Department of Physics and Astronomy
University of Heidelberg

Diploma thesis

in Physics

submitted by

Thomas Pfeil

born in Stuttgart - Bad Cannstatt

January 2011

Configuration Strategies for Neurons and Synaptic Learning in Large-Scale Neuromorphic Hardware Systems

This diploma thesis has been carried out by Thomas Pfeil at the

Kirchhoff Institute for Physics

under the supervision of

Prof. Dr. Karlheinz Meier

Configuration Strategies for Neurons and Synaptic Learning in Large-Scale Neuromorphic Hardware Systems

In this thesis, the neurons and plastic synapses of a novel wafer-scale neuromorphic hardware system developed within the FACETS and BrainScaleS research projects are analyzed by software simulations and by measurements with prototype devices. In order to overcome the historical obstacles towards a highly configurable, large-scale neuromorphic hardware system with plastic synapses, a trade-off between the number of neurons and synapses and the required chip resources has to be made. The major reduction in required resources, as realized by the FACETS system, results in a small-sized, customized mixed-signal circuitry that – especially in the case of synapses – brings up limitations compared to standard computer floating point arithmetic. The intrinsic inhomogeneities and noisiness in real biological systems suggest that such limitations should not critically impede the general functionality of spike-timing dependent plasticity, which is often implemented in established neural network models. The effects of discretizing synaptic weights, as done in hardware implementations of the considered type, are studied by step-wise increasing the level of test scenario complexity towards a network benchmark for synchrony detection. Production imperfections are measured in hardware experiments and integrated into extended studies. Furthermore, a neuron fitting environment is presented that allows an automatic fitting of hardware neurons to arbitrary electrophysiological data, either obtained by biological recordings, or reference software simulations. First experiments show that the applied particle swarm algorithm is capable of optimizing the neuron parameters, and that the prototype test setup calibrated this way can be operated successfully and with high reliability.

Strategien zur Konfiguration von Neuronen und synaptischem Lernen in großskaligen neuromorphen Hardwaresystemen

In dieser Diplomarbeit werden die Neurone und plastischen Synapsen eines innovativen neuromorphen Hardwaresystems, das im Rahmen der Forschungsprojekte FACETS und BrainScaleS entwickelt wird, anhand von Softwaresimulationen und Messungen an Prototypen untersucht. Um die bisherigen Schwierigkeiten in der Entwicklung eines hochkonfigurierbaren, großskaligen neuromorphen Hardwaresystems mit plastischen Synapsen zu überwinden, muss ein Kompromiss zwischen der Anzahl an Neuronen und Synapsen und deren Schaltungsaufwand eingegangen werden. Im FACETS-System angewandte, drastische Einsparungen in der Belegung von Hardwareressourcen – insbesondere für Synapsen – führen zu kleinen, zugeschnittenen Mixed-Signal Schaltkreisen, welche im Gegensatz zur herkömmlichen Fließkomma-Arithmetik funktionale Einschränkungen zur Folge haben. Die Ungleichmäßigkeiten und das Rauschen in biologischen Systemen legen nahe, dass solche Einschränkungen die Funktionalität der korrelationsbasierten synaptischen Plastizität (STDP), die oft in etablierten neuronalen Netzwerken eingesetzt wird, nicht grundlegend beeinflussen sollte. Die auftretenden Effekte einer Diskretisierung von synaptischen Gewichten, wie sie in der betrachteten Art von Hardwaresystemen verwendet wird, werden analysiert, indem die Komplexität der Testszenarien schrittweise erhöht wird bis hin zu einem Netzwerk, welches Synchronizität erkennen kann. In Experimenten mit der Hardware werden Produktionsschwankungen gemessen, deren Charakteristik für weitere Studien verwendet wird. Außerdem wird eine Umgebung präsentiert, die es ermöglicht, Hardwareneurone an beliebige elektrophysiologische Daten, die entweder aus biologischen Aufzeichnungen oder Referenzsimulationen gewonnen werden können, automatisch anzupassen. Erste Experimente bestätigen, dass der verwendete Partikel-Schwarm Algorithmus zur Optimierung von Neuronparametern geeignet ist, und dass der Prototypaufbau in dieser Umgebung mit hoher Verlässlichkeit betrieben werden kann.

Contents

1 Introduction

The desire to explore the human brain goes at least back to Hippocrates (400 BC), the father of Western medicine. Over two thousand years later Santiago Ramón y Cajal (DeFelipe & Jones, 1988) was one of the first scientists who published detailed descriptions of the human brain structure on a microscopic scale. Besides studies on nerve cells, he provided evidence for their connections through synapses and depicted the connectivity structure.

Nowadays scientists have a detailed knowledge about the electrophysiology and morphology of neurons and synapses. Many models of neurons (Section 1.1), synapses (Section 1.2), complex neural mechanisms and neural network architectures have been developed and simulated on large clusters of computers. But the breakthrough in understanding the brain, and especially its capability to learn and adapt to new tasks, is still missing.

One approach is to reverse engineer the cortex (Markram, 2006), which is the outer layer of the brain, by exploring its microanatomy in as much detail as possible, constructing models and re-building the cortex structure in software. The mammalian cortex is arranged in repeating building blocks, called columns, with a minimum number of neurons of about $10^4 - 10^5$ (Markram, 2006) that is assumed to ensure the functionality of one column. The synaptic plasticity within this neural network is considered to be the biological substrate to learning and memory (Morrison et al., 2008).

However, this reverse engineering approach requires an enormous amount of resources to perform the elaborate experiments and run an expensive and power consuming cluster of computers. The performance of neural network simulations on general purpose von Neumann computer architectures is limited. The high connectivity between neurons of approximately 10^3 to 10^4 connections per cell, and hence the highly parallel information processing in neural networks, is in contrast to the sequential nature of the von Neumann information processing and causes an ineffective usage of the hardware resources in terms of chip area and power consumption.

In order to solve this dilemma, many kinds of so-called neuromorphic hardware devices have been developed. The majority is designed to operate in biological real time, mostly because of their application in medical implants(Levi et al., 2008; Fromherz, 2002). Other branches of research are focusing on highly accelerated systems that allow statistical studies on neural networks of the size of a cortical column (Schemmel et al., 2008; ?, 2010).

In this study we focus on the FACETS[1] wafer-scale neuromorphic hardware system (see Section 1.4.2), which employs analog circuits that largely obey the same differential equations as biological neurons, albeit at a highly accelerated pace and with the communication infrastructure implemented as a digital bus system. A related approach is to solve the differential equations digitally as well, but in a highly parallel fashion (Plana et al., 2007).

This wafer-scale hardware device is partly based on its predecessor, the Spikey chip (Section 1.4.1), but represents a radical revision and extension. As the maximum size of a single silicon chip is limited, a full silicon wafer is left in one piece, while its repeating building blocks, so-called reticles, are interconnected using a dedicated post-processing technology (Schemmel et al., 2010). Each reticle hosts eight chips, called HICANNs[2], that are further described in Section 1.4.2.

Throughout this study, the basic components of the HICANN - the neurons and their

[1] Fast Analog Computing with Emergent Transient States
[2] High Input Count Analog Neural Networks

synapses - are analyzed, verified, and improved. These components should be flexibly configurable and behave as measured in biology or described in theoretical models, although their circuitry is simplified in order to accelerate the system, increase the number of neurons and plastic synapses as well as reduce the power consumption. The choice of the trade-offs concerning the number and detail level of these components is crucial for the success of the resulting neuromorphic hardware device and requires a thorough analysis. The thesis at hand approaches this novel challenge with dedicated analysis methods, test scenarios and optimization methods.

In the case of neurons (Section 3), a flexible point neuron model (Section 1.1.3) is implemented by the FACETS wafer-scale hardware system. First experiments by members of our research group show that many of its spike patterns can be reproduced. However, a translation from biological to hardware parameters is required, if network models written in the simulator-independent modelling language PyNN (**?**) should be run on the FACETS wafer-scale system. One approach is calibrating the neurons with reference to software simulations of the same neuron model (Schwartz, 2011). One by one all parameters are determined by specific calibration routines. Once obtained, the translation scheme is stored in a database and can be recalled, e.g. if a set of parameters is used again.

Another approach is to follow the idea of the Quantitative Single-Neuron Modeling Competition (Section 3.1), where a neuron is identified by its spike train response to a given input. Instead of calibrating the parameters individually, all parameters can be optimized at once in order to obtain the target spike train (Rossant et al., 2010). Because this method does not ensure that the membrane potential evolves the same way, but only the spikes occur at the same time, it can be extended by a preparative linear regression (Section 3.2.1) and the combination of spike train and membrane potential trace fitting (Section 3.2.2 and Section 3.2.4).

The hardware implementation of synapses is even more important, as they are numerous, require the most hardware resources (see Figure 6) and are considered to play a fundamental role in learning. In order to maximize the number of synapses on a substrate of limited dimensions, the size of one synapse has to be as small as possible. One approach to reduce the synapse size is to lower the resolution of the synaptic weight by decreasing the number of digital bits representing it. Consequently this and other reductions of the synapse size cause functional limitations compared to floating point software simulations. Throughout Section 2 the main limitations of the hardware synapses implemented in the FACETS wafer-scale system are analyzed in regard to their effects on plastic synapses and neural networks. The limitations are isolated and implemented into a NEST[3]-based (Gewaltig & Diesmann, 2007) software synapse model, which allows to study limitations both individually and collectively. Preparative software simulations utilizing this new framework are a necessary tool to investigate and further improve the FACETS wafer-scale hardware system, even before its final production and assembly, which had not yet taken place at the time of these studies.

Outline

This thesis is divided into two main parts, each of them focused on one of the two main building blocks of one and the same FACETS wafer-scale hardware system. In Section 2 the concept of a hardware synapse is described and then analyzed, starting with preparative software

[3]Neural Simulation Technology

simulations. This method allows us to gain first experience with these plastic hardware synapses without the final hardware system. Additionally, limitations caused by the choice of the trade-offs between the number and detail level of a hardware synapse are isolated in Section 2.1 and analyzed in Section 2.4 in order to verify and improve the design concept of such a hardware synapse. Apart from theoretical studies and software simulations, the production variances of hardware synapses are measured with a real chip-based hardware system in Section 2.4.4 and their effects on neural networks is determined as described in Section 2.4.3.

The second major aspect of this thesis is the fitting of hardware neurons to reference software simulations or biological recordings as described in Section 3. A test setup of a HICANN prototype as described in Section 3.3.3) is assembled and used for first hardware parameter optimizations in Section 3.2. This method allows us to fit the hardware neurons to arbitrary reference data, which can be spike trains only or membrane potential traces.

The relevant materials and methods are introduced individually in both Section 2(synapses and plasticity) and Section 3(neuron parameter fitting).

1.1 From Biology to Neuron Models

Nerve cells, called neurons, are the principal cellular elements that underlie the function of the cortex. A great deal is known about their biophysical mechanisms responsible for neuronal activity, and this knowledge provides a basis for constructing neuron models of different detail levels. These neuron models are divided into two major classes. The first class are neuron models that include the morphology of neurons, e.g. by splitting up the neuron in multiple compartments and applying cable equations. The second class focuses on the electrophysiological properties of neurons and their behavior, if embedded in neural networks, but do not reproduce the spatial extension of the neurons.

In the following, the basic morphology of a nerve cell is introduced (Figure 1): A nerve cell consists of a central cell body or soma, that is surrounded by a membrane. This membrane separates the soma from the extracellular space and hosts a variety of ion channels that are responsible for the electrophysiological properties of a neuron. Specifically, the dynamics of these ion channels shape the so-called action potentials, which are the signals sent out to the neural network via axons. For further details about the morphology and electrophysiology see Dayan & Abbott (2001).

One of the first neuron models including the theory of ion channels is based on the experiments with the giant squid axon performed by Hodgkin & Huxley (1952) as described in Section 1.1.1. This *Hodgkin-Huxley* (HH) neuron model can reproduce the generation of action potentials by taking into account the measured dynamics of three types of ion channels. As the model involves many differential equations and parameters, as well as being often extended by many other types of neuron channels, it is not suitable for fast numerical evaluations that are needed when simulating large networks (Izhikevich, 2003).

Simpler dynamics, leading to both easier mathematical analyses and faster computation, define the so-called *leaky integrate-and-fire* (LIF) neuron model as described in Section 1.1.2. Compared to the HH neuron model, the LIF neuron model does not realize the action potential anymore, but uses a threshold process instead. Due to its oversimplified dynamics, the LIF neuron model is not capable of reproducing the large variety of neuron responses measured in biology.

The so-called *adaptive exponential* (AdEx) neuron model by Brette & Gerstner (2005)

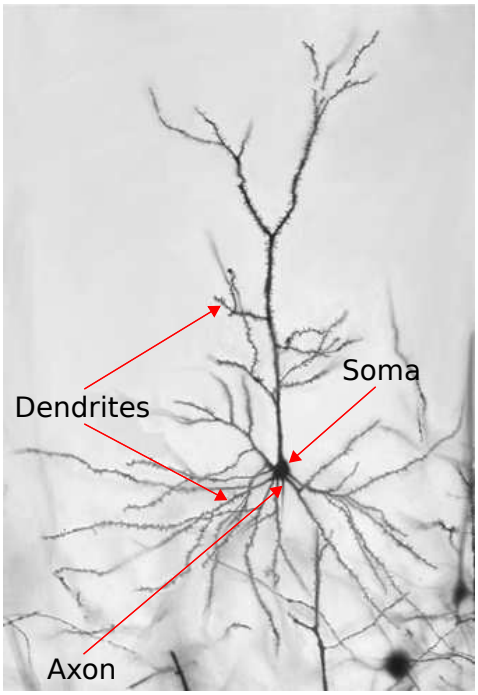


Figure 1: Photograph of a stained cortical neuron from rat. The soma, the dendrites and the axon are indicated. The size of the soma is typically around $10^2\,\mu$m. Courtesy of Terry Robinson.

extends the LIF neuron model by an exponential and an adaptation term, as described in Section 1.1.3. The AdEx neuron model consists of two simple differential equations that are sufficient to reproduce a large variety of firing patterns like spike-frequency adaptation, bursting or chaotic spiking (Naud et al., 2008), and is therefore an ideal candidate for the FACETS wafer-scale hardware system (Section 1.4.2).

1.1.1 The Hodgkin-Huxley Neuron Model

A circuit diagram of the Hodgkin-Huxley neuron model is shown in Figure 2 and is described by the following formulae:

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_L\left(V - E_L\right) - g_K n^4\left(V - E_K\right) - g_{Na} m^3 h\left(V - E_{Na}\right) + I, \tag{1}$$

$$\tau_n(V)\frac{\mathrm{d}n}{\mathrm{d}t} = n_\infty(V) - n. \tag{2}$$

The membrane capacitance C represents the separation between the inside and outside of a nerve cell. Furthermore, the change of the membrane potential V over time is determined by three terms, each for a different kind of ion channel, and an optional current input I. In contrast to the constant leakage conductance g_L, the potassium and sodium conductances g_K and g_{Na} are affected by time- and voltage-dependent so-called *gating equations* for n, m and h. All gating equations are constructed analogously to Equation (2), which is exemplarily written down for the gating parameter n. The gating equation parameters τ_n and n_∞ influence

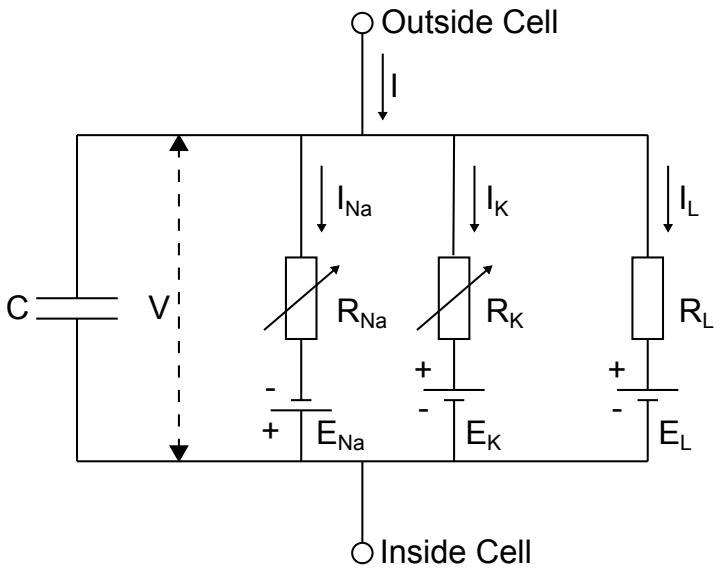


Figure 2: Equivalent circuit diagram of the Hodgkin-Huxley neuron model. The capacitance C represents the cell membrane that isolates the outside from the inside of a cell. The voltage V denotes the resulting membrane potential. The different kinds of ion channels are depicted as time dependent resistances $R = \frac{1}{g}$ that regulate the currents I, which flow according to the difference between the membrane potential V and the reversal potentials E.

the dynamics of n and are, as well as their counterpart for m and h, of no further interest in this study (for further details see Hodgkin & Huxley (1952)). The direction and intensity of the ion channel currents are additionally influenced by their respective reversal potentials E_L, E_K and E_{Na}.

The current input I can be split up into a constant current I_c, and a current caused by incoming signals (Equation (3)). The sum over j describes incoming excitatory spike signals, whereas the sum over k describes the inhibitory spike signals. E_e and E_i are the reversal potentials of the excitatory conductance $g_e(t)$ and the inhibitory conductance $g_i(t)$, respectively:

$$I = I_c - \sum_j g_e(t)(V - E_e) - \sum_k g_i(t)(V - E_i). \tag{3}$$

1.1.2 The Leaky Integrate-and-Fire Neuron Model

Compared to the HH neuron model introduced in Section 1.1.2, the LIF neuron model only considers the constant leakage conductances g_L (Equation (4)). Hence the action potential can not be realized by dynamic conductances anymore, but the membrane potential is set to the reset potential V_r, if a threshold V_{td} is crossed (Equation (5)), and a spike is said to have occurred:

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_L(V - E_L) + I, \tag{4}$$

$$V = V_r, \text{ if } V > V_{td}. \tag{5}$$

1.1.3 The Adaptive Exponential Neuron Model

In order to reproduce a variety of firing patterns, the LIF neuron model of Section 1.1.2 is extended by an adaptation and exponential term. Consequently the temporal evolution of the membrane potential is described by two equations:

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_L \cdot (V - E_L) + g_L \cdot \triangle_{th} \cdot \exp(\frac{V - V_{th}}{\triangle_{th}}) + I - w, \tag{6}$$

$$\tau_w \frac{\mathrm{d}w}{\mathrm{d}t} = a \cdot (V - E_L) - w, \tag{7}$$

and by two reset conditions:

$$V \to V_{reset}, \tag{8}$$

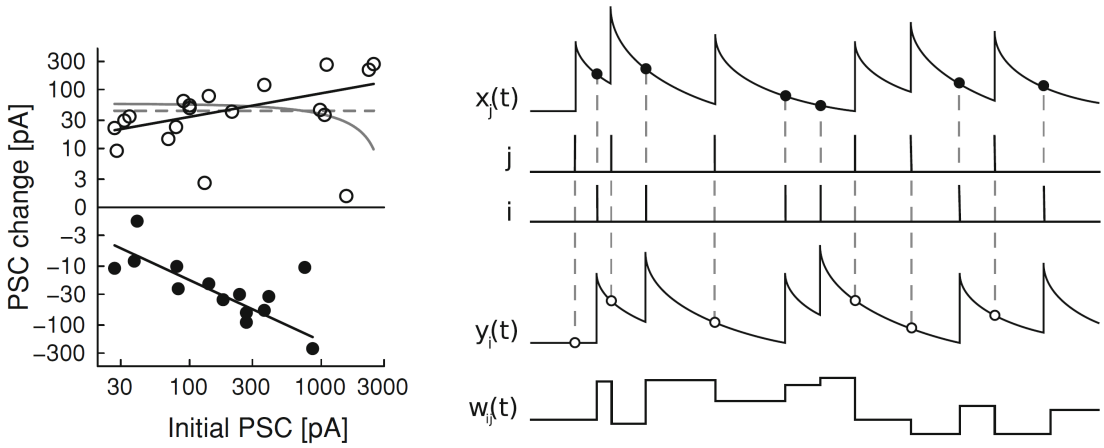
$$w \to w + b. \tag{9}$$

The adaptation feature of the AdEx model is given by the temporal evolution of the adaptation variable w, which is described by Equation (7). Two types of adaptation can be distinguished in the model: *sub-threshold* adaptation described by Equation (7), and *spike-triggered* adaptation, which is described by Equation (9), and is only performed when a spike is emitted. The other main feature of the AdEx model is the exponential term, which can be found in Equation (6). When the membrane voltage reaches the exponential threshold V_{th}, the contribution of this exponential term becomes significant and the membrane potential increases rapidly depending on the slope factor Δ_{th}. Then, when another threshold voltage V_{peak} is reached, a spike is emitted and the reset conditions are applied.

1.2 From Biology to Synapse Models

Synaptic input is received by tree-like extensions of the soma known as dendrites (Figure 1) that receive signals from the axon terminals, called synapses. These synapses are, inter alia, modulated by presynaptic firing rates (Dudek & Bear, 1992), pre- and postsynaptic exact spike timings, postsynaptic membrane potentials (Kelso et al., 1986; Artola et al., 1990) and neuromodulators (Pawlak & Kerr, 2008; Zhang et al., 2009).

In the following, we will focus on the spike-timing dependent plasticity (STDP) only. In contrast to short term plasticity that lasts for several hundred milliseconds, STDP can span hours to months and is therefore a promising candidate for learning processes. One requirement for STDP models is a sensitivity to correlations between pre- and postsynaptic spike times (Morrison et al., 2008). This sensitivity can be realized by simple pair-based STDP models as described in Section 1.2.1. These pair-based STDP models and their hardware implementation (Section 1.4.3) will be one main focus of this study. Nevertheless these models cannot give a full account of STDP, because the dependency of plasticity on the repetition frequency of spike pairs can not be reproduced. Hence this model can be extended by a second spike timing trace towards the so-called *triplet synapse model* by Pfister & Gerstner (2006). One approach to include the dependency on the postsynaptic voltage as well, is modeled by Clopath et al. (2010).



(a) The peak synaptic amplitudes of current-based synapses and their changes before and after performing the Bi & Poo (1998) spike pairing protocol. The weight dependency of an additive (gray dashed), multiplicative (gray solid) and power law (black) STDP model is fitted for the potentiating case.

(b) Time dependency of the two variables $x_j(t)$ and $y_i(t)$ in a pair-based STDP model as well as the resulting weight evolution $w_{ij}(t)$ of a synapse connecting neuron j to neuron i. $x_j(t)$ denotes the history trace for the presynaptic terminal of the synapse, $y_i(t)$ for the postsynaptic one, respectively.

Figure 3: The weight (a) and time (b) dependency of a synaptic weight change as described in Equation (10). Adopted from Morrison et al. (2008).

1.2.1 Pair-Based Spike-Timing Dependent Plasticity

The Hebbian postulate is seen as a fundamental requirement for learning and self-organization within neural networks (Song & Abbott, 2001; Feldman & Brecht, 2005; Dan & Poo, 2006; Markram, 2006). STDP, a particular Hebbian learning process, can be described as a pair-based update rule according to biological measurements, like those of Bi & Poo (1998) and Markram et al. (1997). Those measurements have been interpreted in different ways and led to several STDP models as reviewed by Morrison et al. (2008). Most pair-based STDP models (Song et al., 2000; van Rossum et al., 2000; Gütig et al., 2003; Morrison et al., 2007) separate weight modifications δw into a time-dependent factor $x(t)$ and a weight-dependent factor $F(w)$:

$$\delta w = F(w)x(t). \tag{10}$$

The exponentially decaying time dependency $x(t)$ is common for all those models, with Δt denoting the time between a pair of pre- and postsynaptic spikes:

$$x(t) = \exp(-|\Delta t|/\tau). \tag{11}$$

In accordance with most models, the time constant is assumed to be $\tau = 20\,\text{ms}$ throughout this study. An example for $x(t)$ is shown in Figure 3.

In contrast to the time dependency $x(t)$, the weight dependency $F(w)$ differs between STDP models. Examples for $F_+(w)$ and $F_-(w)$ in case of a causal and acausal spike time correlation, respectively, are given in Table 1 and partly fitted to experimental data (Bi & Poo, 1998; Morrison et al., 2008) in Figure 3.

In case of the FACETS wafer-scale hardware system, $F(w)$ corresponds to a weight update mechanism using a look-up table, and $x(t)$ is implemented by means of local accumulation circuits that measure the spike time correlations (Schemmel et al., 2006, 2007).

In this study, we will focus on the intermediate Gütig STDP model with the parameters $\alpha = 1.05$, $\lambda = 0.005$ and $\mu = 0.4$ (Song et al., 2000; van Rossum et al., 2000; Rubin et al., 2001; Gütig et al., 2003; Morrison et al., 2008), as it combines the additive with the multiplicative STDP model and is furthermore bounded to a limited range of synaptic weights. For details about this choice see Section 2.4.1.

Model name	$F_+(w)$	$F_-(w)$
Additive (equivalent to Song)	λ	$-\lambda\alpha$
Multiplicative	$\lambda(1-w)$	$-\lambda\alpha w$
Gütig	$\lambda(1-w)^\mu$	$-\lambda\alpha w^\mu$
Van Rossum	c_p	$-c_d w$
Power law	λw^μ	$-\lambda\alpha w$

Table 1: Weight-dependency of common pair-based STDP models. F_+ denotes the dependency in case of a causal correlation with $\Delta t > 0$ and F_- in the acausal case, where $\Delta t < 0$. w is the current weight of the synapse. λ, μ, α and c are positive parameters.

1.3 Software Simulations

Software simulations play an important role throughout this study. On the one hand, they are a valuable tool for preparative analyses, as carried out in Section 2, where software models of a hardware synapse are implemented and applied, e.g. in neural networks. On the other hand, software simulations serve as a reference for measurements with existing hardware devices. In Section 3 hardware measurements of neurons are compared and optimized to reproduce the results of a reference software simulation. For all software simulations in this study, the software simulator NEST (Gewaltig & Diesmann, 2007) is used with its standard parameters[4], if not specified further.

