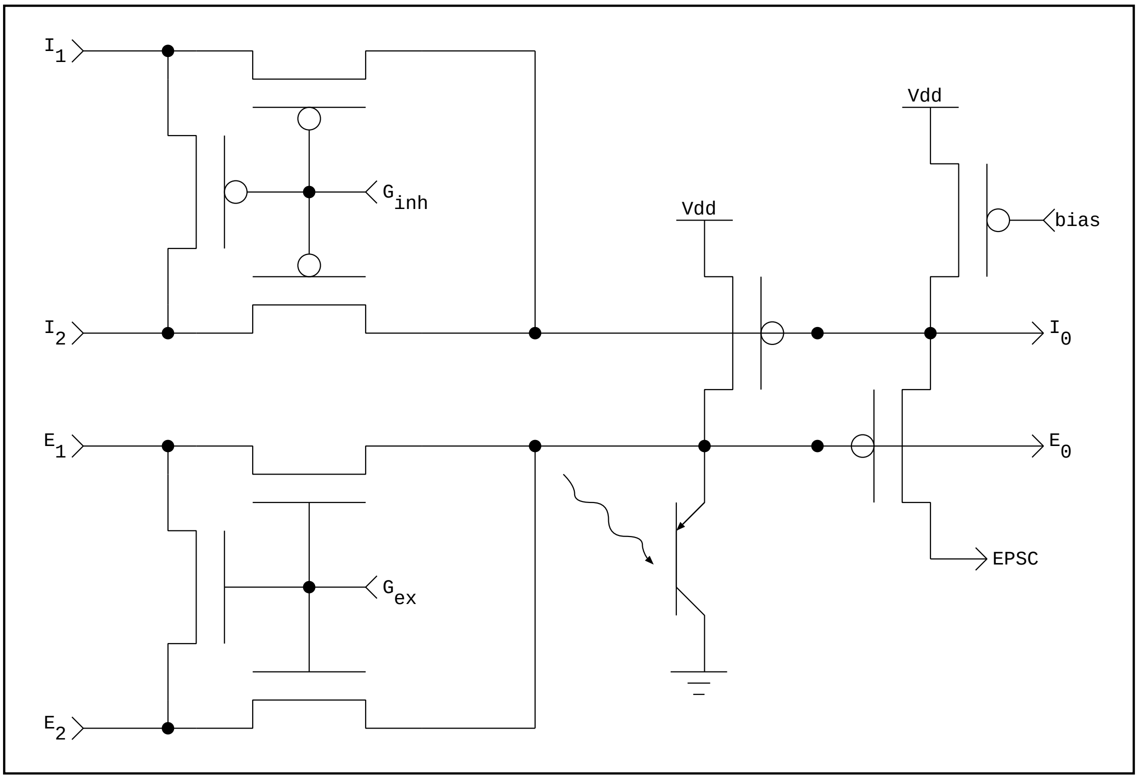


Figure 7.18: Silicon retina pixel according to [7]

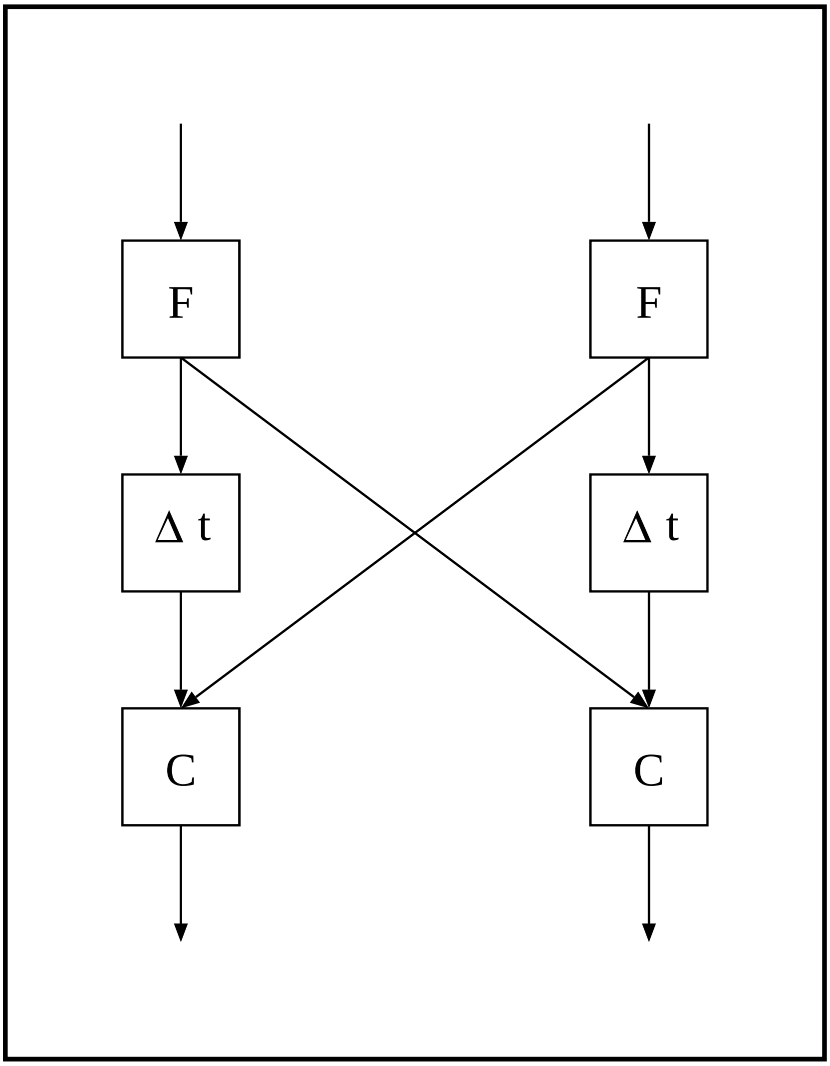


Figure 7.19: The Reichardt Detector. (F: feature (=token) detector / Δt: delay / C: coincidence detector)

Intensity Based

In contrast to token based motion detection algorithms, these work directly on the intensity profile of a moving surface. (A recent example: [52]) Assuming that the surface does not change its emission intensity the following equations describe the speed of the observed surface for a 1D surface:

$$v_x \frac{dy}{dx} = -\frac{dy}{dt} \tag{7.2}$$

for a 2D surface:

$$v_x \frac{dz}{dx} + v_y \frac{dz}{dy} = -\frac{dz}{dt} \tag{7.3}$$

In general, this equation does not have just one solution for $\vec{v} = (v_x, v_y)$ in one point, when the spatial derivatives ($\frac{dz}{dx}$ and $\frac{dz}{dy}$) and temporal derivative ($dzdt$) are given. The situation is better, if one can assume that one observes several points of a rigid object, i.e. points that move with the same speed and direction. But even so there are non trivial cases where a single solution cannot be found. An example thereof is the so called 'aperture problem'. It appears if within a limited field of view the observed moving surface intensity has spatial derivative equal to zero along one direction. Imagine a 45 degree (lower left to upper right) infinitely long edge moving from right to left across your visual field. You would observe the same, if the edge would move from bottom to the top at the same speed.

7.6.2 Feature maps

Feature maps are formed by convolving an image with a feature kernel. Convolution is a mathematical operator on two functions (f and g) that combines them to one function h. 'h is the convolution of f with g' means:

$$h(x) = \int_{-\infty}^{\infty} f(x-s)g(s)ds \tag{7.4}$$

Such a convolution can be performed by a particular neural network: The output neurons represent h(x). They are all connected with the same weight pattern g(s) to the input neurons f(x-s), relative to their position x. In other words if we denote the output neurons more traditionally with indices h_j and f_i, and a weight from f_i to h_j with $g_{i,j}$, then $\forall (i,j) \in \mathbf{N}^2, v \in \mathbf{Z} : g_{i,j} = g_{i+v,j+v}$

In image processing convolution is often used to bring out particular features in pictures. Two examples are illustrated in figures 7.21 to figure 7.27. Figure 7.21 is the original picture. Figure 7.24 shows the same picture with contrast enhancement, i.e. where there has been a local contrast, a transition from dark to bright or vice versa, there is now a bright and a dark bar marking that transition. Areas that where of uniform illumination are now all of a uniform gray. That is a useful preprocessing of a picture for edge extraction. This effect is achieved by the convolution with the function/kernel in figures 7.22(surface plot) and 7.23 (colour encoded contour plot), often referred to as 'difference of Gaussians' or 'mexican hat'.

Another feature is extracted in figure 7.27. There only edges of a particular orientation remain clear in the picture, namely edges with an orientation parallel to the diagonal from lower left to upper right of the picture. Edges that mark a transition from dark to bright from the upper left to the lower right are marked with a bright bar, and transitions from bright to dark, with a dark bar. The picture gets a relief like quality through this. The convolution kernel achieving this is shown in figures 7.25 and 7.26.

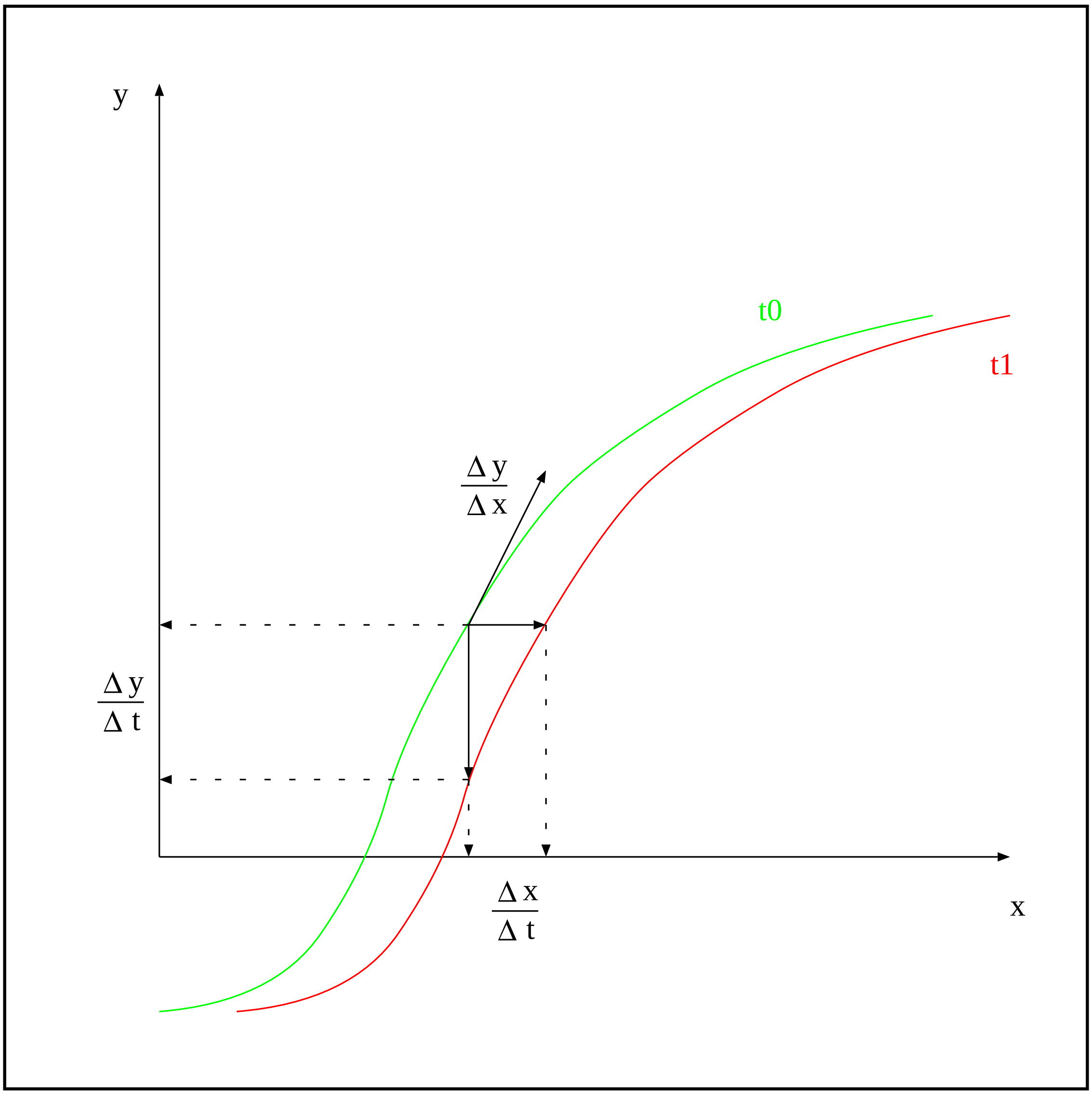


Figure 7.20: An illustration of intensity based motion estimation. The intensity profile of a surface at t_0 (green) and t_1 (red is shown). Assuming that this profile is constant on the object surface, only motion can be responsible for the change in this interval. Dividing the change in time, by the change in space, one obtains the speed of the surface, i.e. in the 1-dimensional case. The 2-dimensional case has unfortunately multiple solutions, when looking at just one pixel.

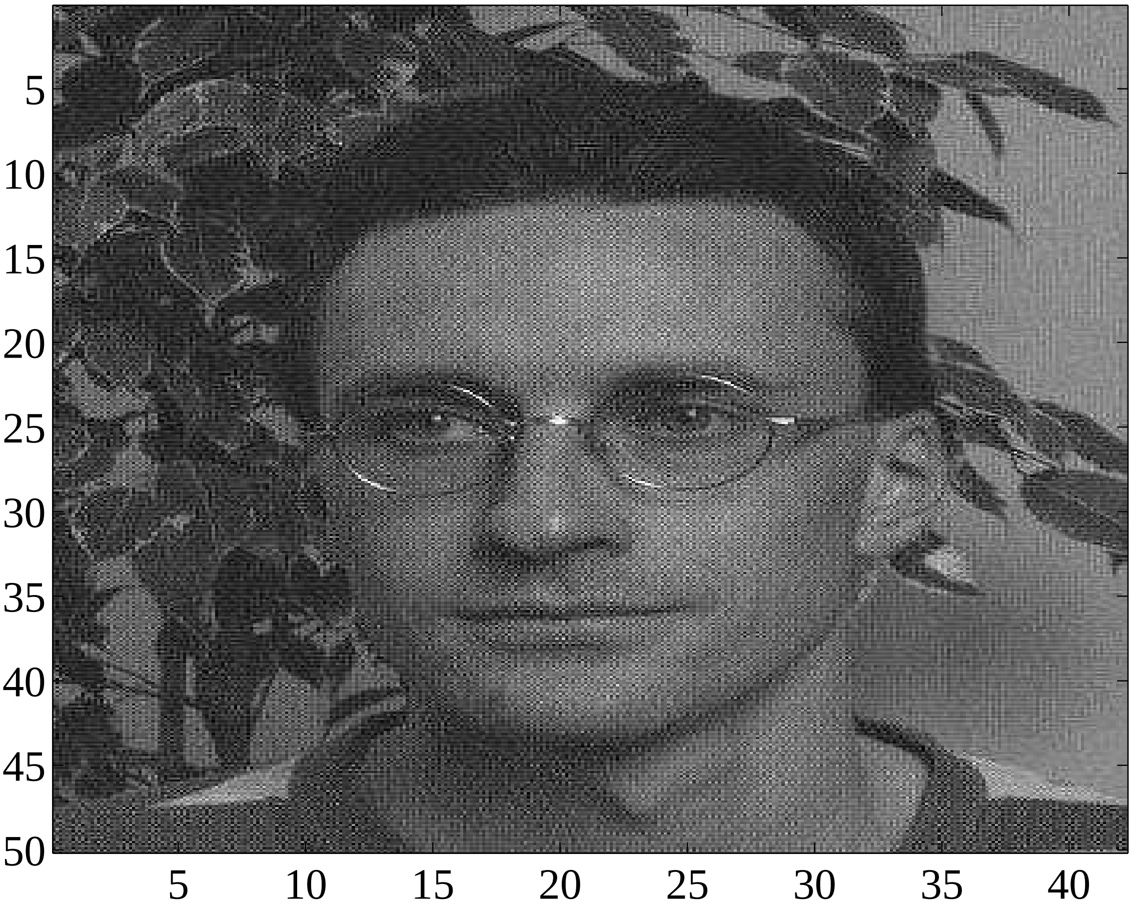


Figure 7.21: A photograph of an every day natural scene, that will be used to illustrate some image processing

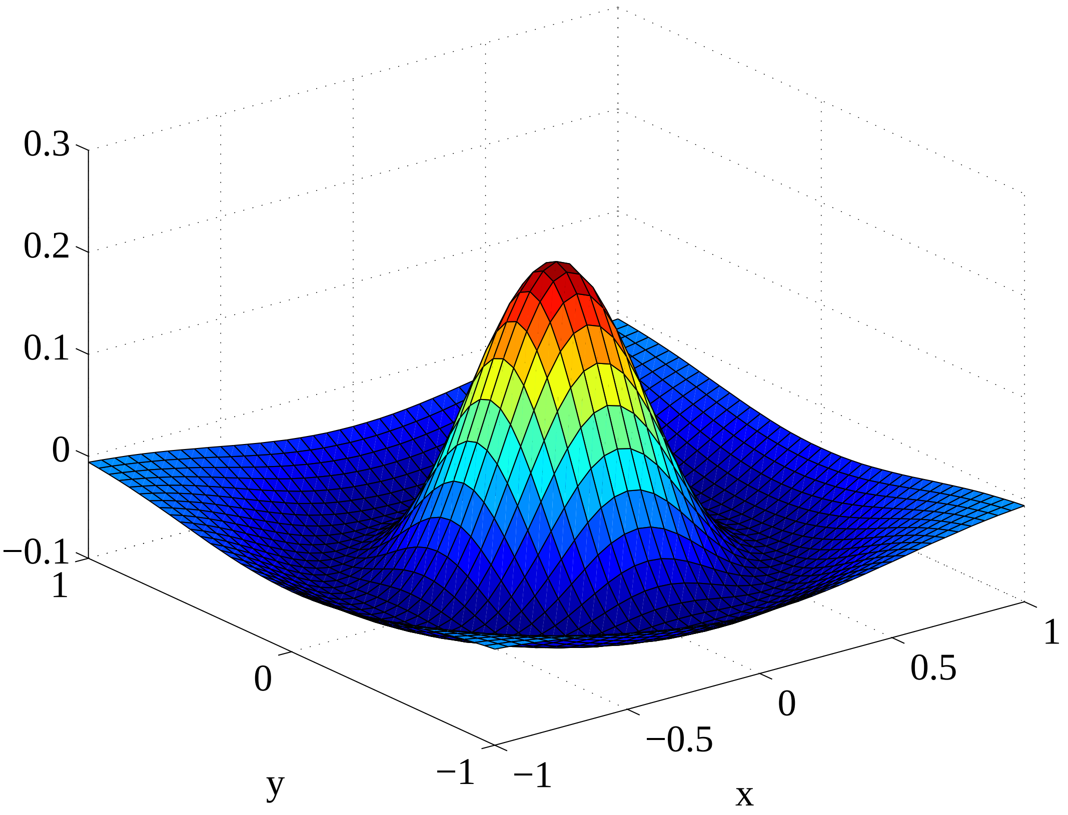


Figure 7.22: 2 dimensional surface plot of a 'difference of Gaussians'- or 'Mexican hat' function.

