TR 3206 S

Stellingen

behorende bij het proefschrift

Analog Neural Networks in Single-Electron Tunneling Technology

van Martijn Goossens

1. Een compacte realisatie van neurale primitieve functies is mogelijk door de fysische eigenschappen van de basisdevices expliciet te benutten.
2. Voor het ontwerp van grote SET-transistorcircuits is het gevolg van offsetladingen een belangrijker aandachtspunt dan vermogensdissipatie.
3. Een neuron en een synaps kunnen elk met één SET-transistor worden gemaakt.
4. Het streven een krachtige signaalprocessor te ontwikkelen moet niet verward worden met het streven biologische neurale netwerken na te maken.
5. Een vergelijking met kleine biologische neurale netwerken leert dat het gebrek aan compacte implementaties van artificiële neurale netwerken niet noodzakelijkerwijs de oorzaak is van het gebrek aan succesvolle toepassingen ervan.
6. Van het in de pers vermelden van de straatwaarde van in beslag genomen verdovende middelen gaat in ieder geval geen preventieve werking uit.
7. Het gebruik van bestrijdingsmiddelen in de druiventeelt heeft misschien wel een grotere invloed op de smaak van de wijn dan de exacte locatie van de wijngaard.
8. Het gezondheidseffect van een ergonomisch toetsenbord is voor een belangrijk deel te danken aan de bijgevoegde handleiding over een goede lichaamshouding.
9. Overmatig acroniemgebruik is een vorm van cryptografie.
10. Het feit dat bij een extra hoge jackpot veel mensen incidenteel meedoen aan een loterij, maakt het spel voor de trouwe speler extra onvoordelig.

11. Zolang op de weerkaarten van veel Europese televisiezenders de omringende landen niet eens getoond worden, blijft Europa een verzameling losse eilanden.

12. De uitspraak "Geniet *maar* drink met mate" suggereert ten onrechte dat genieten normaliter samengaat met overmatig drankgebruik.

13. In tegenstelling tot wat vaak gesuggereerd wordt is het milieubeleid vaak slechts gedreven door de wens het meest storende element uit de natuur te beschermen: de Mens.

14. Het weer is zo goed als je kleding.

15. Een wekker is een blikopener.

16. In mei legt hooguit de helft van alle vogeltjes een ei.

TR 3206

Analog Neural Networks in Single-Electron Tunneling Technology

Published and distributed by:

Delft University Press
Mekelweg 4
2628 CD Delft
The Netherlands
Telephone: +31 15 2783254
Fax: +31 15 2781661
E-mail: DUP@DUP.TUDelft.NL

ISBN 90-407-1721-4 / CIP
NUGI 832

Contents

1 Introduction

One of the great challenges of today's electronic designers is to make more powerful signal processors for even more complex tasks. When we compare the performance of state of the art electronic signal processors with the smallest biological systems however, we see an enormous gap to bridge. In fields like image processing we have a long way to go before biological systems will be outperformed.

The successful increase in the processing power of digital systems is largely due to the incredible reliability and speed increase of the hardware, mainly as a result of shrinking device dimensions. When those dimensions however, shrink below the limits beyond which quantum physics rules, reliability and speed will cease to scale with size.

Biological systems seem to circumvent precisely those factors that threaten the future expansion of digital processing power. Biological systems work relatively slowly and the building blocks are certainly not fully reliable. The total system however does have a high processing power. This holds for our ultimate example, the brain, but also for an ant's nest, for example. The synergy of thousands of small insects does not depend on the failing of an individual. Cooperation between individual ants, each of which performs only simple and basic tasks, results in the successful completion of relatively complex tasks such as nest building, defense against predators and food supply. The total 'power' of such a biological system basically depends on its total size.

The artificial neural network concept is an attempt to follow this idea. A neural network withstands inaccurate and malfunctioning hardware because it is inherently redundant and can adapt to changing circumstances. Provided the network is large, it can evaluate complex tasks quickly.

The problem encountered here is to make neural networks whose size is in some way comparable to the size of biological examples. This is desired because it is large size that results in high processing power.

This thesis investigates whether it is in principle possible to make neural networks with Single Electron Tunneling (SET) technology, and explores ways to optimally exploit the characterististic technological SET properties for extremely compact hardware solutions. Relying on the ability of neural networks to cope with unconventional device functions, this thesis investigates whether the SET transistor can be employed as an elementary neural network building block.

In Chapter 2, the field of neural networks is addressed. It explains why the neural system concept is valuable for powerful signal processors, and what problems hamper the realization of large systems. Chapter 3 describes the design strategy for compact building blocks of large neural networks. The neural devices described in this thesis were designed according to this strategy.

The Single Electron Tunneling transistor is the elementary device on which the neural devices presented in this work are based. Chapter 4 introduces the SET transistor. It describes the properties and how they can be used to design neural building blocks.

One of the basic cells of a neural network is the neuron. In Chapter 5, two neuron designs in SET technology are presented. One with two SET transistors, and one consisting of a single device. It is argued that the latter has the most promising properties for a compact SET neural network. The other basic cell is the synapse, which is the subject of Chapter 6. It is shown that basic synaptic functionality can be obtained with a single SET transistor.

The issue of connecting the neural cells into a network is analyzed in Chapter 7. It is shown that a small SET network can perform the elementary classification tasks.

The learning algorithm is also an essential ingredient of any neural network but it is not extensively discussed in this thesis. With the aid of a two-layer neural network, Chapter 8 describes how the random weight change learning algorithm adapts the SET neuron and synapses to perform a predefined task.

As a result of technological complications, no measurements were obtained from the SET devices that were fabricated for this project (see the micrographs in Chapters 4 and 5). The similarity however, between measurements performed on individual SET transistors known in the literature [1, 2] and the simulation results presented in this thesis, give confidence about their reliability. All the numerical simulations of SET transistors in this thesis were performed with the simulation program 'SIMON' (Simulation of Nano structures). The parameters used to obtain the data are given in Appendix A.

2 Overview

During the past years, we have witnessed a spectacular performance increase of powerful signal processing machines, and in coming years, this will no doubt continue. At some stage however, predicted by the road map [3], the growth of speed and size will come to a grinding halt where the laws of physics meet. To continue from there, we will have to resort to alternative methods [4]. It will no longer be possible to scale down device size and scale up its operation frequency.

Doing things in parallel is faster than doing everything consecutively. Despite the low frequencies in for example the human brain, we have no difficulty processing large quantities of data at a staggering speed. The advantages of human-made parallel processing are already being shown by for example multiprocessor arrays, all solving part of one complicated problem. It is however a great challenge to develop efficient software that exploits the potential processing power of these systems.

A more natural way of processing information is the inherently parallel manner of a neural network. The advantage of a parallel system is that it can be made faster simply by increasing its size. More processing nodes in principle means more processing power.

In Section 2.1, neural networks are compared to other systems. Even though neural networks have advantages in some fields, realizing one that actually shows these advantages remains a great challenge, as described in Section 2.2. Section 2.3 describes what we propose to face the challenge: devices that perform the neural primitive functions.

2.1 Neural networks

An artificial neural network is a machine composed of a large number of similar and interconnected cells consisting of a small set of mostly non-linear and adaptable building blocks. The network operates similarly to the way we believe biological brains work. It is an attractive solution for various kinds of signal processing tasks. The most appealing is probably that a neural network can proficiently solve the type of problems digital computers traditionally have difficulties with, such as recognizing various kinds of patterns, such as speech, handwriting, and faces. Areas that machines have not yet surpassed humans in the way mathematical and database functions have during the past decades.

Some like to call these functions typical 'human' functions, and are afraid that successful artificial neural networks might in the near future take over the world from mankind, as skillfully suggested in some films. Similar fears existed on the widespread introduction of the electronic calculators in the seventies. When my father had just bought the first pocket calculator for the laboratory he was working for at that moment, and he and his colleagues were watching it taking a few moments to evaluate its first sin 60, one of his colleagues exclaimed: "It's thinking!". Until then, algebra was widely seen as a human task. Now, thirty years later, it certainly is no longer the case.

One can indeed say that digital calculators and computers have conquered the world since then, but certainly not *from* mankind. Perhaps *with* mankind is a better description.

There are more differences than similarities between artificial neural networks and biological brains. Those functions that are now hesitantly tried by these new types of machines and that up to now exclusively belonged to the realm of living beings are called 'human' simply because up to now only humans were adept at solving them. A machine however, that can recognize handwriting still cannot *think*, even if it takes a while to evaluate a result. It is just like a powerful computer with a fancy program that combines the rules of chess with a database containing all the important matches that have been played so far. Such a computer can win the game from the human world champion, but that does not make it intelligent, contrary to what some reports would have us believe.

There is an important difference between digital computers and neural networks though, that does stress the similarity between an artificial neural network and biological brains. Whereas the behavior of a digital computer is completely deterministic and predictable, the way a neural network or the brain solves problems is not. Although we are witnessing spectacular improvements in both power consumption and defect reduction, it still is a major limitation for expanding the signal processing performance of digital systems. The typical characteristics of neural networks, such as their non-linear and adaptive operation makes them more suitable for very compact and low power implementations because inevitable errors occurring in the hardware can be dealt with.

Digital systems have only inherent redundancy at the signal representation level. This makes them robust to signal errors. Neural networks have redundancy at the processing cell level. This makes it robust to 'system noise', i.e. malfunctioning functional blocks. Therefore, the neural network system concept is promising for large systems with inaccurate hardware.

Neural networks are suitable to help us solve certain types of problems such as the recognition of patterns, and could become of increasing importance if an efficient way of manufacturing them becomes available.

2.2 Large neural networks

To be a powerful signal processor, a neural network should be large. Among other reasons however, the absence of a good implementation method still hampers the realization and the adoption of such networks in our daily lives.

Different ways to build electronic neural networks are described in Section 2.2.1, and Section 2.2.2 argues that the primitive functions of the network should consist of elementary devices to obtain a high integration density.

2.2.1 Making neural networks

Many attempts are being made by many groups throughout the world to make successful neural networks, and with remarkable successes, mainly enhancing the understanding of neural networks and underlining their potential importance, and sometimes showing this importance with a real application. Except for interesting developments in artificial biological processors [5], and in optical neural networks [6], the three main electronic directions for implementing artificial neural networks are: emulating the system on a digital computer, dedicated digital hardware, and analog hardware. All three are discussed briefly below.

Emulating neural networks

The most flexible way to implement a neural network is with software on a powerful computer. The importance is twofold: to gain understanding of neural network behavior such as in the difficult field of learning, and for commercial applications such as handwriting recognition for personal organizers [7] and the control of complicated chemical plants. These applications are typical examples of very complex tasks that are difficult to describe exactly, but can be solved by a neural network. For many applications, the main disadvantage of software neural network implementations is their low speed and limited network size. Software networks are usually evaluated sequentially per node instead of in parallel, simply because the computer only contains a single processor.

Integrated digital neural networks

To solve this lack of parallelism, integrated digital neural chips have been developed that have a small processor for each neuron [8]. Although these systems are certainly very much faster than their software counterparts, they still have important disadvantages.

Discretizing the signal levels reduces the processing power, and results in learning problems. The power consumption and size of these systems limit their applicability. It is illustrative to compare it with digital computers with tubes, which in the fifties and sixties filled whole rooms with what we would now call modest computing power. These machines incontestably showed that the system concept was very promising, but that it would not mature in this technology. The success of digital computers since the introduction of semiconductor chips, proved both.

Integrated analog neural networks

A more promising approach for making neural networks is the analog electronic implementation. By exploiting the full range of signal values, it allows neural cells to be implemented with just a 'handful' of components, yielding more processing power with the same area and power consumption. The unavoidably resulting errors can be handled by the neural system. The usefulness of the neural network concept is clearly apparent from the unprecedented processing power of only small networks, with relatively few neural cells. Classifying problems that were up to now difficult or even impossible to solve, like those described in [9], can be solved by an analog neural network.

2.2.2 Larger neural networks

In comparison to biological brains, these successful network implementations still have disappointingly few neurons. In fact, we have barely scraped the surface of a whole new undiscovered field of interesting signal processing machines. The human brain contains about 10^{11} neural cells, which is many orders of magnitude larger than the 10^4 cells that we currently call state of the art.

The door to a successful new system concept is there, but to open it, we need the technology that fits.

In such a technology, tiny devices with useful characteristics must be available. The intrinsic properties of this technology must be optimally exploited to obtain compact hardware structures that match the primitive functions of the basic neural building blocks. We will call those compact hardware structures *neural devices.*

2.3 Neural devices

To make very large neural networks, compact hardware structures, or neural devices, which perform the neural primitive functions are the essential ingredient.

Any design is always limited by at least two boundary conditions. The available properties of the technology on one side, and the desired functionality on the other. With regular electronic circuit designs, the available technological properties can be manipulated by finding the circuit configuration that best meets the functional requirements.

When restricting the design to a neural device, consisting of a small circuit, or in the limit of a single technological device, the hardware properties are more or less dictated by the properties of the chosen technology, while the basic properties of this technology are fixed by the laws of physics. If the desired functional properties were also strictly defined, the possibilities for the design of a neural device that exactly fits those functional properties would be too restricted.

Fortunately, in the case of neural networks there is flexibility in the functional requirements. The network's adaptability can namely also be applied to make it cope with diverging types of primitive function variations, in addition to its ability to adjustment at the system function level. There is no list of strictly predefined requirements the primitive functions have to comply with.

This opens a whole new field of design strategies where the designed function is inspired by the requirements on the functional level, but largely determined by the possibilities on the technological level.

Many of the physical properties of semiconductor material have not been thoroughly investigated. The possibilities with the familiar transistor which was discovered half a century ago seem endless. Moreover, many of the alternative devices that have been invented [10, 11] have not found large scale application yet. In addition, many other technologies exist, so if we cannot find what we want in standard silicon technology, alternative technologies may offer a solution. This has been recognized by various research groups [11–15].

In this thesis, Single-Electron Tunneling (SET) transistors are used. These tiny devices have very interesting properties. They are extremely small and low power, which is in itself important, and they have an interesting but exotic electronic behavior. Most important are the periodic transfer function and the offset charges. The SET transistor is described in detail in Chapter 4.

2.4 Conclusions

Neural processing promises to become a successful signal processing system concept for very high volume processing. Its success however, stagnates because no hardware solution is available. The main challenge is to design compact neural cells to implement very large neural networks.

By designing compact *neural devices* that efficiently exploit the inherent technological properties, and that implement the neural primitive functions, the compactness can significantly be improved. Instead of adapting the device's properties to the functional neural requirements, the flexibility of the neural network is used to adapt to the device properties.

In this thesis, it is shown that this flexibility, which has up to now barely been used, could be the key to the success of compact neural network implementations.

3 Design Philosophy

To be powerful signal processors, neural networks should be large. Much larger than the state of the art networks that can be realized today. There are two main aspects that limit the size of networks. The area and power consumption of the neural building blocks is too high to build much larger networks, and the learning behavior of such very large on-chip networks is not known well enough to make these networks operate sensibly.

This thesis concentrates mainly on the first issue: how the neural building block size and power consumption can be effectively reduced. Although the learning algorithm and how it could be implemented compactly is discussed, specific learning strategies for very large neural networks are not addressed.

In this chapter, the design philosophy for the ultimately compact neural building blocks is described. The functional properties of the basic neural building blocks are called the primitive functions. The compactness of the primitive functions is the primary factor determining the network's size. Section 3.1 demonstrates that the primitive function size can be minimized by dealing with inaccuracies at the neural network level instead of at the device or circuit level.

Section 3.2 shows that the learning algorithm of a very large neural network should be embedded on-chip with the rest of the system as much as possible, and therefore also be very compact. The implications this has for the type of learning algorithm are discussed.

The representation of the neural information signals has an important influence on the compactness and power consumption of the networks building blocks. Section 3.3 argues that an analog continuous-time signal is the most compact way to represent the information. The sensitivity to systematic errors does not hamper its application in neural networks because adaptiveness at the neural system level offers enough robustness to cope with it.

Finally, the role of the network topology in determining the neural network compactness is analyzed in Section 3.4. It is argued that the area occupied by the interconnections should be minimized by using a locally connected topology.

3.1 Neural primitive functions

A neural network only has a small set of primitive functions, which are repeatedly used to build a large network. In traditional neural network theory the neural primitive functions are described by well-defined mathematical functions, as explained in Section 3.1.1. To understand neural network behavior using mathematical descriptions and simulation software, those standard descriptions are indispensable. As we will see in Section 3.1.2, it is difficult and not necessary to map those descriptions exactly onto compact hardware realizations. Since an important characteristic of the neural network concept is its tolerance to different types of primitive functions, deviations from the 'ideal' primitive functions in favor of compactness are tolerable. Section 3.1.3 describes the elementary requirements with which the primitive functions should comply in order to be of any use for a neural network. In Chapter 2, a compact realization of neural primitive function was called a neural device. In Section 3.1.4, the desired properties of such devices for successful application in a neural network are listed.

3.1.1 The conventional neural primitive functions

Two types of neural primitive functions can be distinguished: evaluation functions and learning functions.

The evaluation functions

The evaluation functions of a neural network are used to process the useful information flow, to do what the network is intended for. The evaluation functions are contained in two types of building blocks: the neuron and the synapse.

The neuron

The conventional neuron performs a classifying function on the sum of its inputs. Two primitive functions can be defined here, namely the addition function, and the classifying activation function. The conventional mathematical description of the activation function is a step function or a sigmoidally shaped function as a smoother variant. The threshold value, defining the boundary between the classes, is sometimes adjustable to modify the classifying behavior. To summarize, we have the following functions:

- addition,

- activation function (step, sigmoid).

The synapse

The task of the synapse is to weigh the connection strength from a network input to a neuron, or between two neurons. Adjustment of the synaptic weights alters the network behavior. This function is mathematically represented by the multiplier, where the input is multiplied by the weight to obtain the output. So, the synapse has the following functions:

- multiplier,
- weight storage.

The learning functions

The second type of neural primitive functions is used to change the networks adjustment points so that it actually produces the desired results. The particular functions used are described by the learning algorithm. Following this algorithm, the synaptic weights and neuron offsets are adjusted to reduce the error between the network output and the desired output value. The definition of specific primitive learning functions of course depends on the specific learning algorithm, of which diverging types exist. Possibly, new learning schemes have to be devised to deal with the properties of specific implementations or applications. Therefore, no detailed descriptions of the learning primitives are given here.

3.1.2 Compact neural primitive functions

The advantage of using the standard mathematical descriptions of the neural primitive functions is that it makes the network easier to understand and analyze. The relatively simple functions allow us to calculate and predict the behavior under various circumstances, and to optimize the learning algorithm or the topology for better performance. The success of that work encouraged circuit designers to implement the required functions electronically to fully exploit from the parallel structure [16]. It was then noticed that with the learning schemes developed for these networks, the performance is sensitive to deviations from the ideal functional description [17]. The subsequent need for accurate implementations of the required mathematical functions resulted in large circuits: in the case of digital systems, many bits are required, and in the case of analog systems, efforts to compensate for distortions and non-linearities yielded relatively complex circuits for the individual primitive functions [16]. This need for accuracy is remarkable because the neural network concept is in principle robust to inaccuracies.

The desire to predict in detail the network behavior is necessary to gain understanding of neural networks, and as such indispensable for the development of powerful neural systems. The functioning of very large neural networks however, relies on its self-adjustment qualities, and predictability is contradictory to self adjustment. The network should adjust itself, for example to systematic errors such as constant but random offsets, precisely because we cannot possibly do that complex task by hand.

Moreover, the desired functionality is often not exactly known beforehand. Neural networks are often employed in situations where an accurate function specification is impossible because the precise a priori knowledge of the large and complex system concerned is impossible to acquire. In such a case, not the functionality, but the learning behavior to attain this functionality is programmed.

If complete predictability were maintained, the learning process would not be significantly different from deterministically programming the precise function of a digital computer.

To obtain more compact neural primitive functions, the mathematical descriptions that were necessary for predictability should not be adhered to too strictly. Simpler primitive function implementations can be more compact, and that makes larger neural networks feasible.

In the recent past we have seen hesitant steps towards this approach by allowing non-linearities and offsets in the synapse transfer function [17–19]. The most effective learning algorithms today, such as back propagation, rely on accurately known primitive functions. Relaxing the strictness of the primitive function specification must therefore go hand in hand with learning algorithms that support this relaxation.

3.1.3 Elementary primitive function requirements

Before designing compact neural primitive functions that do not strictly comply to the standard mathematical models mentioned in Section 3.1.1, it is necessary to describe the elementary properties the functions do need. These elementary functions are listed below.

The neuron

The neuron collects the signals at its inputs, and performs a classifying function on the combined result. We thus have:

- gather input signals,
- classify the result.

Classifying means that for a certain range of input values, the neuron output should be 'active'. The transition boundary from active to non-active (or vice-versa) is called the threshold value of that neuron. It is useful if the threshold is adjustable. The neuron is described in detail in Chapter 5.

The synapse

The connection strength from a network input to a neuron or between two neurons is given by the synapse. This gives the following list:

- modify connection strength,
- store weight.

The most versatile synapse can have both positive and negative connection strengths. The synaptic weight stores the current weight of the connection. The synapse is analyzed in Chapter 6.

The learning algorithm

The learning algorithm is in charge of adjusting the synaptic weight values and the neuron threshold values of the whole neural network so that the network produces the desired output. The following functions can be distinguished:

- weight adjustment function,
- error signal generation,
- linking the above.

Depending on the type of learning algorithm, an error signal is generated from the internal network signals (unsupervised learning), or by comparing the network output with the desired network output generated by a supervising controller (supervised learning). The learning algorithm describes how the error signal is generated and how the weight updates depend on the error signal. This is the subject of Chapter 8.

The way these elementary functions are implemented depends on the hardware properties. The neural network should be capable of adjusting itself to the specific implementation dictated by the technology.

3.1.4 Neural devices

As argued in Chapter 2, the most compact realization of a primitive function uses a technologically compact hardware structure called a *neural device.* The properties of those devices however, cannot be changed very much, so that exactly specified functional behavior cannot be obtained. Therefore, flexibility is required at the system level to use the available device properties. In this section the basic requirements for neural primitive functions are described. The elementary primitive functions description given in the previous section of course forms the basic ingredient of potential neural devices.

- The device should be small, which is achieved by using a technology with small devices, and within that technology, finding the smallest usable elementary device.
- It should be possible to interconnect the devices to form a network, which means that the input and output signal representation should match, and that the devices should be able to drive each other.
- At least part of the learning algorithm must be implementable in the same technology.
- Most devices have some unwanted, or unusable properties, described as non-idealities. A successful device has non-idealities that can be dealt with somehow, either by the device itself, the surrounding devices in the network, or at the system level by the learning mechanism.
- Finally, the device should preferably not dissipate any static power, that is, it should not dissipate when it is not processing information.

Unilaterality of the devices is not strictly necessary, but if signals flow only in one direction, it does make the design easier.

In this thesis, it is shown that the Single-Electron Tunneling transistor (SET transistor) is a strong candidate for potentially very large neural nets. The SET transistor is extensively described in Chapter 4.

3.2 The learning algorithm

The success of a neural network depends for a large extent on the performance of the learning algorithm that adjusts the synaptic weights and the neuron threshold values to obtain the desired network behavior. Unfortunately, there is no universal algorithm. Especially for large multi-layer networks, possibly with feedback topologies, developing efficient learning strategies is still an important challenge. We know however, by inspecting the ultimate biological example, our brain, that it should in principle be possible for large neural systems to learn (although it may take a lifetime!). Even though improving the learning algorithm performance is not assessed in this work, some attention is given to the learning algorithm because it forms an integral part of the neural network.

In Section 3.2.1, the function of the learning algorithm for a neural network is described. Section 3.2.2 discusses the implications of implementing the learning algorithm in inaccurate hardware, and in Section 3.2.3, the requirements for compact realizations of the algorithm are discussed.

3.2.1 The function of the learning algorithm

The learning algorithm is responsible for adapting the network behavior to the desired behavior. (The complex matter of optimizing the algorithm for a better and faster convergence deserves a research project of its own, and is not addressed in this thesis.)

It is illustrative to distinguish three tasks of the learning algorithm:

1. adapt the network to perform the desired transfer function,
2. adapt the network to input conditions,
3. adapt the network to specific hardware.

This distinction is meaningful because of two reasons.

First, if either of the three is time-variant, the learning adjustments should continue during the recall phase of the neural network. Secondly, if the hardware is predictable enough, the learning could in principle be done separately, not using the neural hardware itself.

Whether the network functionality or the input conditions are time dependent is determined by the application, and is not addressed here. SET hardware however is known to have time-dependent offset charges, and the use of small neural devices as primitive functions implies that the hardware is inaccurate. Therefore, training a SET neural network must continue during the recall phase of the network and the neural hardware itself should be involved. The weight updates cannot be computed externally, independent from the neural hardware, because the specific hardware properties such as malfunctioning devices, device parameter tolerances and other systematic errors would then not be taken into account.

3.2.2 Learning hardware

There are two reasons why the learning algorithm should preferably be embedded with the rest of the neural network on the same chip. First of all because the connections to all the weights can then remain short, and second because for compactness, one would like the learning algorithm to be completely realized using the same technology as the rest of the neural network. Analog on-chip learning hardware implementations inevitably result in inaccuracies in the learning algorithm. The neural network operation however relies on the learning mechanism to deal with the inaccuracies, and the question as to whether it is at all possible to correct inaccuracies with another inaccurate system is legitimate. An accurate 'teacher' is always needed, or in other words, some stable reference and accurate regulating loop is always required to tame an inaccurate system. A successful learning algorithm is therefore at least partly realized with accurate hardware. This means that as long as stable and accurate elements are not available in SET technology, part of the learning algorithm has to be implemented in another way.

The type of errors that such a learning algorithm can deal with are the systematic hardware errors: the errors that are time independent on the scale of the learning convergence time. Examples are malfunctioning or defective devices, the effects of device parameter tolerances, and static random offset charges.

3.2.3 Compact learning algorithms

As with all other parts of the neural network, the learning algorithm hardware should be made as compact as possible. This implies that the implementation of the primitive functions should be based on the available elementary devices. The algorithm should therefore be as simple as possible, requiring no complex computations, and be as robust as possible to hardware inaccuracies, so that an important part can indeed be made using inaccurate hardware.

Finally, learning a large neural network means updating a large number of weights. To access those weights without the overhead of long interconnections, the learning algorithm should operate with only local signals, and to make it faster, all weights should be updated in parallel.

3.3 Information representation

The representation of the information signals in the neural network defines the way the information is coded, how the information is carried by the signal, and the electrical quantities used in the network. This section describes how the information representation influences the compactness and power consumption of the implementation.

3.3.1 The electrical carrier

The electrical quantity used to carry the information is determined by the device properties and the interconnection to other parts of the network. The available quantities are voltage and current (or charge, which is the non-moving variant of current).

For the neuron output, distribution of the signal is important and therefore a voltage is most useful. At the input of the neuron, signals should be added, so current is most appropriate. In Chapter 4, it is argued that the current biased SET transistor with voltage output has promising signal processing properties.

3.3.2 The signal carrier

A variety of signal carriers is available to represent an information signal. Among them, we find:

- instantaneous amplitude,
- instantaneous phase,
- instantaneous frequency,
- pulse position modulation,
- pulse width modulation,
- amplitude modulation,

The question is which of them is best suited for compact, low power representation of signals. There is a trade-off between power consumption and robustness. The power consumption of instantaneous amplitude representation is lowest because in all other cases the overhead of generating the carrier also costs power [20]. If the signal carrier is an instantaneous frequency or phase for instance, the fluctuating signal dissipates, but does not represent information by itself. Instantaneous amplitude representation however, is also the least robust, because noise has a direct influence on the signal.

The implementation compactness of a specific information carrier cannot be judged without considering the specific devices used because it heavily depends on the device properties.

In Chapter 4, we will see that the analog instantaneous amplitude of the SET transfer curve has powerful signal processing properties.

3.3.3 Analog or discrete signals

Discrete signals are more robust to noise and other errors than analog signals because only predefined levels exist. There are however two reasons for not using discretized signals. First, robustness to noise and other errors is not strictly necessary in neural networks because the network is able to correct for errors at the system level. Second, the quantization errors resulting from discretization are not stochastic, but strongly signal related. This can cause serious learning problems because the system 'bounces' every time against the same barrier. The high digital accuracy required to overcome this is at the expense of circuit compactness. Compact analog implementations on the other hand are affected by stochastic errors, which are far easier handled by the system.

3.4 The network topology

The topology of the neural network describes the way the neurons and synapses are interconnected. For a compact network, the area used by interconnections should be minimized. This can be achieved in two ways; by allowing only short interconnections, and by limiting the number of connections. Therefore, locally connected topologies are a strong candidate for large compact neural networks, as described in Chapter 7.

The above is purely a circuit argument in favor of cellular topologies. Successful examples of locally connected neural networks are found in the field of signal processing.

The subject of training cellular neural networks is still largely unsolved, but it will have to be addressed before SET cellular neural networks can operate.

3.5 Conclusions

The standard mathematical descriptions of neural functions are very useful for better understanding of the neural network concept, but they are not the easiest functions to implement in hardware. Since the main goal of making neural hardware is to obtain larger and faster networks, it is essential that the basic functions can be made as simple as possible. When trying to map the mathematical functions onto electronic hardware, the required accuracy results in a larger implementation than necessary.

The smallest possible implementation of a neural building block exploits the specific technological properties in a compact analog hardware structure. Such a neural device cannot perform the precise neural function accurately, but the absence of accuracy and robustness is compensated for at the neural system level by combining the results of many devices, and through adjustment by the learning algorithm.

The learning algorithm of a compact neural network is implemented as much as possible with the same technology as the rest of the network, but as long as this hardware is prone to inaccuracy, some form of accurate 'teacher' that supervises the network remains necessary.

The topology best suited for large compact neural networks is locally connected, but learning aspects with these topologies will have to be investigated before applications become feasible.

4

SET devices and circuits for neural networks

The SET transistor was introduced by Likharev in 1987 [21]. Since then, many proposals for its applicability in VLSI systems have been published [22–26], mainly by translating the currently very successful digital CMOS circuit principles into SET transistor variants. One of the reasons for this attention is the small device size allowing 10^{11} SET transistors to be packed together in an area of 1 cm^2 [22]. Most of the digital proposals have not proven very successful so far, which is at least partly due to a number of unorthodox properties of the devices. The most important one being the random offset charge present in SET transistors. Most digital designs in SET technology assume that the offset charge problem can be solved in the future. As yet, every single SET transistor must be adjusted separately to compensate for the offset charge, making VLSI integration impossible.

As explained in Chapter 3, one should be cautious when mapping an established system concept onto a completely new technology. The digital system concept and CMOS technology for example, have been optimized for each other over the years, and when trying to map the system onto a new technology, it is not at all probable that it fits seamlessly. When trying to use a new technology to build large systems, it is essential to carefully analyze the specific properties of the new devices, and to exploit exactly those properties as much as possible. Only then is the technology used optimally and is there a chance of creating a successful system.

In this chapter, the properties of the Single-Electron Tunneling transistor are described from an electronic circuit design point of view, paying special attention to how SET devices can be used in analog VLSI circuit design.

For an overview of SET technology in general, see for example [27].

4.1 What is Single-Electron Tunneling?

The **tunnel junction** is the basic element in Single-Electron Tunneling (SET) technology, similar to the *pn* junction being the basic element in semiconductor technology. A tunnel junction simply consists of two conductors separated by an extremely thin insulator, the tunnel barrier. Being so thin, the insulator does not completely prevent electrons from passing under all circumstances: it can be considered to be a leaky capacitor. Seen from an electronic point of view, the principal difference between an electric current through a regular conductor and one which passes through a tunnel barrier, is that the former is *continuous*, while the latter can be *quantized* under certain conditions. To make a current flow, a certain amount of energy is required, and because of the quantized nature of the charge carriers in a current across a tunnel junction, a minimum amount of energy must be available to make electrons tunnel. Below this minimum, no current can flow, which is called the **Coulomb blockade**. This phenomenon forms the basis for many circuits in SET technology.

SET junctions can be made in several different ways. Although there are many similarities between them, there are also some differences. The use of metal tunnel junctions is assumed throughout this thesis.

4.2 The SET transistor

The **SET transistor** [2, 21, 27] is the simplest known device that can be constructed with SET junctions. It consists of two tunnel junctions connected in series, and a gate electrode which is either capacitively or resistively connected to the island formed by the node connecting the two junctions [21]. The resistively coupled SET transistor is currently very difficult to make reliably because the gate resistor must have a very large value (see Section 4.2.5). In this work, we concentrate on the metal SET transistor with a capacitively coupled gate, as schematically shown in Figure 4.1. Figure 4.2 shows a Scanning Electron Microscope image of a SET transistor.

The current flow I_d through the SET transistor consists of individual electrons tunneling through the source junction to the island and from the island through the drain junction. Whether electrons tunnel, depends on the charge present on the island enclosed by the dotted box. This charge changes discretely with the elementary charge e when electrons tunnel through the junctions, and it can be modified continuously with charge q by a voltage over capacitor C_g.

4.2.1 Electron transport in a SET transistor

For the sake of simplicity, consider a SET transistor with two identical tunnel junctions and without a gate, biased with a voltage V_{bias}, as shown in Figure 4.3.

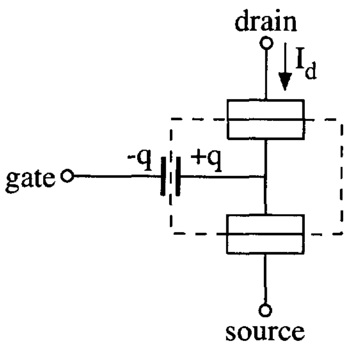


Figure 4.1: The SET transistor.

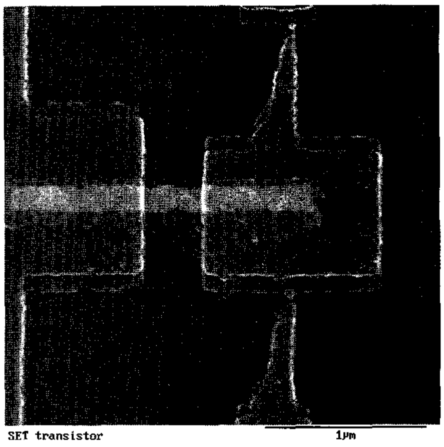


Figure 4.2: Image of a SET transistor.

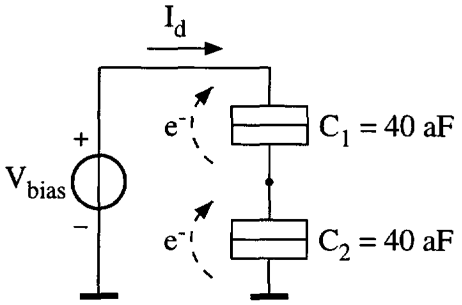


Figure 4.3: Voltage biased SET transistor without a gate.

Electrons can only get on or off the island by tunneling through either the source or the drain tunnel junction. The total island capacitance C_Σ of this SET transistor is so small that the change of the island voltage $\Delta V = \frac{e}{C_\Sigma}$, resulting from adding one electron to the island, is in the order of mV. In this case C_Σ equals the sum of the tunnel junction capacitances $C_\Sigma = C_1 + C_2$. So when $C_1 = C_2 = 40$ aF, adding a single electron to the island decreases its voltage by about 2 mV (Figure 4.4.b) in comparison with the situation at rest (Figure 4.4.a). Removing an electron through the other junction then brings the island potential back to the original level (Figure 4.4.c).