1.4 The FACETS Hardware Systems

Within the FACETS project, two mixed-signal VLSI[5] hardware systems to emulate spiking neural networks have been developed by the Electronic Vision(s) group in collaboration with the technical university of Dresden. The historically older FACETS chip-based system as described in Section 1.4.1 combines a high acceleration with low power consumption and variable neural network configuration. However, this system is limited to one single chip, or to several sparsely connected chips on one backplane (Grübl, 2003; Jeltsch, 2010). The later, parallel developed FACETS wafer-scale hardware system is introduced in Section 1.4.2, and represents an alternative approach by several single chips that are interconnected directly on the silicon wafer. This ensures a high communication bandwidth between these chips and therefore allows a significantly higher number of neurons and synapses interconnected on a wafer.

[4] as released in NEST version 1.9.8718

[5] Very-Large-Scale Integration

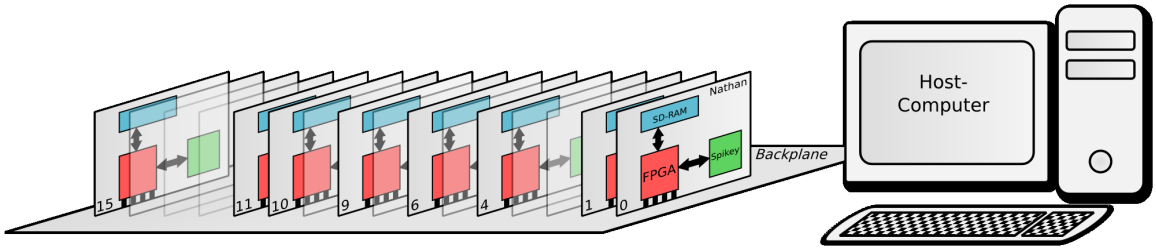


Figure 4: Layout of the FACETS chip-based hardware system. The Spikey chip can be configured by a host computer via a Gigabit Ethernet connection to the backplane, the host board (Nathan) of the Spikey chip and the backplane this host board is plugged in. With the permission of Sebastian Jeltsch.

Both FACETS hardware systems implement highly configurable synapses that occupy the major part of the chip area, whereas the area occupied by neurons is comparably small, as are depicted in Figure 5 and 6. This is in contrary to other neuromorphic hardware systems that focus on the implementation of neurons, e.g. the one developed by Daouzli et al. (2008), which computes plasticity off-chip. This works well for real-time neuron implementations, but fails for highly accelerated hardware systems, because of the limited bandwidth between the hardware device and its host computer.

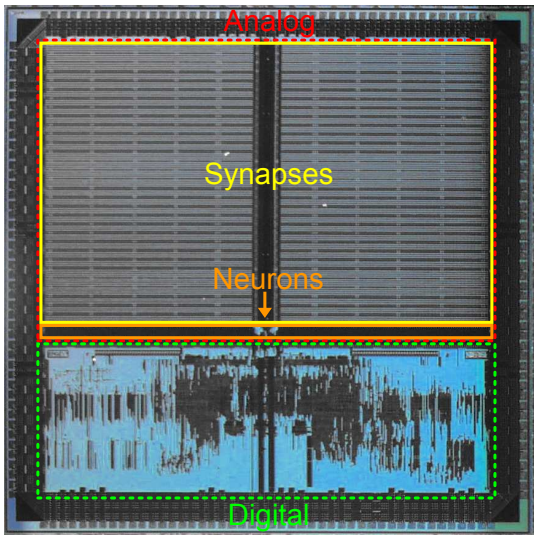
1.4.1 The FACETS Chip-Based Hardware System

The FACETS chip-based hardware system, as described in Schemmel et al. (2006); Grübl (2007), serves currently, with the 4th version of its chip, as a reliable neuromorphic emulation platform. Up to 16 host boards, called *Nathans*, are plugged into a backplane, where each Nathan is hosting a single mixed-signal ASIC[6], called *Spikey*. Each of these Spikey chips can be accessed by a host computer via its Nathan and the backplane (Figure 4). All relevant parameters of the hardware can be configured by using the simulator-independent network description language PyNN (?Brüderle et al., 2009). This enables an easy setup, modification and analysis of hardware-based neural network experiments without the need of having a detailed knowledge about the hardware system.

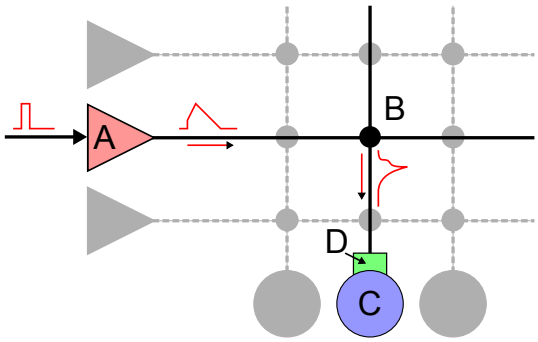
In the following, technical details that are relevant in the context of this thesis are described. Each Spikey chip occupies an area of $5 \times 5\,\mathrm{mm}^2$ and is fabricated in a 180 nm CMOS[7] process. Its basic components are 384 neurons with 256 programmable synapses each, as well as a control circuitry (Figure 5a). Because of their large number, the synapses occupy the major part of the chip resources. The membrane potential V of each neuron is emulated by an analog circuit that mimics the differential equations of the LIF neuron model with conductance-based synapses, which is described in Section 1.1.2. These neuron circuits require a comparably small area and are located below the array of synapses.

The operational transconductance amplifiers that realize the controllable excitatory and inhibitory conductances are located at the input of each neuron circuit. Figure 5b illustrates, how a digital signal is transformed into a time-dependent conductance: The synapse drivers

[6] Application-Specific Integrated Circuit
[7] Complementary Metal-Oxide-Semiconductor



(a) A photograph of the Spikey chip showing the areas of analog neuron (orange) and synapse (yellow) circuits as well as the digital communication circuitry (dashed green).



(b) Synaptic signals in the Spikey chip: The incoming digital pulse representing the output spike sent by a neuron is transformed into a voltage ramp within the synapse driver (A), which is again transformed by the synapse node (B) into a current pulse that finally regulates the synaptic conductance of the target neuron (C) via an operational transconductance amplifier (D).

Figure 5: The Spikey chip (a) as well as a schematic of the signal pathway of the synaptic input as implemented in this chip-based hardware system (b).

between both arrays of synapse nodes are fed with digital event signals and are generating voltage ramps at their arrival. When these ramps arrive at a synapse node, they are transformed into a current ramp with exponentially rising and falling edges with an amplitude according to the weight of this synapse. After traveling down to the neuron circuits, the current course controls the conductance of the neuron input in a linear fashion. The configurable ramp shape of synaptic conductance courses allows a variety of time-dependencies, including exponentially decays or alpha-function behavior, but can only be set row-wise. Throughout this study all experiments are using established ramp shapes that are discussed in detail by Brüderle (2009).

In addition to this, short- and long-term plasticity can be activated. The implementation of the short-term plasticity will not be further discussed, but can looked up in Brüderle (2009) and Bill et al. (2010). The long-term plasticity is implemented as STDP described in Section 1.4.3.

1.4.2 The FACETS Wafer-Scale Hardware System

In order to model large-scale neural networks using a neuromorphic hardware device, the concept of wafer-scale integration in neuromorphic hardware systems was introduced during the FACETS project (Schemmel et al., 2008, 2010). The FACETS wafer-scale hardware system will feature hundreds of ASICs interconnected directly on the wafer. A single wafer will contain up to 180 thousands of neurons and about 40 millions of synapses, working at

an acceleration factor of $10^3 - 10^5$ compared to biological real time, thus allowing on-the-fly exploration of large-scale network topologies.

Compared to the traditional approach of packaging each ASIC separately, the wafer-scale integration provides the necessary communication bandwidth to transport the highly accelerated rate of neural events between individual ASICs. Each of these ASICs, also called HICANNs, contains 512 silicon neurons and about 114.000 synapses (Figure 6).

The neuron implementation used in the FACETS wafer-scale system emulates the AdEx neuron model (Millner et al., 2010). The detailed equations describing the model are given in Section 1.1.3, whereas details about the hardware implementation can be found in Section 3.3.1. Compared to the LIF neuron model (Section 1.1.2) implemented in the chip-based hardware system (Section 1.4.1), the AdEx model adds two important features: adaptation and exponential rise.

Each neuron parameter can be configured individually (except the reset voltage, which is set globally) via 23 analog parameters that are stored in floating-gate analog memory cells (Srowig, 2007). Compared to capacitance-based solution in the Spikey chip, analog floating-gate memory cells have many advantages: they are non-volatile, power-efficient and space-saving. Therefore, the configuration and calibration of individual neurons becomes possible. Switches placed between the floating gate circuits and the neuron circuits allow to change the order of magnitude of the different time constants incorporated into the neuron circuits. Thus, it is possible to change the acceleration factor of the neuron, going from 10^3 to 10^5 compared to biological real time. An acceleration factor of 10^4 is used throughout this study.

Although the analog part of the HICANN synapse circuits are almost identical to those of the chip-based system, the signal shape is different (Schemmel et al., 2011). As the STDP mechanism of the hardware synapses in the HICANN chip was not available at the time of this study, there are no synapse measurements using the HICANN chip.

The neuron can be stimulated in two different ways: by the incoming events from the synapse array, or by a programmable current. The latter can be used to apply biological currents up to some μA to the neuron following a loop of 129 individual 10-bit values. The frequency of these values can be set from $f_l = 3.9\,\text{MHz}$ to $62.5\,\text{MHz}$ in hardware time.

The neuron membrane voltage can be accessed via one of the two analog outputs of the HICANN chip, and read out with a standard oscilloscope.

Because the FACETS wafer-scale system was still in development and production during this study, a first HICANN prototype chip was assembled in a test setup, which is described in Section 3.3.3.

1.4.3 Spike-Timing Dependent Plasticity in Hardware

The spike-timing dependent plasticity of synapses is implemented in the chip-based FACETS hardware system (Section 1.4.1) as follows: Besides measurement and accumulation circuits within the synapse node that record the correlations between the pre- and postsynaptic spikes, a global update controller periodically reads out these accumulated correlations and processes the weights according to pre-programmed look-up tables (LUTs). Details about the design concept of a hardware STDP synapse are given in Section 2.1, and technical details about the accumulation circuits are described in Schemmel et al. (2006).

Nevertheless, a list of hardware parameters is given in Table 2. These must be adjusted appropriately to perform hardware experiments involving STDP. The difference between V_{cthigh}

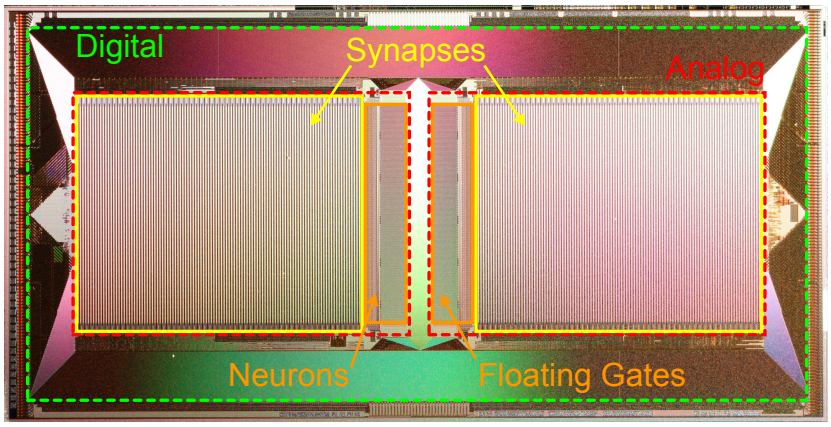


Figure 6: Photograph of the HICANN chip. The neurons, floating gates (orange) and synapses (yellow) are all analog circuits (dashed red), whereas the remaining area is occupied by digital circuits, which are mostly used for communication purposes (green). Courtesy of Marc-Olivier Schwartz.

and V_{ctlow} determines the threshold V_t, which must be crossed by the absolute value of the difference between the accumulated correlations a_c and a_a of the causal and acausal circuit, respectively. In case of such a threshold crossing a correlation flag is set and a weight update is elicited, when the update controller processes this synapse the next time:

$$|a_c - a_a| > V_t. \tag{12}$$

Whether a causal or acausal weight update is performed depends on another correlation flag that checks the following equation:

$$a_c - a_a > 0. \tag{13}$$

In case of the FACETS wafer-scale system (Section 1.4.2) the accumulation circuitry is conceptually the same as in the chip-based system, but the threshold conditions are varied slightly. Instead of the absolute value of the difference between a_c and a_a (Equation (12)), each accumulated correlation is compared to a separate threshold:

$$a_c > V_{tc}, \tag{14}$$

$$a_a > V_{ta}. \tag{15}$$

In the chip-based as well as in the wafer-scale hardware systems both the causal and acausal accumulated correlations are reset, if a weight update is performed. The consequences of such a common reset are analyzed in Section 2.4.3. To circumvent the partly negative effects of a common reset in the wafer-scale system, two rows of synapses can be combined in order to obtain an independent reset, but obviously the available number of synapses will be reduced. This is not possible for the chip-based system.

In Section 2 the STDP implementation of the FACETS wafer-scale system will be analyzed, starting with an even more generalized synapse model. But all experiments in

Parameter	Description	Value
V_{clrc}	Amount of charge that will be accumulated on the capacitor C_1 (see Schemmel et al., 2006) in case of causal spike time correlations	0.90 V
V_{clra}	See Vclrc, but for the acausal circuit	0.94 V
V_{ctlow}	Lower accumulation threshold	0.85 V
V_{cthigh}	Higher accumulation threshold	1.0 V
$adjdel$	Adjustable delay between the pre- and postsynaptic spike	2.5 μA
V_m	Parameter to stretch the STDP time constant τ (see Section 1.2.1)	0.0 V
$I_{bcorreadb}$	Bias current that influences timing issues during read outs of the correlation flags	2.0 μA

Table 2: Descriptions of the hardware parameters of the STDP synapses on the Spikey chip and the values applied during the presented experiments performed with chip number 441. The threshold V_t is defined by the difference $V_{cthigh} - V_{ctlow}$. V_{clra} and V_{clrc} were adjusted to balance the causal and acausal accumulation circuits. For further technical details of these circuits, see Schemmel et al. (2006).

Section 2.4.4 have been carried out with the chip-based system, because the wafer-scale system was not available during this study. Qualitatively, this should not alter the results, because their analog parts of the synapse circuits are implemented almost identically.

2 Long-Term Plasticity in Hardware Synapses

This section investigates the impact of design-specific restrictions of the synapse implementation in the current FACETS wafer-scale hardware system. Limitations caused by the trade-off between the number and size of synapses as described in Section 2.1 are analyzed by preparative software simulations. The question of how many bits are enough to encode synaptic weights as part of functional and plastic neural networks is in the focus of first analyses that are presented in Section 2.4.2. Furthermore, the pre- and postsynaptic activity as well as further hardware restrictions are considered in simulations of single synapses and a neural network benchmark as described in Section 2.4.3. Finally, the production variances of synapse circuits are measured in a real hardware system and are again incorporated into a synapse model in order to study their effects on the benchmark, as shown in Section 2.4.4.

First, a short review of the so-called spike-timing dependent plasticity (STDP) of synapses is given and the corresponding implementation concept for the FACETS hardware systems is introduced. Second, discretized synaptic weights, as used in the FACETS hardware systems, are defined and incorporated in look-up tables (LUTs), the structure of which will then be analyzed. In addition to studies on long-term dynamics of discretized weights, this discretization is integrated into a software version of the synapse model, which enables further NEST-based (Gewaltig & Diesmann, 2007) studies on single synapses and neural networks.

This part of the thesis at hand resulted from a collaboration with Tobias C. Potjans (Potjans, 2011) and is published Pfeil et al. (2012).

2.1 The Concept of the FACETS Hardware Synapse

The reduction of the synapse size in silico will be demonstrated on the basis of the FACETS wafer-scale hardware system. The two main building blocks of such a hardware synapse are a storage for its weight value and a mechanism to update the stored weight according to the correlations between pre- and postsynaptic spikes. In order to achieve one million synapses in one chip (Schemmel et al., 2008), the *accumulation* of correlations and the *weight update controller*, which is resource-intensive due to its large and programmable digital circuits, have to be separated. One approach is to accumulate causal and acausal relationships between pre- and postsynaptic spikes in situ, but to update the synaptic weights by an external mechanism instead of controlling and performing this locally in each synapse. As a consequence, the layout of a single synapse is reduced to an analog accumulation circuit and a digital component that stores the weight (Figure 7).

The accumulation is chosen to be implemented as an analog circuit to reduce its size compared to digital implementations. Since the digital weight storage is likely to exceed the analog accumulation circuit in terms of chip resources, the resolution of weight values has to be kept as small as possible by limiting the number of bits representing them. A further reduction of the analog accumulation circuit is achieved by using the reduced symmetric nearest-neighbor spike pairing scheme (Morrison et al., 2008). Instead of considering all past and future spikes, only the last and the following spike at both terminals of the synapse are taken into account, which does not affect the biological relevance (Burkitt et al., 2004). Last but not least, resetting both the causal and acausal accumulation circuits with a single reset line after processing them can reduce the occupied resources even further.

To achieve a general-purpose modeling environment, the configuration of possible connections between neurons should be highly flexible. Later changes in the layout of synapses are

time-consuming and involve expensive re-manufacturing of the chips. Update rules can be kept flexible by programming them into a global weight update controller responsible for updating all the synapses. Such a mixed-signal layout with analog synapses plus digital weights and weight update mechanisms is beneficial for a variety of STDP models.

Nevertheless, because many synapses are sharing one weight update controller, the frequency of weight updates of a single synapse is limited. This causes the need for an accumulation of sequent spike correlations at a single synapse and therefore has to be taken into account during the update process. Due to limited weight update controller resources, extended arithmetic operations are not realizable and are replaced by programmable LUTs operating on discrete weights. Such a LUT lists for each discretized weight the resulting weights in case of causal or acausal correlations between the pre- and postsynaptic spikes. LUTs do not limit the flexibility of update processes as all kind of STDP models can be used to configure them.

On the contrary to ideal arithmetic operations in software models, the analog accumulation circuits vary due to the manufacturing process. This variance is a possible performance limitation on the lowest level and determines the necessary level of detail for all other limitations.

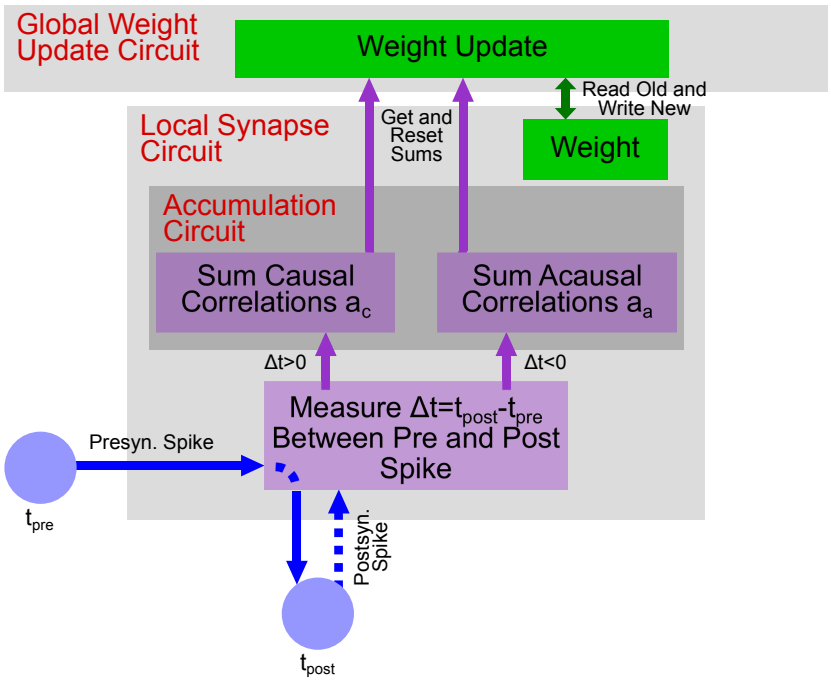


Figure 7: Schematic drawing of a simple hardware synapse representing resource-efficient solutions like the Spikey synapses. Analog circuits are drawn in purple and digital circuits in green. Although only one synapse is shown, the global weight update controller is responsible for reading out and updating many hardware synapses. Spike time correlations between a pre- and postsynaptic neuron are accumulated until they are read out by the global weight update controller, which modifies the digital weight of this hardware synapse.

From Biology to Hardware In this paragraph, an alternative approach towards implementing STDP in the FACETS wafer-scale hardware system is presented. Instead of trans-

Threshold crossing	causal V_t^0	causal V_t^1	causal V_t^2	causal V_t^3
acausal V_t^0	-	-	P	LP
acausal V_t^1	D	D	P	LP
acausal V_t^2	D	D	P	LP
acausal V_t^3	LP	LP	LP	LP

Table 3: Weight update table of a new synapse model that is inspired from biology and is suitable for the current FACETS hardware systems: The weight update of a synapse is dependent on the combination of the highest crossed causal and acausal thresholds V_t ($V_t^0 < V_t^1 < V_t^2 < V_t^3$) and is defined as follows: D denotes a depressing update, P a potentiating and LP a large potentiating update. For more details please see text.

ferring existing STDP models, as described in Section 1.2.1, to the hardware system, a new model is introduced that is appropriate for the current hardware system with its common reset mechanism:

The study of Sjöström et al. (2001) indicates three ranges of spike pair ($\Delta t = 10\,\text{ms}$) frequencies, 0.1 Hz to 10 Hz, 10 Hz to 30 Hz and $> 30\,\text{Hz}$, that will be represented by three individual correlation accumulation thresholds V_t in this new possible hardware synapse model. These thresholds have the same ratio as their frequency counterparts, $\frac{V_t^2}{V_t^1} = \frac{10\,\text{Hz}}{0.1\,\text{Hz}} = 100$ and $\frac{V_t^3}{V_t^2} = \frac{30\,\text{Hz}}{10\,\text{Hz}} = 3$, whereas $V_t^0 = 0$. The behavior for all possible combinations of threshold crossings is listed in Table 3. The according weight change Δw is adopted from Sjöström et al. (2001): 70% of the weight w for the depression case (D), 115% for the potentiation case (P) and 150% for the large potentiation case (LP). Although this biologically motivated model is not influenced by the limitation of a common reset, as the crossing of the thresholds is checked in equidistant time intervals, this time interval itself is dependent on the actual mean firing rate. This model is an interesting approach, but performs only well (not shown here), if the thresholds V_t are adjusted to the mean firing rate, and is therefore not suitable for general purpose network architectures.

2.2 Methods for Configuration and Distributions

Before entire synapses are analyzed in Section 2.3, we focus on the discretization of synaptic weights and weight distributions in this section.

2.2.1 Discretization of Synaptic Weights

A bidirectional transformation between continuous weight values, as assumed for the STDP models described in Section 1.2.1, and discretized weight values is introduced. The range of possible weight values $I = [0, w_{max}]$ is divided into 2^b bins, where b is the number of bits representing a discretized weight value. To avoid asymmetric bins, the two boundary bins of I are half as wide as the others in between. Equation (16) and (17) describe the transformation between continuous weight values w_c and the corresponding discretized weight values w_d:

$$w_d = \text{round}\left(\frac{w_c}{c}\right) \qquad \text{for} \quad w_c \in I, \tag{16}$$

$$w_c^{'} = c w_d. \tag{17}$$

where $c = w_{max}/(2^b - 1)$ denotes the width of a bin. In Equation (16) the tie-breaking rule for rounding is round half to even.

2.2.2 Look-Up Table Configuration

Once discretized weights have been obtained, LUTs can be configured to determine weight changes caused by spike-timings. According to experimental setups for biological measurements (Markram et al., 1997; Bi & Poo, 1998; Sjöström et al., 2001) a typical interval between a pre- and postsynaptic spike is assumed to be $\Delta t_s = 10\,\text{ms}$ and is called a *standard spike pair* (SSP).

Because several spike correlations can be required to reach the next discrete weight value, LUTs are configured by performing reference simulations in continuous weight space. Single STDP weight updates δw as described in Section 1.2.1 are applied n times in a row resulting in a *combined* weight update Δw. In the following a *weight update* refers to this combined weight update Δw, if not stated otherwise. In the context of weight discretization, n is a free parameter that determines the number of locally accumulated correlations before a globally triggered weight update is performed.

The causal and acausal columns of a LUT are obtained by applying Equation (10) with common $x(\Delta t_s)$, but different $F_+(w)$ or $F_-(w)$, respectively, as denoted in Table 1. In order to obtain a complete LUT the continuous representation for each discrete LUT entry according to Equation (17) is taken as the initial weight value. After performing all n sequent weight updates δw on this initial weight value, the resulting continuous weight value is transformed back to its discretized representation according to Equation (16). Example LUTs with different numbers of SSPs are given in Table 4.

Although we are focusing on the Gütig STDP model, the updated weight values can under- or over-run the allowed limit values due to finite weight change steps. If this happens, the weight will be clipped to its maximum or minimum value, respectively.

w	w_+	w_-
0	2	0
1	3	0
2	4	1
3	5	2
4	5	2
5	6	3
6	7	4
7	7	5

(a)

w	w_+	w_-
0	4	0
1	5	0
2	6	0
3	6	0
4	7	1
5	7	1
6	7	2
7	7	2

(b)

w	w_+	w_-
0	1	0
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	5
7	7	6

(c)

Table 4: Example LUTs with a 3-bit weight resolution and 100 (a), 250 (b) and 25 (c) SSPs (standard spike pairs): For each LUT, w denotes the initial entry and w_+ as well as w_- are the resulting weight entries in case of a causal or acausal weight update, respectively.

