DISSERTATION

submitted to the

Combined Faculties for the Natural Sciences and for Mathematics
of the Ruperto-Carola University of Heidelberg, Germany

for the degree of

Doctor of Natural Sciences

Put forward by

M.Sc. Mitja Kleider

born in Hannover, Germany

Oral examination: November 14, 2017

NEURON CIRCUIT CHARACTERIZATION IN A NEUROMORPHIC SYSTEM

REFEREES:

Prof. Dr. Karlheinz Meier

Prof. Dr. Michael Hausmann

ABSTRACT

Spiking neural networks can solve complex tasks in an event-based processing strategy, inspired by the brain. One special kind of neuron model, the AdEx model, allows to reproduce several types of firing patterns, which have been found in biological neurons and may be of functional importance. In this thesis we characterize the analog neuron circuit implementation of this model within the full-custom HICANN ASIC. As the central unit of the BrainScaleS accelerated neuromorphic computing platform, it provides a tool to emulate large neural networks in short time and helps to better understand the brain.

Characterization of the neuron circuits leads to calibration of each sub-circuit, translating the desired AdEx model parameters to their corresponding HICANN parameters for each individual neuron. Device mismatch in VLSI manufacturing leads to expected variation from design parameters. These variations can be counteracted by adjustable parameters within the circuits. A wafer-scale BrainScaleS system contains over $1.9 \cdot 10^5$ neuron circuits with millions of parameters. Due to the large scale of the system, methods need to be fully automated in a robust way.

Characterizations presented in this work are performed from transistor level simulation to wafer-scale hardware measurements. Our commissioning and calibration efforts are enabling neural network experiments, including complex firing patterns that are computationally expensive when implemented in traditional numerical simulations.

ZUSAMMENFASSUNG

Vom Gehirn inspirierte spikende neuronale Netze sind trotz ihres Ereignis-basierten Kommunikations-Ansatzes in der Lage, komplexe Aufgaben lösen. Ein spezielles Neuronenmodell, das AdEx-Modell, kann verschiedene Feuermuster reproduzieren, die in biologischen Neuronen beobachtet wurden und von funktionaler Bedeutung sein könnten. In dieser Arbeit charakterisieren wir die Implementierung dieses Modells durch die analogen Neuronen-Schaltungen des HICANN-ASICs. Der HICANN stellt die zentrale Einheit des neuromorphen BrainScaleS-Systems dar und erlaubt es, große neuronale Netze in kurzer Zeit zu emulieren.

Die Charakterisierung der Neuronen-Schaltungen führt zur Kalibration der einzelnen Schaltungen, die für jedes einzelne Neuron gewünschte AdEx-Parameter in entsprechende HICANN-Parameter übersetzt. Abweichungen der Eigenschaften der Bauelemente während des Herstellungsprozesses führen zu Abweichungen von den Designparametern. Diese können durch einstellbare Parameter ausgeglichen werden. Ein Wafer-System enthält $1.9 \cdot 10^5$ Neuronenschaltungen mit Millionen von Parametern. Aufgrund dieser Größe müssen die Methoden vollständig automatisiert und robust sein.

Die in dieser Arbeit vorgestellten Charakterisierungen werden in Transistor-Simulationen und Wafer-übergreifenden Hardware-Messungen durchgeführt. Unsere Bemühungen zur Inbetriebnahme und Kalibration erlauben es, Netzwerkexperimente durchzuführen, einschließlich komplexer Feuermuster, die in numerischer Simulation viel Rechenaufwand erfordern.

CONTENTS

Part I

INTRODUCTION

1 NEUROMORPHIC COMPUTING

The human brain is a powerful computation system (Piccinini and Bahar, 2013): being energy efficient, robust against defects, and able to learn quickly, it has inspired new approaches to process information by copying aspects of its dynamics and structure. Traditional computing separates memory and logic, while the human brain accomplishes both in a network consisting of roughly 10^{10} neurons and 10^{14} synaptic connections (Pakkenberg et al., 2003) between neurons. Connectivity between units is high, at up to 10^4 incoming synapses per neuron. Boolean logic, which is trading energy for precision, does not seem to be required to accomplish information processing features of a brain. Although lots of data has been collected on properties of neurons and their connectivity, the details of how the brain represents, memorizes and processes information are not yet fully understood.

Modern machine learning algorithms, especially artificial neuronal networks initially inspired by the brain, are able to perform increasingly complex tasks that could previously only be performed by humans: image recognition, decision making, driving cars, or winning the board game Go (Bojarski et al., 2016; Silver et al., 2016) are just a few examples. As the number of everyday applications is growing and energy budget is limited, a new conceptual approach to computing is required. Companies are already building specialized hardware to reduce power consumption of their algorithms (Jouppi et al., 2017). However, there are still some pronounced differences. The algorithms may even be better at specialized tasks than a human, while the human brain is much more universal. In some cases, such as playing Go, the human brain is able to learn from observing very few examples, while artificial networks require way more training examples.

In order to better understand how the brain processes and stores information, numerical simulations of neural networks on traditional hardware are a powerful tool in the field of computational neuroscience. Several simplified neuron models are introduced in chapter 2. Efficiently scaling such simulations to large networks is a difficult task, communication can be the major bottleneck limiting parallel simulations (Migliore et al., 2006). The human brain excels at energy consumption of about $20\,W$ (Drubach, 2000), compared to traditional computers based on the von Neumann architecture. For example, Helias et al. (2012) simulated neural networks based on a simplified neuron model on the K supercomputer, consuming approximately 10^{10} times more energy than an equivalent sized part of the brain. Time evolution in such simulations is about 10^3 times slower than biological time, although methods have been developed to increase performance of such large-scale simulations (Kunkel et al., 2013).

There is an even more radical approach to casting aspects of brain computation into silicon, called neuromorphic computing. The idea to operate electronic transistors similar to neurons was first established by Carver Mead (Mead, 1990; Mead, 1989). Mead proposed to build very-large-scale integration (VLSI) systems with analog circuits working similar to the nervous system of a biological brain, emulating ion flow across the membrane of a neuron by electron flow through a transistor. Originally the term neuromorphic computing was used for subthreshold analog electronics, today it is used in a much broader sense and also includes purely digital and mixed-signal circuits.

There are many diverse examples for neuromorphic systems with different goals, including sensors (Vanarse, Osseiran, and Rassau, 2016), winner-take-all circuits (Indiveri, Chicca, and Douglas, 2009), and systems designed for large neural networks (Furber, 2016). One goal of such neuromorphic computing systems is to perform tasks where the brain outperforms traditional computers, at similarly low power consumption. Aside from the computing aspect in the interdisciplinary field of brain-inspired computing, building such systems may also help the field of neuroscience in exploring and understanding features of the brain such as representation of information, computations, robustness to defects and learning. Due to their neural network architecture, some systems also feature built-in robustness (Kalampokis et al., 2003; Petrovici et al., 2016) and may be able to compensate yield issues in modern chip manufacturing processes.

Compared to numerical integration of differential equations in computer simulations on super computers, the approach of building neuromorphic hardware as simulation tools uses less energy. Neuromorphic systems that are designed for large neural networks are aiming at simulation of these networks in real-time or faster, after an initial configuration phase. Due to the neuromorphic architecture, the speed just depends weakly, if at all, on the network size.

Many analog hardware implementations of spiking neurons exist today (Indiveri, Linares-Barranco, et al., 2011). In this thesis we will work with the BrainScaleS system, a large-scale neuromorphic system that uses an analog neuron model. Apart from the BrainScaleS system, there are several other neuromorphic hardware projects of similar scale, which have also been developed over similar time spans: TrueNorth (Sawada et al., 2016), being developed by IBM, SpiNNaker (Furber, Galluppi, et al., 2014), developed at the University of Manchester, and Neurogrid (Benjamin et al., 2014), developed at Stanford University. Their approaches are complementary, each choosing a different compromise between flexibility, accuracy, power consumption and detailed features of the supported neuron models:

The TrueNorth chip is designed as an application delivery platform for real-time cognitive applications (Esser, Andreopoulos, et al., 2013). One chip consists of 4096 cores of 256 digital integrate-and-fire neurons with 256 synaptic inputs each (Cassidy et al., 2013). Chips can be interconnected for larger applications. Incoming spike events are connected to selected neurons via a 256×256 cross-bar. Output events are connected to one other core, if several cores should be connected, identical neurons that receive the same input can be used to generate identical output to each core. The hardware behaves deterministically and can be simulated in software, which is required for off-line training. The idea is to provide a library of cognitive modules which can be combined for a specific application (Amir et al., 2013). A possible application for example is efficient image classification using deep convolutional networks, processing more than 6000 frames per Joule (Esser, Merolla, et al., 2016).

The SpiNNaker system is similar to a conventional supercomputer with several optimizations towards simulation of neural networks in real time. The processors are integer ARM969 cores originally developed for embedded applications. Communication through a multicast packet router is optimized for large numbers of very small data packets (Furber, 2016), typically single spikes. This approach allows high scalability (Furber, Lester, et al., 2012) and energy efficiency compared to conventional supercomputers while retaining some flexibility with respect to neuron models, but with the restriction of integer operations.

The Neurogrid system is based on subthreshold analog circuits (Benjamin et al., 2014). One board holds 16 Neurocore chips of 256×256 neurons, emulating ion channel activity in analog circuits. Spike communication and routing is digital. The neurons implement a quadratic integrate-and-fire model with shared leaky integrator dendritic circuits, operating in real time. Several parameters and the synapses are shared, reducing the network modeling possibilities, but leading to extremely low power consumption of less than two Watts for the whole board.

A possible application of neuromorphic real-time systems are small mobile robots, especially when they are battery powered. Future applications of systems that are able to process input faster than biological time are temporal data streams: detecting patterns and making predicitions, analyzing large volumes of data, in the areas of science, medicine, business, or civil services. Possible future industry applications of low power neuromorphic systems include mobile medical diagnostics, analyzing Lidar data on an automobile and other tasks that require fast processing of information from high-throughput sensors, because such systems need to respond on a similar time scale as the physical system. In particular for applications requiring high speeds, technology developed for the BrainScaleS system may be suitable to be adapted towards such applications.

In the next chapter we will give an overview of neuron models that lead to the neuron model emulated by the BrainScaleS system. In chapter 3 the BrainScaleS system is introduced, followed by details on its central component, the High Input-Count Analog Neural Network (HICANN) chip (chapter 4), experimental setups (chapter 5) and their control software (chapter 6). Chapter 7 presents the methods that have been developed to characterize the neuron circuits. The application of these methods to a full wafer is shown in chapter 8, followed by a discussion of the results in chapter 9.

2 POINT NEURON MODELS

A biological neuron can be roughly divided into three functional parts: dendrites, soma and axon. The dendrites collect input signals, short electrical pulses, from other neurons and transmit them to the soma. The soma generates an output signal if the input exceeds a certain threshold. The output pulse, called action potential or spike, is transmitted to other neurons via the axon. The junction between two neurons is the synapse, where, most commonly the signal is transmitted chemically between neurons via transmitter molecules. The sending cell is called presynaptic neuron and the receiving cell is called postsynaptic neuron. A single neuron typically connects to 10^4 presynaptic neurons (Gerstner et al., 2014) in a neural network. Action potentials of a given neuron always have roughly the same form. They are essentially binary signals (Sterling and Laughlin, 2015), information is encoded in the timing and number of spikes. The sequence of spikes from a single neuron is called spike train. In some sense, one can view the brain itself as a mixed-signal device: in contrast to the fast digital signalling via spikes, the signal processing in neurons and synapses is an efficient analog chemical process. Usually it is not possible to excite a second spike immediately after a spike, the neuron needs to recover during a refractory period.

Like other cells, neurons have a cell membrane separating its interior from the extracellular space. The concentration of ions inside the cell differs from the outside. This generates an electric potential over the membrane, the cell membrane potential. A typical neuron at rest, without any input, has a negative polarisation of about $-65\,\mathrm{mV}$. The potential changes with incoming spikes. If it increases, reducing (depolarizing) the negative polarization, the synapse is excitatory. If it decreases, the synapse is inhibitory. The change from the resting potential caused by a spike at the postsynaptic neuron is the post-synaptic potential (PSP). The dynamic range of the membrane potential is in the order of tens of millivolts, whereas the resulting action potential has an amplitude of about $100\,\mathrm{mV}$.

There are controversial discussions which level of detail is still relevant in order to realize brain functionality. From the biologist point of view, details at the molecular level are still important. A cognitive scientist might not even care about action potentials. The field of machine learning in computer science is able to perform certain tasks as good or better than humans with fine-tuned artifical neural networks, using very simple neuron models far from biology. However, these models might lack critical features to provide other functionality of an actual brain. In the following we will introduce several models that lead to the physical model implementation which is characterized in this thesis.

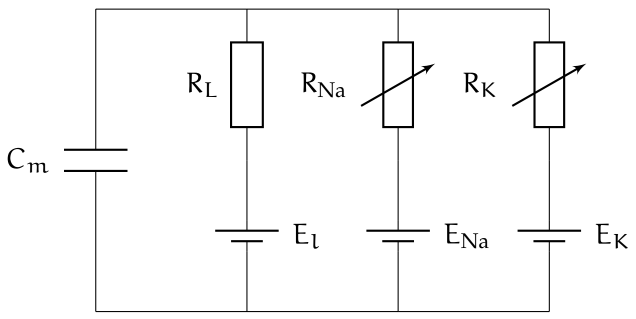


Figure 2.1: Ion channels as modeled by the Hodgkin-Huxley model. The unspecific leakage channel R_L is time-independent. The conductances of the sodium and potassium channels depend on activation and inactivation variables m, n, h.

2.1 HODGKIN-HUXLEY MODEL

The conductance-based Hodgkin-Huxley neuron model consists of a set of non-linear differential equations describing the membrane potential and the state of several ion channels. Ion channels move charge in the form of ions through the membrane. The membrane potential at which there is no net flow through one type of ion channel is called its reversal potential or Nernst potential. In its original form (Hodgkin and Huxley, 1952) the Hodgkin-Huxley neuron model describes three types of ion channels: a sodium channel with resistance R_{Na} and reversal potential E_{Na}, a potassium channel with resistance R_K and reversal potential E_K and an unspecific channel with resistance R_L and reversal potential E_l. This model has later been extended by many additional types of ion channels after they have been discovered. The unspecific leakage channel has a voltage-independent conductance $g_L = \frac{1}{R_L}$.

The conductances of the voltage-gated sodium and potassium channels contain time-dependent gating variables $m(t)$, $n(t)$ and $h(t)$, each with its own differential equation. The resulting currents onto the membrane capacitance C_m cause a change in the membrane potential V_m as

$$C_m \frac{dV_m}{dt} = -\hat{g}_{Na}\, m^3 h\, (V_m - E_{Na}) - \hat{g}_K\, n^4\, (V_m - E_K) - g_L\, (V_m - E_L) + I, \quad (2.1)$$

with additional external current stimulus I. The terms containing the gating variables m, n and h are responsible for spike generation and will be replaced by a reset mechanism in the next section.

The description of action potentials on the level of ion channels forms the basis for detailed biophysical neuron models. These models include varying sets of additional ion channels, which have been discovered in electrophysiological experiments. For example, spike-frequency adaptation is not present in the original Hodgkin-Huxley model.

2.2 LEAKY INTEGRATE-AND-FIRE MODEL

In a rough approximation, neuronal dynamics consist of integration of inputs and a firing mechanism, replacing the coupled nonlinear differential equations from the previous section by a reset condition. The class of neuron models describing action potentials as events are called Integrate-and-Fire models (Gerstner et al., 2014). Firing events are often defined as delta function at the time when the membrane potential is crossing a threshold V_t. The other component is an equation describing the evolution of the membrane potential V_m. The simplest model of this class is the Leaky Integrate-and-Fire (LIF). In the absence of external current stimulus I or synaptic input, the potential is at its resting value E_l. The cell membrane acts like a capacitor C_m that is charged by a current stimulus. The charge will slowly leak through the cell membrane with a conductance g_L:

$$C_m \frac{dV_m}{dt} = -g_L(V_m - E_l) + I. \tag{2.2}$$

The differential equation is complemented by a firing condition: the LIF model introduces a non-continuous firing mechanism by defining a firing threshold V_t. The neuron is firing when V_m crosses V_t from below and its membrane potential is reset immediately to a reset potential V_{reset}:

$$V_m \rightarrow V_{\text{reset}}. \tag{2.3}$$

This approach also reduces computational complexity for computer simulations.

As this model is highly simplified, it can not reproduce many aspects of neuronal dynamics, such as adaptation, bursting and inhibitory rebound. Two major limitations of the basic LIF model are the linear integration of synaptic or external stimulus current and the spiking mechanism that does not keep any memory of previous spikes. If adaptation and refractoriness are added to the model, it is able to predict biological spike times more accurately (Gerstner et al., 2014).

Refractoriness can be added by clamping the membrane potential to V_{reset} during the refractory period τ_{ref} after a spike at t_{spike}:

$$V_m \left(t_{\text{spike}} < t \leqslant t_{\text{spike}} + \tau_{\text{ref}}\right) = V_{\text{reset}}. \tag{2.4}$$

2.3 IZHIKEVICH MODEL

Izhikevich proposed a quadratic integrate-and-fire model that reproduces spiking and bursting behavior of neurons (Izhikevich, 2003):

$$\frac{dV_m}{dt} = c_2 \cdot {V_m}^2 + c_1 \cdot V_m + c_0 - w + I, \tag{2.5}$$

$$\frac{dw}{dt} = a \cdot (b \cdot V_m - w), \tag{2.6}$$

with a reset condition: if $V_m \geqslant 30\,\mathrm{mV}$, set $V_m = c$ and $w = w + d$. The variables V_m and w as well as the parameters a, b, c and d are dimensionless. The constants c_0, c_1 and c_2 in equation (2.5) were fitted to match recorded neuron dynamics. V_m represents the membrane potential, w represents a recovery variable, accounting for activation of the potassium ion current and inactivation of the sodium current. a describes the recovery time scale, b the sensitivity of w to fluctuations of V_m. c is the reset value of V_m, and d is a spike-triggered change of the recovery variable w.

2.4 ADAPTIVE LEAKY INTEGRATE-AND-FIRE MODEL

The LIF model can be extended by adding an adaptation variable, similar to the recovery variable in the Izhikevich model:

$$C_m \frac{dV_m}{dt} = -g_L(V_m - E_l) - w, \tag{2.7}$$

$$\tau_w \frac{dw}{dt} = a\,(V_m - E_l) - w. \tag{2.8}$$

The firing condition remains as defined by the LIF model. For Spike-Triggered Adaptation (STA) the adaptation current is increased by a constant value b at each firing event:

$$w \to w + b. \tag{2.9}$$

2.5 ADAPTIVE EXPONENTIAL LEAKY INTEGRATE-AND-FIRE MODEL

Introduced by Brette and Gerstner (2005), the Adaptive Exponential Leaky Integrate-and-Fire (AdEx) model extends the two-variable model from the previous section by an exponential nonlinearity proposed in Fourcaud-Trocmé et al. (2003). The resulting AdEx model is described by two differential equations:

$$C_m \frac{dV_m}{dt} = -g_L(V_m - E_l) + g_L \Delta_T \exp\left(\frac{V_m - V_T}{\Delta_T}\right) - w + I. \tag{2.10}$$

And as before,

$$\tau_w \frac{dw}{dt} = a\,(V_m - E_l) - w. \tag{2.11}$$

The first equation describes the evolution of the membrane potential V_m on a membrane capacity C_m. There are several current terms, beginning with a leakage current through the leak conductance g_L, pulling the membrane voltage towards the resting potential E_l. The second term on the right hand side is the exponential term with slope factor Δ_T and exponential threshold V_T. Third comes the adaptation current w, its course is described by the second differential equation. Its time evolution is described by the adaptation coupling conductance a and adaptation time constant τ_w. The last term in equation (2.10) I is the sum of input currents, which can be synaptic currents or external stimulus. The firing condition remains as introduced in section 2.2. With the addition of the exponential term, in a purely mathematical description the neuron would be firing at time t^f when its membrane potential V_m diverges towards infinity.

This model is able to reproduce several other neuron models. Removing the adaptation variable w results in the exponential integrate-and-fire model (Fourcaud-Trocmé et al., 2003). In the limit $\Delta_T \to 0$, we obtain the adaptive LIF model. Without exponential term ($\Delta_T \to 0$) and by setting $a = 0$, there remains LIF with STA as proposed in Treves (1993). Setting $b = 0$ results in subthreshold adaptation (Richardson, Brunel, and Hakim, 2003).

There are many similarities to the Izhikevich model: the AdEx model remains fast to simulate numerically, shows the same bifurcation patterns and a large variety of firing patterns. Quantitative fits to experimental data are better when using the AdEx model (Naud et al., 2008). There are also a few differences between the two models: while the Izhikevich model contains a quadratic voltage dependency, the nonlinear dependency of the AdEx model is exponential. The resulting upswing of the action potential in the AdEx model is more realistic (Badel et al., 2008). The AdEx model also remains linear in the subthreshold regime as observed in experiments, while the Izhikevich model shows unrealistic nonlinearities (Badel et al., 2008). The attenuation of high frequency inputs follows as $\frac{1}{f}$ in the AdEx model, compared to $\frac{1}{f^2}$ in the Izhikevich model (Fourcaud-Trocmé et al., 2003).