With a voltage across one of the tunnel junctions smaller than $\frac{e}{2C_\Sigma}$, transfer of an electron across it would decrease the voltage to less than $-\frac{e}{2C_\Sigma}$. This increases the electrostatic energy $E = \frac{e^2}{2C_\Sigma}$ of the system. Since an electron only tunnels if it decreases the electrostatic energy, tunneling is blocked unless the voltage over the junction is higher than $\frac{e}{2C_\Sigma}$. This is called the **Coulomb**

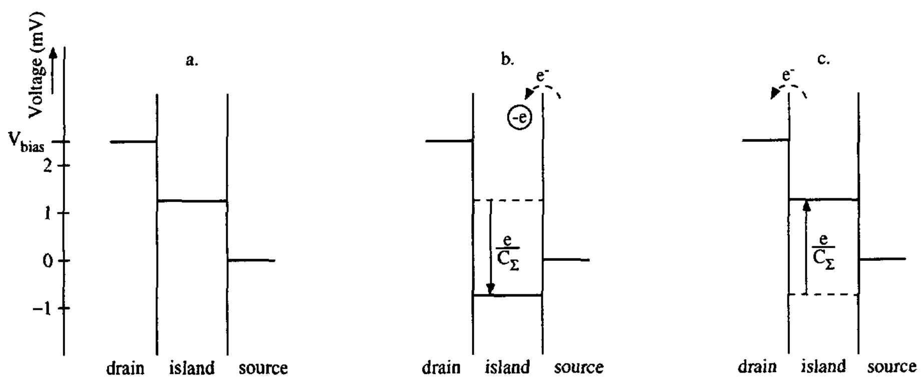


Figure 4.4: An electron is transferred from source to drain. **a.** Before electrons have tunneled through either junction, the island potential of the SET transistor in Figure 4.3 is half of the bias voltage. **b.** When an electron tunnels, in this case through the source junction, the voltage level of the island decreases by $\frac{e}{C_\Sigma}$ (2 mV). **c.** The subsequent tunneling of an electron through the drain junction brings the island potential back to its original level.

blockade. The voltage range $-\frac{e}{2C_\Sigma}\cdots\frac{e}{2C_\Sigma}$ is often called the **Coulomb gap** for that junction.

The key mechanism for the operation of a SET transistor is that tunneling of a single electron through one of its junctions can put that junction in Coulomb blockade, and simultaneously suppresses the blockade of the other junction. Subsequent tunneling of an electron across that other junction blocks it again, and suppresses on its turn the blockade of the first junction. This way, electrons can only pass through the SET transistor one at the time.

This mechanism can be explained with the aid of Figure 4.4. As mentioned before, an excess electron on the island decreases the island potential by a voltage $\frac{e}{C_\Sigma} = 2$ mV in our example. The Coulomb blockade voltage is in our case equal to $\frac{e}{2C_\Sigma} = 1$ mV. This implies that if before tunneling of an electron through the source junction, the island voltage were between 1 mV and 2 mV (Figure 4.4.a), it is between -1 mV and 0 mV after tunneling (Figure 4.4.b). This puts the source junction in Coulomb blockade, and no more electrons can tunnel across it. The voltage across the drain junction however, has now increased to more than 1 mV, suppressing its Coulomb blockade and allowing an electron to tunnel across it (Figure 4.4.c). One electron has now passed from source to drain, and the original voltage situation is restored.

The **threshold voltage** of a SET transistor is defined as the voltage above which a SET transistor at 0 Kelvin starts to conduct current. The Coulomb gap for the two junctions in series equals the sum of the Coulomb gaps of the two individual junctions because the bias voltage is equally divided between the two tunnel junctions. The threshold voltage then equals $V_{\text{th}} = \pm\frac{e}{C_\Sigma}$, as indicated by the solid line of Figure 4.5. The threshold voltage can be decreased to zero with the gate electrode, as explained below.

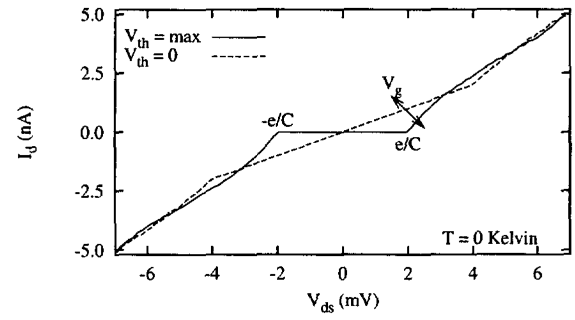


Figure 4.5: A SET transistor conducts current when the voltage across it is larger than the threshold voltage V_{th}.

4.2.2 The influence of the gate on the threshold voltage

The gate electrode of the SET transistor is formed by a capacitive connection to the island. The gate voltage induces a charge on the island, which changes the voltage across the two tunnel junctions, and thus alters the tunneling properties of the SET transistor. In particular, when the voltage across both junctions is so low that both are in Coulomb blockade (Figure 4.6.a), this blockade can be suppressed for one of the junctions by the voltage on the gate. In this case (Figure 4.6.b), the Coulomb blockade is suppressed for the source junction. The tunneling of an electron through that junction re-establishes the Coulomb blockade for the source, and suppresses it for the drain (Figure 4.6.c), so that an electron can pass from the island to the drain (Figure 4.6.d), which restores the situation of Figure 4.6.b.

The maximum reduction of the threshold voltage is obtained when the voltage induced on the island by the gate equals $\frac{e}{2C_\Sigma}$, which exactly suppresses the Coulomb blockade, even with $V_{ds} = 0$ (Figure 4.7). As illustrated by the dotted line of Figure 4.5, the threshold voltage then equals zero.

4.2.3 The origin of the transfer function periodicity

The periodic gate voltage sensitivity is probably the most characteristic property of the SET transistor. It originates from the interaction between the continuous

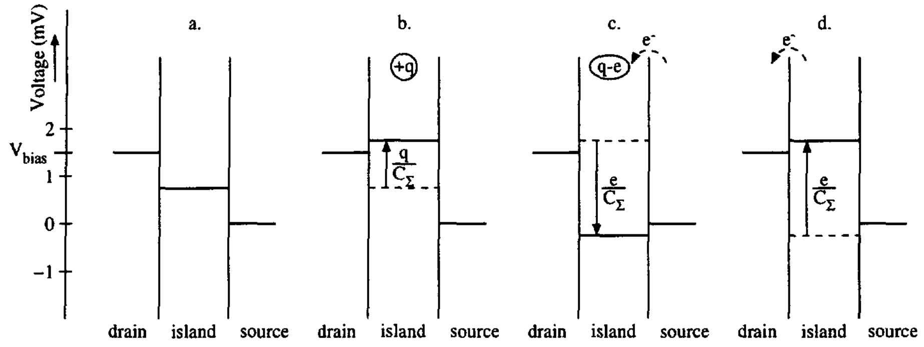


Figure 4.6: The influence of the gate on the island potential. **a.** The bias voltage is in the Coulomb gap. No tunneling is possible. **b.** With the gate voltage, the island potential is increased to overcome the coulomb blockade of the source junction. **c.** The additional electron on the island reduces the island potential to a level that allows tunneling through the drain junction. **d.** After an electron has tunneled across the drain junction, the island voltage rises again to the level of b.

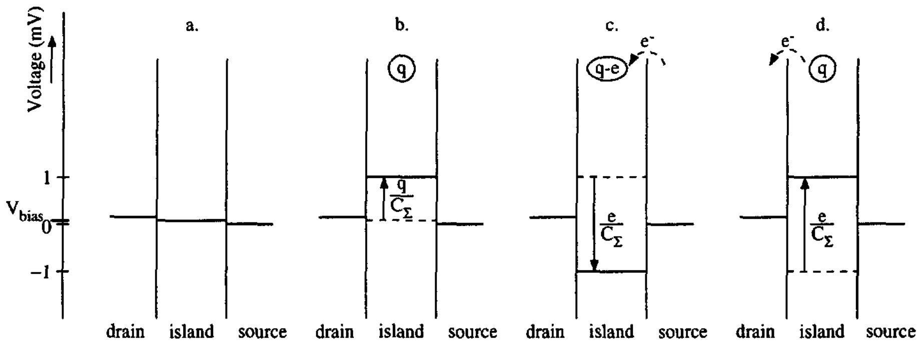


Figure 4.7: With a vanishing bias voltage, the Coulomb blockade is only suppressed when the voltage induced on the island by the gate is equal to $\frac{e}{2C_\Sigma}$ (which is 1 mV in this example). At that point the threshold voltage reduces to zero.

nature of the voltage induced on the island by a conventional capacitor like C_g, and the discrete nature of the charge transfer through the tunnel junctions.

Figure 4.8 illustrates what happens when the gate induces a higher voltage than $\pm\frac{e}{2C_\Sigma}$ on the island. In Figure 4.8.b, a voltage of $\frac{2e+\Delta e}{C_\Sigma}$ is induced on the island. Tunneling of two electrons through the source junction reduces the island voltage by $\frac{2e}{C_\Sigma}$ (Figure 4.8.c). Since no fractional charges can cross a tunnel junction, the effective potential of the island induced by the gate voltage equals $\frac{\Delta e}{C_\Sigma}$. Since every complete electron induced on the island is compensated

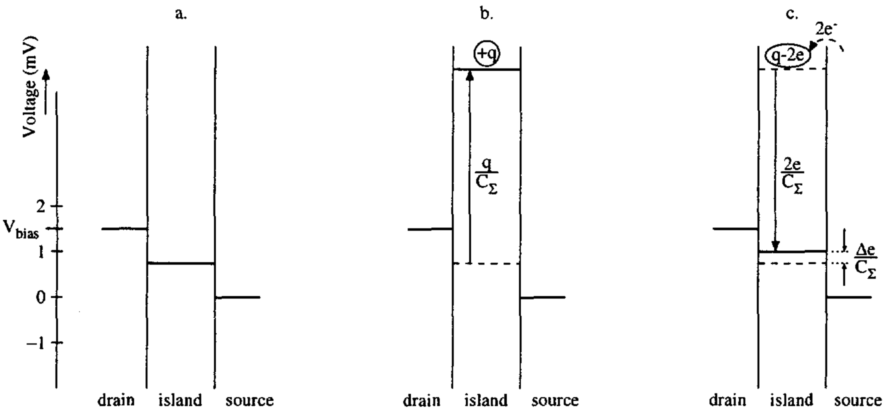


Figure 4.8: When a potential of $\frac{2e+\Delta e}{C_\Sigma}$ is induced on the island by the gate, $\frac{2e}{C_\Sigma}$ can be compensated for by tunneling electrons. The fraction $\frac{\Delta e}{C_\Sigma}$ remains.

for by an electron from either the source or drain, the *effective* voltage on the island is a periodic function of the gate voltage, as illustrated by Figure 4.9. This results in the the threshold voltage being a periodic function of the gate voltage, as illustrated by Figure 4.10.

4.2.4 The electronic properties of the SET transistor

To be able to use SET transistors for electronic circuit design, it is important to have insight in the electronic behavior of the devices. This section concentrates on the specific SET transistor properties such as the model parameters, I-V curves, the transfer function, the island offset charge fluctuations, noise, power dissipation, and current leakage.

The SET transistor parameters

A SET transistor can be characterized by the following parameters (see Figure 4.11).

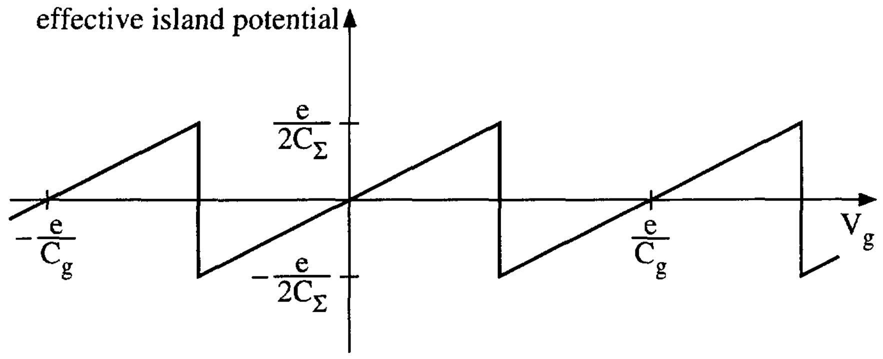


Figure 4.9: The effective island potential with respect to the source as a function of the gate voltage, assuming $C_g \ll C_1, C_2$.

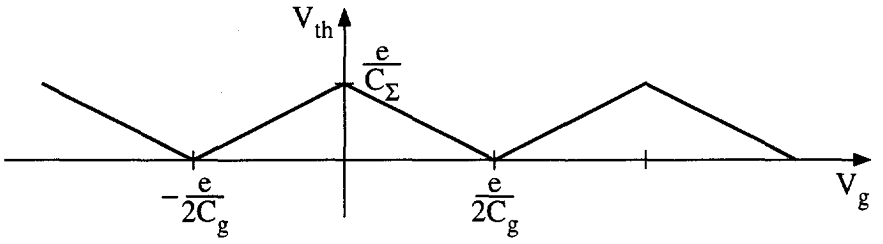


Figure 4.10: The threshold voltage V_{th} as a function of the gate voltage V_g, assuming $C_g \ll C_1, C_2$.

- Tunnel junction capacitances: C_1, C_2
- Tunnel junction resistances: R_1, R_2
- Gate capacitances: C_{g_1}, C_{g_2}
- Stray capacitance: C_s

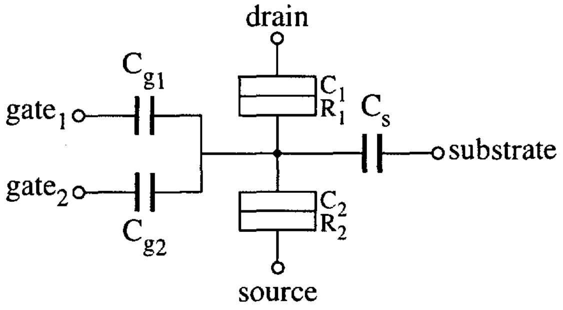


Figure 4.11: The model of a SET transistor

The value of the tunnel junction capacitances is determined by the area of the tunnel junction, the thickness of the tunnel barrier, and the barrier material. A currently common value for the junction overlap area is around 50×50 nm^2 to 100×100 nm^2. The barrier thickness is in the order of 1 nm, which is a few to tens of mono-layers of aluminum oxide. The junction capacitance is in the order of 100 aF. Figure 4.12 shows a picture of a fabricated tunnel junction. The overlap area of the two metal electrodes is approximately 50×50 nm^2.

The tunnel resistance expresses the rate of tunneling, and depends on the transmission properties of the tunnel barrier [27]. Like the tunnel junction capacitance, it also depends on the area and thickness of the tunnel barrier. The two parameters can nevertheless be designed independently in practice because the tunnel resistance is exponentially dependent on the oxide thickness, while the junction capacitance is inversely proportional to it. The desired junction capacitance is determined by the overlap area, while the tunnel resistance can be tuned with the barrier thickness without affecting the capacitance appreciably.

Typical values for the tunnel resistances R_1 and R_2 are in the order of 100 $k\Omega$ to 1 $M\Omega$. Lower values cannot be used, as explained in Section 4.2.5.

The gate capacitance can be constructed either in the same plane as the island (side capacitor) or in a layer below the island (overlap capacitor) [2]. Typical values are between 10 and 50 aF for the side capacitor and 50 aF or larger for the overlap capacitor.

The parasitic stray capacitance C_s of the island to the surroundings depends on the island size and the oxide thickness that separates the island from the

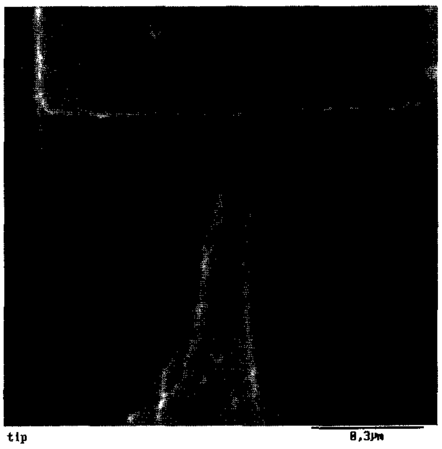


Figure 4.12: One SET junction.

substrate. It is typically in the order of $C_s = 1$ aF [1]. The stray capacitance is ignored in this thesis.

An important derived parameter is the total island capacitance C_Σ, which simply equals

$$C_\Sigma = C_1 + C_2 + C_s + C_{g_1} + C_{g_2}. \tag{4.1}$$

I-V curves

So far, we have examined the flow of individual electrons through the SET transistor. From here on, we consider the average current flow through the device, which is proportional to the number of electrons passing through the SET transistor per second.

The two extreme I-V curves of a SET transistor are shown in Figure 4.13. Coulomb blockade occurs below the threshold voltage V_{th}, which is a periodic function of the gate voltage. It is zero at

$$V_g = \frac{e(n+0.5)}{C_g} \qquad (\text{with } n = 0, \pm1, \pm2, ...), \tag{4.2}$$

because that suppresses the Coulomb blockade completely (see Figure 4.10). At low bias currents, the maximum value of the threshold voltage equals

$$V_{\text{th}_{\text{max}}} = \frac{e}{C_\Sigma}, \tag{4.3}$$

which corresponds to a gate voltage

$$V_g = \frac{ne}{C_g} \tag{4.4}$$

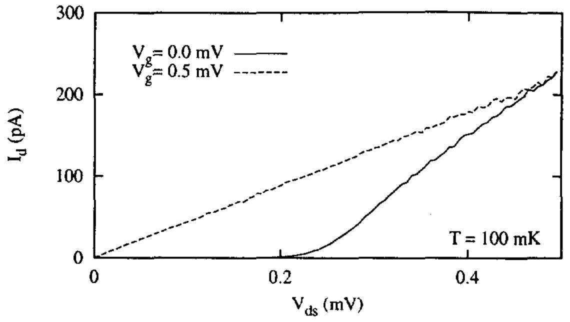


Figure 4.13: I-V curve of a SET transistor for two values of the gate voltage.

(see Figure 4.10). Above the threshold voltage, a current flows through the device. It is equal to $I_d = \frac{V_{ds}}{R_t}$ for high bias, where $R_t = R_{t_1} + R_{t_2}$ is the sum of the tunnel resistances of the two junctions.

The transfer function

The transfer function of a SET transistor is the relationship between an input signal at the gate and the output signal at the drain of the device. The output can be either a voltage or a current, depending on whether the transistor is current or voltage biased. Both situations are described below.

The current biased SET transistor of Figure 4.14 has a transfer function as shown in Figure 4.15. This can be explained with the aid of Figure 4.16, which shows V_{ds}, the two junction voltages V_{di} and V_{is}, and the island charge Q_i as a function of the gate voltage V_g. We see that for the positive slope of V_{ds}, the voltage across the source junction is constantly at the edge of Coulomb blockade for that junction. In this part of the curve, electrons first tunnel through the source junction and then through the drain junction. With increasing V_g, the voltage across the drain junction steadily increases to the Coulomb blockade level. At that point, there is no preference for tunneling first through the drain or the source. As V_g increases further, the voltage across the drain junction remains equal to the Coulomb blockade voltage, which is only possible if the source junction voltage decreases rapidly to counteract the effect of the gate voltage. When V_{ds} is at its minimum, an additional electron is transferred to the island, which restores the original voltage situation.

On the positive slope of the transfer function (at 0 Kelvin) the Coulomb blockade of the source junction is suppressed first, and the drain junction follows (as in Figure 4.6). The value of the positive slope expresses the relation between

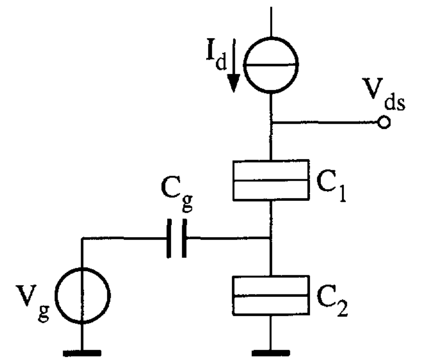


Figure 4.14: Current biased SET transistor.

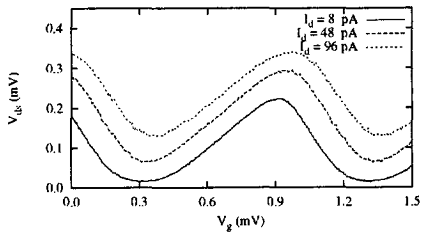


Figure 4.15: The transfer function of the current biased SET transistor.

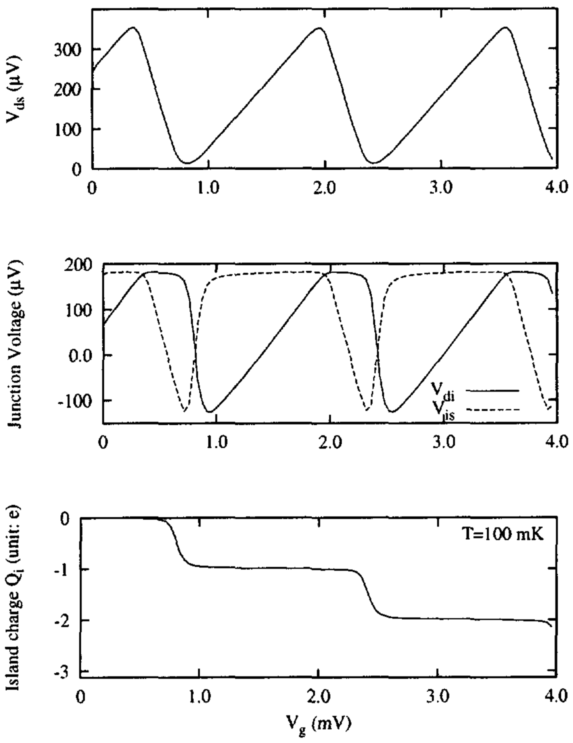


Figure 4.16: The drain-source voltage, junction voltages and island charge as a function of the gate voltage for a current biased SET transistor.

V_g and V_{ds} required to keep the source junction on the verge of blockade:

$$\frac{V_{ds}}{V_g} = \frac{C_g}{C_\Sigma - C_1} \qquad \text{for the positive slope.} \tag{4.5}$$

On the negative slope, the situation is reversed. The Coulomb blockade is first suppressed for the drain junction and the source junction follows. The slope is therefore given by the relation between V_g and V_{ds} that keeps the drain junction voltage constant:

$$\frac{V_{ds}}{V_g} = -\frac{C_g}{C_1} \qquad \text{for the negative slope.} \tag{4.6}$$

This means that on the negative slope, voltage gain is obtained if $C_g > C_1$.

The output voltage is equal to

$$V_{ds} = V_{\text{th}}(V_g) + I_d R_t, \tag{4.7}$$

where the threshold voltage V_{th} is a periodic function of the gate voltage V_g. $I_d R_t$ is a constant term representing the voltage drop over the device caused by the tunnel resistances.

For small bias currents and low temperature, the amplitude of the output voltage is equal to $\frac{e}{C_\Sigma}$, the maximum value of the threshold voltage. For larger bias currents, the amplitude decreases gradually, as can also be deduced from the I-V curve in Figure 4.13, where the lines for the maximum threshold voltage and zero threshold voltage converge at higher values of I_{ds}.

The transfer function of the voltage biased SET transistor of Figure 4.17 is shown in figure 4.18. When the bias voltage is below the maximum threshold voltage, which is 0.2 mV for the SET transistor of Figure 4.17, a current can only flow (at low temperatures) when V_{th} is sufficiently reduced by the gate voltage, which occurs periodically, and results in the current pulses in the V_g domain shown. For higher bias voltages the current no longer drops to zero. The V_g-independent offset in V_{ds} is determined by the tunnel resistance of the two junctions.

Noise

We distinguish three types of noise in a SET transistor:

- quantum fluctuation noise,
- thermal noise,
- charge noise.

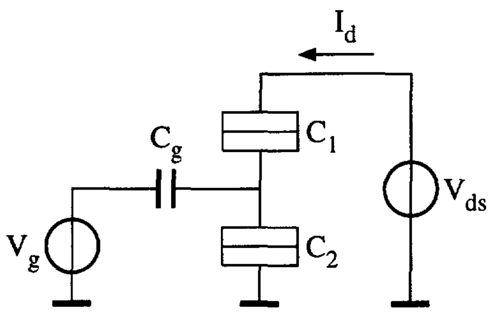


Figure 4.17: Voltage biased SET transistor.

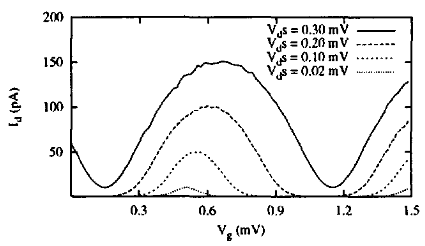


Figure 4.18: Transfer function of the voltage biased SET transistor.

Quantum fluctuation noise results from the Heisenberg uncertainty relation, which states that the uncertainty in the energy of the electrons is $E_q = \frac{h}{R_t C_t}$. This noise should be compared to the electrostatic energy of the electrons $E_c = \frac{e^2}{2C_t}$. The signal to noise ratio therefore is

$$\frac{E_c}{E_q} = \frac{e^2 R_t}{2h} \tag{4.8}$$

so that $R_t \gg \frac{h}{e^2}$ should be fulfilled for a device with low quantum fluctuation noise. $\frac{h}{e^2}$ is called the 'resistance quantum', denoted with the symbol R_K [27] (see Section 4.2.5).

Thermal noise is caused by the thermal energy of the electrons, which can give electrons enough energy to tunnel. As described in Section 4.2.5, small capacitances prevent thermal noise from dominating.

The origin of the offset charge is not fully understood. Some reports indicate it originates in the crystal structure under the island, where crystal defects and impurities form charge traps in the neighborhood of the island [1]. Smaller islands then have less offset charge noise (see also [22] and references therein). Other reports suggest defects and impurities in the tunnel barrier itself could explain the effect (see [28] and references given there). In both cases, reduction of impurities by process improvements can be expected to decrease charge noise. Charge noise has a $\frac{1}{f}$ frequency spectrum [1,2]. Low-frequency charge noise is often called *random offset charge fluctuation*, as described below.

The influence of noise on a neural system depends on the frequency spectrum of the noise sources in conjunction with the time constants of the signal, the learning algorithm and the system function. It is not dealt with in detail in this thesis.

Offset charge

All practical SET transistors have a random offset charge on the island. This results in a completely unpredictable additional term $\frac{q_o}{C_\Sigma}$ in equations (4.2) and (4.4), and on the V_g axis in Figures 4.15 and 4.18. At low frequencies this offset charge is random in size and slowly fluctuates step-wise [1]. It typically remains constant for about one minute to one hour, and then changes by typically a few tenths of e. The intensity of the fluctuations varies considerably from sample to sample, and from one measurement session to the other.

Whatever the cause may be, it appears to be very difficult to eliminate offset charges for the moment[1], and any circuit with SET devices should therefore either be insensitive to the fluctuating offset charge, or else it should be equipped with a mechanism to compensate for it.

Static power dissipation in SET transistors

The power dissipated by SET transistors is an important factor in the success of SET circuits for very densely integrated circuits. Not in the least because the device performance deteriorates quickly as the temperature rises.

Two different sources of power dissipation can be distinguished. First the dynamic power that is dissipated by the signal. This includes (dis-)charging of gate and interconnection capacitances through non-zero resistances. Secondly the static power that may be dissipated in the system when it is at rest, when no information is processed. This static power is dissipated in the biasing circuitry, and in the SET transistors if current flows through them at rest. The maximum power dissipated in a SET transistor is

$$P = I_d(IR_t + V_{\text{th}}). \tag{4.9}$$

All SET transistors that are either voltage biased above the maximum threshold voltage $V_{\text{th}_{\text{max}}}$, or current biased, dissipate static power because current flows through the device when at rest. A SET transistor voltage biased below the maximum threshold voltage can be adjusted to conduct no current, namely when it is in Coulomb blockade.

It is in principle possible to construct complementary circuits with voltage biased SET transistors by always having at least one device in Coulomb blockade for every path between the power supply and ground. Complete logic families can be designed in this way [22, 24, 25]. Like in CMOS technology, the devices do not dissipate static power.

Unfortunately, this attractive approach suffers from one major problem. It only works when there are no island offset charges [22]. The ubiquitous presence of offset charges and their fluctuations necessitates accurate and constant

[1]The early MOS transistors also suffered from offset charges and high $\frac{1}{f}$ noise.

adjustment of every individual island charge in the circuit. Not only to guarantee its correct operation, but also to ensure the complementary behavior that suppresses the static power dissipation.

Fortunately, the static power dissipated by a voltage or current biased SET transistor is relatively low. For the devices used in this thesis, V_{ds} is typically a few hundred μV and I_d is about 10 pA giving a power dissipation of about 10^{-15} W per device. [2]

Therefore, it is more important to design circuits robust to offset charges than low power circuits. Only when offset charges are tamed can SET circuits operate complementary.

Current leakage

As a result of co-tunneling, a small leakage current flows through the SET transistor. Co-tunneling describes the phenomenon that two electrons tunnel simultaneously across the two junctions of the SET transistor [27]. One electron tunnels onto the island and another one simultaneously leaves the island across the other junction. The co-tunneling current is inversely proportional to the junction tunnel resistances. It is in the order of 0.5 pA for tunnel resistances $R_1 = R_2 = 500$ kΩ, so it only affects the behavior of a SET transistor in Coulomb blockade or at extremely small currents.

4.2.5 Electronic boundary conditions

The operation of a SET transistor as described above relies on the fulfilment of two essential conditions regarding the charging energy per electron $E_c = \frac{e^2}{2C_\Sigma}$ of the island.

First, E_c should be larger than the thermal energy $E_T = kT$ to prevent thermal noise from dominating. For the total island capacitance, this implies that $C_\Sigma \ll \frac{e^2}{2kT}$ (about 900 aF at 1 Kelvin and 3 aF at 300 Kelvin). In fact, it ensures that the thermal leakage current is low enough. This restricts the total capacitance connected as a gate. Since more than one gate can be connected to a SET transistor, the maximum number of gates is limited by this requirement.

Second, the quantum fluctuations of the energy on the island should be much smaller than the charging energy E_c. This prevents the electron wave function from extending too much through the tunnel barrier, thereby localizing them to the metal electrodes. This condition is fulfilled when the tunnel resistance of the junctions is much larger than the resistance quantum $R_K = \frac{h}{e^2} \approx 26$ kΩ. If the

[2]For much smaller devices, that can operate at room temperature, V_{ds} is larger while I_d stays constant. This results in a higher power dissipation [24]. On the other hand, the maximum cooling power for mK temperatures is in the order of μW, while at room temperature it is much higher.

tunnel resistances were made smaller than R_K, electrons would spontaneously tunnel across the junctions, completely destroying the Coulomb blockade.

The tunnel resistance of a junction is dependent on the rate of tunneling across a junction [27]. Practical values for the tunnel resistance are larger than 100 kΩ. The resulting relatively high output resistance of SET transistors has practical implications for the way the transistor can be connected to the world. A capacitive load, for instance, reduces the attainable bandwidth. Unless all the island offset charges are compensated for individually, it is not possible to increase the bandwidth by connecting a large number of devices in parallel, as would be possible with regular transistors. The random character of the offset charges levels out the desired behavior [1].

4.3 Using the properties of SET transistors

The properties of SET transistors mentioned in the previous section have to be used effectively when designing circuits with them. This section describes the implications on circuit design of the specific SET transistor properties, such as the periodic transfer function, offset charge fluctuations, limited voltage gain and limited fan-in and fan-out.

4.3.1 Using the periodic transfer function

Since the periodic character of the transfer function of a SET transistor is one of the most fundamental properties of the device, it is important to make use of it. It is however also an unorthodox one, which makes it a challenge to work with. Here we restrict ourselves to the applicability of the periodicity to small neural networks.

The basic function of a neural network is to classify input signals into different categories. We can only use SET transistors to make neural networks if this function can be implemented with a circuit consisting of devices with a periodic transfer function.

Transfer function shapes

For the moment, we are not concerned with the exact mathematical description, but rather with the shape of the function. The type of transfer function of a SET transistor depends on whether the device is current or voltage biased. Both configurations are described below.

The current biased device of Figure 4.14 shows an all positive, continuous transfer function with a well-defined maximum and minimum value. The difference between the maximum and the minimum, the **range** V_R of the output voltage, ideally equals

$$V_R = V_{\max} - V_{\min} = \frac{e}{C_\Sigma}. \tag{4.10}$$

In practical situations, the minimum does not reach zero because of the voltage drop over the finite tunnel resistance and the range is not only dependent on C_Σ, but also on the bias current and temperature. The smaller the bias current and the lower the temperature, the more the range approaches its ideal value.

A voltage biased SET transistor has a transfer function which for large bias voltages is similar to the current biased device, but which is different when the bias voltage is below the threshold voltage, as can be seen in Figure 4.18. The most important difference in shape is that the output current drops to zero for a range of input voltages, so that current only flows for the gate voltage corresponding to an island charge of $\frac{1}{2}e$. Increasing the bias voltage gradually widens the range of V_g for which current flows, until the current no longer drops to zero. The output current range not only depends on the biasing conditions and the temperature, as is mainly true in the current biased case, but also on the tunnel resistance of the junctions. This is a disadvantage because the tunnel resistances cannot be manufactured accurately.

This pulse-modulating property of the transfer functions can be used for the elementary cells of neural networks. In particular, the current pulses in the V_g domain of a SET transistor that is voltage biased below the maximum threshold voltage are similar to the spiking behavior of biological neurons [29] if V_g is made time dependent. The continuous transfer function produced by either the current biased or the voltage biased transistor can also be successfully used for neural networks. This is the subject of the next chapters.

For the moment, consider part of a neural network, as shown in Figure 4.19. The inputs of this cell are generated by the output of other cells. Assuming all cell outputs are generated by the periodic function of SET transistors, the synapses have a limited output range V_{R_s}. The total input signal range $V_{R_{\text{in}}}$ of the cell under consideration is equal to the sum of these ranges $\sum V_R$. The

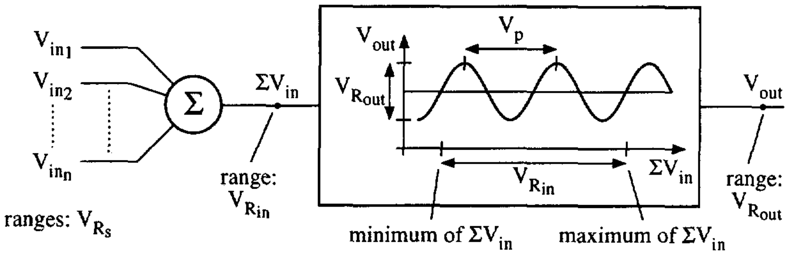


Figure 4.19: One cell of a neural network with a restricted range $V_{R_{\text{in}}}$ at its input, and a period V_P.

relevant part of the transfer function is therefore dependent on the ratio between the its transfer function period V_P and the signal range at its input $V_{R_{\text{in}}}$.

Three different values of this ratio can be distinguished, as indicated in Figures 4.20 to 4.25, which show the three resulting functions of a SET transistor that is voltage biased near the threshold voltage (Figures 4.21, 4.23, and 4.25) and a SET transistor which is current biased (Figures 4.20, 4.22 and 4.24). In Figure 4.24, the dotted lines indicate the effect of different values of the offset charge.