2.2.3 Equilibrium Weight Distributions

In order to analyze the effects of weight discretization on long-term weight dynamics, the equilibrium weight distributions in discretized weight space are determined by assuming an

equiprobable causal and acausal weight evolution. Then these distributions are compared to reference equilibrium distributions in continuous weight space via the mean squared error (MSE). A precalculated LUT as described in Section 2.2.2 is used to perform those sequent weight updates (for details see Section 2.2.3). Two types of reference distributions were taken into account, one determined with an analytical and one with a numerical approach. Discrete weight values are represented by b bits. Results are shown in Section 2.4.2.

Analytical Distribution The time derivative of the analytical distribution $P(w)$ can be described by the master equation adopted from van Rossum et al. (2000):

$$\frac{\partial P(w,t)}{\partial t} = -p_d P(w,t) - p_p P(w,t) + p_d P(w + \Delta w_d, t) + p_p P(w - \Delta w_p, t), \tag{18}$$

where p_d and p_p are the probabilities for depression and potentiation, respectively. A weight step Δw is defined by a sequence of n single weight updates δw. Hence the resulting combined weight steps are calculated by $\Delta w_d(w) = (w + F_-(w))^{(n)} - w$ and $\Delta w_p(w) = (w + F_+(w))^{(n)} - w$, where $f(w)^{(n)}$ is the n-fold recursive evaluation of $f(w)$.

By Taylor expanding $P(w + \Delta w_d, t)$ and $P(w - \Delta w_p, t)$ for small step sizes (Van Kampen, 2007) into

$$P(w + \Delta w_d, t) = P(w,t) + \Delta w_d \frac{\partial}{\partial w} P(w,t) + \frac{1}{2} \Delta w_d^2 \frac{\partial^2}{\partial w^2} P(w,t) + \mathcal{O}(\Delta w_d^3) \qquad \text{and}$$

$$P(w - \Delta w_p, t) = P(w,t) - \Delta w_p \frac{\partial}{\partial w} P(w,t) + \frac{1}{2} \Delta w_p^2 \frac{\partial^2}{\partial w^2} P(w,t) + \mathcal{O}(\Delta w_p^3),$$

we obtain the Fokker-Planck Equation (19) with jump moments $A(w) = p_d \Delta w_d(w) + p_p \Delta w_p(w)$ and $B(w) = p_d \Delta w_d(w)^2 + p_p \Delta w_p(w)^2$ adopted from van Rossum et al. (2000):

$$\begin{aligned} \frac{\partial P(w,t)}{\partial t} &= p_d \left[\Delta w_d \frac{\partial}{\partial w} P(w,t) + \frac{1}{2} \Delta w_d^2 \frac{\partial^2}{\partial w^2} P(w,t) \right] \\ &\quad + p_p \left[-\Delta w_p \frac{\partial}{\partial w} P(w,t) + \frac{1}{2} \Delta w_p^2 \frac{\partial^2}{\partial w^2} P(w,t) \right] \\ &= -\frac{\partial}{\partial w} \left[A(w) P(w,t) \right] + \frac{1}{2} \frac{\partial^2}{\partial w^2} \left[B(w) P(w,t) \right]. \end{aligned} \tag{19}$$

The stationary state of this Fokker-Planck equation can be written in terms of a current $J(w)$:

$$\frac{\partial P(w,t)}{\partial t} = \frac{\partial J(w)}{\partial w} = 0 \tag{20}$$

with

$$J(w) = -A(w) P(w) + \frac{1}{2} \frac{\partial}{\partial w} \left[B(w) P(w) \right]. \tag{21}$$

If reflective boundaries are assumed,

$$J(w) = 0 \tag{22}$$

is valid at the boundaries and consequently over the entire interval I. The application of this boundary condition in Equation (21) has the following solution:

$$P(w) = \frac{N}{B(w)} \exp \left[2 \int_0^w \frac{A(w')}{B(w')} dw' \right], \tag{23}$$

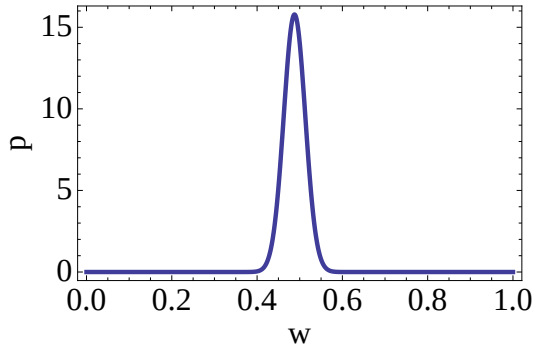


Figure 8: The equilibrium distribution of the probability p for weight values w, which is analytically solved for the multiplicative STDP model assuming one SSP ($n = 1$) and an equal probability of potentiating and depressing weight changes.

where N is a normalization constant, such that $\int_0^1 P(w)dw = 1$ (Gardiner, 2009).

In case of the multiplicative STDP model (see Table 1), the weight step sizes Δw_p and Δw_d that are dependent on n, can be simplified as follows, where w_{i+n} denotes n single weight updates $\delta w = w_{i+1} - w_i$:

$$\begin{aligned} w_{i+1} &= w_i + \lambda(1 - w_i) \\ \Rightarrow w_{i+n} &= w_i(1-\lambda)^n + \lambda\left[\sum_{l=0}^{n-1}(1-\lambda)^l\right] = (w_i - 1)(1-\lambda)^n + 1 \\ \Rightarrow \Delta w_p(w_i) &= w_{i+n} - w_i = (w_i - 1)(1-\lambda)^n + 1 - w_i, \end{aligned} \tag{24}$$

$$\begin{aligned} w_{i+1} &= w_i - \alpha\lambda w_i \\ \Rightarrow w_{i+n} &= w_i(1-\alpha\lambda)^n \\ \Rightarrow \Delta w_d(w_i) &= w_{i+n} - w_i = w_i(1-\alpha\lambda)^n. \end{aligned} \tag{25}$$

The combination of Equation (23) with the weight step sizes above and a uniform probability between causal and acausal weight updates $p_p = p_d = \frac{1}{2}$ can be solved with the computer algebra system Mathematica (Wolfram Research, Inc., 2008). Giving one example, the solution for $n = 1$ is

$$\frac{P(w)}{N} = \exp\left[\frac{2(-1+\alpha)\arctan\left[\frac{-1+w+\alpha^2 w}{\alpha}\right]}{(1+\alpha^2)\lambda}\right] \cdot \left(\lambda^2\left(1 - 2w + \left(1+\alpha^2\right)w^2\right)\right)^{-\frac{1+\alpha+\lambda+\alpha^2\lambda}{\lambda+\alpha^2\lambda}},$$

which is plotted in Figure 8.

However, such an analytical solution is not possible for the intermediate Gütig STDP model, but Equation (23) can be integrated numerically.

Numerical Distribution The numerical distribution is calculated as follows: N uniformly distributed initial weights $\vec{x}$ are generated within $[0, 1)$. Each of those weight values x_i is evaluated by combined weight updates until the resulting distribution of $\vec{x}$ is in an equilibrium state. Whether a weight value x_i is updated either causal or acausal, is determined by drawing from a uniform distribution. Convergence is defined by means of a minimal distance in the

Euclidean norm $\Delta x = ||\vec{x}' - \vec{x}|| = \sqrt{\sum_i |x_i' - x_i|^2}$ of the distributions, covering 2^b bins, between sequent iterations $\vec{x}$ and $\vec{x}'$.

Discretized Numerical Distribution In the following we restrict synaptic weights to discrete values. The procedure for continuous weights mentioned in the paragraph above can also be applied for discretized weights, and each either causal or acausal weight update can be obtained from the previously configured LUT (as described in Section 2.2.2).

2.3 Methods for Temporal Evolution

After analyzing the effects of discretized weights and their distributions in Section 2.2, we focus on synapses in spiking neural networks in this section.

2.3.1 Implementation of Hardware Restrictions

The hardware constraints considered in this study are implemented as a customized synapse model within the framework of the software simulator NEST (Gewaltig & Diesmann, 2007). Besides the following studies on small neural networks, these synapses can be utilized for simulator-based studies on large-scale neural networks as well.

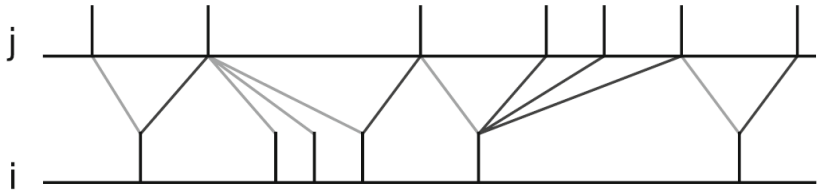
As the LUT within the synapse is configured with n sequent SSPs, the threshold for eliciting updates is set accordingly to $V_t = n \cdot a_{SSP}$. Thereby a_{SSP} is the accumulated correlation within the hardware synapse (as described in Section 2.1), if one SSP is applied. If the threshold V_t is crossed by the so far accumulated either causal or acausal correlations a, the synapse is tagged. When the weight update controller arrives the next time and the synapse is tagged, the discrete synaptic weight is updated by the LUT. Afterwards the branch of accumulation that crossed the threshold V_t is reset to zero (Figure 14). If the synapse is not tagged, when the weight update controller arrives, the correlations are accumulated further without performing any update. In case both accumulations of correlations have crossed the threshold, both are reset without updating the synaptic weight.

As a reference, a NEST synapse model according to (Gütig et al., 2003) was implemented with continuous weights, but with a reduced symmetric nearest-neighbor spike pairing scheme instead of a symmetric all-to-all spike pairing scheme as described in (Morrison et al., 2008) and shown in Figure 9. Comparisons to this model avoid the effects caused by different spike pairing schemes and allow us to analyze other hardware constraints in an isolated way.

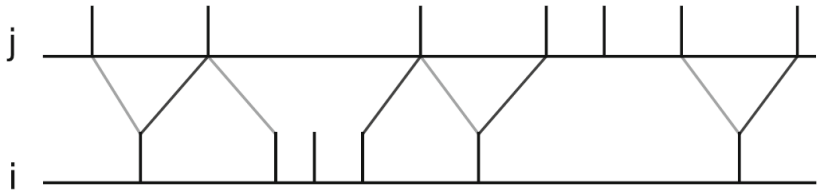
All simulations involving synapses are simulated with NEST using standard parameters[8], if not further specified in the experiments. Spike trains are applied to *parrot neurons* that simply repeat their input, so that the pre- and postsynaptic spikes of a synapse between two of these neurons can be fully controlled.

In the following, the dependency between the frequency of weight updates and the number of SSPs is outlined. A LUT is designed for a specific number of SSPs. Hence the synaptic weight should be updated as soon as one corresponding threshold V_t is crossed. This frequency of weight updates is highly dependent on the firing rate and spike times of the pre- and postsynaptic neuron. High firing rates or high correlations of the spike times cause more frequent weight updates. To avoid these dependencies on firing rates and spike time correlations, we describe the frequency of weight updates in terms of accumulated SSPs until the

[8] as released in NEST version 1.9.8718



(a) Symmetric all-to-all spike pairing scheme.



(b) Reduced symmetric nearest-neighbor spike pairing scheme

Figure 9: Different spike pairing schemes for pair-based STDP models adopted from Morrison et al. (2007). The spike trains of the pre- (j) and postsynaptic (i) synapse terminal are shown. The symmetric all-to-all spike pairing scheme (a) keeps history of all spikes between two spikes at the opposite synapse terminal. In contrast, the reduced symmetric nearest-neighbor spike pairing scheme (b) keeps only a history of the last pre- and postsynaptic spike.

next update is performed, which is identical to the number of SSPs the LUT and threshold are configured with.

2.3.2 Single Synapse Evolution

As a next step, the LUT analysis of Section 2.2.3 is extended to a setup, where the LUT is incorporated into a synapse (Section 2.3.1) that receives pre- and postsynaptic input. In this setup two neurons are connected by a single synapse (Figure 15). Its weight trace is recorded and compared to synapses with continuous weights.

For this purpose, spike trains of the pre- and postsynaptic neurons are correlated by drawing them from a multiple interaction process (MIP) (Kuhn et al., 2003) and shifted by Δt_s to obtain SSPs for correlated spikes. Despite the limitation of the frequency of weight updates that has to be considered later (Section 2.3.3), in this first scenario the threshold crossing of accumulated correlations is checked with a frequency of $f = 10\,\mathrm{kHz}$ to focus on the effects of discretized weights only. This frequency is in accordance with the simulation time step, and prevents the accumulated correlation from crossing the threshold V_t too far, without eliciting an update. Because the character of the weight traces does not change, if the input spike rates applied to each neuron are altered within biological plausible ranges, we assume a stimulation spike rate of $\rho = 10\,\mathrm{Hz}$.

During simulation runs, the synaptic weight is recorded every $\Delta t_r = 3\,\mathrm{s}$ over $T = 150\,\mathrm{s}$ for 30 different random generator seeds. Afterwards the mean weight trace over all simulation runs is compared with that of the continuous STDP model in terms of the MSE. For results see Section 2.4.3.

2.3.3 Network Benchmark

Two populations of 10 neurons each are connected convergingly to an integrate-and-fire neuron with exponentially decaying synaptic conductances[9] (Figure 16A) by either discretized or continuous reference synapses as described in Section 2.3.1. These synapses are excitatory with a time constant of $\tau_{ex} = 0.2\,\text{ms}$ and their initial weights are drawn randomly from a uniform distribution over $[0, 1)$. One population draws its spikes from a MIP with correlation coefficient c (Kuhn et al., 2003), the other from a Poisson process (MIP with $c \to 0$). The common input rate for all presynaptic neurons is adjusted such that the postsynaptic neuron fires with $f \approx 10\,\text{Hz}$ for $c = 0.025$, if using the reference synapses. The synaptic weights are recorded for $T = 2000\,\text{s}$ with a sampling frequency of $f_s = 0.1\,\text{Hz}$.

By applying the Student's t-test (Press et al., 2007) to the weight distribution at the end of the simulation, one obtains the probability p that both populations are not separated. The dependency of p on c is shown in Section 2.4.3.

Further synapse constraints Not only the discretization of synaptic weights, but also the frequency of weight updates and the reset behavior are constraints of the FACETS hardware systems that have to be analyzed.

For studies on the frequency of weight updates, the software model is extended with a time grid that specifies, at which points in time weight updates are allowed.

A common reset means that both the causal and acausal accumulated correlations are reset, if a weight update is performed.

As a basis for a possible compensation mechanism for the common reset, an analog-to-digital converter (ADC) with a 4-bit resolution is introduced to read out the accumulated correlations. Such an ADC requires only a small area in the global update controller and therefore does not increase the resources significantly. An ADC allows to compare the accumulated correlations against multiple thresholds. Implementations of the common reset as well as the ADC are added to the existing software model. For multiple thresholds, the same number of LUTs is needed that have to be chosen carefully (for details and results see Section 2.4.3). To provide symmetry within the order of the sequent causal and acausal weight update, the accumulation branch, of which the higher threshold is crossed, is applied first.

Peri-Stimulus-Time-Histograms The difference between static and STDP synapses on eliciting postsynaptic spikes can be analyzed with peri-stimulus-time-histograms (PSTHs). Therefore spike times are recorded within the last third of an elongated simulation of $T = 3000\,\text{s}$ with $c = 0.025$. During the last $1000\,\text{s}$ the mean weights are already in their equilibrium state, but still fluctuating around it. The first spike of any two presynaptic spikes within a time window of $\Delta t_{on} = 1\,\text{ms}$ is the onset of stimulation. The length of Δt_{on} is chosen small compared to the membrane time constant $\tau_m = 15\,\text{ms}$, such that the excitatory postsynaptic potential of both spikes overlap each other. On the other hand Δt_{on} is chosen large enough to also include coincident spike pairings within the uncorrelated population. The dependency of the postsynaptic firing probability p on the delay Δt_{del} to the onset of stimulation is calculated and plotted as a histogram in Figure 16.

[9] named iaf_cond_exp in NEST

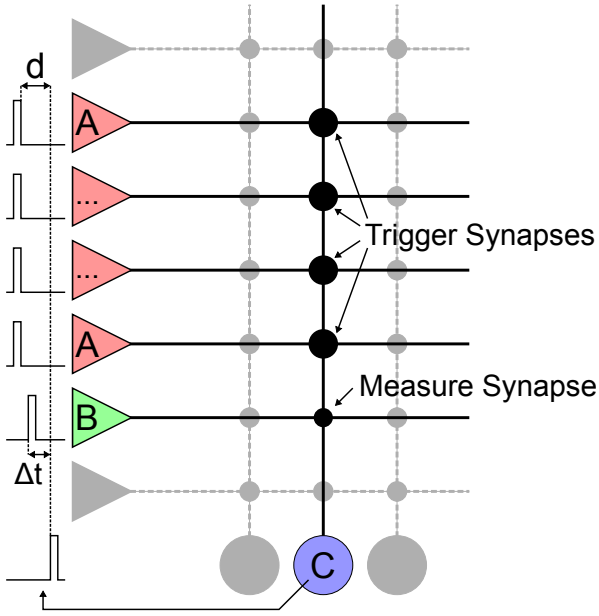


Figure 10: Setup to measure the correlation accumulation in a synapse. The postsynaptic neuron C is stimulated by several synaptic inputs A. The delay d between these presynaptic trigger spikes and the resulting postsynaptic spike is measured. Then the time of a spike sent into input B of the measured synapse can be adjusted to obtain a latency Δt between the pre- and postsynaptic spikes of the measured synapse.

2.3.4 Hardware Variations

The production variance of the ratio between the identically designed causal and acausal accumulation circuits within a synapse as well as the variance between synapses is a possible limitation on a fundamental level.

All measurements shown in the following haven been carried out with a chip-based FACETS hardware system (Schemmel et al., 2006, 2007), because it shares a conceptually nearly identical STDP circuitry as the wafer-scale hardware system, which was still in the assembly process during this study. For both hardware systems, the threshold can be configured only globally shared for all synapses in one row. The variance is measured by recording *STDP curves* and comparing the area under the causal against the area under the acausal branch, as described in the following.

Preparative Measurements In order to perform systematic measurements of hardware synapses, the delay Δt between a pre- and postsynaptic spike has to be adjustable. To this end the postsynaptic spike has to be triggered by synapses other than the measured one as illustrated in Figure 10. This setup allows the calculation of the spike time t_m that is the presynaptic input of the measured synapse: $t_m = t_s + d - \Delta t$, where t_s is the spike time that is the input for the trigger synapses, and d the measured delay between the presynaptic trigger spikes and the resulting postsynaptic spike. The synaptic weight of the measured synapse is set to $w = 0$ and may not evolve over time, whereas the weights and number of the trigger synapses have still to be determined.

Spike trains of 100 equidistant spikes with a frequency of 10 Hz are used as an input for

the triggers. The number of triggers, the weight of the trigger synapses and the stimulation strength *drviout* that is set row-wise for all synapses, are varied. After each emulation run, the recorded postsynaptic spikes are assigned to their presynaptic counterpart. The parameter *drviout* is increased until $> 99\%$ of all presynaptic spikes have exactly one postsynaptic spike within 5 ms in biological time. Then *drviout* is further increased until the percentage of well-triggered outputs drops below this limit, which is caused by multiple spikes for one stimulation, or the maximal value $drviout_{max} = 2.5\,\mathrm{V}$ is reached. The mean of these lower and upper limits is taken as the final value for *drviout* that decreases for high numbers of triggers and high synaptic weights of these, as shown in Figure 11A. For both cases this can be explained by the increase of the summed postsynaptic potentials.

Finally, the mean $D = \overline{d}$ over the delays d within one single run is measured and its mean $\overline{D}$ over 20 runs, the according standard deviation σ_D and the mean over the standard deviations of single runs $\overline{\sigma}_d$ are calculated and plotted in Figure 11B, C and D, respectively. In order to keep the variance of the measured delay d between and within single emulation runs as low as possible, the measured data suggests to use 4 triggers with synaptic weights of $w = 15$ throughout the hardware measurements. A lower number of triggers or lower weights cause an unstable spike elicitation, because the spike threshold V_{td} is crossed with a small slope of the membrane potential. On the other hand, as few as possible triggers should be used to avoid displacements of their postsynaptic potential due to different signal runtimes and different time stamps caused by the digital signal processing.

STDP Curve Measurement Before recording an STDP curve, the row-wise stimulation strength *drviout* is determined as described in the paragraph above, and the delay Δt is measured. The STDP curves themselves are recorded by applying equidistant pairs of pre- and postsynaptic spikes with a predefined latency Δt, as also explained in the paragraph above. The first 10 spike pairs are discarded due to pre-charge parasitic capacitances in the hardware circuitry and therefore ensure regular measured latencies Δt_m. Additionally, the pre- and postsynaptic spike trains are shifted slightly as long as Δt_m is measured within a tolerance range of $\Delta t \pm 0.02\,\mathrm{ms}$ in biological time. A measurement will only be continued, if each presynaptic spike has a postsynaptic counterpart, otherwise the last emulation run will be repeated. In order to obtain STDP curves, the number of sequent spike pairs in each emulation run is increased until the threshold V_t is crossed and hence a correlation flag is set (Figure 18A), or a maximum number of 250 spike pairs is reached. The inverse of this necessary number of spike pairs is plotted against the measured latency Δt_m between the pre- and postsynaptic spike train in Figure 18C. Due to the low repetition frequency (10 Hz) of single spike pairs, only the correlations within and not between spike pairs are accumulated.

Such STDP curves were recorded for 60 synapses within one synapse row. The balance between the mean of the causal and the mean of the acausal branches within this set of synapses has been coarsely adjusted beforehand by tuning the parameters $Vclra$ and $Vclrc$ as listed in Table 2, as this is possible row-wise. Because we are only interested in the variance of the synapse circuits, the exact offset is of minor importance.

Although the presented procedure was optimized to reduce the number of hardware emulation runs by starting at larger numbers of spike pairs for long latencies Δt, approximately 1000 emulation runs are needed to obtain one STDP curve. Each run takes approximately 0.85 s, which is mainly due to the low communication bandwidth between the host computer and the hardware system as already described by Brüderle (2009).

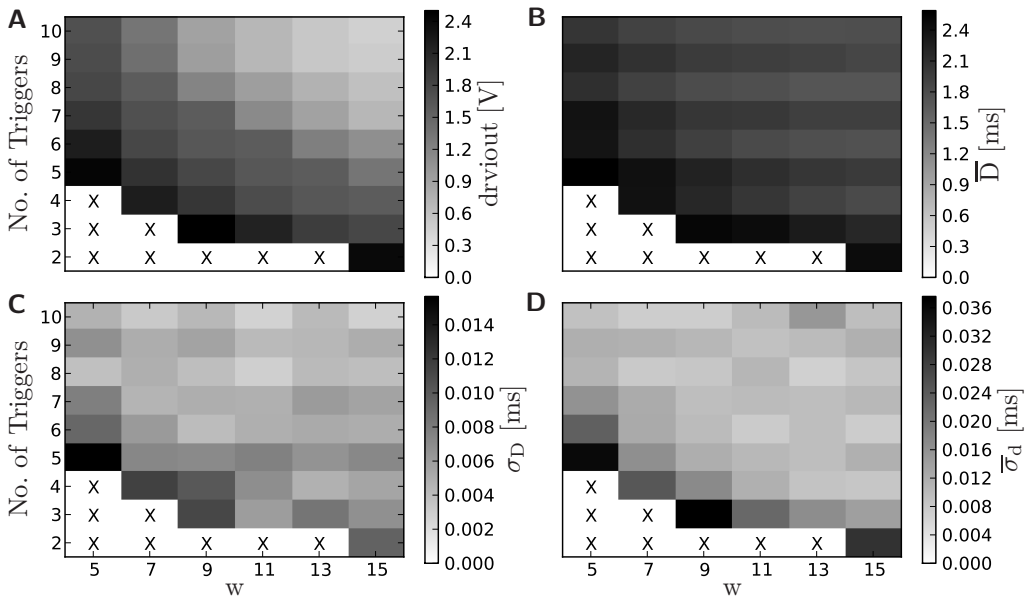


Figure 11: Delay measurements of the FACETS chip-based hardware system. All times are given in their biological interpretation. The mean delay $D = \overline{d}$ of each emulation run is obtained from the delays d between 100 spike pairs. The data points that are indicated with crosses are invalid, because no *drviout* value ensured a proper elicitation of one postsynaptic spike for each bunch of stimulation spikes (as described in Section 2.3.4). **(A)** The stimulation strength *drviout* as a function of the number of triggers and their discretized synaptic weights w. **(B)** The mean delay $\overline{D}$ of the postsynaptic spike over all runs. **(C)** The standard deviation σ_D of the mean in (B). **(D)** The mean $\overline{\sigma}_d$ of the standard deviations determined for each emulation run.

During the process of these STDP curve recordings a major bug within the synapse layout was found. The shape of STDP curves is dependent on and distorted by a signal pulse that determines the refractory period of the postsynaptic neuron. This bug will be solved in the next revision of the Spikey chip.

Analyses of Measurement Data The total area $A_t = A_a + A_c$ under each STDP curve (Figure 18C) is calculated and normalized by the mean of the absolute area $A_{abs} = |A_a| + |A_c|$ over all STDP curves. An ideally balanced, total area A_t would vanish, indicating that both branches are fully symmetric, as assumed for many pair-based STDP models as described in Section 1.2.1. The standard deviation σ_a of these areas is taken as one measure for the variance of the accumulation circuits. Besides this asymmetry, another measure is the standard deviation σ_t of the absolute areas A_{abs}. Therefore the absolute areas under each STDP curve are again normalized by $\overline{A_{abs}}$ and furthermore the mean of all these normalized areas is subtracted. For results see Section 2.4.4.

Software Simulation In order to predict the effects of the hardware variations on the network example, these variations are integrated into software simulations. First, the two standard deviations σ_a and σ_t are calculated from the data recorded as described in the paragraphs above, assuming Gaussian distributed measurement data. Then the thresholds for the causal and acausal branch are drawn from these two overlaying Gaussian distributions and applied to the existing, hardware-inspired software model (Section 2.3.1). Again the same network benchmark as described in Section 2.3.3 is used, but with a fixed correlation coefficient of $c = 0.025$ and an 8-bit LUT configured with 12 SSPs. The standard deviations of both Gaussian distributions are varied independently to quantify the influence of the production imperfections on the fundamental ability of synchrony detection as shown in Section 2.4.4.