In summary, the Adaptive Exponential Leaky Integrate-and-Fire (AdEx) model is a good compromise between computational complexity and reproducibility of firing characteristics which have been observed in biological experiments.

This model is implemented in the BrainScaleS system introduced in chapter 3. The firing patterns presented in (Naud et al., 2008) have been reproduced using this system in Schwartz (2013), Tran (2013), and Kiene (2014).

Part II

BRAINSCALES SYSTEM

3 BRAINSCALES SYSTEM

The neuromorphic hardware located in Heidelberg is called BrainScaleS. The idea behind this large-scale system is that configurable synaptic connectivity and weights as well as configurable individual neuron parameters provide the flexibility to realize various functionality in artificial neural circuits. Simulating learning on traditional computing architectures takes a long time, even if it is possible to provide the large amount of required energy. Our approach is to develop a new computing architecture which is much faster at emulating neural networks than traditional computing architectures and requiring less energy at the same time.

The central component of the BrainScaleS system is the full-custom, mixed-signal HICANN chip, which is introduced in chapter 4. There are three key concepts: each wafer module features a high neuron count without depending on external connections, due to an architecture called *wafer-scale integration* (Schemmel, Fieres, and Meier, 2008). In order to achieve higher bandwidths and use less energy in signal transmission compared to single chips being connected externally via a printed circuit board (PCB), the silicon wafer used to manufacture HICANN chips side-by-side is not cut into separate chips. 384 chips on the wafer are instead interconnected by additional metal layers in a post-processing step and directly connected to a main PCB using elastomeric connectors. The next concept is *physical emulation*: a set of differential equations (section 2.5) is not solved numerically, but emulated by a physical model implementation of the differential equations. This limits the system to the physically implemented model equations, but allows for a very fast and energy efficient time evolution (Schemmel, Fieres, and Meier, 2008). The choice of conductances and capacitances allows the system to emulate the model 10^4 times faster than real time, whereas computer simulations usually are slower than real time. This acceleration allows to evolve network dynamics in experiments faster than real-time and thus gain insights into processes that would take too much time to simulate them numerically. The third key concept is learning being integrated into the system, which also benefits from the accelerated time evolution. There is already support for Short Term Plasticity (STP) (Schemmel, Grübl, et al., 2006; Billaudelle, 2014) and Spike-Timing Dependent Plasticity (STDP) (Schemmel, Grübl, et al., 2006; Nonnenmacher, 2015). The next generation of the system will contain a plasticity processor to facilitate learning (Friedmann, Frémaux, et al., 2013; Friedmann, Schemmel, et al., 2016; Friedmann, 2013).

Apart from researching and designing circuits for energy-efficient realization of new computing paradigms, the BrainScaleS system is intended as a tool for theretical neuroscientists. Access for early adopters is already available via the neuromorphic computing platform of the Human Brain Project (HBP). First proof of concept networks have already been published (Schmitt et al., 2016). As simulation of a day is compressed to seconds, the next generation of the BrainScaleS system will also allow to observe learning over a larger time scale.

The physical emulation of the neuron dynamics is implemented by analog electronic circuits, detailed in the following chapters. Spike communication is realized by transmitting binary information via digital circuits, and received by 4 bit synapses (Millner, 2012; HBP SP9 partners, 2014). The system is highly configurable: each neuron has individual neuron parameters to control its behavior, the synaptic connectivity can be reconfigured to implement arbitrary neural networks. The synaptic connectivity is designed to be scalable, such that a single model neuron can receive input from up to 10^4 sources.

4 HICANN CHIP

The HICANN mixed-signal Application Specific Integrated Circuit (ASIC) is the central component of the BrainScaleS system. It implements the AdEx model in full-custom analog circuits (Millner, 2012), which is discussed in section 4.1.

In addition, HICANN contains a digital part to configure and connect these analog circuits. Communication with HICANN takes place via the Automated Repeat reQuest (ARQ) protocol (Philipp, 2008; Karasenko, 2014). On chip the neural network activity is transmitted digitally (Hartel, 2016; Jeltsch, 2014). The digital spikes are then processed by the synaptic input circuits and converted to a current (section 4.1.6).

4.1 NEURON CIRCUIT

The analog Dendritic Membrane Circuit (denmem) is the central subject of investigation in this work. In the following, we give an overview over the circuits relevant for this thesis and explore in detail how to control their behavior. Figure 4.1 shows a simplified schematic. We are using the term *denmem* as established in Millner (2012) instead of neuron, because a model neuron can consist of several denmems. This circuit realizes the physical AdEx model implementation (equations (2.3) and (2.9) to (2.11)) with an electronic capacitor C_m for the membrane capacitance, which can be digitally configured to two values (section 4.2.1): a smaller capacitance $C_{\text{smallcap}} = 0.16\,\text{pF}$ and a bigger capacitance $C_{\text{bigcap}} = 2.16\,\text{pF}$. There are individual circuits to implement the currents from each term in equation (2.10) on the membrane capacitance: the leakage term (section 4.1.1), adaptation term (section 4.1.3) and exponential term (section 4.1.4), as well as the reset mechanism for equation (2.3) (section 4.1.2). This approach allows to observe the membrane potential as a time-continuous quantity, a physical voltage.

In total, there are 512 denmems on a single HICANN chip. The membrane capacitances of neighboring denmems can be interconnected up to a block of 64 denmems to act as a single model neuron (Jeltsch, 2014). This can be used to achive a higher number of available synaptic inputs (see section 4.1.7). The boundary of these blocks of 64 denmems can not be shifted, smaller block sizes are possible within these boundaries.

Further details on the design can be found in Millner (2012), where the full denmem circuit is discussed in detail. One exception is the revised synaptic input circuit, which has been discussed and characterized in Koke (2016).

There are two types of adjustable parameters in these circuits: 23 parameters are shared by 128 or 256 denmems. They are also known as global parameters, intro-

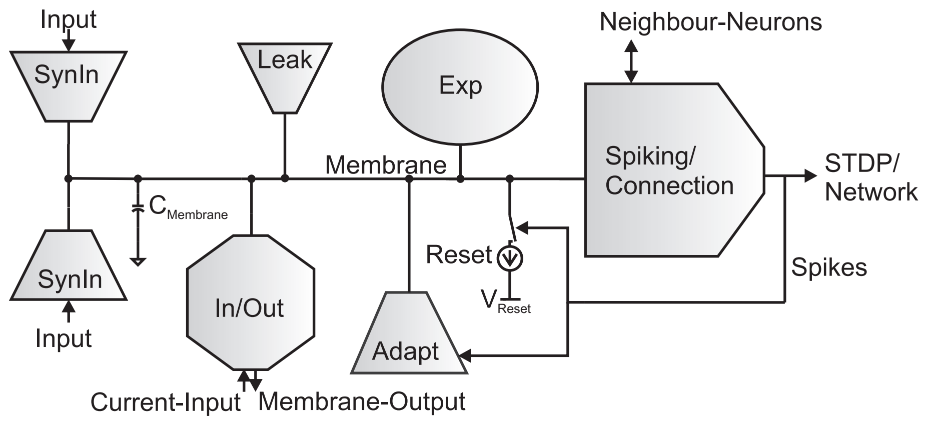


Figure 4.1: Simplified schematic of the AdEx physical model implementation in a denmem circuit (Millner, 2012). Most sub-circuits are introduced in this chapter. From left to right there is the excitatory and inhibitory synaptic input circuit, the membrane capacitance C_m, external current stimulus and membrane voltage readout, leakage term, adaptation term, exponential term, reset mechanism and spiking mechanism. The denmem can be interconnected with neighboring denmems for higher synaptic connectivity. The spiking mechanism is also triggering the reset mechanism as well as Spike-Triggered Adaptation (STA). Schematic from Millner et al. (2010).

duced in section 4.2.4. Another 22 parameters can be adjusted for each denmem individually, known as neuron parameters (section 4.2.3).

4.1.1 *Leakage Term*

Several conductances in HICANN are implemented by an operational transconductance amplifier (OTA). This circuit has two voltage inputs and ideally outputs a current proportional to the product of a conductance, which is controlled by a biasing current, as well as the voltage difference at the inputs (Geiger and Sanchez-Sinencio, 1985):

$$I_{\text{out}} \propto \sqrt{I_{\text{bias}}} \cdot (V_+ - V_-)\,. \tag{4.1}$$

The leakage conductance g_L is implemented by a single OTA with bias current I_{gl} (figure 4.2). The resting potential E_l is connected to its positive input and its output is connected to the membrane capacitance as well as the negative input, resulting in a leakage current onto the membrane towards the equilibrium at E_l.

$$I_{\text{leak}} = -g_L\,(V_m - E_l) \propto \sqrt{I_{\text{gl}}}\,(E_l - V_m)\,. \tag{4.2}$$

The real behavior of the OTA circuit implemented in HICANN has been simulated in detail in previous works (Millner, 2012; Koke, 2016; Kiene, 2014). Figure 4.3

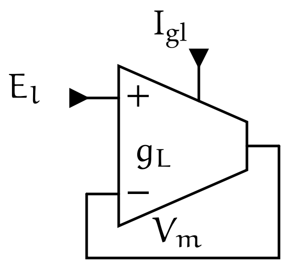


Figure 4.2: The leakage conductance g_L is realized by an OTA with bias current I_{gl}. The positive input is connected to the resting potential E_l, the output is connected to the membrane capacitance with voltage V_m as well as the negative input, resulting in a leakage current onto the membrane towards the equilibrium at E_l (adapted from Millner (2012)).

shows its characteristic curves for the conductance and the resulting output current as a function of the differential voltage between its inputs. Due to saturation effects, the OTA's conductance depends on the voltage-difference at its inputs. Depending on the network model this effect may be negligible. We can operate at small voltage

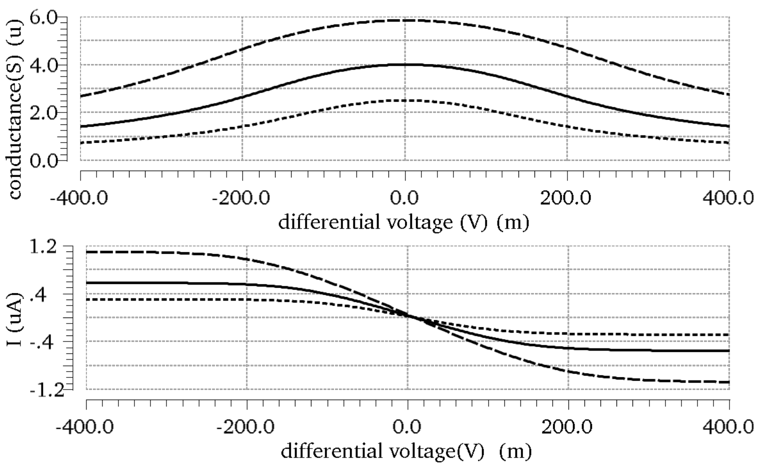


Figure 4.3: Characteristic curves of the OTA used in several HICANN circuits, obtained from transistor level simulations. At the top, the conductance of the OTA is plotted depending on the differential voltage at its inputs. The three line styles depict three different bias current settings. At the bottom, the resulting current output of the OTA is shown. The ideal conductance would be constant, resulting in a linear current. The actual conductance shows a dependency on the voltage difference at the inputs with saturation at high differential voltage. In order to avoid deviations from the AdEx model with constant conductance, the parameter translation (section 4.2.5) and neuron configuration can be chosen to avoid large differential voltages. From (Millner, 2012).

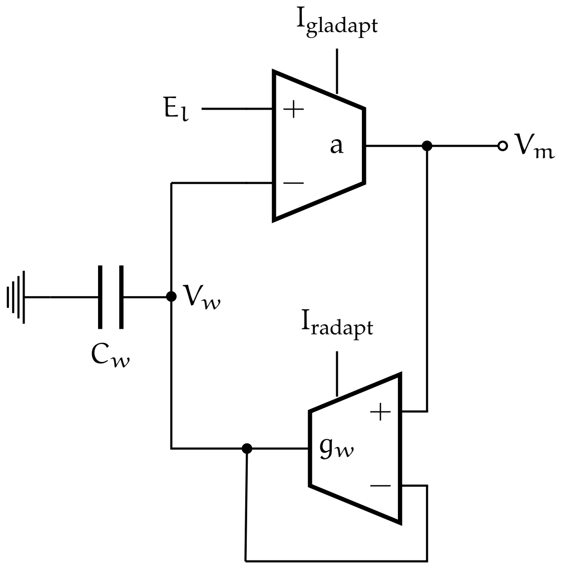


Figure 4.4: Simplified schematic of the adaptation circuit. Adaptation in form of equations (4.3) and (4.7) is realized using two OTAs and a capacitor C_w. The capacitor is storing the state w as voltage V_w. The output of the conductance a, with bias current $I_{gladapt}$, is the adaptation current w. V_w is coupled to V_m via the conductance g_w (equation (4.7)), with bias current I_{radapt}.

differences by choosing an appropriate parameter translation (section 4.2.5) and corresponding neuron parameters in order to remain close to the theoretical model.

4.1.2 *Reset Mechanism*

The theoretical model defines that if the membrane voltage crosses the spike threshold, V_m is set to the reset potential V_{reset}. In physical emulation there is a current pulling the membrane voltage V_m to the reset potential V_{reset}. Its strength is adjustable via global parameter I_{reset}. It is designed to be strong enough to dominate in comparison to leakage and synaptic currents, resulting in an almost instantaneous reset. The duration during which the membrane is held at V_{reset} can be adjusted via parameter I_{pl}, resulting in a refractory period τ_{ref}. The reset is triggered by a comparator with inputs V_m and V_t, which also triggers the emission of a digital spike signal (see section 4.1.7).

4.1.3 *Adaptation Term*

On HICANN, the present value of the adaptation variable w is stored as voltage V_w on a capacitor $C_w = 2\,\text{pF}$ (figure 4.4). The capacitances C_w of interconnected denmems are not connected to each other. The adaptation current w onto the membrane in equation (2.10) is the output of an OTA, the conductance a, with E_l and V_w at its input terminals. This gives the following relation:

$$w = a\left(V_w - E_l\right), \tag{4.3}$$

and its time-derivative

$$\frac{dw}{dt} = a \cdot \frac{dV_w}{dt}. \tag{4.4}$$

Putting equations (4.3) and (4.4) into equation (2.11) gives

$$-\cancel{a}\tau_w \frac{dV_w}{dt} = \cancel{a}\left(V_w - E_l\right) - \cancel{a}\left(V_m - E_l\right) \tag{4.5}$$

$$\Leftrightarrow -\tau_w \frac{dV_w}{dt} = \left(V_w - V_m\right) \tag{4.6}$$

$$\Leftrightarrow -C_w \frac{dV_w}{dt} = g_w\left(V_w - V_m\right). \tag{4.7}$$

In the last step, we have introduced another conductance g_w to realize the time constant $\tau_w = \frac{C_w}{g_w}$. Equation (4.7) is also implemented by an OTA, very similar to the leakage circuit.

Spike-Triggered Adaptation (STA) as defined in equation (2.9) is triggered by the digital spike signal. The increase of current w by b can be expressed as a voltage increase by V_b on C_w:

$$b = a \cdot V_b = a \cdot \frac{q_b}{C_w}, \tag{4.8}$$

equation (2.9) becomes

$$V_w \to V_w + V_b. \tag{4.9}$$

The required charge q_b flowing on the capacitor is added by enabling an adjustable current I_{fire} during the digital firing pulse with pulse length $t_{\text{fire}} \ll \tau_w$ and τ_m:

$$q_b = t_{\text{fire}} \cdot I_{\text{fire}}. \tag{4.10}$$

The bias current I_{fire} can be used to set the desired value for b. When the adaptation term is reset, V_w is connected to V_m by a transistor not shown in figure 4.4. The reset signal opens the transistor, circumventing g_w and resulting in $V_w = V_m$. If V_w is at the resting potential, equation (4.3) becomes $w = 0$. This concludes the required level of detail to control the adaptation term. Additional details on this realization and its limitations can be found in Millner (2012).

4.1.4 *Exponential Term*

The exponential term in equation (2.10) is an exponential current

$$I_{\text{exp}} = g_L \cdot \Delta_T \exp\left(\frac{V_m - V_T}{\Delta_T}\right) \tag{4.11}$$

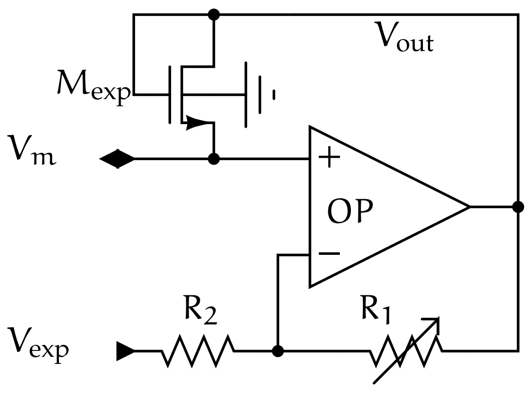


Figure 4.5: Simplified schematic of the exponential term. The transistor M_{exp} is operated in its subthreshold regime to provide a current depending exponentially on $V_{out} - V_m$. The threshold at which the OpAmp becomes active is determined by its negative input V_- and can be controlled by V_{exp}. The slope of the exponent is controlled by the voltage divider consisting of an adjustable resistance R_1 and constant resistance R_2. Adapted from (Millner, 2012).

with a threshold voltage V_T and slope factor Δ_T.

The exponential feedback was implemented by connecting the output of an operational amplifier (OpAmp) to a transistor M_{exp} (figure 4.5) operated in its subthreshold regime: its gate-to-source voltage is below the conducting threshold voltage, $V_{GS} \leqslant V_{th}$. This configuration results in a subthreshold leakage current

$$I_{DS} = I_{D0} \exp\left(\frac{V_{GS} - V_{th}}{n \cdot u_t}\right) \tag{4.12}$$

from drain to source which depends exponentially V_{GS} (Allen and Holberg, 2002). I_{D0} is a constant with a temperature dependency. The temperature is assumed to be constant over time in local areas of the system. The constant n is called subthreshold swing parameter and characterizes the subthreshold slope of the transistor (Millner, 2012; Hu, Liu, and Jin, 1998). $u_t = \frac{k_B \cdot T}{q}$ is the thermal voltage, with Boltzmann constant k_B, elementary charge q and temperature T.

In order to calculate V_{GS}, we need to know the voltage V_{out} at the output of the OpAmp. Assuming an ideal OpAmp, there is no current flowing through its input terminals. Let us consider the current flowing from the OpAmp's output to V_{exp}:

$$I_1 = \frac{V_{out} - V_{exp}}{R_1 + R_2}. \tag{4.13}$$

The same current is flowing through R_2, giving us the voltage at its negative terminal V_-:

$$\begin{aligned} V_- &= V_{exp} + R_2 \cdot I_1 \\ &= V_{exp} + R_2 \cdot \frac{V_{out} - V_{exp}}{R_1 + R_2} \\ &= V_{exp} + (V_{out} - V_{exp}) \cdot \frac{R_2}{R_1 + R_2}. \end{aligned} \tag{4.14}$$

The OpAmp regulates its output V_{out} to equalize the voltages present at its input terminals, such that $V_+ \overset{!}{=} V_-$. We can hence write

$$V_m = V_{exp} + \left(V_{out} - V_{exp}\right) \cdot \frac{R_2}{R_1 + R_2} \tag{4.15}$$

$$\Leftrightarrow V_{out} = \left(V_m - V_{exp}\right) \frac{R_1 + R_2}{R_2} + V_{exp}. \tag{4.16}$$

Now we can calculate V_{GS} of M_{exp}:

$$V_{GS} = V_{out} - V_m \tag{4.17}$$

$$= \left(V_m - V_{exp}\right) \frac{R_1 + R_2}{R_2} - \left(V_m - V_{exp}\right) \frac{R_2}{R_2} \tag{4.18}$$

$$= \left(V_m - V_{exp}\right) \frac{R_1}{R_2}. \tag{4.19}$$

By inserting equation (4.19) into equation (4.12), we obtain

$$I_{Mexp} \equiv I_{DS} \approx I_{D0} \exp\left(\frac{\left(V_m - V_{exp}\right) \frac{R_1}{R_2} - V_{th}}{n \cdot u_t}\right) \tag{4.20}$$

$$= \alpha \exp\left(\frac{V_m - V_{exp}}{\frac{\beta}{R_1}}\right) \tag{4.21}$$

for the current through M_{exp}, with two constants $\alpha = I_{D0} \exp\left(\frac{-V_{th}}{n \cdot u_t}\right)$ and $\beta = R_2 \cdot n \cdot u_t$. The adjustable resistance R_1 in the voltage divider is realized with a current mirror (figure 4.6). If M_1 is in saturation, the total current through R_1 is adjustable as scaled mirror of the bias I_{rexp}:

$$I_1 \approx (4 + 1) \cdot I_{rexp} = 5 \cdot I_{rexp}. \tag{4.22}$$

Therefore R_1 in the voltage divider is inversely proportional to I_{rexp}, assuming ohmic behavior. Due to its design, R_1 does not behave like an ideal resistor, but also has a dependency on the voltage difference. Correct behavior of the exponential term is most relevant at V_m close to V_T, where R_1 behaves as intended. Combined with saturation of the OpAmp, I_{Mexp} will decrease if $V_m \gg V_T$ (Millner, 2012). If this saturation is occuring just before reaching the spiking threshold V_t, the effect

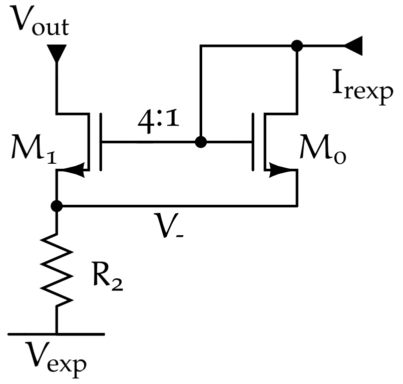


Figure 4.6: Schematic of the voltage divider as part of the exponential circuit (figure 4.5). The current through M_1 is controlled via I_{rexp}. Adapted from (Millner, 2012).

is negligible. In transistor level simulations at $I_{Mexp} = 100\,nA$ the relation between I_{rexp} and Δ_T is almost linear in the range of $I_{rexp} \approx 100\,nA$ to $800\,nA$, resulting in $\Delta_T \approx 3\,mV$ to $14\,mV$ (Millner, 2012). Larger values for I_{rexp} in this configuration lead to saturation of the OpAmp.