$V_P \ll V_{R_{\text{in}}}$ If the maximum input span of a cell is much larger than the period of the transfer function, the result can be described as a pulse generator whose output pulse repetition rate is determined by the input signal. This could be used as a pulse train generator in a pulse-based network, where the number of pulses is a function of the change in input signal. The function is independent of offset charges since in this case a small shift of the pulses does not influence the overall function. This applies for both the current biased and the voltage biased device. In the case of voltage bias, the width of the pulses can be changed with the bias voltage as long as it remains below the threshold voltage.

A periodic function is also an inherent modulo-operator. It could be used for folding analog-to-digital converters [30], and for AD converters based on a modulo-classifier [31]. As shown in Chapter 6, it can also be used to obtain high resolution synaptic weight updates from low resolution quantized charges.

$V_P \approx V_{R_{\text{in}}}$ If the input span of the cell is of the same order of magnitude as the period of the transfer function, the resulting function maps the input space onto a predominantly high or a predominantly low region. The position of the high or low region can be adjusted using the offset. The boundary between high and low is more pronounced for voltage bias below the threshold voltage, and the width can be modified by adjusting the bias voltage. This type of function can be used for a classifying function in which part of the input space should be mapped on one output value and the rest of the input space on another output value.

$V_P \gg V_{R_{\text{in}}}$ If the input span of the cell is much smaller than one period of the cell's transfer function, the resulting function resembles a multiplier in the case of the current biased device and in the case of a voltage biased device if it is biased above the threshold voltage. The multiplication factor can be modified by means of the offset. The multiplication factors are given by slope values of the transfer function. A device biased with a voltage below the threshold voltage shows a function that can be zero over the complete input range, or can have a partial non-zero value, depending on the offset value.

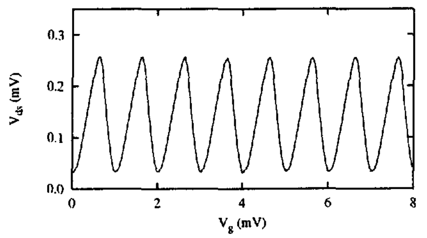


Figure 4.20: Current bias, $V_P \ll V_{R_{in}}$

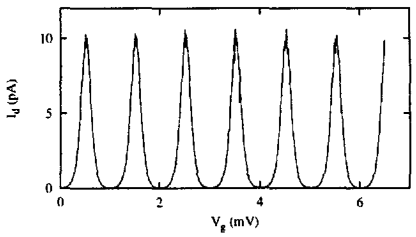


Figure 4.21: Voltage bias, $V_P \ll V_{R_{in}}$

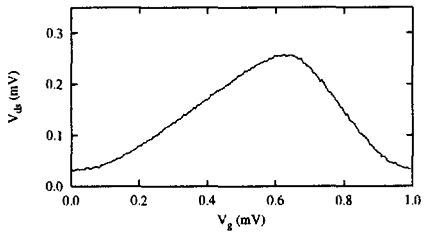


Figure 4.22: Current bias, $V_P \approx V_{R_{in}}$

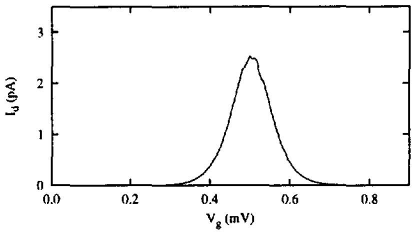


Figure 4.23: Voltage bias, $V_P \approx V_{R_{in}}$

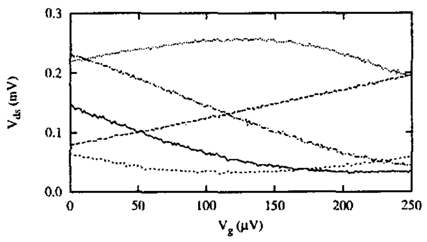


Figure 4.24: Current bias, $V_P \gg V_{R_{in}}$

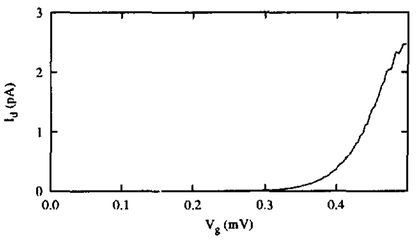


Figure 4.25: Voltage bias, $V_P \gg V_{R_{in}}$

Compensation of signal-independent offset for periodic devices

At some stage most analog circuits need signal-independent offset compensation. SET transistors are no exception. As an example, the multiplier-like function in Figure 4.24 has an offset value at its output that depends on which part of the function is used. Possibly the most important advantage of periodic transfer functions for analog circuits is that as far as its range is concerned, one can be certain that such an offset can be compensated for with a signal no larger than the period of the device. *Any* offset present at the input of a device can be dealt with, how ever large it may be, because it is always folded back to a single period.

4.3.2 The fan-in of a SET transistor

The total number of gates that can be attached to a single SET transistor is limited because of five reasons.

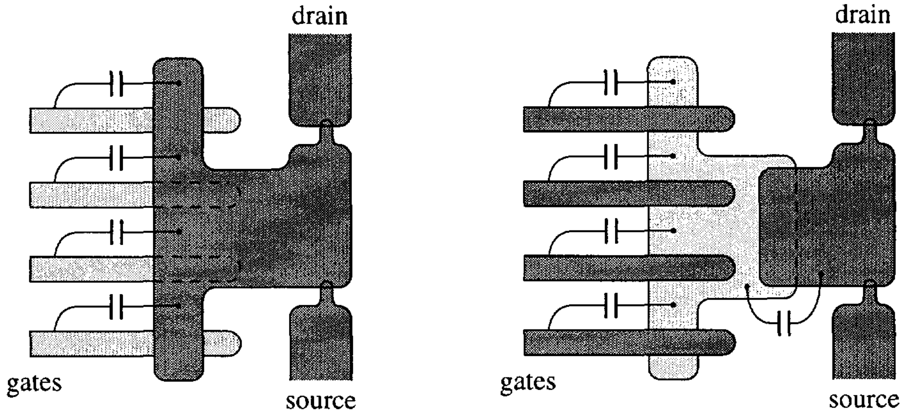


Figure 4.26: Two different ways to couple the gate of a SET transistor to the island: **a.** directly, and **b.** using an intermediate coupling capacitor.

1. The required output range of the SET transistor. This depends on the total island capacitance C_Σ, which is the sum of all gate capacitances (see Section 4.2.5). If a larger number of gate connections is desired, the total effective gate capacitance can be reduced by introducing an additional coupling capacitor between the island and the gate capacitors at the cost of even larger transistor size, as shown in Figure 4.26.b. This is discussed in Section 5.3.

2. The technological minimum size of the gate capacitors.

3. Parasitic cross-capacitance between the gate electrodes and the source and drain electrodes.

4. The required precision and size variation of the gate capacitances. The minimum-size capacitors are usually inaccurate.

5. The size of the island. Smaller islands have less parasitic capacitances to the substrate, and to source and drain.

4.3.3 The fan-out of a SET transistor

At low frequencies, the SET transistor performance is not influenced by the capacitive load at its output. So the fan-out of the device to capacitive gates is large at low frequencies. A high capacitive load however, does decrease the device's bandwidth as a result of its high output impedance.

4.3.4 Limited bandwidth

In Section 4.2.5 it was shown that the output impedance of a SET transistor is high because the tunnel resistance of the two junctions should be much higher than the resistance quantum R_K. This limits the intrinsic bandwidth of the device to $B = \frac{1}{2\pi R_t C_g}$, which for a current state of the art device with $C_g =$ 200 aF gives 10 GHz. This high value will increase further in the future as the capacitances decrease in size. It compares well with the prediction of future CMOS speed [3].

In a circuit however, the bandwidth of a SET transistor is determined by the capacitive load of the rest of the circuit. This load consists of line capacitance and the gate capacitances of the other devices. Especially for large neural networks where an output of a SET transistor is connected to a large number of other devices, it may determine the network bandwidth.

The bandwidth of the circuit can be improved by reducing the number of interconnections and devices to it, for example by using a cellular topology with nearest neighbor communication (see Section 7.1.1).

The connection of the circuit to the outside limits the bandwidth most in the current measurement setup. The long cables to the room-temperature environment are responsible for capacitances up to about 0.1 nF, resulting in a bandwidth in the order of a few kHz. This is not a fundamental problem making the design of SET circuits more difficult, but a practical problem that limits the current measurement possibilities. This subject will also be relevant in SET technology chips of the future when low power SET transistor output signals have to be interfaced to the relatively high capacitive loads of the bonding pads and wires, without loss of speed.

A possible solution, described in [2,32,33], is to integrate buffer amplifiers using HEMTs (High Electron Mobility Transistors) on the same chip to lower the effective output impedance.

The capacitance at the output of the SET transistor has no influence on the quasi-stationary transfer function of the device.

4.3.5 Offset charge fluctuations

The presence of offset charge fluctuations on the island of SET transistors (see Page 33) makes it a particular challenge to use them in a circuit. After all, the transfer function can change completely as a result of some random value that is added to the input signal. Neural networks are known to be robust to variations in the hardware, and if large systems able to cope with it are developed, neural networks will most likely be among them.

In general, there are several ways random offset charge fluctuations can be dealt with in a system, as shown in Figure 4.27. If the offset charges are orthogonal to the signal domain, no further measures need to be taken. Alternatively, three different solutions are at our disposal if the offset charges are in the signal domain. All four options are discussed below.

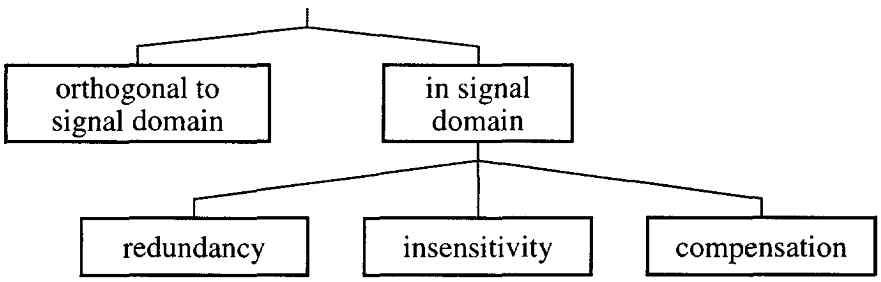


Figure 4.27: Various ways to deal with the random offset charges in a SET circuit.

Orthogonality

Orthogonality of the offset charge fluctuation and the information carrying signal is achieved when fluctuations in the offset charge do not affect the information content of the signal. This is the case with pulse-based systems such as the offset charge independent dynamic memory described in [34]. The information content of a sequence of pulses can be independent of the position shifts caused by offset charge fluctuations.

Insensitivity

If the offset charge fluctuations have an amplitude significantly smaller than the signal amplitudes, the device operation could be insensitive to the fluctuations. Although some reports suggest that for extremely small islands the fluctuations in time could be negligible [22], the random static background charge remains large, so that it is as yet impossible to fully rely on insensitivity [22, 26].

Redundancy

If a system has redundant building blocks, a malfunctioning device, for instance resulting from an offset charge change, could be replaced by another device. This of course only works if relatively few devices are affected with offset charges.

Compensation

To compensate for the offset charge fluctuations, some measure of the error should be extracted from the circuit and used to add or remove charge from the island until the error is minimal. The charge locked loop often used to measure single devices [1] is a good example of such a circuit. It is however too complex to implement for every transistor on a VLSI chip, which is one of the drawbacks common to most system applications that have been proposed for SET technology up to now.

A self learning neural network also contains a control mechanism. The learning algorithm of a neural network measures the difference between the actual and desired output signal and adjusts the synaptic weights to minimize this error. This means that if a neural network is designed such that the offset charge fluctuations can be considered to be synaptic weight fluctuations, the learning algorithm can take care of compensating for the charge fluctuations while it is doing its regular weight adjustment task. This scheme works as long as the learning algorithm converges fast enough to keep up with the random offset charge fluctuations.

Of course, the learning algorithm also needs to be made, preferably using the same technology, shifting the problem of offset charge fluctuations at least partly to the learning circuitry. The learning algorithm is dealt with in Chapter 8.

Alternatively, a small circuit of SET junctions would have to be designed such that it compensates for its own offset charges. Such a circuit has up to now not yet been found.

4.3.6 The signal domains

To decide on the signal domain used to represent the information carrying signal, both the device properties and the system level network requirements, such as the interconnection between synapses and neurons must be considered. In this section, the discussion concentrates on the device aspects such as information transfer between SET transistors, extracting the output signal from the device, and implementing the bias source. The implications at the network level are elaborated in Chapter 7.

Information transfer

The basic signal carrier at the gate of a SET transistor is charge. The information of an electrical input signal arriving at the island of a SET transistor is given by the charge contents of that signal. So to transport information to a SET transistor input, a well defined amount of charge Q should be transported by a current flow $i(t)$ from one node (A) in the circuit to another (B). It is stored at B on capacitor C (see Figure 4.28.a). This can be achieved by letting a current $i(t)$ flow from A to B until the desired amount of charge Q has arrived (Figure 4.28.b). The charge at node A is equal to (Figure 4.28.c):

$$q_A = Q - \int i(t)\ dt, \tag{4.11}$$

and the charge arising at node B equals (see Figure 4.28.d):

$$q_B = \int i(t)\ dt. \tag{4.12}$$

When a voltage source is available to deliver the charge Q, a practical way of controlling the amount of charge transferred is by charging a capacitor C (see Figure 4.29). The current flow stops when $q_B = VC = Q$, so that the desired amount of charge Q can simply be controlled with the voltage source U.

Alternatively, a current source can be used to deliver the charge Q. It can in principle be controlled by a feed back mechanism that senses at A or at B if the desired amount of charge has been transferred (see Figure 4.30), or by a feed forward mechanism that lets a predefined amount of charge pass. Pulse width modulating synapses [29] are an illustrative example of the feed forward system. A voltage source can be seen as an example of the feed back control mechanism.

The use of a voltage source as in Figure 4.29 is the most straightforward way to transfer charge to a capacitively coupled island. Therefore a voltage as the signal carrier at the output of a SET transistor is attractive for signal transport between SET transistors.

Extract output signal

The signal representation at the output of a SET transistor depends on how it is biased. The signal can best be represented by a voltage source if the

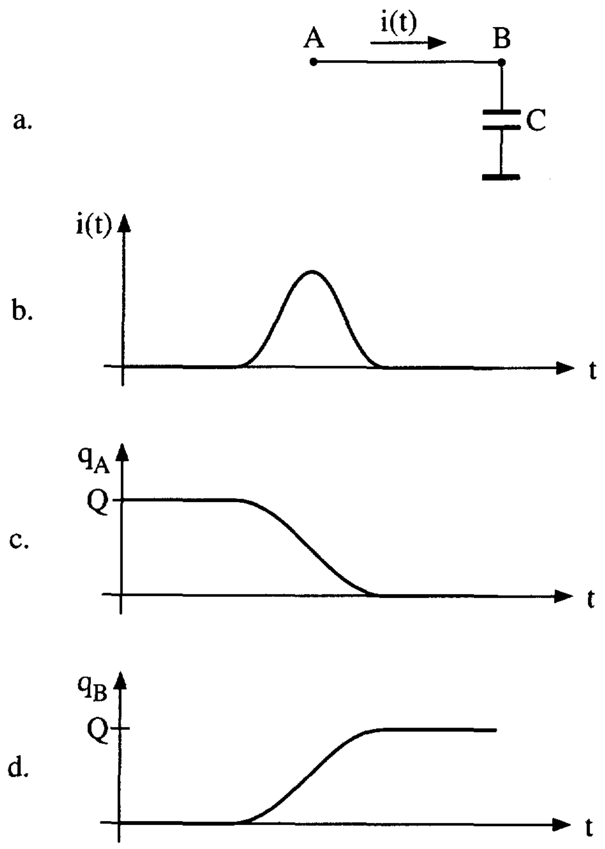


Figure 4.28: **a.** Transport of charge from node A to node B. **b.** The current flow from A to B. **c.** The corresponding charge decrease at node A. **d.** The resulting charge increase at node B.

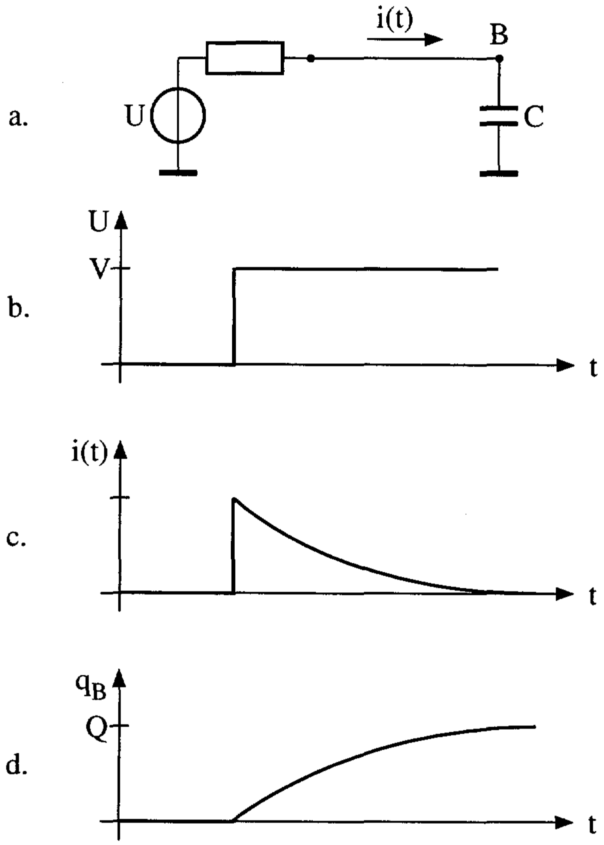


Figure 4.29: Charge Q is delivered by a voltage source U

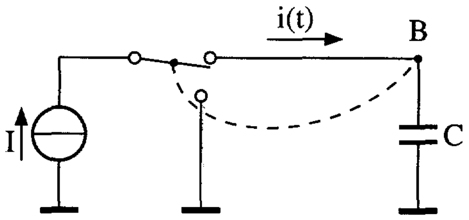


Figure 4.30: Control mechanism to deliver charge Q with a current source I.

device is current biased, and by a current source if the device is voltage biased (Figures 4.14 and 4.17).

If the SET transistor is voltage biased, the current flowing through the bias source must be extracted with some kind of low impedance current sensing element, comparable to the current mirror in regular electronics, as illustrated by Figure 4.31.a. At the moment, no such element or circuit is known to exist in SET technology. Therefore, although implementation of the bias source itself is straightforward, the use of a voltage biased SET transistor is rejected in this thesis.

The output signal of a current-biased SET transistor is a voltage, which can readily be extracted at the output of the device (see Figure 4.31.b). Moreover, as described above, a voltage carrier is attractive for the information transfer between SET transistors. So, even though the implementation of a current bias source needs special attention (see below), the current biased SET transistor is the most attractive single-device configuration.

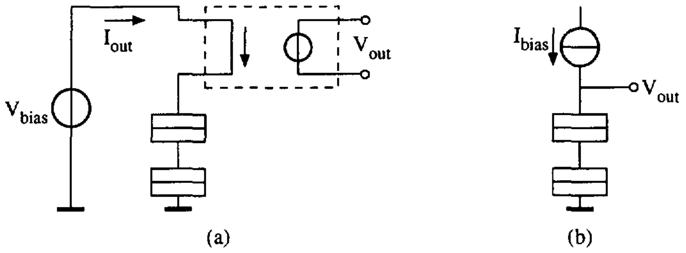


Figure 4.31: Extracting the output signal from a SET transistor. **a.** Current extraction from a voltage biased SET transistor, and **b.** Voltage extraction from a current biased SET transistor.

Current biasing a SET transistor

Since the impedance of a SET transistor is at least 100 kΩ (see Section 4.2.5), making a current source for it is a challenge on its own. The current source impedance must be much larger than the transistors impedance. Moreover, every individual neuron and synapse needs its own current source.

To locally convert a power supply voltage to a bias current requires a large resistance. In principle several types of resistors could be used, such as ohmic resistors, tunnel resistors, or pinch resistors (if semiconductor material is available). All three have disadvantages such as large area consumption, inaccuracy, or unreliability.

To use the current-biased SET transistor in large scale integrated circuits, a good solution for generating the bias current, such as a good high-resistance material or a reliable high resistance tunnel junction, needs to be found. Alternatively, another topology with SET transistors in series instead of in parallel could alleviate the problems because a single current source could then bias more than one device.

Conclusion

In this thesis, the current-biased SET transistor is used as single-device building block for neural networks because its output signal can easily be used as input of another device. The less attractive biasing conditions are taken for granted because it can in principle be solved, and only manifests itself in the device biasing, and not in the signal behavior.

4.3.7 Gain

In a long chain of cells, power gain is required to maintain the signal level. As in a MOS transistor and any other device with a purely capacitive input terminal, the energy gain at low frequencies is extremely large. In this thesis we do not consider the behavior of SET transistors at high frequencies.

Instead of considering the island charge as the SET transistor input, the voltage across the gate capacitor could be considered to be the input quantity. We then have

$$V_g = \frac{Q}{C_g} \tag{4.13}$$

The advantage of this approach for the current biased-SET transistor is that it can now be seen as a voltage converter. It is much more difficult to obtain voltage gain rather than power gain in a SET transistor. Voltage gain is obtained when the transfer from input voltage at the gate capacitor to the output voltage across the two tunnel junctions is locally larger than one in absolute value. It can only be obtained for the negative slope, and when $C_g > C_1$ (equation 4.6). It is wanted when a voltage amplifier is made with SET transistors [35].

The maximum voltage gain of SET transistors is limited technologically because $C_g \gg C_1$ is difficult to make: for a given minimum tunnel junction capacitance, a high ratio between C_g and C_1 results in a high total island capacitance C_Σ, and thus in a low operating temperature. The limited voltage gain has consequences for the way the devices can be used for neural networks. Although voltage gain is clearly desirable for neural network implementations, and even required for feed back topologies, it is not absolutely indispensable for feed forward topologies.

Voltage gain in the neural cells ensures that the output voltage range is at least equal to the input voltage range, so that the signal does not die out as it propagates through the network. A feed-forward network with a limited number

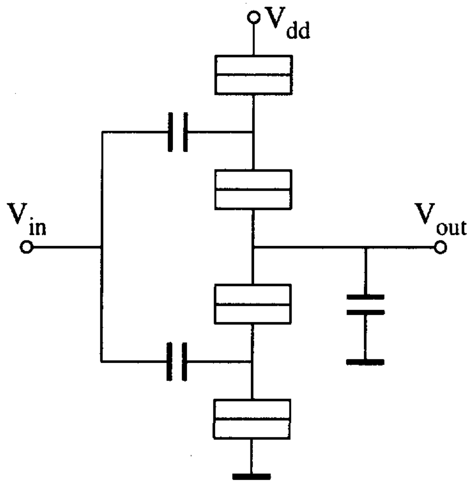


Figure 4.32: The SET inverter

of layers could operate without voltage gain. The maximum number of layers is then limited by the minimum signal level that can be accepted at the network output, and by the signal level deterioration caused by the SET transistors.

Neural networks with topological feed back loops can only operate if loop gain is present, otherwise the signals die out.

4.3.8 The SET inverter

An alternative to the current-biased SET transistor as the building block for large systems is known as the SET 'inverter' [25,26,36], as shown in Figure 4.32. It is a direct translation of the CMOS inverter, as pointed out in [21]. The advantage of this circuit configuration is that it is completely based on voltages. It is biased with a voltage, and has both an input and output voltage representation. The drawback that prevents it from being used in large systems is its two offset-charge adjustment points. Only with correctly adjusted island charges does the device reveal its complementary operation and inverting function.

4.4 SET transistor circuit simulation

In order to design circuits with SET transistors, it is important to have simulation models and their parameters. This section describes how to extract model parameters from measured data of SET transistors, and three different simulation levels.

4.4.1 Parameter extraction

All SET transistor models used for the simulator software described below have the same set of parameters, which is identical to the SET transistor parameters introduced in Section 4.2.4. These parameters should of course be known when attempting to simulate the behavior of a specific device. This section describes how the parameters can be extracted from measured device data [37].

The tunnel resistances R_1 and R_2

The sum of the two tunnel resistances $R_1 + R_2$ can be extracted from the I-V curve, where the slope equals $R_1 + R_2$ for biasing conditions much larger than the maximum threshold voltage. This can even be done at room temperature. The individual tunnel resistances can be determined by creating an asymmetric voltage distribution across the two junctions. The asymmetry between the tunnel resistances can be deduced from the asymmetry of the transfer curves.

The gate capacitance C_g

The gate capacitance can be determined by measuring the period of the transfer function. One period of the transfer function equals $\frac{e}{C_g}$, when the voltage on the gate is considered to be the input signal. When more gates are present, each one can be measured individually.

The stray capacitance C_s

The stray capacitance of the island to the environment can be measured by sweeping all bias voltages with respect to the environment. This environment then serves as the gate. The period of the transfer function measured like this gives the stray capacitance C_s.

The tunnel junction capacitances C_1 and C_2

The tunnel junction capacitors can be obtained indirectly from the $V_{ds} - V_g$ curves by measuring the two slopes and the output voltage range. The output voltage range gives the total island capacitance C_Σ. The negative slope equals $-\frac{C_g}{C_1}$, so that C_1 can be calculated using the previously measured C_g. The positive slope equals $\frac{C_g}{C_\Sigma - C_1}$, from which C_2 can now be extracted. A disadvantage of determining C_2 in this way, is that its accuracy is determined by measurement errors in C_g, C_Σ, and C_1. C_2 can also be obtained by interchanging the source and drain electrodes and considering it as C_1, which can be extracted from the negative slope.

The above measurement of slopes and output voltage range have to be performed at the lowest possible temperature because it influences the output voltage range and the slopes (see Figure 6.12).

The island capacitance can be obtained in a second way, namely by adding all the individual measured capacitors: $C_\Sigma = C_1 + C_2 + C_g + C_s$. Discrepancies between this value and the one obtained from the voltage output range can be attributed to inaccurate measurements of the latter due to non-zero temperatures, and to parasitic capacitances between the island and the source and drain electrodes. Since these capacitances do not contribute to the tunnel capacitances, they are not taken into account.

4.4.2 Tunneling-based modeling

The physically most accurate way to simulate SET circuits is by calculating each individual tunnel event separately [38]. This is done by calculating the rates Γ of every possible tunnel event in the circuit, based on the change in electrostatic energy ΔE in the circuit resulting from that event.

$$\Gamma = \frac{\Delta E}{e^2 R_T (1 - e^{-\frac{\Delta E}{kT}})}, \tag{4.14}$$

where R_T is the tunnel resistance of the junction, k is Boltzmann's constant, and T is the temperature. (See [27] and the references given there). These tunnel events are described by a Poisson distribution, and weighed with this tunnel rate, one of the possible tunnel events is chosen at random to occur. Averaging over many events using the Monte Carlo method yields the circuit behavior for that time step.

Except for accuracy, this method has the advantage of being able to handle an arbitrary circuit of tunnel junctions and other circuit components because interactions between the various parts of the circuit are automatically accounted for.

A currently popular simulation program that works in this way is SIMON-'Simulation of Nano-structures' [38]. It was used to generate most of the SET curves in this thesis (see Appendix A).

A disadvantage of Monte Carlo simulators is without doubt the large number of iterations required and it is therefore best suited for small circuits. For larger circuits and for complete systems, we have to resort to alternative simulation methods.

4.4.3 Current-based modeling

Calculating the current through a junction can be done much faster if the circuit consists of small sub-circuits without mutual charging interactions, such as SET transistors. Instead of deducing the current through the junction from a large number of randomly distributed tunnel events, one could also deduce it directly from the tunnel probabilities. For the SET transistor we find [1]

$$I = \sum_{n=-L}^{L} (\Gamma_n^- - \Gamma_n^+)P_n, \tag{4.15}$$

where I is the current to the island through the junction under consideration, n is the number of electrons on the island, Γ_n^+ is the tunnel rate that increases n, Γ_n^- is the tunnel rate for decreasing n, and P_n is the probability of finding n electrons on the island. L is the approximation of infinity such that the probability P_n is negligibly small for $|n| > L$.

4.4.4 Behavioral modeling

When analyzing the behavior of a large network of SET transistors, the simplest possible model of the device that still has the relevant shape properties should be used. The essential parameters for the transfer function shape are

- The period
- The output range
- The random offset charge
- The temperature dependence

To simulate neural networks, a simple mathematical equation was constructed with parabolas and straight lines to generate the SET transistor transfer function.

As illustrated in Figure 4.33, one period of the transfer function is divided into consecutive line segments $f_1 ... f_5$ that were periodically repeated. Temperature is modeled by the parameter that determines the boundary between the straight line and the parabola, as indicated by the dotted line in the figure. The choice of this particular solution was based on the model definition possibilities available in PSTAR, the circuit simulation program for which the model was made.

In Figure 4.34, two simulated transfer functions of the same SET transistor are compared. The continuous line was generated with the Monte Carlo simulator SIMON, and the dotted line is calculated with the segments of Figure 4.33 using the circuit simulator PSTAR. We see that the period, amplitude and shape of the two transfer functions are very similar. Only the offset charge is different. No effort was made to match the offset charges because the two

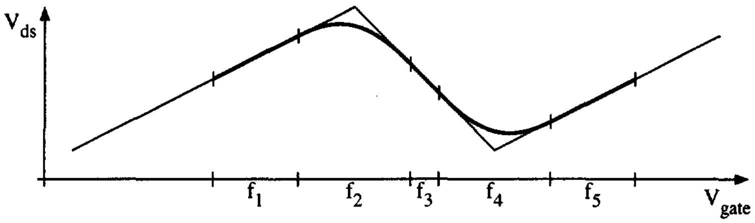


Figure 4.33: Construction of the SET transistor transfer function with five line segments.

curves can better be compared when they do not overlap. At higher temperatures the difference between the two is larger because the PSTAR model approaches parabolic peaks and troughs, while the physical orthodox theory model approaches a sinusoid [39].

This means that the simple mathematical equation can be used to model the SET transistor for system-level simulations. The model was used in Chapter 8 to simulate the behavior of a two-layer SET network.

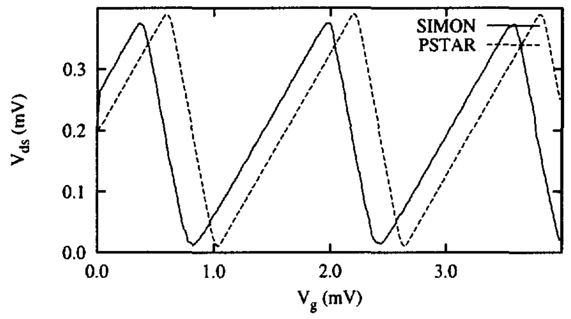


Figure 4.34: Comparison between the transfer function generated by the line segment model (PSTAR) and the one calculated with the orthodox theory (SIMON).

4.5 Conclusions

A SET transistor is a small device that transports individual electrons from its source to drain. The rate of electron transfer can be influenced by a voltage on a capacitively coupled gate. The transfer function of the device is a periodic function of the gate voltage.

A SET transistor driven by another such device only uses a restricted part of its periodic transfer function because the range produced by the device at its input is also limited. The ratio between this range and the value of the transfer function period has an important influence on the actual transfer function shape of the transistor under consideration. It is therefore an important design parameter.

Random offset charge fluctuations influence the device behavior significantly. To prevent them from determining the circuit behavior, measures should be taken to deal with the effects of these fluctuations. Several options can be explored, such as making the information carrying signal orthogonal to the fluctuations, or compensating for the fluctuations to cancel their effects.

Current-biased SET transistors are used throughout this thesis, despite the more complicated biasing and presence of static power dissipation, because a current-biased SET transistor combines a voltage output signal carrier with a single offset charge. The voltage output can be extracted from the device and is compatible with the input signal of other devices, and hence ensures efficient interconnection. As long as offset charges are present, they form a more important impediment to large scale integration than power dissipation.

The circuit simulation of SET devices is possible on several levels. The most accurate and calculation-intensive one is based on averaging individual tunnel events. The behavior of any circuit of tunnel junctions can be calculated this way. For circuits of individual SET transistors without mutual charging interactions, the current can be obtained from the tunnel rates, which is much less calculation intensive, and for system level calculations, simple behavioral modeling can describe individual SET transistors.

5 Design of a SET transistor neuron

The role of a neuron in a neural network is to map the information presented at its inputs by synapses onto its output space by means of an activation function.

This chapter starts in Section 5.1 with a definition of the neuron. Section 5.2 analyzes the two required functions of a neuron. It is followed by the design of an addition circuit in Section 5.3, which is common in the two neuron designs with SET transistors of the succeeding two sections. The neuron described in Section 5.4 was designed to produce an output similar to the conventional neuron and the neuron described in Section 5.5 was designed to make optimal use of the technological properties of the SET devices. In Section 5.7, the alternative neurons are compared.

5.1 Definition of a neuron

A neuron collects and combines information from the network input or from outputs of other neurons, and classifies them with a non-linear function. The most commonly used configuration consists of an addition of inputs, followed by a non-linear, classifying function, as described below. Alternatives such as distributed classifying functions, followed by an addition are also reported, but are not dealt with in this thesis.

In formal artificial neural network theory, the synaptic weighing function is sometimes considered to be part of the neuron. If it is, then all operations in the general neural network equation

$$O = f(o_n + \Sigma_i w_i x_i) \qquad (5.1)$$

are performed by the neuron. In Equation (5.1), x_i are the synaptic inputs, w_i the synaptic weights, and o_n is the threshold value of the neuron activation function. The neuron with included synapses is often used for mathematical descriptions of neural networks. The neuron without the synaptic function is usually preferred for electronic circuit design because in practice, synapses are separate electronic building blocks.

In this work, the neuron is defined as in Figure 5.1, so that the synapses are not part of the neuron. The design of a synapse is the subject of Chapter 6. The neuron collects the output signals of the synapses and performs a non-linear operation, called the activation function, on the sum of these signals.

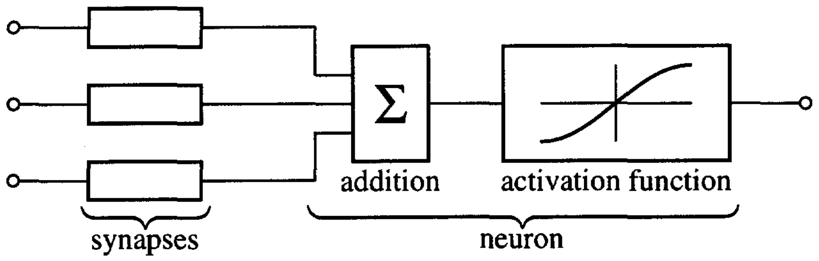


Figure 5.1: Functional block diagram of a neuron

5.2 Essential properties of a neuron

Before an implementation of a neuron is attempted, it is important to analyze its essential properties. The neuron consists of two main functions, namely the addition function and the activation function, and both are described below.

5.2.1 The addition function

The addition function has the task of combining the inputs of the neuron into a single signal that can be accepted by the activation function for further processing. In general, it is desirable to permit an unlimited number of inputs to this function because it allows freedom in the design of the network topology (see Chapter 7). An implementation of the addition function in SET technology is described in Section 5.3.