2.4 Results

The restrictions caused by a small-size hardware implementation of a synapse (Section 2.1) are split into single isolated effects to allow separate analyses of each effect and hence the determination of bottlenecks. For this purpose, the study considers different levels of complexity.

The most limiting restriction of the FACETS hardware system (Section 2.1) are the finite resolution of synaptic weights, the frequency of weight updates and the variation of accumulated correlations in hardware. In the following, these restrictions are discussed and experimentally studied in detail. First, the dynamics of discretized synaptic weights are analyzed in Section 2.4.2, because of their fundamental role in plastic synapses. Furthermore, our studies are extended to whole synapses incorporating LUTs (Section 2.4.3) and lastly, to a simple network task that serves as a preparative verification before applying such synapses to more complex network setups (Section 2.4.3).

2.4.1 Feasible STDP Models

Different interpretations of biological measurements (Markram et al., 1997; Bi & Poo, 1998; Sjöström et al., 2001) led to several STDP models as described in Section 1.2.1. But which of those STDP models reviewed by Morrison et al. (2008) are feasible for discretization?

To answer this question, the transformation from continuous to discretized synaptic weights is crucial (Section 2.2.1):

In order to obtain a high resolution, we distinguish between bounded and unbounded STDP models. The *van Rossum* as well as the *power law* STDP model are meant to be unbounded (van Rossum et al., 2000; Morrison et al., 2007) and have long tails towards high synaptic weights. Including these long tails into a range of equidistant discretized weights causes a low resolution for the occurring majority of low weight values. On the other hand, cutting the tail causes an accumulation of weights at the highest discrete value. One solution would be to bin the weight range with bins of variable size. Large bins for high and small bins for low synaptic weights. However, a variable bin size would require more hardware efforts.

In contrast to unbounded STDP models, the *Gütig* STDP model is bounded to a finite weight range (Gütig et al., 2003). These boundaries are due to the limiting character of the weight-dependent factor $F(w)$ listed in Table 1 and the clipping to the upper and lower weight limit, if one of those boundaries is exceeded. Clipping occurs predominantly, if the step size Δw is large for weights near the boundaries. This is the case for small μ, especially for $\mu = 0$, which is also called the *additive* STDP model. Because of the straightforward transformation to discretized weights by binning the whole bounded weight range as described in Section 2.2.1, the Gütig STDP model is well suited for hardware realization.

Throughout this study, the intermediate Gütig model ($\mu = 0.4$) is used, since it represents a mixture of the common *multiplicative* ($\mu = 1$) and additive STDP model and provides a balance between stability and sensitivity of competitive synaptic learning (Gütig et al., 2003).

2.4.2 Configuration and Distributions of Discretized Weights

Following the question whether a 4-bit synaptic weight resolution is sufficient leads us to the question of how the discretization of synaptic weights influences their dynamics. These LUTs are not only a discretized weight representation, but also influence the dynamics in case of causal and acausal correlations. Studying the nature of LUTs is in other words analyzing the effects of discretization on the dynamics of synaptic weights. In order to illustrate the dynamics of discretized weights graphically, equilibrium distributions are determined as described in Section 2.2.3. Those equilibrium distributions are the direct consequence of the pattern stored in LUTs, i.e. how the discretized weights are projecting to each other.

Configuration and Dynamic Range When continuous-like weight evolutions should be reproduced, the weight resolution and the step sizes Δw of weight changes due to accumulated spike time correlations are dependent on each other. These step sizes are expressed with the number of sequent SSPs that are used for configuring the LUT (Section 2.2.2). In the following, two examples are given that define the upper and lower limit of the dynamic range.

First, if the weight resolution is low, the number of SSPs should be at least high enough to ensure a sufficiently large weight step such that the next LUT entries can be reached according to the transformations introduced in Section 2.2.1. If this minimum number of SSPs is not given, the LUT entries project to themselves and prevent synaptic weights from evaluating dynamically. Examples are given in Table 4c and Figure 12A (15 SSPs), where the middle LUT entries are projecting to themselves and therefore represent dead ends.

Second, many sequent SSPs can be necessary for constructing the LUT, e.g. if the frequency of weight updates is low due to hardware limitations. If the number of SSPs is too

high, middle LUT entries are not reached by others. Consequently the equilibrium distribution becomes bimodal and the weights are jumping from one extreme value to the other as shown in Table 4b and Figure 12A (225 and 500 SSPs). This behavior prevents proper synapse dynamics, too.

The effects leading to the upper and lower limit of SSPs are combined and described as the percentage of *non-functional LUT entries*. Non-functional LUT entries are defined as entries projecting to themselves in case of causal and acausal weight updates or as entries receiving no projection from any other LUT entry.

For low numbers of SSPs the percentage of non-functional LUT entries is dominated by the effect of rounding. Only a small decrease in the number of SSPs can make all LUT entries round to themselves instead of neighboring entries (<10 SSPs in Figure 12B). In contrast to this, the percentage of non-functional LUT entries for high numbers of SSPs is dependent on the step size of a weight update. The larger the step size the larger the gap of unreachable LUT entries becomes (>200 SSPs in Figure 12B).

For higher resolutions, meaning more bits that represent the discretized weight, weight updates with high frequencies (low number of SSPs) are less affected by rounding errors as the high resolution can resolve the small weight steps (Figure 12C). A resolution of 8 bits is already enough to resolve the smallest weight steps resulting from the minimum number of SSPs. The upper limit of SSPs has a minor influence on the following studies, because it occurs only for very low frequencies of weight updates, where the amount of required accumulated correlation is too high, to be stored in the hardware circuitry.

Equilibrium Distributions Assuming pre- and postsynaptic spike trains to be those of a Poisson process, as for example seen in balanced random networks, results in uni-modal weight distributions (Morrison et al., 2007), if the resolution and frequency of weight updates are chosen within the dynamic range discussed in Section 2.4.2 (see especially Figure 12A). In order to quantify the effect of discretization on the time scale of minutes to months, we analyze long-term dynamics in this section. These are especially interesting as simulations over long periods are one crucial benefit of highly accelerated neuromorphic hardware devices. Long-term dynamics are realized by updating a discrete synaptic weight multiple times according to a LUT as described in Section 2.2.3. Applying Poisson input and simulating over long periods cause the weight distribution to converge against an equilibrium. In the following the equilibrium distributions of discrete weights are compared to reference distributions. The deviation is used to quantify the distortion of discretization.

Our first approach was to use a reference distribution that is obtained with analytical methods as employed by van Rossum et al. (2000). Taylor expanding the underlying master equation in combination with the equilibrium boundary conditions result in a differential Equation (19) that can be solved analytically, or in case of many SSPs, numerically. But this Taylor expansion in combination with the applied boundary conditions does not hold for large step sizes as specified later in this section. Replacing the weight clipping approach by reflective boundaries, which were assumed to solve Equation (19), does not improve the results (not shown in this study).

Because the analytical solution does not hold for large step sizes Δw, an equilibrium distribution obtained by sequent evaluations of continuous weight values is taken as a reference. For details about these equilibrium distributions see Section 2.2.3.

The Figure 13A shows the effect of discretization on the equilibrium state. The distortion

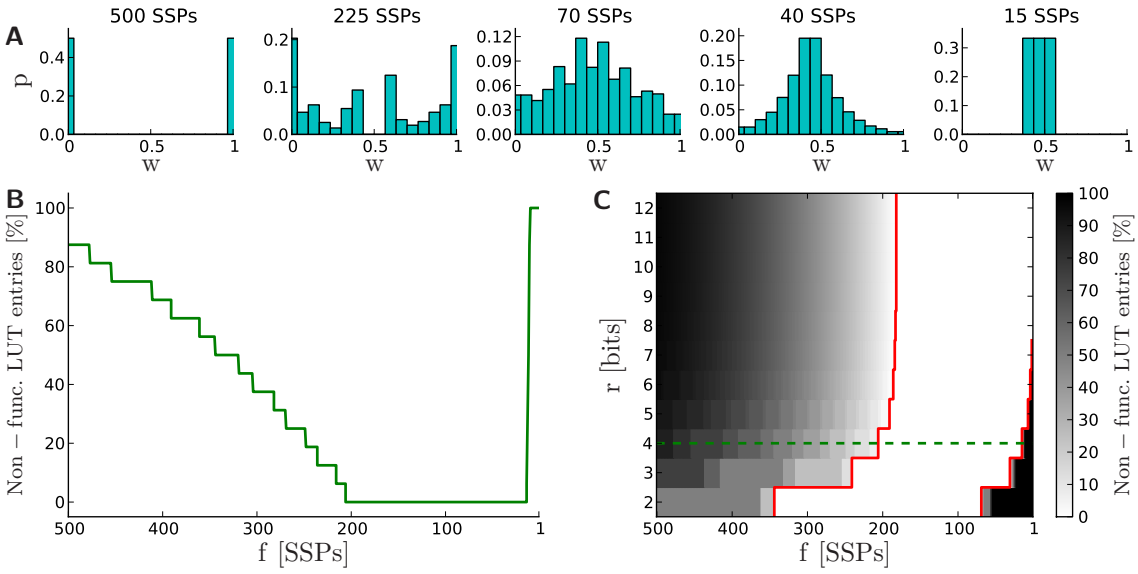


Figure 12: **(A)** Equilibrium weight distributions assuming a 4-bit weight resolution and 500, 225, 70, 40 and 15 SSPs. The histograms for 225 and 500 SSPs show that there are no projections to the middle LUT entries. Less SSPs lead to the usage of all LUT entries (70 SSPs) and a further decrease narrows the equilibrium distribution (40 SSPs). If the step sizes are even smaller (15 SSPs), the middle LUT entries are projecting to themselves. **(B)** The percentage of LUT entries with a 4-bit resolution that are non-functional and therefore restricting the dynamics of discretized weights. The limit towards low numbers of SSPs is caused by rounding errors, whereas the other limit is due to large weight steps. **(C)** Same as (B), but with varying resolution r. The dashed green line indicates the position of (B). The red line marks the acceptable dynamic range (non-functional LUT entries $< 10\%$).

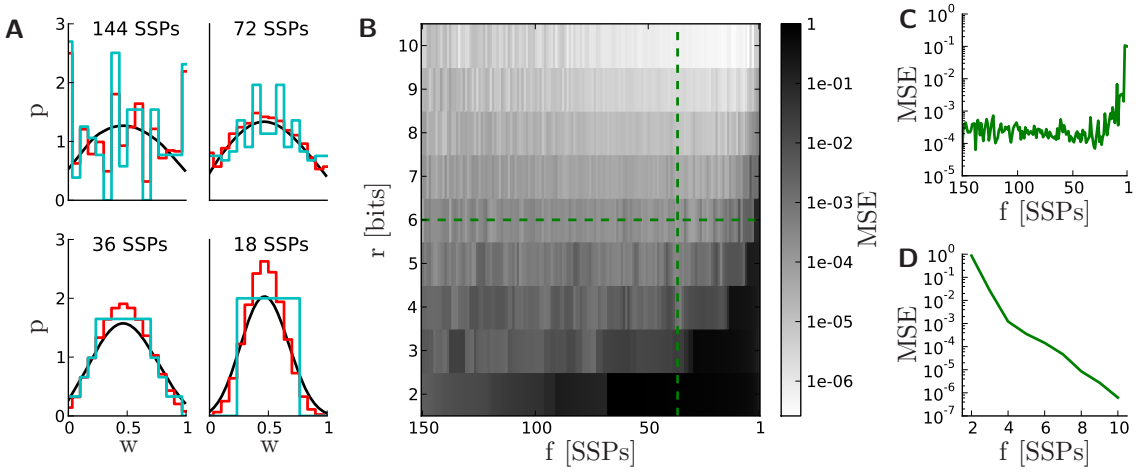


Figure 13: **(A)** Equilibrium weight distributions obtained by the numerical integration of the analytical solution (black), a numerical solution (red) and a discretized numerical solution with a 4-bit resolution (cyan). From upper left to lower right with a decreasing number of SSPs. **(B)** The mean squared error between the numerical and the discretized numerical equilibrium distributions as a function of the resolution r and the frequency of weight updates f. **(C),(D)** Cross sections of (B) at 6 bits and 36 SSPs, respectively.

is large for small numbers of SSPs, because the step sizes are smaller and hence rounding errors occur. For high numbers of SSPs the numerical distribution shows the same erratic behavior as the discretized numerical distribution. The peaks are due to boundary effects: If the probability accumulates at the boundaries because of large step sizes, the probabilities for the following weight values increase as well. These boundary effects in combination with large step sizes are not sensitive to discretization anymore.

As we are interested in the difference between discretized and continuous weights the MSE between their equilibrium distributions is calculated and plotted in Figure 13B. The similarities to Figure 12C suggest that the distortion of the equilibrium distribution is a direct consequence of the weight discretization. Hence these simple LUT analysis as described in Section 2.4.2 allows us to estimate the weight distortions within long-term simulations.

2.4.3 Temporal Evolution of Discretized Weights

The effects of discretized weights in combination with correlation accumulation can be seen in Figure 14. As the pre- and postsynaptic spike trains are correlated in a causal fashion, the causal accumulation of correlation is increasing much faster than the acausal one (Figure 14A). The causal accumulation trace crosses the threshold first and evokes a potentiating weight update twice before the acausal evokes a depressing weight update (Figure 14B). The first two causal weight steps Δw direct to the neighboring discretized weights. In contrast to this, the acausal one is rounded to the next but one LUT entry as explained in Section 2.4.3.

Furthermore a synapse with continuous weight, but with the reduced symmetric nearest-neighbor spike pairing scheme of the hardware synapse, was implemented as described in Section 2.3.1. This spike pairing scheme causes a different strength of response to the same causal correlations compared to the symmetric all-to-all spike pairing scheme as can be seen

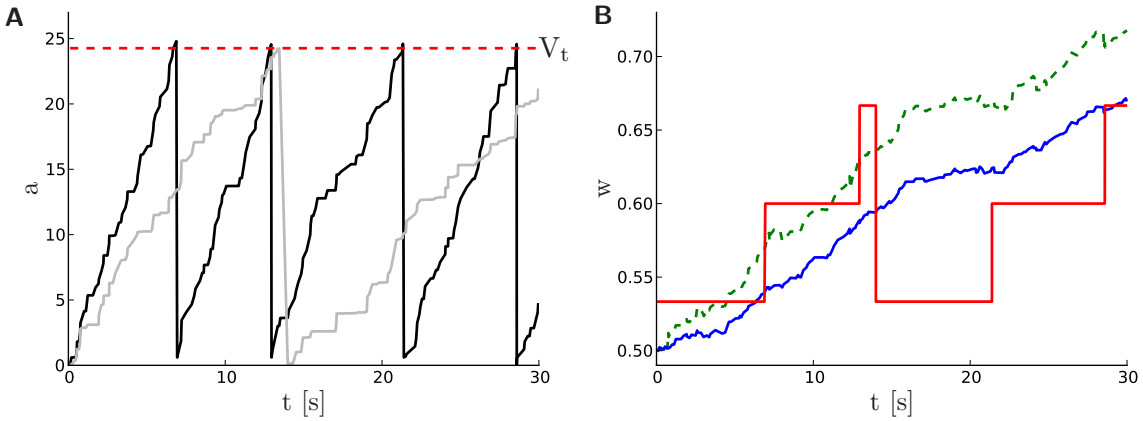


Figure 14: **(A)** Evolution of the accumulated correlation a for the causal (black) and acausal (gray) case. The weight is updated and a is reset, if the threshold V_t is crossed. **(B)** Corresponding weight trace (red) using a 4-bit LUT, which is configured with the intermediate Gütig STDP model and $n = 30$ SSPs. The reference weight traces with continuous weights and the symmetric all-to-all spike pairing scheme (dashed green) as well as the reduced symmetric nearest-neighbor spike pairing scheme (blue) for the same stimulation pattern (MIP input with $c = 0.5$ and $r = 10\,\mathrm{Hz}$).

in Figure 14B.

Single Synapse Dynamics Now we extend our studies on LUTs to a single synapse incorporating a LUT as described in Section 2.3.1. In addition to the analysis on discretized weights (Section 2.4.2) a pre- and postsynaptic neuron are introduced that determine the weight evolution of the synapse by their spike times (Figure 15A). The weight trace of a single run is shown in Figure 15B that shows an increased standard deviation for discretized weight traces compared to those for continuous weights. The difference is caused by sparser, but bigger weight changes in case of discretized weights. The standard deviation increases further with lower resolutions (not shown here).

The difference between both traces using discretized and continuous weights for a given correlation coefficient c is determined by calculating the MSE. This difference is dependent on the resolution and the number of SSPs as shown in Figure 15C. The difference is particularly significant for low numbers of SSPs as the relative rounding errors occurring during the configuration of the LUTs are high. To further illustrate the possibly critical impact of these rounding errors, the slight asymmetry α in Gütig's STDP model can be taken as an example. In an extreme case the symmetric branch could be rounded down, while the asymmetric branch is rounded up as can be seen in Figure 14. This would increase the former slight asymmetry drastically and therefore enlarge the difference between both traces. The high impact of rounding is the major reason for the dominating role of the LUT onto the resulting weight trace.

To get a deeper insight into the dynamics of single weight traces, we will discuss the relationship between the resolution, the number of SSPs and the frequency of weight updates in detail. The number of SSPs for a given weight resolution defines the threshold as described in Section 2.3.1 and therefore influences the frequency of weight updates. The lower the number

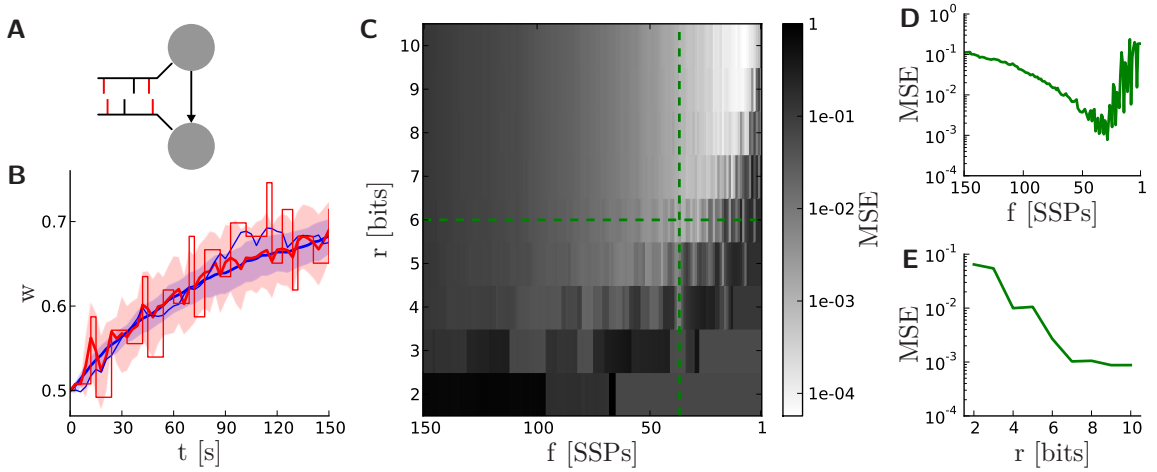


Figure 15: **(A)** Network layout for the single synapse analyses. A STDP synapse (arrow) connects two neurons, the output spike trains of which are correlated as depicted with the red bars. **(B)** Weight traces for discretized weights (6 bits, 36 SSPs) in red and continuous weights in blue for a correlation coefficient of $c = 0.2$ and an initial weight $w_0 = 50$. The means and standard deviations over 30 seeds are plotted as bold lines and transparent areas, respectively. Single weight traces for one arbitrarily chosen seed are depicted as thin lines. **(C)** Mean squared error (MSE) between the discretized and continuous weight traces plotted over the resolution r and the frequency of weight updates f. **(D),(E)** Cross sections of G at 6 bits and 36 SSPs, respectively.

of SSPs, the higher the frequency of weight updates. A high frequency of weight updates would allow us to closely keep up with the continuous weight evolution chronologically, but the less SSPs the smaller the weight steps get, and therefore the higher the relative rounding errors while constructing the LUT (Section 2.4.2). To reduce the rounding errors and resolve such small weight steps the weight resolution has to be chosen sufficiently high. If the weight resolution is limited, a lower bound of SSPs defines an upper limit of the frequency of weight updates, as it takes longer to accumulate enough correlations to cross the corresponding threshold. Assuming a resolution of 4 bits, 15 SSPs are required to have a LUT with a small percentage of non-functional entries, but at least 36 SSPs are necessary for proper long-term dynamics (Figure 12).

Varying the initial weight w_0 or the correlation coefficient c does not change this outcome qualitatively, just the absolute difference of both mean weight traces.

Network Benchmark: Synchrony Detection Subsequently we focus less on the exact weight trace of a single synapse (Section 2.4.3), but rather on populations of synapses that fulfill a task, in this case the detection of synchronous firing within neural networks. During the solving of this task, exact weight traces may play an inferior role as compared to e.g. the means over synapse populations. The principle of synchrony detection is a crucial feature of various neural networks that implement STDP, e.g. Davison & Frégnac (2006). Here it is introduced by means of a simple benchmark, avoiding difficult analyses of complex neural networks.

The benchmark network is described in Section 2.3.3 as well as illustrated in Figure 16A.

Using continuous synaptic weights, the postsynaptic neuron is more likely to fire, if a couple of neurons of the correlated population are firing synchronously. Consequently all STDP synapses that connect to the correlated population are strengthened more often than those connected to the uncorrelated population. This is shown by the lower probability for synchronous presynaptic spikes evoking a postsynaptic spike for $\Delta t < 170\,\text{ms}$ plotted as a PSTH (Section 2.3.3) in Figure 16D, if static synapses are applied instead of STDP synapses. Later postsynaptic spikes are barely influenced by the presynaptic spikes, because the membrane potential τ_m and the excitatory synaptic time constant τ_{ex} are small compared to the delay between the pre- and postsynaptic spikes.

With respect to STDP applications, the discretization of synaptic weights should have as little effect on the ability of distinguishing both populations as possible. But if discretized weights are applied, the standard deviation of the weight traces belonging to one population rises with decreasing weight resolution (Figure 16C). This fact is again closely connected to the configuration of LUTs that incorporate the transformations from continuous to discretized weights with all its rounding issues (Section 2.2.2 and Section 2.4.2). On the other hand, discretized weights can also be advantageous under certain circumstances. For example, rounding can separate the two groups of synaptic weights: synaptic weights of one group are rounded up, while the others are rounded down. But those cases are unlikely and can be compensated by runs over different random generator seeds.

Taking the mean of these runs, the ability of separating the two groups of synaptic weights increases with the correlation coefficient c (Figure 16E). The corresponding LUT is constructed by using 36 SSPs for a resolution of 4 bits, as the rounding errors are small for this update frequency (Figure 12B). Increasing the resolution, but not the number of SSPs, does not change the performance significantly. The performance can even get worse, if the rounding in case of a lower weight resolution facilitates the separation of the synapse groups (compare 4 bits to 5 bits in Figure 16E). However, reducing the number of SSPs allows the synapses to detect fluctuations of synchrony on smaller time scales and consequently improve the performance. Such an decrease in the number of SSPs is not possible for the 4 bit case, because rounding errors prevent proper synapse dynamics (Section 2.4.2).

Further Constraints of the FACETS Hardware System In addition to the discretization of synaptic weights that has been analyzed so far, we now consider other restrictions of the hardware implementation. First, we apply the limited frequency of weight updates and second, we analyze the effects of a common reset. For this purpose we compare the performance gain of introducing a second reset line to the performance gain of a more detailed readout of the accumulation circuits via ADCs as described in Section 2.3.3.

From a hardware point of view the frequency of weight updates is limited quite significantly, since groups of synapses are updated in sequence and each weight update requires a minimum time to be performed. Instead of performing a weight update, whenever the accumulated correlations cross the threshold, in our modified model this crossing is checked in equidistant time intervals. If the frequency of weight updates is too low, the threshold is exceeded strongly and consequently the weight evolution is distorted compared to the continuously updating case.