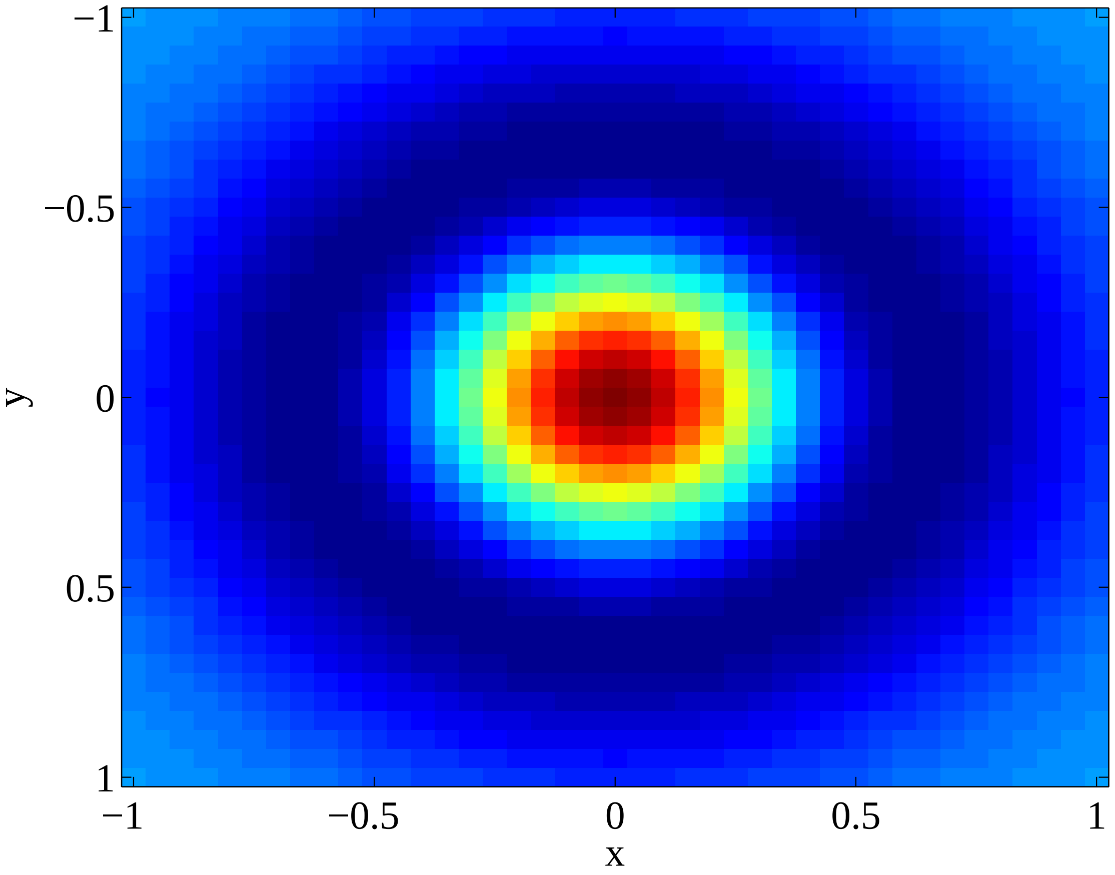


Figure 7.23: 2 dimensional colour code plot of a ‘difference of Gaussians’- or ‘Mexican hat’ function.

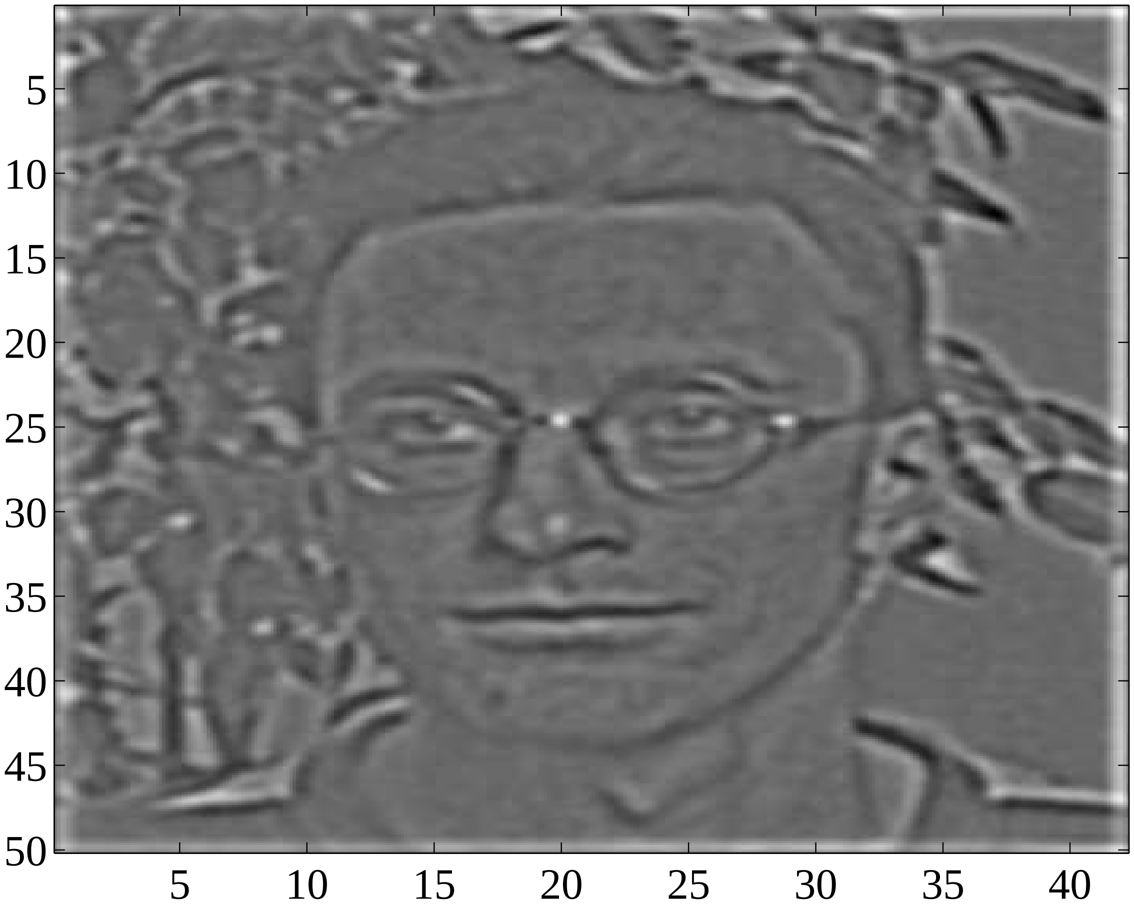


Figure 7.24: The image of figure 7.21 that has been convolved with a 'difference of Gaussians' convolution kernel: Local contrast, mostly edges, are enhanced. Uniform surfaces appear gray.

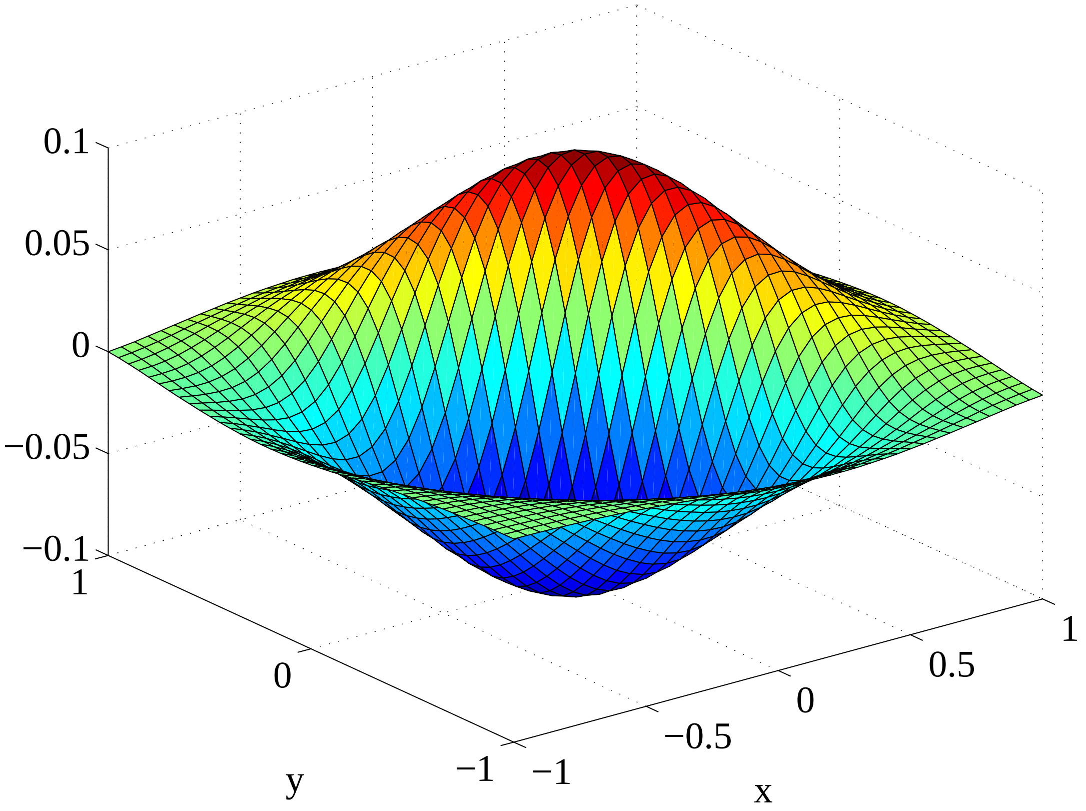


Figure 7.25: A function suited as a convolution kernel for extracting edges at an angle of 45 degrees in surface plot

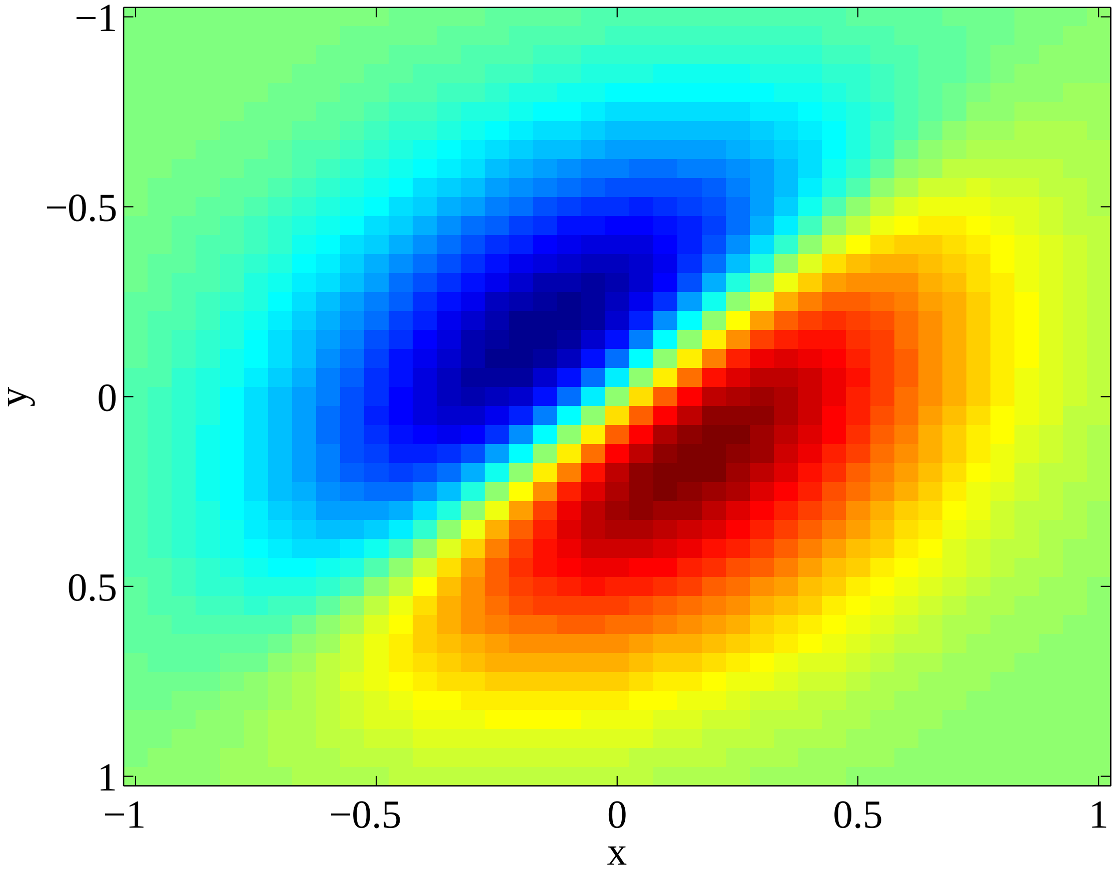


Figure 7.26: A function suited as a convolution kernel for extracting edges at an angle of 45 degrees in colour code plot

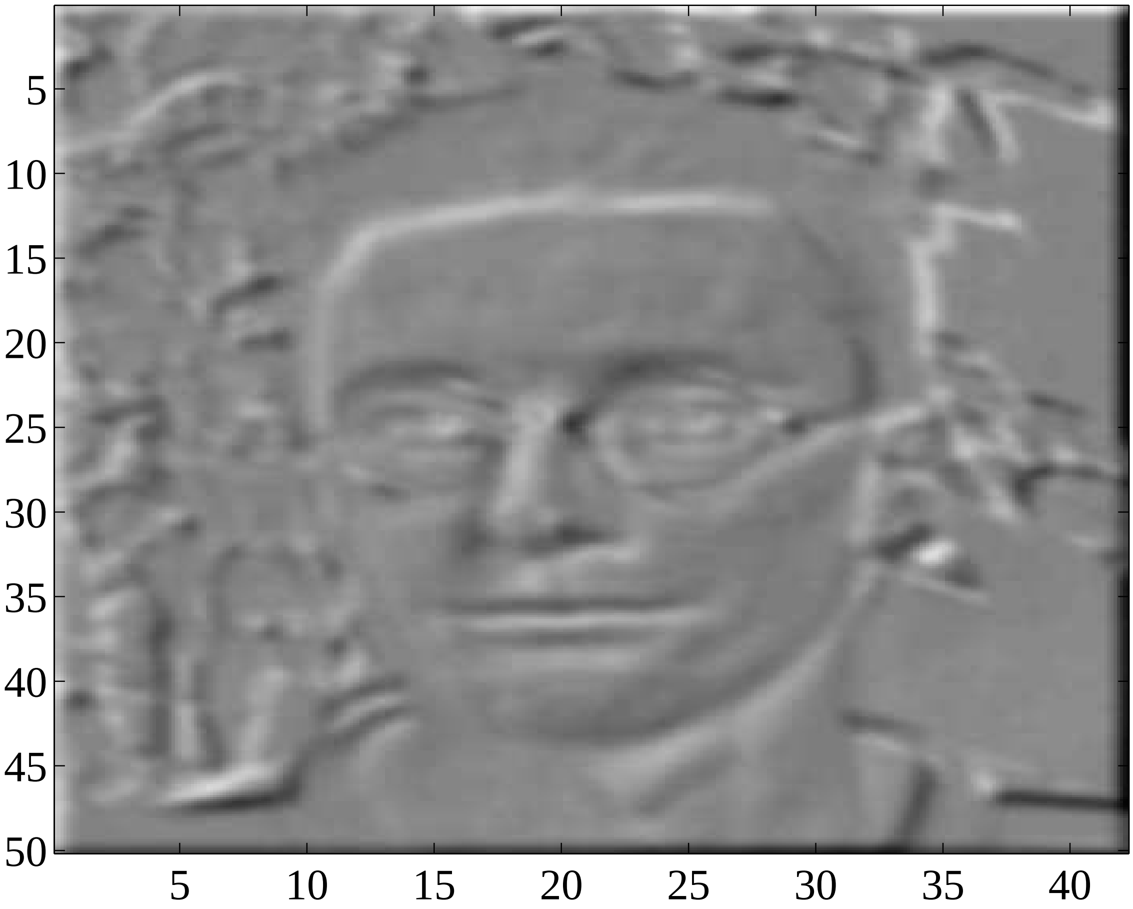


Figure 7.27: The original image convolved with the 45 degree edge extraction kernel. Edges at 45 degrees are emphasized, whereas edges at 135 degrees are suppressed. The picture looks like a flat relief.

Along the visual processing pathway of the nervous system such feature maps can indeed be found. For example in the retina there are the bipolar cells that respond to the same feature (strongly resembling a Mexican hat function) in different positions and that is also true for cells in the LGN (lateral genicleate nucleus, the part of the Thalamus relaying visual signals). Also in visual cortex one finds topologically organized cells that respond to the same feature for all positions of the visual field. But visual cortex extracts many different features and thus one could say that it consists of several colocalized feature maps and just from anatomical clues it is not possible to see which cells belong to which feature maps.