5.2.2 The activation function

The activation function of a neuron has the role of dividing the input space of the synapses connected to the neuron into different regions. Its presence is essential in classifying neural networks because the actual classification is performed by this function. (The synaptic weights determine how the classification is done.) For most learning algorithms, the convergence success of a learning session is closely linked to the cooperation between the activation function and the learning algorithm. This particularly holds for most steepest descent learning schemes like back propagation and the generalized delta rule. These schemes rely heavily on the mathematical description of the activation function and its derivative because the weight corrections are calculated based on the mathematical description of the synapse and neuron. These calculations often require that the activation function be monotonic and continuously differentiable [40,41].

By far the most generally used activation function is the sigmoidal function as described by Equation (5.2) and shown in the activation function block in Figure 5.1.

$$f(x) = \frac{1 - e^{-ax}}{1 + e^{-ax}}, \tag{5.2}$$

where a is a gain parameter and x is the input value. It is a monotonous function with saturation regions for large values of $|x|$. It has a clear threshold shape, a simple mathematical description, and a simple continuous derivative. This activation function divides the neuron input plane into two halves. In some cases the threshold value can be adjusted by the learning algorithm. The saturation regions at both ends of its input space allow the network to generalize classification into similar, unlearnt patterns because the neuron output is relatively insensitive to small variations of its input signal in the saturation region [40,42].

The sigmoidal activation function is however not necessarily the perfect choice, and alternatives are investigated by various groups. There are two important reasons for investigating alternative activation functions for the neuron. First of all, it is often reported that steepest descent learning in combination with a sigmoid activation function is rather slow and requires too many neurons. Neurons with multi-level activation functions [42], and non-monotonic activation functions such as Gaussian functions [41,43] are reported as alternatives that either shorten the convergence time or decrease the number of neurons required. An activation function based on the negative differential resistance is also reported [44]. Such a function can be compactly generated by a tunnel diode or a small SET transistor circuit [45]. Non-monotonic activation functions have more processing power per layer, and are able to solve some non-linearly separable problems in a single layer. The use of periodic functions for activation functions is also reported [41,43,46,47]. Classification is possible with these different activation functions, but they require modification of the learning algorithm.

The second reason for investigating alternative activation functions is that the sigmoidal activation function in combination with the purely mathematical steepest descent learning algorithm does not necessarily yield the most compact implementation [48], while the success of the neural network concept is based on large, and therefore necessarily compact realizations. In SET technology for example, the choice of an activation function based on a periodic function is a more obvious choice than a sigmoidal activation function, as explained in Section 5.7.

5.3 Implementation of the addition function

Combining the output signals of the synapses at the input of a neuron is usually done by an addition. The addition function circuit used for the neurons described in Sections 5.4 and 5.5 is the same for both cases, and is therefore described separately below.

5.3.1 Signal carrier domain

Adding analog signals electrically can in principle be done in either the current or voltage domain, or by conversion to one of those domains from another domain. For topological reasons, addition in the voltage domain is not attractive, while in the current domain it is. Following Kirchoff's current law, current addition can be realized efficiently at a single node. It is for this reason that the current domain is very popular in designs of the neural addition function in analog neural networks [48–52] and for many other types of analog signal processing circuits.

Charge is the fundamental input quantity at the island of the SET transistor, but the gate capacitor makes the device sensitive to a gate voltage. Connecting more gates to the island results in a capacitive addition circuit with voltage inputs. The properties of a capacitive charge addition circuit are investigated below.

5.3.2 Capacitive charge addition circuit

The basic capacitive addition circuit is shown in Figure 5.2, where the summed output is used by the gate of a SET transistor as an illustration of how the summed charge is used and to analyze the effect of the capacitors on the SET transistor behavior.

The effective charge Q on the SET transistor island can be expressed in terms of the input voltages V_{g_1} ... V_{g_n}:

$$Q = \sum_{i=1}^{n} V_{g_i} C_{g_i} \tag{5.3}$$

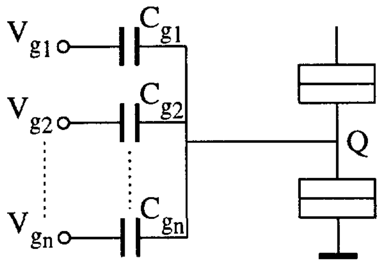


Figure 5.2: Basic capacitive addition circuit at the input of a SET transistor.

(ignoring the voltage drop over the source junction). The output voltage of the current biased SET transistor is a periodic function of V_{g_i}. For each of those inputs, the corresponding period V_{P_i} of the SET transistor transfer function is

$$V_{P_i} = \frac{e}{C_{g_i}}. \tag{5.4}$$

The total capacitance of the island in this configuration equals

$$C_\Sigma = C_1 + C_2 + \sum_{i=1}^{n} C_{g_i}. \tag{5.5}$$

The stray capacitor C_s is ignored in this thesis (see Page 28). The number of capacitive gate connections to a SET transistor is limited because of two reasons. First, the output voltage range of a SET transistor decreases as C_Σ increases. A smaller output voltage range thus makes the device more susceptible to thermal noise, resulting in a lowering of the maximum operating temperature. With the current minimum capacitor size of about 50 aF, the maximum number of gate capacitors for a signal to noise ratio of 10 dB at 500 mK, is about two. This is not enough for a neural network, where many more inputs per neuron are usually desired. Therefore, technological efforts should be undertaken to further reduce the minimum gate capacitor size.

Secondly, the size of the island increases with the number of capacitive inputs because each input requires some space under the island to make the capacitor. This increases the SET transistor size and the charge noise on the island [2].

If necessary, a larger number of inputs can be coupled to the SET transistor island without increasing the island capacitance by introducing an additional coupling capacitor between the addition node and the island, as illustrated in Figure 5.3. Ignoring the voltage drop over the source junction and assuming identical C_{g_i}, the charge on the island is equal to

$$Q = C_g \frac{C_c + (n-1)C_g}{C_c + nC_g} \sum_{i=1}^{n} V_i \tag{5.6}$$

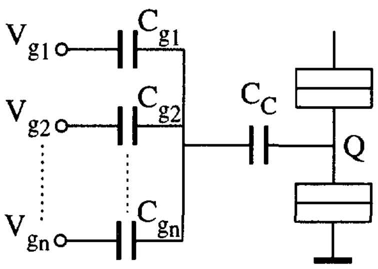


Figure 5.3: Capacitive addition circuit with a coupling capacitor C_c to reduce the load on the SET transistor.

and the total island capacitance equals:

$$C_\Sigma = C_1 + C_2 + \frac{nC_gC_c}{nC_g + C_c}, \tag{5.7}$$

which is always smaller than $C_1 + C_2 + C_c$, regardless of the number of inputs n. A drawback of this capacitive divider is the amplitude loss of the synapse signals V_{g_i}. One period of the neuron SET transistor now corresponds to

$$V_{P_i} = e\frac{C_c + nC_g}{C_g(C_c + (n-1)C_g)}. \tag{5.8}$$

For a four-input adder with all capacitors C_{g_i} equal for example, a five times larger synapse output voltage is required to span the full period of the neuron SET transistor. To prevent a reduced influence on the neuron state, the synapse output range should be increased. Alternatively, the synapses would have to 'cooperate' to change the neuron state, which reduces the processing power. Since both options are unattractive, the coupling capacitor should be avoided in neural networks.

Coupling capacitor C_c is introduced in the neuron design of section 5.4 despite the above arguments against it because this simplification in the production process increases the chance of obtaining a properly working device. It does no harm there because the design is not intended for use in a network.

For a small number of synapses, no coupling capacitor is required in the capacitive addition circuit. This improves the available signal range. In Section 7.3.2, an example is given with two synapses per neuron.

5.4 The SET cascade neuron

The sigmoidally shaped transfer function has proven to be successful for the neuron activation function, as was explained in Section 5.2.2. It is therefore useful to investigate how a transfer function similar to the sigmoid can be obtained. In this section a cascade of two SET transistors is analyzed and its applicability as a neural activation function is discussed [53,54]. The capacitive addition circuitry described in Section 5.3.2 is used as the input to the neuron.

Section 5.4.1 explains the way the transfer function shape is constructed from the individual SET transistor transfer functions. Subsequently, in Section 5.4.2, the capacitor sizes for the two SET transistors are calculated, and in Section 5.4.3, the sensitivity of the transfer function shape to parameter variations is discussed.

5.4.1 Function construction

The function shape of Figure 5.5 is produced by a cascade of two SET transistors, as in Figure 5.4. This function is obtained when the voltage ranges and transfer function periods of the two transistors match in the way described below, and when the two offset voltages are adjusted so that the correct part of the individual SET transistor curves is used. This is explained using Figure 5.6, where for clarity the transfer functions are drawn for the ideal case at a temperature of 0 K. For higher temperatures, the function shape does not change significantly (see Section 5.4.3). Figure 5.6.a shows the output voltage V_{out_1} of transistor T_1 as a function of its inputs $V_{\mathrm{in}_1} + V_{\mathrm{in}_2}$. To obtain the desired behavior, the range $V_{R_{\mathrm{in}}}$ at the input of T_1 equals half of one period. This yields the following relation between $V_{R_{\mathrm{in}}}$ and the gate capacitance C_{g_1}:

$$V_{R_{\mathrm{in}}} = \frac{e}{2C_{g_1}}. \tag{5.9}$$

The output voltage range $V_{R_{\mathrm{out}_1}}$ of T_1 is equal to

$$V_{R_{\mathrm{out}_1}} = \frac{e}{C_{\Sigma_1}}, \tag{5.10}$$

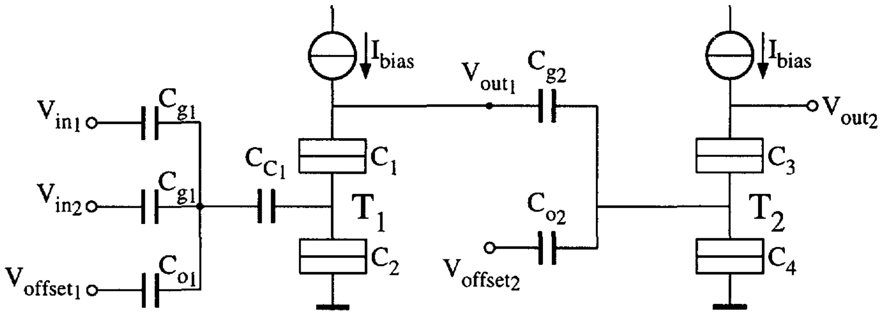


Figure 5.4: A two-SET transistor neuron.

where C_{Σ_1} is the total capacitance of the island of T_1 .

The output of T_1 serves as the input to T_2, as shown in Figure 5.6.b. It shows the output of transistor T_2 plotted on the horizontal axis as a function of its gate voltage V_{out_1} on the vertical axis. As with T_1, the range of its input value should correspond to half the transfer function period, and it should contain the negative slope. This results in the following relation between $V_{R_{\text{out}_1}}$ and the gate capacitance C_{g_2}.

$$V_{R_{\text{out}_1}} = \frac{e}{2C_{g_2}}. \tag{5.11}$$

The resulting transfer function from the input to the output of the cascade circuit is plotted in Figure 5.6.c. It features a sharp threshold with a slope equal to the product of the negative slopes of the individual SET transistors, and a saturation region on either side. The saturation region is not entirely monotonic, which may form an obstacle for the current learning algorithm.

Outside the region of interest, the transfer function is periodic, as indicated by the thin lines in Figure 5.6. The positive slope of the thin line equals the product of the positive slope of T_1 and the negative slope of T_2. At non-zero temperatures, the sharp corners are rounded off, and the graph in Figure 5.5 results. The position of the threshold can be changed by adjusting the offset voltage of T_1, as indicated by the dotted line in Figure 5.5.

5.4.2 Calculating the capacitor sizes

In this section, the capacitor sizes for a single SET cascade neuron are determined. The aim was to fabricate a single neuron to measure its transfer function. The design decisions are therefore based both on obtaining the desired function, and on the restrictions in the current production technique of SET devices. It

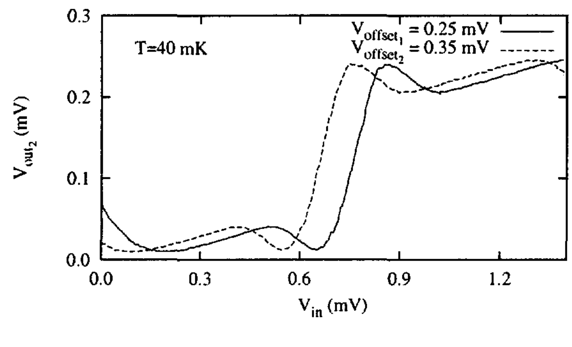


Figure 5.5: Transfer function of the cascade neuron.

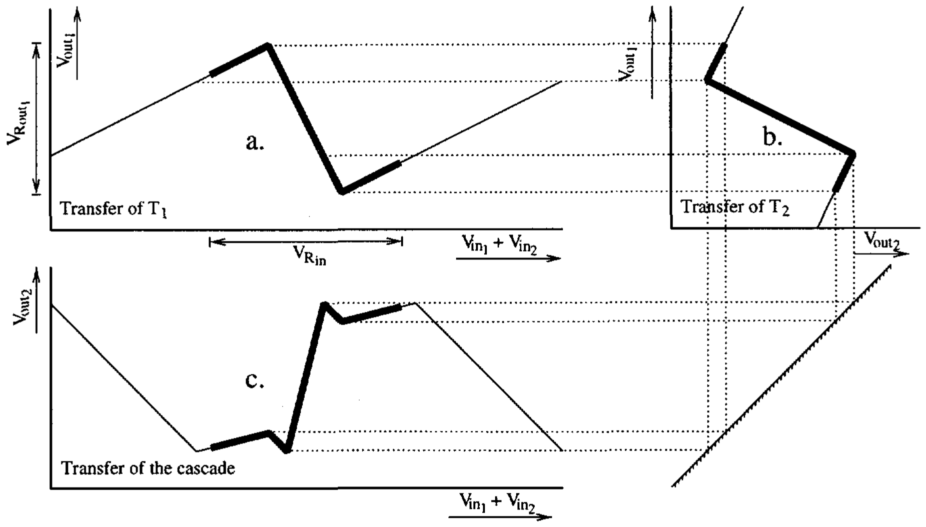


Figure 5.6: Construction of the transfer function of the neuron. **a.** transfer of T_1, **b.** transfer of T_2, **c.** transfer of the cascade.

was not optimized for connection in a large network. A picture of the realized neuron is shown in Figure 5.7.

The tunnel junctions

With the current production process, it is desirable that the tunnel capacitances of a SET transistor are designed to be equal. In other words,

$$C_1 = C_2, \tag{5.12a}$$
$$C_3 = C_4. \tag{5.12b}$$

To obtain a threshold with a steep slope, both SET transistors need voltage gain in their negative slope area, which is only the case if $C_g > C_1$. The gate capacitance C_g however cannot currently be made much larger than the tunnel capacitances [2, 35] because it requires a large island size, which increases the stray capacitance and offset charge fluctuations (see Page 33). A ratio of 2 is feasible, so that

$$\frac{C_{g_1}}{C_1} = 2 \tag{5.13a}$$
$$\frac{C_{g_2}}{C_3} = 2. \tag{5.13b}$$

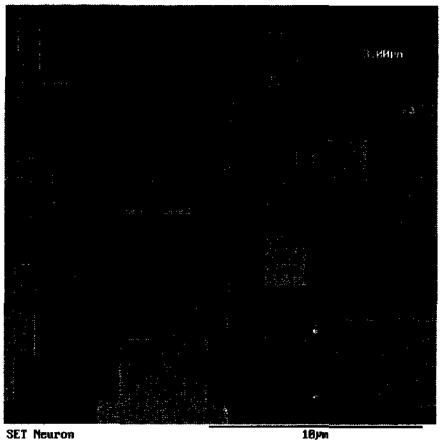


Figure 5.7: Micrograph of the cascade neuron.

This results in a negative slope of -2 for each SET transistor, and thus a slope of $+4$ for the threshold of the neuron. The positive slope of the transistors then equals

$$\frac{C_g}{C_\Sigma - C_1} = \frac{2}{7}. \tag{5.14}$$

The gate coupling capacitances

For the realization of a single neuron, without connections to other synapses or neurons, only the coupling between the two transistors T_1 and T_2 needs to be discussed. The output voltage range of T_1 needs to match the input period of T_2. With equations (5.10) and 5.11, we find

$$C_{g_2} = \tfrac{1}{2} C_{\Sigma_1}, \tag{5.15}$$

where C_{Σ_1} can be approximated as

$$C_{\Sigma_1} = C_1 + C_2 + C_{c_1} \tag{5.16}$$

if C_{g_1} and C_{o_1} are much larger than C_{c_1}, which is the case here.

Filling in the numbers

A currently common junction capacitor size is 100 aF. Filling in $C_1 = C_3 = 100$ aF in equations (5.12b) and 5.13b yields

$$C_{c_1} = 200 \text{ aF} \tag{5.17a}$$
$$C_{g_2} = 200 \text{ aF} \tag{5.17b}$$
$$C_2 = 100 \text{ aF} \tag{5.17c}$$
$$C_4 = 100 \text{ aF} \tag{5.17d}$$

For simplicity, $C_{o_1} = C_{g_1}$ and $C_{o_2} = C_{g_2}$. The simulation results of Figures 5.5, and 5.8 through 5.10 were generated with these parameters.

The three input capacitors, can be made arbitrarily large. When the neuron is part of a neural network, its size will be determined by the size of the synapses in front of the neuron, which is subject of Section 7.2.2.

The output range of this neuron is:

$$V_{\text{out}} = \frac{e}{C_{\Sigma_2}} = 300\ \mu\text{V}. \tag{5.18}$$

With a bias current of 16 pA, the static power consumption of the two SET transistors of this neuron is approximately 10 fW.

5.4.3 Sensitivity to parameter variations

The neuron transfer function shown in Figure 5.5 is only obtained when all the device parameters and offset voltages have the correct values. This necessity can easily be understood from Figure 5.6. If the period or output range of either device changes due to parameter variations of the SET transistor, or if the position of the transfer function of either SET transistor is shifted along its input axis due to offset charges, the shape of the overall transfer function changes.

As an example, the effect of offset charge fluctuations is illustrated by Figures 5.8 and 5.9. In both cases, the circuit of Figure 5.4 is simulated, but with other values for respectively V_{offset1} and V_{offset2}. It is clear that variations in V_{offset_1} only affect the position of the threshold, but that V_{offset_2} actually influences the shape of the transfer. Changing other parameters has similar consequences.

Offset charge fluctuations could in principle be adjusted by a learning algorithm that changes the voltage sources V_{offset1} and V_{offset2} as necessary. Learning algorithms would however have difficulty handing the radical function changes resulting from charge fluctuations of the second SET transistor, making this SET neuron circuit not very attractive.

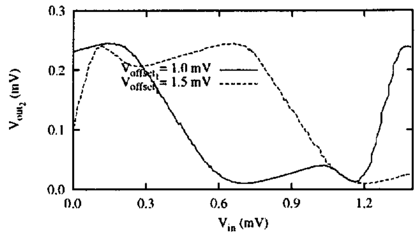


Figure 5.8: Changing V_{offset_1} only shifts the neuron transfer function.

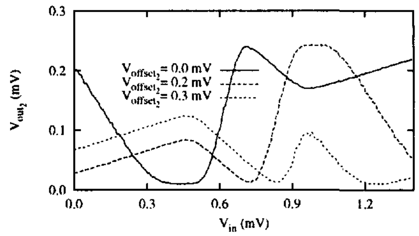


Figure 5.9: Changing V_{offset_2} changes the shape of the neuron transfer function

Variations in other parameter values should also be addressed because capacitor sizes and the like cannot be adjusted when the neuron is made. On the other hand, they are time independent so the learning algorithm has to adjust the network only once to the actual parameters. Variations of 10% of all capacitor sizes still gives an acceptable transfer function shape.

The sensitivity to temperature variations is shown in Figure 5.10. We see that the amplitude decreases and the peaks diminish, as expected, but the general shape is essentially the same. In fact, the smaller variations in the *low* and *high* regions for higher temperatures may even be beneficial for the network operation.

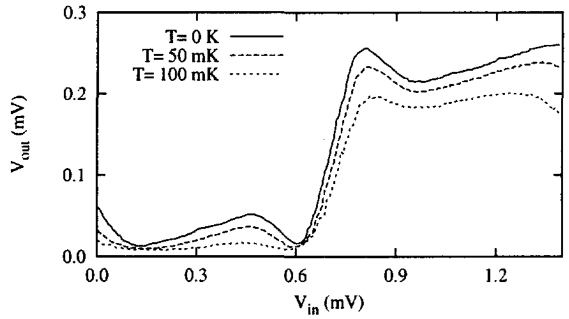


Figure 5.10: The effect of temperature change on the transfer function.

5.4.4 Conclusion

It is possible to create a transfer function with two saturation regions using two SET transistors. There are however unsolved problems that indicate that the cascade neuron is probably not the optimal SET neuron design. The presence of *two* offset charge adjustment points of which one drastically influences the activation function shape, is probably too complex for the learning algorithm. The effects of the non-monotonic saturation regions also remain to be investigated.

5.5 A one-SET transistor neuron

To design the smallest possible neuron using SET technology, it is important that the properties of the available devices are fully exploited. The potential for compactness of design increases accordingly

as more of the inherent properties of the technology are exploited, instead of being suppressed as unwanted parasitics. The compactness of the neuron implementation is of course enhanced by minimizing the hardware complexity of the neural device.

One of the most conspicuous properties of SET transistors is without doubt their periodic transfer function. Its ubiquity in SET transistor-based circuits makes its use almost compulsory for efficient designs.

A second property of SET devices is that the fundamental input quantity at the capacitively coupled island of the SET transistor is the electrical charge. As we have seen in Section 5.3.2, it is used by the capacitive addition circuit.

Other properties of the SET transistor that should be taken into account for circuit design are:

- inaccurate parameter values,
- offset-charge fluctuations,
- and weight-dependent offsets.

Of course, not all the properties of the devices can be used beneficially. The concerned properties can then perhaps best be described as non-idealities. The neuron implementation should be made robust against these potential non-idealities. Either inherently, by the neuron circuit itself, or with aid of the rest of the neural network, such as the synapses and the learning algorithm.

Except for being robust against non-idealities in the neuron itself, the neuron circuit should in turn help to compensate for the non-idealities in the other parts of the circuit.

The basic functions to be performed by the neuron are to collect the output signals from the synapses and perform a non-linear operation with classifying properties on it. Since in principle a single SET transistor performs these basic operations, it is worth investigating whether a single device can operate as a neuron.

5.5.1 Using the periodicity of the transfer function

If the input range of a SET transistor is limited, only a restricted number of periods of the transfer function is actually used. This is the case when the input signal is generated by another SET transistor because SET transistors have a limited output range (Section 4.3.1). The number of transfer function periods spanned by the neuron input range is determined by the gate capacitors of the inputs, and can therefore not be adjusted after circuit manufacture. There are three possibilities for the ratio α_n between the neuron input range $V_{R_{in}}$ and the period size V_p, as was illustrated by Figures 4.20 through 4.25.

$$\alpha_n = \frac{V_{R_{in}}}{V_P} \tag{5.19}$$

If $V_P \ll V_{R_{\text{in}}}$, the resulting function is approximately linear, and thus looses its non-linear, classifying properties. It is not advantageous compared to leaving the activation function out altogether. $V_p \ll V_{R_{\text{in}}}$ is therefore unsuitable for implementing the activation function.

If $V_p \approx V_{R_{in}}$, the resulting function divides the input space into two or more regions each belonging to either of two classes: *low* or *high*. The number of regions depends on the exact ratio between V_p and $V_{R_{in}}$. Division of the input space into classes is exactly what is required to make a classifying cell. In contrast to most traditional activation functions, the SET neuron does not have saturation regions. Since the ranges of all signals in the neural network are limited by the periodic nature of the SET transistor transfer function, the neuron output range is nonetheless restricted. The exact value of this range depends on the device parameters and the temperature.

When the number of regions is very large, in other words if $V_P \gg V_{R_{in}}$, the activation function chops the input space into a large number of small segments, with each consecutive segment alternately belonging to one of the two classes. This is in principle a classifying operation, but whether it can be used as a neural activation function remains to be seen.

These types of activation function may require learning strategies other than sigmoid neurons, similar to what is shown in [41], where a new error function is used for the non-monotonic Gaussian activation functions in a feed-forward neural network with the generalized delta learning rule. The reason for introducing a new error function is that gradient descent learning algorithms with conventional error function definitions only accept monotonous and continuously differentiable activation functions.

The same paper reports that the use of Gaussian activation functions instead of sigmoidal ones yields faster convergence with less neurons. This is confirmed and extended to periodic activation functions in [46], where it is shown that sinusoidal neurons can learn a variety of functions such as parity problems, and encoding functions with more than one output. Convergence is significantly faster and with less neurons than for networks with sigmoidal neurons. Similar results are reported in [43], where it is concluded that networks with periodic neurons are less susceptible to additive noise, and that less neurons are needed to solve the same problems. Back propagation is used to teach a feed-forward network, but the need for a different error function definition is not mentioned.

A big advantage of periodic functions is that compensation for unwanted offsets of *any* size can be guaranteed by a compensation signal range of at most the period size, because any offset is always folded back to this range.

5.5.2 The device parameters

The degrees of freedom left for the design of a circuit consisting of a single device are the values of the device parameters. The device parameters determine the transfer function shape of the device and should therefore be discussed in detail.

The ratio α_n for which $V_p \approx V_{R_{in}}$ looks most promising for a neuron activation function and is therefore analyzed further in the remainder of this section. The exact value of the ratio is not determined here because the tolerances on the device parameters and the ambient temperature influence this ratio considerably. The performance of the neuron should therefore not be dependent on the exact value for successful operation.

The input voltage range $V_{R_{in}}$ of the neuron is determined by the outputs of the synapses connected to the neuron and by the number of synapses present. The required value for the neuron gate capacitors C_g is determined by both the neuron and the synapse circuit, and is therefore postponed to Section 7.3.2 where the interaction and interconnection between the synapses and neurons is examined.

The same holds for the output voltage range of the neuron, which is determined by the total island capacitance C_Σ. Its required value is among other things dependent on the input stage of the synapses in the next layer of the neural network.

We can at this moment decide on the slope values of the neuron because they can be designed independently of C_g and C_Σ by sizing the tunnel junction capacitances. There are three aspects to consider when choosing the slope values:

- classification performance transfer function,
- production of the tunnel junctions,
- and gain of the transfer function.

Classification performance

The two slopes characterize the boundary between the regions belonging to the *high* and the *low* classes of the neuron. The discrimination ability between two adjacent points in the input space of a neuron is larger for steeper slopes of the activation function. This implies that steeper slopes of the SET transistor transfer function are to be preferred. The two slopes however, cannot be adjusted separately if the output voltage range and the input period are to remain constant (see Section 6.3.1). A large negative slope automatically results in a small positive slope, as illustrated by Figure 5.11. This means that the

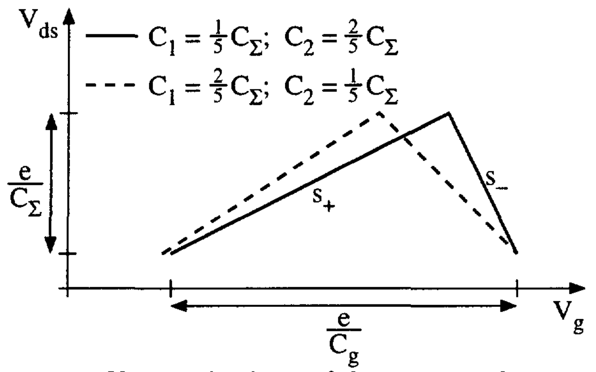


Figure 5.11: Varying the slopes of the activation function.

discrimination ability for classification at one boundary could increase at the expense of the other boundary. For large values of the negative slope, the discrimination ability on the positive slope is not very sensitive to improvements on the negative slope. A side effect of changing the transfer function slopes is a slight change in the size of the two classes. A steeper negative slope results in a smaller *high* region.

Manufacture of the tunnel junctions

The two slopes are determined by capacitance ratios, which can be changed by adjusting the tunnel junction capacitances. In theory, that is. In practice, the tunnel junction capacitance sizes cannot be manufactured accurately yet, and one should be prepared for large tolerances. A successful neural network should be robust against these variations. The network should adapt to the actual activation function by self-adjustment using a learning algorithm. Moreover, the values of the tunnel junction capacitances cannot be chosen freely currently. Only tunnel capacitances of approximately the same size can be fabricated reliably.

Gain of the transfer function

A symmetric neuron transfer function cannot have gain, which would limit its applicability to feed-forward topologies (see Section 4.3.7).

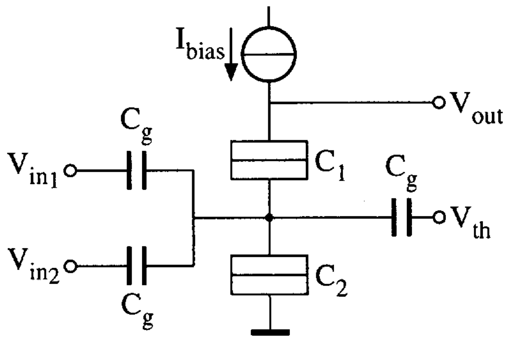


Figure 5.12: The one-SET transistor neuron device.

With both tunnel junctions equal to C, and gate capacitances of twice that value, the following dimensions are obtained for the SET neuron of figure 5.12:

$$C_1 = C \tag{5.20a}$$
$$C_2 = C \tag{5.20b}$$
$$C_g = 2C \tag{5.20c}$$
$$C_\Sigma = 3C_g + C_1 + C_2 = 8C \tag{5.20d}$$
$$s_- = -2 \tag{5.20e}$$
$$s_+ = \tfrac{1}{2} \tag{5.20f}$$

We will see in Chapter 7 that a neuron with these parameters can classify.

5.5.3 Inaccuracy of the parameter values

With the current production process for SET transistors, parameters that characterize the devices are not accurately known. Although improvements are expected in the future, some variations will always remain and can fortunately be dealt with by the neural network. The effects parameter variations have on a single-SET transistor neuron are listed below.

- Inaccurate tunnel resistance
  Affects the weight-dependent output voltage of the device. Too low values results in device failure (see Section 4.2.5). Tunnel resistance is difficult to control because it is exponentially dependent on the oxide thickness.

- Inaccurate tunnel capacitance
  Affects the slope of the transfer functions, together with gate capacitance. Also plays a role in the output signal range through C_Σ.

- Other inaccurate capacitances
  The other capacitors present include the gate capacitance C_g, and parasitics such as stray and interconnection capacitances. Here, we ignore the stray capacitance. The capacitive load of interconnections to other network elements only affects the bandwidth (Section 4.3.4) and is not considered here either. The magnitude of the period at the input of a device depends on C_g so it results in uncertainties in the period size. It also affects the output signal range through C_Σ.

These inaccuracies at the device and circuit level can be adjusted for at the system level if the system is self-learning. It is for this reason that self-adjustment is compulsory in large systems containing inaccurate devices.

The weight-dependent output voltage of the device is fully compensated for by offset adjustments in the next layer of the neural network. The transfer function periodicity ensures offsets of any size can be adjusted within the range of a single period. Device failure resulting from extreme tunnel junction resistances must be dealt with by built-in redundancy.

The transfer function shape inaccuracies resulting from inaccurately known capacitances must be dealt with by adjusting the synaptic weights and neuron thresholds. This is only possible within certain boundaries. To correct a too low slope with a higher synaptic slope is only possible as long as the maximum available synaptic slope is high enough.

5.5.4 Offset-charge fluctuations

The offset-charge fluctuations present in all SET transistors manifest themselves in this case as a random value added to the sum of the neuron inputs, projecting this sum at an unpredictable position on the input axis of the periodic transfer function. This is equivalent to an unwanted change in the neuron threshold value.

In conventional neural networks, the threshold value is often made adjustable by including an additional synapse with unity input value at the neuron input [40]. Adjustment of the threshold value is thus equivalent to adjustment of the additional weight, and can be handled by the learning algorithm

Provided that the learning algorithm is able to correct the threshold value of the activation function before the next offset charge fluctuation changes it again, the learning algorithm is expected to be capable of compensating for the offset charge fluctuations.

5.5.5 Weight-dependent offsets

As we have seen, the hardware introduces non-idealities in the neuron. In turn, the neuron can also help to reduce the effects of hardware non-idealities in other building blocks.

The output signal of the SET transistor synapse which is described in Chapter 6, has a weight-dependent offset voltage whose value depends on the synaptic weight (see Section 6.3.1). Each synapse connected to the input of a neuron therefore contributes some value at the neuron input. The total offset it results in may attain relatively large values and could seriously disturb the neuron threshold value if it is not compensated for.

To compensate for these offsets, the learning algorithm must adjust the neuron threshold value. The required adjustment range of this input is always equal to the period size, independent of the magnitude of the offset present at the neuron input because the neuron transfer function is periodic. Whether a learning algorithm can be developed that is indeed able to do this kind of adjustment is subject of Chapter 8.

5.6 Other neuron circuits

There are two other neuron circuits with SET devices described in the literature at the moment. Both are based on the 'inverter' circuit originally proposed in [25] and shown in Figure 5.13. In Section 5.7, the SET neurons proposed in this thesis are compared to the two designs described below.

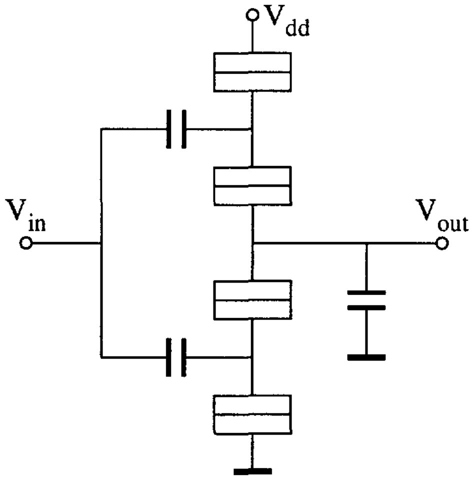


Figure 5.13: SET inverter circuit.

5.6.1 Inverter neuron

Kirihara and Taniguchi [36] proposed neural hardware for the synaptic multiplier and the neuron activation function, using one SET inverter for each. The multiplying operation is obtained by a weight-dependent voltage on the ground terminal of the inverter, which is copied to the load capacitor when the input is active. The activation function is generated from the non-linear transfer function of the SET inverter. The use of SET inverters in this way assumes the absence of offset-charge fluctuations.

5.6.2 Stochastic Boltzmann neuron

In [55,56], Akazawa and Amemiya describe how to change the circuit parameters of a SET inverter so that its stability diagram contains an unstable region. In that region, the inverter constantly alternates between the possible internal states. The probability of finding the circuit in a certain state depends on the voltage present at its input. This device is proposed for implementing the stochastic-response unit of a Boltzmann machine neuron. Its behavior however relies on precise adjustments at the input to obtain the stochastic response, and its success depends on the absence of offset charge fluctuations.

5.7 Discussion

Two single-electron tunneling neurons have been described in this chapter, namely the SET cascade neuron with two SET transistors, and the SET neuron device with one SET transistor. Two other SET neurons [36,55,56], built with a so-called SET inverter have been mentioned briefly. The main difference between the neuron designs lies in the way the activation function is generated. The addition function principle is the same for all of them. In this section, the neurons are compared.