Comparing equation (4.21) to equation (4.11), we see that

$$\Delta_T = \frac{\beta}{R_1}. \tag{4.23}$$

The factor $g_L\Delta_T$ is determined by V_{exp}:

$$g_L\Delta_T \exp\left(\frac{V_m - V_T}{\Delta_T}\right) \overset{!}{=} \alpha \exp\left(\frac{V_m - V_{exp}}{\frac{\beta}{R_1}}\right) \tag{4.24}$$

$$\Rightarrow \exp\left(\frac{V_m - V_T - (V_m - V_{exp})}{\Delta_T}\right) = \frac{\alpha}{g_L\Delta_T} \tag{4.25}$$

$$\frac{V_{exp} - V_T}{\Delta_T} = \ln\left(\frac{\alpha}{g_L\Delta_T}\right) \tag{4.26}$$

$$V_{exp} = V_T - \Delta_T \ln\left(\frac{g_L\Delta_T}{\alpha}\right). \tag{4.27}$$

Note that g_L represents the model's leak conductance and is otherwise unrelated to the OTA bias of the leakage term. From equation (4.27) we can see that the value of V_{exp} required to achieve a fixed V_T will decrease when Δ_T is increased.

In summary, the exponential slope Δ_T is proportional to I_{rexp}. The exponential threshold voltage V_T depends on both I_{rexp} and V_{exp} and is linear with V_{exp} at a fixed I_{rexp}.

4.1.5 *Current Input and Membrane Output*

The denmem can be stimulated by a programmable current source, intended for single neuron experiments. Amplitude configuration is stored in the floating gate (FG) cell memory, allowing 129 discrete steps which can optionally be repeated periodically. When connected to the membrane capacitor, the current stimulus line introduces a non-negligible additional parasitic capacitance (Schmidt, 2014; Millner, 2012) which adds to C_m. It should only be connected to a single denmem at the same time to avoid low impedance effects (Hartel, 2016).

This sub-circuit, denoted as *In/Out* in figure 4.1, also contains the circuitry to read out the membrane voltage V_m. This feature is used to deduce circuit behavior from voltage traces (chapter 7). A HICANN chip has two output amplifiers for simultaneous membrane voltage readout of two neurons, i. e. interconnected or single denmems. Additionally, the two analog readout lines are shared by all HICANNs on a reticle, limiting the simultaneous membrane voltage readout to two neurons per reticle in multi-HICANN experiments or parallel calibration runs. The output amplifier causes a systematic error by shifting the voltage by $(4.5 \pm 0.8)\,\mathrm{mV}$ (Millner, 2012), which can be equalized by our software (section 7.4).

4.1.6 *Synaptic Input and Layer 1 Routing*

The synaptic inputs generate an exponentially shaped conductance trace between a synaptic reversal potential E_{syn} and the membrane V_m. Digital spike signals are routed over a continuous-time serial bus system, called Layer 1 (L1), using both on-chip and post-processing lines for intra-wafer communication (Fieres, Schemmel, and Meier, 2008). Both inputs, inhibitory and excitatory, can be activated by incoming signals, generating current pulses I_{syn} arriving at the circuit in figure 4.7. The neuron implementation in HICANN revision 4 has two different synaptic input circuits, one for excitatory and one for inhibitory input. Their circuit parameters are suffixed by "x" or "i", respectively.

The two synaptic input circuits per denmem in the previous revision were idential. We have analyzed the previous implementation and provided possible workarounds for the raised issues in Schmidt (2014). These have been fixed in the new synaptic input implementation of HICANN revision 4, which has been characterized in Koke (2016) and is not analyzed in this thesis.

The L1 network is not part of the denmem, but provides input to the synapse circuit. Spike events are transmitted as digital signals on the L1 network, with a 6 bit event address (EA). Two of these bits are used to select a synapse driver. The other four bits are matched by the individual synapses. For digital spike communication between one or several HICANNs there are 256 vertical and 64 horizontal L1 buses, which can be connected to each other via crossbar switches: a horizontal line can be connected by crossbar switches to four vertical lines on the left and right side of the chip each (Fieres, Schemmel, and Meier, 2008; HBP SP9 partners, 2014). Vertical

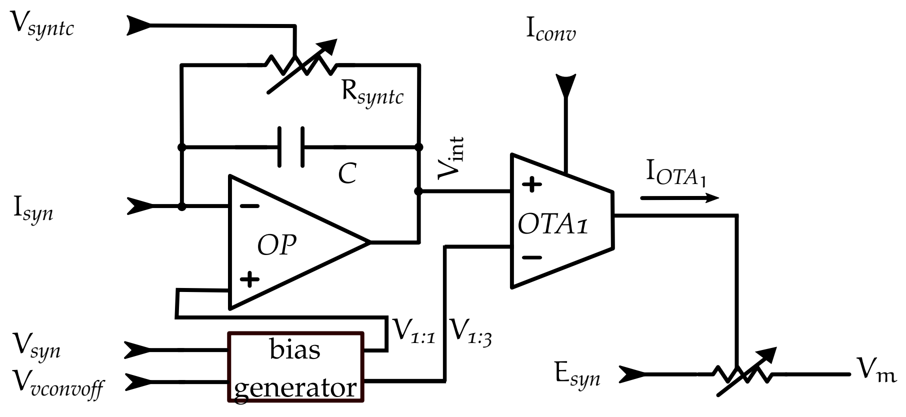


Figure 4.7: Schematic of the synaptic input circuit. Current pulses I_{syn} are integrated by the OpAmp and decay exponentially in the RC circuit with time constant τ_{syn}, controlled by V_{syntc}, towards V_{syn}. The integrated voltage V_{int} generates a current from the OTA, which controls the resistive element connecting E_{syn} to V_m. In order to compensate an offset of V_{int} to V_{syn} in the absence of input, an offset to V_{syn} can be added at the negative input of the OTA via $V_{convoff}$. Schematic adapted from Koke (2016).

L1 lines can be connected to synapse drivers, where one synapse driver can be fed from 16 different vertical lines. Synapse drivers with configurable L1 address decoder can be connected to a vertical bus lane via a switch matrix. The signal from the synapse driver is then scaled by V_{gmax} and the synaptic weight, resulting in a current pulse I_{syn} to the connected synaptic input if the address is matching (Koke, 2016):

$$I_{syn} = V_{gmax} \cdot g_{scale} \cdot \frac{w}{g_{div}}. \tag{4.28}$$

One HICANN has a synapse array of 512×220 horizontal and vertical synapses. Compared to HICANN revision 2, four vertical synapse lines were removed, because the new synapse circuit in HICANN revision 4 required additional space. The L1 network allows to reconfigure connectivity for each target experiment (section 6.1).

At the chip boundaries, repeaters help to propagate the differential spike signal to the neighbouring chip. L1 repeaters use delay-locked loops (DLLs) to align their timing. These circuits need to be *locked* before an experiment and will stabilize themselves by spike activity present during an experiment.

There are two mechanisms to inject additional spikes into the L1 bus during experiments: spike events can be either sent from the FPGA Communication PCB (FCP) (introduced in section 5.2.1) at user-defined times or generated by background generators inside HICANN. The eight background generators on a chip can be configured to emit periodic spike events or Poisson-distributed events. As the pseudo-random number generator is periodic, the random events are not perfectly Poisson-distributed (Petrovici, 2015). In random mode, the target address is restricted to 0 in HICANN revision 4. Because the repeaters require spike events during an experiment to maintain their DLL lock (Ziegler, 2013), background generators are usually configured for periodic events on address 0.

4.1.7 *Spiking Mechanism and Interconnection*

In the chip's layout, two neighboring neurons are combined in a pair and share their spiking and connection circuit (Millner, 2012).

The connection circuit allows to connect the membrane capacitor to neighboring denmems, forming a multi-denmem neuron with more synaptic inputs than a single one. As the sum of currents increases linearly with the membrane capacitance, the behavior remains identical to a single denmem as long as all denmems are configured for identical behavior (Schmidt, 2014).

If the membrane potential of a denmem is higher than its firing threshold potential, a digital firing signal can be emitted with the denmem's configured address on the L1 bus. In case of interconnected denmems, only one denmem of the group is configured to emit a spike signal (Jeltsch, 2014).

Addresses that are not associated with a background generator can be used to identify spike times from up to 63 distinguishable denmems simultaneously in calibration experiments. As long as the experiment does not require any synaptic connectivity and DLL locking, address 0 could be used for this purpose as well.

4.2 NEURON PARAMETERS

The denmem circuits feature many adjustable parameters, some of which have been mentioned in the previous section. Here we will give an overview how they are stored and which parameters can be configured to influence the individual analog circuits.

4.2.1 *Time Constant Scaling and Digital Configuration*

The membrane time constant τ_m and adaptation time constant τ_w depend on the conductances g_l, a and g_w. These are controlled by analog bias currents (compare section 4.1). Analog parameters are stored in floating gate (FG) cells, which are introduced in the next section.

On HICANN there are digital switches that rescale time constants by amplifying the bias currents to cover a larger range of values for the time constants. Two bits each for the bias currents I_{gl}, I_{gladapt} and I_{radapt} allow three different current mirror scaling selections: slow, normal and fast. They can be set individually for each bias current, resulting in current scaling of 1:1, 1:3 or 1:27. Each scaling is available once for the upper and once for the lower two FG blocks, i. e. the first or second group of 256 denmems each.

Additionally the membrane capacitance can be configured. The switch *bigcap* enables or disables the connection of several capacitors to a larger membrane capacitance. The small capacitance is 0.16 pF, enabling *bigcap* results in a sum of $C_m = 2.16\,\text{pF}$. The acceleration factor α_{acc} of the physical emulation on hardware compared to the model results from the relation between capacitances and conductances:

$$\alpha_{\text{acc}} = \frac{C_{\text{Model}}}{g_{\text{Model}}} \cdot \frac{g_{\text{Hardware}}}{C_{\text{Hardware}}}. \tag{4.29}$$

The achievable conductance value ranges were chosen such that typical model ranges can be reached at $\alpha_{\text{acc}} = 10^4$. This factor is used for parameter translation in section 4.2.5.

Taking the conductances measured for the default capacitance (C_{bigcap}) as a reference, all conductances are scaled by $g_{\text{smallcap}} = g_{\text{bigcap}} \cdot \frac{C_{\text{smallcap}}}{C_{\text{bigcap}}}$ if the small capacitance is chosen. For higher accuracy, the actual conductances can be measured again in separate experiments with C_{smallcap}.

The digital denmem configuration also allows to set the switches for membrane voltage readout and current input (section 4.1.5).

4.2.2 *Floating Gate Memory*

The analog parameters are stored in floating gate (FG) cells (Millner, 2012; Hock, 2009; Srowig et al., 2007; Loock, 2006) .

There are two types of FG cells in HICANN: voltage cells and current cells, with a programmable voltage range from 0 V to 1.8 V and a current range of 0 µA to 2.5 µA. The resulting value is programmed from a 10 bit Digital-to-Analog Converter (DAC) reference output before an experiment run. A fully isolated Positive Metal–Oxide–Semiconductor (PMOS) transistor gate is charged and discharged using Fowler-Nordheim-Tunneling (Lande et al., 1996). This requires a large programming voltage of at least 10 V. A more detailed description of this process is given in Koke (2016), Hartel (2016), and Millner (2012). Additional parameters allow adjusting the FG controller's behavior during programming.

The charge on the FG determines the cell's output: in a voltage cell the FG transistor together with a biasing transistor forms a source follower, buffering the gate voltage as the cell's output. The output current of current cells is provided by a current mirror. Several current parameters can be scaled by additional current mirrors (section 4.2.1). A current cell which is configured to 0 DAC will still output a small current, which is estimated to range between 20 nA and 50 nA. By repeatedly changing the charge stored in the FG from zero to a desired value, a trial-to-trial

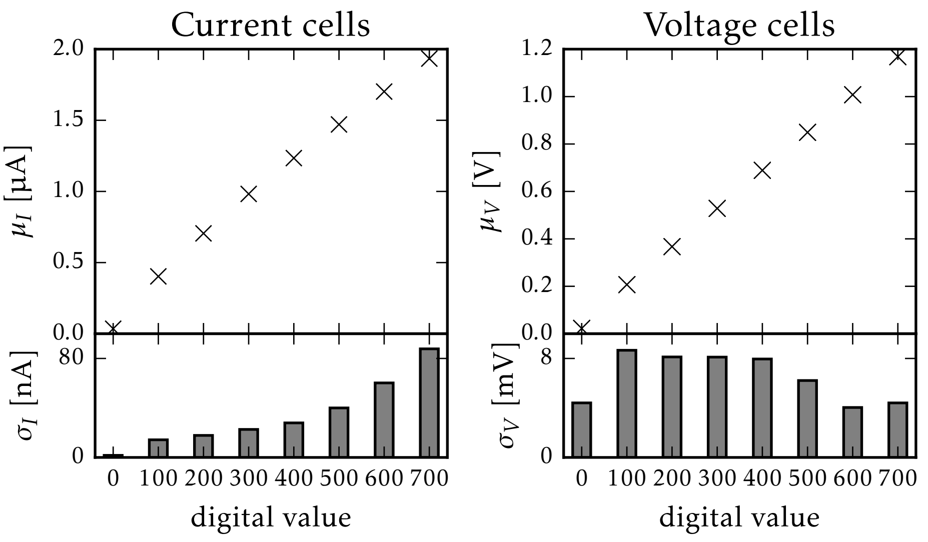


Figure 4.8: Floating gate cell trial-to-trial variation. The top part shows the mean, the bottom part shows the standard deviation of the current or voltage resulting from repeatedly configuring identical digital settings of FG current and voltage cells over 100 repetitions. Current cell precision decreases for higher currents, voltage cell precision shows a standard deviation between 5 mV to 9 mV over the measurable range. Figure from (Koke, 2016).

variation in the order of several millivolts is observed for voltage cells (figure 4.8). Current cells show a variation that is growing with the desired output current.

Physically, HICANN's analog parameter storage is divided into four blocks of $129 \cdot 24$ FG cells. The first column of 24 parameters stores the shared parameters (table 4.3). The other 128 columns store neuron parameters (table 4.2) for 128 denmems. Note that not all cells are used (compare table 4.1), leaving room for future circuit improvements.

Table 4.1 provides an overview how analog parameters have been layouted to connect to FG cells. The functionality of the parameters is described in sections 4.2.3 and 4.2.4.

4.2.3 *Neuron Parameters*

Neuron parameters, potentials and bias currents, can be configured for an individual denmem. This enables us to fine-tune each circuit to get as close to the desired behavior as possible. The resulting precision is limited by the ability to deduce the desired behavior from the accessible data (see section 4.1.5), as well as the precision of setting a value (see section 4.2.2). Table 4.2 shows the available parameters and their function. The parameters for 512 denmems are supplied by four different FG blocks. The connection scheme is depicted in figure 4.9.

row	type	global parameter	left denmem	right denmem
0	voltage	V_{reset} (left: even, right: odd blocks)	-	E_{synx}
1	current	I_{OPbias} (intern. OP bias)	I_{bexp}	I_{convi}
2	voltage	V_{dllres}	$V_{convoffx}$	E_{syni}
3	current	V_{bout} (left), V_{bexp} (right)	I_{convi}	I_{convx}
4	voltage	V_{fac}	$V_{convoffi}$	V_{exp}
5	current	I_{breset}	$I_{spikeamp}$	I_{intbbx}
6	voltage	V_{dep}	E_l	V_{synx}
7	current	I_{bstim}	I_{fire}	I_{intbbi}
8	voltage	V_{thigh}	V_{syni}	V_{syntci}
9	current	$V_{gmax<3>}$	$I_{gladapt}$	I_{pl}
10	voltage	V_{tlow}	V_{syntci}	V_{syni}
11	current	$V_{gmax<0>}$	I_{gl}	$I_{gladapt}$
12	voltage	V_{clra} (left), V_{crlc} (right)	V_t	V_{syntcx}
13	current	$V_{gmax<1>}$	I_{pl}	I_{rexp}
14	voltage	V_{stdf}	V_{syntcx}	V_t
15	current	$V_{gmax<2>}$	I_{radapt}	I_{bexp}
16	voltage	V_m	E_{synx}	E_l
17	current	V_{bstdf}	I_{convx}	$I_{spikeamp}$
18	voltage	-	E_{syni}	$V_{convoffi}$
19	current	V_{dtc}	I_{intbbx}	I_{fire}
20	voltage	-	V_{exp}	$V_{convoffx}$
21	current	V_{br}	I_{intbbi}	I_{gl}
22	voltage	-	V_{synx}	-
23	current	V_{ccas}, V_{cbias}	I_{rexp}	I_{radapt}

Table 4.1: Correspondence of row in the floating gate array with global parameter (first column) or neuron parameter (following 128 columns), adapted for HICANN revision 4 from HBP SP9 partners (2014). There are four FG blocks in total, storing parameters for 512 denmems. Even blocks are 0 and 2, odd blocks are 1 and 3. The type of FG cell is alternating between voltage and current cell. Global current parameters starting with the letter 'V' are part of distributed current mirrors and are usually matching the naming in the schematic.

neuron parameter	function
E_l	leakage potential, section 4.1.1
I_{pl}	refractory time bias, section 4.1.2
I_{gl}	leak conductance bias, section 4.1.1
I_{fire}	adaptation current increase (b) bias, section 4.1.3
$I_{gladapt}$	adaptation conductance, section 4.1.3
I_{radapt}	adaptation time constant, section 4.1.3
V_{exp}	exponential threshold control, section 4.1.4
I_{bexp}	V_{exp} buffer bias, section 4.1.4
I_{rexp}	exponential slope and threshold control, section 4.1.4
V_t	spike threshold, section 4.1.7
V_{synx}	excitatory synaptic input reference voltage
V_{syni}	inhibitory synaptic input reference voltage
I_{intbbx}	integrator bias in excitatory synaptic input
I_{intbbi}	integrator bias in inhibitory synaptic input
I_{convx}	OTA bias in excitatory synaptic input
I_{convi}	OTA bias in inhibitory synaptic input
$V_{convoffx}$	excitatory synapse integrator offset
$V_{convoffi}$	inhibitory synapse integrator offset
E_{synx}	excitatory synaptic reversal potential
E_{syni}	inhibitory synaptic reversal potential
V_{syntcx}	excitatory synaptic time constant
V_{syntci}	inhibitory synaptic time constant
$I_{spikeamp}$	spike threshold comparator bias, section 4.1.7

Table 4.2: Parameters which can be set for each individual denmem and the summary of their function. Details on each parameter can be found in the corresponding section or reference therein. Synaptic input parameters are shortly introduced in section 4.1.6 and explained in detail in Koke (2016).

4.2.4 *Shared Parameters*

Global parameters of a FG block are shared between 128 denmems. The only shared parameter directly present in equation (2.10) is V_{reset}. The fact that this parameter is shared is utilized to determine a voltage readout offset in section 7.4. Other shared parameters are used for STP (Billaudelle, 2014), STDP (Nonnenmacher, 2015), energy saving functionality or other biases. With the exception of STP calibration

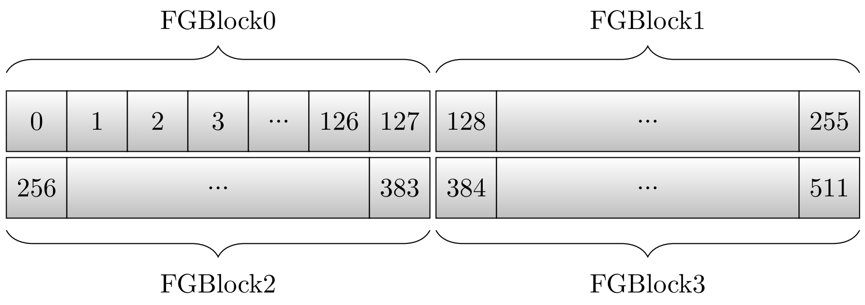


Figure 4.9: Denmem coordinates for a single HICANN and associated FG block that supplies neuron parameters to each denmem. Even blocks are left, odd blocks are right. Blocks 0 and 1 supply the top half of denmems, 2 and 3 supply the bottom half. These coordinates and terms are used in the software interface introduced in chapter 6.

these parameters are usually not changed during calibration. Table 4.3 lists the available shared parameters and their function.