An example for an image processing system that is based on convolutions alternated with some other biologically plausible processing steps has been proposed by Stephen Grossberg etal. [16]: The boundary contour system (BCS). An aVLSI implementation was proposed by Serrano etal. [50]

Chapter 8

Cochleomorphic Circuits

8.1 The Cochlea

The cochlea is a mechanical filter for hydrodynamic waves. Figure 8.1 shows a cross section of the outer ear, middle ear and inner ear (i.e. the cochlea). Sound is channeled by the outer ear. The filter property of the outer ear is direction sensitive, so sound is subtly changed dependent on the direction it comes from. This allows us to judge if a sound comes, for example, from above or behind us. In the middle ear the oscillations of the eardrum get conveyed by the ossicals, i.e. the body's smallest bones, to the oval window. It is believed that not much filtering is going on in this step, just some clipping of loud sounds. Past the oval window the sound waves enter the liquid that fills the cochlea. A traveling wave can be observed along the basilar membrane (cross section in figure 8.2). The stiffness of the basilar membrane is decreasing deeper into the cochlea, therefore it acts as a low pass filter with decreasing cutoff frequency with distance from the oval window. The outer hair cells provide active feedback that causes resonance close to the cutoff. Thus, something like a spectral analysis is performed mechanically in the cochlea, where different frequency components of the input signal are extracted in different positions. The inner hair cells then translate mechanical motion into electrical signals to enervate the ganglion cells that then send action potentials via the auditory nerve to the brain. There is also feedback from the brain to the outer hair cells that is believed to actively modify the resonance properties.

8.2 Silicon Cochlea

In [32] it has been suggested that a 'silicon retina' can be obtained for the use in artificial systems or implants by modeling mechanical and electrical properties of the biological cochlea in aVLSI CMOS electronics. The local mechanical filter properties of a section of the basilar membrane closely match the electrical filter properties of a suggested second order filters (figure 8.4). Cascading these filters with exponentially decreasing time constants one obtains a model of the entire basilar membrane.

The individual second order stage can be described by the differential equation (8.1):

$$\ddot{V}_{out} = \frac{1}{\tau_1 \tau_2}(V_{out} - V_{in}) - \left(\frac{1}{\tau_1} + \frac{1}{\tau_2} - \frac{1}{\tau_3}\right)\dot{V}_{out} \tag{8.1}$$

To make a classical second order filter, τ_1 and τ_2 need to be the same. The spectrum of such individual filters can be seen in figure 8.5. They resonate (amplify) close to their cut off frequency and have a moderate roll off (steepness of the cut off). The resonance can be controlled

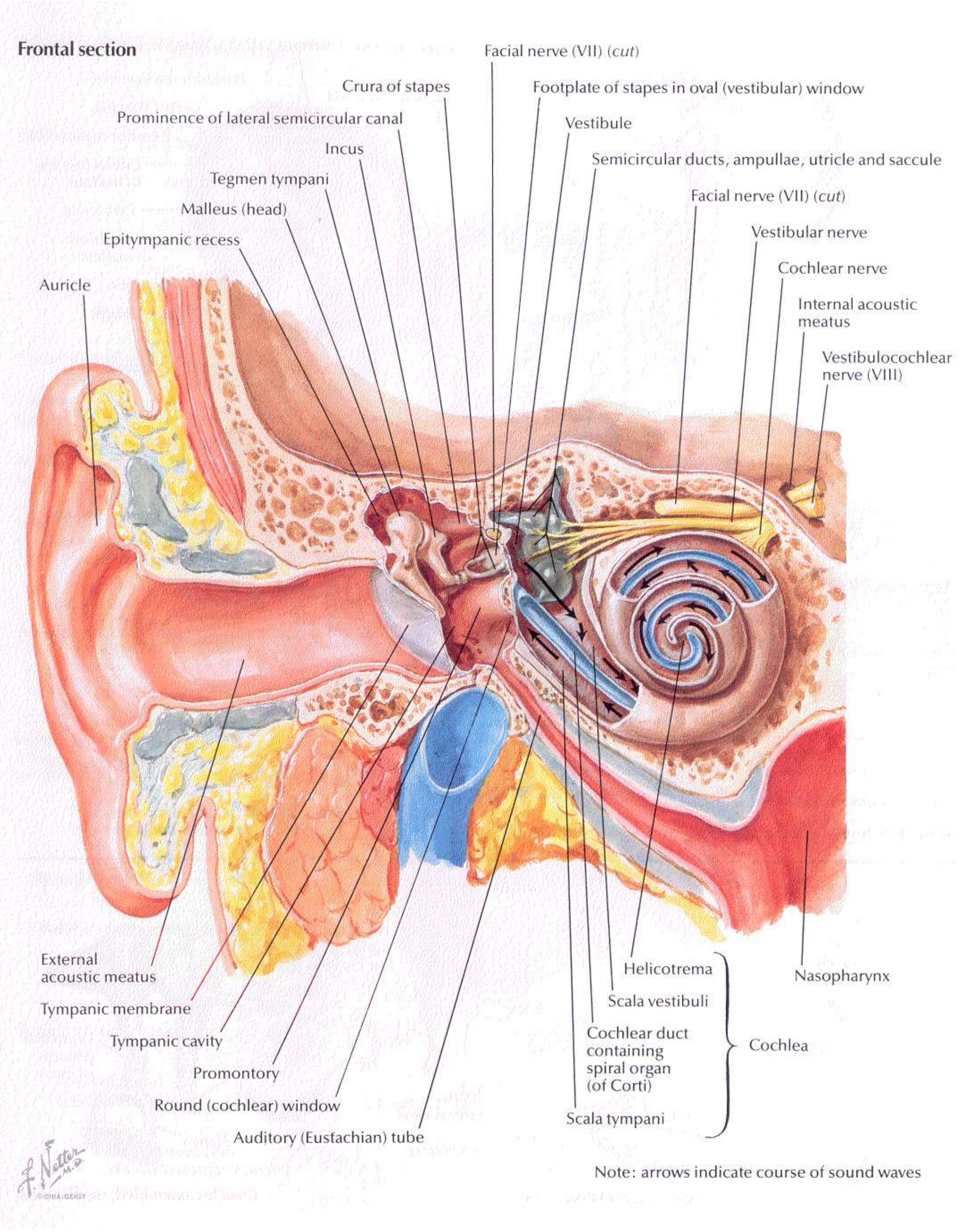


Figure 8.1: Cross section of the ear according to [40]

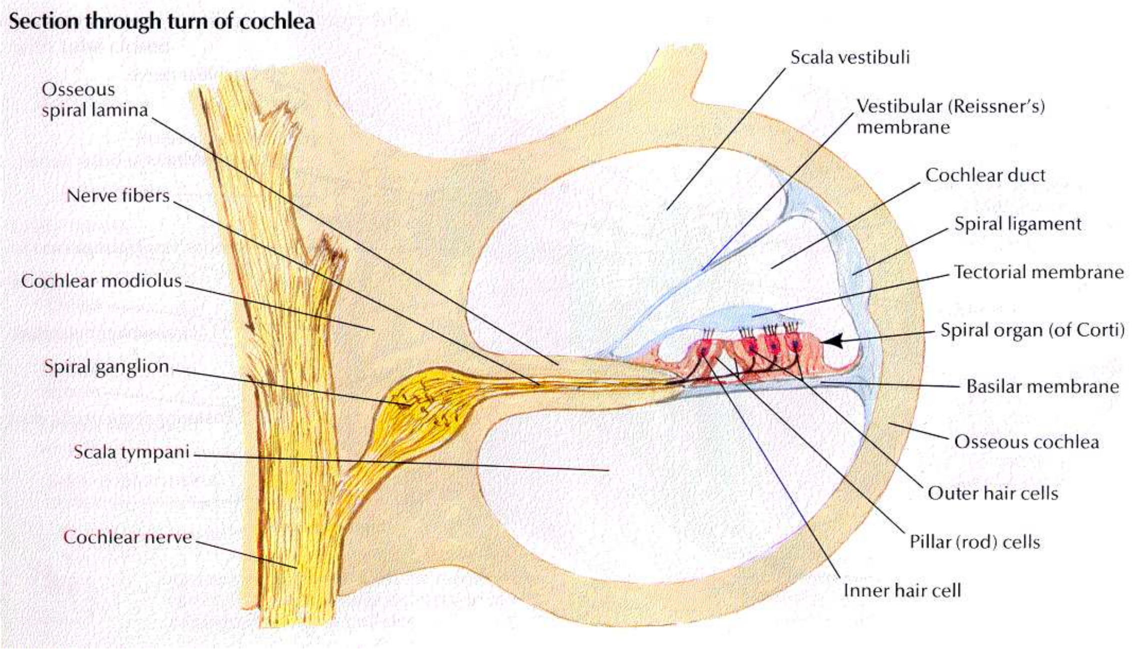


Figure 8.2: Cross section of the cochlea according to [40]

by τ_3: the longer this time constant is the less resonance. For the figures 8.5 and 8.6 it was set to 120% of $\tau_1 = \tau_2$. The basilar membrane is said to decrease its time constant exponentially with distance from the oval window. So the spacing of the time constant of the displayed filters is also exponential. This is rather easily achieved, as these time constants are inversely proportional with the bias current of the transconductance amplifiers. And those bias currents are themselves exponentially dependent on the bias voltage. Thus, a linear spacing of the bias voltages along the cascade yields the desired result. This can be provided by a series of resistors, for example.

Now by cascading those filters, the filter properties change to the liking of engineers and nature alike: the amplification of the resonance frequency gets bigger (since the amplified frequencies overlap from stage to stage), the specifity to it gets sharper, and the roll off steeper. This is shown in figure 8.6. Accumulating resonance this way is also less threatening to circuit stability.

The authors of [32] also model the inner hair cells (as half wave rectifiers) and the ganglion cells (as integrate and fire neurons). Others have also taken the approach further (as for example in [49]) and different versions of the basic model presented here exist. Since the system closely matches the properties of the Cochlea, it is a good analog front end for all kinds of natural speech processing.

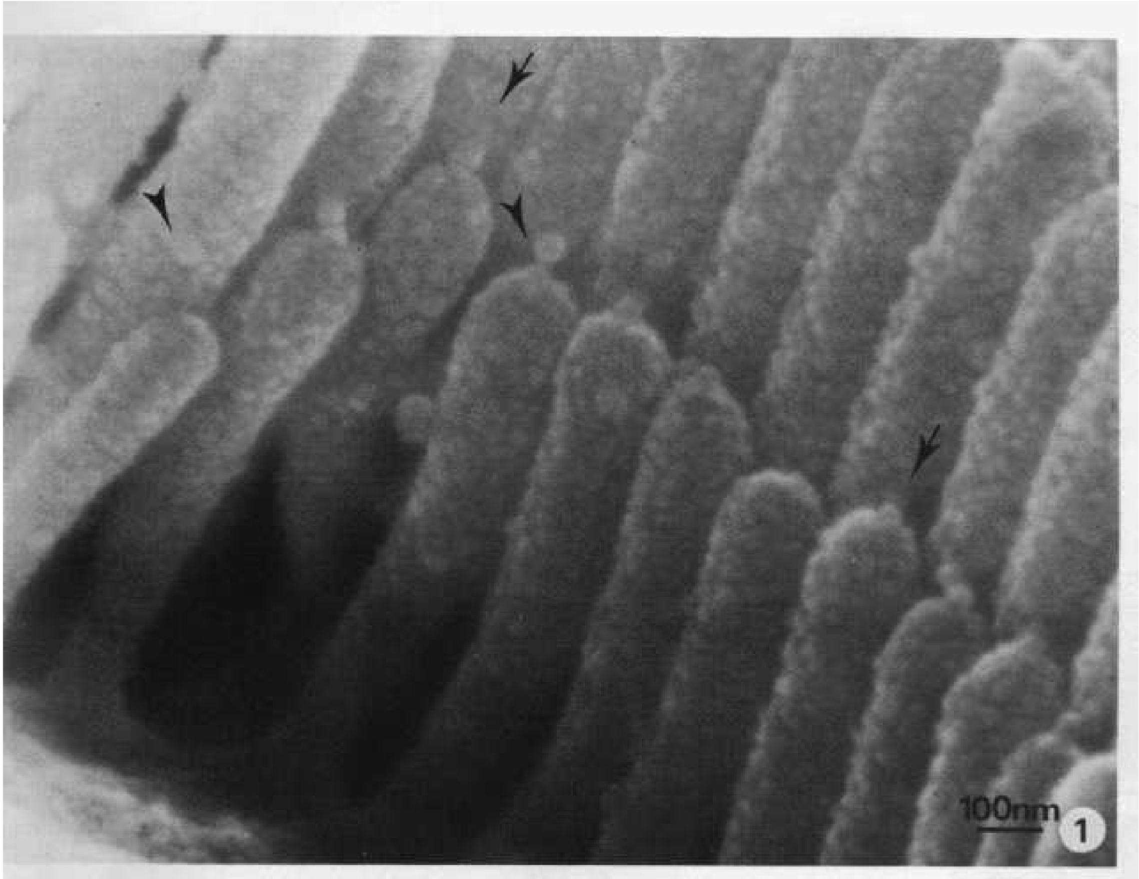


Figure 8.3: An electron microscope image of the hairs of hair cells

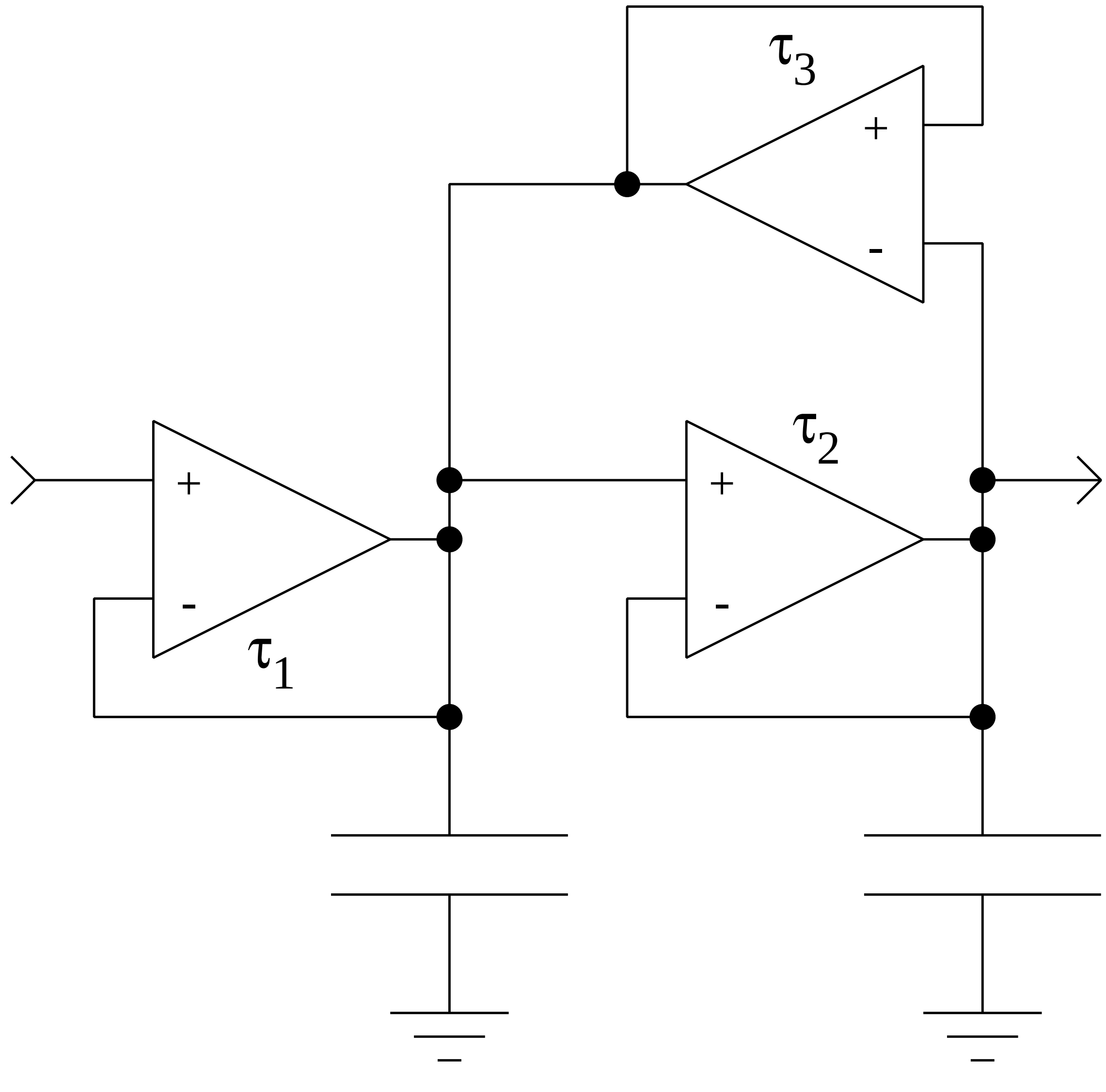


Figure 8.4: A second order filter stage modeling the basilar membrane

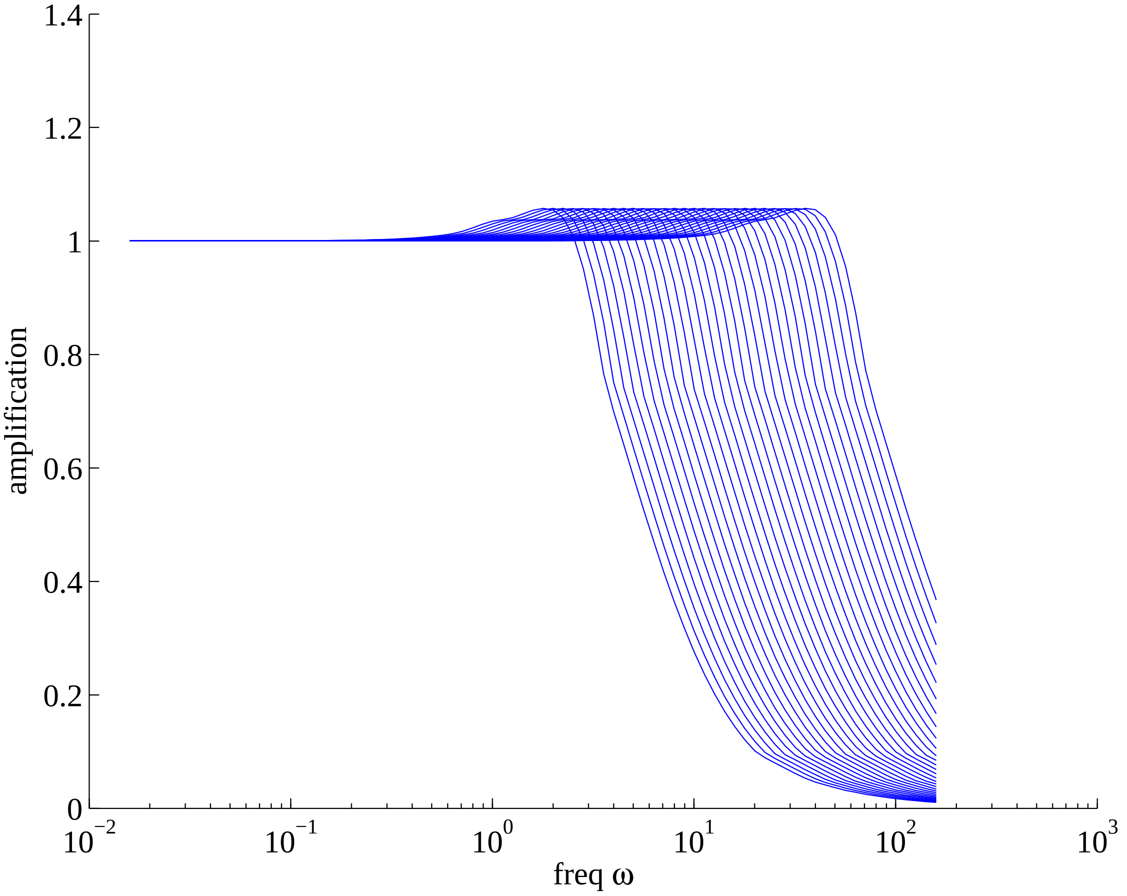


Figure 8.5: Spectrum of resonant second order filter stages tuned with exponentially increasing time constants