5.7.1 The neurons with two SET transistors

Both neurons with two SET transistors, the 'cascade' neuron and the 'inverter', have a relatively steep threshold function, which is an advantage for the use in neural networks because it gives the neuron good discriminatory properties. The 'cascade neuron' features a simple way to adjust the threshold value, namely by modifying the offset gate voltage of the front SET transistor. This is useful because neural learning algorithms usually rely on adjusting this parameter.

An important advantage of the complementary structure of the inverter-based neurons is said to be their zero static power dissipation [36]. Some remarks can however be made about this. First of all, if the offset charge fluctuations remain present in SET technology, all SET islands in a large neural system of inverter neurons will have to be adjusted, which is rather impractical. Since the low power dissipation is based on complementary operation, which relies on well-adjusted offset charges, it is difficult to achieve zero static power dissipation in large systems.

Second, at 10 kHz the dynamic power dissipation of the inverter neuron is equal to the static power dissipation of the cascade neuron scaled down to the same size as the inverter and biased at 1 pA. So at operating frequencies well above 10 kHz, the static power dissipation of the single SET transistor becomes negligible in comparison to the dynamic power dissipation. Since the intrinsic speed of a SET transistor is in the order of 10 GHz, operating frequencies well above 10 kHz are to be expected.

The 'inverter' neuron has the advantage of combining voltage bias with an output voltage, as opposed to the current biased 'cascade' neuron and the single SET neuron device. Voltage bias is more practical to implement than current bias because a voltage source is easier to distribute across the chip to all neurons, and easier to generate.

Neurons with two SET transistors also have other disadvantages, like required matching of the devices and the required voltage gain. The most critical disadvantage however, is probably the presence of two SET islands. Both are afflicted with random offset charges, which need to be compensated to obtain the specified behavior. The fact that there are two such adjustment points makes automatic adjustment, for example with the learning algorithm, more complicated. Moreover, the complementary behavior of the 'inverter' neuron on which its low power consumption relies, only works when the two offset charges are both adjusted optimally. Offset charge adjustment requirements form a larger impediment to VLSI neural networks than static power consumption.

5.7.2 The one-SET transistor neuron

The one-SET neuron device has some important advantages when compared to the two-SET versions. It is about half the size (not counting the bias circuitry), and it only has one offset charge adjustment point because there only is one SET transistor. This single adjustment point also has the function of threshold value adjustment, which is familiar to the neural learning algorithm. This means that offset charge fluctuations are mapped onto standard neural network behavior, and do not need to be adjusted separately. The independence from the ubiquitous offset charge fluctuations is probably the key to the successful large scale application of SET technology.

The less steep threshold function compared to the two-SET variants and the absence of saturation regions in the one-SET neuron transfer function does not hamper the classification abilities of the device, nor its learning abilities, as will be shown in Chapters 7 and 8.

5.8 Conclusions

In this chapter we have seen that it is possible to make a neuron in several ways using SET transistors. It is possible to generate an activation function similar to the standard sigmoidal function using two SET transistors. Two other neuron designs, both based on the SET inverter with two SET transistors are known in the literature. The most important problem with the designs based on two SET transistors is that they have two offset charge adjustment points, and that the transfer function shape is drastically different when the offset charges are not compensated for. This puts an extra burden on the learning algorithm.

The smallest possible neuron in SET technology, with only a single SET transistor, only has a single offset charge to adjust. It is expected the learning algorithm can cope with it because it has the effect of changing the neuron threshold, and adjustment of this threshold is a regular learning algorithm task.

The periodicity of the SET transistor transfer function gives the one-SET neuron the ability to cope with any weight-dependent offset value at its input because it can always be compensated for with an adjustment range equal to the period size. This neuron promises to be a successful neural building block.

6

A SET Synapse

The synapses of a neural network form the collective, distributed memory of the system. The signal processing task of a synapse is straightforward. It determines the connection strength between two neurons, based on the value of the synaptic weight. The value of the synaptic weight is adjusted by a learning algorithm until the network has the desired behavior.

Because of the large number of synapses in a neural network, it is essential that the individual synaptic cells are as small as possible.

In Section 6.1, the properties that are important for a successful synapse are analyzed and, based on that, Section 6.2 gives the basic concept of an extremely small synapse with a single SET transistor. The design of a SET synapse is described in detail in Section 6.3.

6.1 The essential properties of a synapse

In order to design a compact synapse, it is necessary to first establish which properties are absolutely indispensable for correct operation. The design effort can then be focused on specifically implementing those properties.

6.1.1 The synapse transfer function

The functional role of a synapse in a neural network is to modify the connection strength between two neurons or between a network input and a neuron. The most common situation can be described by the mathematical multiplication operation in which the output of the synapse equals the product of its weight and input. The resulting transfer function from input to output is a straight line through the origin, with a slope adjustable with the weight, as illustrated by Figure 6.1. The most complete implementation of this multiplier is one that

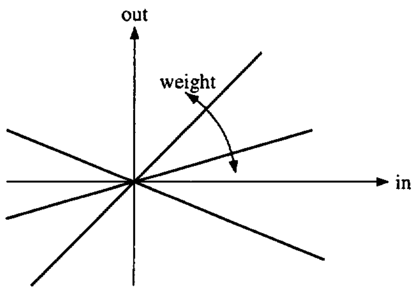


Figure 6.1: Transfer function of an ideal multiplier.

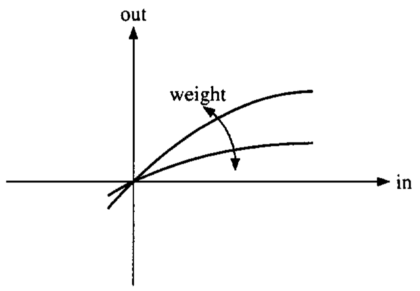


Figure 6.2: Non-linearity in a multiplier transfer function.

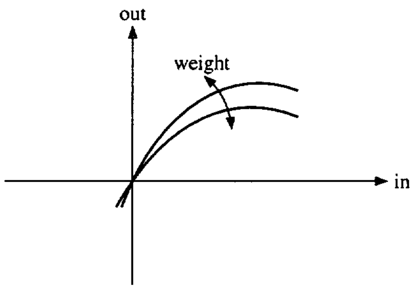


Figure 6.3: Non-monotonicity in a multiplier transfer function.

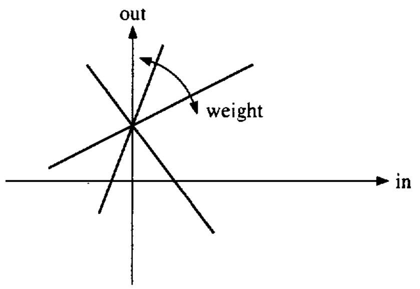


Figure 6.4: Constant offset in a multiplier transfer function.

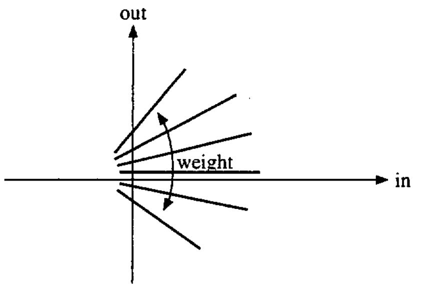


Figure 6.5: Weight dependent offset in a multiplier transfer.

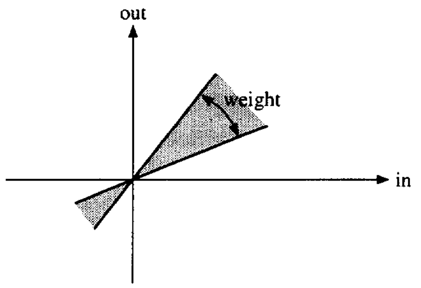


Figure 6.6: Restricted slope range in a multiplier transfer function.

can operate in all four quadrants so that the output and the input can have positive, as well as negative values. The four quadrants are however not required for all types of neural network. The required polarity at the synapse output is determined by the neuron input stage, while the polarity at the synapse input should be compatible with either the signals produced by the neuron outputs, or by the network input.

The polarity of the synaptic weight tells whether the synapse is excitatory (positive valued weights) or inhibitory (negative valued weights). The learning algorithm of the network often dictates whether synapses should be excitatory, inhibitory, or both. Back-propagation learning, for example, requires synapses with positive and negative weights, while some forms of the unsupervised Hebbian learning rule use either only inhibitory or only excitatory ones [57, 58]. Synapses that can be both are however used most.

Synapse circuit designs are often driven by imitating as closely as possible the ideal linear multiplying operation because the learning algorithms used calculate the weight updates based on, among other things, an ideal multiplier as synapse. An ideal multiplier is however not so easy to make. The resulting circuit is too large, complicated, and therefore other learning algorithms with less stringent requirements have to be devised.

6.1.2 Non-idealities in a synapse transfer function

There are four types of deviation from the ideal multiplier that can be present in a synapse implementation. They are non-linearity, non-monotonicity, offset, and a restricted slope range. The question that should be asked is to what extent they are still acceptable for a successful synapse. It is an important question because unacceptable non-idealities should be avoided to prevent obtaining an unusable synapse (obviously), and because no effort should be put into correcting any deviation from the ideal multiplier that does not hamper correct network operation. Designing a non-ideal multiplier instead of an ideal one can lead to significantly smaller synapse circuits [9, 59].

Non-linearity of the transfer

Non-linearity of the synapse transfer function (Figure 6.2) corresponds to the weight being dependent on the input value. It deforms the straight boundaries of the neuron threshold, which results in non-uniformities in the input space of the neuron. The learning algorithm should be able to correct for these non-uniformities because it is just a deformation of the input space. Classification remains in principle possible by appropriately adjusting the complete network.

Limited input or output range

In general, the signal range is one of the two factors (the other factor being noise) that determine the dynamic range. Because the number of distinguishable states or patterns in the signal is proportional to the dynamic range, fewer patterns can be distinguished if the range is limited. It is therefore desirable to have a high dynamic range.

The input range of the multiplier limits the pattern input space of the synapse. It should match the network input range or neuron output range that drives this synapse. The output range of the synaptic multiplier should match the range available at the neuron input for optimal performance.

Non-monotonicity of the transfer

The effect of a non-monotonous synapse transfer function (Figure 6.3) is that the mapping function maps different parts of the input space onto the same point in the output space. This results in classification problems if the points in the input space belong to different patterns: the synapse would consistently map them onto the same position in the output space, making it impossible to separate them. The input space is folded by the neuron, and it is therefore not obvious that a neural network can cope with non-monotonicity.

Periodicity of the transfer

Periodicity is a special form of non-monotonicity. It is mentioned here because SET transistors have a periodic transfer function. Whether periodicity in the signal input space of the synapse can find an application is not very obvious, and deserves a separate investigation. In this thesis, it is avoided by restricting the input range to a small fraction of the transfer function period.

The periodic transfer function of the SET transistor is however exploited in an other way for the synapse, as is explained in Section 6.3.2.

Offsets

Offsets in the synaptic multiplication function appear as additional terms in the multiplication function. These terms can be constant, or depend on the input signal, the weight, the ambient temperature, supply voltage, or time.

Offsets that do not depend on the weight or input signal result in a translation of the input, output, or weight space. Figure 6.4 gives an example of a translation in the output space. The learning algorithm is capable of correcting for this type of non-idealities, for example by adjusting the weights or neuron thresholds.

Offsets that depend on the input signal cause non-linearities in the mapping function. This was discussed separately above.

Finally, offsets that depend on the weight value can result in scaling errors, as described separately below. It may also result in weight dependent offsets at the neuron output, as illustrated by Figure 6.5. In that case, the input space translation changes as the weight changes.

Restricted slope range

A restricted slope range (Figure 6.6) limits the available connection strengths of the synapse. The effect can be discussed separately for the lower boundary and the upper boundary (both in absolute values). If the slope range is limited at the lower boundary, a slope of zero cannot be achieved. This means that the connection represented by this synapse cannot be switched off completely. If in that case switching off a connection is desired, it would have to be compensated for by another connection with the opposite behavior.

If the maximum connection strength is limited by an upper boundary, the synapse may not be able to span the full input range of the neuron on its own. That means that the contribution of several synapses is required to let the neuron change state.

The upper and lower boundaries may have different values for the positive and the negative slopes.

To cope with these non-idealities, the learning algorithm should be capable of distributing the signal contributions to the neurons to all the synapses connected to it, and to deal with asymmetric weight strengths, which may prove to be very difficult.

Weight imprecision

An imprecise weight results in uncertainty in the actual multiplication factor. A learning algorithm that adjusts the weights by measuring the actual error produced by the network automatically corrects for this type of non-ideality.

Weight leakage

In practical situations dynamic weight storage is often afflicted with some form of leakage. The weight does not retain its information indefinitely. It could also be called 'forgetting' if we stay in neural terminology. An obvious remedy to forgetting is re-learning. Slow leakage should therefore not be detrimental to neural networks that periodically relearn. An alternative to relearning is weight refreshing by downloading the weight information from an external system. This is rather impractical for large systems, and is therefore not dealt with in this thesis.

Weight quantization noise

The presence of a minimum weight-change step results in quantization of the synaptic weight value. In digital implementations of the weight storage, the minimum step is determined by the number of bits, and in an analog implementation of charge stored on a capacitor, the quantization of charge limits the step size. Quantization of charge on a capacitor becomes apparent when the electrostatic energy $E = \frac{e^2}{2C}$ of one electron on a capacitor is larger than the thermal energy kT (see Section 4.2.5). We will see in Section 6.3.2 that the periodic transfer function can be used to obtain much smaller steps.

Weight quantization noise is a serious problem in neural networks because it limits fine tuning of the weights during learning and can lead to convergence problems.

Signal noise

Thermal noise is always present at the output of a SET transistor if its temperature is non-zero. This thermal noise is responsible for rounding off the sharp maxima and minima of the transfer function, and as such is indispensable for the synapse operation. This rounded-off shape is an *average* in time, which is therefore only obtained by filtering the high-frequency components responsible for the deviations from this average.

6.2 Concept of a SET synapse device

The implementation of a synapse should be a minimum size circuit with a transfer function whose slope can be adjusted by a learning mechanism, and whose operating point can be stored. This section describes how these elementary properties of a synapse can be implemented with a single SET transistor [53].

As illustrated by Figure 6.7, the operation of the synaptic device is based on the small-signal behavior of the SET transistor. For small-signal variations, the transfer function slope is approximately constant and its value depends on the operating point, as illustrated by Figure 6.8.

This means that if the input voltage range of the device V_{R_i} is much smaller than the period V_p, the transfer function slope depends on the operating point, which is given by the island charge of the SET transistor (see Section 4.3.1). Figures 6.10 and 6.9 illustrate this for different values of the island charge. The large-signal island charge adjustments represent the synaptic weight. The concept relies on the gradual transition between the positive to the negative slope at temperatures higher than 0 Kelvin. The required temperature depends on the total capacitance of the SET transistors used. The smaller the capacitance, the higher the temperature.

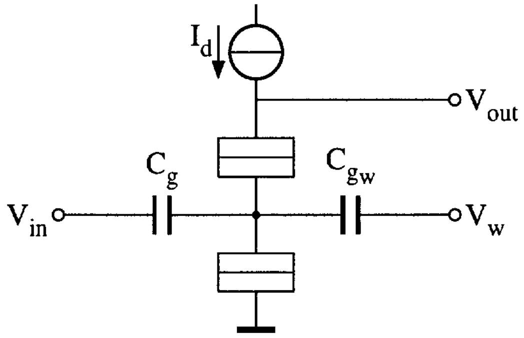


Figure 6.7: The SET synapse device

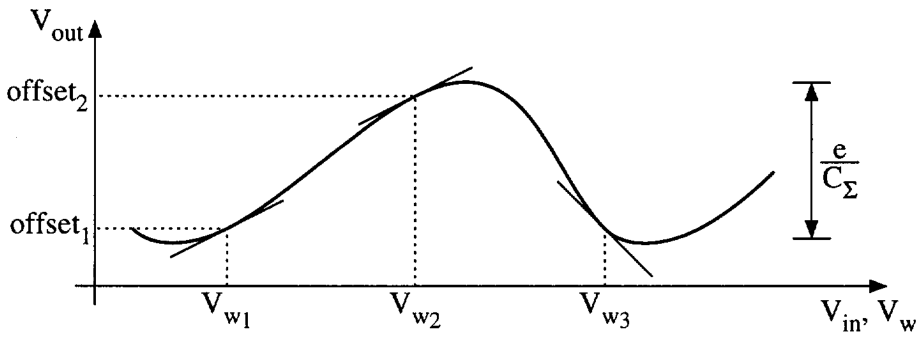
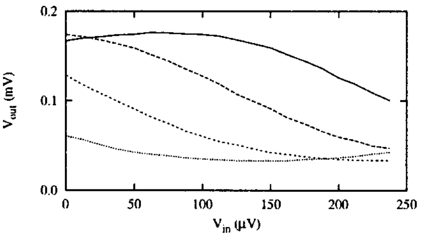


Figure 6.8: Small-signal behavior of a SET transistor for three values of the weight V_w.



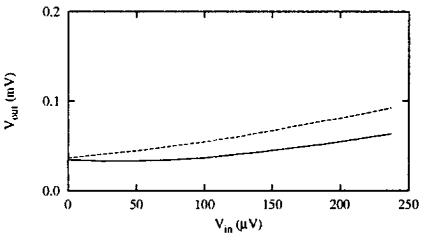


Figure 6.9: Transfer function of the SET synapse device for negative weights.

Figure 6.10: Transfer function of the SET synapse device for positive weights.

If a SET transistor is used in this way, it has the elementary characteristics of a synapse. The slope of its transfer function is modified by changing the island charge of the transistor. The slope can have both positive and negative values so that the synapse can be both inhibitory and excitatory. The island charge of the device is adjusted by a second capacitive gate, which can be used to connect the synaptic weight (see Figure 6.7). On first sight, the addition of weight and the input signal at the synapse island may appear to have little in common with a multiplier. The non-linearity however of the SET transistor transfer prohibits application of the superposition principle. It is this non-linear character on which the synaptic multiplier is based.

It is shown in Chapter 7 that this SET synapse can drive the SET neuron of Chapter 5.

6.3 Implementation of a SET synapse device

The synapse consists of three functional blocks, as illustrated by Figure 6.11, which are described separately below. In Section 6.3.1, the effects of the specific SET transistor properties on the synaptic multiplier are analyzed. Section 6.3.2 describes how the synaptic weight can be stored, and Section 6.3.3 argues that there is need for further research on the currently available circuits to modify the weights.

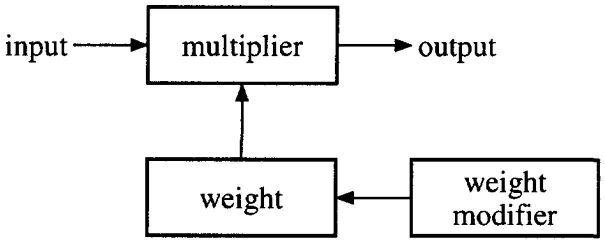


Figure 6.11: Three functional blocks can be distinguished in a synapse: the multiplier, the weight storage, and the weight modifier.

6.3.1 The multiplier

The synapse transfer function as shown in Figures 6.9 and 6.10, is clearly not an ideal linear multiplier. Therefore, this section starts with an analysis of the properties of this synapse. Following the list of deviations described in Section 6.1.2, the non-linearity, non-monotonicity, offsets and range of slopes are discussed. In addition, other SET-specific features such as the implications of the transfer function periodicity, the output range and random offset charge fluctuations are analyzed.

Transfer function non-linearity

The degree of non-linearity of the synapse transfer function depends on two factors. First, it depends on the temperature. At lower temperatures, a SET transistor transfer function approaches the piece-wise-linear shape, as illustrated by Figure 6.12, which shows simulation results of a SET transistor at temperatures of 0, 200 and 400 mK.

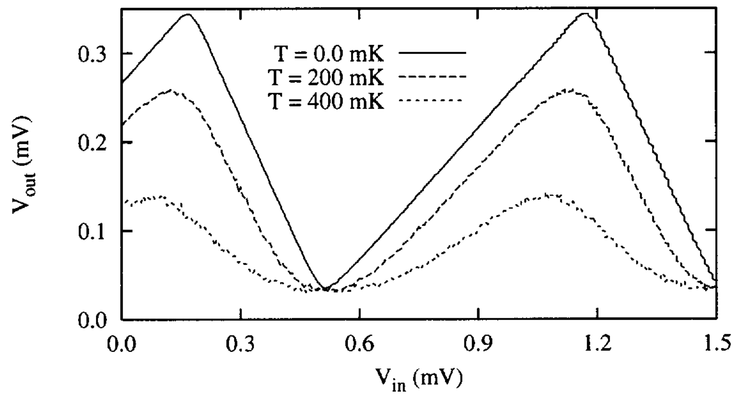


Figure 6.12: Transfer function of a SET transistor at three different temperatures. The output range and the maximum slope decrease with rising temperature.

The thermal noise increase associated with an increase in temperature has an important effect on the available slope values of the transfer function. The theoretically straight lines approach a sinusoid at higher temperatures. This effect is used to generate the range of slopes necessary for the synapse device. The synapse would not work properly if the device were at 0 Kelvin. Of course this also means an increase in thermal noise, which must be suppressed by time averaging (i.e. working at lower frequencies). The maximum attainable slopes that are discussed below are therefore unreachable theoretical maxima.

The synapse non-linearity also depends on the portion of the SET transistor transfer function period used. A larger portion gives more non-linearity. This portion is expressed in the ratio α_s between the SET transistor transfer function period V_{p_s} and the range of its input V_{R_n}:

$$\alpha_s = \frac{V_{P_s}}{V_{R_n}}. \tag{6.1}$$

Since non-linearity of the synapse transfer is in itself not a problem for the neural network, it leaves us free to fit the ratio α_s (within certain limits) to other needs, elsewhere in the network, and does not prevent increasing the temperature for a more gradual transition from positive to negative slopes.

Transfer function non-monotonicity

Making the ratio α_s too large results in non-monotonicity of the synapse transfer function because the positive and the negative slope may become part of the transfer function simultaneously. Non-monotonicity makes the application of the synapse more complex, but it cannot be avoided completely for small slope values because slope values around zero only occur at the maxima and minima, at a transition from a positive to a negative slope. Reducing the value of the ratio α_s reduces the degree of this non-monotonicity.

Weight dependent offsets

When the operating point of a synapse is changed by adjusting the weight, not only the slope of the transfer function changes, but also the signal independent value at the output, as can be seen from Figure 6.8. It can be modeled by a change of the neuron threshold value, and can therefore in principle be compensated for by the neuron (Section 5.5.5). Any weight correction that was meant purely to change the slope of the synapse must be accompanied by an adjustment of the neuron threshold to compensate for the level change. This might be difficult for some types of learning algorithm.

The transfer function slopes

The range of slopes that can be produced by the SET synapse device is limited by the maximum gradients of the SET transistor transfer function. The limit is different for the positive and the negative slope. Especially the positive slope range is small.

With fixed device dimensions, a lower ambient temperature has three important effects on the synapse transfer function.

- The transition region between the maximum positive and negative slope decreases. This reduces the range of weight voltages for which the slope actually changes.

- The output voltage range of the device increases, which increases the value of the maximum positive and negative slope.

- The thermal noise level of the output voltage decreases. Together with the previous item, it results in an increase in the dynamic range.

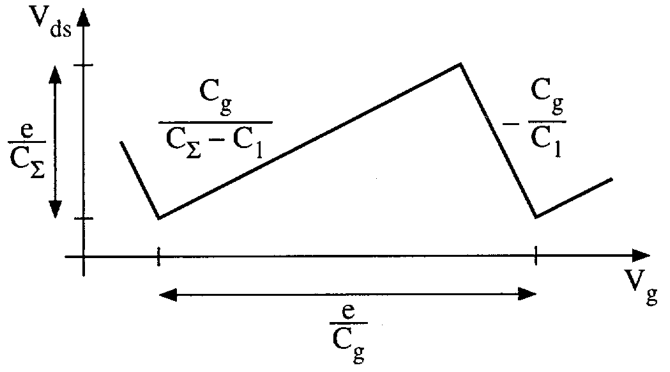


Figure 6.13: The transfer function of a SET transistor at 0 Kelvin.

As the device temperature approaches zero Kelvin, its transfer function approaches the piece-wise linear function of Figure 6.13 which gives the maximum obtainable synapse slopes. Straightforward electrostatic energy calculations yield the following equations for the four relevant quantities [21].

$$\text{Output range :} \quad V_R = \frac{e}{C_\Sigma} \tag{6.2a}$$

$$\text{Period :} \quad V_P = \frac{e}{C_g} \tag{6.2b}$$

$$\text{Maximum positive slope :} \quad s_{+_{\text{max}}} = \frac{C_g}{C_\Sigma - C_1} \tag{6.2c}$$

$$\text{Maximum negative slope :} \quad s_{-_{\text{max}}} = \frac{C_g}{C_1} \tag{6.2d}$$

The slopes of the transfer function of a SET transistor are dependent on the ratios between the various components of the island capacitance C_Σ. To change the slopes while keeping C_Σ constant, capacitance should be exchanged between the capacitors contained in C_Σ. To simplify the argument, it is assumed here that there is only one gate present, and that the parasitic stray capacitors can be ignored. The total capacitance C_Σ is then shared between C_g, C_1, and C_2 (see Figure 6.14).

The maximum and minimum values of attainable slope can be found by determining the extreme values of $(1 - \frac{C_1}{C_\Sigma})$, which expresses the position of the transfer function top within one period, as illustrated in Figure 6.15.

The top of the transfer function is located to the right end of the period when C_1 is given the smallest possible part of C_Σ, and the rest is shared by C_2 and C_g. In this case, the absolute value is maximal for the negative slope, while it is minimal for the positive one. Alternatively, giving more of the total

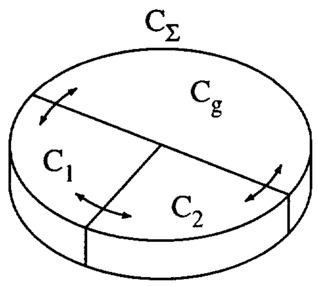


Figure 6.14: The total capacitance C_Σ is shared between C_g, C_1 and C_2. With the current production process, $C_1 = C_2$ is desired.

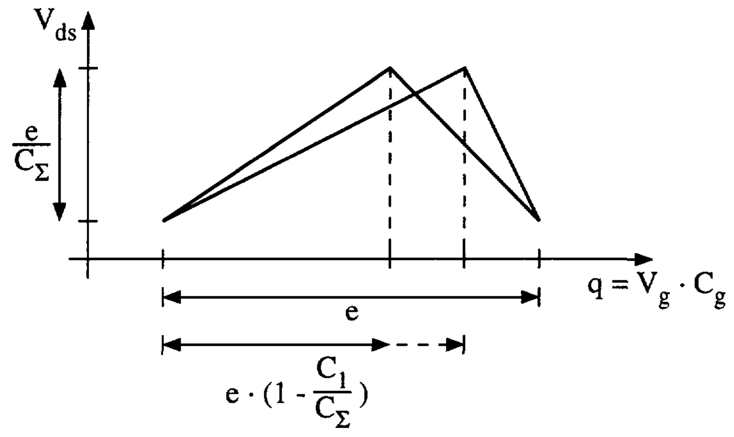


Figure 6.15: With a constant output voltage range, the position of the transfer function maximum determines the value of the slopes.

available capacitance C_Σ to C_1 shifts the transfer function top to the left, which decreases the absolute value of the negative slope, while increasing the positive one.

We are not free to position the top wherever we like because of the following boundary conditions. Except for the minimum capacitor value dictated by the technology, the tunnel capacitances C_1 and C_2 should be approximately equal while the gate capacitance also determines the ratio between the period and the output range of the neuron device of the previous layer and cannot therefore be chosen freely. Moreover, the technology does not provide accurate capacitor values at the moment.

It is difficult to describe definite limits for the slopes because they depend on continuously improving technological manufacturing methods. Therefore, in the following discussion we restrict ourselves to general indications. It is assumed that technological developments will allow a slight relaxation in the future of the requirement that $C_1 \approx C_2$.

To begin with, the positive slope cannot be made larger than 1, while the negative slope can be made smaller than -1 by making $C_g > C_1$ (see equations (6.2c) and (6.2d)). Is it possible to make the two slopes equal? In principle it is, it translates into $C_g + C_1 = C_2$. If C_g is relatively large compared to C_1 however, the difference in size between C_1 and C_2 is also large, which is not feasible with today's technology. So if a symmetric transfer function is desired, C_g should be made relatively small compared to the two junction capacitances. To make the positive slope have a larger absolute value than the negative one, the same argument as for equal slopes holds, but in a more extreme manner.

A negative slope larger than the positive one (as an absolute value) is more natural to make because the two tunnel junctions can be of the same size. Additionally, it is more desirable because it makes voltage gain available. A negative slope larger than 1 as an absolute value requires $C_g > C_1$, which is technologically feasible. Ratios of about 5 are currently routine, but values larger than 10 are still difficult.

The slopes of the SET synapse device are limited at the upper boundary by the available capacitor sizes, and that those boundaries differ for the positive and the negative slopes. The tightest slope boundary holds for the positive slope, which is always smaller than 1. The negative slope can reach much higher values (as an absolute value), although it is technologically difficult to make, and it is at the expense of the range of the positive slopes.

The lower boundary for the synaptic connection strength is zero because it is obtained with the operating point at a maximum or a minimum of the SET transistor transfer function, at the transition from the positive to the negative slope. At this point, a slight non-monotonicity is present in the SET synapse. It should be determined whether this hampers the SET synapse performance.

The transfer function period

The transfer function period of a SET transistor expressed in volts is determined only by the gate capacitance C_g of the gate under consideration. Expressed as a charge on the island, it is equal to the elementary charge e. If the device has more than one gate, each gate has its own period. The ratio α_s between the period of the SET synapse device input V_{P_s} and the output voltage range of the neuron in the previous layer V_{R_n} is defined as (Equation (6.1)):

$$\alpha_s = \frac{V_{R_n}}{V_{P_s}}. \tag{6.3}$$

The synapse performance should not rely on an exact ratio because it should be insensitive to manufacturing tolerances. In Section 7.3.2, it is shown that $\alpha_s \approx \frac{1}{4}$ is a good choice.

To obtain a certain ratio, the neuron SET transistor can be scaled slightly to match the desired value of the synapse gate capacitance C_g. This scaling is of course technologically limited. The freedom of choosing C_g more or less independently of the required ratio is useful because C_g also influences the value of the transfer function slopes.

The output range

The output voltage range of the synapse SET transistor device equals $\frac{e}{C_\Sigma}$ at 0 Kelvin. The total island capacitance C_Σ is equal to

$$C_\Sigma = C_g + C_{g_w} + C_1 + C_2. \tag{6.4}$$

C_g and C_{g_w} are the two gate capacitors, C_1 and C_2 are the tunnel capacitances. Except for technological manufacturing restrictions, the value of C_Σ can be designed independently of the ratios between the individual capacitors, because the transfer function slopes are independent of C_Σ, as shown by Figure 6.16. Within technological limits, C_Σ can be changed by scaling the complete device. It can be used to adjust the output range of the device so that it matches the transfer function period of another device, such as a neuron. This is the subject of Chapter 7.

Capacitor values

To obtain a synapse that can be realized at the moment, $C_1 = C_2$ is a good choice. With

$$C_g = C_1 + C_2, \tag{6.5}$$

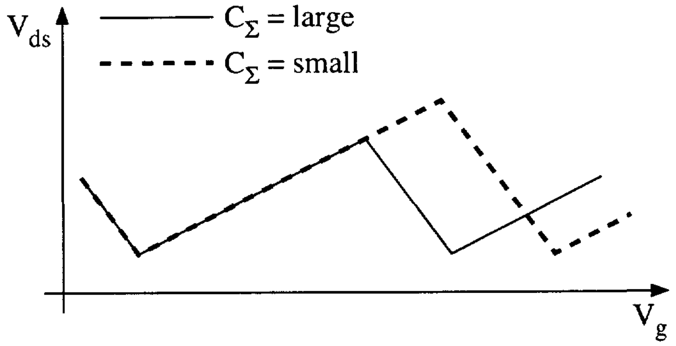


Figure 6.16: One period of the transfer function of a SET transistor for two values of C_Σ with constant capacitance ratios. Only the output voltage range changes, while the shape remains the same.

we have a maximum positive slope

$$s_{+_{\max}} = \frac{2}{5} \tag{6.6a}$$

and a maximum negative slope

$$s_{-_{\max}} = -2. \tag{6.6b}$$

We will see in Chapter 7 that this gives satisfactory results. The actual capacitor dimensions are determined in Section 7.3.3.

The influence of the bias current

The value of the bias current does not influence the slopes very much, as is shown in Figure 6.17. This implies that the circuits generating the synapse bias currents are not critical. The weight-dependent offset does depend on the bias current, but it is compensated for by the neuron (Section 5.5.5).

Random offset charge fluctuations

We have seen that the island charge of the device determines the slope of the transfer function, and that the synaptic weight can be implemented by an additional gate connection to the island (Section 6.2). This implies that offset charge fluctuations on the SET transistor island have the same effect as the weight changes, and that the physical random variations in offset charge can be modeled in neural network terms as random variations of the synaptic weight. This is an important property of the SET synapse device, because it means that the offset charge fluctuations, which are fundamentally present in all SET

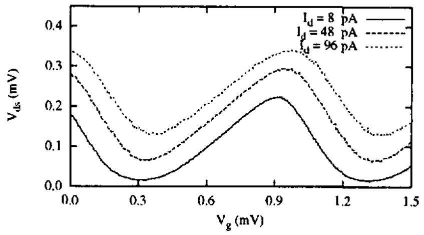


Figure 6.17: Transfer function of a SET transistor for bias currents from 8 pA to 192 pA.

transistors at the device physics level, can be mapped directly onto the neural network system level. On the condition that the learning algorithm is capable of readjusting the weights to compensate for a charge fluctuation before the next charge fluctuation occurs, the offset-charge fluctuations of the synapse device do not hamper the operation of the network. The transients appearing at the network output serve as an error signal for weight adjustments. The synapse is then robust to offset-charge fluctuations.

6.3.2 Weight storage

Weight storage can in principle be done in various ways. Here, only the storage of charge on a capacitor is dealt with because it is the most practical choice when all the signals are charges or voltages. The weight for the SET synapse device

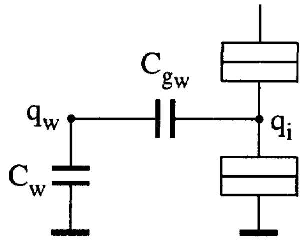


Figure 6.18: The synaptic weight is stored on capacitor C_w.

can be stored as a charge on a single capacitor C_w, which is connected to the island of the device through a gate capacitor C_{g_w}, as illustrated by Figure 6.18.

The charge change Δq_i on the island, resulting from changing the charge q_w on the weight capacitor by Δq_w, equals

$$\Delta q_i = \Delta q_w \frac{C_{g_w}}{C_w + C_{g_w}} \qquad (6.7)$$

(neglecting the voltage across the source junction).