The association between shared parameters and FG blocks is different from the connectivity for neuron parameters (figure 4.9). Even denmems are supplied by even FG blocks, and blocks 0 and 1 connect to the top half, blocks 1 and 2 connect to the bottom half. Figure 4.10 shows the connection scheme.

4.2.5 *Parameter Translation*

HICANN's implementation of the AdEx model operates in a different parameter range than found in biological neuron models. The model neuron parameters need to be rescaled to the range that is realized by the physical model. Analog voltages on HICANN range from 0 V to 1.8 V. The OTA and other circuits were designed for a common mode of 0.9 V. Therefore, the resting potential should be close to or a bit

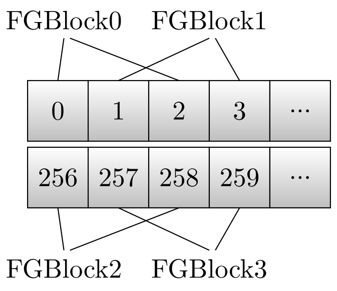


Figure 4.10: Denmem coordinates for a single HICANN and associated FG block that supplies shared parameters to each denmem. Connectivity for shared parameters, including V_{reset}, is different than connectivity for neuron parameters (figure 4.9). Even denmems are supplied by even blocks, 0 and 1 connect to top, 2 and 3 connect to bottom denmems.

shared parameter	function
I_{breset}	reset current bias, section 4.1.2
I_{bstim}	current stimulus bias, section 4.1.5
V_{bexp} (right only)	V_{exp} buffer bias, section 4.1.4
V_{bout} (left only)	neuron readout bias, section 4.1.5
V_{br}	STDP readout bias
V_{bstdf}	STP bias
V_{ccas}	L1 input amplifier bias
V_{clra} (left only)	acausal STDP accumulation rate
V_{clrc} (right only)	causal STDP accumulation rate
V_{dep}	STP depression mode offset
V_{dllres}	L1 repeater DLL reset voltage
V_{dtc}	STP recovery current control
V_{fac}	STP facilitation mode offset
V_{gmax0}	max. synaptic conductance reference current (!) 0
V_{gmax1}	max. synaptic conductance reference current (!) 1
V_{gmax2}	max. synaptic conductance reference current (!) 2
V_{gmax3}	max. synaptic conductance reference current (!) 3
V_m	causal STDP
V_{reset}	reset potential, section 4.1.2
V_{stdf}	Short Term Depression and Facilitation (STDF) reset voltage
V_{thigh}	STDP readout comparison voltage
V_{tlow}	STDP readout comparison voltage
int op bias	FG cell internal amplifier bias

Table 4.3: Parameters which are shared between all denmems connected to one of the four FG blocks. Two parameters appear on the left or right blocks only, they are connected to denmems belonging to the opposite block as well. STDP related parameters are explained in Nonnenmacher (2015), STP / STDF related parameters are explained in Billaudelle (2014).

below this value. Additionally, there is an upper limit for the membrane voltage at approximately 1.2 V by design. Rising above this voltage can lead to undefined behavior and is avoided by setting the spiking threshold $V_t \leqslant 1.2\,V$ (Millner, 2012). The maximum threshold V_t should be 1.1 V to have a safety range for variation. There is one voltage that was designed to be above this limit: the excitatory synaptic reversal potential E_{synx} can go up to 1.8 V.

The resting potential in biological neurons is typically $-65\,\mathrm{mV}$ with spike pulse amplitudes of about $100\,\mathrm{mV}$ (Gerstner et al., 2014), and an overall range from $-100\,\mathrm{mV}$ to $0\,\mathrm{mV}$. Biological voltages from the AdEx model can be translated to the hardware domain by introducing a scaling factor α_V and offset δ_V:

$$V_{\text{Hardware}} = \delta_V + \alpha_V \cdot V_{\text{Biology}}. \tag{4.30}$$

The scaling factor α_V results from the ratio of the dynamic voltage ranges:

$$\alpha_V = \frac{\Delta V_{\text{Hardware}}}{\Delta V_{\text{Biology}}}. \tag{4.31}$$

The offset δ_V is obtained by mapping any reference potential, V_t for example, from the biological model to its hardware equivalent:

$$\delta_V = V_{\text{ref,Hardware}} - \alpha_V \cdot V_{\text{ref,Biology}}. \tag{4.32}$$

These values should be chosen such that the membrane voltage will be near the circuits' operating point of $0.9\,\mathrm{V}$, the membrane voltage will stay below the maximum of $1.2\,\mathrm{V}$ and the voltage range is limited to approximately $200\,\mathrm{mV}$ around E_l in order to avoid saturation of any OTA (section 4.1.1). If OTA saturation is acceptable for a particular network model, the range around E_l can be larger. Decreasing the voltage range further would increase the OTA's linearity, but at the same time would increase effects of readout noise (figure 5.8) and cross talk. As changing the voltage scaling will also affect PSP heights, a different scaling changes the available range. Another aspect that needs to be considered is that V_{reset} is shared between 128 denmems. If a model requires different values of V_{reset} in a shared block, individual denmem voltages can be shifted from the common V_{reset} by an additional offset δ_{reset} added to all potentials. The dynamics will remain unchanged since potential differences are still the same. Choosing a dynamic hardware voltage range of $(1.0\,\mathrm{V} - 0.7\,\mathrm{V}) = 0.2\,\mathrm{V}$, a biological dynamic range of $100\,\mathrm{mV}$ and the spike threshold at $1.0\,\mathrm{V}$ and $0\,\mathrm{mV}$ respectively as a reference, we get $\alpha_V = 3$ and $\delta_V = 1.0\,\mathrm{V}$. Default values for the translation are $\alpha_V = 10$ and $\delta_V = 1.2\,\mathrm{V}$. They can be overwritten if deviating membrane potential ranges are needed.

Since the equations are physically emulated in accelerated time (compare section 4.2.1), biological time or real time is translated to hardware time by the acceleration factor $\alpha_{\text{acc}} = 10^4$:

$$t_{\text{Biology}} = t_{\text{Hardware}} \cdot \alpha_{\text{acc}}. \tag{4.33}$$

This way, for example, one year of neural network activity in biology can be emulated in $60 \cdot 24 \cdot 365/10^4 \approx 53$ minutes on hardware.

The above scaling of time and voltage at constant capacitance C_m also results in

$$\Delta_{\text{T, Hardware}} = \Delta_{\text{T, Biology}} \cdot \alpha_V, \tag{4.34}$$

$$\tau_{\text{Hardware}} = \frac{\tau_{\text{Biology}}}{\alpha_{\text{acc}}} \quad \text{for all time constants.} \tag{4.35}$$

With a fixed acceleration factor we can calculate the scaling of conductances from equation (4.29):

$$g_{\text{Hardware}} = \underbrace{\alpha_{\text{acc}} \cdot \frac{C_{\text{Hardware}}}{C_{\text{Biology}}}}_{=:\alpha_{\text{cond}}} \cdot g_{\text{Biology}}. \tag{4.36}$$

If the inverse ratio of capacitances is equal to the acceleration factor ($\alpha_{\text{cond}} = 1$), the conductances are equal in the physical hardware and biological model.

The conductance scaling also applies to adaptation term variables:

$$a_{\text{Hardware}} = \alpha_{\text{cond}} \cdot a_{\text{Biology}}, \tag{4.37}$$

$$b_{\text{Hardware}} = \alpha_V \cdot \alpha_{\text{cond}} \cdot b_{\text{Biology}}. \tag{4.38}$$

4.3 TRANSISTOR-LEVEL SIMULATION

The HICANN ASIC design can be simulated at transistor level, using the Spectre Circuit Simulator (Cadence Design Systems, 2012) which is shipped in the chip design suite, Cadence Virtuoso. Necessary parameters for the transistor models (Hu, Liu, and Jin, 1998) are provided by the chip manufacturer. In case of HICANN the manufaturer is UMC.

During circut design, it is an important part of the design process to manually execute simulations and change parameters. In a testbench design, the circuit under investigation is connected to inputs such as ideal current and voltage sources. Currents through nodes and voltages between nodes can be evaluated for given static or time-dependent input values.

For full reproducibility the testbench needs to be saved together with all variable input values. Therefore, we have developed an approach that allows to set the input to our testbenches for HICANN components from a Python script. The Python interface is realized by *teststand* (Billaudelle, 2017). Selected voltages and currents are returned from the simulation for further analysis in the script. This approach allows to build a testbench in Virtuoso, and to define simulations in a Python script. This lowers the barrier to create reproducible simulation results and automated benchmarks for experienced users of Cadence suite. At the same time this approach allows anyone to run experiments using transistor level simulations for a prebuilt testbench without detailed knowledge of the circuit design tools. Development of algorithms for hardware characterization can start before the actual chip has been manufactured, using such a prebuilt testbench.

We have integrated the simulation of a denmem testbench in the hardware experiment software stack such that single denmem experiments can be run on a transistor level simulation with the same Application Programming Interface (API) as for single denmem experiments on actual hardware.

4.4 MISMATCH AND NONLINEARITY

Mismatch causes time-independent random variations in physical quantities of identically designed devices (Pelgrom, Duinmaijer, Welbers, et al., 1989), also referred to as fixed-pattern noise. This mismatch is the result of several random processes which occur during fabrication of the complementary metal–oxide–semiconductor (CMOS) devices. These variations are part of the transistor models (Hu, Liu, and Jin, 1998). By using the upper and lower limits of variation in a worst-case analysis, digital circuits can be designed to not be affected, as digital logic defines a valid range corresponding to a binary value. As soon as analog values matter, it becomes more difficult: these time-independent variations of circuit design parameters lead to deviations in the resulting analog behavior compared to the ideal design. Additionally there are nonlinear effects in components of the circuits due to saturation (Razavi, 2001). There are several approaches to compensate for these effects:

- Deviations are characterized and counteracted by a tunable parameter in the circuit (circuit parameter calibration).
- Deviations are characterized and introduced in calculations using the analog value (analysis calibration).
- The circuit is designed such that it counteracts the effects of its deviation (self-calibration).

HICANN provides many parameters for circuit calibration, which is the central topic in chapter 7. A simple example of analysis calibration is introduced in section 5.4.1.

4.5 DEFECT INFORMATION

Single denmems or even whole HICANN chips might be classified as not functional by our software. A defect can be a non-responding or misbehaving component, for example a single denmem or an L1 repeater. We also classify a component as not functional if a target parameter value required by our classification is outside the achievable range.

In the BrainScaleS system we can route around non-functional components when emulating networks on a wafer with minor defects. If a denmem or synapse driver was classified as defect, it is not used and another one is used instead. If a whole HICANN was classified as defect, spikes are routed in the L1 network around this particular HICANN. Defect information of a wafer is stored in a defect database (Klähn, 2013) by the hardware characterization routines (section 6.2). It can be looked up by the mapping process (section 6.1) when placing and routing neural networks to ignore denmems or other components.

5 EXPERIMENTAL SETUPS

There are several types of experimental setups suitable to run experiments on HICANN being introduced in this chapter, in chronological order. Several components of the experiment platforms are re-used in later iterations. They are introduced with the earliest setup that is using the particular component.

The Demonstrator Setup is the earliest experiment platform, designed to test individual HICANN prototype chips and serving as a prototype platform before scaling up to a full wafer. It provides up to 8 HICANNs with the required supply voltages and contains the first variant of the FCP to communicate digitally with the chip.

Two wafer systems as prototypes for the BrainScaleS system scale the features of the Demonstrator Setup to a full wafer with 384 HICANNs.

Cube Setups are designed as test setups for an updated FCP and replace the Demonstrator Setup for single chip experiments. They are also used to test wafers that are not yet mounted in a full wafer system using a needle card to contact pads on the wafer.

The final BrainScaleS system consists of 20 wafer modules built with the updated FCP.

Each of these setups has been used to develop the software required for this thesis, although earlier setups can be regarded as prototypes to develop the final BrainScaleS system. The description of setups is held at the level of detail that was required to operate the setups for this thesis. A more detailed description is provided in the case that the information is not available in previous publications. The most detailed description of the BrainScaleS system as being used in the HBP neuromorphic platform can be found in HBP SP9 partners (2014).

Analog readout on all setup types is handled by the Analog Readout Module (AnaRM), introduced in section 5.4.

5.1 DEMONSTRATOR SETUP

The *Demonstrator Setup* – also informally named *Vertical Setup* by the vertical orientation of its PCBs – was the first prototype setup for HICANN (Schemmel, Grübl, et al., 2012; Millner, 2012; Schwartz, 2013). This type of setup was used to test the single chip prototype versions of HICANN, produced as part of a Multi-Project Wafer (MPW) before ordering a full wafer. Communication between a host computer and HICANN is provided by the same FPGA Communication PCB (FCP) as used for the prototype wafer setups (Thanasoulis, Partzsch, et al., 2012).

Due to its size and number of components, the Demonstrator Setup is more portable than the larger wafer setup. Transportation to workshops or other demonstration purposes is possible (Schemmel, Grübl, et al., 2012). However, the setup was not designed for frequent transportation. The stacked PCBs need to be disassembled before transportation to avoid damage caused by vibration. HICANN PCB and FCP connectors are rated for a low number of connection cycles.

The Demonstrator Setup consists of several PCBs:

- System Emulator Board (SEB) (labeled IBoard on silkscreen),
- FCP with Virtex-5 FPGA and 4 Digital Network Chips (DNCs),
- HICANN PCB,
- AnaRM or oscilloscope.

The components are also used in later experimental setups and will be introduced in the following.

5.1.1 *System Emulator Board*

The System Emulator Board (SEB) as the central PCB of the Demonstrator setup connects to the other PCBs via fine-pitch Samtec connectors and coaxial cables for analog readout. There are four connectors for HICANN PCBs and one connector for a Virtex-5 FPGA PCB. Its main purpose aside from connecting other PCBs is providing supply voltages to one or multiple HICANNs. The Virtex-5 FPGA PCB is powered by its own small power supply PCB.

A single HICANN can be powered using a linear or switching regulator for the 1.8 V supply. The SEB can hold 8 HICANNs in total, two each on up to four HICANN PCBs. The populated linear regulators can not provide enough power to several HICANNs, in this case switching regulators need to be used. The regulator type is selectable via jumpers. Some SEBs in our inventory are populated with just one of the two regulator types. SEBs assembled in 2015 (labeled as IB5-IB7) for Cube Setups (section 5.2) contain both types. If the linear regulator should be used, jumpers must be placed at the pins with silkscreen labels "SR ON" (actually disabling the switching regulator) and "LDD ON" (enabling the linear regulator). For the switching regulator selection both of these jumpers should be absent.

The linear regulator is still sufficient to power two HICANNs. Using more than a single chip at the same time has not been neccessary during this thesis, as the wafer systems became available for multi-HICANN experiments.

Supply voltages (table 5.1) are set by configuring a DAC via I2C. Before first operation, regulator circuits need to be adjusted towards the configured voltage using the corresponding potentiometer. The multiplexers for two analog readout connectors are also set via I2C. On Demonstrator Setups, the required commands are being sent from the Virtex-5 FPGA PCB.

voltage	function	comment
VDD5	for level shifter to 12 V domain	should be $\approx \frac{\text{VDD11}}{2}$, usually 5.5 V
VDD11	FG programming voltage (section 4.2.2)	up to $\approx$ 12.5 V (VDD12 in other places)
VDD33	LVDS supply for high-speed (HS) interface	3.3 V
VDD25	FG source follower supply	2.5 V
VDD	HICANN digital supply	1.8 V
VDDA	HICANN analog supply	1.8 V
VOH	L1 high potential	usually 1.1 V
VOL	L1 low potential	usually 0.9 V
VDDBUS	synapse line driver supply	$\approx$ 1.1 V, power saving feature
VDD_{DNC}	HICANN DNC interface supply	same source as VDD
VDDA_{DNC}	HICANN DNC interface analog supply	same source as VDDA
$\text{VBIAS}_{\text{LVDS}}$	common mode for HS link	1.25 V, usually supplied by DNC/FCP

Table 5.1: Supply voltages provided by the SEB to a HICANN PCB. HICANN DNC interface supply voltages have separate lines to allow measuring their current towards the chip separately on the SEB. In some places, VDD11 is called VDD12: it can be set between 10 V to 12.5 V. All voltages are generated from an external supply of 13.8 V. There are switching regulators for 6 V (source for VDD5), 3.3 V and 1.8 V. Other voltages are generated from these voltages using linear regulators. The voltage drop through the source follower transistor supplied by VDD25 is ≈ 0.6 V, leaving a margin ($2.5\,\text{V} - 0.6\,\text{V} = 1.9\,\text{V}$) for output voltages up to 1.8 V.

5.1.2 *Virtex-5 FPGA PCB*

The main task of the Virtex-5 FPGA PCB is communication to HICANN (Scholze et al., 2011; Thanasoulis, Vogginger, et al., 2014; Müller, 2014). This includes configuring HICANN as well as sending and receiving spikes. Besides a Virtex-5 FPGA, the PCB contains four custom DNCs (Dresden, 2010; Dresden, 2008) to communicate to a group of 8 HICANNs each. Such a group is called reticle and yielded from the possible area of exposure during the wafer manufacturing process, which fits 8 HICANNs. The board is connected via ethernet to a host computer running our control software.

One Virtex-5 FPGA PCB is replaced by four Kintex-7 FPGA PCBs, called FPGA Communication PCB (HBP SP9 partners, 2014), to reduce cost and complexity. This replacement is described in section 5.2.1.

5.1.3 *HICANN PCB*

Up to four HICANN PCBs can be attached to the SEB. Each can hold up to two HICANNs and has no further function. A close-up photo of this PCB without bond-

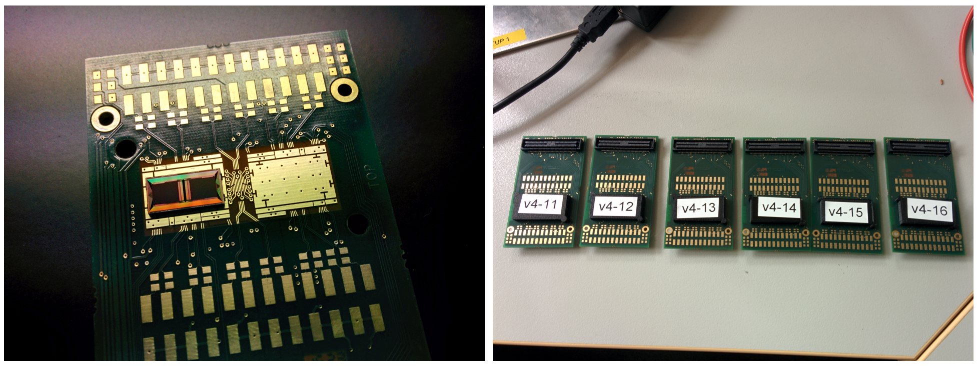


Figure 5.1: Left: Single HICANN on HICANN PCB without bonding wires. Supply voltage lines are accessible on the large pads. Right: Bonded HICANNs below the labeled caps are lined up for testing.

ing wires is shown in figure 5.1. For commissioning of the HICANN revision 4 prototype chips, one or two HICANN chips are wire bonded to a HICANN PCB. After testing the PCB for short circuits and measuring the resistance to ground of all available pads (table 5.1), the PCB is mounted on the SEB to perform experiments on the bonded chips. Bonding HICANN revision 4 in-house with a F&K Delvotek 6400 aluminum wire bonding machine resulted in a low yield of functional chips, which was solved by external bonding with a F&S Bondtec 5610 with gold ball head. In order to protect the chips and wires from dust, light and accidental mechanical stresses, both chips on a HICANN PCB are covered by a 3d printed cap. An alternative is a so-called *glob top*, an expoxy blob covering chip and wires. This approach has been used for the predecessor of HICANN with the drawback that the wires are not accessible to correct bonding issues afterwards. Additionally it was not clear if the mechanical stress during hardening of the epoxy worsens weak bonding wire connections. Therefore, in the testing phase, a removable cap is the preferred solution.

Two HICANNs on the same PCB can be connected to each other via L1 connections if the corresponding pads are bonded. The connection layout is different to the wafer postprocessing and would require adapted software support in the Hardware Abstraction Layer (HAL). We limited our experiments on this type of setup to run on a single HICANN. Due to the universality of the implementation, the same experiments can run on a wafer setup (section 5.3) as well.

5.2 CUBE SETUP

The Cube Setup with a new central PCB, labeled *cube-io*, was originally designed for prototyping and testing the Kintex-7 FCPs (see below). One cube-io connects to two SEBs, four Kintex-7 FCPs[1] and one frontend PCB holding several jacks for

1 One FCP connects to one of each SEB, the other two FCPs are not connected to HICANNs.

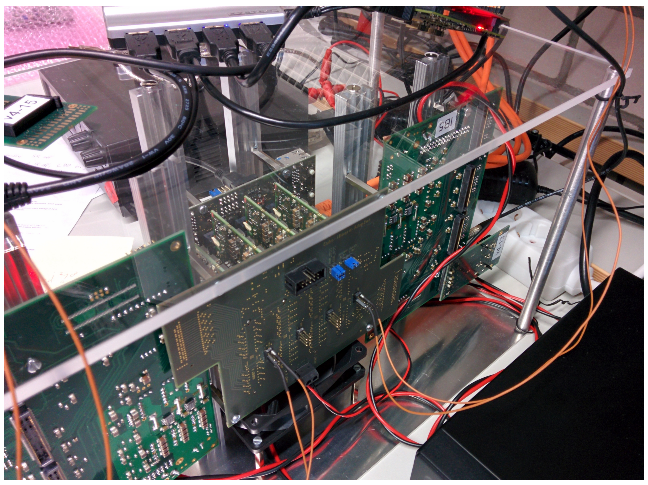


Figure 5.2: Cube Setup consisting of cube-io board in the center, with four FCPs mounted perpendicular and two SEBs left and right, holding one HICANN PCB each. AnaRMs are placed on top (cut off) and connected each to the trigger output of the central PCB as well as the two analog outputs of the SEB.