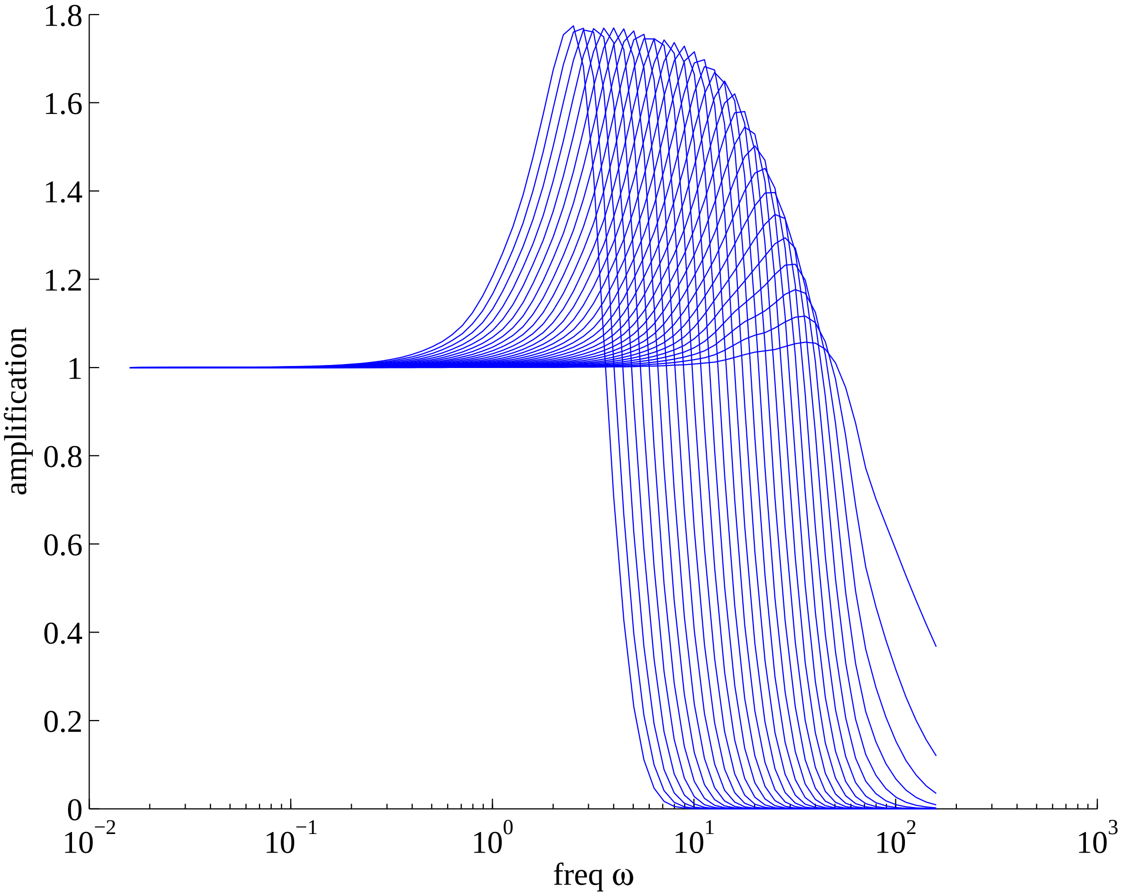


Figure 8.6: Spectrum of resonant second order filter stages tuned with exponentially increasing time constants when put in a cascade

Chapter 9

Neuromorphic Learning

9.1 Neural Learning Algorithms

What is learning? The answer is not so easy. Many equally valid definitions exist. The view of people who work with machine learning might be formulated as this:

Definition: Learning is the search of a parameter space in order to optimize performance.

In this view there is a vocabulary of terms that we are going to use as learning is discussed further:

$\vec{w}$ The vector of parameters that defines the multi-dimensional search space of the learning algorithm. In neural networks these parameters are the synaptic *weights* and they are sometimes also organized in a matrix $\mathbf{W}$, instead of in a vector.

$\vec{x}$ The (sensory) input to the system. Also organized as a matrix sometimes, e.g. when the input is an image.

$\vec{y}(\vec{x},\vec{w})$ The (behavioural) output of the system that changes with learning.

$\vec{d}(\vec{x})$ The 'desired' output of the system, a teacher or expert opinion. It is normally not defined for the whole input space spanned by $\vec{x}$. (Thus, the system needs to *generalize* what we learn from a teacher and apply it to unknown situations $\vec{x}$)

$P(\vec{y})$ A performance evaluation function to judge the quality of the learning state of the system. In general this function can be stochastic. It can also have multiple extrema.

$E(\vec{y},\vec{d})$ An error function, a special case of a performance evaluation function, that evaluates system performance as compared to the expert/teacher.

μ The learning rate. A parameter that is used by many learning algorithms influencing the learning speed and quality.

9.1.1 An overview of classes of learning algorithms

There are several attributes that have been used to classify learning rules. Maybe at the top level of a classification they can be divided into supervised and unsupervised learning rules. The following list tries to give an example of how that classification could be further refined.

- supervised
    - supervised by expert (learning a target function)
        - continuous target functions
            - gradient descent, Error Backpropagation (for space continuous target functions, interpolation)
            - temporal difference learning (TDλ, for time continuous target functions)
            - statistical methods, interpolation
            - weight perturbation (can also be used in reinforcement)
        - supervised classification
            - supervised LVQ
            - support vector machines
    - reinforcement, supervised by critic (optimizing performance)
        - associative reward-penalty
        - evolutionary algorithms
        - deduction, induction (predicate logic learning)
- unsupervised (optimizing statistics, data reduction/compression)
    - Correlation, Association, Hebbian Learning, Spike based learning
    - Competitive Learning, LVQ, Competitive Hebbian Learning
    - Optimizing data reduction, PCA, ICA

9.1.2 Supervised Learning

The performance measure in supervised and reinforcement learning is dependent on feedback external to the system, i.e. feedback from an expert or critic. These algorithms are mainly based in psychology and mathematics. How supervised learning operates on the level of the neural system is a mystery not yet solved.

Example: General Gradient Descent Optimization and the Error Backpropagation Algorithm for ANNs

The most famous supervised learning algorithm, the Error Backpropagation algorithm [47, 22], is applied to multi layer Perceptrons (MLP, see also chapter 4) which are neural like structures. It is a **gradient descent** algorithm that minimizes an error function. That could for example be the square of the difference of a desired output to the actual net output over the input range where $\vec{d}$ is defined or over a predefined subset of inputs.

$$E = \|\vec{d} - \vec{y}\| \tag{9.1}$$

The weights are initialized randomly, the error is computed, and the derivative of the error with respect to every weight. The weights are then corrected proportional to their derivative and the learning rate μ in the inverse direction of the derivative.

$$\frac{d\vec{w}}{dt} = -\mu \frac{dE}{d\vec{w}} \tag{9.2}$$

If μ is small enough, this will necessarily result in a reduction of the error. This is repeated until the error is acceptable. The name 'Error Backpropagation' is a result of a particularly elegant algorithm to compute these derivatives in a MLP, by retrogradely traversing the neural network (check the literature [47, 22] for details!). The general concept though, is simply that of gradient descent in an optimization problem, that can also be applied to all kinds of other parameterized functions, of which MLPs and ANNs are but one example.

9.1.3 Reinforcement learning

Reinforcement learning is similar to learning with an expert, but there is no desired output $\vec{d}$ defined for any input examples. Thus the performance measure P is not an error function but some judgment on the quality of the output. One quite famous example of this category of learning algorithms is TD(λ) [54, 55]. Genetic algorithms could also fall under this definition.

9.1.4 Unsupervised learning

Unsupervised learning also tries to optimize performance. But this time the performance measure does not depend on direct external feedback. Mostly it tries to optimize encoding of a huge input space to a limited capacity information channel. It tries to get as much relevant information as possible over that channel and adapts according to the input statistics.

Hebbian learning

Hebbian learning is THE form of learning that has actually been observed in real neurons. Hebb formulated it as follows [19]:

> When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.

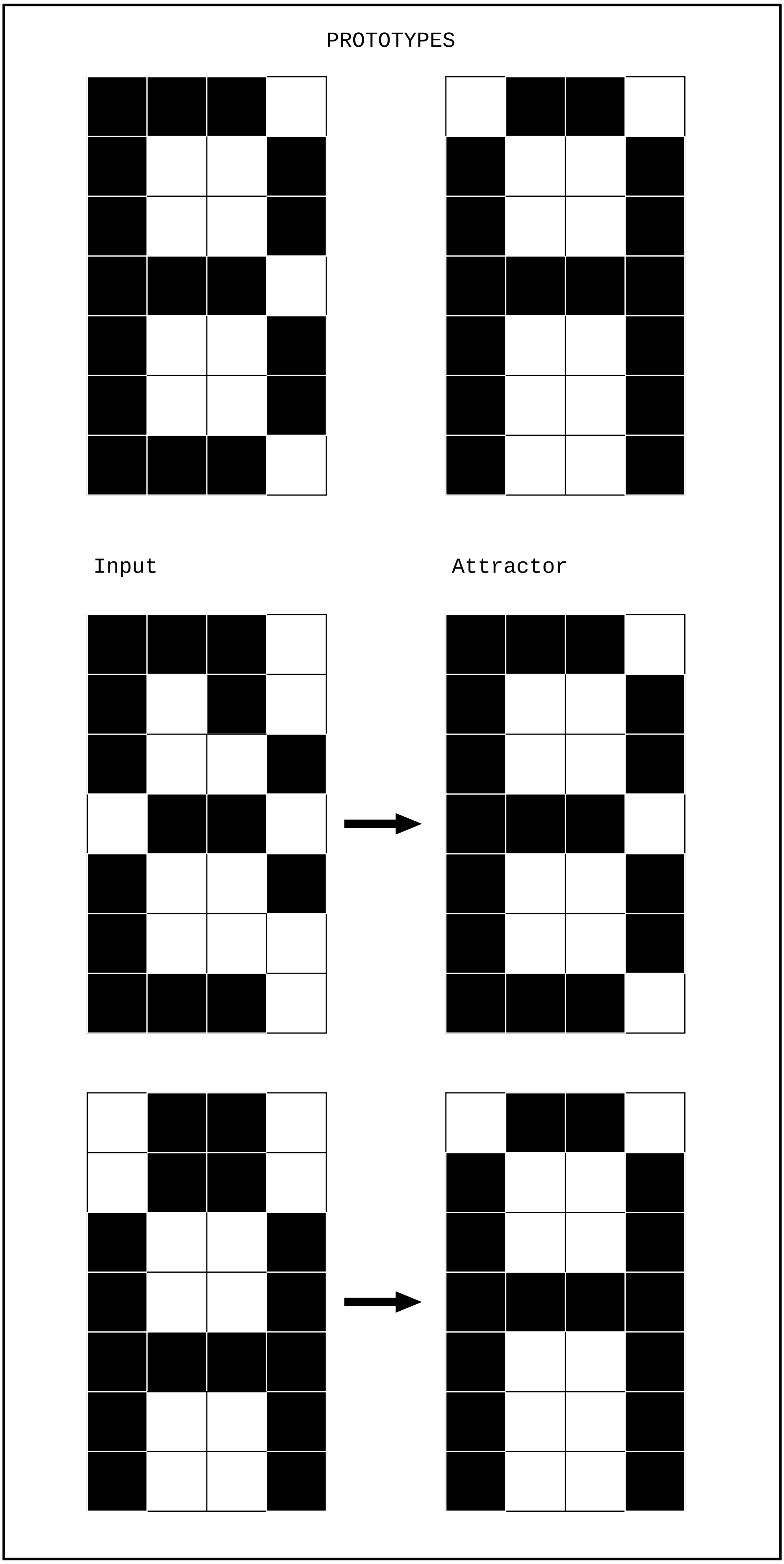
This postulate inspired a number of so called Hebbian learning rules. They contain a positive term of the input to the synapse multiplied with output of the neuron in their weight update rule. Thus, they 'reward' correlations of input and output by a weight increase.

$$\frac{dw_i}{dt} = ...x_i y_i... \tag{9.3}$$

This kind of Hebbian learning changes the weight vector's direction $\frac{\vec{w}}{\|\vec{w}\|}$ towards the direction of the input vector $\frac{\vec{x}}{\|\vec{x}\|}$, i.e. the angle α between the weight vector and the input vector is reduced. If the neuronal model is that of a perceptron, then the output is defined as:

$$y = f(\vec{w}^T \vec{x}) = f(\cos\alpha \|\vec{w}\| \|\vec{x}\|) \tag{9.4}$$

Where f is normally a monotonically increasing function. Which means that the response to that particular input is increased by the learning.


PROTOTYPES
Input
Attractor

Associative memory

Hebbian learning has for example been used to train 'associative memory' or 'attractor neural networks' or 'Hoppfield Tank networks'. These are neural networks that are normally arranged in a two dimensional topology. In other words, neurons can be thought of as pixels in a two-dimensional image. Every neuron receives one individual input. The input to the entire net can therefore again be thought of as an image. And all neurons are 'horizontally', bidirectionally connected to all other neurons, or a random selection of other neurons, or a neighbourhood pattern (The later is called Cellular Neural Network.)

In a training phase input images that should be stored are presented to the neuron's inputs (figure 9.1, top). Hebbian learning strengthens the connections between neurons that are active in the same image and weakens or makes inhibitory the connections between neuron pairs where one is active and the other is not.

$$\frac{dw_{i,j}}{dt} = \frac{1}{t}(-w_{i,j} + x_j y_i) \tag{9.5}$$

where t is the number of input patterns that have been presented so far. This learning rule works with binary (0/1) model neurons with threshold 0.5).

When the learning is completed and all input images 'stored' one can briefly present other input patterns to the net and let it run freely afterwards, driven only by the recursive connections between the neurons. It turns out that the net will approach one of the stored input patterns which have become stable states of the recursive activity in the network (figure 9.1, middle and bottom). Like this input patterns that are close to a particular stored pattern, are 'associated' with that pattern. Letter recognition is a typical task applied to that kind of network.

The presence of such associative memory in the brain is debatable. A main argument against it are that it requires time to converge and usually we are really fast in recognizing objects. It is also not quite clear how to use the associated pattern on a next level of computation. It would need to be read or interpreted somehow.

Competitive learning

Competitive learning is an important concept that can also be used for fuzzy pattern recognition. In contrast to associative memory the recovery of a pattern is not expressed by the output of the whole neuron assembly but strong activity of individual neurons will represent the presence of a specific input patterns. That means in other words that each neuron reacts to a specific chunk of the input space. The basic idea is that neurons adapt to the present input in a way that the output gets increased for that present input. They do adapt to that input with different speeds such that that neuron that has the strongest output already adapts the most. Actually in many variants of competitive learning ONLY the strongest neuron is allowed to further adapt. This can be achieved by making the degree of adaption proportional to the neurons output. (This is for example the case for classical Hebbian learning.) And if the neurons are connected to form a WTA network, then indeed only the winner adapts to the present input.

One classic example that will be used to illustrate competitive learning in the following is 'Learning Vector Quantisation' (LVQ).

Learning Vector Quantisation (LVQ)

We want a system to observe a certain input space. (That could for example be output of a number of sensors, e.g. a camera.) Certain inputs should trigger certain responses. An easy way of achieving this would be to store an action for every possible input. But that most often would exceed the storage capacity of a system, most certainly so, if the input space is infinite. The

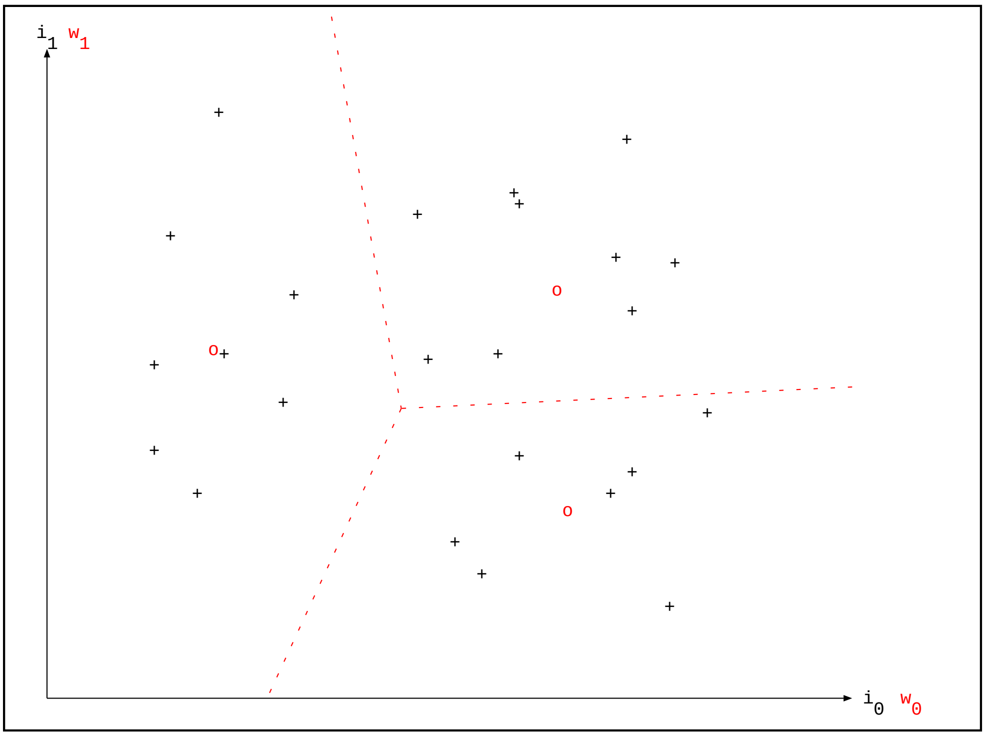


Figure 9.2: Classification with LVQ

solution is to 'quantize' the input space by **classification**: One finds rules that project the inputs into a number of classes. And the number of classes is chosen small enough that the appropriate action for each class of inputs can be stored.