Fine-tuning of synaptic weights usually requires small weight steps. The most elegant way of obtaining small weight steps exploits the periodicity of the SET synapse transfer function. If C_w is *almost* equal to C_g, a change in q_w of two electrons results in a change of *almost* one electron on the SET transistor island. As illustrated by Figure 6.19, the operating point on the synapse transfer function shifts by almost one period, resulting in a small weight change Δw. In this way, the weight adjustment step-size depends on small size differences between C_w and C_g

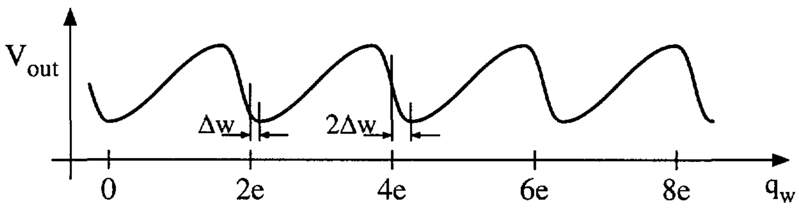


Figure 6.19: The periodicity of the synapse transfer function is exploited to obtain small weight changes. If $C_w \approx C_g$, changing the charge on the weight capacitor C_w by two electrons modifies the weight by a small fraction Δw.

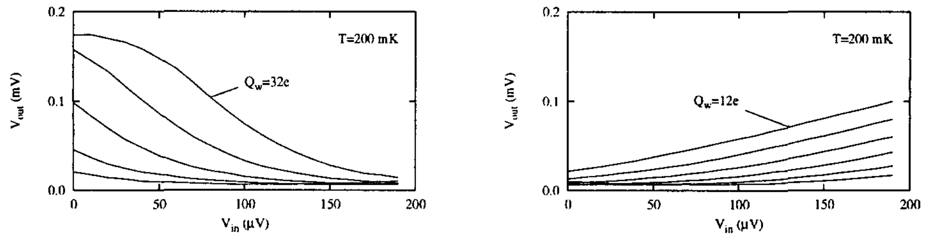


Figure 6.20: The synapse transfer function for $\Delta q_w = 2e$ weight charge increments. In this case, $C_w = 180$ aF and $C_g = 200$ aF.

The result for the synapse transfer function is presented in Figure 6.20. It shows the slope increments resulting from increasing the weight charge q_w in $\delta q_w = 2e$ steps.

Alternatively, small island charge changes on the scale of e are required to adjust the synaptic weights. Equation (6.7) then dictates that to scale down the elementary charge, C_w should be larger than C_g, so that to obtain small weight steps, C_w should be much larger than C_g. To fine tune the weight, impractically large weight capacitors C_w are required.

To obtain inhibitory as well as excitatory behavior of the synapse device, no bipolar voltages are required on the weight capacitors thanks to the periodicity of the transfer function. This is an advantage because it makes the weight modification circuitry simpler and it eliminates the need for a negative power supply.

6.3.3 Weight modifiers

To modify the weight of the SET synapse device, individual electrons must be added and removed from the weight capacitor C_w. In principle, the so-called 'turnstile' [60], or the 'electron pump' [61] can be used to manipulate individual electrons, and its potential application in neural circuits has been suggested in [54] and proposed in [36]. The circuits in question however only function properly when the island offset charges are adjusted properly. This makes them very impractical for use in large systems because the offset charges of such circuits cannot all be adjusted individually. The alternative is to accept that SET circuits cannot always operate properly, and to let the learning algorithm simply cope with it. Whether this is possible is certainly not trivial because a teacher needs accuracy to be effective. This discussion is postponed until Chapter 8.

6.4 Conclusions

A compact synaptic multiplying function can be made with only a single SET transistor by concentrating on obtaining the basic functional requirements.

The resulting SET synapse device does not produce an exact multiplying function, but the essential ingredient, namely a variable transfer function slope that is adjustable with the weight, is present. No insurmountable difficulties are expected from the characteristic device properties that make the synapse different from the standard synaptic multiplier.

Offset charge fluctuations in the SET synapse device are in principle adjustable by a learning algorithm because they are equivalent to weight fluctuations. Small weight changes are obtained with a small weight capacitor by exploiting the periodicity of the SET transistor.

With the SET synapse device described in this chapter, it is shown that effective matching of the specific technological properties to the required functionalities can result in an extremely compact functional device.

7

Design of a SET neural network

The performance of a neural network strongly depends on the way the cells are interconnected, on the efficiency of the signal transfer from cell to cell and on the learning algorithm performance.

In Section 7.1 it is argued that a locally interconnected neural network is the best choice for the implementation of a large neural network. This is based on both system and technological aspects of hardware implementation.

Section 7.2 shows that the SET neuron and synapse described in Chapters 5 and 6, can be interconnected to form a large neural network. It also shows how the signal amplitudes propagate through the network.

In Section 7.3, the simulation results of a neural network of one SET neuron and two SET synapses are presented. It shows that this SET network can classify in a way similar to a conventional neural network.

The learning algorithm is the subject of Chapter 8.

7.1 Neural network topologies

The ultimate goal of building neural networks with single-electron tunneling technology is to make large neural networks with high processing power. The technology-level and the system-level requirements on the topology of such a large neural network are described here from an implementation point of view.

7.1.1 System level

For most complex VLSI systems, the interconnections between the functional blocks use more space than the functional blocks themselves. Moreover, long interconnections suffer from long delays and crosstalk to other parts of the system. Therefore, effort should be put into minimizing the number of interconnections and their lengths [11].

For low communication intensity between the functional blocks, the processing power per building block should be high. If this is achieved, the number of wires in the system can be reduced [11]. The length of those wires should also be reduced. Both can be realized by choosing the appropriate system architecture.

The periodic, non-linear character of the SET transistor transfer function gives it a relatively high potential processing power, which makes it a good candidate for the implementation of functional blocks.

A neural network with a cellular topology and nearest neighbor connections is a good example of a system where the length and the number of the interconnections is minimized [4, 11, 62–64]. Moreover, the regular structure of a cellular neural network is a prerequisite for the design and realization of large systems [11].

7.1.2 Device level

The limited fan-in and fan-out of the SET transistor, but especially the fan-in (see Sections 4.3.2 and 4.3.3), puts a heavy constraint on the network topologies that can be realized with SET transistors. In Section 7.2, it is shown that the low fan-in and small voltage gain limits the number of inputs of a neuron. Requiring only a limited number of neuron inputs, cellular topology is a favorable choice for the implementation of SET neural networks.

7.2 Interconnecting synapses and neurons

A neural network arises when individual synapses and neurons are interconnected. In this section, the required properties of the neural cells that make the interconnections successful are described and the implications for the SET neuron and SET synapse described in Chapters 5 and 6 are analyzed.

Figure 7.1 displays a small part of a neural network, which illustrates that the output signal produced by a neuron is distributed to synapses in the next layer, while the neuron input collects the signals produced by the synapses in its own layer. This general setup holds for both multilayer feed-forward topologies and for cellular nearest neighbor topologies.

It is desirable that no additional buffers are required for the internal signals in the network between the synapses and the neurons. These buffers would

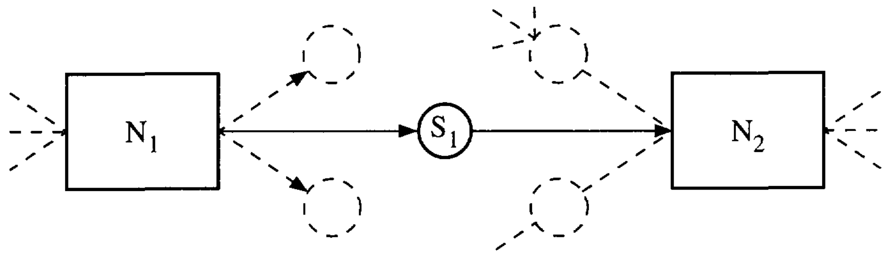


Figure 7.1: Part of a multi-layer neural network.

consume power and area, and therefore reduce the compactness of the neural network. This means that the synapse and neural cells should be designed such that they can be directly interconnected. The implications for the electrical domains are discussed below. It also means that the signal strength of all the neurons in consecutive layers should be the same to prevent a signal propagating through the network from dying out. This is the subject of Section 7.2.2.

7.2.1 The electrical domains in a SET neural network

We have seen in Chapters 5 and 6 that both the neuron and synapse device have a voltage as the signal carrier at their input as well as their output.

The output signal of a neuron is distributed to the synapses in the next layer of the neural network. With a voltage output signal carrier, the neuron is compatible with this requirement. It is matched by the synapse input carrier, which is also a voltage.

The capacitive addition input of the neuron neatly corresponds to the voltage delivered by the synapse output. Nevertheless, it may not be the optimum solution. The need for a capacitive connection for every individual synaptic connection limits the number of neuron inputs, which restricts the topological possibilities of the network. Further research for more optimal synapse-to-neuron connections should be done, for example using width modulation of current pulses [29].

7.2.2 The signal amplitudes in a neural network

Consider a large neural network with many layers. In general, a large multilayer neural network of this sort can only operate if the signals do not die out as they pass through the network. In this section, the signal propagation is analyzed for such a network, part of which is shown in Figure 7.2. SET neuron and synapse devices of the type described in Sections 5.5 and 6.2 respectively are used. The output signals of the neurons with range V_{R_n} pass through synapses, and the output signals of a number of synapses with range V_{R_s} are collected at the input of the neurons in the next layer. The signal ranges of the neurons and synapses are followed for one layer in the neural network, and the neuron output signal range is compared to that of the previous layer. As long as the neurons of

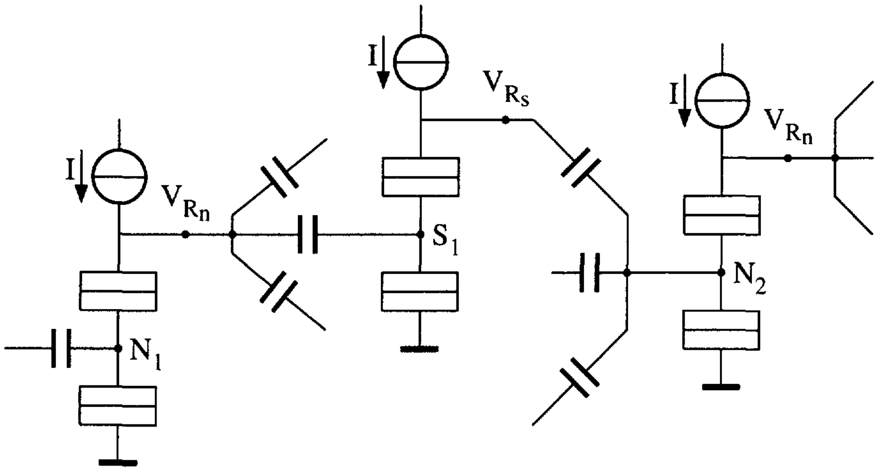


Figure 7.2: Part of a SET neural network.

consecutive layers have the same output range, and the synapses have enough output range to ensure that the total neuron output range is spanned, the signal will not die out. This is equivalent to requiring that the gain in the chain should be at least one.[1]

For pure feed-forward topologies with a small number of layers and for a funnel-like topology, the requirement is less strict. Some amplitude loss per layer can be tolerated, as long as the signal to noise ratio stays acceptable. Since we are concerned with large neural networks however, this option is not acceptable.

Below, it is shown that for the configuration of Figure 7.2 the restricted maximum slope of the synapse and of the neuron threshold function limits the number of synapses that can be connected to a neuron if every synapse is to be able to change the neuron state.

The output voltage range V_{R_n} of a neuron N_1 equals the fraction α_s of the SET transistor period V_{p_s} used by the synapse S_1 (see Equation (6.1)):

$$V_{R_n} = \alpha_s V_{P_s} \tag{7.1}$$

[1]The weight input of the synapse is not part of this chain, and is therefore not considered in this discussion.

which yields a ratio between the synapse gate capacitance C_{g_s} and the total neuron island capacitance C_{Σ_n} (see equations (6.2a) and (6.2b)):

$$\alpha_s = \frac{C_{g_s}}{C_{\Sigma_n}}. \tag{7.2}$$

The signal output range of a synapse depends on the range of the signal presented at its input by the neuron of the previous layer, and on the slope s of the synapse transfer function. As a result of its non-linearity, the synaptic slope not only depends on the synaptic weight w, but also on the input signal (see Section 6.1.2). For simplicity, this dependence is ignored here. The error thus introduced in this analysis is dependent on the fraction α_s. For a smaller α_s, the synapse transfer function better approaches a straight line, and the slope dependence of the input signal is smaller.

With the maximum slope $s_{\max}$ of the SET synapse transfer function, the maximum signal output range V_{R_s} of the synapse can be expressed in terms of the voltage range at its input, which equals the output voltage range of the neuron V_{R_n}:

$$V_{R_s} = s_{\max} V_{R_n} \tag{7.3}$$

The part of the neuron transfer function period V_{P_n} spanned by the output range V_{R_s} of a synapse is given by α_n (see Equation (5.19)):

$$\alpha_n V_{P_n} = V_{R_s}. \tag{7.4}$$

With (7.3), the period V_{P_n} of neuron N_2 can be expressed in terms of the output range V_{R_n} of neuron N_1:

$$\alpha_n V_{P_n} = s_{\max} V_{R_n} \tag{7.5}$$

To ensure that the signals propagating through the network do not reduce in amplitude as they pass from layer to layer, the SET neurons should all have at least the same size so that they also have the same output voltage range.

Therefore, Equation (7.5) also holds for one neuron. Filling in (6.2a) and (6.2b) for V_{P_n} and V_{R_n}, we have

$$\alpha_n C_{\Sigma_n} = s_{\max} C_{g_n} \tag{7.6}$$

The total island capacitance C_{Σ_n} of the neuron equals

$$C_{\Sigma_n} = C_{1_n} + C_{2_n} + (n+1) C_{g_n} \tag{7.7}$$

when all $(n+1)$ gate capacitors C_{g_n} are equal in size. There is one more gate capacitor than the number n of inputs to the neuron because one gate is used to adjust the threshold value. (In Figure 5.12, this capacitor was called $C_{g_{th}}$.) For a large number of inputs, C_{Σ_n} can be approximated by

$$C_{\Sigma_n} = (n+1)C_{g_n}. \tag{7.8}$$

With this approximation, Equation (7.6) can be simplified to

$$\alpha_n = \frac{s_{\max}}{n+1}, \tag{7.9}$$

which specifies that the fraction α_n of the neuron period spanned by a single synaptic output is a function of the maximum slope of the synapse transfer $s_{\max}$ and the total number of synapses n connected to the neuron. This means that for a fixed fraction α_n, the number of synapses that can be connected to the neuron depends on the maximum synaptic slope $s_{\max}$.

To enable a synapse to change the neuron state, it should span half of the neuron period, assuming a symmetric neuron transfer function (see Figure 7.3). With the absolute value of the maximum synaptic slope $s_{\max} = 2$

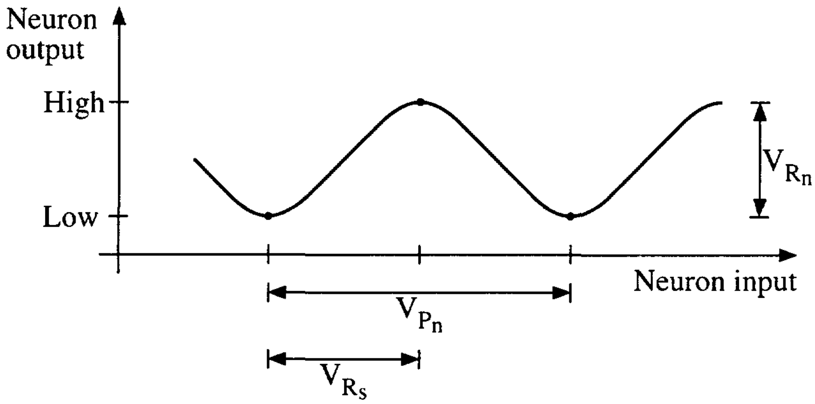


Figure 7.3: If $\alpha_n = 0.5$, a single synapse can change the state of a neuron.

(Equation (6.6b)), the maximum number of synapses that can be connected to a neuron equals $n = 3$. A single synapse can only change the neuron state if its slope is negative because the maximum positive slope equals $s_+ = 0.5$.

There are three reasons why the actual situation is a little worse. First, with $n = 3$, the assumption made to obtain Equation (7.6), namely that the tunnel capacitors can be ignored, is not entirely justified. Second, at non-zero temperatures, the actual output voltage range of SET transistors is smaller than the assumed $\frac{e}{C_\Sigma}$. Finally, the slope of the synapse transfer function is a function of the input signal, so that Equation (7.3) is optimistic. As a result, the individual synapses cannot always change the neuron state. On the other hand, if the neuron has an asymmetric transfer function, a synapse can more easily change the state in one direction (*high* to *low*) than in the other (*low* to *high*) because the distance V_a between the *low* and *high* state is then larger than the distance V_b between the *high* and *low* state, as can be seen by comparing Figures 7.3 and 7.4.

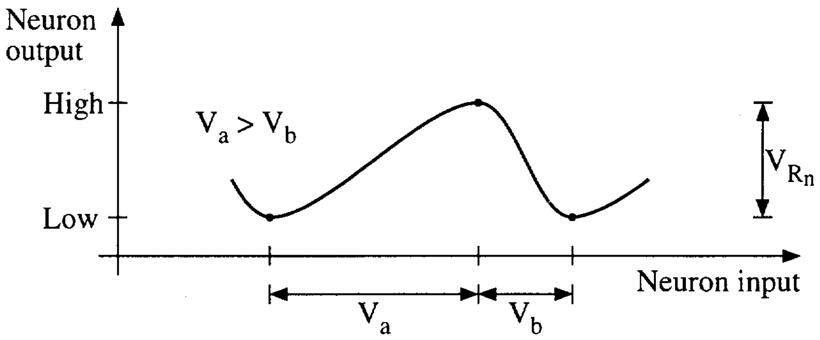


Figure 7.4: With an asymmetric neuron transfer function, the neuron state can more easily be changed in one direction than in the other

Three synapses per neuron is not very much, so what can be done to increase this number for the configuration of Figure 7.2?

- Allow larger maximum slopes in the synapse. This works only for the negative slope, and has the opposite effect for the positive one. If the learning algorithm can deal with strong asymmetry in the synapse, it is an acceptable solution.

- Limit the number of layers in the network so that signal amplitude deterioration is not detrimental for the network. A large neural network with a limited number of layers however, usually requires a large number of inputs per neuron, which was problematic in the first place.

- Do not require that individual synapses should be able to change the neuron state. This is only possible when the learning algorithm is capable of distributing the contributions of neuron inputs over more synapses.

- Increase the maximum negative slope of the (asymmetric) neuron transfer function.

No miracles are to be expected from the above measures to enhance the fan-in of the SET neuron device. The maximum fan-in of the neuron remains dependent on the maximum negative slope of the SET transistors used. Therefore, a sparsely connected topology such as a cellular neural network with nearest neighbor connections is required for a SET neural network.

7.3 A one layer neural network

To show the classification properties of a SET neural network, the smallest possible neural network is discussed in this section. To show the theoretical behavior of such a network, it is first described in Section 7.3.1 for the case with ideal multiplying synapses and a step activation function. In Section 7.3.2, an implementation is analyzed with SET synapse and neuron devices as described in Sections 5.5 and 6.2. It is shown that for a network with two synapses and one neuron, all the SET transistors and all the coupling capacitors can be identical. This uniformity is an attractive advantage for actual manufacturing.

7.3.1 Basics of single-layer feed-forward networks

The most elementary neural network is a single-layer feed-forward neural network with only a single neuron, as shown in Figure 7.5 for the case where the neuron has only two inputs. This basic system is also the elementary building

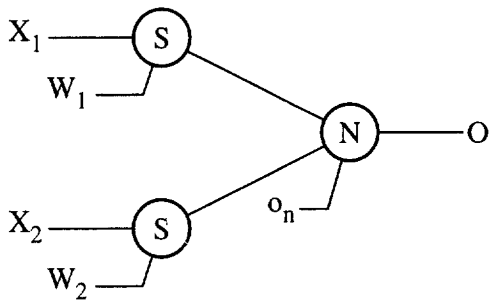


Figure 7.5: Basic neural network with two synapses and one neuron.

block of a cellular neural system.

It is used here to describe the behavior of single-layer feed-forward neural networks. The basic function performed by a neuron with two synapses is to separate two regions in the two-dimensional space spanned by the two inputs by a straight line, as illustrated in Figure 7.6.

The neuron activation function is defined as follows

$$O = \begin{cases} -1 & \text{if } X < 0 \\ +1 & \text{if } X > 0 \,, \end{cases} \tag{7.10}$$

where X equals the sum of the neuron threshold th and the weighted input signals $x_i w_i$:

$$X = th + \sum_i w_i x_i \tag{7.11}$$

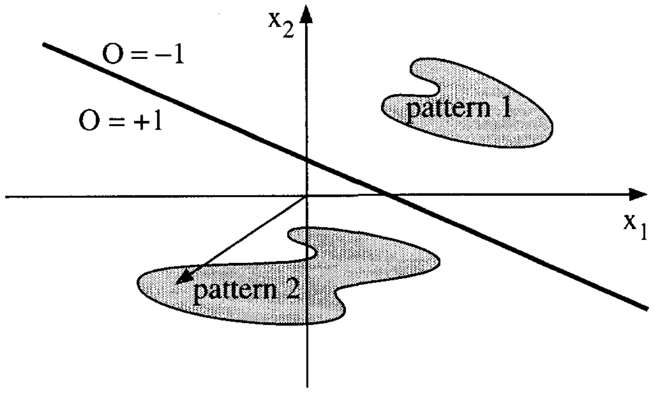


Figure 7.6: Two disjoint regions in the input space are separated by the neuron by a straight line. The arrow shows an input vector (x_1, x_2) that is a member of pattern 2.

This means that the boundary between $O = -1$ and $O = +1$ is given by $X = 0$. In the case of two inputs, this gives

$$th + w_1 x_1 + w_2 x_2 = 0, \tag{7.12}$$

which describes the straight line that divides the input space of Figure 7.6 in to two. The position of this boundary is determined by the synaptic weights w_1 and w_2, and the neural threshold value th.

A suitable learning algorithm adjusts the two weights and the neuron threshold until the boundary actually separates the two regions.

7.3.2 A SET neural network

The SET synapse device described in Section 6.2 and the SET neuron device of Section 5.5 can be connected to form a single-layer neural network such as described above. In the simplest case, one neuron and two synapses are used. Figure 7.7 depicts the schematic diagram of the SET neural network analyzed here. Since the neuron only has two inputs, no additional coupling capacitor is required to reduce the island capacitance of the neuron SET transistor (see Section 5.3.2).

We have seen (Equation (7.2)) that

$$C_{\Sigma_n} = \frac{1}{\alpha_s} C_{g_s}, \tag{7.13}$$

where α_s is the used fraction of the period of the synapse SET transistor.

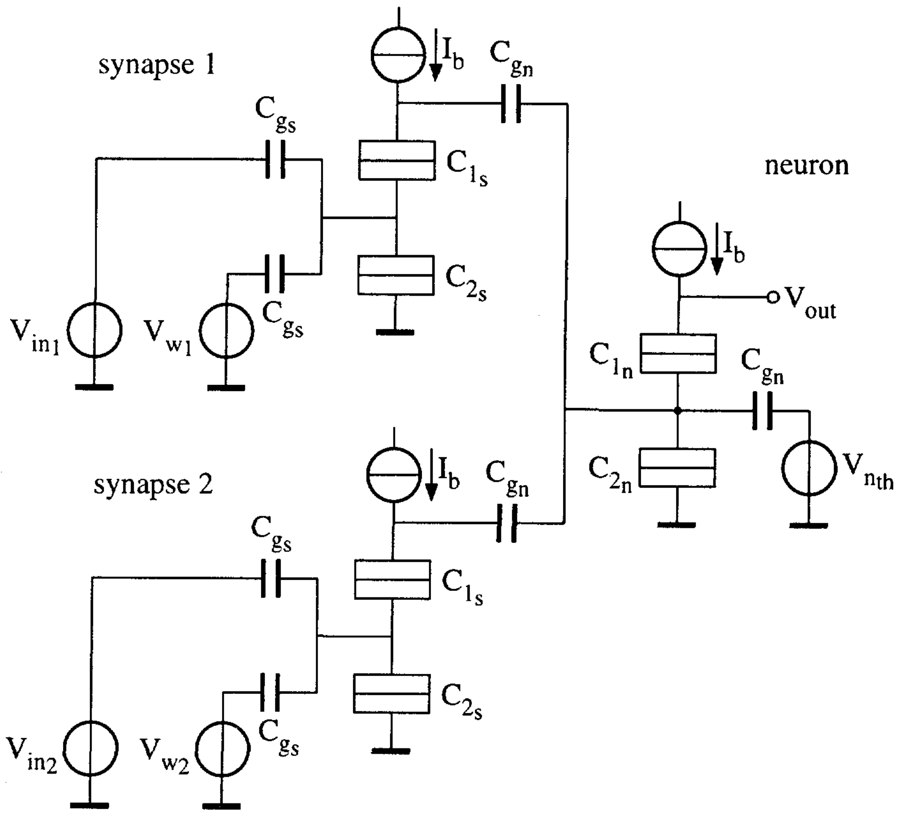


Figure 7.7: A neural network with SET transistors.

With the expressions for the total island capacitance C_{Σ_n} of this two-input neuron (Equation (5.20e)):

$$C_{\Sigma_n} = 4C_{g_n} \tag{7.14}$$

we find

$$C_{g_n} = \frac{1}{4\alpha_s} C_{g_s} \tag{7.15}$$

The fraction α_s of the period used by the synapse SET transistor is thus determined by the ratio between the gate capacitors of the neuron and the synapse. For equal gate capacitors, $\alpha_s = \frac{1}{4}$.

We have also seen that

$$C_{\Sigma_n} = \frac{s_{\max}}{\alpha_n} C_{g_n}, \tag{7.16}$$

(Equation (7.6)), where α_n is the number of periods of the neuron SET transistor spanned, and $s_{\max}$ is the maximum slope of the synapse. Combining this with Equation (7.14), we find that

$$4\alpha_n = s_{\max} \tag{7.17}$$

so that $\alpha_n = \frac{1}{2}$ if $s_{\max} = 2$.

If the gate capacitors are equal in size, all the SET transistors, all the tunnel junctions, and all the gate capacitors in the circuit are identical. This uniformity enormously simplifies the manufacture.

If more inputs per neuron are desired, α_s should be smaller to keep the SET transistors equal, or else the synapse devices should be made larger than the neuron devices (see Equation (7.15)).

7.3.3 Simulation results

This neural network was simulated with SIMON (Simulation of Nano Structures, see Appendix A).

A currently common size for the junction capacitors is

$$C_1 = C_2 = 100 \text{ aF}. \tag{7.18a}$$

Metal tunnel junctions of this size can be made using the shadow mask evaporation technique [65], and the resulting devices can be measured in a cryostat at a temperature below 1 Kelvin.

With equations (6.5) and (5.20d), we find for the gate capacitors:

$$C_g = 200 \text{ aF} \tag{7.18b}$$

which can also be manufactured readily.

The remaining parameters were also given realistic values:

$$\text{Tunnel resistances} : 100\ \text{k}\Omega \tag{7.18c}$$

$$\text{Bias currents} : 16\ \text{pA} \tag{7.18d}$$

$$\text{Temperature} : 50\ \text{mK} \tag{7.18e}$$

The neuron

With the above parameters, the neuron transfer function as shown in Figure 7.8 is obtained. The value of the threshold $V_{n_{th}}$ determines the position of the function along the horizontal axis. With $V_{n_{th}} = 0.2$ mV, the transfer function shifts 0.2 mV to the left, as indicated by the dotted curve. Larger values of $V_{n_{th}}$ shift the transfer function further to the left. The actual input range used in the network depends on the output range of the synapses. For the synapse described below it is about 0.25 mV (see Figure 7.9). With $C_\Sigma = 800$ aF, the output voltage range should be $V_{R_n} = \frac{e}{C_\Sigma} = 0.2$ mV. As can be seen from Figure 7.8, the average range is smaller, which is due to the non-zero temperature.

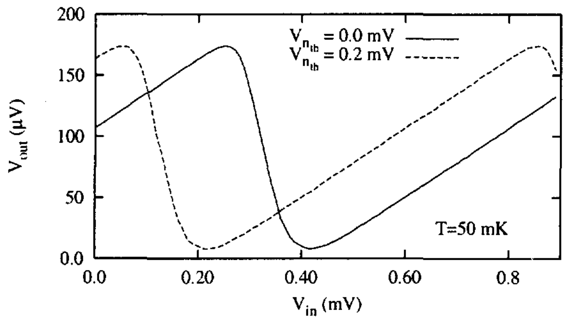


Figure 7.8: The transfer function of the neuron for two values of the neuron threshold $V_{n_{th}}$.

The synapse

The synapse transfer function for four different values of the synaptic weight is shown in Figure 7.9. The consequences of an asymmetric transfer function for the range of synaptic slopes and the effect of the relatively low temperature of 50 mK on the available slopes is immediately apparent. It is also clear that with a ratio α_s between the synapse period V_{P_s} and its input range V_{R_n} equal to $\frac{1}{4}$, the maximum output range of the synapse is smaller than was assumed in Equation (7.3) because of the dependence of the slope on the input signal, which was ignored in the equation.

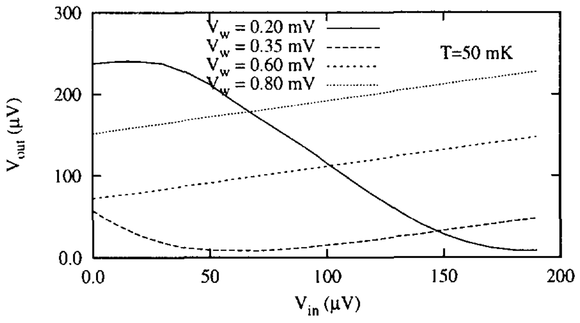


Figure 7.9: The transfer function of the synapse for four values of the weight V_w.

With $V_w = 0.35$ mV, the average slope is about zero, and with $V_w = 0.2$ mV, it has its maximum negative value. A positive slope is obtained for $V_w =$ 0.6 to 0.8 mV.

The above implies that a single synapse cannot change the neuron state on its own when it has a positive slope, and that the neuron output signal in that case has a smaller range than the previous layer.

The network

Let us investigate how this small neural network can classify. The network should be capable of dividing the input space into two regions, and the decision boundary position should be adjustable with the neuron threshold $V_{n_{th}}$ and the two weights V_{w_1} and V_{w_2}, as in Figure 7.6.

With one of the synapses adjusted to a slope of zero with $V_{w_2} = 0.35$ mV, the neuron output becomes independent of V_{in_2}, and the boundary between the *low* and *high* state can be adjusted with V_{w_1}, as shown in Figures 7.10 through 7.12. Note in these figures that the voltage range of the neuron output V_{out} is equal to the voltage range of the synapse inputs V_{in_1} and V_{in_2}. This illustrates that the signal amplitude does not deteriorate in this network.

By changing the weight V_{w_2}, dependence on the second input V_{in_2} is introduced for the boundary between the low and high state of the neuron. $V_{w_2} = 0.8$ mV gives a positive synaptic slope, which tilts the boundary towards higher values of V_{in_1} for higher values of V_{in_2} as illustrated by Figure 7.13, while a negative synaptic slope ($V_{w_2} = 0.2$ to 0.3 mV) does the opposite (Figure 7.14 and 7.15). The difference between the range of available positive and negative slopes is clear in these figures. The position of the boundary can also be shifted as in Figures 7.10 through 7.12.

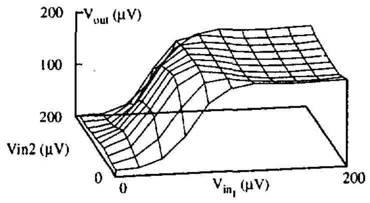


Figure 7.10: The input space is divided into two regions, independent of input V_{in_2}. The boundary is positioned by adjusting V_{w_1}. $V_{w_1} = 0.25$; $V_{w_2} = 0.35$; $V_{n_{\text{th}}} = 0.14$ mV.

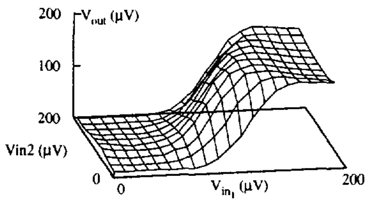


Figure 7.11: As Figure 7.10. The boundary is positioned at $V_{\text{in}_1} \approx$ 100 mV. $V_{w_1} = 0.20$; $V_{w_2} = 0.35$; $V_{n_{\text{th}}} = 0.14$ mV.

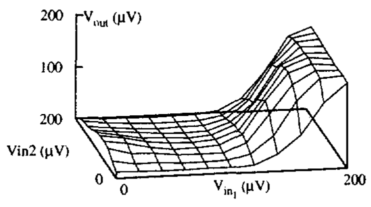


Figure 7.12: As Figure 7.10. The boundary is positioned at $V_{\text{in}_1} \approx$ 150 mV. $V_{w_1} = 0.15$; $V_{w_2} = 0.35$; $V_{n_{\text{th}}} = 0.14$ mV.

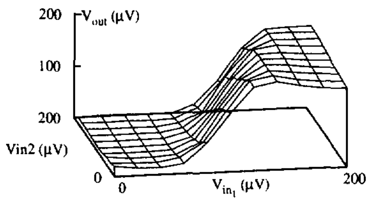


Figure 7.13: The boundary between the low and high state is tilted to the right by introducing a positive dependence of the second input. This is done by adjusting the synaptic weight V_{w_2} for a positive slope of synapse S_2. $V_{w_1} = 0.20$; $V_{w_2} = 0.80$; $V_{n_{\text{th}}} = 0.14$ mV.

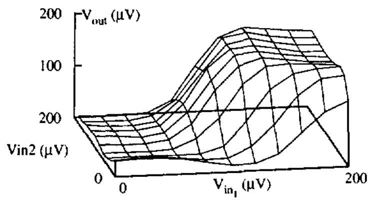


Figure 7.14: A negative slope of synapse S_2 tilts the decision boundary to the left. $V_{w_1} = 0.20$; $V_{w_2} = 0.30$; $V_{n_{th}} = 0.14$ mV.

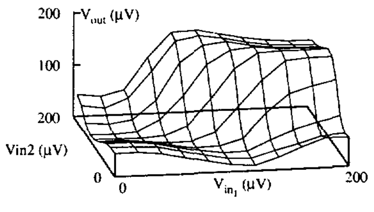


Figure 7.15: A steeper negative slope of synapse S_2 gives a more pronounced tilt of the decision boundary. $V_{w_1} = 0.22$; $V_{w_2} = 0.22$; $V_{n_{th}} = 0.10$ mV.

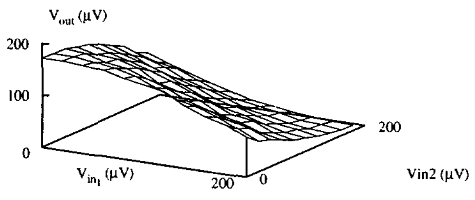


Figure 7.16: An inverting function realized by using the positive slope of the synapses together with the negative slope of the neuron. (Note the different view angle compared to the other figures.) $V_{w_1} = 0.60$; $V_{w_2} = 0.60$; $V_{n_{th}} = 0.11$ mV.