FPGA connectivity, including ethernet. As it allows to connect two SEBs to one FPGA PCB each, a total number of $2 \cdot 8$ HICANNs can be attached to a single Cube Setup.

This type of setup has replaced the Demonstrator Setup (section 5.1) for single-chip experiments. We built three Cube Setups – with PCBs and mechanical structures provided by TU Dresden – for evaluation of HICANN revision 4, which were used for HICANN experiments until the HICANN revision 4 wafer systems were completed. The setups are also used for wafer testing (figure 5.3) and connecting the first embedded wafer prototype (Güttler, 2017), a HICANN revision 2 wafer embedded inside a PCB.

5.2.1 *Kintex-7 FPGA PCB*

In the Kintex-7 FCP, the DNC became part of the FPGA design. Therefore, one FCP connects to one reticle on a wafer, i. e. 8 HICANNs, or up to four HICANN PCBs in case of the Cube Setup.

The design was ported from the Virtex-5 design and developed further from there (Müller, 2014). I2C functionality to set voltages on the SEB and configure the analog multiplexer as part of the previous design was dropped in this new design because the wafer system does not require this functionality. Supply voltages of the wafer setups are set by a Raspberry Pi (Hellenbrand, 2013). Similarly, we also realized setting the SEB voltages and analog multiplexer configuration on Cube Setups using a Raspberry Pi.

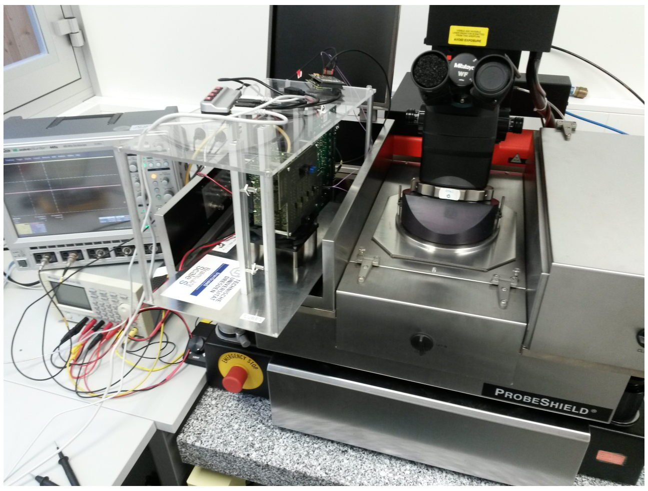


Figure 5.3: Cube setup connected to needle card inside a wafer probing machine. The needle card is positioned to contact a single HICANN of a wafer without postprocessing. This HICANN can then be accessed as if it was mounted to the SEB. Photo by Andreas Grübl.

5.3 WAFER SETUP

The first two wafer setup prototypes were built for 12 Virtex-5 FPGA PCBs (section 5.1.2) each. The final BrainScaleS system is built with 48 Kintex-7 FCPs (section 5.2.1).

On a wafer, 8 HICANNs are grouped into one reticle (figure 5.4). The physical dimensions of a single HICANN prototype chip, $5\,\mathrm{mm} \cdot 10\,\mathrm{mm}$, were predetermined by the block size of the MPW production. For wafer manufacture, the maximum area of exposure in this process is slightly above $2\,\mathrm{cm} \cdot 2\,\mathrm{cm}$. This area fits 8 HICANNs, removing edge structures and adding L1 interconnections between HICANNs on the reticle. A full wafer consists of 48 reticles, giving a total of 384 HICANNs per wafer. Connections for power supply, HS link, JTAG to each HICANN and L1 interconnection lines between reticles are added in a postprocessing step for wafer-scale integration (Husmann and Zoglauer, 2010; Güttler, 2017; Mauch, 2016).

Every system features $384 \cdot 2/8 = 96$ simultaneous analog readouts. The minimum required number of AnaRMs (with 8 channels each) to connect all 96 lines is $96/8 = 12$, limiting the possible simultaneous analog readouts to 12.

Supply voltages are provided by two PCBs named *PowerIt* and *AuxPWR* (Sterzenbach, 2014), similar to the SEB but designed for larger scale. An explosion drawing of a single wafer module is depicted in figure 5.5. Currently there are 20 assembled BrainScaleS systems. They were made accessible as part of the neuromorphic platform in the Human Brain Project (HBP) under the name Neuromorphic Physical Model version 1 (NM-PM1).

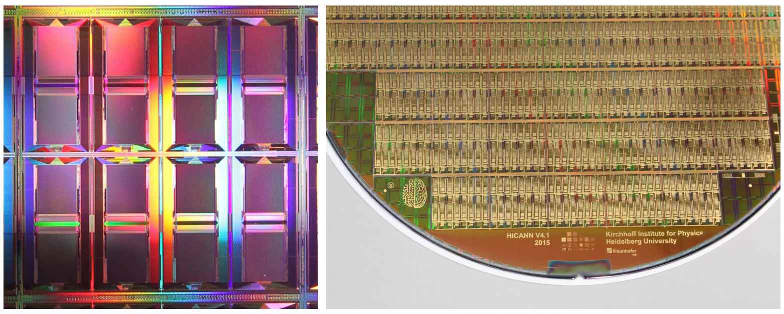


Figure 5.4: Left: Close-up of a single reticle, consisting of 8 HICANNs. Right: HICANN revision 4 wafer with postprocessing layer, adding connections for power, L1, and HS link. Photos by Andreas Grübl.

5.4 ANALOG READOUT MODULE

Part of each setup is a custom Analog Readout Module (AnaRM) to digitize the membrane voltage of individual denmems. Analog voltage output from HICANN ranges between 0 and 1.8 V. After 50 Ω termination and pre-amplification the signal is sampled by the Analog-to-Digital Converter (ADC) chip ADS6125 on the AnaRM, which is able to convert analog voltage from 0 V to 2 V to a 12 bit digital value. The clock frequency can be up to 125 MHz. A detailed analysis of this chip's performance can be found in Epp (2016). We are operating the ADC at 96 MHz because there is already a clock generator for Universal Serial Bus (USB) at 48 MHz = 96 MHz/2 present in the Spartan 6 field-programmable gate array

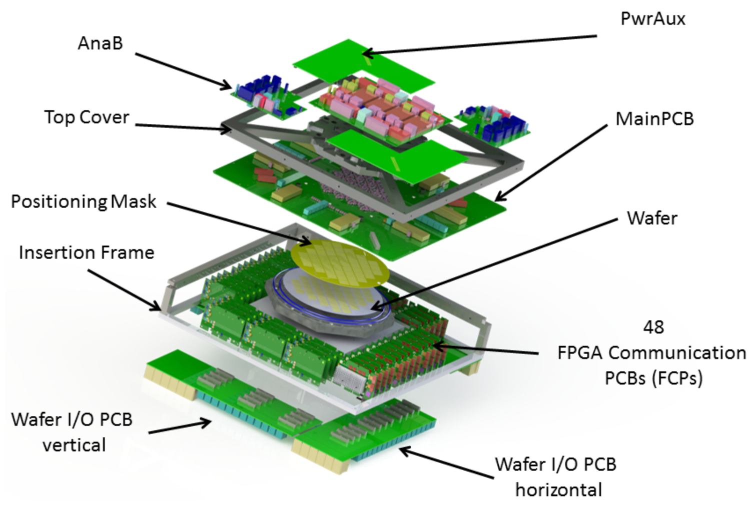


Figure 5.5: Explosion drawing of the components making up a wafer module. Below the actual wafer we see the 48 FCPs, communicating to one reticle each. AnaRMs (not in the drawing) are connected to the MainPCB by shielded ribbon cables. From (HBP SP9 partners, 2014), where each component is explained in detail.

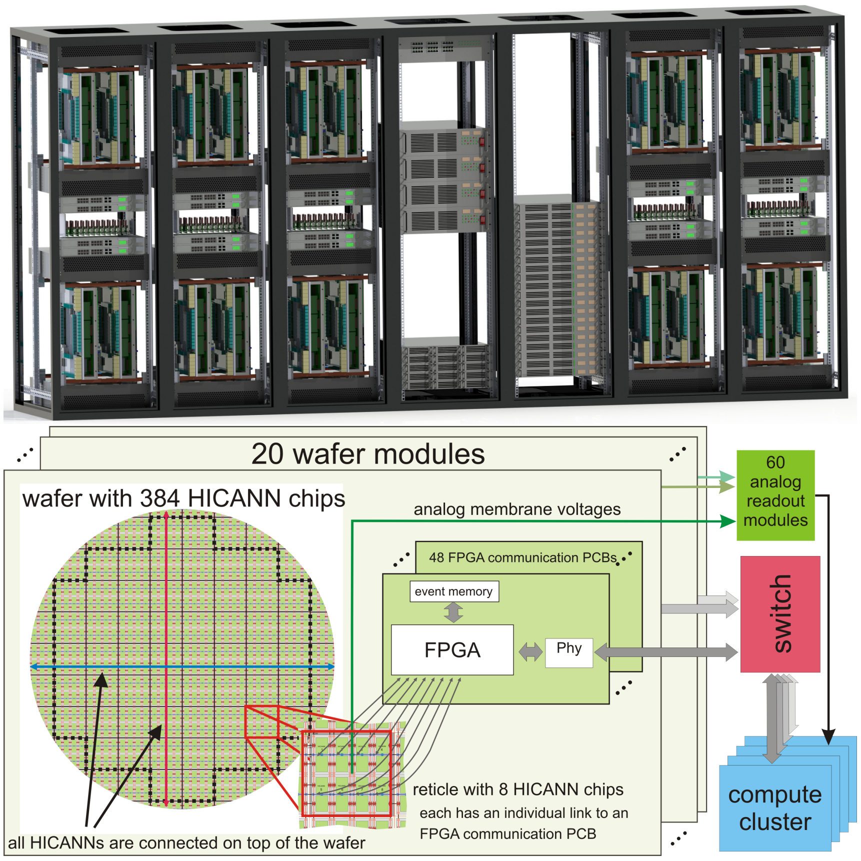

20 wafer modules
wafer with 384 HICANN chips
analog membrane voltages
60 analog readout modules
48 FPGA communication PCBs
event memory
FPGA
Phy
switch
compute cluster
reticle with 8 HICANN chips
each has an individual link to an FPGA communication PCB
all HICANNs are connected on top of the wafer

Figure 5.6: Rendering of the NM-PM1 machine. Racks are holding 20 wafer modules, 5 AnaRM crates consisting of 12 modules each, network switches, a compute cluster and disk storage. The bottom sketch shows how data is exchanged: HICANNs are communicating to FCPs, which have an ethernet link to a compute cluster. AnaRMs are connected via USB to a host computer and can be read from the compute cluster as well. From HBP SP9 partners (2014).

(FPGA), using less energy than a separate clock generator for 125 MHz. The voltage range from 0 V to 0.9 V after termination is covered by 11 bits. The ADC clock and HICANN clock are not synchronized and need to be re-aligned in software before analog traces can be compared to spike timestamps.

One module consists of the *Flyspi* FPGA PCB, which also contains the ADC, and the Analog Frontend PCB (AnaFP). The AnaFP features a multiplexer chain to select one of 8 channels. This channel is connected to a 50 Ω termination and a preamplifier to match the ADC's input voltage range (HBP SP9 partners, 2014). An earlier version of the AnaRM has been tested in Gorel (2013), the current version has been analyzed in Epp (2016). The transformation of raw values to voltages and its calibration has been explained in section 5.4.1.

One AnaRM is sufficient to connect both analog outputs of the SEB, as long as they are not needed simultaneously. The total number of analog outputs on a wafer

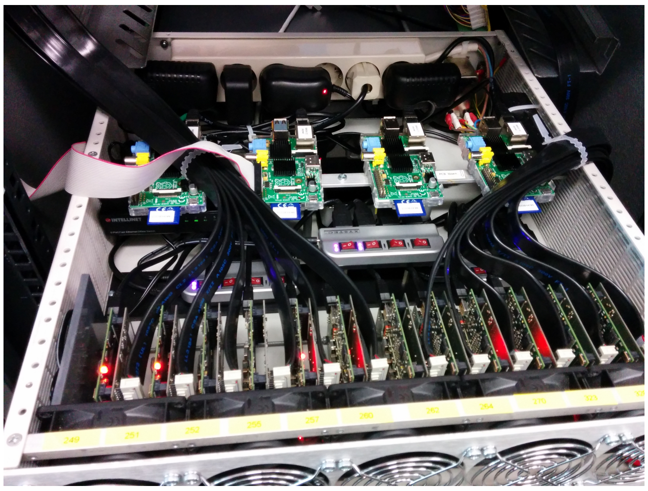


Figure 5.7: Crate holding 12 AnaRMs connected to 3 USB hubs for power and data transmission. The crate also contains 4 Raspberry Pis, controlling one wafer module each.

system is two per reticle, i. e. $384/8 \cdot 2$. Each AnaRM has 8 channels. If we need to connect all analog outputs of a full wafer to an AnaRM channel, for example for denmem characterization, $384/8 \cdot 2/8 = 12$ AnaRM are required. Part of the BrainScaleS system are crates of 12 AnaRMs (figure 5.7), which can be attached to one wafer.

In an integration test for the AnaRM, two floating gate cell voltages are connected to both analog outputs for each HICANN. From the resulting trace of one assumably constant voltage, we determine the standard deviation σ and the peak-to-peak differences of traces (figure 5.8).

5.4.1 *ADC Calibration*

Before we can use analog voltage traces from any AnaRM, each board needs one initial calibration. The XILINX Spartan 6 FPGA on the AnaRM stores raw ADC samples in memory, which can then be read out via USB. The function mapping each raw value to a voltage has the form

$$V_{\text{source}} = c_0 + c_1 \cdot u + c_2 \cdot u^2, \tag{5.1}$$

with raw digital values $u \in [0, 4096]$ and typical coefficients specified as $c_0 = 2.0\,\text{V}$, $c_1 = -6.6 \cdot 10^{-4}\,\text{V}$, $c_2 = 5.7 \cdot 10^{-9}\,\text{V}$ (HBP SP9 partners, 2014).

We connect all eight multiplexed channels to a voltage source, a Keithley 2635 source meter, in series with a $50\,\Omega$ resistor to match the expected termination of the HICANN analog output to perform the initial calibration. The measured termination in HICANN is actually $41\,\Omega$ to $44\,\Omega$ (Koke, 2016). We measure the raw digital value for several input voltages for each channel. The calibration curve is

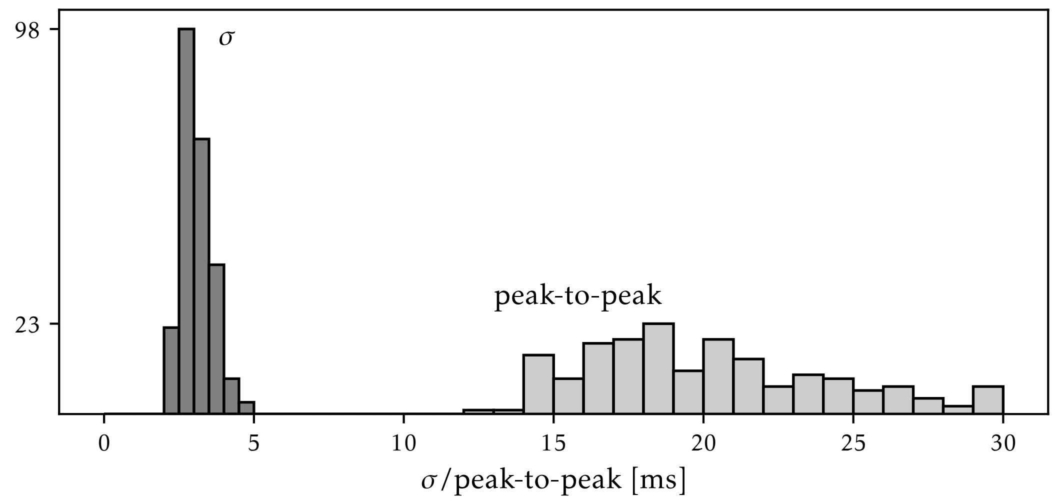


Figure 5.8: Readout noise statistics. In an integration test for the AnaRM, two floating gate cell voltages are connected to both analog outputs for each HICANN. From the resulting trace of one assumably constant voltage, we determine the standard deviation σ and the differences from highest to lowest value (peak-to-peak) of traces for 240 HICANNs on Wafer 21 with module W_G06. The results are similar to those published in Koke, 2016 for a different wafer.

a least squares fit to equation (5.1). More advanced functions have been investigated (Gorel, 2013) but equation (5.1) turned out to be sufficient. The actual distribution over 85 calibrated AnaRMs with 8 channels each is shown in figure 5.9.

Similar ADC boards like the Red Pitaya (Epp, 2016) store their coefficients on the board. Coefficients for AnaRM are stored externally with the corresponding serial number and loaded by the measurement software.

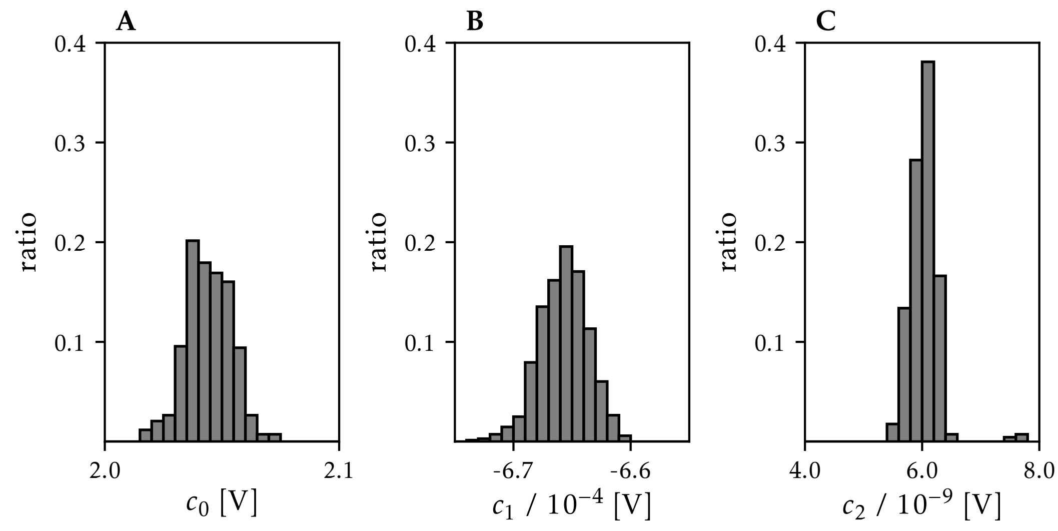


Figure 5.9: Distribution of actual ADC calibration coefficients for equation (5.1) near the typical values $c_0 = 2.0\,\text{V}$, $c_1 = -6.6 \cdot 10^{-4}\,\text{V}$, $c_2 = 5.7 \cdot 10^{-9}\,\text{V}$ specified in (HBP SP9 partners, 2014) over 85 calibrated AnaRMs with 8 channels each ($85 \cdot 8 = 680$ values). The outliers around $c_2 = 7.5 \cdot 10^{-9}\,\text{V}$ are all 8 channels belonging to the same board with serial number B201258.

6 CONTROL SOFTWARE

Configuring the various components of the BrainScaleS system requires many individual software components, most of which are hidden for an experimenter who wants to emulate neural networks.

Each BrainScaleS system and its components are controlled by host computers, which communicate with the system's FCPs via Ethernet (Müller, 2014). Additionally the supply voltages are set and monitored via I2C on a Raspberry Pi (Hellenbrand, 2013). The AnaRM trace data can be read from their USB host via Ethernet as well.

The preparation of a network experiment involves several layers of software (figure 6.1):

- A high level network description is written by the experimenter in *PyNN*, a Python package for simulator-independent specification of neuronal network models (Davison et al., 2008; Brüderle, Müller, et al., 2009). The PyNN implementation for the BrainScaleS system is called *PyHMF*.

- The network is mapped onto one or multiple HICANNs and their available connectivity options by our mapping software, *marocco* (section 6.1). For each analog circuit, the mapping software is applying the individual neuron's characteristic functions from a database, which was created by our calibration framework for experiments (cake) (section 6.2).

- The mapping result is applied to our hardware interfaces, StHAL and HALbe (section 6.2.2).

- HALbe communicates via the ARQ communication layer to FCPs, which communicate to HICANN. Spikes are transmitted back in the opposite direction.

- FCPs or HALbe trigger AnaRM measurements, analog traces are read back via HALbe.

Python bindings for each software layer written in C++ are generated automatically (Koke, 2016). These bindings lower the barrier to use the full software stack for beginners and speed up prototyping of new ideas. The calibration framework is written entirely in Python, allowing to take advantage of scientific toolkits such as *numpy* (Walt, Colbert, and Varoquaux, 2011), *scipy* (Jones, Oliphant, Peterson, et al., 2001–) and *pandas* (McKinney, 2010; McKinney, 2012) for data processing as well as *matplotlib* (Hunter, 2007) and *Jupyter* (Perez and Granger, 2007) for visualization, quick evaluation and organization.