This distortion is analyzed quantitatively using the same network benchmark as presented in Section 2.4.3. In addition to the application of discretized weights the frequency of weight updates is limited. Consistent with the studies of only discretized weights (Section 2.4.3),

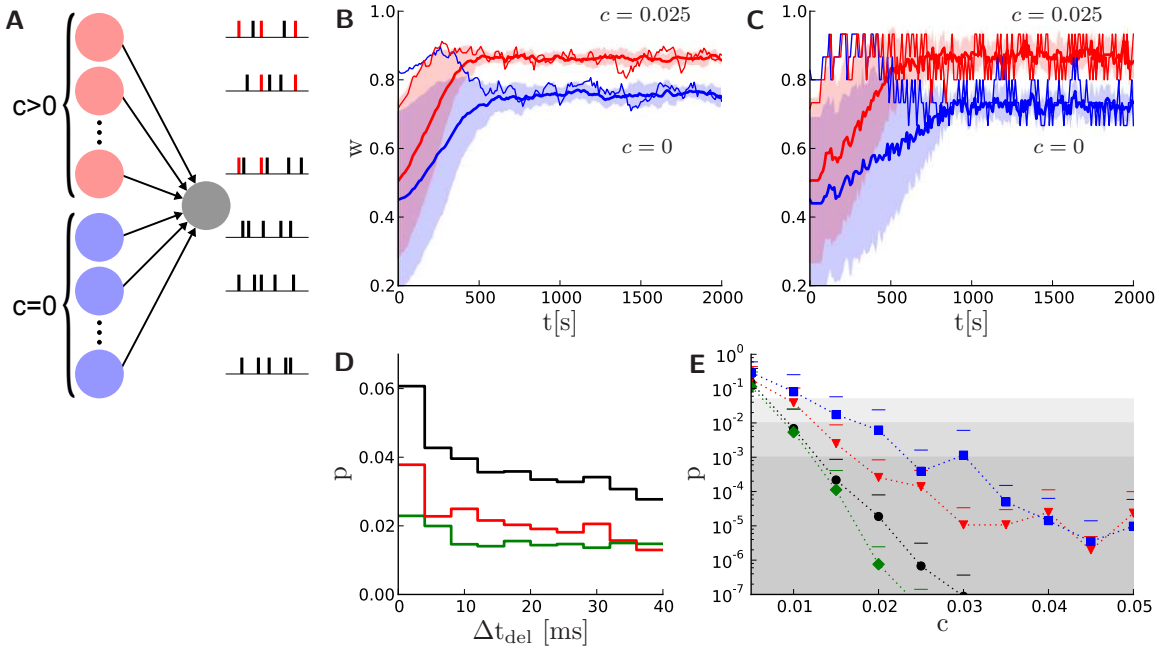


Figure 16: **(A)** Schematic of the benchmark network layout. Two populations of neurons are connected to one neuron by STDP synapses (arrows). On the right hand side example spike trains sent by the input neurons are shown. Red bars indicate correlated spiking. **(B)** The mean weight traces (thick lines) and their standard deviations (transparent areas) are plotted for both the STDP synapses that connect the correlated ($c = 0.025$, blue) and the uncorrelated ($c = 0$, red) neuron population for an arbitrarily chosen random number generator seed. The thin lines represent a single synapse randomly picked from each group. **(C)** Like (B), but the STDP synapses have discretized weights with a 4-bit resolution and the LUT is configured with 36 SSPs. **(D)** Peri-stimulus-time-histogram as described in Section 2.3.3 using continuous weights. The red histogram shows the difference between a simulation using STDP synapses (black) and static synapses (green). **(E)** The probability p that the groups of synaptic weights connected to the correlated neuron population can not be separated from the uncorrelated one as a function of the correlation coefficient c. Black circles denote continuous synaptic weights, red triangles show discretized weights with a 4-bit resolution using a LUT configured with 36 SSPs. Blue squares and green diamonds represent discretized weights using a LUT with a resolution of 8 bits and configured with 36 and 12 SSPs, respectively. For further details see Section 2.4.3. The background shading represents the significance levels from light to dark: < 0.05, < 0.01 and < 0.001.

a low weight resolution with a high number of SSPs and a high weight resolution with a low number of SSPs are compared to the reference synapse (Section 2.3.1). The Figure 17A shows that the ability to detect synchrony in the input stays almost constant down to a weight update frequency of $f = 1\,\mathrm{Hz}$. Lower frequencies distort the weight evaluations drastically. The current FACETS hardware system has a frequency of weight updates above $f = 1\,\mathrm{Hz}$ that is sufficient for its 4-bit weight resolution.

Another major constraint of the FACETS hardware system is its single reset line for both the causal and acausal accumulation circuit. This means that both circuits are reset, if a weight update is performed. The performance gain of a second reset line to reset the circuits independently is illustrated in Figure 17B. Synapses with limitation to one reset line perform especially bad, if the frequency of weight updates is low due to a high number of SSPs. The cause lies in the underestimation of the accumulated correlations of the accumulation branch, that failed in crossing the threshold first. As a consequence, the weights of the correlated and uncorrelated synapse population increase both due to the fact that the causality dominates the acausality, caused by our feed-forward network architecture. As a solution, the small fluctuations around the balance of both causalities could be taken into account, by using very high update frequencies, meaning very low numbers of SSPs. In connection with low resolutions of synaptic weights, such low numbers of SSPs result in a loss of dynamics (Section 2.4.2).

One approach to solve this issue is to introduce an ADC that allows the comparison of the accumulated correlations to multiple thresholds. LUTs with 11 to 41 SSPs (in steps of 2) with a 4-bit resolution as well as 1 to 46 SSPs (in steps of 3) with a 8-bit resolution are applied. The lower limit of SSPs is chosen according to the dynamic range of LUTs obtained in Section 2.4.2, whereas the upper limit is chosen to cover a reasonable dynamical range consistent with the results of Section 2.4.2 and Section 2.4.3. In order to exploit the whole range of thresholds for the given firing rates the frequency of weight updates is set to $f_u = 0.2\,\mathrm{Hz}$. Nevertheless such an ADC does only compensate for a single reset line, if high weight resolutions are used (Figure 17B). Only LUTs with a small percentage of non-functional entries (Section 2.4.2) that are configured with a low number of SSPs, allow to update the inferior accumulation branch, if the superior branch crosses the threshold and elicits an update. LUTs with a 4-bit resolution have to be constructed with at least 11 SSPs (Section 2.4.2) and are therefore not feasible for the usage in a system with single reset lines. LUTs with an 8-bit resolution are performing well, because small fluctuations down to one SSP can be resolved.

Nevertheless, the current revision of the FACETS wafer-scale building block, called HICANN[10] (Schemmel et al., 2008; Millner et al., 2010), offers the possibility to combine vertically adjacent synapses. Each synapse can be configured to accumulate only either causal or acausal correlations, while both are updated in a common process, enabling a behavior that mimics a second reset line. The implementation of a real second reset line into the current hardware system is not possible without major design changes, but is considered for future chip revisions.

2.4.4 Synapse Variations

In Section 2.4.3 the effects of hardware constraints caused by the trade-off between the number and size of synapses has been discussed in detail. So far, the study is independent of a

[10] High Input Count Analog Neural Network

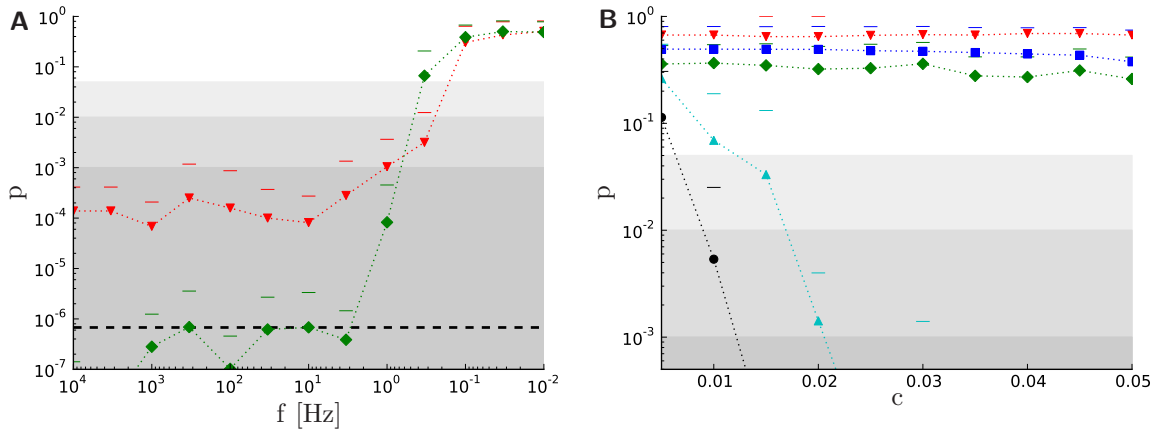


Figure 17: **(A)** Probability p that the two neuron populations are not separable as a function of the frequency of weight updates f. Weights are discretized with a 4-bit resolution (36 SSPs) in red triangles and 8-bit resolution (12 SSPs) in green diamonds. The corresponding probability for the unrestricted reference synapse is plotted as a black dashed line. **(B)** The same probability as in (A) is plotted against the correlation coefficient of the correlated neuron population in Figure 16A. The STDP implementation with a common reset is depicted with red down triangles (4-bit weights and 36 SSPs) and green diamonds (8-bit weights and 12 SSPs) and the one with a 4-bit ADC with blue squares (4-bit weights) and cyan up triangles (8-bit weights). As reference with neither a common reset nor using an ADC, but including discretized weights (8-bit weights and 12 SSPs) in black circles.

real hardware system, e.g. neglecting the transistor imperfections due to the manufacturing process. These imperfections are the limitation on the lowest level, as they influence the accumulation process itself. The smaller and denser the transistors, the larger the variances from their theoretical properties (Pelgrom et al., 1989). Hence the variance between the causal and acausal accumulation circuit within individual synapses as well as the variance between synapses (Section 2.3.4) were measured on an already available FACETS chip-based system (Schemmel et al., 2006, 2007).

The Figure 18A shows the applied spike pattern and the evaluation of the triggered accumulation circuit. The numbers of spike pairs applied until the threshold is crossed for a certain latency Δt determines the STDP curves as plotted for different synapses in Figure 18B. The deviation σ_a of the asymmetry within a synapse as well as the deviation σ_t between synapses (for details see Section 2.3.4) are obtained from the histograms shown in Figure 18D and E, assuming Gaussian distributions.

Subsequent software simulations incorporating these variances as described in Section 2.3.4 are crucial for the further hardware development, as the synapse circuits can be modified in order to reduce the production imperfections, but then they consume more resources on the chip. Furthermore the question of the minimal resolution of synaptic weights and the effects of other constraints can be discussed with respect to real hardware measurements.

The Figure 18F shows the ability of synchrony detection (for details see Section 2.3.4) in dependency on σ_a and σ_t. The functionality of our network example is decreasing significantly with increasing asymmetry between the causal and acausal branch within a synapse, but is hardly affected by variations between synapses up to more than 30%. Thus modifying the

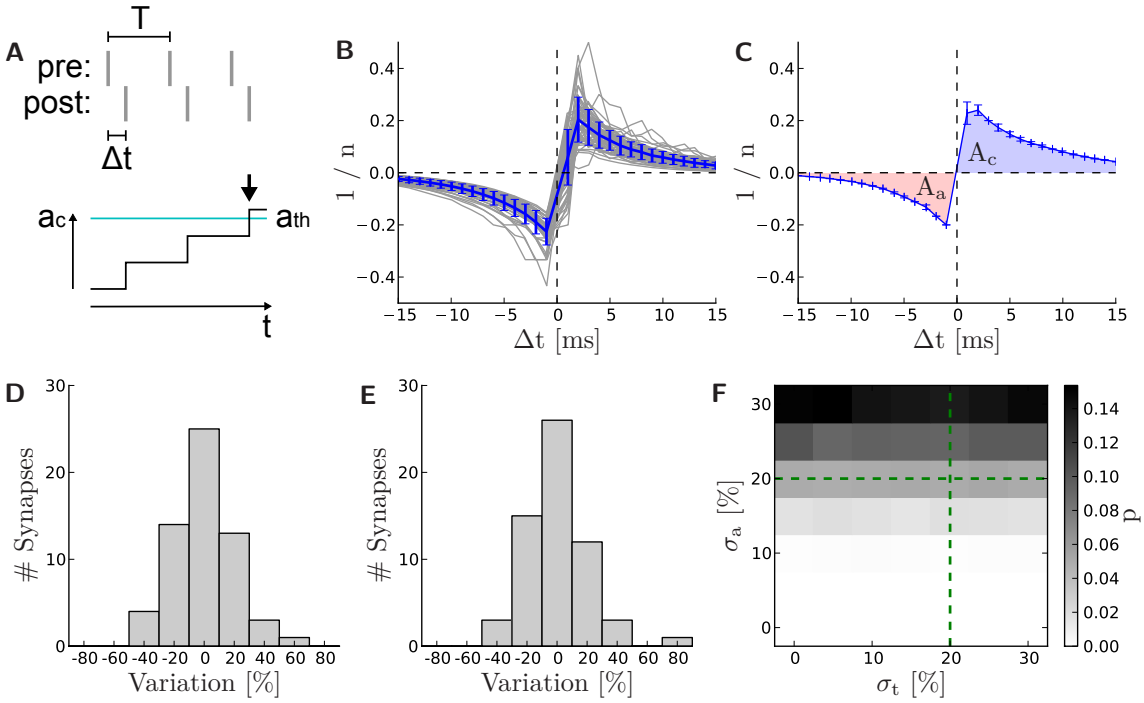


Figure 18: **(A)** The STDP curve measurement setup. At the top the spike patterns of the pre- and postsynaptic neurons. At the bottom the accumulated correlations that cross the threshold (arrow) and therefore define the number of spike pairs up to this point in time (here $n = 3$). **(B)** The STDP curves of 60 synapses within one synapse row (gray) as well as their mean with error (blue). **(C)** One arbitrary STDP curve with the area under the curve indicated. Red for the acausal and blue for the causal section. **(D)** The asymmetry between the causal and acausal branch area within one synapse row with a measured deviation of $\sigma_a = 22\%$. **(E)** The absolute areas within one synapse row with a measured deviation of $\sigma_t = 20\%$. **(F)** The ability of synchrony detection in dependency on the hardware variances σ_a and σ_t. The measured variances of the FACETS chip-based system of (D) and (E) are depicted as green dashed lines.

synapse layout in order to reduce the asymmetry within one synapse can be expected to increase the synchrony detection performance of hardware neurons profoundly.

2.5 Conclusion drawn from this Section

The answer to the question, whether a synaptic weight resolution of 4 bits is enough, is obviously dependent on the network architecture, the task to be solved, the maximal possible frequency of weight updates, the variances due to production imperfections as well as the synapse model to be emulated in the neuromorphic hardware system. For example, the idea of assuming bistable synapses, that is synapses with a 1 bit weight resolution, is still present in recent modeling approaches, e.g. those by Amit & Fusi (1994) as well as Clopath et al. (2008). On the other hand, various STDP models relying on the biological measurements of Bi & Poo (1998) are assuming continuous strengths of synapses, allowing a competition between synapses in a more differentiated manner (Section 1.2.1).

If continuous synaptic weights are assumed, the distributions of these weights are shown to be narrow in large-scale neural networks Morrison et al. (2007). Consequently, when having only limited hardware resources available, an optimal resolution at weight values of high probabilities must be found. But synaptic weight discretization not only limits the accuracy of a single weight value (Section 2.4.3), but also broadens their equilibrium distribution in large neural networks spanning the whole dynamic range as illustrated in Figure 13A, which happens due to increased step sizes (Section 2.4.3). Since this effect is not avoidable for the considered hardware systems, a linear transformation between continuous and discrete weights was sufficient for this study (Section 2.2.1).

The distortion of synapse dynamics, caused by synaptic weight discretization, can be explained by simple analyses of LUTs (Section 2.4.2). In contrast to the problem of rounding for small weight steps, too large step sizes make it impossible to obtain proper synapse dynamics. Once a LUT has been configured with improper entries, all synapses are static or will evolve in an erratic way. Comparing the equilibrium weight distributions of discrete weights to those of continuous weights reflects the configuration of the LUT and gives an impression about both long-term (Section 2.4.2) and short-term dynamics (Section 2.4.3). Consequently, such simple LUT analyses are a necessary tool to evaluate a synapse design.

In addition to the evolution of single synapses, a benchmark network model was defined and investigated, in order to analyze the competition between two neuron populations (Section 2.4.3). Also in this context, the simple LUT analysis of Section 2.4.2 served to obtain a decent starting point. This network layout is sensitive to small fluctuations, which can be followed better, if the frequency of weight updates and necessarily the weight resolution is higher. This interplay of weight resolution, frequency of weight updates and weight distribution is a crucial consequence of the weight discretization. Nevertheless the benchmark network with 4-bit synaptic weights was able to distinguish correlated from uncorrelated neuron populations down to small correlation coefficients (Figure 16E).

Besides the limitation of a low weight resolution, we now analyze further restrictions of the current FACETS wafer-scale hardware system (Section 2.1): the update frequency and a common reset mechanism. Regardless of the exact weight trace of each single synapse, but interpreting the final weight distribution within the network, hardware synapses can be robust against the frequency of weight updates (Section 2.4.3). The current FACETS wafer-scale system is above the lower limit of the update frequency of about $f = 1\,\mathrm{Hz}$ on biological time scale, if less than $\frac{1}{8}$ of all synapses are activated for the use with STDP (Figure 19). Introducing ADCs, as described in Section 2.3.3, does not compensate for a common reset, as long as the weight resolution, which determines the lower limit of weight changes, is not increased, as well (Section 2.4.3). Nevertheless, a satisfactory benchmark performance can be obtained by combining two synapses of the FACETS hardware system, one for evaluating causal and one for evaluating acausal correlations. Although the double assignment of synapses is a drawback in terms of synapse resources, these combined synapses allow STDP experiments with the current hardware system design. The costly implementation of two reset lines for each neuron will be considered in future hardware systems.

Considering the low resolution, 4-bit synaptic weights are sufficient to obtain a good performance in terms of the network benchmark (Section 2.4.3). A higher resolution would suffer from a too low frequency of weight updates given by the FACETS wafer-scale hardware system. Vice versa, a higher frequency would not improve the performance of synapses with a weight resolution as little as 4 bits significantly, because the threshold for the accumulated correlations is quite high and hence weight updates are elicited infrequently (Section 2.4.3).

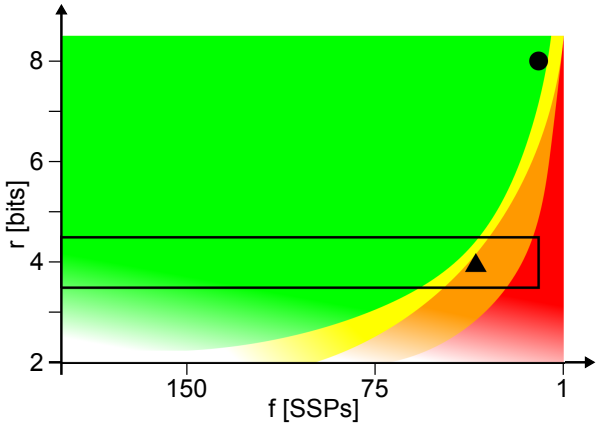


Figure 19: The sweet spots of weight dynamics for a 4-bit (triangle) and 8-bit (circle) weight resolution in terms of SSPs are plotted with respect to the limits arising from the weight discretization (Section 2.4.2) in orange, the equilibrium weight distributions (Section 2.4.2) in yellow and the single synapse weight traces (Section 2.4.3) in green. In red the area of non-functional LUT entries is depicted (Section 2.4.2). The rectangle marks the area realizable by the FACETS wafer-scale hardware systems.

In addition the benefit of an increased weight resolution is low, because the variances of the synapse circuits due to production imperfections limit the hardware system on its transistor level (Section 2.4.4). In conclusion a 4-bit weight resolution is sufficient for the hardware synapses implemented. In order to improve the performance, e.g. in the network example, an increase of the weight resolution and update frequency as well as hardware synapses with less variation between their causal and acausal branch are required.

Although we focused on the intermediate Gütig STDP model, all other STDP models listed in Table 1 can be analyzed the same way and have conceptually the same result, unaffected from their individual equilibrium weight distributions.

As a next step, our hardware synapse model can replace the regular STDP synapses in software simulations of established neural networks examples, to test their robustness and applicability for physical emulation in the FACETS wafer-scale hardware system. If such a neural network, or a modification of it, qualitatively reproduces the software simulations, it can be emulated by the hardware system, with similar results to be expected. Thus, preparative software simulations allow foregoing modifications of the network architecture to ensure the compatibility with the real hardware system.

With respect to more complex STDP models, the hardware system is currently extended by a microcontroller that is in control of all weight updates and is programmable with more complex STDP rules as for example the one of Clopath et al. (2008).

3 Parameter Fitting Environment for Hardware Neurons

So far, only synapses of the FACETS wafer-scale hardware system were analyzed, verified and improved. This part will provide studies that focus on the second essential building block of the same FACETS wafer-scale hardware system, the neurons. For this purpose, a neuron fitting environment is introduced that utilizes the FACETS hardware system as a possible emulation backend. The highly accelerated wafer-scale system represents a promising tool for statistical optimization methods (Section 3.2.5), which allows an automatic fitting of neuron parameters to reference neuron activity. The reference neuron activity can be defined by biological measurements, as e.g. in the Quantitative Single-Neuron Modeling Competition introduced in Section 3.1, or by software simulations of arbitrary neuron models. As the AdEx neuron model is implemented in the FACETS wafer-scale system (Section 1.1.3, Section 3.3.1), a large variety of different neuron types as described by Naud et al. (2008) can be reproduced by this hardware system.

The neuron fitting environment described in the following is one possible solution to automatically find a neuron parameter set that reproduces a given neuron activity best, which is provided by biological recordings, or software simulations. Both a software simulator, e.g. NEST, and a hardware system can be used as a possible backend for an optimization algorithm that searches the neuron parameter space for the best parameter, which requires the emulation of certain of these parameter sets. The task to find such a best parameter set is a requirement for the Quantitative Single-Neuron Modeling Competition, but can also be applied to map neuron parameters of the AdEx neuron model, or any other in terms of firing patterns comparable neuron model, to the hardware system. Hence, this environment can be used to verify the neuron hardware implementation.

However, this approach only considers a specific set of reference data and is consequently not as general as the actual procedure of a parameter translation from neuron model parameters to hardware parameters. This parameter translation calibrates each neuron parameter step-by-step and uses a database to recall former calibrations (?Schwartz, 2011).

The Quantitative Single-Neuron Modeling Competition will be introduced in Section 3.1. With the motivation to participate at this competition, an optimization algorithm will be described in Section 3.2, which optimizes towards matching spike times between the competition data and the hardware emulation. The utilization of spike times takes advantage of the FACETS wafer-scale hardware system, because they are available digitally and therefore can be read out rapidly, which allows to outperform software simulations. In contrast, reading out membrane potentials of neurons requires an external oscilloscope sampling the membrane circuits, which significantly slows down the fitting procedure.

First results of experiments considering spike times only, as well as a combination of spike times and membrane potentials are shown in Section 3.4.2.

All experiments with the FACETS wafer-scale system were carried out in collaboration with Marc-Olivier Schwartz (Schwartz, 2011), who also contributed to hardware descriptions (parts of Section 1.4.2 and Section 3.3) in this thesis.

3.1 Quantitative Single-Neuron Modeling Competition

The *Quantitative Single-Neuron Modeling Competition* was launched by the INCF[11], in order to determine, how well neuron models can reproduce physiological measurements, which

[11] International Neuroinformatics Coordinating Facility

is described by Gerstner & Naud (2009). Biophysical models of the style published by Hodgkin & Huxley (1952) considering ion channels compete with simple LIF-like models that generate spikes as a threshold process. The biophysical models are preferred by electrophysiologists, but are not as amenable to mathematical analysis as the LIF-like models. The modeling competition allows the comparison of neuron models on the basis of spike times to bridge the gap between experimentalists and modelers.

The competition of 2009 consists of four independent challenges. Two are addressing the spike time prediction for a given current input into the soma of both regular pyramidal and fast spiking cells of cortical layer 5 (Jovilet et al., 2008). The third challenge is about spike timing in multi-compartment neuron models, while the fourth task involves a spike time prediction on the basis of presynaptic spike times only. The fourth task is most interesting for the FACETS wafer-scale hardware system, because the neuron input and output are spike trains, thereby appropriate for fast data transmission between the hardware system and the host computer. Nevertheless, this task does not only require to fit the neuron parameters, but also the additional synaptic parameters, and therefore complicates the parameter tuning. Furthermore, the spike transmission to and from the HICANN test setup was not yet supported when the hardware experiments for this thesis were performed, but once supported the HICANN test setup will be a promising tool for future neuron fitting experiments. For the time being, we focus on the first two challenges, where a current input is provided, which can be realized with the hardware system as described in Figure 25.

The current input is a combination of simple calibration current waveforms spanning 17.5 s and another 42.5 s of exponentially decaying random pulses with two different time constants that are randomly superposed as described in Naud (2011). The reference data was obtained by injecting this current into real biological neurons, from which the voltage traces were recorded with the current clamp method. The participants of the challenge are provided with the whole current input, but only with the first 38 s of the voltage traces. The spike times within the last 22 s is to be predicted by the neuron model, the parameters of which are based on the prior observed activity as is shown in Figure 20.

3.2 Methods for Neuron Parameter Tuning

In order to fit existing neuron models to noisy biological measurements, several approaches are introduced in the literature, e.g. statistical methods that smooth the noisy biological recordings (Huys & Paninski, 2009), the multiple objective optimization incorporating several error functions (Druckmann et al., 2007), and the automatic fitting of spike times (Rossant et al., 2010). Because Rossant et al. (2010) specifically address the Quantitative Single-Neuron Modeling Competition (Section 3.1) with its only spike time dependent score and focus on statistical optimization methods requiring fast simulation platforms, their approach matches the advantages of the FACETS wafer-scale system best and is taken as a starting point.

The current HICANN test setup, which is a prototype system of the FACETS wafer-scale hardware system, will be described in Section 3.3.3. However, this prototype system can not take advantage of the enormous computation power of the HICANN chip as described in Section 1.4.2, as the main bottlenecks are the provisional, low communication bandwidth via a JTAG[12]/USB interface and the time consuming membrane voltage readouts via an oscilloscope.

[12] Join Test Action Group

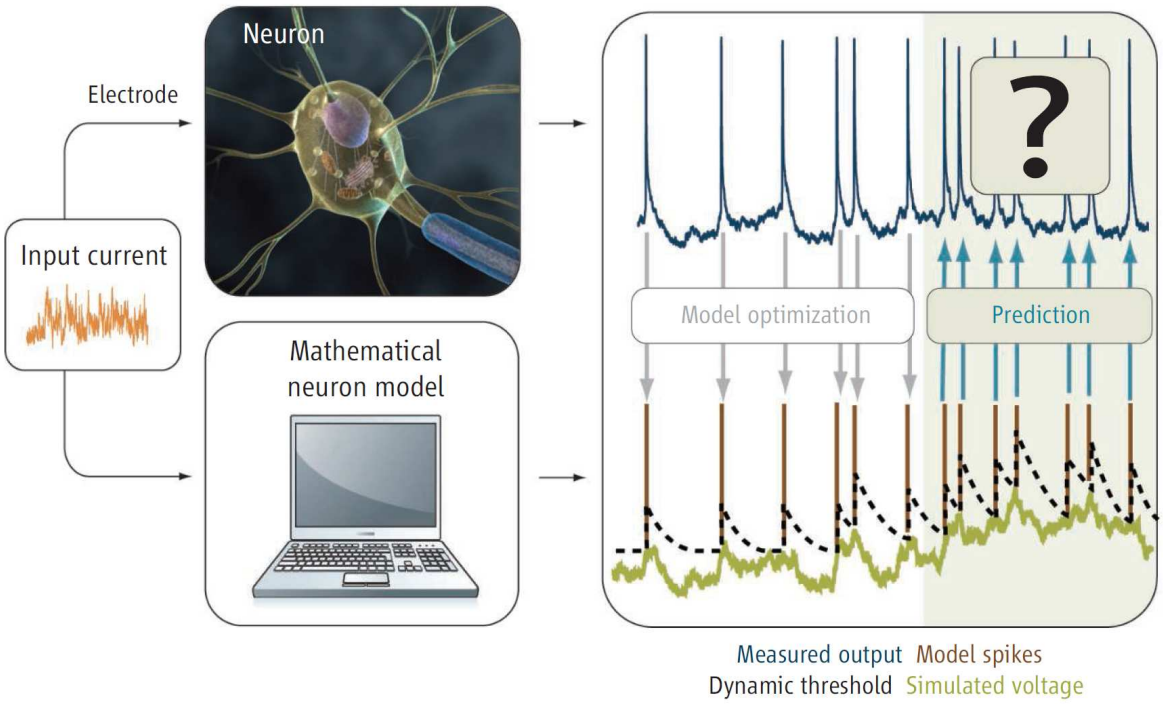


Figure 20: Illustration of the first two tasks of the Quantitative Single-Neuron Modeling Competition adopted from Gerstner & Naud (2009). For the first part of a predefined current input biological recordings are provided. Any participating neuron model shall be optimized to these recordings in order to predict the biological spike times resulting from the second part of the current input.