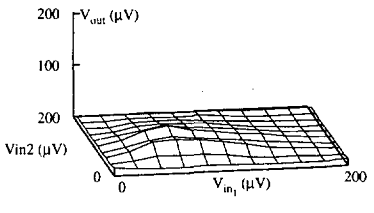


Figure 7.17: Zero sensitivity of the neuron output for the two inputs is achieved when the neuron is adjusted at the transfer function trough and the synapses at a peak. $V_{w_1} = 0.35$; $V_{w_2} = 0.35$; $V_{n_{th}} = 0.34$ mV.

By using negative synaptic slopes and a positive neuron transfer, or vice versa, the position of the high and low state with respect to the input values can be reversed. A high state for low input values, and a low state for high input values is shown in Figure 7.16. Finally when both the synapse transfer is adjusted to a peak while the neuron is in a trough, the neuron output is low for all values of the synapse inputs (Figure 7.17). A constantly high neuron output can be obtained in a similar manner.

7.4 Conclusions

The topology of a large neural network should be locally and sparsely connected and have a regular network structure because short and few interconnections save space, and reduce the load of the individual cells. The regular pattern is necessary because it is easier to manufacture than an irregular structure.

The choice of a sparsely connected topology holds in particular for the SET network described in this chapter because the neurons have a limited fan-in. The limited differential voltage gain of the SET transistors used restricts the number of synapses per neuron. For a SET transistor with a maximum gain of 2 (as an absolute value), the maximum number of synapses per neuron is 3. The neuron fan-in is higher if SET transistors with a higher gain are used, or if the individual synapses are not required to change the neuron state.

Simulation results show that with a two-synapse, one-neuron neural network, classification of the input space into two regions is possible, and that the neuron can drive a subsequent layer of synapses, so that a larger network can be constructed.

8 The learning algorithm

The learning algorithm of a neural network is possibly the most complicated part of a neural system to understand and to make. It teaches the network the desired behavior, and can adapt the behavior to changing circumstances and conditions.

Teaching a large neural network successfully is still difficult at the moment, and the problems with long convergence times and local minima will certainly not diminish as the network size increases. The complex topic of finding an optimal learning strategy for very large SET-based neural networks deserves a full research project of its own.

This chapter describes the classification and training performance of two-layer neural networks with SET synapses and neurons. The aim is twofold: first to show that two-layer networks of periodic synapses and neurons are feasible and can classify and learn predefined functions; and second to analyze the compatibility of the learning algorithm with the stochastic and periodic properties of the hardware used to implement the synapses, neurons and (part of) the learning algorithm.

First of all, the required properties of a neural learning algorithm are described in Section 8.1. It is argued that for a large and compact neural network, a supervised learning algorithm that is simple to implement, and that can handle inaccuracies in the neuron and synapse hardware, is to be preferred.

Section 8.2 describes the random weight change learning algorithm [52,66], which complies with the above requirements. Parameters that describe its behavior and that model hardware non-idealities are introduced.

Section 8.3 describes simulations with which the performance of the random weight change learning algorithm and the classification abilities of two layer SET neural networks is analyzed. The results show that a neural network with periodic synapses and neurons can learn the XOR function, and that errors occurring in the learning algorithm due to hardware non-idealities can partly be handled by the learning algorithm itself.

8.1 Required properties of the learning algorithm

The learning algorithm adjusts the synaptic weights and neuron thresholds so that the neural network performs the desired functionality. In this section, the required properties of such an algorithm for a large and compact neural network implementation are analyzed. In Section 8.1.1, different kinds of learning algorithms are described. It is argued that a supervised algorithm that periodically updates the weights is most suitable for a neural network implemented with inaccurate hardware. In Section 8.1.2, the compatibility of the learning algorithm with the specific neural hardware properties is analyzed.

8.1.1 Types of learning algorithms

Several types of learning algorithms can be distinguished:

- supervised or unsupervised learning,
- adaptive or non-adaptive learning,

and if the network learns adaptively, this can be done

- continuously or periodically.

These possibilities are clarified below.

Supervised or unsupervised learning

The algorithm can either be steered by a teacher signal supplied to the network by a (partly) external supervising system (Figure 8.1.a), or it can be steered by an internal signal, generated from the neural network signals by the learning system itself (Figure 8.1.b). The former is called supervised learning, while the latter is known as unsupervised learning.

Supervised learning is to be preferred for SET neural networks because SET hardware has stochastically fluctuating properties. In a supervised learning system, the system accuracy fully depends on the reliability of the error signal and the presence of enough loop-gain; the accuracy of the implementation of the rest of the system is of less importance. Therefore, the implementation of the teacher and the error signal generator must be accurate, and can in principle not be implemented in SET technology, while the rest of the system can.

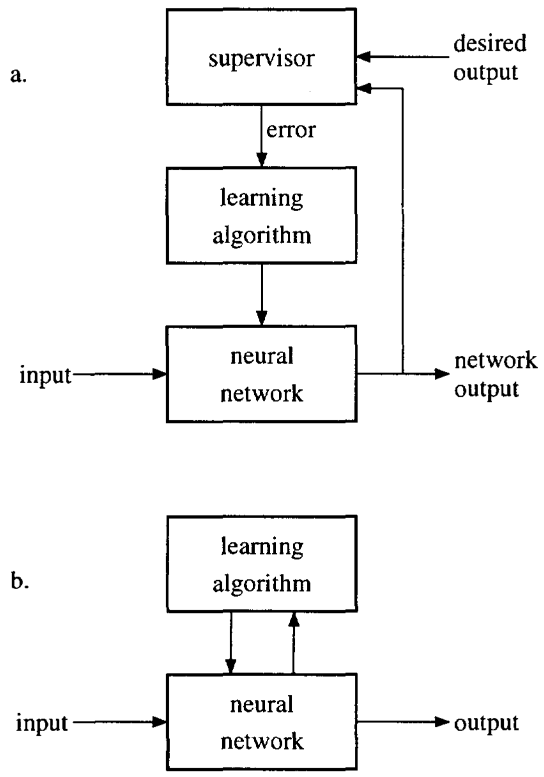


Figure 8.1: **a.** Supervised learning, and **b.** unsupervised learning.

By contrast, an unsupervised learning system does not dispose of a central teacher, and the system accuracy therefore relies on accurate components throughout the system.

Adaptive or non-adaptive learning

The weights of a neural network that learns non-adaptively are only adjusted once, before the network begins to operate. During operational use, the network cannot adapt to changing circumstances, hence the name 'non-adaptive'. It can be used if the neural hardware is robust and stable, and if the task of the neural network never changes. Handwriting recognition for postal code reading is a successful application of non-adaptive learning.

In many other cases, adjustment of the synaptic weights and neuron offsets during the operational use of the neural network remains necessary to improve the performance or adapt it to changing circumstances. Both the conditions in its environment, and the properties of the neural network itself may change in time. In this case, an adaptive neural network is required. A good example of changing conditions in the environment of a neural network would be a human face recognition system that adapts itself to changing hair-dress and aging of the persons it should recognize.

Unfortunately, there are also plenty of examples of changing properties inside the neural network itself that require weight value adjustments. This is in particular the case with hardware such as SET technology, where island offset charge fluctuations frequently influence the device behavior, and parameter shifts due to temperature drifts and weight leakage also require weight adjustments.

The presence of these hardware non-idealities makes the choice of an adaptive neural system essential when working with SET technology.

Continuous or periodic learning

The learning algorithm of an adaptive neural network can be active continuously or only periodically. The network can only learn continuously if the learning and the actual use of the network can happen simultaneously. Most supervised learning algorithms do not allow this because specific learning examples of which the desired network output is known must be supplied to the network during learning. Therefore, most supervised learning algorithms are only active during limited time intervals. The rest of the time is used for useful network operation. Some supervised learning systems however, do show simultaneous learning and network operation. Biological neural networks convincingly show that operational use and adaptation can be tightly interwoven. Take as an example the task of reading a new piece of text, written with a hitherto unknown handwriting. While reading the text, the parts of the brain that decipher the letters and words are supervised by other parts of the brain that contain a priori knowledge of the language. We gradually learn and apply the specific characteristics of this handwriting, and become adept at recognizing the way the letters are formed.

8.1.2 Compatibility with hardware

The specific properties of the devices in SET technology put important constraints on the algorithm. Here we describe the consequence for the learning topology, the effects of the synapse and neuron properties, and the consequences of implementing (part of) the learning algorithm itself in SET hardware.

Compatibility of the learning topology with the hardware

The topology of a learning algorithm is closely related with the hardware implementation of the synapses, neurons, and the learning algorithm itself. Choices made on this subject must therefore reflect the available hardware properties.

Two types of learning topologies can be distinguished:

- global learning
- and local learning.

In a global learning topology, the weight adjustments depend on signals that may be topologically far away from the weight. This is algorithmically advantageous because more information is available to determine weight updates. A global learning algorithm may therefore converge faster.

A local learning algorithm determines adjustments of a particular weight only from information that is available locally. For an implementation in hardware, this requires far less and shorter interconnections, which is a welcome advantage for SET technology because only sparsely connected systems can be realized. Therefore, for a large SET neural network implementation, some degree of local learning is required. Strictly speaking, the supervised learning algorithm, which was favored in the previous section, cannot be purely local because the teacher signal is only externally available. As long as the number of control signals that must be globally distributed is low, the algorithm concerned does not lose all the benefits of local learning.

Additionally, the algorithm can adjust

- each weight consecutively,
- or all the weights simultaneously.

Adjusting each weight in successive learning iterations has the algorithmic advantage of making founded adjustment decisions per weight possible because the rest of the neural system is known to be invariant. This makes successful convergence more likely. On the other hand, it is a very lengthy procedure for a large neural network with many weights.

The opposite extreme of adjusting all weights at every learning cycle is difficult to solve deterministically on a global level because of the large number of variables involved. Therefore, such a learning algorithm must determine the weight adjustments from locally available information.

A successful learning algorithm for a large neural network is therefore a compromise between the simultaneous adjustment of all the weights and the use of enough global information to guarantee fast and successful convergence.

Compatibility with synapse and neuron hardware

In general, a neural learning algorithm can either use a priori knowledge of the synapse and neuron properties or not. An example of a learning algorithm that uses knowledge of the synapse and neuron properties is the back-propagation algorithm (see for example [40,67]). It uses the information that the synapse is a perfect multiplier and information on the derivative of the neuron activation function to calculate the optimal weight changes for minimizing the global error. The use of this a priori knowledge makes the algorithm fast and reliable. If the synapse and neuron implementations do not exactly conform to the ideal models however, this a priori knowledge is wrong, and the algorithm fails. Backpropagation is therefore less suitable for training inaccurate neural network implementations [67].

Two examples of learning algorithms that work without a priori knowledge of the neural network implementation are the Madaline Rule III [48,68,69] and the Random Weight Change algorithm [52,66]. These algorithms are based on observing the effect on the overall error when the weights in the neural network are changed by a small amount δW. Measuring the sensitivity of the error to weight changes makes the algorithms insensitive to the specific implementation of synapses and neurons, or to the topology of the network [48].

In general, because they have less information available, these algorithms do not perform as well as those that base the weight updates on specific synapse and neuron properties. Therefore, if a priori knowledge of synapses and neurons is available, it is advisable to use it.

The SET neuron and synapse devices described in Chapters 5 and 6 have a number of characteristic properties that make them quite different from standard neural hardware:

- both neuron and synapse transfer functions consist of part of a periodic transfer function;
- offset charges give the devices a random offset at their inputs;
- the neuron has no real saturation regions;
- the synaptic slope values have a limited range, and depend on temperature;
- the output ranges depend among other things on temperature.

The inaccuracy in some of these properties makes a learning algorithm for SET neural networks that fully relies on exact knowledge of the neural cells unfeasible. Having said this however, some characteristics are certain and predictable, such as the transfer function periodicity, and the absence of saturation regions. A learning algorithm tailored for SET neural networks exploits this a priori knowledge to obtain an optimal learning performance.

Compatibility of the learning algorithm with its hardware implementation

At least part of the learning algorithm is preferably implemented locally, near the weights to be updated (Section 8.1.2). Following this line of thought, it is indispensable to build this part of the learning hardware using the same technology as the rest of the neural network. To obtain a compact realization, the same rules apply as for implementing other network elements: the primitive functions must be compactly designed with small hardware structures, optimally exploiting the technological properties (Chapter 3). For compact learning hardware, the algorithm itself should therefore be developed not only with its learning performance in mind, but also with the available technological functions in mind.

A consequence of implementing part of the algorithm with stochastically fluctuating hardware such as SET technology is that the network must be at least partly supervised, and that the teacher signal and error generation must be accurate for the learning algorithm to be successful. This error signal may have to be distributed across the network.

Conclusion

The learning algorithm for a large SET neural network is preferably realized locally, near the weights it should update. For compactness, it should also be implemented as much as possible in SET technology. The stochastic properties of this technology however prevent the implementation of the entire algorithm in SET technology and dictate the use of a supervised learning algorithm. To guarantee the accuracy of the neural system, the critical, supervising functions must be provided externally. The learning algorithm performance is enhanced if the algorithm can exploit the characteristic properties of the synapses and neurons.

8.2 Random Weight Change learning

The Random Weight Change learning algorithm [52, 66] was chosen to train a small SET neural network [70–73] because this algorithm complies to most requirements described in the previous section: it does not rely on a priori knowledge of the neuron and synapse transfer function, it is relatively simple to implement because it does not consist of complex mathematical operations, and it mainly uses locally available signals to determine the weight updates; only the control of this local hardware is performed by a set of global control signals which are broadcast all over the network.

Section 8.2.1 explains the algorithm and Section 8.2.2 describes the parameters that control it. Based on a block diagram for implementing this algorithm in hardware, Section 8.2.3 analyses the types of errors that may occur if this algorithm were to be partially realized with SET hardware. The simulations performed to investigate its performance for a small neural network are described in Section 8.3.

8.2.1 The algorithm

A flow diagram of the random weight change learning algorithm is shown in Figure 8.2. For one pattern, all weights W_i of the neural network are changed by

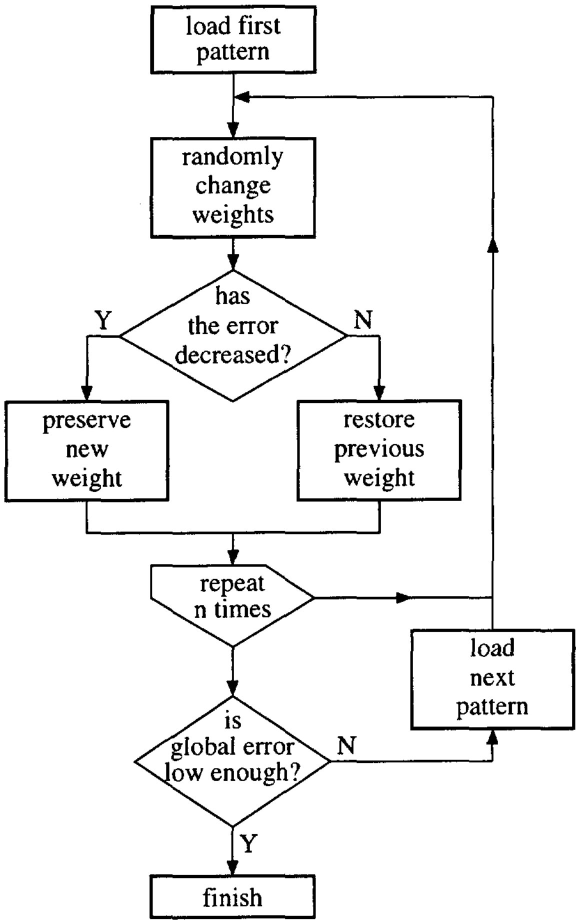


Figure 8.2: Flow diagram of the Random Weight Change learning algorithm.

a small amount δW_i. The absolute value of δW_i is the same for all weights, while its sign is random. The effect of this weight change is measured by comparing the network output with the desired output. If the error has decreased in comparison to what it was before the weights were changed, the new weights are probably better and they are preserved. If the error E has increased, the previous weights were better so they are restored.

In this way, the algorithm tries to improve the weights a predefined number of times, before it switches to the next pattern. All the patterns are cyclically presented to the network until the global error E_g remains below a certain level for all patterns. An alternative to cyclically presenting the patterns to the network would be to present them in random order. This possibility is not analyzed here.

The error E_p used to determine whether a particular weight perturbation was an improvement for the current pattern p is defined as

$$E_p = c(V_{\mathrm{ref}_p} - V_{\mathrm{out}_p})^2, \tag{8.1}$$

where V_{ref} is the desired output for the pattern under consideration, V_{out} is the actual network output, and c is a normalization constant.

The global error E_g used as a stopping criterion is a running mean of the most recent errors E for each pattern:

$$E_g = \frac{c}{4}\sum_{j=0}^{3}(V_{\mathrm{ref}} - V_{\mathrm{out}})^2. \tag{8.2}$$

The weight change δW equals

$$\delta W_i = r_{\pm_i}\eta E_p, \tag{8.3}$$

where $r_{\pm_i}$ is randomly generated for each weight w_i, and equals either $+1$ or -1. The learning gain η is a global parameter, adjustable for optimal performance. The weight perturbation therefore has a random sign and its magnitude depends on the error. The weights are less perturbed when the error decreases. This drastically speeds up the weight change process at the beginning, while preserving the advantage of fine tuning at the end.

8.2.2 The control parameters

The learning algorithm has two control parameters to adjust its performance.

- The number of trials per pattern n
- The learning gain η

The number of trials n controls how many times the learning algorithm tries to improve the weight values before switching to the next pattern. The algorithm performance does not depend very much on this parameter, but some general remarks can be made. If it is too small (smaller than about 10 trials per pattern for the networks of Section 8.3.2), training convergence slows down because it is unlikely that the weight perturbation decreases the error for the pattern under consideration. On the other hand, if it is too large ($n > 50$),

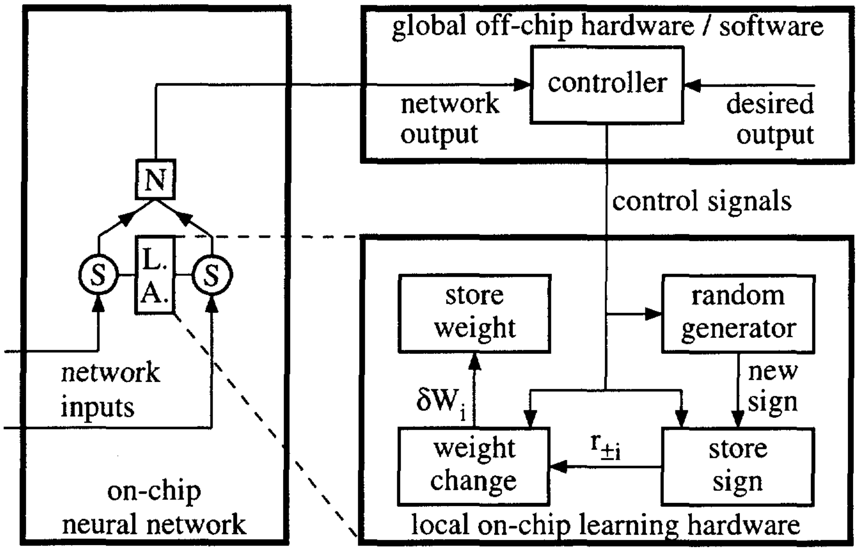


Figure 8.3: Block diagram of the learning algorithm hardware.

convergence starts slowly because the weights are changed too much in each cycle. This 'damages' the weight adjustments made for previous patterns.

For much larger networks, the lower limit is expected to increase because it becomes less likely to find a global weight improvement if there are many weights to adjust. This could well be an important limitation of this learning algorithm's applicability.

The learning gain η determines how the weight perturbation depends on the error E_p. If it is too small, the error decreases too slowly, and if it is too large, the error does not reach a low value.

8.2.3 Hardware-related errors

Figure 8.3 presents a block diagram of the primitive functions necessary for the random weight change learning algorithm. The learning algorithm is split up into local SET hardware and global hardware. The global, off-chip part of the algorithm calculates the error, controls the step size, and decides whether the weights should be perturbed or restored. These control signals are the same throughout the network and must be distributed to all neural cells. The local, on-chip hardware performs all the actions which are different for each individual weight. The errors that may occur in these local functional blocks resulting from offset charge fluctuations are described below.

Errors in the weight storage block

The weights are stored as a voltage or a charge on capacitors that are directly connected to the gate of the neuron or synapse (see Section 6.3.2). The voltage on this capacitor represents the weight or threshold value. Because the transfer functions of the synapses and neurons are periodic, the full range of the output can be addressed by positive values. The weight is periodically updated, hence some leakage to or from the capacitor is permitted. As described in Section 6.3.2, charge quantization results in a minimum weight step size, which can be kept small by using suitable capacitor sizes. Weight value quantization is not modeled in the simulations described in Section 8.3.

Errors in the weight changing block

The weight changing block combines the step size δ and the sign $r_{\pm_i}$ into the change in weight δW_i. This could be a packet of charge to be added or removed from the weight capacitor. Malfunctioning can affect the learning behavior in two ways. When a 'previous weight' should be restored (see Figure 8.2), errors in the weight changing block have the effect that the 'restored weight' is not exactly equal to the value the weight had before perturbation. The learning behavior is also affected if the step size does not decrease to small values when convergence is reached. The uncontrollability of the step size is modeled with the *relative step size error* parameter.

Errors in the sign storage block

The sign provider delivers the appropriate sign $r_{\pm_i}$ at its output for the weight changing block. It is a new and random sign generated by the random generator when the weight is perturbed, and it equals the previous sign when the weight is restored. Memorizing this previous sign is assumed to take place in this block. Alternatively, it could be part of the random generator block described below. We can expect that a circuit implementation of this function will not always produce the desired sign. Hence, *sign error probability* is introduced, which gives the probability that a sign $r_{\pm_i}$ is wrong.

Random sign generator

For each new trial a new random sign $r_{\pm_i}$ should be generated for weight w_i. This function is not very critical, as long as a random signal is produced. This block is considered ideal in the simulations described in Section 8.3.2 because a good random generator design is expected to be capable of dealing with random offset charge fluctuations, which is the main cause of errors in SET technology.

8.3 Training a two-layer SET neural network

We have seen in Chapter 7 that a single SET neuron with two SET synaptic inputs can perform the elementary classification operations demanded by a neural network. It is therefore interesting to investigate the classification behavior and the learning abilities of larger neural networks.

Figure 8.4 gives an example of a fully connected two-layer network with an arbitrary number of inputs, hidden neurons (first layer) and output neurons (in the second layer). With a two-layer network of linear synapses and sigmoid neu-

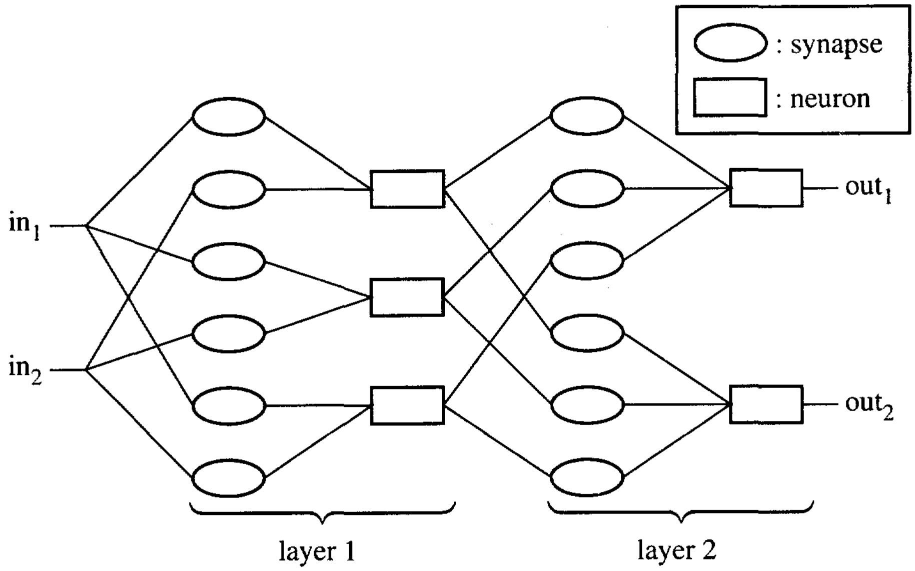


Figure 8.4: An example of a two-layer neural network that can be simulated with the software. This network has two inputs, three neurons in its hidden layer (the first layer) and two output neurons (the second layer).

rons, in principle any non-linearly separable classification can be realized [40]. Although two layers are not always necessary to separate classes non-linearly if the network has non-linear synapses and neurons [46, 59], a two-layer network gives more classification flexibility, and the presence of a hidden layer is a greater challenge for the learning algorithm.

Although a fully connected network is not the optimum topology for very large SET neural networks (see Section 7.1), we use this topology here because it is easier to understand, and because the learning algorithm is known to work for feed-forward networks. It is a good start towards more complex, future systems.

Section 8.3.1 describes software to simulate a two-layer neural network with periodic synapses and neurons, trained with the random weight change learning algorithm. In principle, the network layers may have any size. To avoid very long simulation times, we restrict ourselves to layers with a small numbers of neurons and synapses.

The results obtained with this software are presented in Section 8.3.2. It is shown that a small two-layer network can successfully be trained to classify the XOR function, and the learning algorithm is robust to the type of errors occurring in SET technology [70].

8.3.1 The SET neural simulator

The interactive software to simulate the network was written in Matlab. Two relevant aspects of the program are described below.

- The synapse and neuron models
- The learning algorithm

Synapse and neuron models

To shorten the simulation time, only the transfer function of the synapses and neurons are modeled by the program. Any arbitrary function can be defined for this purpose by means of a tabular definition in the software. The model described in Section 4.4.4 was used to fill the table to simulate the SET transistor transfer function. This means that for example the limited fan-in of SET transistors, its stochastic character, and its precise temperature behavior are not taken into account.

SET transistors can only produce unipolar output signals, therefore only positive inputs are supplied to the simulated network, and the *low* and *high* neuron output states are defined as in Figure 8.5, where *low* is scaled to +0.5 and *high* is scaled to +1.5. Output range of the devices is normalized to 0 to 2. The synapses also produce only positive output values. Inhibitory synapses are obtained with the negative slope of the SET transistor transfer function, as explained in Section 6.3.1. The synaptic weight was implemented by scaling the output amplitude of the concerned SET transistor.

The learning algorithm

The random weight change learning algorithm is implemented algorithmically in the software. Four parameters that influence the learning algorithm performance are (see Sections 8.2.1 and 8.2.3):

- number of trials n,
- learning gain η,

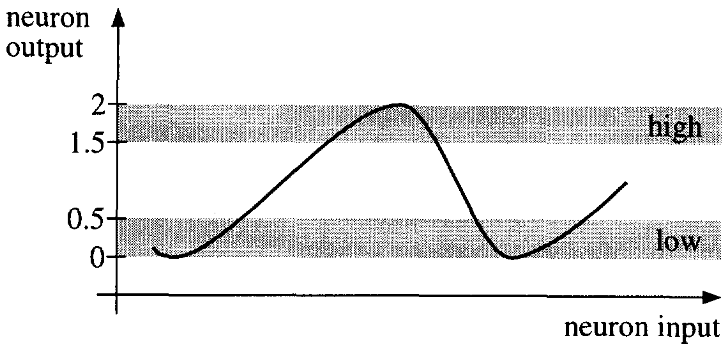


Figure 8.5: Definition of the *low* and *high* state of a neuron.

- relative step-size error,
- sign error probability.

Since each learning cycle optimizes the weights for only one input-output combination, the error E might increase suddenly when switching to another pattern.

Only when the overall error E_g stays low for all patterns, has the function been learned. This stop criterion could be automated, but in the software used here it was done through visual inspection by the operator.

8.3.2 Simulation results

Although networks of arbitrary size can be simulated, most runs were done on a small network. Larger networks give comparable results [70], but both the simulation time per learning cycle and the number of learning cycles increases. The five experiments shown in the table below were performed on a network with two inputs, two hidden neurons and one output neuron, as shown in Figure 8.6. Up to five simulation runs were done for each of the experiments. The results described below show representative selections of these runs.

Experiment	Neuron type	Synapse type	non-ideality
1	Sigmoid	Multiplier	
2	Periodic	Multiplier	
3	Periodic	Periodic	
4	Periodic	Periodic	Noise
5	Periodic	Periodic	Learning errors

The insets in Figures 8.7 to 8.11 show the truth tables produced by the network after learning the XOR function. The training set consisted of the ideal XOR truth table shown below. Two versions were used, one with bipolar values for the conventional and the sinusoidal models and one with only positive values for the SET transistor models.

$\mathbf{in_1}$	$\mathbf{in_2}$	**out**
-1	-1	-1
-1	1	1
1	-1	1
1	1	-1

$\mathbf{in_1}$	$\mathbf{in_2}$	**out**
0.5	0.5	0.5
0.5	1.5	1.5
1.5	0.5	1.5
1.5	1.5	0.5

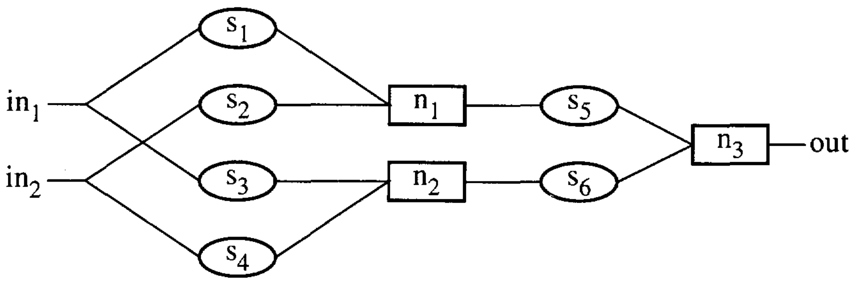


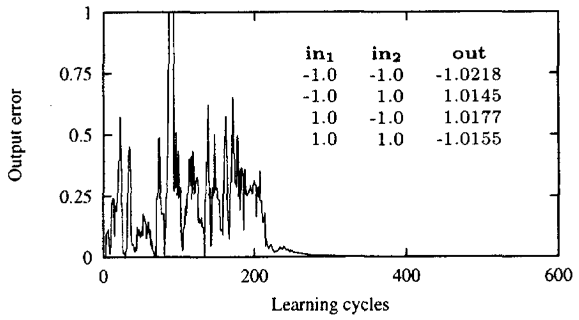
Figure 8.6: The network used to train the XOR function.

Conventional neuron and synapse transfers

To verify the correct functioning of the software, and for the purpose of comparison, the first experiment was done with conventional neural cells: a linear multiplier as a synapse and a sigmoidal neuron. The output error as a function of the learning cycle is shown in Figure 8.7. We see that the error initially fluctuates more or less randomly, searching for weight values that suit all four test patterns. After about 200 trials, the error remains low for all four test patterns. The weight updates then become small, and fine tuning then reduces the error to zero.

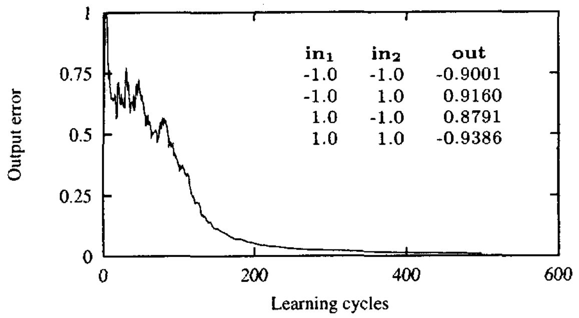
Periodic neurons and multiplier synapses

Subsequently, the sigmoidal neuron activation function was replaced by a sinusoid, while the linear synapses were retained. The resulting output error as a function of the learning cycle is shown in Figure 8.8. In contrast to the previous case, the error initially never drops to very low values and subsequently gradually decreases. The lack of sharp dips could be attributed to the absence of saturation regions in the sigmoidal neuron activation function. (However see also the remarks made about this for the next experiment.)



in_1	in_2	out
-1.0	-1.0	-1.0218
-1.0	1.0	1.0145
1.0	-1.0	1.0177
1.0	1.0	-1.0155

Figure 8.7: Convergence of a network with conventional synapses and neurons. Inset: XOR truth table produced by the network after learning. $\eta = 0.16$; $n = 13$.

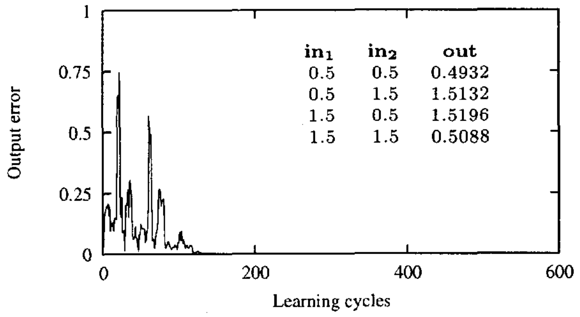


in_1	in_2	out
-1.0	-1.0	-0.9001
-1.0	1.0	0.9160
1.0	-1.0	0.8791
1.0	1.0	-0.9386

Figure 8.8: Convergence of a network with multiplying synapses and sinusoidal neurons. Inset: XOR truth table produced by the network after learning. $\eta = 0.1$; $n = 13$.

Periodic neurons and synapses

Figure 8.9 shows the convergence of the network when the transfer function from input to output for both the synapse and neuron functions is periodic. From here on, purely positive signals were used for the neuron and synapse input and output, as is the case with real SET devices. The network converges faster than with conventional neurons and synapses. We attribute this to the fact that with periodic neurons and synapses, the output value range of both neuron and synapse is limited, so that the output is never far away from the desired value, and consequently the weight change needed to reach the optimum value is always small: its maximum value is equal to the period of the function. When the synapse also has a periodic transfer function, the sharp dips observed in Figure 8.7 return. This contradicts the explanation given for their absence in the previous experiment.



in_1	in_2	out
0.5	0.5	0.4932
0.5	1.5	1.5132
1.5	0.5	1.5196
1.5	1.5	0.5088

Figure 8.9: Convergence of a network with sinusoidal synapses and neurons. Inset: XOR truth table produced by the network after convergence. $\eta = 0.1$; $n = 5$.

Noisy signals

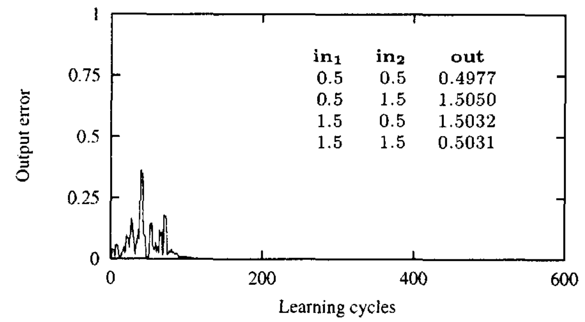
The effect of additive noise was analyzed by superposing −20 dB noise on the transfer function. The resulting convergence graph is shown in Figure 8.10. As can be seen, the presence of noise seems to be beneficial for convergence.

The amount of noise present in actual hardware devices is not accurately known yet, but is expected to be severely dependent on the temperature [1].

Errors in the learning algorithm

The two types of hardware errors for the learning algorithm, introduced in Section 8.2.3 were analyzed:

- Sign errors,



in$_1$	in$_2$	out
0.5	0.5	0.4977
0.5	1.5	1.5050
1.5	0.5	1.5032
1.5	1.5	0.5031

Figure 8.10: Network with periodic synapses and neurons and −20 dB additive noise. Convergence of the network during 500 learning cycles. Inset: XOR truth table produced by the network. $\eta = 0.1$; $n = 8$

- Step-size errors.