One way to classify vectors of real numbers are competitive neurons or winner take all (WTA) neurons. Winner take all neurons compete among each other in a network and eventually only one neuron remains active. The basis for a WTA network can be almost any neuronal model. All the neurons in the network receive the same feed-forward input, i.e. they share the same input space. Now in addition to that basis model, every neuron inhibits every other neuron in the network. If the inhibition that a neuron exerts on its fellow neurons is equal to its activity, the only stable state of the network is when only the strongest neuron remains active. Every neuron represents one class of inputs.

A particular kind of neurons are abstract neurons whose activity is inversely proportional to the distance of the input vector to the weight vector.

$$y = \|\vec{w} - \vec{x}\|^{-1} \tag{9.6}$$

Other distance measures than the Euklidian one can also be used and the choice depends on the particular application. If you use such neurons as WTA neurons, then the one whose weight vector is closest to the input vector wins. In the figure 9.2 it is illustrated how such neurons quantize an input space.

We humans too approach our daily live with subdividing the perceived world into digestible chunks. We deal with 'similar' situations in the same way and what we do think of as similar is actually determined by our own kind of 'distance function'.

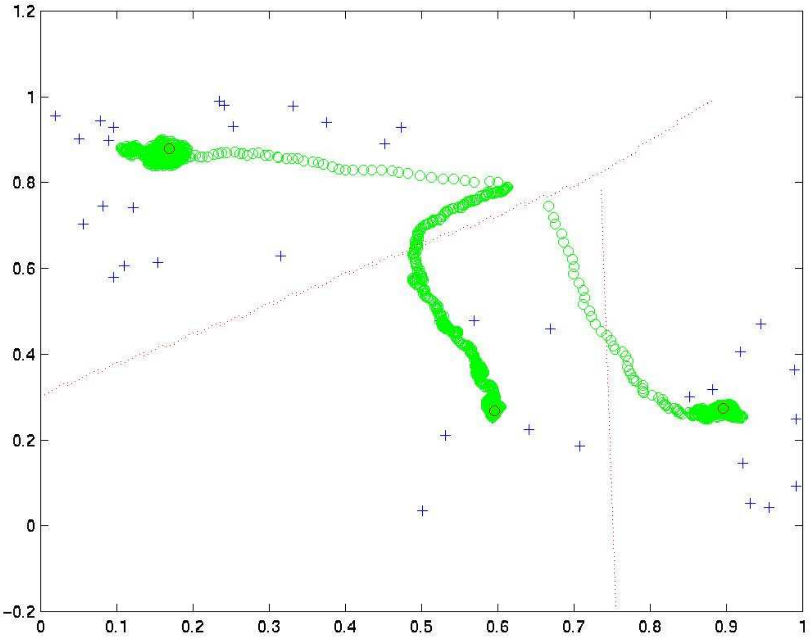


Figure 9.3: Dynamics in Learning Vector Quantization

Given a particular distance function that determines the activities of WTA neurons a quantization can actually be learnt by the neurons by unsupervised learning. This makes sense as a first data reduction stage in a more extensive learning system. Unsupervised learning does not know about any action that could be an appropriate response to a certain class of inputs. It classifies the inputs purely by statistical criterions, e.g. it tries that the classes give a good/optimal representation of all the input examples that it experiences during learning.

The basic idea is quite simple. The weight vectors are randomly initialized at first. Then the networks begins to watch the input space. The winner who is active at any one time corrects its weight-vector in the direction of the input vector. For example

$$\frac{d\vec{w}}{dt} = \mu y(\vec{x} - \vec{w}) \tag{9.7}$$

An example of how the weight vectors move is shown in figure 9.3. It is an example of three neurons with two inputs that they all share. It is convenient to draw the inputs and the weight vector into the same coordinate system. All the inputs are the crosses. They appear repeatedly in random order. The green circles are the weight vectors of the three neurons, drawn repeatedly during learning. They start close to each other and then start to move towards clusters in the input distribution. Actually there are two intentionally distinct clusters in the lower left and upper right. One neuron becomes responsible for the upper left cluster. The other two neurons define their own subclusters within the lower right cluster. The dashed lines are the separation lines of the three clusters after learning is completed.

Competitive Hebbian Learning

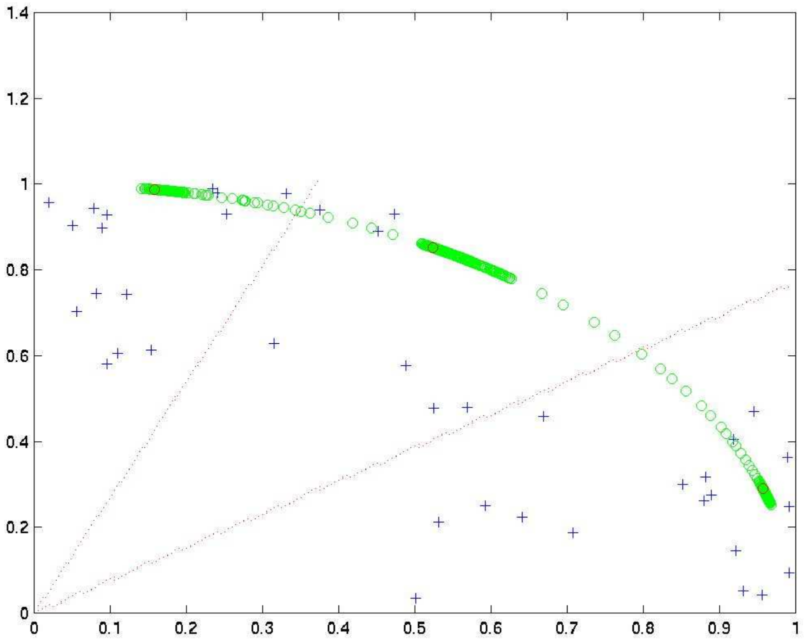


Figure 9.4: Dynamics in competitive Hebbian learning with Oja's learning rule

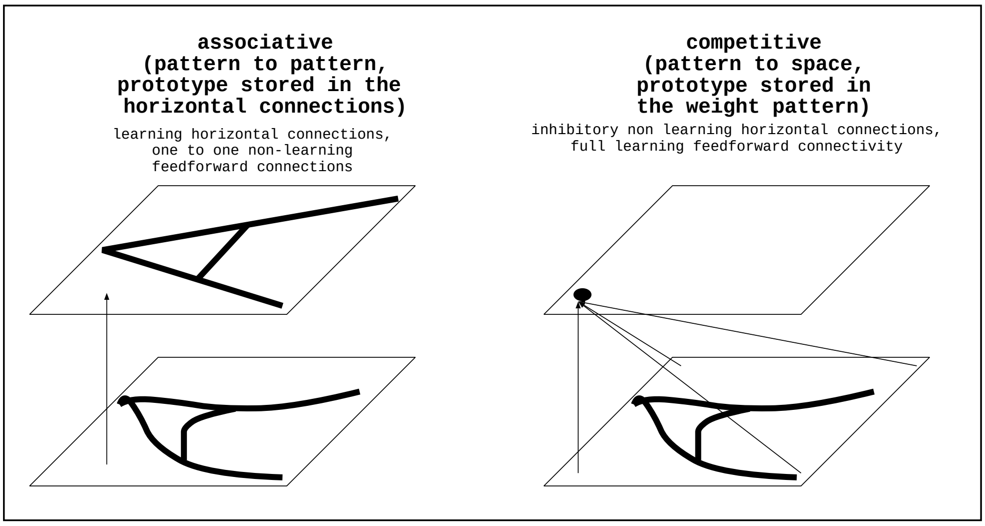


Figure 9.5: An illustration on the differences of competitive learning an associative memory.

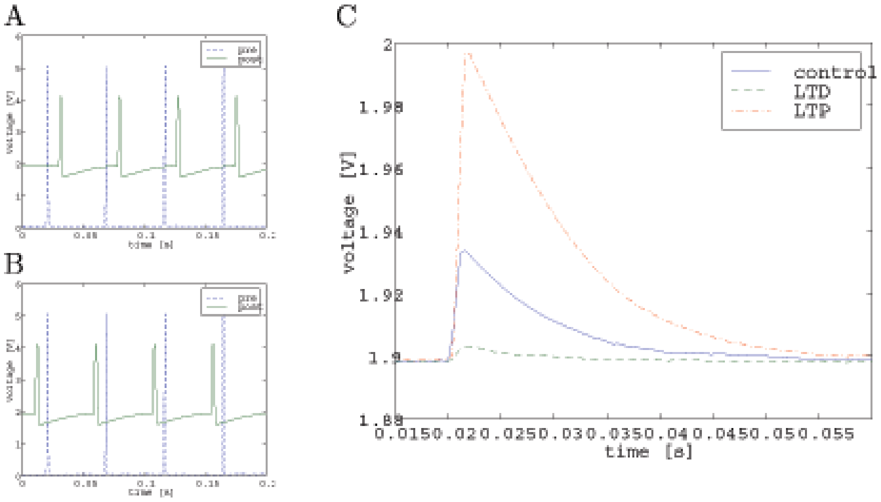


Figure 9.6: Spike based learning behaviour implemented into a silicon neuron

Hebbian learning in a competitive network with (more plausible) linear perceptrons can be used to the same ends as LVQ. The difference is that the input space is now divided along rays from the coordinate system origin. The relative length of the weight vectors and the difference in their angle determines how wide that chunk is. Figure 9.4 illustrates this division and the dynamics of the learning process with one particular Hebbian learning rule:

$$\frac{d\vec{w}}{dt} = y(\mu_0\vec{x} - \mu_1 y\vec{w}) \tag{9.8}$$

This rule is known as Oja's learning rule [30]. It is a Hebbian learning rule that keeps the length of the weight vectors constant (on condition that the neuron is a perceptron with the identity as the activation function). That's why the weight vectors, marked as green circles, only move along a circle around the origin. The inputs (the crosses) are the same as in the LVQ learning example in figure 9.3. The separation of the space by the neurons happens now according to a different rule, along separation lines through the coordinate space origin. That is because the neurons' outputs are defined according to equation 9.4, where for a given input vector $\vec{x}$ and a constant weight vector length $\|\vec{w}\|$ the output only depends on the angle α between input and weight vector. Thus, the neuron with its weight vector at the smallest angle to the input vector wins.

The input to a ANN trained by competitive learning, like for associative memory, can for example be an image. But unlike in associative memory, every neuron receives input from the entire picture, not just from one pixel. And the stored prototypes would not be represented in the inter neuron weight pattern, but one individual neuron would store a prototype image in its weight pattern and be responsible for that particular image. Figure 9.5 is an attempt at illustrating those distinguishing properties of the two approaches.

The representation of input patterns by individual neurons has also been observed in recordings in visual cortex and seems, thus, a more plausible neurophysiological model.

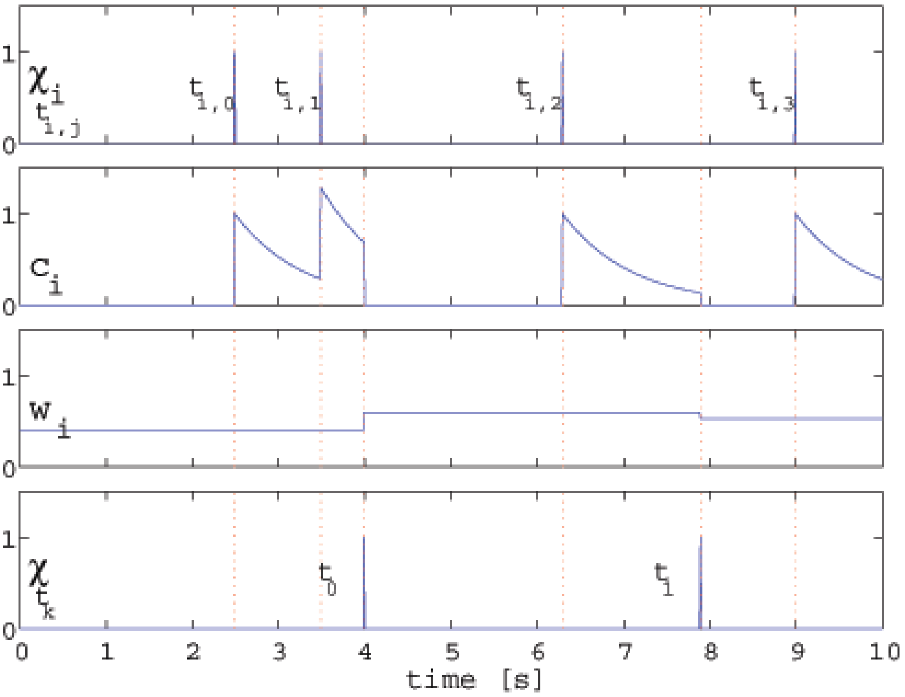


Figure 9.7: Simulation variables involved in a particular spike based learning algorithm at one synapse.

Spike based learning

Spike based learning is also known as spike timing dependent plasticity (STDP). All the learning rules we have been talking about until now can work on Perceptron-like neurons, either with a continuous or binary input and output. But lately there has been a lot of talk about so called spike based learning rules. Those operate on neurons that receive and send spike signals in a manner that does not allow for them to be simplified to average frequency signals. To handle such signals in differential equations they are often expressed as sums of Dirac delta functions:

$$\begin{aligned} x_i &= \sum_k \delta(t - t_{i,k}) \\ y &= \sum_s \delta(t - t_s) \end{aligned} \tag{9.9}$$

Such spike based learning behaviour has been observed in physiological experiments [36] and it has been suggested that it can be used in the manner of Hebbian learning to correlate spikes (Instead of frequency signals), although concrete practical applications are still sparse. Normally spike based learning rules increase weights of synapses whose spike input precedes an output-action potential.

Figure 9.6 shows spike based learning in a silicon neuron. On the left hand (A and B) the input and the output are depicted. The output looks more like a biological action potential with hyperpolarisation (overshoot) after the pulse. The inputs are the narrow pulses. On the right (C) the EPSPs (electrical postsynaptic potential) that are caused on the cell membrane voltage by a single spike input to the cell are shown. The middle trace is an EPSP before learning. The top trace is an EPSP after the stimulation pattern in figure A has been applied for 50 pulses. And finally the lower trace is the result of the stimulation pattern in figure B.

It is not yet very clear how nature uses such a spike based learning behaviour. It could for example be used like Hebbian learning with a very short time constant. Such that single spikes within a certain short time window are considered correlated. Similar properties like in 'normal' Hebbian learning can be achieved by this, only that the variables are average frequencies no longer but spike occurrences. The time delay between the incoming and outgoing spike however makes it hard to have symmetric bidirectional connections. It seems rather that in case of bidirectional connections between neurons one of the directions will be suppressed.

One learning rule that leads to such behaviour is the one I described in my thesis [17]. There is a variable that we called 'correlation signal' present in every synapse ($\vec{c}$). It gets increased by presynaptic pulses and the weight is updated with every postsynaptic pulse, the change dependent on that correlation signal (Note that y is now a sum of Dirac delta functions):

$$\frac{d\vec{w}}{dt} = (\mu_0 \vec{c} - \mu_1 \vec{w}) y \tag{9.10}$$

In figure 9.7, the weight changing behaviour of a single synapse is shown. Presynaptic spikes are depicted on the top axis, postsynaptic ones on the bottom. c evolves as described in the text and is reset by postsynaptic APs and w according to (9.10).

9.2 Analogue Storage

We defined learning as the search of a parameter space. If a learning algorithm should be implemented in aVLSI, this parameter space and the state of the search needs to be represented

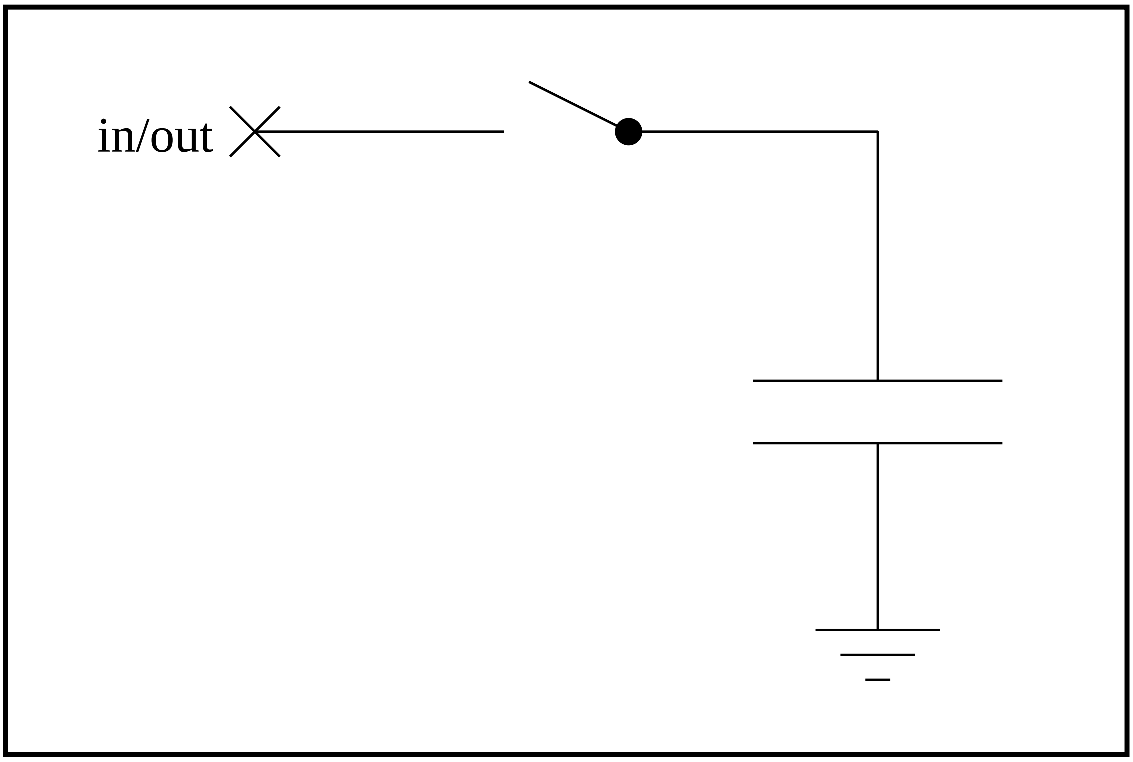


Figure 9.8: Capacitive dynamic analog storage

somehow. In most learning algorithms that parameter space is continuous and could be represented by a vector of real numbers. How can such analog parameters be represented and stored in aVLSI? Several methods have been explored:

- volatile short term storage
  - capacitive storage/dynamic storage
- volatile long term storage
  - digital storage and DAC
  - capacitive storage with refresh
  - multi level flipflops
- non-volatile storage
  - floating gates, analog flash ROM

9.2.1 Dynamic Analogue Storage

The problem of analog storage on a aVLSI chip is simple if one has no big requirements on the duration of the retention. Simple capacitances that are connected to transistors for addition and removal of charge, can hold analog voltages with a leakage rate in the order of millivolts per second.