Therefore the statistical algorithm of Rossant et al. (2010) has to be reduced by means of the number of emulation runs. One approach is to decrease the parameter space to be searched by choosing coarsely appropriate initial parameter sets. This can be achieved by a beforehand linear regression to the provided competition data, which is adopted from Mensi et al. (2012). In order to apply a linear regression, the exponential term of the AdEx neuron model (Section 1.1.3) has to be neglected. Therefore the competition data is prepared by cutting out the spikes (Section 3.2.3). Once obtained, the linear regression results can be taken as an improved guess for the initial parameter sets.

If not only the spike times, but also the membrane potential should be predicted, the error function of the similarity between spike trains, called gamma score (Section 3.2.2), is extended by an error function that is based on the similarity between the hardware voltage trace and the biological membrane potential as described in Section 3.2.4. The workflow consisting of such a linear regression followed by a parameter optimization is presented in Section 3.4.1, where a software simulation with a periodic current pulses input is used as reference.

3.2.1 Linear Regression

A linear regression using the competition data requires a reduction of Equation (6) and (7) of the AdEx neuron model, as described in Section 1.1.3, to linear terms only. Neglecting the exponential term and substituting the sub-threshold adaptation w by a new parameter Ω, as defined in Equation (26), results in Equation (27):

$$w = a(\Omega - E_L), \tag{26}$$

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_L V + g_L E_L + I - a\Omega + aE_L - b\left(\sum_i \exp\left(-\frac{t-t_i}{\tau_w}\right)\right). \tag{27}$$

Because the exponential term is mainly dominant during spike generations, these spikes are cut out within the voltage and current data of the competition. To this end $\Delta t_p = 5\,\mathrm{ms}$ are cut out before and $\Delta t_{sw} = 2\,\mathrm{ms}$ after each spike that is extracted from the membrane potential as described in Section 3.2.3. The latter cutting of Δt_{sw} is done in order to cover the full spike width.

Applying the substitution of Equation (26) to the adaptation Equation (7) results in

$$\tau_w\frac{\mathrm{d}\Omega}{\mathrm{d}t} = V - \Omega, \tag{28}$$

which can be solved by a forward Euler integration. In addition the derivative of the membrane potential $\frac{dV}{dt}$ is determined by calculating the difference quotient between sequent data points of the membrane potential. On the other hand, the spike-triggered adaptation Θ is calculated by the spike times extracted from the competition data. Finally, all input vectors $\frac{dV}{dt}$, V, I, Ω and Θ are given and Equation (27) can be written as the following linear matrix equation:

$$C\frac{\mathrm{d}V}{\mathrm{d}t} + g_L V - I + a\Omega + b\Theta = (a + g_L)E_L \tag{29}$$

with $\Theta = \sum_i \exp\left(-\frac{t-t_i}{\tau_w}\right)$. If a certain value of τ_w is assumed, Equation (29) can be solved numerically by least squares methods (Oliphant, 2006). All other neuron parameters than τ_w in Equation (29) are determined by the results of this linear regression. In order to obtain

τ_w itself, this linear regression is repeated for each value of τ_w within a biologically plausible range. Once all neuron parameters C_m, E_L, g_L, a, b and τ_w are obtained, each of these parameters except τ_w is independently varied in both directions until the MSE between the competition membrane potential and the one that is simulated with the varied parameter, is 5 times as high as the MSE between the competition membrane potential and the one that is simulated with the linear regression parameter set. If the MSE does not increase enough while a certain parameter is decreased, this parameter is bound to a factor 10^{-3} of the original value.

Results of such a linear regression are shown in Section 3.4.1.

3.2.2 Similarity of Spike Trains

Jovilet et al. (2008) discuss several methods of measuring the similarity between two spike trains and are favoring the *coincidence factor* Γ (Kistler et al., 1997), because it is widespread and considers precise spike times instead of only firing rates.

A measurement spike train with N_m spikes is compared to a reference spike train with N_r spikes. Γ is the number of spike coincidences minus the coincidences occurring by chance relative to the total number of spikes in both spike trains:

$$\Gamma = \frac{N_c - \langle N_c \rangle}{\frac{1}{2}(N_m + N_r)} \frac{1}{N},$$

where N_c is the number of coincidences with a precision of $\Delta = \pm 2\,\mathrm{ms}$ and $\langle N_c \rangle = 2f\Delta N_r$ is the expected number of coincidences generated by a homogeneous Poisson process with the same firing rate f as the spike train of the measurement. Finally $N = 1 - 2f\Delta$ normalizes Γ to a maximum value of 1. If the measurement spike train is chosen randomly, $\Gamma \approx 0$. The lower bound is $\Gamma = -1$ for highly negatively correlated spike trains.

3.2.3 Spike Time Extraction

The extraction of spike times out of membrane potential traces is used, if no spike signals are provided. This is the case for biological recordings and membrane potential recordings of the HICANN test setup. On the other hand, the FACETS wafer-scale system will provide digital time stamps for each evoked spike. In contrast to the biological recordings, the AdEx neuron model does not reproduce the whole dynamics of a single spike, but rather describes a threshold process. Consequently, two different mechanisms to extract spike times are required. One to extract spike times from biological recordings, and another for hardware recordings of the AdEx neuron model, where the spikes have different peak values due to their steep, preceding membrane potentials and the limited time resolution of the recordings.

In case of biological recordings, the same threshold V_{peak} as in the AdEx model is applied. Whenever this threshold is crossed, a spike is registered, followed by a period of $\Delta t_{sw} = 2\,\mathrm{ms}$, which is chosen to exceed the spike width, and within which other spikes can not be detected.

If a hardware recording is provided, spikes can be detected by finding the abrupt drop of the membrane potential, as it is reset. For this purpose, the minimum and maximum value of the recorded membrane potential trace is determined. If the membrane potential falls from the upper 15% of this previously defined voltage range to the lower 15% within $\Delta t_r = 1.2\,\mathrm{ms}$ in biological time, a spike is registered. This method fails if the reset potential is above the resting potential. This can be the case while optimizing the neuron parameters. In order to

avoid the consideration of the resting potential, the middle, approximately 60% of the current pulse width, as applied as neuron input in later experiments as described in Section 3.3.4, is taken to determine the value of the minimum and maximum membrane potential. This part of the considered data is not influenced by the charging and discharging of the membrane potential caused by the edges of the current pulse, and consequently the minimum voltage is defined by the reset potential.

3.2.4 Membrane Potential Comparison

The membrane potential of the competition data and the hardware recordings are compared as follows. Because the recorded hardware voltages can not be transformed to their biological representation directly, a best fit between these two traces has to be obtained. To this end, four degrees of freedom have to be taken into account: the shift and stretch of the time axis as well as the ones for the voltage axis. The stretch of the time axis can be neglected, because the current input of the hardware system was measured to be temporally highly precise (not shown here). For the comparison between both voltage traces the hardware recordings are interpolated to the time grid of the competition data.

First, both membrane potential traces are cut at their ends to provide space for time shifts. Then the time shift is varied over a reasonable range, and for each time shift the best voltage shift and stretch are obtained by the least square method (Oliphant, 2006). The time shift with the least MSE, together with the according voltage transformations, define the best fit between both traces. To decrease the computation time, the initial values for the voltage stretch and shift are determined by the comparison of the voltage ranges, as well as the comparison of the minimum voltages after their stretching, respectively.

3.2.5 Particle Swarm Optimization

A general optimization problem is defined as follows. A function $\Gamma : A \to \mathbb{R}$ is given and $\vec{x}_0 \in A$ with $\Gamma(\vec{x}_0) \geqq \Gamma(\vec{x})$ for all $\vec{x} \in A$ is sought after. The fitting of neuron spike times is a non-linear optimization problem, meaning that $\Gamma(\vec{x})$ is not convex. So far there are no algorithms available that ensure the convergence to an optimal solution in finite time. Available classes of algorithms for approximating the optimal solution are for example genetic (Mitchell, 1998; Druckmann et al., 2007), evolutionary (Deb & Kalyanmoy, 2001) or statistical algorithms. Despite the usually better performance of evolutionary algorithms, we have chosen a statistical algorithm called particle swarm algorithm (Poli et al., 2007; Eberhart & Kennedy, 1995; Kennedy & Eberhart, 2002), because it has shown to be sufficient for the fitting of neuron spike times (Rossant et al., 2010).

The particle swarm optimization deploys a set of particles that evolve in the neuron parameter space towards optimal values. For this purpose, neurons are simulated using the parameter set of any particle i, represented as a vector $\vec{x}_i$, and the gamma factor (Section 3.2.2) is calculated. After all particles are processed, particle i is accelerated towards a mixture of its best previous location $\vec{x}_i^l(n)$ and the location $\vec{x}^g(n)$ of the global best particle, both of iteration n. A combination of stochastic and deterministic terms within the following update rule prevents particles from getting stuck in local maxima:

$$\vec{v}_i(n+1) = \omega\vec{v}_i(n) + c_l r_l[\vec{x}_i^l(n) - \vec{x}_i(n)] + c_g r_g[\vec{x}^g(n) - \vec{x}_i(n)], \tag{30}$$

$$\vec{x}_i(n+1) = \vec{x}_i(n) + \vec{v}_i(n+1), \tag{31}$$

where $\vec{x}_i(n)$ and $\vec{v}_i(n)$ are the position and velocity of particle i within parameter space. Consistent with Rossant et al. (2010), we assume $c_l = 0.1$, $c_g = 1.5$ and $\omega = 0.9$, whereas r_l as well as r_g are uniformly distributed random numbers over $[0, 1)$ independently drawn for each component of $\vec{x}_i$. The initial velocity of all particles is $\vec{v}_i(0) = 0$ and their positions $\vec{x}_i(0)$ are uniformly sampled within the user-specified parameter ranges. The boundary conditions for the neuron parameters are the limits of a pre-defined parameter space to stay within a physiologically plausible regime, e.g. positive parameter values. If a particle leaves this parameter space due to its velocity, it will be clipped to the boundary values. However, in contrast to the study of Rossant et al. (2010), original research articles about particle swarm optimizations (Poli et al., 2007) propose random initial positions and velocities.

In order to improve the convergence rate, the values of c_l and c_g can be optimized by another particle swarm optimization as suggested by (Meissner et al., 2006). This is not dependent on the simulation backend, because these parameters are specific for the type of optimization problem, here neuron spike time fitting. Once obtained by using a fast software simulation backend, the optimization parameters can then be applied to the mentioned, provisional HICANN test setup, as described in Section 3.3.3, which can not be operated as fast as software simulations yet. Another possible approach would be a general purpose particle swarm optimization reviewed by Poli et al. (2007), which does not require optimization parameters anymore, and has still to be tested.

The optimization algorithm itself can possibly be improved by replacing the globally best position with a grid of particles, within only the best position of the neighboring particles is considered for particle accelerations. These neighboring best positions lower the probability of missing a global maximum, but slow down the convergence rate (Poli et al., 2007). Another beneficial modification could be a mixture of evolutionary and self-adapting particle swarm optimization algorithms as proposed by Miranda & Fonseca (2002).

In the following experiments, the initial parameter ranges as well as the boundaries of the parameter space are chosen according to the experience gained while developing a parameter translation between biological and hardware neuron parameters that is still under development by Schwartz (2011). All parameter ranges are listed in Table 5, whereas the boundaries are given by the working point of the hardware system and are common for all voltages and currents, respectively.

3.3 Methods for Hardware Integration

In this part of the thesis, the hardware implementation of the AdEx neuron model (theoretically described in Section 1.1.3) is introduced in Section 3.3.1, and the FACETS wafer-scale hardware system is used as backend for a neuron fitting environment. However, the FACETS wafer-scale hardware system is still in the final stage of development, and is therefore replaced by a prototype HICANN chip that is assembled in a test setup. Because this so-called HICANN test setup does not provide all features of the wafer-scale system yet, two fitting environments are described for each hardware system separately in Section 3.3.2 and Section 3.3.3, respectively. Soon, the necessary software structure will be available, in order to replace the rather slow JTAG/USB interface of the HICANN test setup by a DNC[13]

[13] Digital Network Chip

Parameter	min. value	max. value	steps	min. bound	max. bound
E_L [mV]	500	700	3	300	1100
V_{peak} [mV]	1000	1000	0	300	1100
V_{reset} [mV]	700	700	0	300	1100
I_{gL} [nA]	100	600	3	50	2000
$I_{gLadapt}$ [nA]	100	300	3	50	2000
I_{fireb} [nA]	800	1800	3	50	2000
I_{radapt} [nA]	300	500	3	50	2000
V_{exp} [mV]	750	750	0	300	1100
I_{rexp} [nA]	250	1000	3	50	2000

Table 5: Initial neuron parameter ranges as well as boundaries of their parameter space to be explored by an optimization algorithm. Each parameter is initialized uniformly between the minimum and maximum value with the listed number of steps. For detailed descriptions of the hardware neuron parameters see Section 3.3.1.

(Section 3.3.3), which allows high communication bandwidths between the hardware system and the host computer.

3.3.1 Hardware Neuron Implementation

The neuron type integrated on the HICANN chip emulates the AdEx model introduced in Section 1.1.3. As described by Millner et al. (2010), each neuron circuit is controlled by 23 analog parameters generated by the floating gates array, including the AdEx parameters, the synaptic input parameters, and their bias parameters.

For the fitting experiment, only the 9 parameters controlling the AdEx model were varied, while the other 14 parameters (synaptic inputs parameters and bias parameters) were always fixed. The neuron capacitance C_m is not controlled by a quasi-continuous analog parameter, but can be chosen between two values: 2 pF or 400 fF. The latter value should only be used for the 10^5 acceleration mode, so the capacitance was always set to 2 pF for the following fitting experiments.

Four of the nine free hardware parameters are related to the leaky integrate-and-fire model: V_{reset}, V_{peak}, E_L and I_{gL}. The three voltages are directly linked to their AdEx model counterparts, whereas I_{gL} is controlling the membrane conductance g_L.

Another three hardware parameters control the adaptation terms: $I_{gLadapt}$ controls the sub-threshold adaptation parameter a, I_{radapt} controls the adaptation time constant τ_w, and I_{fireb} controls the spike-triggered adaptation parameter b.

Finally, two other parameters control the exponential term: V_{exp} sets the exponential threshold voltage V_{th}, and I_{rexp} controls the exponential slope factor Δ_{th}, but also influences V_{th}.

3.3.2 The Wafer-Scale Fitting Environment

The fitting environment utilizing the FACETS wafer-scale hardware system as described in Section 1.4.2 is summarized in Figure 21. The fitting procedure consists of a preparation of the competition data (described in Section 3.2.3), the linear regression (as introduced in Section 3.2.1), and the optimization method (Section 3.2.5), which runs emulations of

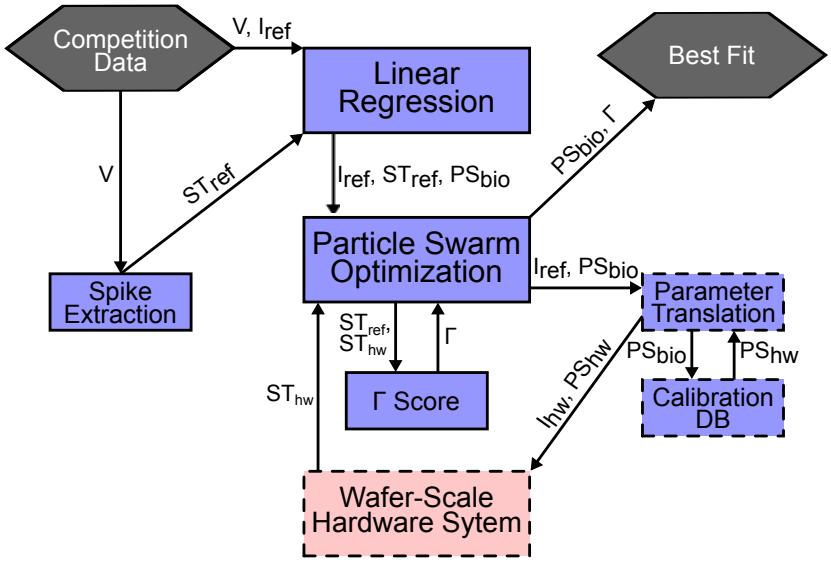


Figure 21: Schematic of the wafer-scale neuron fitting environment. The black hexagons depict the input and output of the fitting procedure. The dark purple boxes are software modules, whereas the bright red boxes denote hardware modules. Arrows indicate the flow of data between the modules. After extracting the spike times (ST) from the competition data, they are used together with the voltage (V) and current data (I_{ref}) to perform a linear regression. The results of the linear regression define the initial neuron parameter sets (PS) for the particle swarm optimization. Each emulation run initialized by the particle swarm optimization requires translations from the biological to the hardware parameter space. The resulting spike times are returned to the optimization algorithm, which calculates the gamma score Γ. After performing several iterations over all particles, the best position, here highest gamma score, in the parameter space is the best fit for the competition data set. Modules with dashed frames are not available or completed yet.

different neuron parameter sets on the FACETS wafer-scale hardware system. However, these sets of neuron parameters have to be translated to hardware parameter sets (Schwartz, 2011), before the hardware device can be configured with the resulting values. Because the parameter translation did not support the translation of all neuron parameters at the time of this study, the parameter optimization could not yet be performed on sets of biological parameters. Hence, the linear regression, which results in biological parameters, is needless at this point, too. Consequently, the experiment setup in Figure 21 could not be realized, even if the FACETS wafer-scale system backend would be replaced by a HICANN prototype, as introduced in Section 3.3.3. Nevertheless, important modules of the fitting environment - the optimization method and hardware operation - can be tested by the HICANN test setup described in Section 3.3.3.

Once the parameter translation is completed and the wafer-scale system is present, they can fill the only two gaps of the existing and tested fitting environment.

3.3.3 The HICANN Prototype Fitting Environment

A HICANN test setup, as shown in Figure 22, was assembled as follows: The hardware environment is composed of the so-called iBoard, where the HICANN chip is plugged in, a Xilinx Virtex-5 LXT FPGA[14] board connected to a computer via a JTAG/USB interface, and a LeCroy WaveMaster oscilloscope. The hardware environment receives parameters from the optimization algorithm, and must send back the spike times of the hardware neuron, as shown in Figure 23.

As mentioned in Section 3.3.1, the hardware neuron behavior is controlled by 23 analog parameters generated by a floating gates array located on the HICANN chip. To control this array, a Python interface has been developed to easily interface with the fitting software, change the different parameters and program the floating gate array. When receiving parameter sets from the particle swarm optimization algorithm (Section 3.2.5), the Python floating gates interface first converts the currents and voltages to digital 10-bit values that can be written into the floating gates array. Then, the interface generates an XML file with all the parameter information, which will be processed by a test mode written in C++. The intensity of the current stimulus (Section 3.3.4) is also defined in the test mode. Finally, this test mode is called to program the floating gates array, set the current stimulus, and switch the analog output to the desired neuron circuit.

At the other end of the hardware environment, the spike times must be returned as a Python array to the fitting algorithm. For this experiment, one of the HICANN analog outputs was used to record the membrane potential. After the programming of the floating gates array, the scope, which is also controlled by a Python interface, is adjusted to observe the membrane potential with an appropriate resolution. The scope resolution was set to 50.000 samples, corresponding to a time step of 0.02 ms in biological time. The Python script controlling the oscilloscope is automatically setting the voltage per division and the voltage offset, such that the voltage trace occupies all the scope display. The script also adjusts the trigger level of the oscilloscope to ensure a correct and stable readout. Once the oscilloscope is set correctly, the data is acquired and retrieved as a two-dimensional array into the Python software.

Then, the relevant part of the voltage trace is extracted. Indeed, as the current pulses described in Section 3.3.4 are 33 μs long, three of them can be seen on the oscilloscope (with 10 μs/division). Therefore, the oscilloscope voltage trace is cut within the Python script to keep only one of these pulses. Finally, a spike-detection algorithm (Section 3.2.3) is run on the cut voltage trace, before returning the spikes times to the particle swarm optimization algorithm.

Each particle took around 16 seconds in total to be processed: 6 seconds to erase and rewrite the whole floating gates array, 10 seconds to adjust the scope ranges and retrieve the data, and far less than one second to process the data, calculate the gamma score (Section 3.2.2), and run the algorithm. If the voltage traces should also be fitted (Section 3.2.4), the runtime is elongated by another 6 seconds. This voltage trace fitting can be accelerated drastically by executing a fitting algorithm implemented in a compiled programming language, e.g. C/C++, instead of executing the rather slow script language Python. Re-writing the floating gates can also be reduced to a small fraction, if the JTAG/USB interface is replaced by a DNC that provides a high communication bandwidth. If the membrane potential recordings should be preserved, they can be accelerated by leaving out the adjustment of the

[14] Field Programmable Gate Array

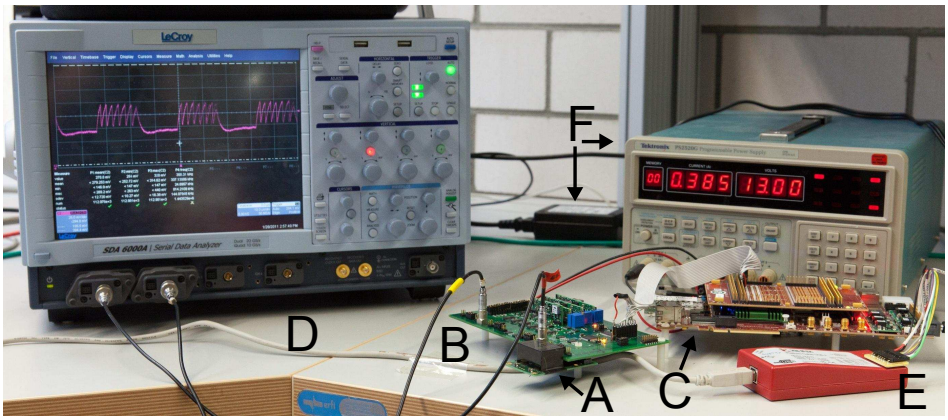


Figure 22: Photograph of the HICANN test setup. The HICANN chip (A), which is covered by a black box, is plugged into the iBoard (B) that is connected to the FPGA board (C). The oscilloscope (D) is connected to a host computer and records membrane potentials from the HICANN chip via the iBoard. The neuron parameters are configured via the JTAG/USB interface (E) that serves as a link between the FPGA board and the host computer. The power supplies (F) provide the power for both boards and the chip.

voltage scale, which is resulting in less resolution voltage recordings. Combining all of these improvements can decrease the emulation runtime to below 5 s, and even below 1 s, if only spike times are recorded.

3.3.4 Hardware Current Input

A neuron within the HICANN chip can receive two different types of current input: Synaptic currents and a configurable background current, the latter of which is more suitable for the neuron competition, because no synaptic parameters are involved. Because the FACETS wafer-scale system is optimized for spike processing, the background current was implemented for debugging purposes only, and is therefore implemented in a simple and resource-saving manner. The background current follows a loop of 129 current buffer values, as described in Section 1.4.2, with a frequency of $f_l = 3.9\,\text{MHz}$ to $62.5\,\text{MHz}$ in hardware time scale, which corresponds to a bin size of $h_{bin} = 2.56\,\text{ms}$ to $0.16\,\text{ms}$ in biological time, whereas the competition current data is provided with a temporal resolution of $h = 0.1\,\text{ms}$. Consequently f_l should be chosen as high as possible, to preserve as much detail of the competition data as possible. On the other hand, the current buffer has to be re-configured each loop cycle, in order to provide the full length of the competition current data. This requires a precise timing of the host computer or an FPGA, and is hence less difficult to realize for low frequencies f_l.

The effect of a necessary data reduction, in order to transform the competition current data to the bin size of the hardware system, was analyzed by Alexander Kononov (Kononov, 2010) using preparative software simulations. Although the AdEx neuron model (Section 1.1.3) was extended by a second adaptation term in this study, Figure 24 gives a qualitative impression about such a data reduction.