Sign errors occur when the learning algorithm stored the sign of the weight change incorrectly. This results in errors when restoring the weight to its previous value. Simulations show that up to 30% of the stored weights may contain a sign error without affecting the convergence of the learning algorithm [70].

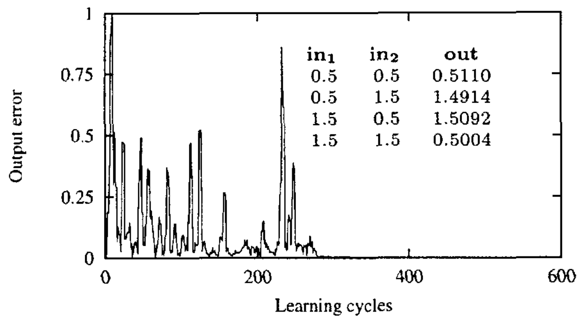
Step-size errors occur when the absolute value of the weight change δW is incorrect when restoring the weight to its previous value. When the step to restore the weight randomly varies between 0 and its nominal value, the learning algorithm still converges. When both types of errors are combined however, convergence is much more difficult. Up to 10% of both errors can still be corrected by the learning algorithm simultaneously, as shown in Figure 8.11.

8.4 Conclusions

A two-layer neural network with periodic neurons and synapses can be trained to classify a predefined function such as the XOR function.

The random weight change learning algorithm is suitable for training the network dealt with in this chapter. It complies to the following requirements: it functions independently of the specific neuron and synapse transfer function and it is a supervised learning algorithm so that when the learning procedure converges to a solution, the accuracy of the learnt function only depends on the teacher, the generation of the error signal, and the presence of enough loop-gain.

The algorithm also has several features that facilitate implementing it in hardware for very large neural networks. It uses locally available signals to



in_1	in_2	out
0.5	0.5	0.5110
0.5	1.5	1.4914
1.5	0.5	1.5092
1.5	1.5	0.5004

Figure 8.11: Convergence of the network during 500 learning cycles with random errors in the learning circuitry. Inset: XOR truth table produced by the network. $\eta = 0.1$; $n = 5$; sign error probability=0.1; relative weight error = 0.1.

determine weight updates. Second, the algorithm does not include complicated operations, a prerequisite for a compact implementation. Third, all weights are adjusted in parallel, which is advisable for fast learning algorithms. Finally, the algorithm is robust to inaccuracies in its own hardware, allowing it to be implemented in for example SET technology.

In spite of the above mentioned assets, this learning algorithm is probably not the optimum choice for large neural systems. As a result of its stochastic character, convergence may be too slow for large systems.

We can however conclude from the results presented in this chapter that despite the non-standard transfer functions of the SET synapses and neurons, small neural networks with their periodic functions can learn to classify simple functions.

9 Conclusions and future research

This thesis shows that Single Electron Tunneling (SET) transistors promise an efficient realization of neural network hardware. The two basic functional blocks of a neural network, namely the synapse and the neuron, can each be made with only a single SET transistor.

Two factors are important for a compact neural cell design. First, the technological devices used should have small feature size and low power consumption. Hence, SET technology was chosen because the operation of the SET transistor is based on the quantized nature of charge, and is therefore a very small and low power device. Second, the inherent processing power of the devices should be exploited. For the SET transistor this implies that its transfer function periodicity and non-linearity must be explicitly employed.

The neural network system concept was chosen for two main reasons: first the high processing power of an array of non-linear cells in parallel has the prospect of making powerful signal processors possible and second, the adaptability of a self-learning system makes it in principle robust to implementations prone to inaccuracies and errors. Such non-idealities are unavoidable in compact hardware for very large systems because the smallest hardware building blocks are inherently inaccurate.

The single-SET transistor synapse and neuron devices presented in this thesis owe their compactness to efficient exploitation of the non-linearity and periodicity of the devices, and rely on the adjustment skills of the neural network system concept to bridge the gap between the formal neural function description and the actual SET transistor properties.

In addition to inaccuracies, the SET transistor's functionality is severely affected by the consequences of random offset charges. The success of using the devices depends on how those non-idealities can be handled in the design. For the SET synapse and neuron, the random offset charges are equivalent to random weight and threshold values respectively, and should therefore be handled by a learning algorithm, which adjusts those values anyway.

The SET synapse and neuron devices can be interconnected to form a classifying neural network. The most important restriction originates from the limited capacitive fan-in of SET transistors. Only networks with sparsely connected topologies can therefore be realized. The maximum number of synapses per neuron depends on the voltage gain of the SET transistors used.

A successful learning algorithm for a SET neural network can cope with the specific properties of SET transistors, and is robust to errors that may occur in its own hardware. An algorithm that complies to these requirements for small neural networks is the Random Weight Change learning algorithm. It can train a small neural network of periodic synapses and neurons to perform simple functions, and can correct the type of errors that would occur if part of the algorithm were implemented with SET devices itself.

Many open questions remain to be answered before large neural networks in SET technology find their way into applications. At the technological level, smaller devices will be needed to allow operation at higher temperatures. For building large systems, higher yields are indispensable and less offset charge fluctuations desirable.

At the circuit level, SET technology can almost certainly be further exploited by employing the possibility of individual electron transport. The periodic transfer function can probably be better exploited, for example by using a hierarchical classification method based on the modulo operation of periodic functions. The fan-in problem of SET transistors is hopefully solvable by using a different circuit configuration. A study of basic electronic SET circuits could yield interesting and useful circuit configurations.

At the neural network level, the effects of the transfer function periodicity on the stability of feedback loops is one of the serious issues to be addressed when SET feed-back networks are studied. The classification properties and application areas of large neural networks with sparsely connected topologies is another important issue to work on.

Finally, all facets of the learning behavior of SET neural networks must be thoroughly investigated. Among the open questions to be addressed, we find convergence and stability, and compact hardware implementation with imprecise and stochastic hardware.

Even though many unanswered questions remain, we can conclude that compact neural building blocks in SET technology promise to be a successful approach for realizing compact, neural networks.

A

Simulation parameters

The single electron tunneling simulation program SIMON—Simulation of Nano structures [38] was used to calculate most of the curves shown in this thesis. As explained in Section 4.4.2, the algorithm used by the program to calculate the circuit behavior is based on the 'orthodox theory of electron tunneling' [21,27], which yields accurate results for an arbitrary circuit configuration as long as the junctions have a high tunnel resistance: $R_T \gg R_K \approx 26$ kΩ. The algorithm is based on calculating each individual tunnel event separately. Since tunneling is a stochastic process, the program averages many tunnel events to find the circuit behavior for a certain time step. For this thesis, the quasi-stationary Monte Carlo simulation method was used.

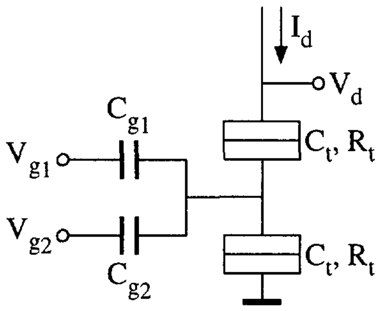


Figure A.1: The device parameters of the SET transistor used for the simulations with SIMON used in this thesis.

All the SET transistor simulations in this thesis were done on essentially the same SET transistor, which is shown in Figure A.1. Only the gate capacitance and the biasing varied, depending on the application.

The simulation parameters that were used throughout this thesis are listed below.

Parameter	**Value**	**Description**
Tunnel order	1	No co-tunneling
Simulation time	1 second	Only quasi-stationary behavior
'Event number'	10^4 – 10^5	Number of tunnel events averaged per time step

Neither current sources nor ohmic resistors are currently available in SIMON. A current source is therefore modeled by the Thevenin equivalent, replacing the resistor by a tunnel junction with zero capacitance, shown in Figure A.2. The most important difference with an ideal current source, besides its limited

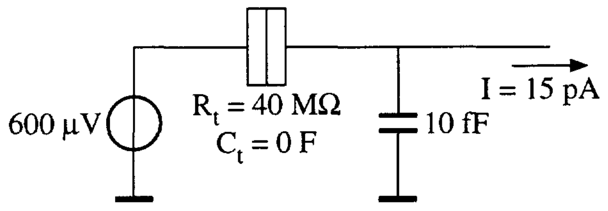


Figure A.2: Thevenin equivalent of a current source using a tunnel junction as an impedance.

impedance, is that it does not produce a steady stream of electrons, but rather a stochastic one, where the individual tunnel events are described by a Poisson distribution.

Bibliography

[1] S.M. Verbrugh, *Development of a single electron turnstile as a current standard*, Ph.D. thesis, Delft University of Technology, 1995.

[2] E.H. Visscher, *Technology and applications of single-electron tunneling devices*, Ph.D. thesis, Delft University of Technology, 1996.

[3] A.W. Wieder, "Status, trends and challenges in microelectronics for the next 10 to 15 years," *Electrical Engineering*, vol. 79, pp. 79–84, 1996.

[4] J. Camargo da Costa, M. Goossens, C. Verhoeven, and A. van Roermund, "Applications of single-electron transistors," in *Proceedings of the Workshop on circuits, Systems and signal processing*, Utrecht, 19971, pp. 109–114, STW Technology Foundation.

[5] R.R. Birge, B. Parsons, Q.W. Song, and J.R. Tallent, "Protein-based three-dimensional memories and associative processors," in *Molecular Electronics*, J. Jornter and M. Ratner, Eds., chapter 15, pp. 439–472. Blackwell Science Ltd, Oxford, 1997.

[6] J.J.H.B. Schleipen, S.B. Colak, E.C. Mos, and C.T.H. Liedenbaum, "An injection laser neural network," in *Proceedings of the Fourth International Conference on Microelectronics for Neural Networks and Fuzzy Systems*, Los Almitos, California, 1994, pp. 8–11, IEEE Computer Society Press.

[7] R.F. Lyon and L.S. Yaeger, "On-line hand-printing recognition with neural networks," in *Proceedings of the 5th International Conference on Microelectronics for Neural Networks and Fuzzy Systems (MicroNeuro'96)*, 1996, pp. 201–212.

[8] W. Eppler, T. Fischer, H. Gemmeke, T. Becher, and G. Kock, "High speed neural network chip on PCI board," in *Proceedings of the 6th International Conference on Microelectronics for Neural Networks, Evolutionary and Fuzzy Systems (MicroNeuro'97)*, 1997, pp. 9–17.

[9] P. Masa, *NeuroClassifier. Analog VLSI neural network for very high speed pattern classification*, Ph.D. thesis, Twente University, 1995.

[10] P.M. Petroff, "Quantum structure with reduced dimensionality and quantum functional devices," *Future Electron Device Journal*, vol. 5, no. Suppl.2, pp. 5–19, 1994.

[11] K.F. Goser, C. Pacha, A. Kanstein, and M.L. Rossmann, "Aspects of system and circuits for nanoelectronics," *Proceedings of the IEEE*, vol. 85, no. 4, pp. 558–573, April 1997.

[12] S. Bandyopadhyay and V.P. Roychowdhury, "Computational paradigms in nanoelectronics: quantum coupled single electron logic and neuromorphic networks," *Japanese Journal of Applied Physics, Part 1*, vol. 35, no. 6A, pp. 3350–3362, 1996.

[13] V.P. Roychowdhury, D.B. Janes, and S. Bandyopadhyay, "Nanoelectronic architecture for boolean logic," *Proceedings of the IEEE*, vol. 85, no. 4, pp. 574–588, 1997.

[14] V.P. Roychowdhury, D.B. Janes, S. Bandyopadhyay, and X. Wang, "Collective computational activity in self-assembled arrays of quantum dots: A novel neuromorphic architecture for nanoelectronics," *IEEE Transactions on Electron Devices*, vol. 43, no. 10, pp. 1688–1699, 1996.

[15] J. Hoekstra, J. Camargo da Costa, M.J. Goossens, C.J.M Verhoeven, and AHM van Roermund, "The application of neural networks for nanoelectronic circuits," in *Neural Networks and their Applications (NEURAP)*, N.Giambiasi, Ed., Marseille, France, 1998, pp. 27–30, IUSPIM.

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

[17] B.K. Dolenko and H.C. Card, "Tolerance to analog hardware of on-chip learning in back propagation networks," *IEEE Transactions on Neural Networks*, vol. 6, no. 5, pp. 1045–1052, September 1995.

[18] J.B. Lont and W. Guggenbühl, "Analog CMOS implementation of a multilayer perceptron with nonlinear synapses," *IEEE Transactions on Neural Networks*, vol. 3, no. 3, pp. 457–465, 1992.

[19] P.W. Hollis and J.J. Paulos, "A neural network learning algorithm tailored for VLSI implementation," *IEEE Transactions on Neural Networks*, vol. 64, pp. 784–791, 1994.

[20] R. Otte, *Low-Power Wireless Optical Transmission*, Ph.D. thesis, Delft University of Technology, 1998.

[21] K.K. Likharev, "Single-electron transistors: Electrostatic analogs of the DC SQUIDS," *IEEE Tansactions on Magnetics*, vol. MAG-23, no. 2, pp. 1142–1145, March 1987.

[22] R.H. Chen, A.N. Korotkov, and K.K. Likharev, "Single electron transistor logic," *Applied Physics Letters*, vol. 68, no. 14, pp. 1954–1956, April 1996.

[23] M.I. Lutwyche and Y. Wada, "Estimate of the ultimate performance of the single-electron transistor," *Journal of Applied Physics*, vol. 75, no. 7, pp. 3654–3661, April 1994.

[24] A.N. Korothov, R.H. Chen, and K.L. Likharev, "Possible performance of capacitively coupled single-electron transistors in digital circuits," *Journal of Applied Physics*, vol. 78, no. 4, pp. 2520–2530, 1995.

[25] J.R. Tucker, "Complementary digital logic based on the 'Coulomb blockade'," *Journal of Applied Physics*, vol. 72, no. 9, pp. 4399–4413, November 1992.

[26] R.H. Chen and K.K. Likharev, "Multiple-junction single-electron transistors for digital applications," *Applied Physics Letters*, vol. 72, no. 1, pp. 61–63, 1998.

[27] H. Grabert and M.H. Devoret, Eds., *Single Charge Tunneling, Coulomb Blockade Phenomena in Nanostructures*, vol. 294 of *NATO ASI Series B:Physics*, Plenum, New York, 1992.

[28] A.B. Zorin, F.J. Ahlers, J. Niemeyer, T. Weimann, and H. Wolf, "Background charge noise in metallic single-electron tunneling devices," *Physical Review B*, vol. 53, no. 20, pp. 13682–13687, May 1996.

[29] A. Murray and L. Tarassenko, *Analogue Neural VLSI, A pulse stream approach*, Chapman & Hall Neural Computing Series. Chapman & Hall, London, 1994.

[30] R.J. van de Plassche, *High-speed and High-Resolution Analog-to-Digital and Digital-to-Analog Converters*, Ph.D. thesis, Delft University of Technology, 1989.

[31] S.J. Ahn and D.M. Kim, "Asynchronous analogue-to-digital converter for single-electron circuits," *Electronics letters*, vol. 34, no. 2, pp. 172–173, January 1998.

[32] J. Pettersson, P. Wahlgren, P. Delsing, D.B. Haviland, and T. Claeson, "Extending the high-frequency limit of a single-electron transistor by on-chip impedance transformation," *Physical review B*, vol. 53, no. 20, pp. 13272–13278, May 1996.

[33] A.N.R. van Neerbos, "Design of a cryogenic transimpedance amplifier for single electron tunneling devices," M.S. thesis, Delft University of Technology, 1996.

[34] K.K. Likharev and A.N. Korotkov, "Ultradense hybrid SET/FET dynamic RAM: Feasibility of background-charge-independent room-temperature single-electron digital circuits," in *Proceedings of the ISDRS*, Charlottesville, VA, 1995.

[35] G. Zimmerli, R.L. Kautz, and J.M. Martinis, "Voltage gain in the single electron transistor," *Applied Physics Letters*, vol. 61, no. 21, pp. 2616–2618, November 1992.

[36] M. Kirihara and K. Taniguchi, "A single electron neuron device," *Japanese Journal of Applied Physics*, vol. 36, no. 6B, pp. 4172–4175, June 1997.

[37] P. Hadley, ," Private communication.

[38] C. Wasshuber and H. Kosina, "A single-electron device and circuit simulator," *Superlattices and Microstructures*, vol. 21, no. 1, pp. 37–42, 1997.

[39] M.H.L. Kouwenhoven, *High Performance Frequency Demodulation Systems*, Ph.D. thesis, Delft University of Technology, 1998.

[40] J.M. Zurada, *Introduction to Neural Systems*, West Publishing Company, 1992.

[41] M.R.W. Dawson and D.P. Schopflocher, "Modifying the generalized delta rule to train networks of non-monotonic processors for pattern classification," *Connection science*, vol. 4, no. 1, pp. 19–31, 1992.

[42] A. Malinowski, T.J. Cholewo, and J.M. Zurada, "Capabilities and limitations of feedforward neural networks with multilevel neurons," in *Proc. IEEE ISCAS*, 1995, pp. 131–134.

[43] K. Hara and K. Nakayama, "Comparison of activation functions in multilayer neural network for pattern classification," in *Proceedings of the IEEE International Conference on Neural Networks*, 1994, pp. 2997–3002.

[44] F.C. Hoppensteadt, *An introduction to the mathematics of neurons*, Cambridge Studies in Mathematical Biology. Cambridge University Press, Cambridge, 1986.

[45] C. P. Heij, D. C. Dixon, P. Hadley, and J. E. Mooij, "A single-electron tunneling device displaying negative differential resistance," 1998, Submitted to Applied Physics Letters.

[46] D.B. Mc.Caughan, "On the properties of periodic perceptrons," in *Proceedings of the IEEE International Conference on Neural Networks*, 1997, pp. 188–192.

[47] C. Pacha and K. Goser, "Application of terminal dynamics in cellular neural networks," in *Proceedings of the 5th International Conference on Microelectronics for Neural Networks (MicroNeuro'96)*. 1996, pp. 305–310, IEEE Computer Society Press, Los Alamitos, CA.

[48] A.J. Montalvo, R.S. Gyurcsik, and J.J. Paulos, "An analog VLSI neural network with on-chip perturbation learning," *IEEE Journal of Solid State Circuits*, vol. 32, no. 4, pp. 535–543, April 1997.

[49] J.A. Lansner and T. Lehmann, "An analog CMOS chip set for neural networks with arbitrary topologies," *IEEE Transactions on Neural Networks*, vol. 4, no. 3, pp. 441–444, May 1993.

[50] A.J. Annema, *Analysis, Modelling and Implementation of Analog Integrated Neural Networks*, Ph.D. thesis, University of Twente, 1994.

[51] Y. Arima, K. Mashshiko, K. Okada, T. Yamada, A. Maeda, H.Kondoh, and S. Kayano, "A self-learning neural network chip with 125 neurons and 10k self-organization synapses," *IEEE Journal of Solid State Circuits*, vol. 26, no. 4, pp. 607–611, 1991.

[52] K.Hirotsu and M.A. Brooke, "An analog neural network chip with random weight change learning algorithm," in *Proceedings of 1993 international Joint Conference on Neural Networks*, 1993, pp. 3031–3034.

[53] M.J. Goossens, C.J.M Verhoeven, and A.H.M van Roermund, "Concepts for ultra-low-power and very-high-density single-electron neural networks," in *International Symposium on nonlinear Theory and its Applications*, 1995, pp. 679–682.

[54] M.J. Goossens, C.J.M Verhoeven, and A.H.M van Roermund, "Single electron tunneling technology for neural networks," in *Proceedings of the 5th International Conference on Microelectronics for Neural Networks (MicroNeuro'96)*, Los Alamitos, CA, 1996, pp. 125–130, IEEE Computer Society Press.

[55] M. Akazawa and Y. Amemiya, "Eliciting the potential functions of single-electron circuits," *IEICE transactions electron.*, vol. E80, no. 7, pp. 849–858, July 1997.

[56] M. Akazawa and Y. Amemiya, "Boltzmann machine neuron circuit using single-electron tunneling," *Applied Physics Letters*, vol. 70, no. 5, pp. 670–672, February 1997.

[57] F Palmieri and Jie Zhu, "Self-association and Hebbian learning in linear neural networks," *IEEE Transactions on Neural Networks*, vol. 6, no. 5, pp. 1165–1184, september 1995.

[58] F Palmieri, J Zhu, and Chang C., "Anti-Hebbian learning in topologically constrained linear networks: A tutorial," *IEEE Transactions on Neural Networks*, vol. 4, no. 5, pp. 748–761, september 1993.

[59] P. Masa, K. Hoen, and H. Wallinga, "A high speed analog neural processor," *IEEE Micro*, pp. 40–50, June 1994.

[60] L.J. Geerligs, V.F. Anderegg, P.A.M. Holweg, J.E. Mooij, H. Pothier, D. Esteve, C. Urbina, and M.H. Devoret, "Frequency-locked turnstile device for single electrons," *Phys. Rev. Lett.*, vol. 64, pp. 2691, 1990.

[61] H. Pothier, P. Lafarge, P.F. Orfila, C. Urbina, D. Esteve, and M.H. Devoret, "Single electron pump fabricated with ultrasmall normal tunnel junctions," *Physica B*, vol. 169, pp. 573–574, 1991.

[62] L.O. Chua and L. Yang, "Cellular neural networks: Theory," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1257–1772, October 1988.

[63] L.O. Chua and L. Yang, "Cellular neural networks: Applications," *IEEE Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1273–1290, October 1988.

[64] L.O. Chua and T. Roska, "The CNN paradigm," *IEEE Transactions on Circuits and Systems- I: Fundamental Theory and Applications*, vol. 40, no. 3, pp. 147–156, March 1993.

[65] G.J. Dolan, "Offset masks for lift-off photoprocessing," *Applied Physics Letters*, vol. 31, no. 5, pp. 337–339, September 1977.

[66] B. Burton, F. Kamran, R.G. Harley, T.G. Habetler, M. Brooke, and R. Poddar, "Identification and control of induction motor stator currents using fast on-line random training of a neural network," in *Conference record of the 1995 IEEE Industry Application Conference*, New York, 1996, pp. 1781–1787.

[67] B. Widrow and M.A. Lehr, "30 years of adaptive neural networks: Perceptron, madaline, and backpropagation," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1415–1442, 1990.

[68] D. Andes, B. Widrow, M. Lehr, and E. Wan, "MRIII: A robust algorithm for training analog neural networks," in *Proc. Int. Joint Conf. Neural Networks*, 1990, vol. I, pp. 553–556.

[69] M. Jabri and B. Flower, "Weight perturbation: An optimal architecture and learning technique for analog VLSI feedforward and recurrent multilayer networks," *IEEE Transactions on Neural Networks*, vol. 3, no. 1, pp. 154–157, January 1992.

[70] J.H. Ritskes, "Self-learning capabilities of neural nets in SET-technology," M.S. thesis, Delft University of Technology, 1997.

[71] M.J. Goossens, J.H. Ritskes, C.J.M Verhoeven, and A.H.M van Roermund, "Neural networks with periodic single electron tunneling transistors," in *Proceedings of the European Conference on Circuit Theory and Design*, Á.I. Csurgay and T. Roska, Eds., Budapest, 1997, pp. 936–941, Technical University of Budapest.

[72] M.J. Goossens, J.H. Ritskes, C.J.M Verhoeven, and A.H.M van Roermund, "Self-learning capabilities of neural nets in SET-technology," in *Proceedings of the 6th International Conference on Microelectronics for Neural Networks, Evolutionary & Fuzzy Systems*, H. Klar, A. König, and U. Ramacher, Eds., Dresden, 1997, pp. 217–224, University of Technology Dresden.

[73] M.J. Goossens, J.H. Ritskes, C.J.M Verhoeven, and A.H.M van Roermund, "Learning single electron neural nets," in *Proceedings of the 8th annual workshop on Circuits, Systems, and Signal Processing*, 1997, pp. 936–941.

Summary

In this thesis, compact Single Electron Tunneling building blocks for large artificial neural networks are presented.

Chapter 1 introduces the thesis, and Chapter 2 gives an overview of the field of neural network implementations. It is argued that the neural network system concept is valuable for powerful signal processing systems, but that successful large scale applications are impossible as long as no compact hardware implementation is available. The solution is to optimally use the available physical and technological properties, which results in compact *neural devices* that perform the primitive neural functions.

Chapter 3 describes the two basic principles on which the design of the compact neural cells is based. First of all, the design should not focus on strictly defined primitive function definitions. An attempt to accurately implement those functions does not yield the most compact realization possible because the device properties are not optimally exploited. Second, the neural network system concept is used because hardware inaccuracies are unavoidable in compact designs. The resulting uncertainties can be eliminated at the system level by combining the results of large numbers of devices, and by the learning algorithm of the neural network.

The Single Electron Tunneling (SET) transistor, which is the basic device used in this thesis, is the subject of Chapter 4. Its operation is based on the tunneling of individual electrons through the device. The tunnel events are influenced by the voltage on a capacitive gate. The transfer function from gate to output is a periodic function of the gate voltage. This property makes the device unique but also versatile, as shown in the following chapters. The device unfortunately also has an important drawback: it suffers from random offset charges, which seriously influence its functionality. The success of any design with SET transistors depends on the consequences these offsets have for the specific application. The success of SET neural networks in particular also depends on the consequences the limited fan-in of the devices has on the network topology, and on the ability of the learning algorithm to cope with the specific SET properties.

Chapter 5 presents two ways of making the neural activation function with SET transistors, and compares them with two other SET neurons known from the literature. The single-SET neuron described in this chapter is the favorite one because it is compact, and because its random offset charge is equivalent to a random neuron threshold value, which can be adjusted by a learning algorithm. The non-linear transfer function of this neuron is formed by part of one period of the SET transistor transfer function.

A single SET transistor is also sufficient for the synapse, as shown in Chapter 6. The synaptic multiplication is obtained from the small-signal behavior of the SET transistor transfer function. The differential slope of this transfer function depends on the operating point on the large signal function. The resulting function is therefore similar to a synaptic multiplier if the operating point is controlled by the synaptic weight. In this case, the random offset charge has an effect equivalent to a random weight value, and all synaptic weights are adjusted by the learning algorithm anyway.

In Chapter 7, the interconnection between SET synapses and neurons is analyzed. The limited fan-in and voltage gain of SET transistors results in the choice of a sparsely connected network topology. A small one-layer neural network of SET neuron and synapse devices can perform elementary classification tasks needed for neural network operation. The decision boundary between an 'active' and an 'inactive' neuron output is positioned in the input space by adjusting the synaptic weights and neuron thresholds.

A learning algorithm is indispensable for neural networks. Chapter 8 analyzes the required properties of a learning algorithm for SET neural networks and subsequently describes how the random weight change learning algorithm adjusts all the synaptic weights and neuron thresholds simultaneously by measuring the effect of small random weight changes. This makes the algorithm independent of the specific neural cell implementation. With this learning algorithm, a two-layer SET neural network can learn the XOR function, even in the presence of the type of errors that may occur if part of the algorithm itself were also implemented in SET technology.

The SET neuron device and the SET synapse device are extremely compact neural cell implementations, which show that respecting the device properties and relying on the adaptiveness of the neural system is a successful approach towards large and compact artificial neural networks.

Samenvatting

Dit proefschrift beschrijft hoe compacte bouwblokjes voor grote artificiële neurale netwerken gemaakt kunnen worden met *Single-Electron Tunneling* transistoren. 'Single-electron tunneling' verwijst naar het verschijnsel dat de elektronen per stuk door dit type transistor stromen.

Na het inleidende hoofdstuk 1, geeft hoofdstuk 2 een kort overzicht van de problemen en mogelijkheden die zich voordoen bij het implementeren van elektronische neurale netwerken. Grote geïntegreerde neurale netwerken zullen naar verwachting krachtige signaalprocessoren opleveren, maar de realisatie van dergelijke netwerken is vooralsnog onmogelijk omdat er geen compacte implementatievorm beschikbaar is. Door de fysische en technologische eigenschappen optimaal te benutten, kunnen de primitieve functies van een neuraal netwerk gerealiseerd worden door middel van compacte *neurale devices*, een implementatievorm die naar verwachting de realisatie van grote neurale netwerken wel mogelijk maakt.

Hoofdstuk 3 beschrijft de twee basisgedachten die ten grondslag liggen aan het ontwerp van de compacte neurale bouwblokjes. Allereerst is het van belang dat men zich niet aan strikt omschreven primitieve functiedefinities houdt. Een nauwgezette realisatie van deze functies zal namelijk in het algemeen niet het meest compacte ontwerp opleveren, omdat de technologische eigenschappen dan niet optimaal gebruikt worden. Ten tweede worden neurale netwerken gebruikt, omdat onnauwkeurigheden onvermijdelijk zijn in compacte elektronische circuits. De onzekerheden die hieruit voortvloeien kunnen op systeemniveau geëlimineerd worden door op geschikte wijze de resulaten van zeer vele devices te combineren en door het leermechanisme van het neurale netwerk.

Hoofdstuk 4 behandelt uitgebreid de basisbouwsteen uit dit proefschrift: de Single Electron Tunneling (SET) transistor. De werking van de SET-transistor is gebaseerd op het tunnelen van individuele elektronen door het device. Met een spanning op de gate van de transistor kan het tunnelproces beïnvloed worden, hetgeen aan de uitgang van het device kan worden gedetecteerd. Het bijzondere van de SET-transistor is dat de overdrachtsfunctie van de gate naar de uitgang een periodieke functie is. Zoals zal blijken uit de volgende hoofdstukken, is

de SET-transistor hierdoor niet alleen een unieke, maar vooral ook een veelzijdige component. De SET-transistor heeft echter ook een belangrijke nadelige eigenschap. Zijn gedrag wordt namelijk sterk beïnvloed door willekeurige *offset*-ladingen. Het succes van een toepassing van SET-transistoren hangt dan ook sterk af van de invloed van deze offset-ladingen op de functionaliteit van het ontwerp.

In hoofdstuk 5 worden twee configuraties van SET-transistoren beschreven waarmee de activatiefunctie van het neuron gerealiseerd kan worden. Uit een vergelijking met twee andere SET-neuronen uit de literatuur, blijkt dat een van de in dit hoofdstuk beschreven neuronen de voorkeur geniet. De betreffende neuron bestaat uit slechts één SET-transistor, wat een zeer compacte implementatie is. Bovendien is zijn willekeurige offset-lading equivalent met een willekeurige waarde van de drempelwaarde van het neuron, waardoor de offset door het leeralgoritme bijgeregeld kan worden. De niet-lineaire overdrachtsfunctie van dit neuron wordt gevormd door een gedeelte van een periode van de overdrachtsfunctie van de SET-transistor.

Hoofdstuk 6 laat zien hoe met een enkele SET-transistor ook een synaps gemaakt kan worden. De vermenigvuldigingsfactor van de synaps wordt gegeven door de helling van het klein-signaalgedrag van de SET-transistor. Deze lokale helling hangt af van het instelpunt op de groot-signaalfunctie en als dit instelpunt afhankelijk gemaakt wordt van het gewicht van de synaps, is het totale gedrag van de SET-transistor vergelijkbaar met de gewenste vermenigvuldigende werking. Ook in dit geval kan het leeralgoritme de willekeurige offset afregelen, omdat de offset-lading het zelfde effect heeft als een willekeurige afwijking van de gewichtswaarde die toch al door het leeralgoritme afgesteld moet worden.

In hoofdstuk 7 komt het onderling verbinden van neuronen en synapsen aan de orde. De beperkte toegestane belasting aan de ingang van een SET-transistor en de beperkte lokale spanningsversterking maken de keuze voor een netwerktopologie met weinig verbindingen noodzakelijk. Het hoofdstuk laat zien dat een klein enkel-laags neuraal netwerk met SET-neuronen en -synapsen de elementaire neurale classificaties uit kan voeren: de positie van de scheidingslijn tussen een actieve en een niet-actieve neuronuitgang kan namelijk veranderd worden met behulp van de gewichtswaarden van de synapsen en de drempelwaarden van de neuronen.

Het leeralgoritme is een onmisbaar onderdeel van een neuraal netwerk. Hoofdstuk 8 analyseert waaraan een leeralgoritme voor neurale netwerken op basis van SET-transistoren moet voldoen. Vervolgens wordt het *Random Weight Change* leeralgoritme beschreven, dat kleine willekeurige veranderingen aanbrengt aan de gewichten en drempelwaarden om te bepalen hoe deze aangepast moeten worden zodat het netwerk het gewenste gedrag gaat vertonen. Hierdoor is het algoritme niet afhankelijk van specifieke synaps- en neuroneigenschappen en kan het een tweelaags neuraal netwerk van SET-transistoren de XOR-functie leren.

Dat lukt zelfs als er fouten optreden van het type dat voor zou komen indien een gedeelte van het leeralgoritme zelf ook in SET-technologie geïmplementeerd wordt.

Het SET-neuron en de SET-synaps uit dit proefschrift zijn bijzonder compacte implementaties van neurale cellen die laten zien dat het benutten van specifieke eigenschappen van devices en het vertrouwen op de adaptieve werking van een neuraal systeem een succesvolle weg belooft te zijn naar grote compacte artificiële neurale netwerken.

Acknowledgements

It would not have been possible to write this thesis without the help of many people.

First of all, I would like to thank my promotor, Prof.dr.ir. A.H.M. van Roermund for his guidance and all the time he spent on valuable discussions and critically reading my work. I would also like to thank my co-promotor dr.ir. C.J.M. Verhoeven for asking me to start this challenging and intriguing work for his fresh and innovative ideas during the numerous discussions we have had, and for his scrupulous manuscript reading.

I would also like to express my gratitude to Dr. Jaap Hoekstra, who stimulated me enormously during the final year with the many long and short deliberations, his prompt and critical reading, his neural network expertise, and his overall encouragement.

I am also greatly indebted to Dr. José Camargo da Costa for his enthusiasm, his stimulating optimism, his friendship and the lengthy and inspiring discussions.

I would like to thank Dr. Peter Hadley for patiently answering all my questions on SET transistors, and the members of the Quantum Transport Lab for their contributions to the pleasant meetings on Friday mornings.

My thanks go the M.Sc. students who contributed to this work. Arnoud van Neerbos for his work on a cryogenic amplifier, Jacob Ritskes for writing the computer program and obtaining the simulation results of Chapter 8, and Roelof Klunder for his critical remarks, and especially for writing the computer program *Simone*, the invaluable partner for 'SIMON'.

Yifang Chen and Caspar v.d. Wal are acknowledged for showing me how to behave in a cleanroom, and patiently teaching me the principles of making SET devices. Michiel Koolen and Simon Bootsma for their efforts to produce working devices. Alas, the odds were against us! And José Camargo Da Costa and Paul Teunissen for working out a thorough measurement setup. It is a pity I never got a chance to use it.

Special thanks go to Rob Janse for drawing and editing almost all the diagrams of this thesis, Simon North for finding and correcting numerous language

errors, to my roommates Jan Mulder and Frank Kuijstermans for the pleasant working days, and to all the other members of the Electronics Lab for providing a stimulating atmosphere.

Finally, I am very grateful to my parents for all their support and to my family and friends for all the relaxing moments.

Biography

Martijn Goossens was born in Amsterdam, The Netherlands, on the 10th of February, 1970. He started his studies in Electrical Engineering at the Delft University of Technology in 1989, and was awarded the M.Sc. degree in 1993. Subsequently, he joined the Electronics Research Laboratory at the same university to work towards this Ph.D. thesis. In 1998 he joined Philips Research Laboratories.