Pro
- Real analog
- Easy to write
- As gate readable uninvasively by current through a transistor
- Compact

Contra
- Only short term, leaking some mV per second

9.2.2 Static Analogue Storage

Static storage as well as dynamic storage is volatile storage. Volatile storage needs a power supply to maintain its state. In static storage that state is maintained through some sort of internal feedback. The classical digital example is the flip-flop. Positive feedback drives it to one of two rails. But there is up to now no fully analog static memory available. Usually quantized (multi-level) storage is used, with a limited resolution. This can for instance be achieved by the combination of digital memory and analog to digital / digital to analog (AD/DA) conversion.

A AD/DA conversion memory cell has been used in a learning context in [20]. The storage cell that they used is described in figure 9.9. While 'restore' is high, the stored current is mirrored to the synaptic current I_{syn}. When the signal 'restore' goes low and the signals C_i are applied in the sequence at the bottom of the graph, the current I_w is stored into the memory cell. The bit blocks have a bias current that is doubled from left to right. Thus the first cell to receive a low control pulse C_2 has the biggest bias current, i.e. $4I_b$. The current I_w (that should not exceed $8I_b$) is compared in it to that bias current. The result of that comparison sets the latch (the feedback double inverters on top) in that 1 bit memory block. As C_2 goes high again, this 1 bit block now supplies the current $4I_b$ if $I_w > 4I_b$, otherwise it supplies no current. Thus, as the cell C_1 is now coupled to the common line, the current drawn from it is the rest of $\frac{I_w}{4I_b}$. The process is repeated until the last cell is reached and all 1 bit blocks together now supply I_w with a precision of I_b.

Pro of the AD/DA static memory
- Easy to write
- Uninvasive direct reading

Contra of the AD/DA static memory
- Only multi-level
- Digital control circuitry and signals

But also real multi-level static memory is possible, i.e. with no internal binary representation. This has been introduced as weak multi-level static memory in [18, 46].

An important building block is the so called 'fusing' amplifier (figure 9.10). It operates like a transconduction amplifier only when the two input voltages are close enough together. If they are too far apart, the output current will turn off. If this circuit is connected as a follower, then it will drive the output towards the input voltage, if they are close already, otherwise it will not influence the output voltage.

If the outputs of an array of those fusing followers with different input voltages are coupled together (figure 9.11), this output will always be driven towards the closest of those input voltages. This results in a multi-level memory cell, very similar in principle to the binary double inverter flip-flop. If the bias currents of the fusing followers are weak and a capacitance is coupled to the output, this capacitance can be used as a fully analog capacitive dynamic storage cell for short term storage. Only on a long time scale will its content be discretized by the weak driving force of the fusing followers.

Pro for weak multi-level memory
- Easy to write

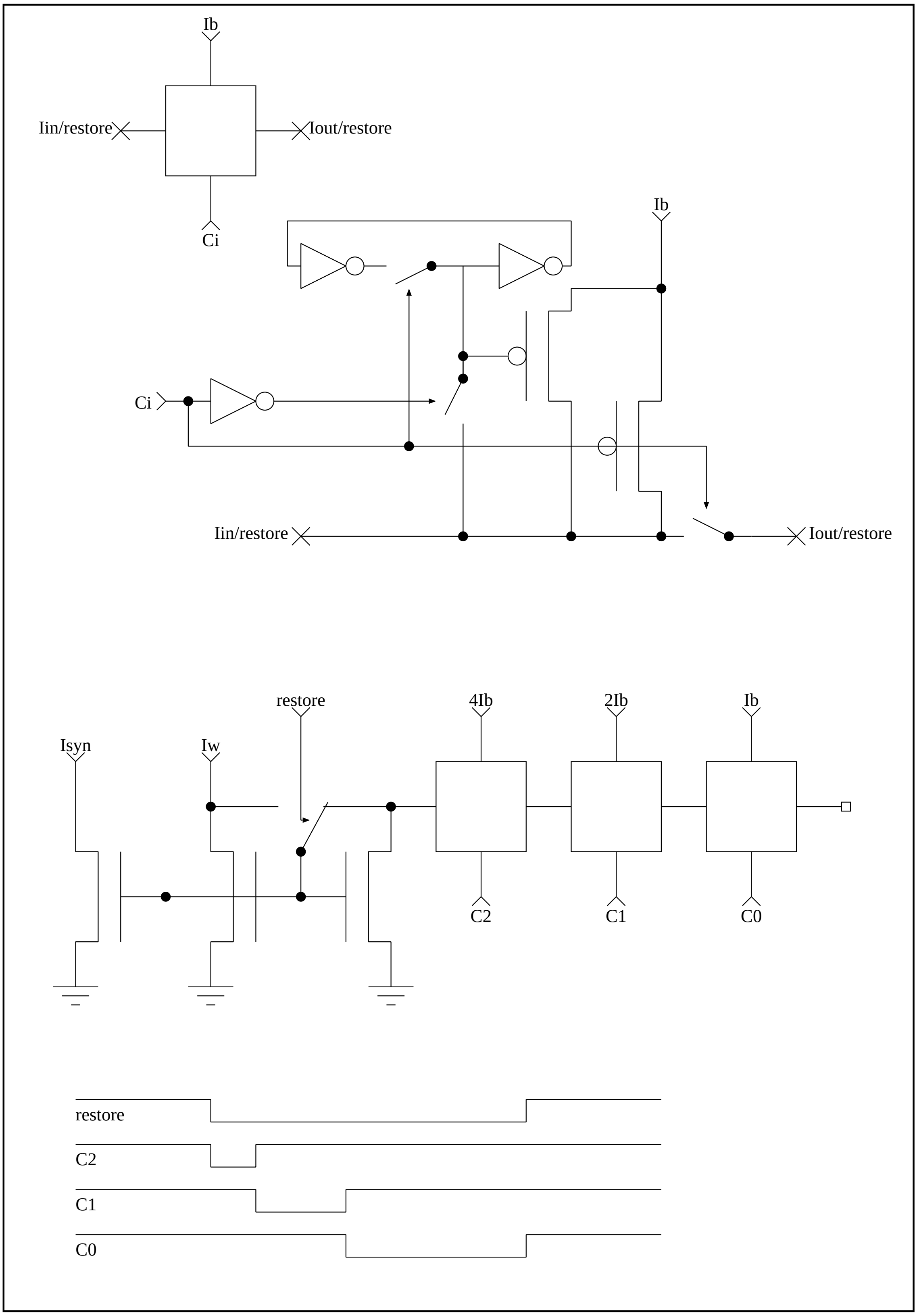


Figure 9.9: An implementation of AD/DA multi-level static storage

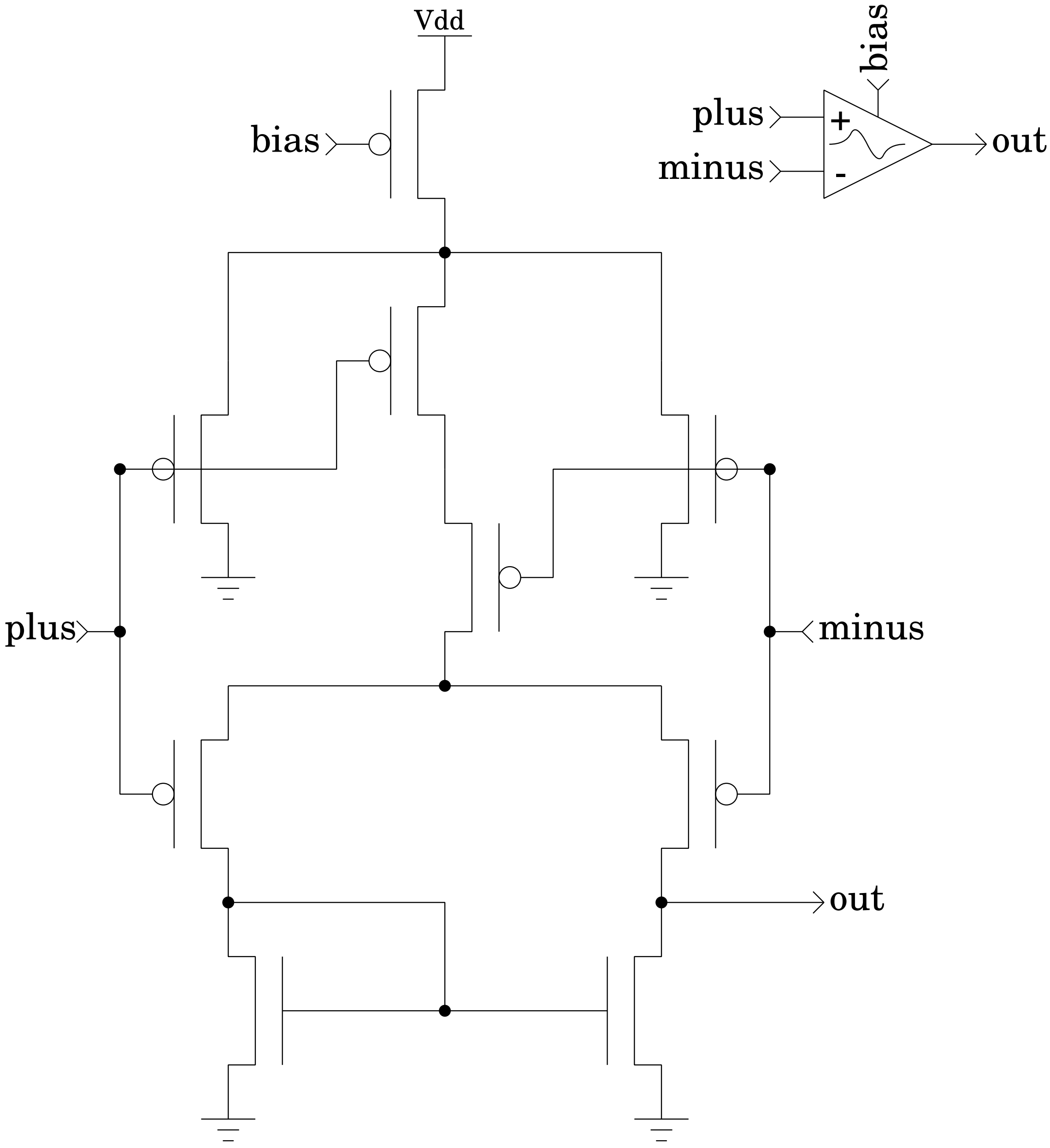


Figure 9.10: A 'fusing' amplifier, that turns of if the inputs are too far apart

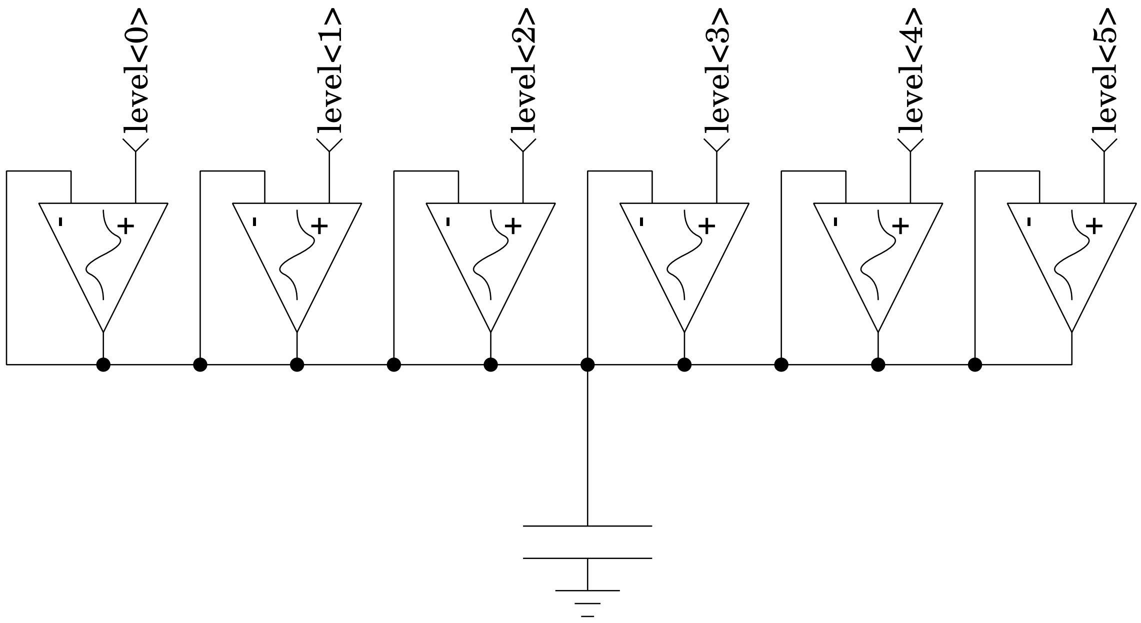


Figure 9.11: An example of real multi-level static storage: weak multi-level static memory

- As gate readable uninvasively by current through a transistor
- 'Real analog' on a short time scale
- No digital control signals necessary

Contra for weak multi-level memory
- Only multi-level on a long time scale
- Space consuming (linear increase with number of levels)

9.2.3 Non-Volatile Analogue Storage

In many aspects idealized non volatile analog storage is the most desirable way of analog storage for many applications. In reality the techniques involved are tricky and sometimes unpractical. For non-volatile storage in a electronics context, floating gate and magnetic storage have been used in the digital world. Floating gates (FG) are most promising to the neuromorphic engineer, since they are devices that can be implemented on a CMOS chip, right beside the processing circuitry. They have been used for analogue storage also [28, 25, 13]. FGs are conductors with no physical connection to any other conductors . Charge that is deposited on a FG will remain there indefinitely in an ideal world and for years in real circuits. Still charge can be added or removed across the energy barrier of about 3.2eV posed by the insulation oxide around a CMOS polysilicon conductor by several techniques (figure 9.12).

Fowler Nordheim tunneling [14] Tunneling is caused by the electron's quantum mechanic property, sometimes to be were it is not supposed to be. By increasing the voltage across the insulation, the distance from where the electrons are supposed to be to the point by which they have enough energy to be in the conductive band is reduced (figure 9.13). This increases the probability of the tunneling. It has been modeled by the following equation:

$$I_g = I_{T0} e^{-\frac{V_f}{V_{ox}}} \tag{9.11}$$

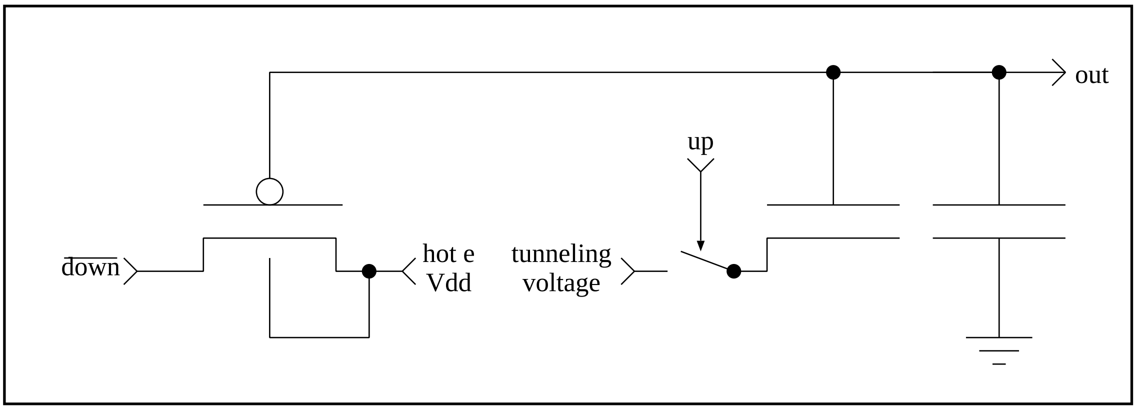


Figure 9.12: Real analog and non-volatile: Storage on a floating gate (flash ROM, EEPROM)

In a 2.0μm Orbit CMOS process, for example, the parameters were I_{T0}=984V and V_f=0.0818A.

Hot electron injection For hot electron injection, electrons must have a lot of kinetic energy (¿3.2eV), i.e. they are hot. This can be achieved by the intense electric field between the channel and the drain of a CMOS transistor. The model for the hot electron injection current is:

$$I_g = I_S \beta e^{\frac{V_{DC}}{V_{inj}}} \tag{9.12}$$

Again in 2.0μm Orbit CMOS, β=1.17e-21 and V_{inj}=0.10V .

Tunneling is used in digital flash ROMS or EEPROMS. Specialized processes with thin gate oxide make tunneling possible with voltages close to the supply voltage. In the 0.6 AMS process that we used for this in the past, about 15 V are required. Hot electron injection is usually an undesired effect, often present and sometimes bothersome in submicron processes. It can be put to work however in this context. Another method to change the charge on a floating gate besides tunneling and hot electron injection is UV-light.