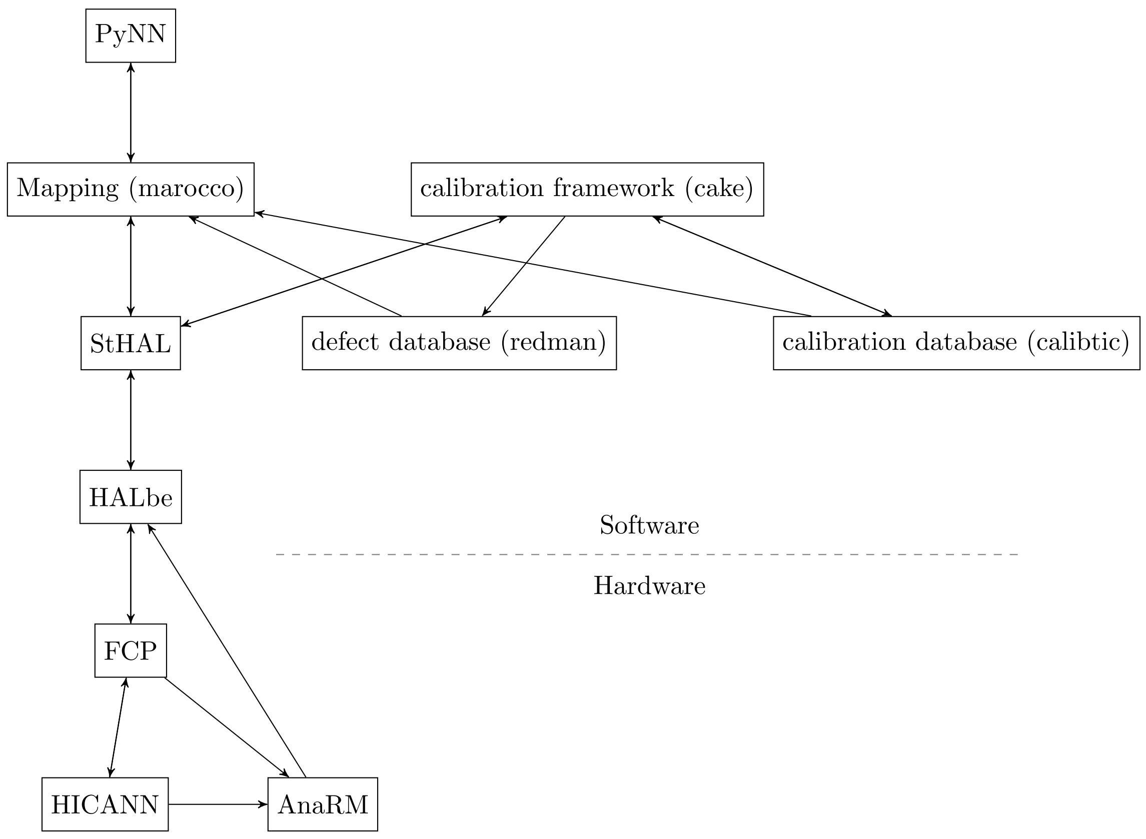


Figure 6.1: Flow of information through the software layers and hardware communication involved in a neural network experiment. Arrows indicate data being provided to another layer. A neural network described in PyNN is mapped to the Brain-ScaleS system by the mapping tool *marocco*, which configures HICANNs via their corresponding FCP through the stateful (StHAL) and stateless (HALbe) hardware interface. Mismatch correction data and defect information were generated earlier by the calibration framework (cake) and stored in databases (calibtic and redman, respectively). This information is used during the mapping process. After the experiment, analog traces from the AnaRMs and spike events from HICANNs/FCPs are reported back up to the PyNN interface.

6.1 EXECUTING NEURAL NETWORK EXPERIMENTS

A common interface to define neural networks – called PyNN (Davison et al., 2008; Brüderle, Petrovici, et al., 2011) – has been developed to allow running the same network definition on any simulator backend supporting this interface and the chosen features.

The default level of abstraction in PyNN is a population of neurons. It is also possible to select or modify a subset of a population or a combination of several populations or subsets. Synaptic connectivity between populations is defined with the type of connection. One-to-one connects each neuron in a population to one neuron in another population. There are also one-to-N, N-to-one and all-to-all connections. Additionally there are probabilistic connections with constant or distance dependent probability, the latter requires to add a spatial position to each neuron.

Our implementation of the PyNN interface, *PyHMF*, supports LIF and AdEx neurons. The available parameter range for physical emulation is given by the chosen transformation (section 4.2.5) and the range and precision that is available from the electrical circuits. The maximum number of synapses is determined by the number of physical synaptic input lines and their connectivity options. During the design phase, parameter ranges, bandwidths and number of synapses have been estimated from available biological data and network models. A short overview intended for modelers has been created as living document (Müller et al., 2014). If it turns out that essential parameter ranges are not covered by the physical model, it might be possible to include the required values in future designs. An analysis of parameter range requirements and benchmarks can be found in (Müller, 2017). In the long term, we expect the advantage of accelerated time and – in future versions – learning in accelerated time to be essential for gaining insights into biological learning processes at a large scale in finite time.

The neural network experiment described in *PyNN* is mapped to hardware components by the mapping tool *marocco*. Based on the hierarchical model description of a neural network, neurons and L1 connectivity is placed on the BrainScaleS system while taking physical constraints into account. The marocco API allows to manually select which HICANN should be used to place populations. Model neurons consist of one or several interconnected denmems, depending on the required number of synaptic input connections. The last step during the mapping process is the parameter translation described in section 4.2.5. Analog circuit parameters are chosen according to the characterization created by the calibration framework introduced in the following section.

6.2 CALIBRATION FRAMEWORK

The main task of the calibration framework for experiments (cake), is to characterize individual components and provide the gained information to the mapping process.

6.2.1 *Core Concepts*

There are two types of information on the circuit behavior we can obtain: membrane voltage traces and spike timestamps. Parameters like time constants or conductances can not be directly measured, but can be deduced from the voltage trace to a certain extend. The methods are discussed in detail in chapter 7.

There are two membrane voltage outputs per HICANN (section 4.1.5), which means for a wafer setup, 96 outputs are available simultaneously. With the currently available AnaRM, 12 voltage traces could be read out simultaneously as only one of 8 channels can be recorded at the same time (section 5.3). This will change with the successor of the current AnaRM, called Ananas, which is still under development (Ilmberger, 2017).

For debugging purposes, additional voltages can be read out: the output voltage for any floating gate cell, fire signal and voltage of the DLL lock in the synapse driver for one synapse driver on each half of HICANN (HBP SP9 partners, 2014). These voltages are not used for neuron circuit calibration, but have proven very useful in the commissioning phase.

The calibration framework is built in a modular fashion. This way we avoid code duplication and provide options for customization of existing methods. The general approach is as follows:

1. Generate an experiment with custom configuration and steps that change parameters.

2. Record ADC and/or spike data.

3. Run analysis for each step and denmem.

4. Merge step results to obtain calibration curve.

Experiments consisting of several parameter sweep steps are planned by an *experiment builder* module. It can be customized for each method, basic customization would be a different trace analysis routine, but also allows special functionality like an initial measurement to determine the Inter-Spike Interval (ISI) with zero refractory period at a given parameter set for refractory period calibration (section 7.10). The experiment can be repeated to average out trial-to-trial variation, the duration of the analog measurement can be adjusted if longer measurements are required, for example for adaptation. A sweep step usually changes only the parameter that is characterized. Other parameters can be updated simulateously, for example if they depend on the sweeped parameter's value. We can also change the speedup configuration (section 4.2.1) between steps.

All experiment steps and repetitions planned by the experiment builder module are stored on disk, allowing to resume experiments after they were interrupted. This also allows to reconstruct the experiment configuration after the experiment has finished. Optionally, the recorded analog traces can be stored for further analysis. Traces which have already been analyzed by the framework are not stored by default as this would require large amounts of disk space and increase the total runtime. The *analyzer* module chosen for an experiment can deduce a single or several quantities from a membrane voltage trace, for example:

- mean and standard deviation,

- absolute minimum and maximum value,

- minimum and maximum values averaged over multiple intervals to reduce noise effects,

- baseline of a trace (requires non-zero refractory period),
- ISI or spike frequency and amplitude,
- slopes of rising and falling edge.

It is trivial to add additional or derived analyzers providing additional information. These quantities are then interpreted by a *calibrator* module, which fits an appropriate function mapping the desired model parameter to the corresponding hardware configuration value. The calibrator module also defines a value range (domain) over which the function is valid. This is used to limit electronic parameters to the range that the circuit was designed for. The application of this functionality is demonstrated in chapter 7.

6.2.2 *Hardware Interface*

The calibration framework uses same hardware abstraction layers that are also used by the mapping tool. Additionally it is possible to use transistor level simulation instead (section 4.3). The Hardware Abstraction Layer Backend (HALbe) provides low-level access, performing the required bit formatting and communication. This level is also suitable for early testing and debugging purposes. The Stateful Hardware Abstraction Layer (StHAL) builds on top of HALbe, remembering the state that already has been configured (Koke, 2016). It also provides higher level functionality for tasks like analog voltage readout. As the BrainScaleS system consists of many identical components such as HICANNs or denmems, there is a well-defined coordinate system to address a single unit (Jeltsch, 2014).

The early proof-of-concept implementation of a calibration framework (Schwartz, 2013) was still based on the first evaluation scripts for HICANN. Hardware communication in cake is based entirely on StHAL. This makes the transition from a single HICANN to wafer-scale or even multi-system calibration runs straightforward: iterate over coordinates and spawn jobs to the job queue. As long as resource dependencies are properly managed (Müller, 2014), this already provides sufficient parallelism. The hardware abstraction allowed us to run on all types of setups (chapter 5) by just changing the coordinate configuration, including the embedded wafer prototype (Güttler, 2017).

6.2.3 *Transformation Storage*

As discussed in section 4.1, several neuron circuit parameters can be adjusted to achieve the corresponding AdEx model parameter behavior. Their relation is determined by the calibration framework and stored as transformations in a database. This information is stored in a type-safe binary format which is written by the calibration framework to avoid user error and increase performance.

The result provided by the calibration framework is a lookup from desired analog behavior to programmable discrete 10 bit floating gate parameters. A unified software interface for both calibration and mapping (calibtic) allows to apply these transformations to a model neuron with parameters defined in PyNN, resulting in hardware parameters that will lead to the desired model behavior.

The next chapter presents how these transformations are obtained for each individual circuit.

6.3 METHODOLOGY

We have contributed to establish a workflow that supports maintenance of a system at the scale and complexity of the BrainScaleS system in the long term, with a few central points:

- Establishing unit tests and integration tests, code review and style guidance, leading to higher code quality (McIntosh et al., 2014), stability and robustness, as well as readability, and thereby improving collaboration.
- Improved readability of code, documentation and specification, leading to lower chance of remaining errors and higher chance of future improvements (Fowler and Beck, 1999).
- Improved verbosity on all software layers to increase usability, especially for students and external users who are new to the field. It also allows experienced users to figure out what is going on faster, increasing productivity and reducing mistakes.
- Documentation and visualisation of understood and/or corrected errors helps to deal with similar occurrences in the future and is a valuable resource for newcomers.
- The simulation workflow described in section 4.3 makes it easier to create reproducible simulations directly combined with analysis of the results.

As this result is often not mentioned explicitly, we want to emphasize its importance. The overall usability and readability has already enabled many students to contribute (Pilz, 2016; Kungl, 2016; Epp, 2016; Friedrich, 2015; Alevi, 2015; Nonnenmacher, 2015; Schmidt, 2014; Billaudelle, 2014; Tran, 2013; Klähn, 2013; Pape, 2013; Ziegler, 2013). Thanks to thorough code review the code quality is not reduced by lesser experienced or rushed contributors while all contributors learn how to improve their work at the same time.

While the above points may not be scientifically rewarding, I consider them essential for the long term success of a project at this scale. This leads us from the software overview to its application: In the following we use the calibration framework to characterize analog behavior of single denmems, providing the foundation for network experiments.

Part III

ADEX CALIBRATION

7 PARAMETER ESTIMATION METHODS

Before network experiments can be emulated on the BrainScaleS system, we need to know which neuron circuit parameter settings will result in the desired model parameters.

The methods developed to characterize each sub-circuit are introduced in this chapter. We evaluate the presented methods both in transistor level simulations (section 4.3) and hardware measurements, if applicable.

7.1 EXPERIMENTAL SETUP

In order to characterize the circuits, we typically sweep over a range of values of the circuit parameter corresponding to the desired model parameter (section 4.1) and analyze the resulting behavior. The resulting transformation, mapping the model parameter in the scaled hardware domain (compare section 4.2.5) to the required analog parameter, is known as calibration function. The calibration function of the ideal denmem without variation, obtained from transistor level simulations, is called *ideal calibration*. It can be used as an approximation as long as the actual circuit behavior of an individual denmem, affected by mismatch, has not been characterized. Each denmem needs to be characterized individually to be able to configure the observable model parameters as close to their desired value as possible.

We expect the error of an observable to consist of a systematic error and a statistical error, dominated by the trial-to-trial variation of the FG cell output:

$$\sigma^2 = \sigma_{\mathrm{tt}}^2 + \sigma_{\mathrm{sys}}^2 \tag{7.1}$$

Increasing the number of points will increase both precision and experiment duration. If the number of points is small, the offset of a single point due to trial-to-trial variation of the corresponding floating gate (FG) cell output will have more impact on the resulting linear regression fit.

7.1.1 *Hardware Measurements*

Compared to transistor level simulations, hardware measurements require more advanced algorithms due to additional effects such as readout noise (figure 5.8) or trial-to-trial variation (figure 4.8). FG current cells are expected to provide an output current above $0\,\mathrm{A}$ when configured to 0, estimated at $20\,\mathrm{nA}$ to $50\,\mathrm{nA}$. We take the resulting effects into consideration for the presented methods. In our transistor level simulations the FG cells are replaced by ideal current and voltage sources.

The minimum output current is configurable and set to $50\,nA$ by default. Several methods depend on a well-known configuration, adding a dependency on parameters being calibrated before the method can be used. When the parameter value is used in a calculation to determine another parameter, the error on this parameter impacts the precision at which we can determine the dependent parameter via error propagation. Such dependencies also specify the order of execution. For example, membrane time constant calibration requires knowing the values of V_t and V_{reset}. Calibrating these parameters earlier allows to set the parameters to a desired value, up to the precision of trial-to-trial variation plus systematic error. Methods presented in this chapter are presented in the order of typical execution during a full calibration run, an order that does not have dependencies on later methods.

As writing new values to the floating gate cells requires the largest amount of time, i. e. between 15 and 30 seconds depending on the floating gate controller settings, we configure all 512 denmems at once and only switch the analog output configuration between measurements of individual denmems.

7.2 DISABLING TERMS

When we analyze any term, we want to minimize the currents from other terms. The synaptic input, adaptation term and exponential term are typically disabled in each method as long as they are not currently being characterized. The parameters required to disable the terms are listed in table 7.1

Setting I_{conv} to $0\,nA$ disables the OTA in the synaptic circuit. As the current does not actually reach zero (see section 4.2.2), we additionally set the maximum offset potential $V_{convoff}$ to make sure the voltage difference at the OTA inputs is negative, also resulting in zero current output. The reversal potentials E_{syn} will have no effect in this setting, choosing them close to the resting potential would lead to smaller effects even if the resistor was enabled. Due to their non-symmetric design in HICANN revision 4 the excitatory potential must be above E_l, the inhibitory potential must be below E_l. The synaptic time constant plays no role in disabled synaptic inputs.

Disabling the adaptation current onto the membrane is achieved by turning off the OTA for the conductance a, which will minimize w in equation (4.3). The conductance bias $I_{gladapt}$ is set to digital 0, resulting in the smallest possible conductance close to zero. The conductance bias I_{radapt} is set to maximum conductance, such that the voltage V_w follows V_m quickly, leading to no differential voltage at the OTA for a when V_m is at E_l. The other mechanism that is increasing w is the increase by b in equation (2.9), which is disabled by setting I_{fire} to zero.

Simulations investigating how to disable the exponential current have been performed in Stöckel (2014). The OpAmp bias I_{bexp} in figure 4.5 is set to 0, minimizing its output current. The threshold at which the exponential term becomes active is controlled by V_{exp}. Setting it to maximum voltage effectively disables the exponential term.

synaptic input		adaptation term		exponential term	
parameter	value	parameter	value	parameter	value
$I_{\text{convx/i}}$	$0\,nA$	I_{gladapt}	$0\,nA$	V_{exp}	1.8V
$V_{\text{convoffx/i}}$	$1.8\,V$	I_{radapt}	$2500\,nA$	I_{bexp}	$0\,nA$
E_{synx}	slightly above E_l	I_{fire}	$0\,nA$		
E_{syni}	slightly below E_l				
$V_{\text{syntcx/i}}$	any				

Table 7.1: Parameter values to disable synaptic input, adaptation term and exponential term.

7.3 PREVIOUS WORK

Some of the methods have been presented for *Spikey* (Brüderle, 2009; Pfeil, 2015), the predecessor of HICANN: ADC calibration (section 5.4.1), analog output calibration (section 7.4), membrane time constant calibration (section 7.8) and refractory period calibration (section 7.10). Building on this work, early methods for HICANN have been presented in Schwartz (2013) and ported to the calibration framework (section 6.2) by the author. When applying the methods to hardware measurements, they usually required modifications to cope with the effects introduced above. Additional methods and enhancements have been developed and refined for the previous chip revision, HICANN revision 2, and were published in Schmidt (2014). The latest implementation has several other contributors, especially Dominik Schmidt, Christoph Koke and Sebastian Schmitt, and more recently Alexander Kugele. An evaluation of the differential equation fitting based method to determine the adaptaion coupling parameter a and time constant τ_w has been performed in Friedrich (2015). Previously unpublished methods in this work are the full characterization of the adaptation and exponential term. The presented methods are integrated in our system software such that they can be applied to any experimental setup. They are built in a configurable manner, allowing to quickly make changes or build a custom calibration tailored to optimize for short measurement duration or high precision.

7.4 OUTPUT BUFFER OFFSET

Output buffer offset calibration is required before analyzing any other membrane voltage traces because the variation of the membrane voltage readout buffer (section 4.1.5) is causing a fixed voltage offset for each denmem. The following method has been introduced in Millner (2012). We want to determine the output offset from the actual potential of each individual denmem. All 64 interconnectable denmems are connected, we measure their common membrane voltage while keeping them

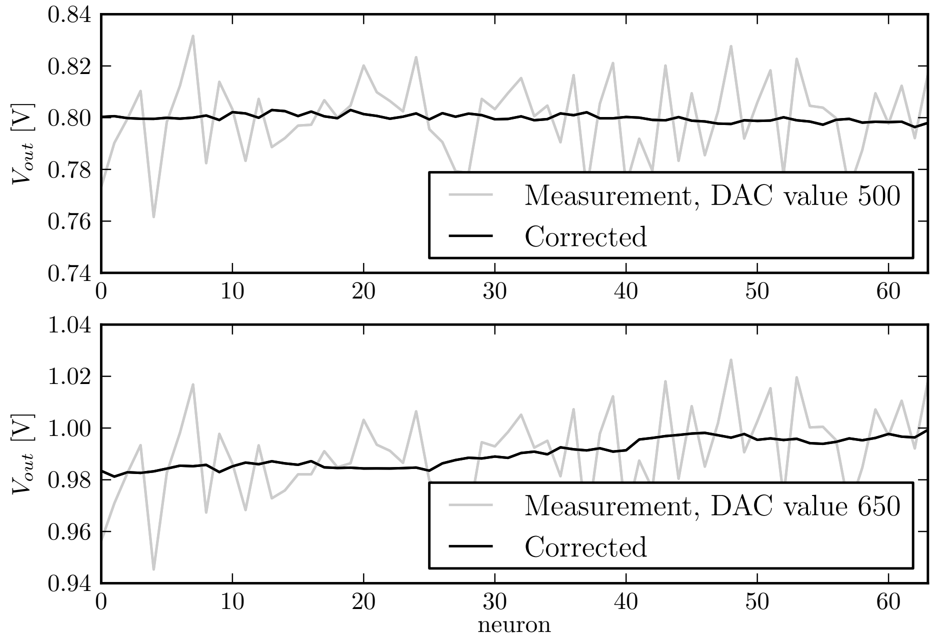


Figure 7.1: Output buffer offset as published in Millner (2012). The mean output voltage of all denmems is independent from the denmem position for $E_l \approx 0.9\,V$ (500 DAC), but measured values show a position dependency for lower or higher E_l configurations, which is not present in our measurements (figure 7.2) on the current system.

at the resting potential. Assuming that the resistance of the interconnection is negligible, and the output buffer offsets are distributed symmetrically around zero, we can measure the voltage on each denmem and determine the offset from the mean. According to Millner (2012), the mean is independent from the denmem position for $E_l \approx 0.9\,V$, but measured values showed a position dependent value for lower or higher voltages (figure 7.1). One possible explanation (Millner, 2012) is a position dependent variation of the leakage OTAs due to the manufacturing process. These OTAs are calibrated after offset calibration. Another explanation would be a supply voltage drop on the power rail (Millner, 2012). The first cause can be avoided by using another common voltage of the interconnected denmems that can be measured: V_{reset} as a global parameter is shared by every fourth denmem of the interconnected rectangular block. We are analyzing the baseline of a trace when denmems are configured with a resting potential above the spike threshold and long refractory period. This method works equally well (Schmidt, 2014).