The implementation of a periodic re-configuration of the current buffer exceeded the time frame of this thesis. Hence, periodic current pulses of $I = 800\,\text{pA}$ in biological units are used as input for the hardware neurons, where a single pulse spans one half of the current

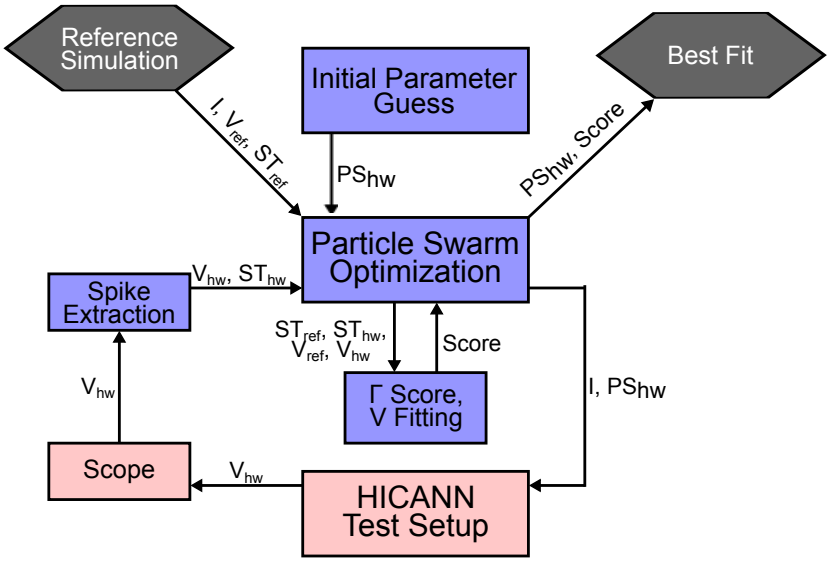


Figure 23: Schematic of the HICANN test setup incorporated into the neuron fitting environment. In comparison to the procedure of the FACETS wafer-scale hardware system in Figure 21, the competition data is replaced by a software reference simulation. Furthermore, the initial parameter sets are defined by an educated guess instead of a linear regression. Additionally, the optimization algorithm is operating directly on the hardware parameters, and consequently the parameter translation is not required anymore. Because no DNC was present in this first prototype hardware setup, no digital spike time stamps are returned by the HICANN test setup, but a membrane potential is recorded with an oscilloscope, from which the spike times can be extracted.

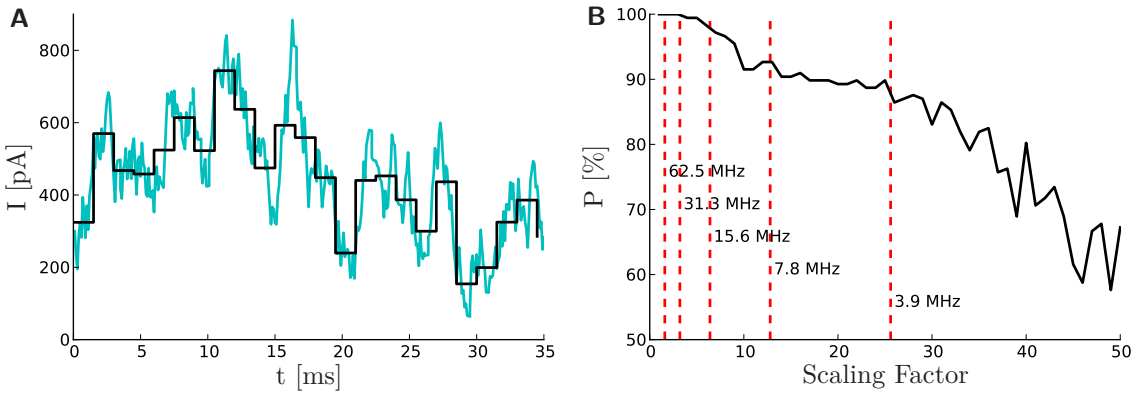


Figure 24: **(A)** A section of the competition current input (cyan) in comparison to a reduced current input that is defined by the mean over 15 consecutive competition current values (black). This reduced current input corresponds approximately to a frequency of 31.3 MHz in (B). **(B)** Percentage P of matching spike times between software simulations using the competition current and reduced current input. The reduction level of a current input is determined by bin size that is by a *scaling factor* larger than the competition bin size $h = 0.1\,\text{ms}$.

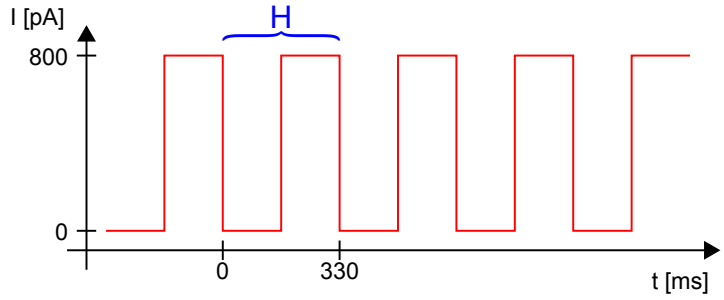


Figure 25: The periodic input current that is applied to hardware neurons. H denotes the current data stored in the current buffer, which is replayed in a loop.

buffer, as illustrated in Figure 25. In Section 3.4 results are shown for software simulations and hardware simulations using current pulses as input.

Because the oscilloscope is triggered by the rise of the membrane potential caused by the rising edge of the current pulse, a time shift is possibly necessary to adjust the recordings to the reference.

This this periodically repeating stimulation scheme, the competition data can not be used anymore as reference. Consequently, a NEST-based software simulation is used as reference with AdEx parameters as listed in Table 6. These are the standard parameters published in Brette & Gerstner (2005), except of b, which was chosen approximately half as high ($b = 35\,\text{pA}$ compared to $80.5\,\text{pA}$), because transistor level simulations of the analog circuits indicated that b should be rather small to stay within the dynamic range of the circuit design (Schwartz, 2011).

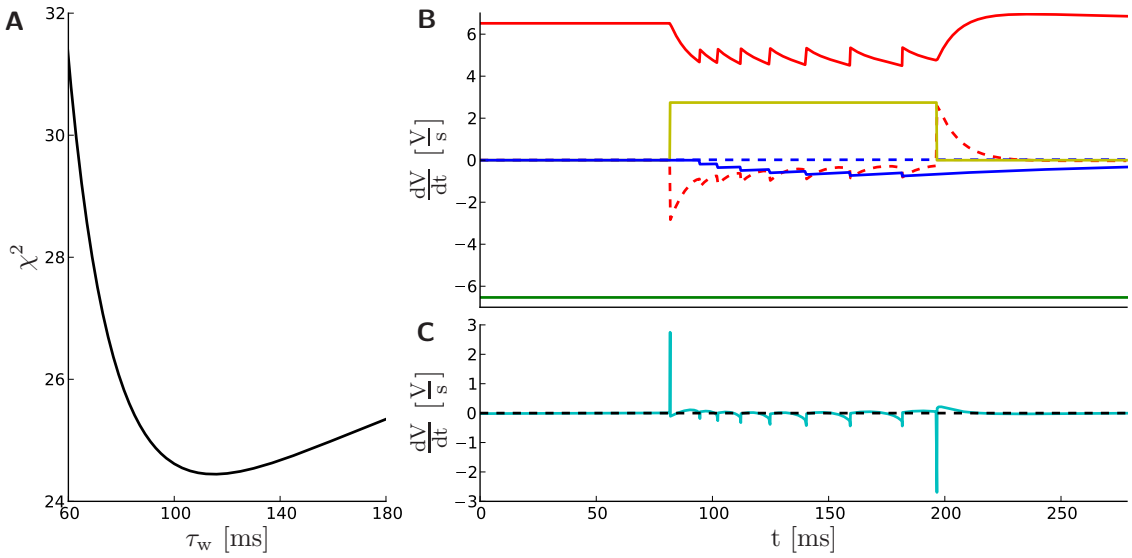


Figure 26: **(A)** The residuum χ^2 of linear regressions on the reference data set is plotted against the adaptation time constant τ_w. The residuum is minimal for $\tau_w = 115.0\,\text{ms}$, for which the decomposition in single terms is shown in (B). **(B)** Showing all single terms of the linear regression, as denoted in Equation (29), for a current pulse (yellow) using $\tau_w = 115.0\,\text{ms}$: the terms including the membrane potential (solid red), the derivative of the membrane potential (dashed red), the current pulse (yellow), the sub-threshold (dashed blue) and spike-triggered (solid blue) adaptation as well as the constant term (green). The sum of these terms is plotted in (C). **(C)** The sum (cyan) of all terms plotted in (B). The comparison with the dashed baseline in terms of a MSE results in the residuum, as plotted over τ_w in (A).

3.4 Experiments and Results

This section is subdivided into two parts: First, a linear regression is performed with the reference simulation data set, as described in Section 3.2.1. Then the results of the linear regression together with all other AdEx neuron parameters (Table 6) are optimized via a particle swarm algorithm, as described in Section 3.2.5 using the NEST software simulator as backend. The results verify that the two-step method of a linear regression to reduce the parameter space, in which the initial parameter sets of the statistical optimization method is chosen, solves the optimization problem satisfyingly.

Second, the HICANN test setup described in Section 3.3.3 is used as backend for a particle swarm algorithm that directly optimizes the hardware parameters instead of their biological counterparts. The optimization of hardware parameters is necessary, because the automated parameter translation between biological and hardware parameters, as used in Section 3.3.2, in not completed yet. Once a parameter translation is provided, the linear regression can be combined with the optimization method that utilizes the HICANN test setup, or later the FACETS wafer-scale system as backend.

Parameter	Reference	Linear Regression	Parameter Range	Initial Values	PSO
C_m [pF]	281.0	291.6	181.0 to 426.9	2	336.4
g_L [nS]	30.0	26.9	22.2 to 35.8	2	32.0
E_L [mV]	-70.6	-70.3	-70.3 to -77.3	2	-70.0
V_{peak} [mV]	0.0		0.0 to 0.0	1	0.0
V_{reset} [mV]	-60.0		-50.0 to -70.0	4	-60.3
a [nS]	4.0	0.2	0.0 to 2.1	2	9.6
b [pA]	35.0	55.6	31.4 to 89.5	2	39.2
τ_w [ms]	144.0	115.0	100.0 to 200.0	4	119.8
V_{th} [mV]	-50.4		-40.0 to -60.0	4	-52.8
Δ_T [mV]	2.0		0.5 to 5.0	4	3.6

Table 6: Results of the preparative software fitting experiment. From left to right are listed: the reference neuron parameters adapted from Brette & Gerstner (2005), the results of a linear regression as described in Section 3.2.1 without the non-linear exponential term and threshold potentials, the parameter range and number of equidistant values within this range used as initial parameter sets for a particle swarm optimization, and finally, the results obtained from the particle swarm optimization (PSO) that is described in Section 3.2.5.

3.4.1 Software Simulator Experiments

For the following software experiments, a single current pulse was used instead of a periodic series of pulses, as applied during the hardware experiments in Section 3.4.2.

The result of a linear regression (described in Section 3.2.1) on the reference current, voltage and spike data is listed in Table 6 as well as visualized in Figure 26. The fact that the residuum vanishes least near spikes, as can be seen as many small peaks within the current pulse in Figure 26C, suggests a lasting influence of the exponential term and non-linearities within the current pulse and membrane potential, although the spikes are cut out.

The resulting neuron parameters were used to define the initial parameter sets for the particle swarm optimization, which are listed in Table 6 and further described in Section 3.2.1. The parameter ranges obtained by the linear regression, except the one for τ_w, are split into two equidistant initial values, whereas all other parameter ranges are split into four equidistant values, and only V_{peak} has a fixed value. The parameter range of τ_w was chosen manually, because its automatic determination is not implemented yet. The particle swarm optimization (Section 3.2.5) of these 8192 particles resulted, after 100 iterations, in a best position in neuron parameter space that is plotted in Figure 27. The spike times using this best fit parameter set, and the reference simulation are identical in terms of the gamma factor (Section 3.2.2), their membrane potential fits quite well and the neuron parameters are almost identical as listed in Table 6.

Nevertheless, these results should be compared to the case, where the parameters obtained by the linear regression are fixed, and only the other ones are varied. Such an optimization strategy would require far less particles and therefore less software simulation runs. However, the results of the linear regression are most likely not the best fit neuron parameters, but more a rough estimation.

The current simulation runtime with NEST for one current pulse, which are 300 ms in bio-

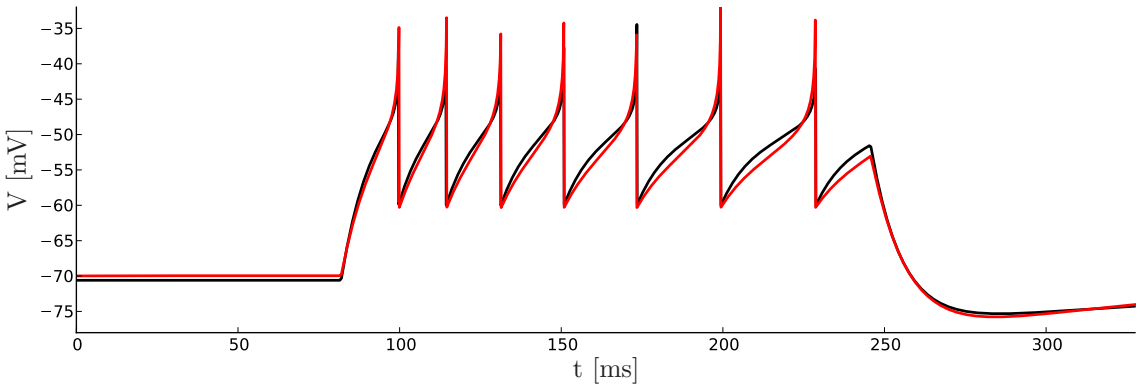


Figure 27: The resulting membrane potential of the best fit neuron parameter set obtained by the particle swarm optimization (red) with NEST as backend, and the reference NEST simulation (black).

logical time, is 13 ms on one core of an AMD Phenom II X4 955 processor. Consequently, the optimization run with ≈ 8000 particles and 100 iterations requires $\approx 3\,h$ of NEST simulation runtime.

3.4.2 Hardware Experiments

In this section the HICANN test setup described in Section 3.3.3 was utilized as backend for the optimization algorithm introduced in Section 3.2.5. First, only spike times were fitted, but due to the simplified task of periodic current pulses the membrane potential was fitted as well. Different to the single current pulse for the software simulations in Section 3.4.1, periodic pulses are used as neuron input for the hardware experiments, because this simplifies the experiment setup.

Spike time fitting After only three iterations with 729 particles, as denoted in Table 5, the particle swarm optimization algorithm resulted in several particles with a maximum score of $\Gamma = 1$. The measured membrane potential of an arbitrarily chosen particle is plotted in Figure 28C, and another particle with a different set of parameters is plotted in Figure 28E. The membrane potentials of both particles differ significantly from each other, although their spike times are identical. This fact suggests that the task with periodic current pulse input is oversimplified, and therefore has several different parameter sets as solution.

In addition, the applied optimization algorithm is not designed to deal with multiple best position particles. Currently, the first occurrence of the best score is taken as a global attractor within parameter space, and hence suppresses all other best particles.

One approach to distinguish between these best particles, is fitting the membrane potential as well. So far, this does not limit the performance significantly, because the membrane potential has to be recorded anyway, in order to extract the spike times from it. A fitting of the membrane potential only was not satisfactory, because the spike generation does not change the membrane potential enough to be considered. Consequently, the neuron parameters were optimized towards a membrane potential without spikes, but it reproduced the charging and discharging behavior at the edges of the current pulse very well.

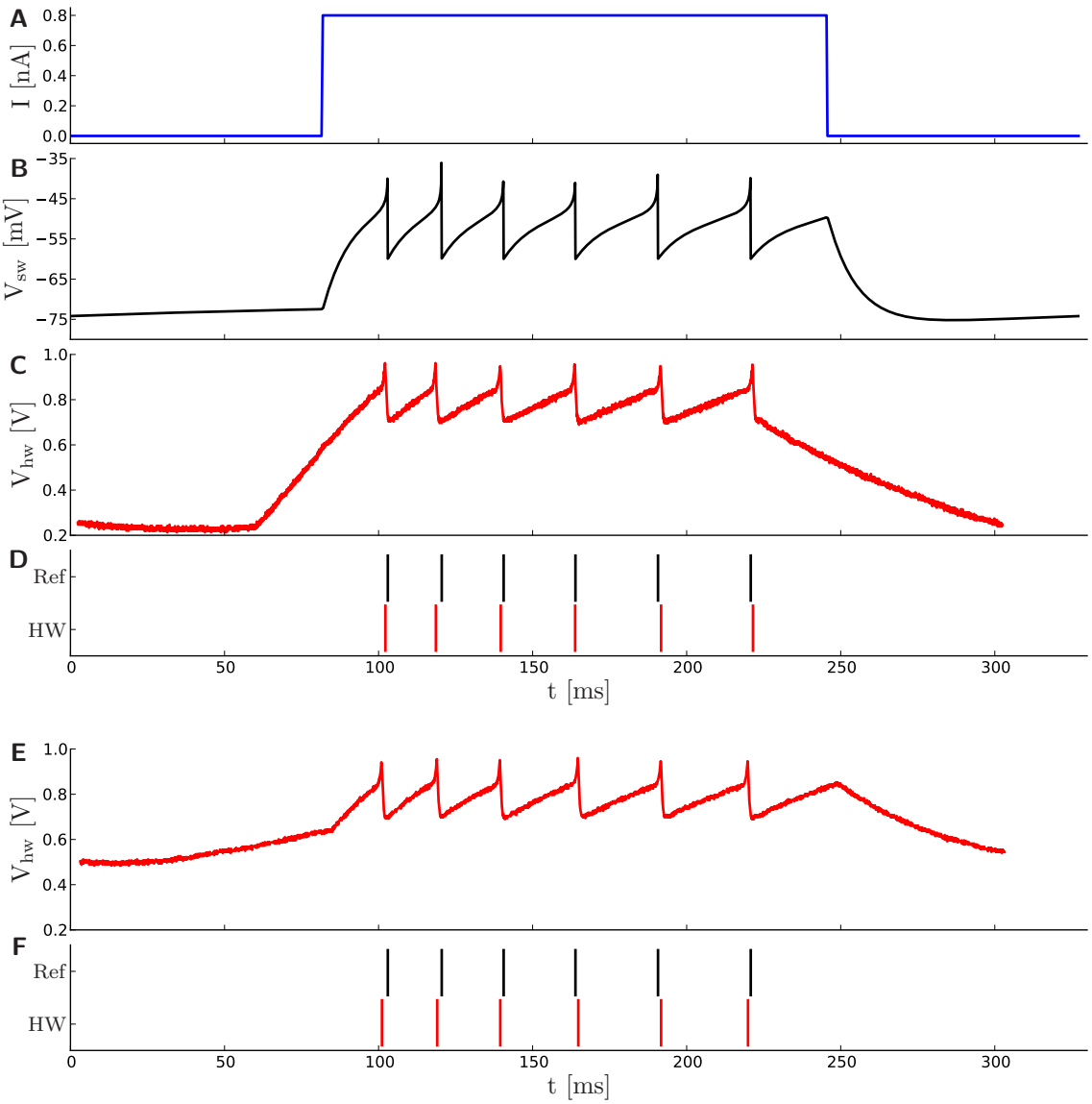


Figure 28: Results of the particle swarm optimization with the HICANN test setup as backend. **(A)** One current pulse within the periodic neuron input. **(B)** Membrane potential of the NEST-based reference simulation. **(C)** Hardware measurement of the membrane potential with hardware voltage scale, but biological time scale. **(D)** Comparison of the spike times between the reference simulation and the hardware recording. All spikes fit within $\pm 2\,\mathrm{ms}$ resulting in a gamma score of $\Gamma = 1$. **(E)** Measurement of another particle with best score $\Gamma = 1$, as shown in **(F)**.

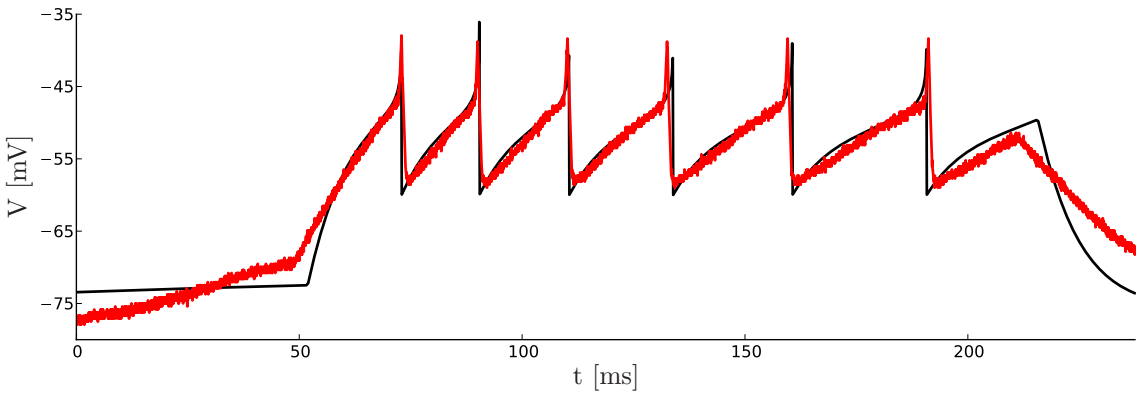


Figure 29: Hardware measurement of the best particle (red) in comparison with the reference software membrane potential (black) resulting in a score of $\Omega = 1.00051$.

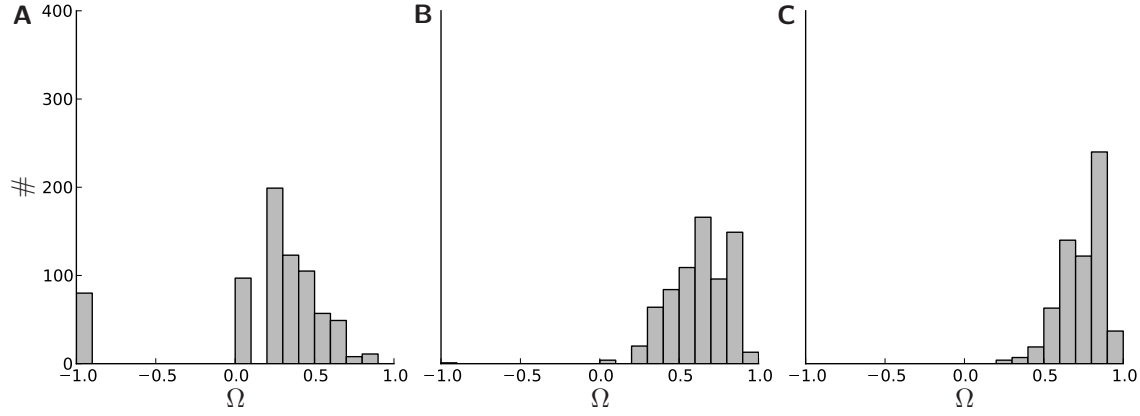


Figure 30: The distribution of the best Ω scores of each particle after iteration 1 (A), 5 (B) and 10 (C), respectively. A score of $\Omega = -1$ indicates measurements without spikes.

Spike time and membrane potential fitting In addition to only spike time fitting as introduced in the last paragraph, in the following the membrane potential is fitted, too. The inverse of the MSE ϵ between the reference and measured membrane potential is weighted with a factor $c = 10^{-5}$ and added to the score Γ to obtain the overall score Ω:

$$\Omega = c\frac{1}{\epsilon} + \Gamma. \tag{32}$$

This score Ω ensures that the membrane potential does not outweigh the matching spike times, but distinguishes between best spike time fittings. The optimization result after 10 iterations with again 729 particles is shown in Figure 29, where the additional consideration of the membrane potential apart from the spike times can be observed.

The evolution of the best Ω score of each particle over iterations can be seen in Figure 30. The optimization parameters chosen in Section 3.2.5 ensure that the parameter space is continuously explored, as indicated by the tail after 10 iterations in Figure 30C. This can be explained in detail by the movement of a single particle, which is accelerated towards the globally best position, but later on moves even further due to remaining velocity. Conse-

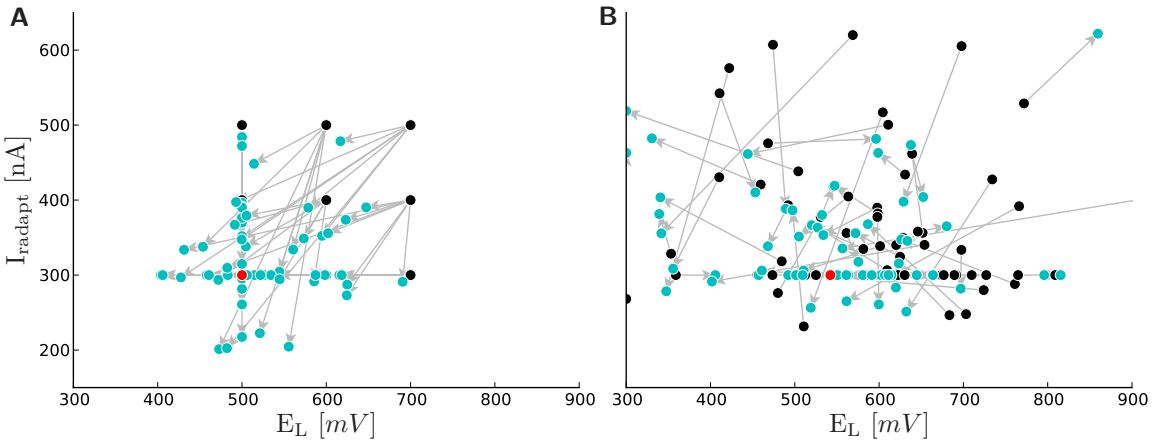


Figure 31: Particle movement in the parameter plane of E_L and I_{radapt}. Black particles denote the position before and cyan particles after the iteration. The best position before the movement is depicted in red. **(A)** The first iteration. Note the equidistant initial positions. **(B)** The seventh iteration. Note the exploration of the parameter space, which is explained in detail in the text.

quently all particles are oscillating around the globally best position. An example of these particle movements are shown in Figure 31.

There, many particles gather at $I_{radapt} = 300\,\text{nA}$, because this is the value for each best position up to this iteration, and all other particles with the same initial parameter are stuck to it. This effect can be avoided by a at least partly random initialization of the particle positions.