A difficulty in controlling tunneling in standard CMOS processes that require voltages beyond the supply voltage is the switching of these high voltages. One can employ NFETs with lower doping in the drain. These transistors are capable to stand a high source-drain voltage. A cross section of such a transistor is shown in figure 9.14. The break through voltage of this HV-NFET is very high. In the 0.6 AMS process somewhere above 30V. A switch that switches its output between a high voltage and a voltage that is determined by the break through voltage of the normal PFETs is shown in figure 9.15. Note that the PFETs her actually are exposed to a high voltage and they are actually breaking through. But since the current is limited in this situation, they do not get damaged by this.

Pro for FG non-volatile analog storage

- Real analog
- As gate readable uninvasively by current through a transistor
- Rather compact
- Stores value even without power supply

Contra for FG non-volatile analog storage

- Difficult to write

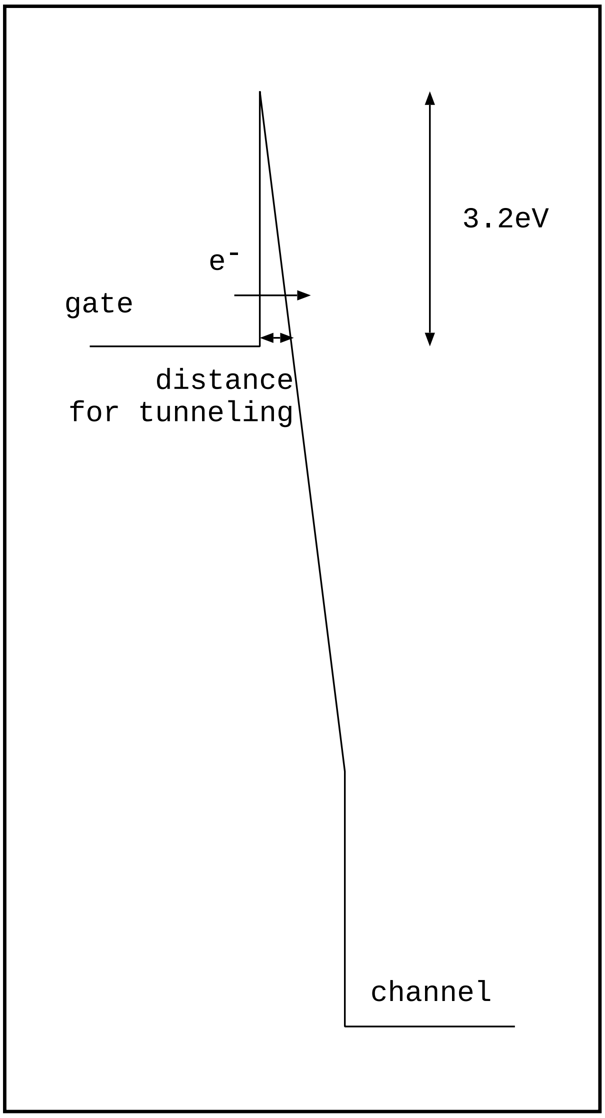


Figure 9.13: Band diagram for tunneling through the gate oxide

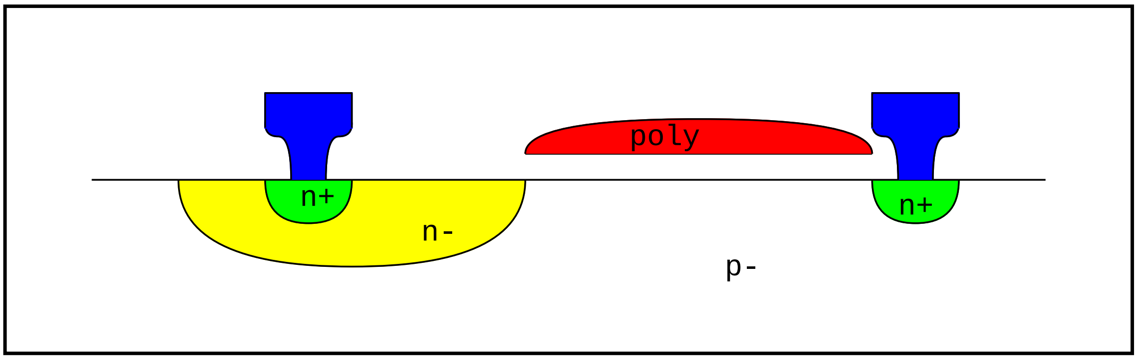


Figure 9.14: High voltage NFET

- Tunneling and hot carrier injection efficacy changes with use. Limited lifetime (write cycles in flash ROMs)
- Leaky in more advanced processes (thinner gate oxide ¡ 3nm)

9.3 Neuromorphic Learning Circuits

The most prominent learning algorithm in the context of simulated artificial neural networks, is the Error Backpropagation algorithm (gradient descent) used with multi layer Perceptrons (MLP). However, since it is a mathematical model, it does not lend itself to direct implementation into aVLSI. It is prone to all the weaknesses of realizing simple mathematics in analog hardware: Nonlinearities, offsets, bounded range, etc. The result: simple maths, complicated circuits.

In contrast, neuromorphic engineers try to use the building blocks of aVLSI (instead of maths) to define their model. The resulting circuits are thus comparatively simple, whereas a precise mathematical description of them would be complex. Still, qualitatively they are mostly Hebbian learning circuits, as this is a learning behaviour that is actually based in neurophysiological observations.

9.3.1 Hebbian learning circuits

An elegant weight normalizing learning circuit in aVLSI has been introduced by Chris Diorio [11, 12]. His group uses floating gate transistors to implement very compact learning systems with non-volatile analog storage (figure 9.16).

This implementation is strange in some ways. The learning rule itself is not part of the circuit: the signals controlling the learning are independent input signals that are superimposed on the input signals. The synapse input (X) is active high and can be analog continuous. One half of the learning input is active low pulses superimposed on this input signal. If an active low X learning input (lowering the gate voltage) meets with a Y active high pulse (increasing the tunnel voltage), tunneling to that particular floating gate is induced (because the difference between floating gate and tunneling node is big enough only in that synapse now) and the synaptic connection gets strengthened. To get a sort of Hebbian behaviour the Y inputs could be the same as or be closely correlated to the neuron's output and the active low learning inputs on the X inputs should be correlated with the input activity.

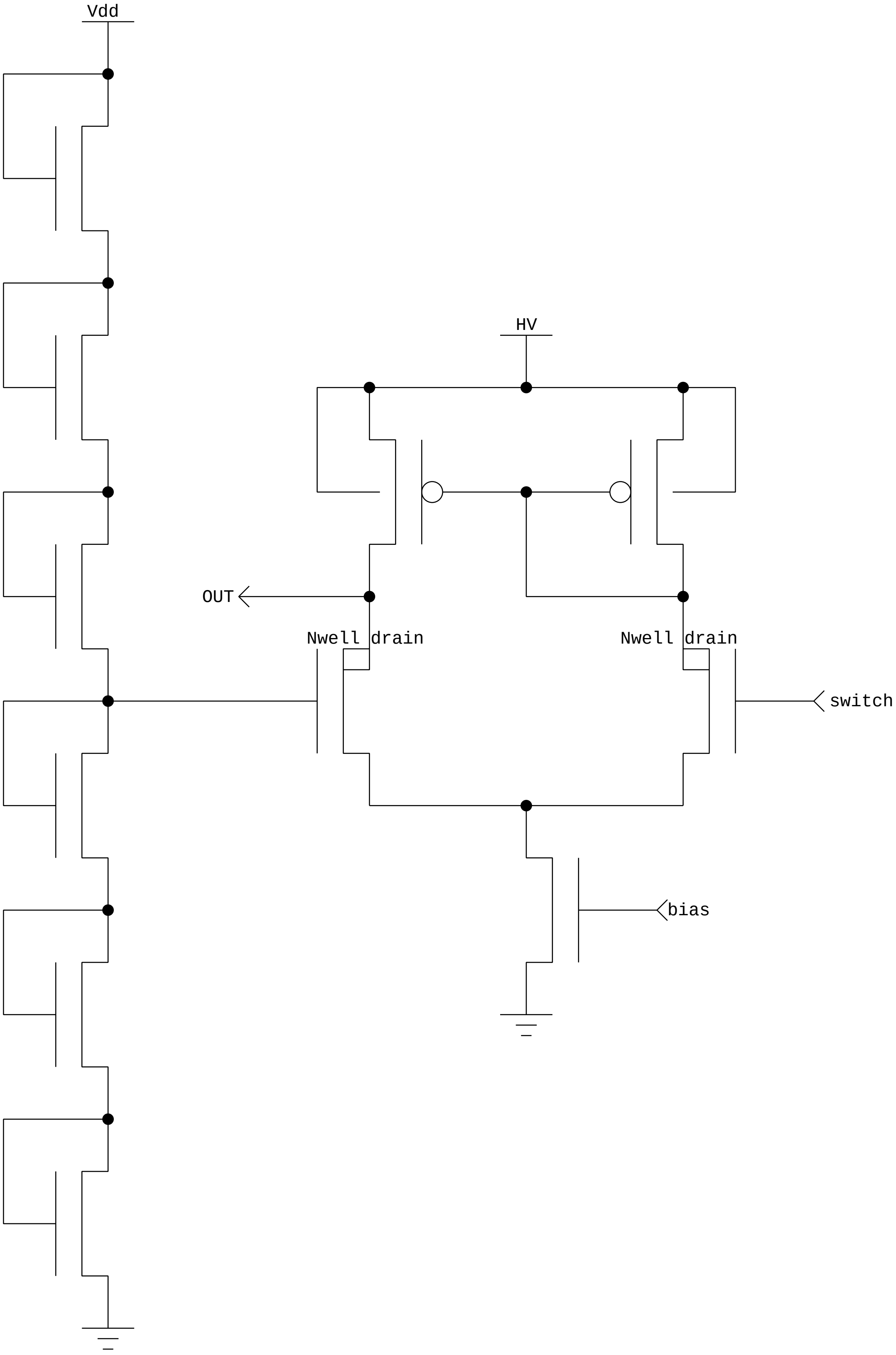


Figure 9.15: On chip high voltage switch

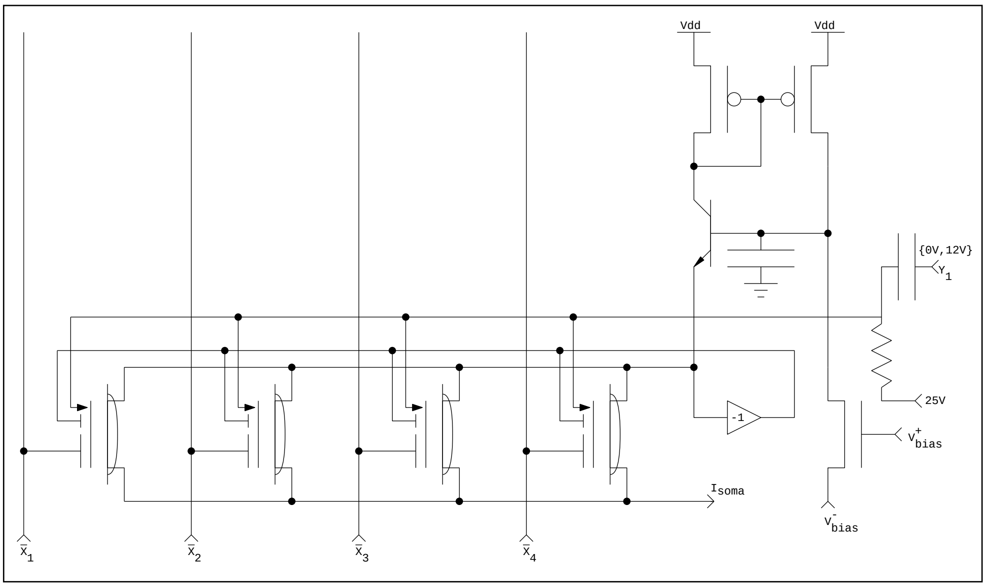


Figure 9.16: An explicitly weight normalizing circuit with off chip controlled learning behaviour [11, 12]

The normalization circuit on the right basically compares the total current of the synapses with a bias current. If it exceeds that bias current, the drain voltage to the synapse transistors is increased such that hot electron injection starts to lower all floating gates until the total current is equal to the bias current again. (Inverted capacitive feedback to the gates turns down the synapse at the same time to stabilize the current output during that readaption phase). This normalization is also dependent on the input because if any input causes a higher total output, it will also be activated. It limits the maximum of the total output current.

Another synaptic model has been introduced by Fusi etal.[15]. It is depicted in figure 9.17. It uses a kind of digital storage and the synapse has two states only. The transition between the two states depends on correlations in input activity and the membrane voltage.

The synapse is attached to an integrate-and-fire neuron. The signal 'MembV' is a digital signal resulting from a comparison of the membrane voltage with a threshold. If the membrane is above this threshold 'MembV' is low and vice versa. Thus whenever an 'InSpike' meets with a high membrane voltage, the voltage on the capacitor is increased and vice versa. As long as this voltage is above 'Thresh' it is slowly drifting towards Vdd. And the synapse is in the strong state: both branches supplying synaptic current, are open. If however, repeated input activity that meets with a low membrane potential drives the voltage on the capacitor below 'Thresh', then the voltage is slowly drifting towards Gnd and the synapse is in its weak state: the left branch supplying synaptic current is switched off. The overall behaviour is qualitatively Hebbian: correlation of inputs with high membrane voltages increase synaptic efficacy, whereas input activity that is not correlated with an excited neural state, decreases that efficacy.

9.3.2 A spike based learning circuit

The following circuits (block diagram in figure 9.18) presented in this subsection are based on the example learning rule mentioned in 9.1.4. There is a separate floating gate memory cells that

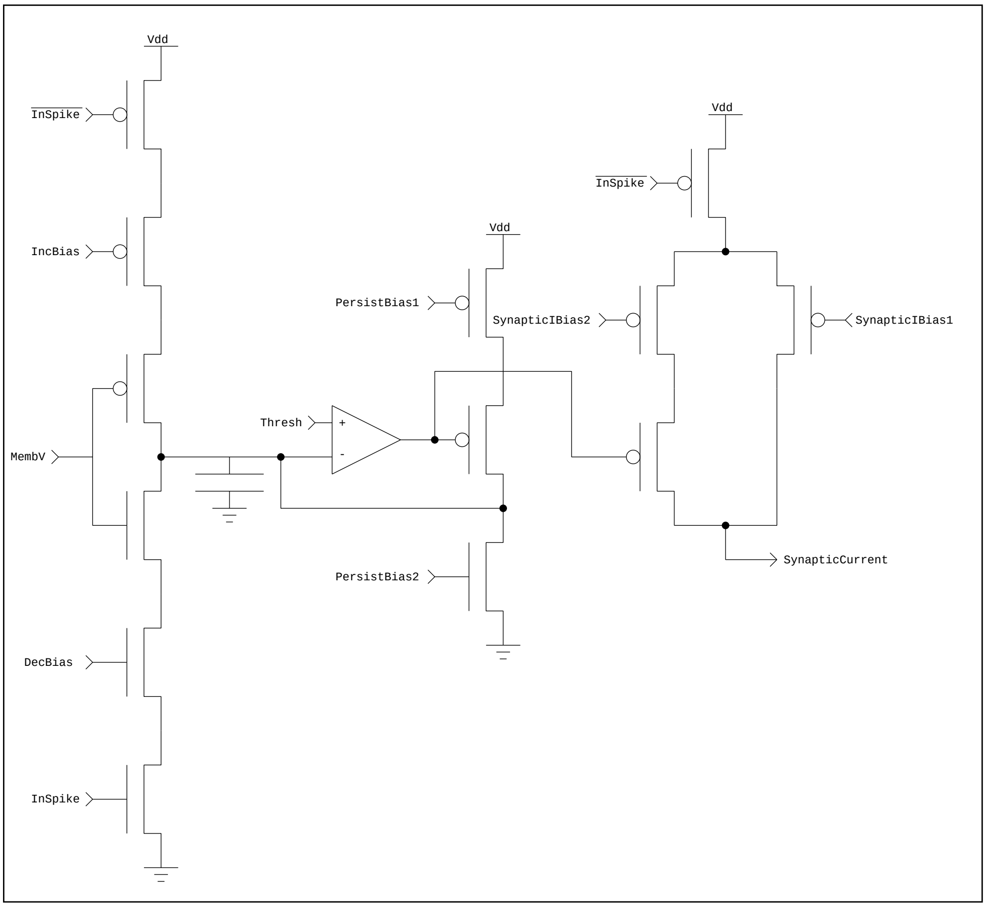


Figure 9.17: A bistable learning circuit [15]

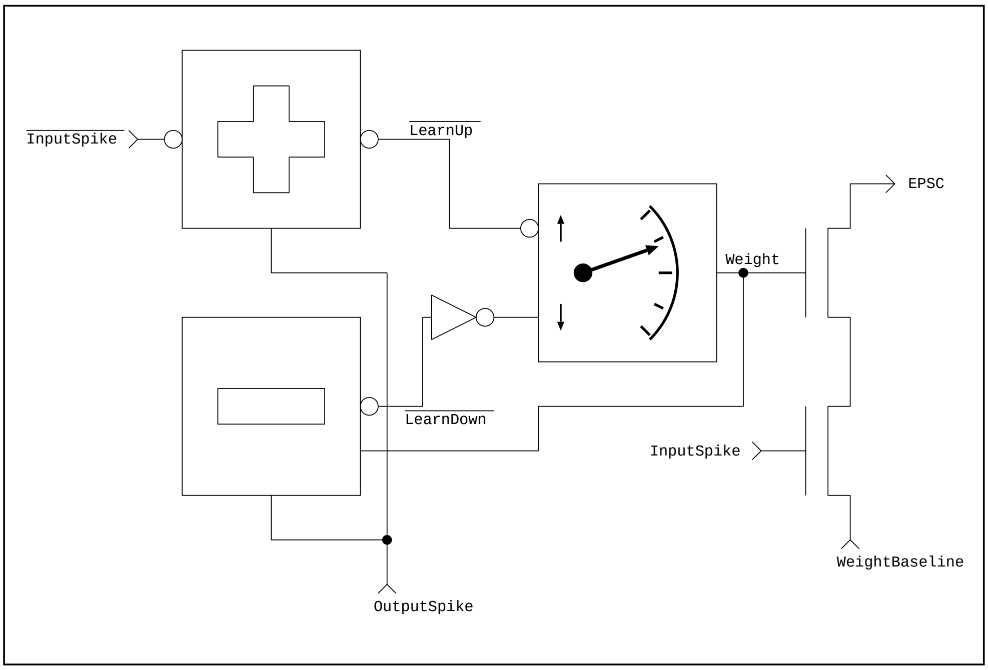

$\overline{\text{InputSpike}}$
$\overline{\text{LearnUp}}$
EPSC
Weight
$\overline{\text{LearnDown}}$
InputSpike
WeightBaseline
OutputSpike

Figure 9.18: Blockdiagram of an example of a spike based learning circuit [17]

stores an analog voltage and is controlled by digital input signals that move the content up or down (figure 9.21).