However, this undesired position dependency could not be observed with the current system: We measure the output buffer offset at three different values of E_l, $0.7\,V$, $0.9\,V$ and $1.1\,V$, interconnecting a each block of 64 denmems. E_l is not calibrated yet, the average is expected to deviate from the target values. We do not

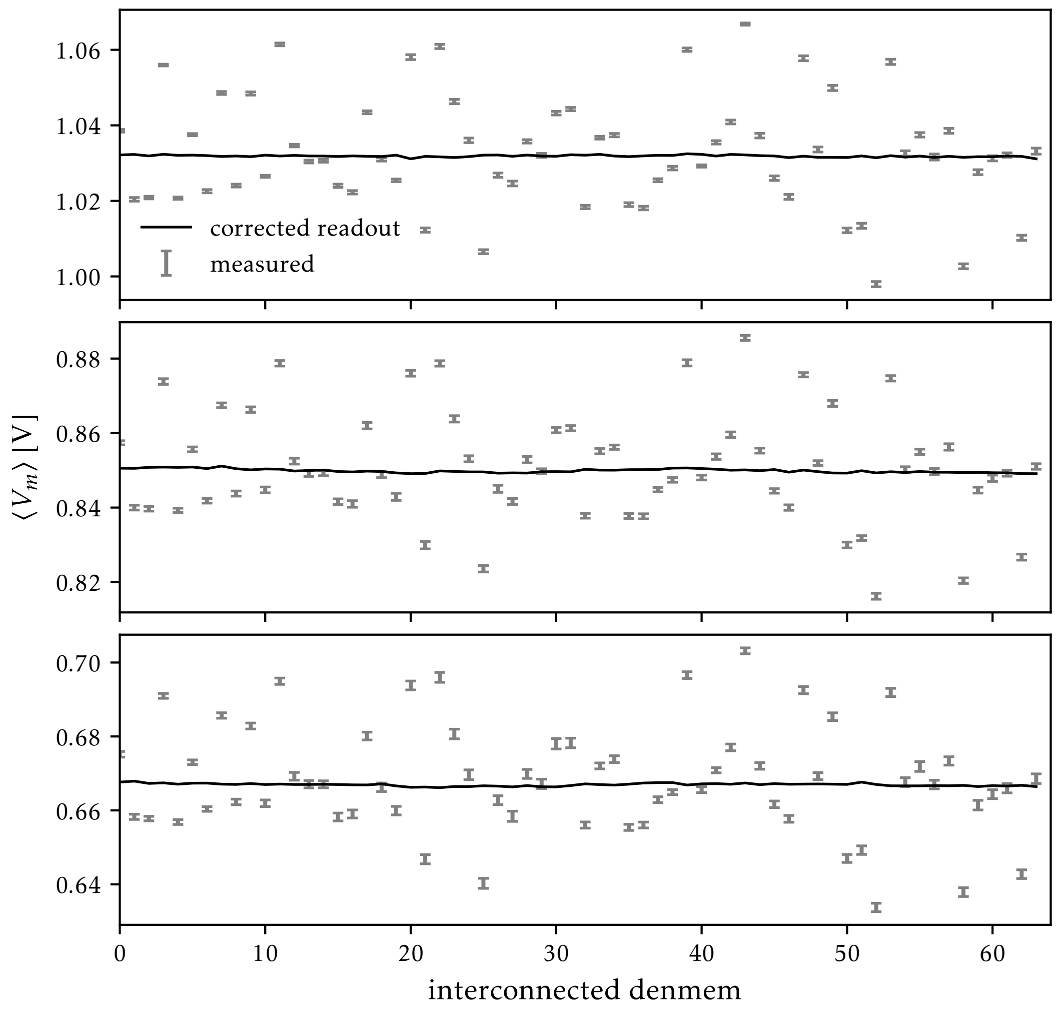


Figure 7.2: Exemplary output buffer offset measurement for one block of 64 interconnected denmems at different uncalibrated resting potentials $E_l = 0.7\,V$ (bottom), $0.9\,V$ (middle), $1.1\,V$ (top). Measured offsets (gray error bars) are averaged from 10 repetitions, the error bars show the standard deviation. The black line shows a corrected readout potential for each denmem from a single repetition, which shows only very small variations. In contrast to the measurement presented in Millner, 2012 we do not observe a denmem position-dependency for resting potentials lower or higher than $0.9\,V$.

observe any denmem position dependency (see figure 7.2), therefore no further workaround is required. We still calibrate the output buffer offset at the operating point of $0.9\,V$ in the default configuration.

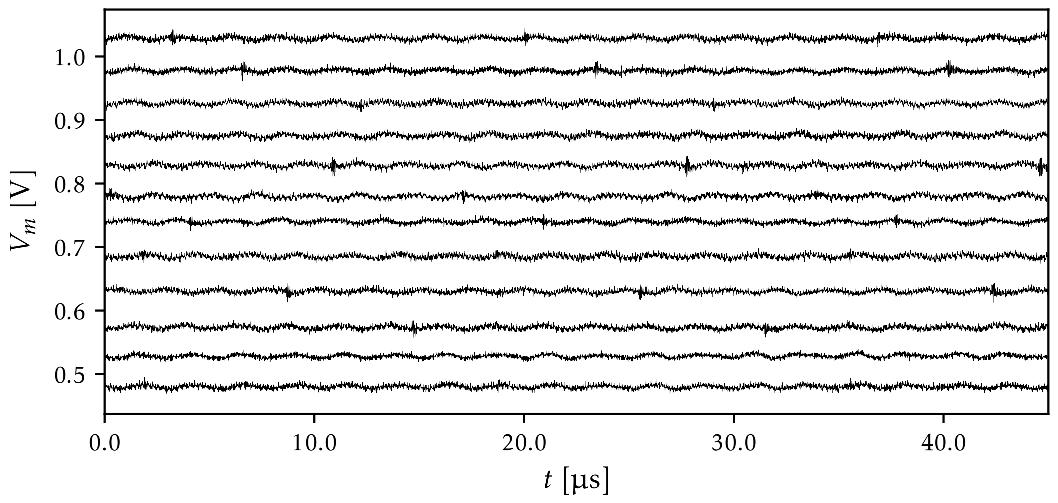


Figure 7.3: Exemplary membrane voltage traces for E_l calibration. Shown are 12 different target values from 0.5 V to 1.1 V. All traces show readout noise, which is averaged out when calculating the mean membrane voltage. Several traces show additional periodic readout noise peaks, probably caused by switching regulators in the system. During previous improvements in the BrainScaleS system, the noise has been reduced further.

7.5 RESTING POTENTIAL

After the output buffer offset is compensated, we can calibrate the resting potential E_l for each denmem. We configure the denmem to receive no current from the synaptic inputs and keep the resting potential below the spike threshold. Adaptation and exponential term are disabled as well (see section 7.2). Without any input, the membrane potential stays at $V_{\text{rest}} = E_l$. The measured membrane voltage trace (figure 7.3) shows readout noise that we assume is not present on the membrane capacitance itself. The averaged voltage is the measured resting potential. It shows a linear dependency on the configured DAC value (figure 7.4). In order to increase precision, due to the linear depence we can either measure additional DAC values or increase the number of repetitions for each value. Trial-to-trial variation of the floating gate output will result in a non-compensable spread of approximately 5 mV to 10 mV (figure 4.8).

7.6 THRESHOLD POTENTIAL

The spike threshold potential V_t can be determined from any spiking membrane voltage trace. A simple way to get a denmem to spike without involving the current stimulus or the synaptic input, which is not calibrated at this point, is to set the

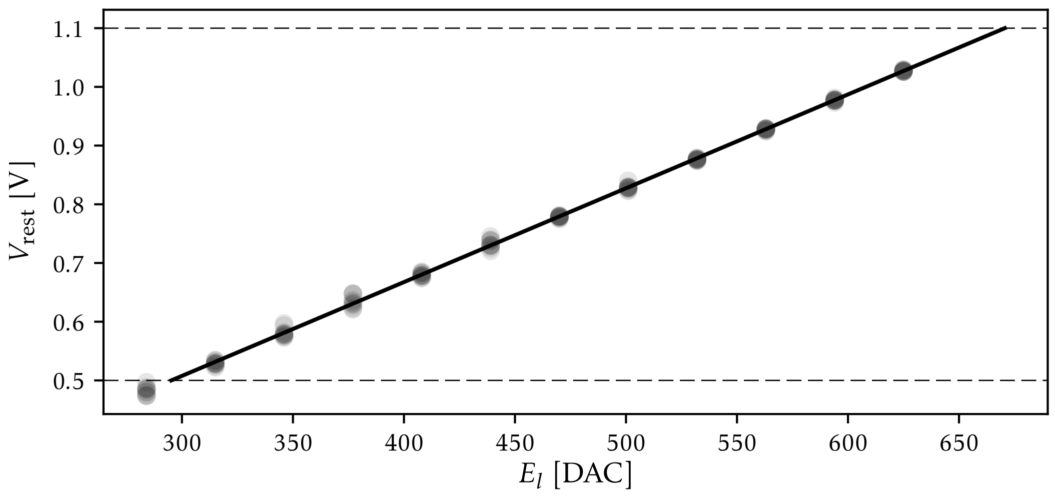


Figure 7.4: Resulting resting potential for different DAC settings of E_l. In this example, each step was measured with 10 repetitions, the resulting trial-to-trial variation is depicted as individual points. The transformation from desired resting potential to required E_l setting, resulting from a linear regression fit, is shown as solid line. Its start and end mark the target domain between 0.5 V and 1.1 V and corresponding DAC settings.

resting potential above the threshold potential. Other terms remain disabled as in the resting potential calibration.

The membrane voltage trace is subject to readout noise (see figure 7.5), which can add to the maximum at a spike. Therefore we can not directly use the maximum value of the trace as the spike threshold. This would yield a value which is too high, as at several spikes the noise adds to the maximum. We instead detect all peaks and average the maximum voltages at each peak to cancel out the readout noise. Because the maximum of other peaks is reduced by the noise, we expect the average maximum to be closer to the real V_t. The resulting threshold is depicted in figure 7.5. Here you can also see that we chose no refractory time and a fast membrane time constant in order to collect as many spikes as possible in any recording interval.

In this method we neglegted that the ADC sample does not always lie exactly on the peak, adding a systematic error. We estimate this error ϵ_t by assuming a linear slope of the membrane potential rising from V_{reset} to V_t:

$$\epsilon_t = \frac{V_t - V_{\text{reset}}}{T_{\text{ISI}}} \cdot \Delta t \tag{7.2}$$

with a time interval between two samples of $\Delta t = \frac{1}{f_{\text{ADC}}}$. With $V_t - V_{\text{reset}} = 200\,\text{mV}$, $T_{\text{ISI}} = 1.5\,\mu\text{s}$ and $f_{\text{ADC}} = 96\,\text{MHz}$ we get $\epsilon_t = 1.39\,\text{mV}$. Choosing a longer membrane time constant, resulting in a longer ISI, will increase the probability that the

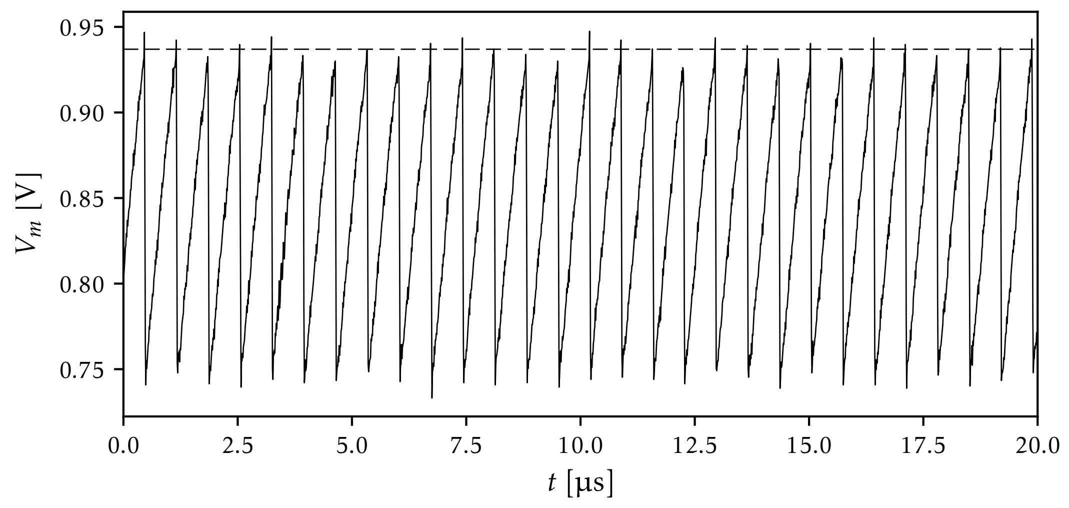


Figure 7.5: Exemplary section of a V_t calibration trace. The actual recording duration is longer (100 µs). As the membrane voltage trace is subject to readout noise, we detect all peaks and average the maximum voltages at each peak. The resulting threshold is plotted as dashed line. To average over as many spikes as possible in any recording interval, we chose no refractory time and a fast membrane time constant.

maximum ADC sample of a single spike is close to the actual peak and reduce the systematic error.

There is a linear dependency from the configured DAC value to the resulting threshold voltage (figure 7.6). The measured points are chosen to lie within the expected range of configurable threshold voltages. The resulting transformation is stored without domain boundaries, however the maximum threshold should remain below 1.2 V (compare section 4.2.5).

7.7 RESET POTENTIAL

The reset potential V_{reset} could be determined in a similar manner as the threshold potential: Our analyzer provides the mean over each minimum voltage following a peak. For a non-zero refractory period the reset potential is present over many ADC samples. Taking readout noise into account, using the minimum voltage during the refractory period after each spike introduces a systematic error: After averaging over each spike the result is shifted down in the order of the standard deviation of readout noise (lower line in figure 7.7). This systematic error is avoided by determining the baseline of the voltage trace after a spike instead (upper line in figure 7.7).

As V_{reset} is a parameter shared by all denmems of a FG block, the measured values of all denmems in a block are averaged for a point in figure 7.8.

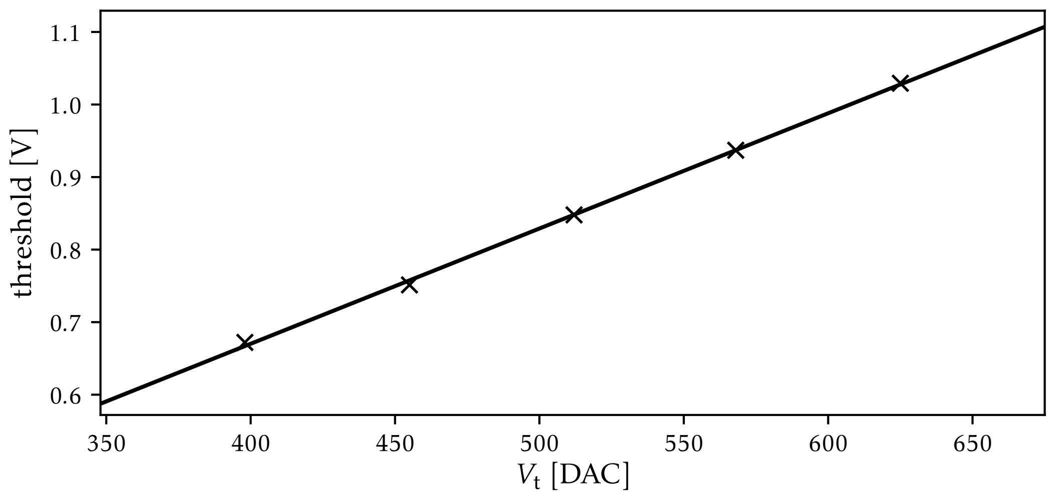


Figure 7.6: Exemplary V_t calibration result. The spike thresholds resulting from several V_t setting steps are used for linear regression. In this example only a few steps were configured with a single repetition to achieve a short experiment duration.

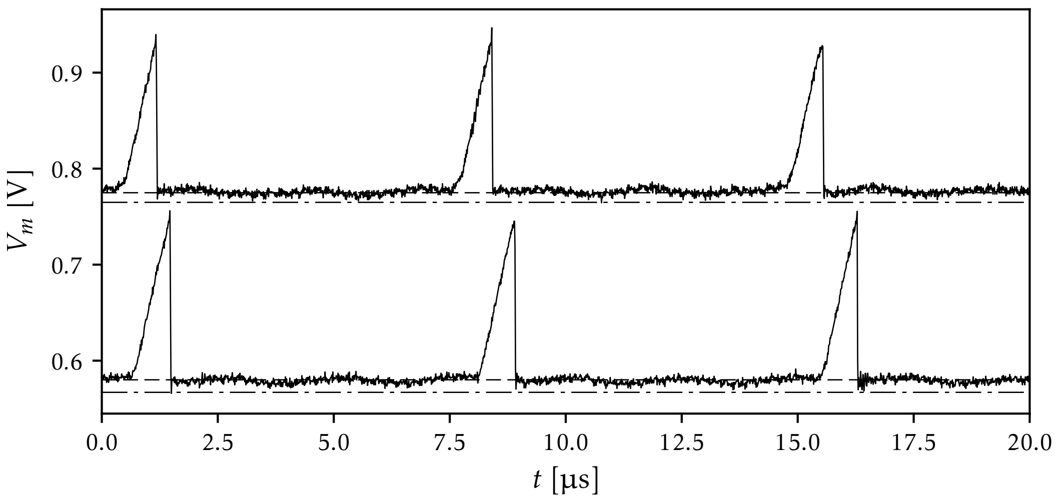


Figure 7.7: Exemplary V_{reset} calibration traces for different V_{reset} and V_t configurations. The refractory period is set to its maximum value in order to record many samples at V_{reset}. Using the minimum voltage during the refractory period after each spike introduces a systematic error due to readout noise: When averaging over the minimum after each spike, the result is lower than the reset potential (lower line). Determining the baseline of the voltage trace following a spike and averaging over all baselines following the spikes yields the desired result (upper line).

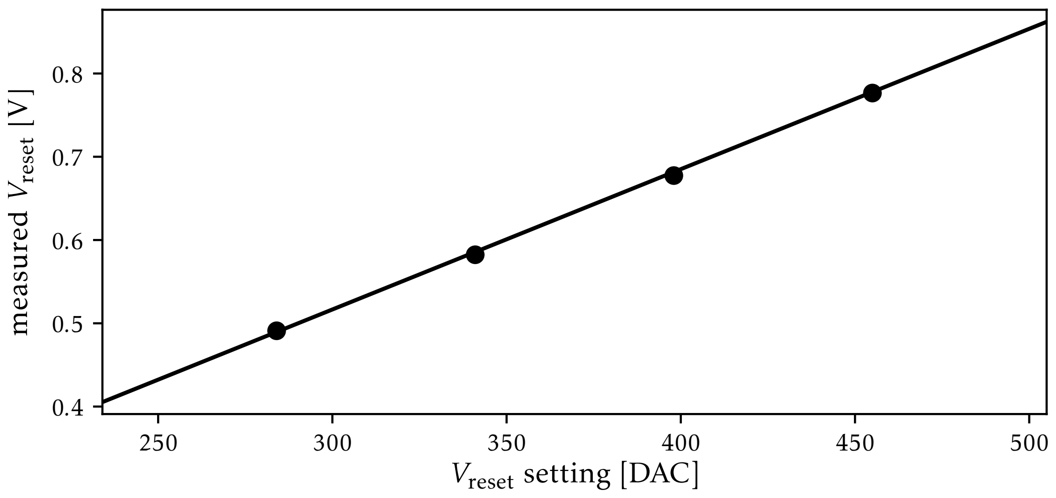


Figure 7.8: The measured V_{reset}, determined by finding the baseline, is averaged over the 128 denmems sharing V_{reset}. It shows a linear relation to its digital setting. The standard deviation between 128 denmems of 3.5 mV to 7 mV is not visible as it is smaller than the plotted dots.

7.8 MEMBRANE TIME CONSTANT

The membrane time constant τ_m is given by the chosen membrane capacitance C_m, which can have two different values, the bias current I_{gl} of the leakage OTA (section 4.1.1), as well as the current mirror setting for bias current amplification (section 4.2.1).

7.8.1 *ISI-based Method*

There is a straightforward method to determine τ_m which has been used in Schwartz (2013): The resting potential E_l is set above the spike threshold V_t, refractory period, adaptation and exponential term are disabled. Equation (2.10) simplifies to

$$\tau_m \frac{dV_m}{dt} = -(V_m - E_l). \tag{7.3}$$

As the membrane voltage rises from the reset potential towards the resting potential, it will reach the spike threshold and trigger a spike (figure 7.9). The resulting periodic spike frequency $f = \frac{1}{T_{ISI}}$ is measured. Without refractory period, the solution to (7.3) in this case is

$$\tau_m = T_{ISI} \cdot \ln\left(\frac{V_{reset} - E_l}{V_t - E_l}\right)^{-1} \tag{7.4}$$

with boundary conditions $V(0) = V_{reset}$ and $V(T_{ISI}) = V_t$.