The parameter distributions of the particle best positions after 10 iterations are shown in Figure 32. The distributions for E_L, $I_{gLadapt}$ and I_{rexp} show Gaussian-like distributions around the best particle value. On the other hand, the distributions of I_{gL} and I_{radapt} have high peaks near one of the initial values, because the best particles within the optimization did not leave this initial value significantly. However, the distribution of the spike-triggered adaptation hardware parameter I_{fireb} is concentrated at the lower boundary of allowed currents. In combination with the fact that we lowered the reference spike-triggered adaptation b, this low value of I_{fireb} suggests that the spike-triggered adaptation was compensated by the sub-threshold adaptation a (corresponds to $I_{gLadapt}$ in hardware), and the adaptation time constant τ_w (corresponds to I_{radapt} in hardware). For detailed descriptions of the hardware neuron parameters see Section 3.3.1.

3.5 Conclusion drawn from this Section

As shown in Section 3.4.2, the current HICANN test setup described in Section 3.3.3 can be utilized as backend for a statistical optimization method introduced in Section 3.2.5. The simple parameter fitting task in case of a periodic current pulse allowed us to setup an automatic fitting environment (Section 3.3.3) that results in satisfactory hardware neuron parameter sets as shown in Figure 29, although the emulation runtime of the HICANN test setup is many times higher than that of the future FACETS wafer-scale hardware system.

As a next step, the existing fitting environment can be easily extended by different periodic

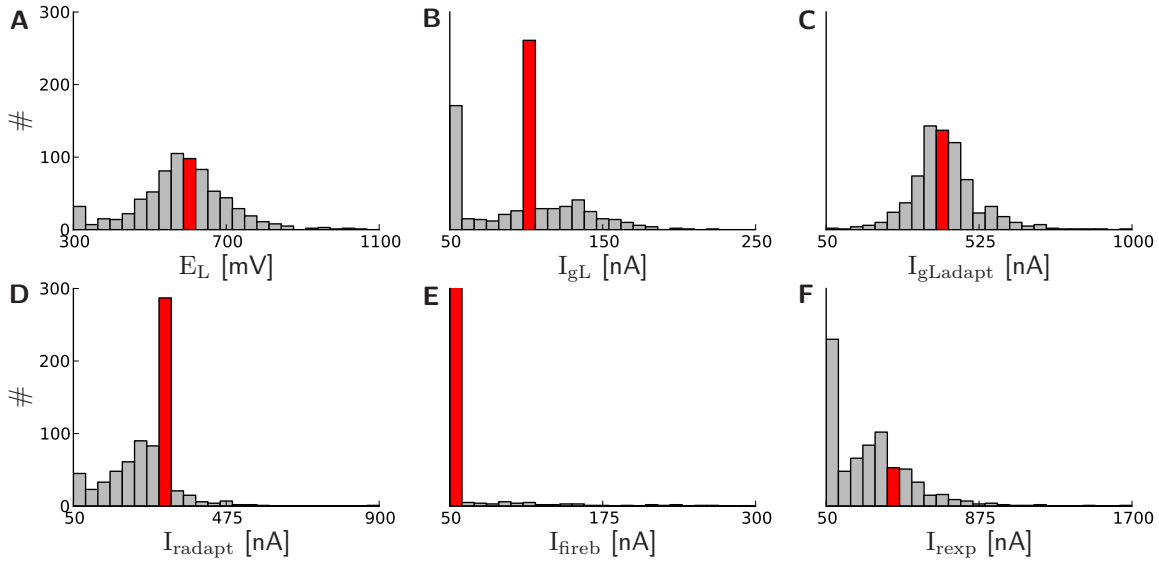


Figure 32: Parameter distributions of all particle best positions after 10 iterations. The bin containing the parameter of the globally best particle is indicated in red.

stimuli. If only spike times are fitted and the solutions are spread over the parameter space, as shown in Figure 28, a large variety of stimuli would be able to narrow down the wide range of parameters. The replacement of the membrane potential fitting with spike based fitting methods is a necessary step to exploit the very high temporal acceleration of the future FACETS wafer-scale hardware system, because spike data can be generated on-chip and is rapidly transmitted to the host computer. Later, the variety of periodic currents can be replaced by continuous current traces, as e.g. provided by the Quantitative Single-Neuron Modeling Competition (Figure 20), periodically written to the current buffer by an on-board FPGA.

In addition, the optimization algorithm can be improved as suggested in Section 3.2.5: Simply varying the initial parameter sets by small random variations would most likely improve the exploration of parameter space and avoid the accumulation of many particles at a certain parameter value, as shown in Figure 32.

Another reasonable modification would be a parallel fitting of as many neurons as possible. This would provide a tool to predict spike times statistically, which is the task of the neuron competition in Section 3.1. Currently a maximum number of four neurons can be fitted in parallel, because only four current sources, as further explained in Section 3.3.4, are implemented within one HICANN. Replacing the current input with spike input of hardware neurons would allow to stimulate and measure all neurons at once, with the only limit being the communication bandwidth. Additionally, different reference spike trains can be used, and hence a large number of neurons in the FACETS wafer-scale system can be configured at once to reference neurons that generate similar firing patterns compared the AdEx model (Section 1.1.3). Nevertheless, using spike trains as stimulation requires the additional fitting of all synaptic parameters.

On the host computer, the major computational task, except for the provisional membrane potential fitting, is the calculation of the factor Γ, as described in Section 3.2.2, which has to

be determined independently between neurons and hence can be distributed onto computer clusters. As an alternative for calibrating a whole wafer-scale system at once, the gamma factor calculation could be performed on a FPGA, such that only the resulting gamma factor has to be returned to the host computer. A hardware system equipped like this would extend the current hardware system to an even more powerful neuron fitting emulation backend.

4 Achievements and Outlook

The two main building blocks of the FACETS wafer-scale system - its plastic synapses and its neurons - have been studied with preparative software simulations and hardware experiments. First of all, they have been compared to their counterparts in software simulators, e.g. NEST, because an important early application that will help to establish the novel FACETS wafer-scale hardware system as an emulation backend is the implementation of neural networks that are already successfully realized in software simulations. Later on, the workflow step via theoretical models and software simulations can optionally be left out, and neural network architectures might be adapted from biology, or found with optimization methods directly, thereby exploiting the hardware-specific benefits.

Regarding plastic synapses in the FACETS hardware, their level of detail is significantly reduced compared to software implementations. Synaptic distributions are broadened, if only a 4-bit weight resolution is allowed, as exemplarily shown in Figure 13. Still, certain network tasks, as e.g. the synchrony detection presented in Section 2.4.3, can be solved adequately. The global update mechanism, common resets as well as quite high fluctuations within the production of chips are other drawbacks that have been compared to software simulations.

The historical question about the hardware design, whether a 4-bit weight resolution is enough, could be answered in context of the other hardware restrictions as well as hardware measurements. As long as the production variance of the synapse circuits is not decreased, which affects the performance of the synapses on a fundamental level, a weight resolution of 4 bits has been found to be sufficient for detecting synchrony within a neural network.

Another bottleneck, the common reset of both the causal and acausal accumulation circuits, was identified. As a consequence drawn from the presented work, the hardware layout was already modified to compensate the common reset by combining two synapse rows, one for the causal and the other for the acausal correlations. In future design studies two reset lines for each synapse should be considered.

However, the influence of the mentioned limitations of hardware synapses on other network architectures as well as the loss of dynamics on small time scales, as discussed in Section 2.5, have still to be analyzed. This can be done by preserving the method of preparative software simulations introduced in this thesis, because the hardware limitations were implemented into a NEST-based software model, which can be easily incorporated into large-scale networks. The existing network benchmark should be extended to other general network concepts, in order to further analyze the hardware synapse design. If only single synapse dynamics should be analyzed in detail, a transistor level simulation would be more appropriate, but this is inapplicable for network simulations.

Regarding the second main building block, the neuron implementation is capable of reproducing a large variety of firing patterns, which can be obtained by using the neuron fitting environment presented in this thesis (Section 3.5), or as shown by Schwartz (2011). In contrast to the low resolution of synaptic weights, all individual hardware neuron parameters (introduced in Section 3.3.1), except V_{reset}, can be set independently for each neuron with a 10-bit precision.

The neuron fitting environment presented in this study has different fields of application. First, it was motivated by the possible participation in the Quantitative Single-Neuron Modeling Competition utilizing the FACETS wafer-scale hardware system. Once a continuous current input is established, and the communication bandwidth to the hardware system is increased, the competition data can be fitted automatically, and spike times stimulated by

the ongoing current input can be predicted.

Second, the neuron fitting environment was extended for the optional use of a software simulation as reference, instead of biological data. A possible application could be the mapping of neuron parameters to the hardware system. Although the neuron fitting is time consuming compared the a step-by-step neuron calibration, which is currently developed by Schwartz (2011), it can be beneficial to fit the hardware neurons to arbitrary neuron models. Of course this is only possible, if the complexity of firing patterns in the reference neuron model and the AdEx neuron model is comparable.

Third, the hardware backend of the fitting algorithm can be replaced by a software simulator, e.g. NEST, that provides the AdEx neuron model, in order to fit the AdEx model to an arbitrary reference neuron model without the need of hardware resources. Once the AdEx neuron parameters are obtained, they can be translated to the hardware system by using the previously mentioned calibration database.

Comparing the detail level of the neuron and synapse implementation in the FACETS wafer-scale hardware system shows that, in contrast to software simulators, neurons are designed in much more detail than synapses. This fact is mainly determined by the large number of synapses, which is several orders of magnitudes greater than the number of neurons. The results of this thesis may raise the question, whether more resources should be spent on synapses, which are assumed to be the fundamental substrate for learning. But when discussing this point, it has to be mentioned that, compared to other large-scale mixed-signal neuromorphic systems, as e.g. by Merolla & Boahen (2006), the FACETS wafer-scale hardware system already provides a highly configurable array of synapses .

If this uniquely large size of the current wafer-scale system should be preserved, the synapse layout can not be increased with the intention to reduce the hardware restrictions. An appropriate balance between the number and size of synapses has to be provided to claim the position of a large-scale system, and on the other hand not to lose the interest of the modelling community by restricting the variety of applicable network models too much. The unique size of the system is essential to boost the development of new computation paradigms and subsequently support their high-speed execution. One solution to realize many detailed synapses is migrating from a 180 nm to a 65 nm fabrication process, which decreases the size of digital circuits and is currently under development within the FACETS and BrainScaleS project.

The synapse model implemented within this study will be available for public use within one of the next NEST releases. This allows the extension of the benchmark network that is introduced within this thesis (Section 2.3.3) to other existing or novel neural network models without much effort. The collective experiences of applying this synapse model to other network architectures are an important criterion, before fundamental modifications of the hardware design should be considered. Distributed to the large community of the FACETS and BrainScaleS project as well as beyond, the hardware synapse model allows other research groups to benchmark their network models with the intention to emulate their network models on the FACETS wafer-scale hardware system. Such benchmark results represent an important feedback that is essential in order to adapt the hardware design to the needs of common network models.

Besides only mimicking theoretical models, more attempts could be made to copy the strategies of biological architectures directly into silicon. In the thesis at hand, a new STDP model was developed (Section 2.1) that employed the existing hardware design by taking biological measurements of Sjöström et al. (2001) into account. While being unsuccessful in

terms of generalization to arbitrary pre- and postsynaptic spike rates, this line of research should nevertheless be pursued further.

Not only local plasticity is supposed to play an important role in learning processes, but also global plasticity. A prominent example for global plasticity is reward-related learning, e.g. modulated through neuromodulators (Gu, 2002; Reynolds & Wickens, 2002). Currently, modifications within the hardware design to support such global plasticity rules are being investigated.

Although software and prototype-based preparative studies are an important tool to analyze, verify and improve hardware layouts, they have to be extended to hardware experiments with the target systems. Past utilizations of hardware systems showed that hands-on experiments are indispensable to eventually identify context-dependent bottlenecks and non-obvious design problems. And often not only the basic concept, but also the final implementation in silico are sources of further distortions.

References

Amit, D., & Fusi, S. (1994). Learning in neural networks with material synapses. *Neural Comput. 6*(5), 957–982.

Artola, A., Bröcher, S., & Singer, W. (1990). Different voltage dependent thresholds for inducing long-term depression and long-term potentiation in slices of rat visual cortex. *Nature 347*, 69–72.

Bi, G., & Poo, M. (1998). Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type. *J. Neurosci. 18*, 10464–10472.

Bill, J., Schuch, K., Brüderle, D., Schemmel, J., Maass, W., & Meier, K. (2010). Compensating inhomogeneities of neuromorphic VLSI devices via short-term synaptic plasticity. *Front. Comput. Neurosci. 4*(129).

Brette, R., & Gerstner, W. (2005). Adaptive exponential integrate-and-fire model as an effective description of neuronal activity. *J Neurophysiol 94*(5), 3637–3642.

Brüderle, D. (2009). *Neuroscientific Modeling with a Mixed-Signal VLSI Hardware System*. Ph. D. thesis.

Brüderle, D., Müller, E., Davison, A., Muller, E., Schemmel, J., & Meier, K. (2009). Establishing a novel modeling tool: A python-based interface for a neuromorphic hardware system. *Fed. Proc. 3*(17). doi:10.3389/neuro.11.017.2009.

Burkitt, A. N., Meffin, H., & Grayden, D. B. (2004). Spike-timing-dependent plasticity: the relationship to rate-based learning for models with weight dynamics determined by a stable fixed point. *Neural Comput. 16*, 885–940.

Clopath, C., Büsing, L., Vasilaki, E., & Gerstner, W. (2010). Connectivity reflects coding: a model of voltage-based STDP with homeostasis. *Nat. Neurosci. 13*, 344–352.

Clopath, C., Ziegler, L., Vasilaki, E., Büsing, L., & Gerstner, W. (2008). Tag-Trigger-Consolidation: A model of early and late long-term-potentiation and depression. *PLoS Comput. Biol. 4*(12), e1000248.

Dan, Y., & Poo, M.-m. (2006). Spike timing-dependent plasticity: From synapse to perception. *PhysiolRev 86*, 1033–1048.

Daouzli, A., Saighi, S., Buhry, L., Bornat, Y., & Renaud, S. (2008). Weights convergence and spikes correlation in an adaptive neural network implemented on VLSI. In *Proceedings of the International Conference on Bio-inspired Systems and Signal Processing*, Funchal, pp. 286–291.

Davison, A. P., & Frégnac, Y. (2006). Learning cross-modal spatial transformations through spike timing-dependent plasticity. *J. Neurosci. 26*(21), 5604–5615.

Dayan, P., & Abbott, L. F. (2001). *Theoretical Neuroscience*. Cambridge: MIT Press.

Deb, K., & Kalyanmoy, D. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms* (1 ed.). Wiley.

DeFelipe, J., & Jones, E. G. (1988). *Cajal on the Cerebral Cortex: an annotated translation of the complete writings*. New York: Oxford University Press.

Druckmann, S., Banitt, Y., Gidon, A. A., SchÃŒrmann, F., Markram, H., & Segev, I. (2007). A novel multiple objective optimization framework for constraining conductance-based neuron models by experimental data. *Frontiers in Neuroscience 1*(1), 7–18. doi:10.3389/neuro.01/1.1.001.2007.

Dudek, S., & Bear, M. (1992). Homosynaptic long-term depression in area CA1 of hippocampus and effects of n-methyl-d-aspartate receptor blockade. *Proc. Natl. Acad. Sci. USA 89*(10), 4363–4367.

Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43.

Feldman, D. E., & Brecht, M. (2005). Map plasticity in somatosensory cortex. *Science 310*(5749), 810–815. doi:10.1126/science.1115807.

Fromherz, P. (2002). Electrical interfacing of nerve cells and semiconductor chips. *ChemPhysChem 3*(3), 276–284.

Gardiner, C. (2009). *Stochastic Methods: A Handbook for the Natural and Social Sciences* (4th ed.). Berlin, Heidelberg: Springer.

Gerstner, W., & Naud, R. (2009). How good are neuron models? *Science 326*(5951), 379–380.

Gewaltig, M.-O., & Diesmann, M. (2007). NEST (NEural Simulation Tool). *Scholarpedia 2*(4), 1430.

Grübl, A. (2003). Eine FPGA-basierte Platform für neuronale Netze. Master's thesis, University of Heidelberg.

Grübl, A. (2007). *VLSI Implementation of a Spiking Neural Network*. Ph. D. thesis. Document No. HD-KIP 07-10.

Gu, Q. (2002). Neuromodulatory transmitter systems in the cortex and their role in cortical plasticity. *Neuroscience 4*, 815–835.

Gütig, R., Aharonov, R., Rotter, S., & Sompolinsky, H. (2003). Learning input correlations through nonlinear temporally asymmetric Hebbian plasticity. *J. Neurosci. 23*(9), 3697–3714.

Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *J. Physiol. (Lond) 117*, 500–544.

Huys, Q. J. M., & Paninski, L. (2009). Smoothing of, and parameter estimation from, noisy biophysical recordings. *PLoS Computational Biology 5*(5), e1000379.

Izhikevich, E. (2003). Simple model of spiking neurons. *IEEE Transactions on neural networks 14*(6), 1569–1572.

Jeltsch, S. (2010). Computing with transient states on a neuromorphic multi-chip environment. Master's thesis.

Jovilet, R., Kobayashi, R., Rauch, A., Naud, R., Shinomoto, S., & Gerstner, W. (2008). A benchmark test for a quantitative assessment of simple neuron models. *J. Neurosci. Methods 169*, 417–424.

Kelso, S. R., Ganong, A. H., & Brown, T. H. (1986). Hebbian synapses in hippocampus. *Proc. Natl. Acad. Sci. USA 83*, 5326–5330.

Kennedy, J., & Eberhart, R. (2002). Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, Volume 4, pp. 1942–1948.

Kistler, W. M., Gerstner, W., & van Hemmen, J. L. (1997). Reduction of the Hodgkin-Huxley equations to a single-variable threshold model. *Neural Comput. 9*, 1015–1045.

Kononov, A. (2010). AdEx model with two adaptive terms and its restrictions due to hardware limitations of the HICANN chip. Internship, Ruprecht-Karls-University, Heidelberg.

Kuhn, A., Aertsen, A., & Rotter, S. (2003). Higher-order statistics of input ensembles and the response of simple model neurons. *Neural Comput. 1*(15), 67–101.

Levi, T., Lewis, N., Saighi, S., Tomas, J., Bornat, Y., & Renaud, S. (2008). *VLSI circuits for biomedical applications*, Chapter 12, pp. 241–264. Norwood: Artech House.

Markram, H. (2006). The blue brain project. *Nat. Rev. Neurosci. 7*, 153–160.

Markram, H., Lübke, J., Frotscher, M., & Sakmann, B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. *Science 275*, 213–215.

Meissner, M., Schmuker, M., & Schneider, G. (2006). Optimized Particle Swarm Optimization (OPSO) and its application to artificial neural network training. *BMC Bioinformatics 7*(1), 125.

Mensi, S., Naud, R., Pozzorini, C., Avermann, M., Petersen, C. C. H., & Gerstner, W. (2012). Parameter extraction and classification of three cortical neuron types reveals two distinct adaptation mechanisms. *J. Neurophysiol. 107*(6), 1756–1775.

Merolla, P., & Boahen, K. (2006). Dynamic computation in a recurrent network of heterogeneous silicon neurons. In *Proceedings of the 2006 International Symposium on Circuits and Systems (ISCAS)*, Island of Kos, pp. 4539–4542. IEEE Press.

Millner, S., Grübl, A., Schemmel, J., Meier, K., & Schwartz, M.-O. (2010). A VLSI implementation of the adaptive exponential integrate-and-fire neuron model. In *Advances in Neural Information Processing Systems (NIPS)*, Volume 23, Vancouver, pp. 1642–1650.

Miranda, V., & Fonseca, N. (2002). New Evolutionary Particle Swarm Algorithm (EPSO) applied to voltage/VAR control. In *14th Power Systems Computation Conference (PSCC)*, Sevilla, Spain.

Mitchell, M. (1998). *An Introduction to Genetic Algorithms (Complex Adaptive Systems)*. The MIT Press.

Morrison, A., Aertsen, A., & Diesmann, M. (2007). Spike-timing dependent plasticity in balanced random networks. *Neural Comput. 19*, 1437–1467.

Morrison, A., Diesmann, M., & Gerstner, W. (2008). Phenomenological models of synaptic plasticity based on spike-timing. *Biol. Cybern. 98*, 459–478.

Naud, R. (2011). Quantitative Single-Neuron Modeling Competition 2009. `http://www.incf.org/Events/competitions/spike-time-prediction`.

Naud, R., Marcille, N., Clopath, C., & Gerstner, W. (2008). Firing patterns in the adaptive exponential integrate-and-fire model. *Biological Cybernetics 99*(4-5), 335–347.

Oliphant, T. E. (2006). *Guide to NumPy*. USA: Trelgol Publishing. http://numpy.scipy.org.

Pawlak, V., & Kerr, J. N. (2008). Dopamine receptor activation is required for corticostriatal spike-timing-dependent plasticity. *J. Neurosci. 28*(10), 2435–2446.

Pelgrom, M., Duinmaijer, A., & Welbers, A. (1989). Matching properties of MOS transistors. *IEEE J. Solid-St. Circ. 24*(5), 1433–1439.

Pfeil, T., Potjans, T. C., Schrader, S., Potjans, W., Schemmel, J., Diesmann, M., & Meier, K. (2012). Is a 4-bit synaptic weight resolution enough? - constraints on enabling spike-timing dependent plasticity in neuromorphic hardware. *Front. Neurosci. 6*(90).

Pfister, J.-P., & Gerstner, W. (2006). Triplets of spikes in a model of spike timing-dependent plasticity. *J. Neurosci. 26*, 9673–9682.

Plana, L. A., Furber, S. B., Temple, S., Khan, M., Shi, Y., Wu, J., & Yang, S. (2007). A GALS infrastructure for a massively parallel multiprocessor. *IEEE Des. Test Comput. 24*(5), 454–463.

Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle swarm optimization. *Swarm Intelligence 1*(1), 33–57.

Potjans, T. C. (2011). personal communication.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical Recipes: The Art of Scientific Computing* (3rd ed.). Cambridge University Press.

Reynolds, J. N., & Wickens, J. R. (2002). Dopamine-dependent plasticity of corticostriatal synapses. *Neural Networks 15*, 507–521.

Rossant, C., Goodman, D. F., Platkiewicz, J., & Brette, R. (2010). Automatic fitting of spiking neuron models to electrophysiological recordings. *Frontiers in Neuroinformatics 4*.

Rubin, J., Lee, D., & Sompolinsky, H. (2001). Equilibrium properties of temporally asymmetric Hebbian plasticity. *Phys. Rev. Lett. 86*, 364–367.

Schemmel, J., Brüderle, D., Grübl, A., Hock, M., Meier, K., & Millner, S. (2010). A wafer-scale neuromorphic hardware system for large-scale neural modeling. In *Proceedings of the 2010 International Symposium on Circuits and Systems (ISCAS)*, Paris, pp. 1947–1950. IEEE Press.

Schemmel, J., Brüderle, D., Meier, K., & Ostendorf, B. (2007). Modeling synaptic plasticity within networks of highly accelerated I&F neurons. In *Proceedings of the 2007 International Symposium on Circuits and Systems (ISCAS)*, New Orleans. IEEE Press.

Schemmel, J., Fieres, J., & Meier, K. (2008). Wafer-scale integration of analog neural networks. In *Proceedings of the 2008 International Joint Conference on Neural Networks (IJCNN)*, Hong Kong. IEEE Press.

Schemmel, J., Grübl, A., & Millner, S. (2011). Specification of the HICANN microchip. FACETS project internal documentation.

Schemmel, J., Gruebl, A., Meier, K., & Mueller, E. (2006). Implementing synaptic plasticity in a VLSI spiking neural network model. In *Proceedings of the 2006 International Joint Conference on Neural Networks (IJCNN)*, Vancouver, pp. 1–6. IEEE Press.

Schwartz, M.-O. (2011). personal communication.

Sjöström, P., Turrigiano, G., & Nelson, S. (2001). Rate, timing, and cooperativity jointly determine cortical synaptic plasticity. *Neuron 32*, 1149–1164.

Song, S., & Abbott, L. F. (2001). Cortical development and remapping through spike timing-dependent plasticity. *Neuron 32*, 339–350.

Song, S., Miller, K. D., & Abbott, L. F. (2000). Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nat. Neurosci. 3*(9), 919–926.

Srowig, A. (2007). Analog floating gate memory in a 0.18 um single-poly CMOS process. *Internal FACETS documentation*.

Van Kampen, N. G. (2007). *Stochastic Processes in Physics and Chemistry, Third Edition (North-Holland Personal Library)* (3 ed.). North Holland.

van Rossum, M. C. W., Bi, G., & Turrigiano, G. G. (2000). Stable Hebbian learning from spike timing-dependent plasticity. *J. Neurosci. 20*(23), 8812–8821.

Wolfram Research, Inc. (2008). Mathematica edition: Version 7.0.

Zhang, J., Lau, P., & Bi, G. (2009). Gain in sensitivity and loss in temporal contrast of STDP by dopaminergic modulation at hippocampal synapses. *Proc. Natl. Acad. Sci. USA 106*, 13028–13033.

Acknowledgments

God for his mercy

My parents and the family of my sister for being my lovely family, always supporting me and placing their trust in me

My love Thesa

All my friends in Heidelberg, Kornwestheim and everywhere else for all the pleasure

Prof. Dr. Meier and Dr. Johannes Schemmel for supporting my thesis

Dr. Markus Diesmann for being my second referee

Daniel for being more than a supervisor and for his endless proof reading of my thesis, whereas is like “whereasses”

Tobias for having great ideas at the other side of the phone line and for standing my humor

Sebastian J. for holding the camera and his smiling

Marco for his assistance in all HICANN experiments

Eric for his telephone helpline

Mihai, Bernie and Matthias for beating me at the table next door

Paul for his genius raps

Andi G. for his assistance in setting up the STDP measurements

All proof readers for spending their time

All not yet mentioned Vision(s) for being around in good and bad times

The Gerstner group for their helpful tips

All Sakura Karatekas for helping me to purify my mind

Sarnen for its inspiring countryside

The KIP toilets for being a suitable backend for each type

Statement of Originality

I certify that this thesis, and the research to which it refers, are the product of my own work. Any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline.

Heidelberg, October 26, 2012

.....
(signature)