A 'learn up' block (figure 9.19) computes the length of a pulse according to the positive term in the weight update rule for every output spike (according to formula (9.10)). The voltage on the capacitance is incremented with every input spike and decays with a constant leakage current. With a output spike, the current comparator to the right and an additional leakage current from the capacitor are turned on. The current comparator issues a low voltage until the capacitor voltage falls below a given level (chosen as low as possible), thus the resulting active low pulse is roughly proportional to the voltage on the capacitor.

The 'learn down' circuit (figure 9.20) computes the pulse length for the negative term in the learning rule. The pulse length, and thus the magnitude of this negative term, like in formula (9.10) is dependent on the momentary weight. The basic circuit is a NAND that receives the output spike as a direct input and the inverse of the output spike as a delayed input. The delay of the falling flank of the inverted spike input determines the pulse duration. That delay is proportional to the inverse of the current that is used to discharge the capacitor. That current is exponentially dependent on the weight voltage that sits at the source of the current limiting transistor.

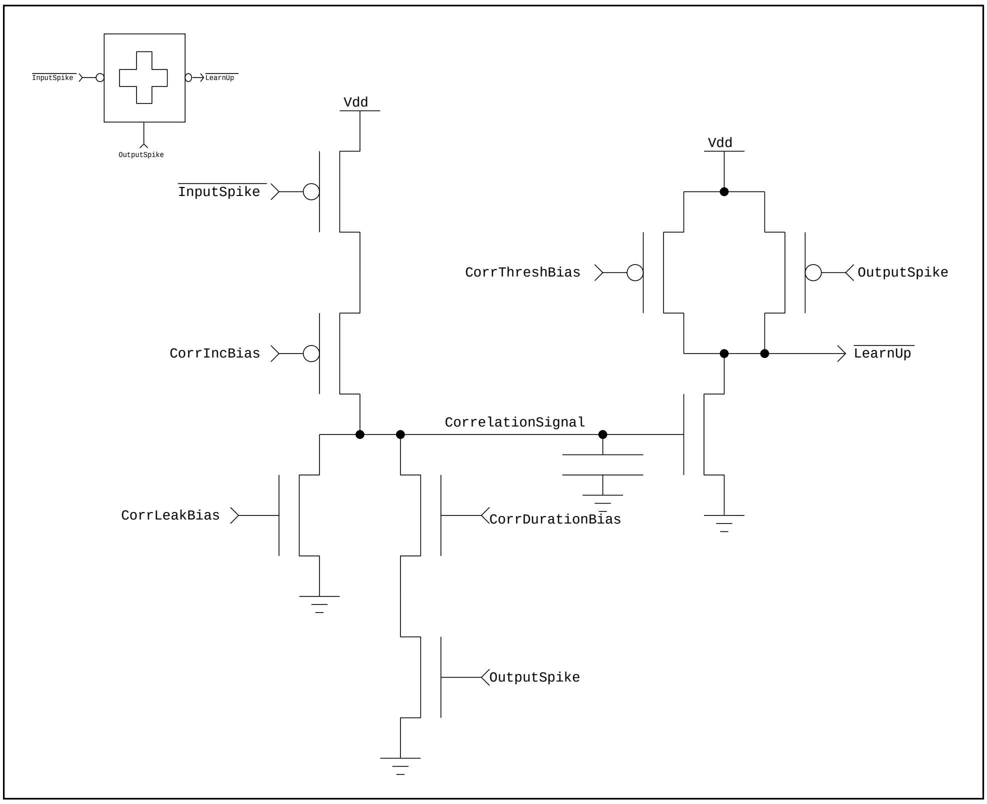


Figure 9.19: Circuit computing the positive term of the weight update

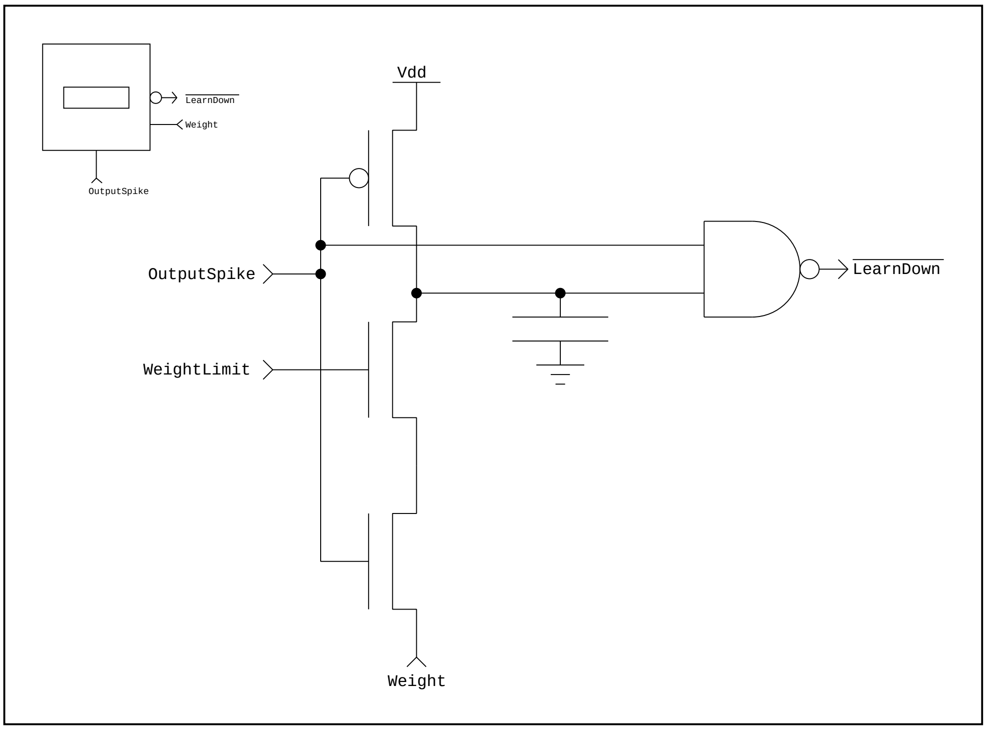


Figure 9.20: Circuit computing the negative term of the weight update rule.

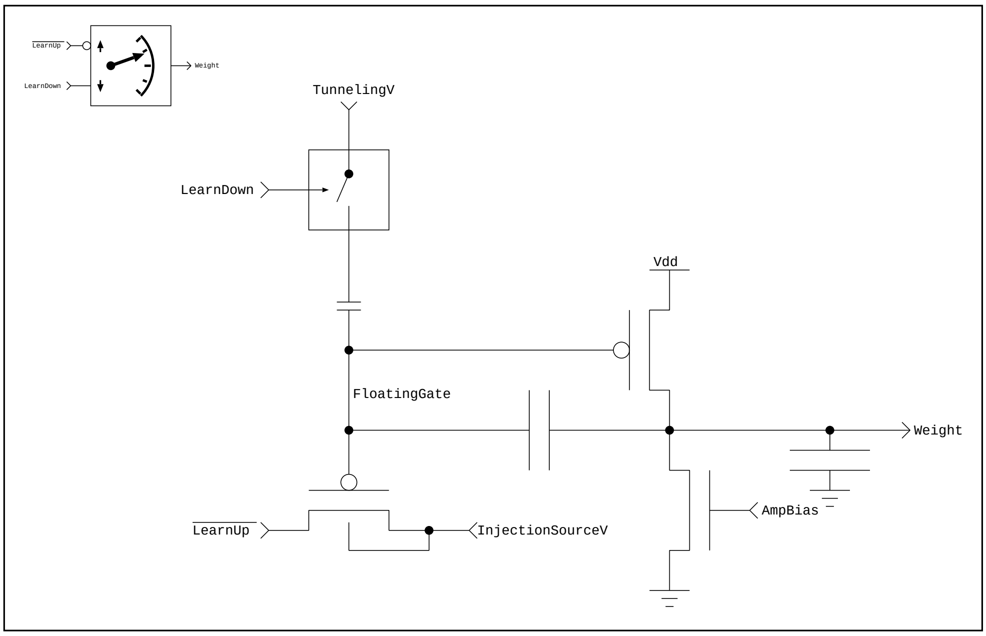


Figure 9.21: Floating gate analog storage cell

Bibliography

[1] M. Abeles. *Corticonics, Neural Circuits of the Cerebral Cortex.* Cambridge University Press, 1991.

[2] W. Bair and C. Koch. Temporal precision of spike trains in extrastriate cortex of the behaving macaque monkey. *Neural Computation*, 8:1185–1202, 1996.

[3] G. G. Blasdel. Differential imaging of ocular dominance columns and orientation selectivity in monkey striate cortex. *Journal of Neuroscience*, 12:3115–3138, 1992.

[4] G. G. Blasdel. Orientation selectivity, preference, and continuity in monkey striate cortex. *Journal of Neuroscience*, 12:3139–3161, 1992.

[5] G. G. Blasdel and G. Salama. Voltage-sensitive dyes reveal a modular organization in monkey striate cortex. *Nature*, 321:579–585, 1986.

[6] K. Boahen. A throughput-on-demand address-event transmitter for neuromorphic chips. In *Adv. Res. in VLSI*, pages 72–86. IEEE Comp. Soc. Press, 1999.

[7] K. A. Boahen. The retinomorphic approach: Pixel-parallel adaptive amplification, filtering, and quantization. In T. S. Lande, editor, *Neuromorphic Systems Engineering*, chapter 11, pages 129–150. Kluwer Academic Publishers, Boston, 1998.

[8] S. H. Cardoso. 1997. http://www.epub.org.br/cm/n02/mente/neurobiologia˙i.htm.

[9] T. Delbrück and C. Mead. An electronic photoreceptor sensitive to small changes in intensity. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1, pages 720–726, San Mateo CA, 1989. Morgan Kaufman.

[10] W. Denk. 2002. http://wbmo.mpimf-heidelberg.mpg.de/Biomedizinische.Optik.html.

[11] C. Diorio, P. Hasler, B. A. Minch, and C. Mead. A single-transistor silicon synapse. *IEEE Trans. Electron Devices*, 43(11):1972–1980, 1996.

[12] C. Diorio, P. Hasler, B. A. Minch, and C. Mead. A floating-gate MOS learning array with locally computed weight updates. *IEEE Transactions on Electron Devices*, 44(12):2281–2289, December 1997.

[13] C. Diorio, S. Mahajan, P. Hasler, B. Minch, and C. Mead. A high-resolution non-volatile analog memory cell. *Proc. IEEE Intl. Symp. on Circuits and Systems*, 3:2233–2236, 1995.

[14] R.H. Fowler and L. Nordheim. In *Proc. Roy. Soc. Lond.*, number 173 in A119, 1928.

[15] S. Fusi, M. Annunziato, D. Badoni, A. Salamon, and D.J.Amit. Spike-driven synaptic plasticity: theory, simulation, VLSI implementation. *Neural Computation*, 12:2227–2258, 2000.

[16] S. Grossberg, E. Mingolla, and J. Williamson. Synthetic aperture radar processing by a multiple scale neural system for boundary and surface representation. *Neural Networks*, 8(7/8):1005–1028, 1995.

[17] P. Häfliger. *A spike based learning rule and its implementation in analog hardware.* PhD thesis, ETH Zürich, Switzerland, 2000. http://www.ifi.uio.no/~hafliger.

[18] P. Häfliger and H. Kolle Riis. A multi-level static memory cell. In *Proc. of IEEE ISCAS*, volume 1, pages 22–25, Bangkok, Thailand, May 2003.

[19] D. O. Hebb. page 62. Wiley, New York, 1949.

[20] P. Heim and M. A. Jabri. Long-term CMOS static storage cell performing AD/DA conversion for analogue neural network implementations. *Electronic Letters*, 30(25), December 1994.

[21] A. E. Hernandez, A. Martinez, E. C. Wong, L. R. Frank, and R. B. Buxton. Bilingual research projects, fMRI. 1999. http://crl.ucsd.edu/bilingual/fMRI3.html.

[22] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation.* Addison Wesley, 1991.

[23] A. L. Hodgkin and A. F. Huxley. The components of membrane conductance in the giant axon of loglio. *Journal of Physiology*, 116:473, 1952.

[24] A. L. Hodgkin and A. F. Huxley. Current carried by sodium and potassium ions through the membrane of the giant axon of loglio. *Journal of Physiology*, 116:449, 1952.

[25] M. Holler, S. Tam, H. Castro, and R. Benson. An electrically trainable artificial neural network (ETANN) with 10240 'floating gate' synapses. *International Joint Conference on Neural Networks*, (II):191–196, June 1989.

[26] J.C. Horton and D.R. Hocking. Intrinsic variability of ocular dominance column periodicity in normal macaque monkeys. *Journal of Neuroscience*, 16:7228–7239, 1996.

[27] D. Hubel and T. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962.

[28] D. Kahng and S. M. Sze. A floating gate and its application to memory devices. *The Bell System technical Journal*, pages 1288–1295, July/August 1967.

[29] Z. Kalayjian, J. Waskiewicz, D. Yochelson, and A. G. Andreou. Asynchronous sampling of 2d arrays using winner-takes-all arbitration. In *ISCAS*, volume 3, pages 393–396, 1996.

[30] Tuevo Kohonen. *Self-Organization and Associative Memory*, page 99. Springer, Berlin, 1984.

[31] J. Kramer, R. Sarpeshkar, and C. Koch. Pulse-based analog VLSI velocity sensors. *IEEE Trans. on Circ. and Sys.-II*, 44(2):86–100, February 1997.

[32] J. Lazzaro and C. A. Mead. Circuit models of sensory transduction in the cochlea. In C. A. Mead and M. Ismail, editors, *Analog VLSI Implementations of Neural Networks*, pages 85–101. Kluwer Academic Press, 1989.

[33] M. Mahowald. *An Analog VLSI System for Stereoscopic Vision.* Kluwer, 1994.

[34] M. Mahowald and R. Douglas. A silicon neuron. *Nature*, 354:515–518, 1991.

[35] J. T. Marienborg and T. S. Lande. Analog state transmission with digital hardware. In *NORCHIP*, pages 249–256, 1998.

[36] H. Markram, J. Lübke, M. Frotscher, and B. Sakmann. Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. *Science*, 275:213–215, 1997.

[37] M. E. McCourt. Central visual pathways. 1997. http://www.psychology.psych.ndsu.nodak.edu/mccourt/website/htdocs/HomeP%age/Psy460/Central%20visual%20pathways/Central%20Visual%20Pathways.html.

[38] C. Mead. *Analog VLSI and Neural Systems.* Addison Wesley, 1989.

[39] A. Mortara and E. A. Vittoz. A communication architecture tailored for analog VLSI artificial neural networks: intrinsic performance and limitations. *IEEE Trans. on Neural Networks*, 5:459–466, 1994.

[40] F. H. Netter. *Atlas of Human Anatomy.* Ciba-Geigy Corp., Arsley, USA, 1989.

[41] J. Nolte. *The Human Brain.* Mosby, 1988.

[42] J. O'Keefe and M.L. Recce. Phase relationship between hippocampal place units and the eeg theta rhythm. *Hippocampus*, 3(3):317–330, July 1993.

[43] C. G. Phillips and R. Porter. *Cortico-Spinal Neurones: Their Role in Movement.* Academic Press, London, 1977.

[44] P. O. Pouliquen and A. G. Andreou. Bit-serial address-event representation. *Proceedings of Conference on Information Sciences and Systems*, March 1999.

[45] C. Rasche, R. Douglas, and M. Mahowald. Characterization of a silicon pyramidal neuron. In L. S. Smith and A. Hamilton, editors, *Neuromorphic Systems: Engineering Silicon from Neurobiology*, chapter 14, pages 169–177. World Scientific, 1998.

[46] H. Kolle Riis and P. Häfliger. Spike based learning with weak multi-level static memory. In *Proc. of IEEE ISCAS*, volume V, pages 393–395, Vancouver, Canada, May 2004.

[47] D. Rumelhart, G. Hinton, and R. Williams. *Parallel Distributed Processing*, chapter 8: Learning internal representations by error propagation, pages 319–362. MIT Press, Cambridge, MA, 1986.

[48] S. Sands and M. Pflieger. 1999. http://www.neuro.com.

[49] R. Sarpeshkar, R. F. Lyon, and C. Mead. A low-power wide-dynamic-range analog VLSI cochlea. In T. S. Lande, editor, *Neuromorphic Systems Engineering*, chapter 3, pages 49–103. Kluwer Academic Publishers, Boston, 1998.

[50] T. Serrano-Gotarredona, A. G. Andreou, and B. Linares-Barranco. AER image filtering architecture for vision-processing systems. *IEEE Transactions on Circuits and Systems*, 46(9):1064–1071, 1999.

[51] G. M. Shepherd. *Neurobiology.* Oxford University Press, 3 edition, 1994.

[52] A. Stocker and R. Douglas. Computation of smooth optical flow in a feedback connected analog network. In *Advances in Neural Information Processing Systems (NIPS)*, volume 11, pages 706–712, 1999.

[53] C. Stricker and A. Cowan. Private communication. 2002. .

[54] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3:9–44, 1988.

[55] G. Tesauro. Practical issues in temporal difference learning. *Machine learning*, 8:257–277, 1992.

[56] D. C. van Essen, C. H. Anderson, and D. J. Felleman. Information processing in the primate visual system: An integrated systems perspective. *Science*, 255:419–422, 1992.

[57] E. Vittoz. Analog VLSI signal processing: Why, where and how? *Analog Integrated Circuits and Signal Processing*, pages 27–44, July 1994.