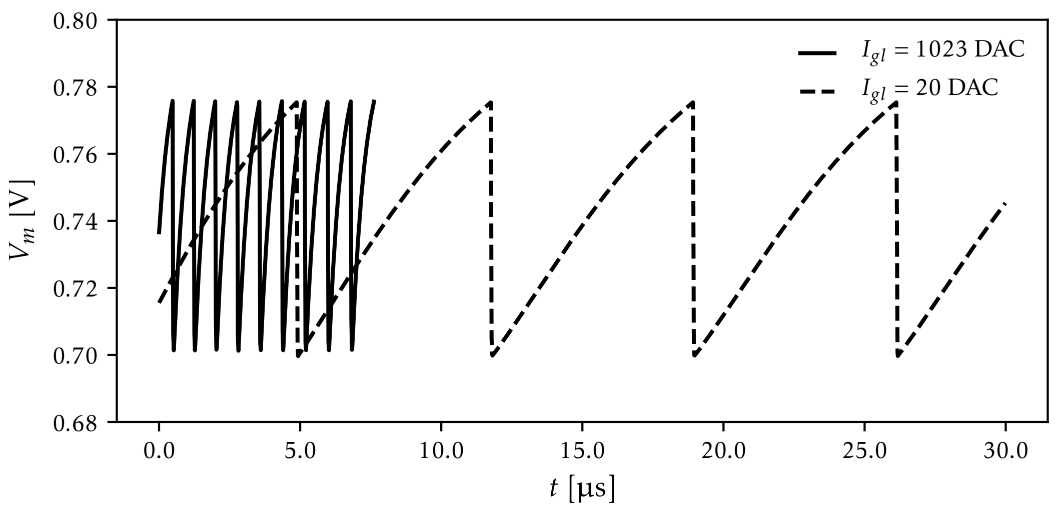


Figure 7.9: Transistor level simulations of two different membrane time constants. The resting potential E_l is set above the spike threshold V_t, refractory period, adaptation and exponential term are disabled. The membrane voltage raises from the reset potential towards the resting potential, until it triggers a spike at the spike threshold. Leakage conductance I_{gl} settings were chosen to show the slowest and fastest ideally possible membrane time constant, assuming that the smallest possible current cell output is $\approx 50\,nA$, i.e. 20 DAC. Although both simulations were executed with the same target duration, our testbench is designed to stop the simulation after a configurable number of spikes, which is 10 by default. This will save computation time, as the time resolution of the transistor level simulation is increased during a spike.

The resulting membrane time constants when sweeping over I_{gl} in transistor level simulations of an ideal denmem, for normal speedup setting and big membrane capacitance, are shown in figure 7.10. From the OTA characteristics we expect that $g_L \propto \sqrt{I_{gl}}$ (Millner, 2012), resulting in $I_{gl} \propto \frac{1}{{\tau_m}^2}$. Both the simulated and measured time constants deviate from this form. It was found that adding a term $I_{gl} \propto \frac{1}{\tau_m}$ compensates these deviations, resulting in a curve of the shape

$$I_{gl} = \frac{c_1}{\tau_m} + \frac{c_2}{{\tau_m}^2}. \tag{7.5}$$

Although this method works well in simulation, it depends on knowing the values of V_{reset}, V_t and E_l. In simulation they are directly accessible. For the hardware measurement these potentials are calibrated before measuring the time constant and therefore can be configured to the desired value. However, they depend linearly on the output of FG voltage cells, which are subject to non-negligible trial-to-trial variation of approximately $4\,mV$. V_{reset} and V_t can be measured from the voltage trace, but E_l is not directly visible with this method. Neglegting errors on

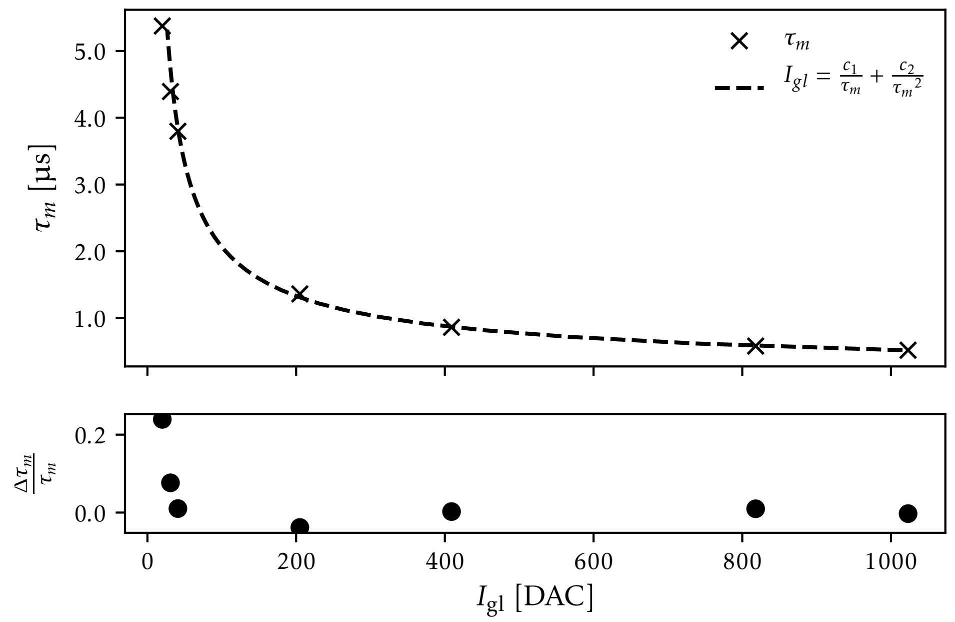


Figure 7.10: Fit of equation (7.5) to I_{gl} as function of τ_m, obtained from the ISI via equation (7.4) while sweeping I_{gl} in transistor level simulations of an ideal denmem. The inverse relation $I_{gl}(\tau_m)$ is the calibration curve with coefficients $c_1 \approx 100.12\,\mathrm{DAC}\cdot\mu\mathrm{s}$, $c_2 \approx 220.26\,\mathrm{DAC}\cdot\mu\mathrm{s}^2$.

V_{reset} and V_t, we estimate the error cause by trial-to-trial variation of E_l by Gaussian error propagation of equation (7.4):

$$\frac{\Delta\tau_m}{\tau_m} = \ln\left(\frac{V_{\mathrm{reset}} - E_l}{V_t - E_l}\right)^{-1} \frac{V_{\mathrm{reset}} - V_t}{(V_{\mathrm{reset}} - E_l)(E_l - V_t)} \cdot \Delta E_l. \tag{7.6}$$

With the values we used above, $V_{\mathrm{reset}} = 0.74\,\mathrm{V}$, $V_t = 0.77\,\mathrm{V}$ and $E_l = 0.8\,\mathrm{V}$, an error of 1% on E_l in equation (7.6) results in an error of 19% in τ_m. The error can be reduced by choosing a smaller E_l, for example $E_l = 0.7\,\mathrm{V}$, $V_t = 0.65\,\mathrm{V}$ and $V_{\mathrm{reset}} = 0.6\,\mathrm{V}$ already reduces the error in τ_m to 10% at an error of 1% on E_l.

7.9 STIMULATION-BASED METHOD

We have developed an alternative method, presented in Schmidt (2014), to circumvent this issue: we stimulate the denmem with a periodic current pulse to raise the membrane voltage without triggering a spike. The denmem is configured such that V_t is above E_l. After the current pulse stops, the membrane voltage decays towards the resting potential (figure 7.11). We determine the time constant by fitting the falling flank, where the current input is off. At the raising flank, the current input is present and the trace could be subject to non-constant current input. The OTA conductance for a configured bias current should be constant, as required by the

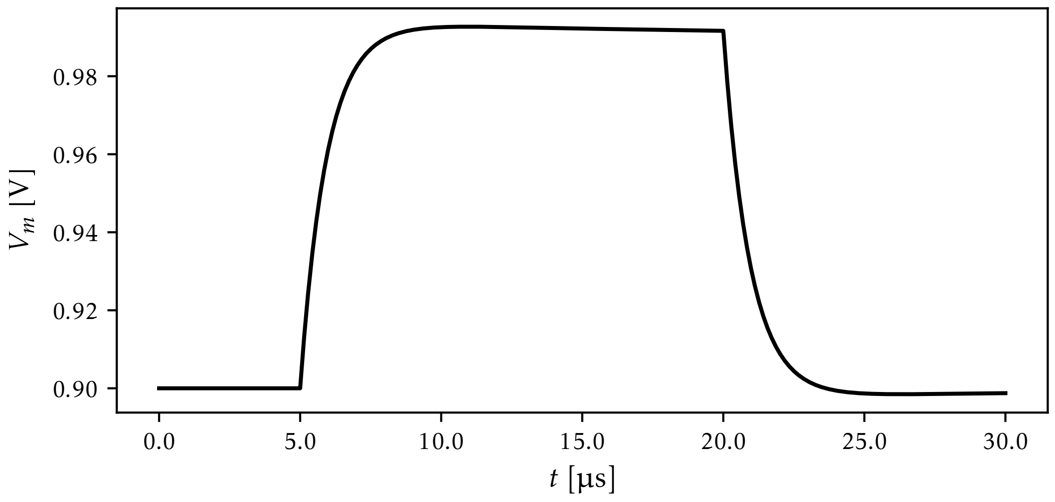


Figure 7.11: Simulated current stimulus on a single AdEx neuron, using NEST (Gewaltig and Diesmann, 2007). The rectangular current pulse, active from 5 µs to 20 µs, raises the membrane voltage without triggering a spike. After the current pulse ends, the membrane voltage decays towards the resting potential.

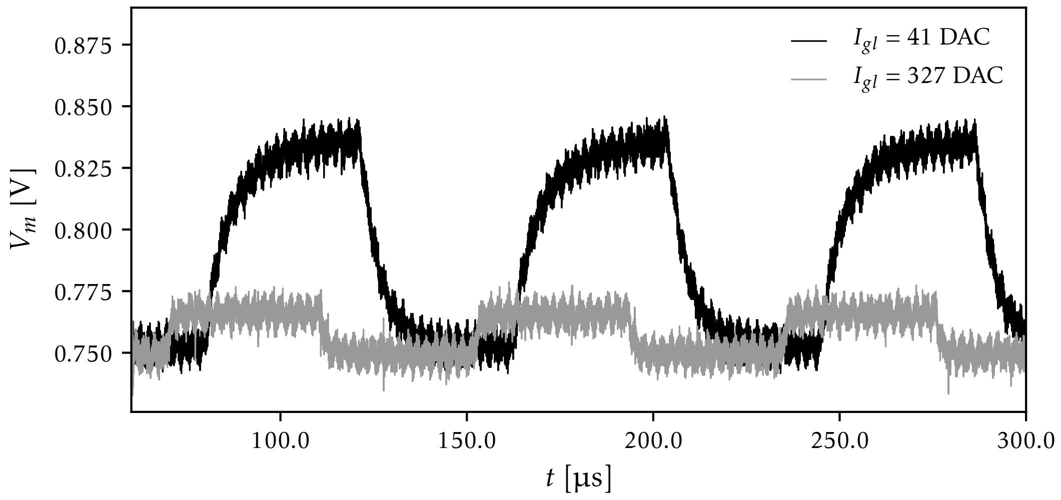


Figure 7.12: Exemplary traces recorded during I_{gl} calibration. A periodic current stimulus is raising the membrane voltage, revealing the effective membrane time constant τ_m. To reduce the noise in these periodic traces we average over one stimulus period, resulting in a considerably less noisy trace in figure 7.13.

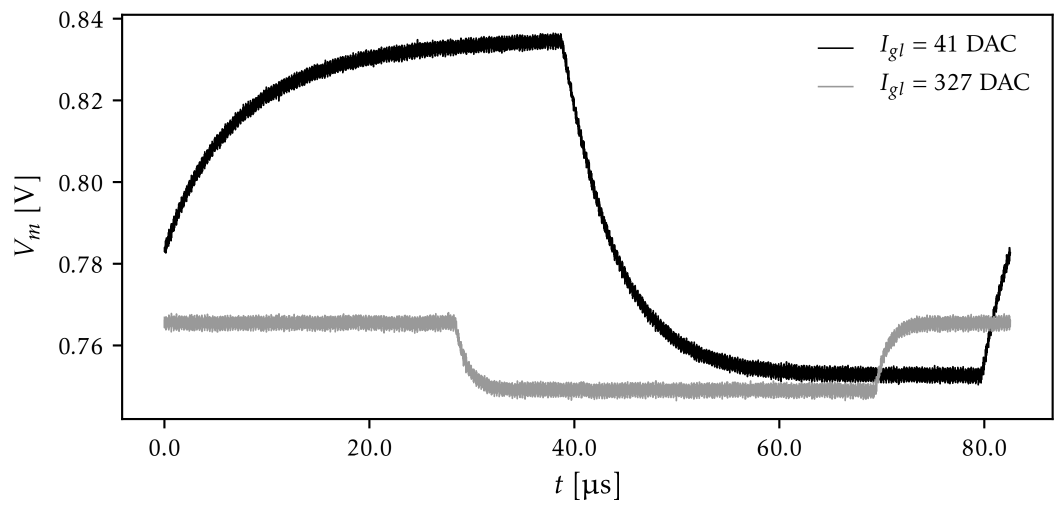


Figure 7.13: Traces recorded during I_{gl} calibration after averaging over the stimulation period. For the current stimulus, 129 stimulation pulse amplitude values are being iterated and held over the configured pulse length, from 1 to 16 clock cycles of the slow clock. This clock results from dividing the PLL frequency by four, $f_{slow} = \frac{f_{PLL}}{4}$. At maximum pulse length and $f_{PLL} = 100\,\text{MHz}$ the resulting stimulation period is $T_{stim} = 129 \cdot 16/f_{slow} = 82.56\,\mu\text{s}$.

AdEx model. We choose a configuration which keeps the voltage difference from resting potential to membrane potential with current stimulus below 150 mV. This way we avoid OTA saturation and keep it in an almost constant regime (compare figure 4.3).

The traces (figure 7.12) are still subject to noise. The selected constant current stimulus results in a raised membrane voltage of almost 150 mV for slow membrane time constants, but the dominating leakage term at fast membrane time constants results in relatively small changes of membrane voltage for larger I_{gl}. Averaging over one stimulus period reduces the noise in these periodic traces. This requires another prior measurement: The AnaRM sampling rate of nominal 96 MHz and configured HICANN clock of nominal 100 MHz are independent from each other, slight variations on each frequency require a correction factor when matching time of ADC samples with the current stimulus period. The corresponding correction factor is determined by comparing spikes at a known interval to the recorded ADC trace: we stimulate the synapse driver debug output with spikes from a background generator at 100 MHz and record approximately 2000 spikes from this output with the AnaRM. By comparing the measured ISI to the expected one, we calculate the correction factor between HICANN and AnaRM clocks. This method and its significance for averaging has been introduced in Koke (2016) for PSPs. The trace of one averaged current pulse stimulation period, obtained from averaging figure 7.12, is shown in figure 7.13.

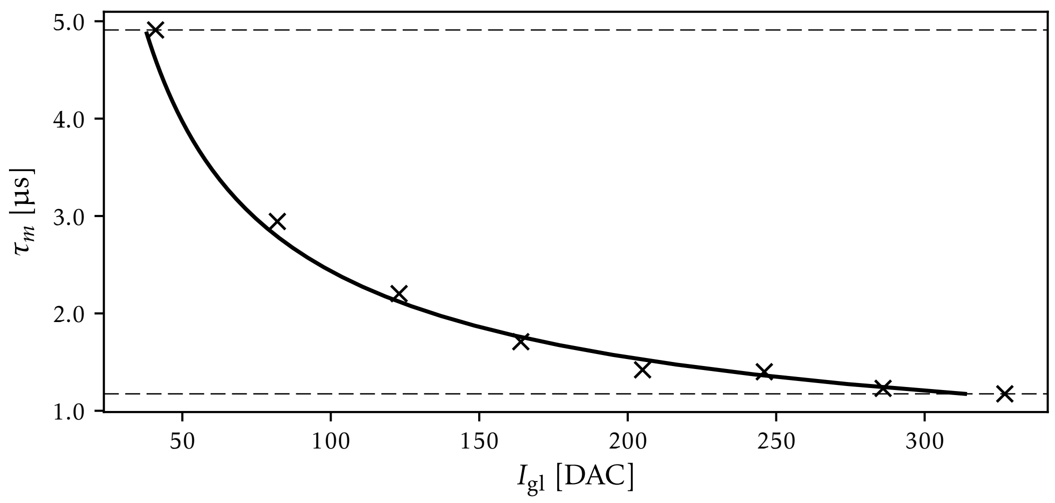


Figure 7.14: Exemplary I_{gl} calibration for a single denmem. The membrane time constants are obtained by fitting the decaying membrane potential after a rectangular current stimulus for different values of I_{gl}. The straight line shows a fit of equation (7.5) to I_{gl} as function of τ_m. The domain is defined as the minimum and maximum time constants that could be observed (dashed lines).

As before, the resulting membrane time constants for different values of I_{gl} are fitted to equation (7.5) to obtain the individual calibration curve for each denmem. An example is shown in figure 7.14.

7.10 REFRACTORY PERIOD

The relation between bias current I_{pl} and the resulting refractory period τ_{ref} is determined by configuring the denmem to spike periodically ($E_l > V_t$) without adaptation (table 7.1) and sweeping over I_{pl}. We first measure the resulting ISI for our default configuration (see appendix B for details) without refractory period, i. e. $I_{pl} = 1023\,\mathrm{DAC}$, as reference. We then sweep I_{pl} and subtract the reference ISI from the ISI with the current I_{pl} setting to calculate the refractory period. The recording time is chosen such that at least 10 spikes are recorded for the longest refractory period. This method works under the assumption that the refractory period is actually zero in the first measurement. Transistor level simulations showed that duration of the reset current at $I_{pl} = 1023\,\mathrm{DAC}$ is approximately $50\,\mathrm{ns}$.

The ISI is obtained by detecting spikes from the membrane voltage trace, for example in figure 7.15. This method also allows to use digital spike timestamps in the future. Digital timestamps are not supported (yet) in our testbench interface for transistor level simulations and did not work reliably on the FCP when they were initially evaluated before this method was implemented. However, the method can be switched to use spike timestamps easily.

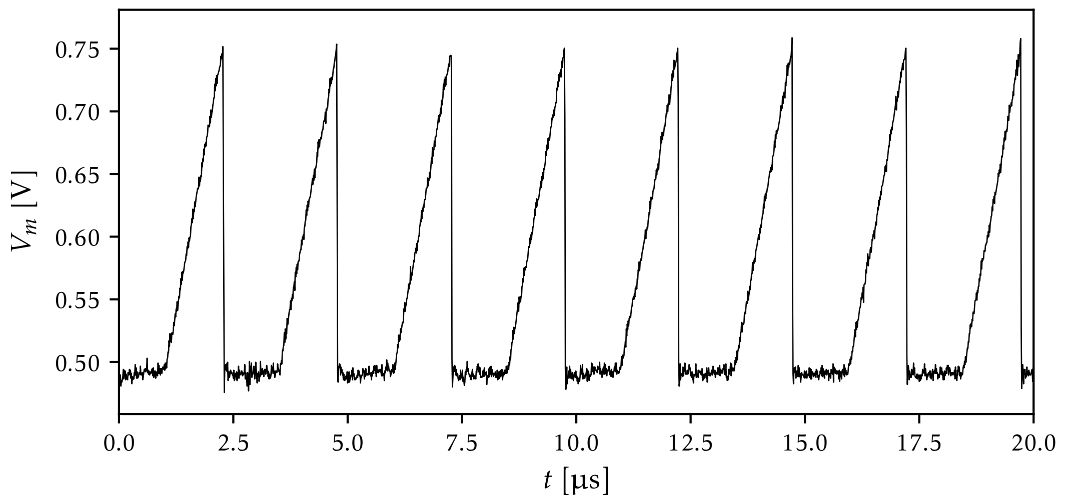


Figure 7.15: Exemplary membrane voltage trace during refractory time calibration. To calculate the refractory time, the ISI at zero refractory period is subtracted from the ISI of this trace. Currently this method is still based on the membrane voltage trace, but it allows to use spike timestamps instead in the future.

Once the resulting refractory period for multiple I_{pl} is known, we can fit the calibration function $I_{pl} = \frac{1}{c_0 + c_1 \cdot \tau_{ref}}$ to the points (figure 7.16). As this function is most sensitive for small I_{pl}, we collect more points in this area. Trial-to-trial variation has a large impact for long refractory periods, i. e. small I_{pl}. We configure a larger number of repetitions for this method compared to other methods with linear parameter dependency to minimize the non-linear effect during calibration. In our configuration optimized for speed, the number of repetitions is still 4, while other methods run at a single repetition.

7.11 ADAPTATION PARAMETERS

There are three adjustable parameters of the adapation term (compare section 4.1.3): the adaptation coupling a as conductance of the corresponding OTA via $I_{gladapt}$, the adaptation time constant τ_w resulting from the current onto C_w through the OTA g_w, which is controlled via I_{radapt}, and b as voltage increase V_b, controlled via I_{fire}. The value of the dynamic adaptation variable w is stored as voltage V_w on C_w. As this voltage is not directly accessible, we need some theoretical considerations to extract the adaptation current w from the accessible information, either the membrane voltage trace, or spike timestamps.

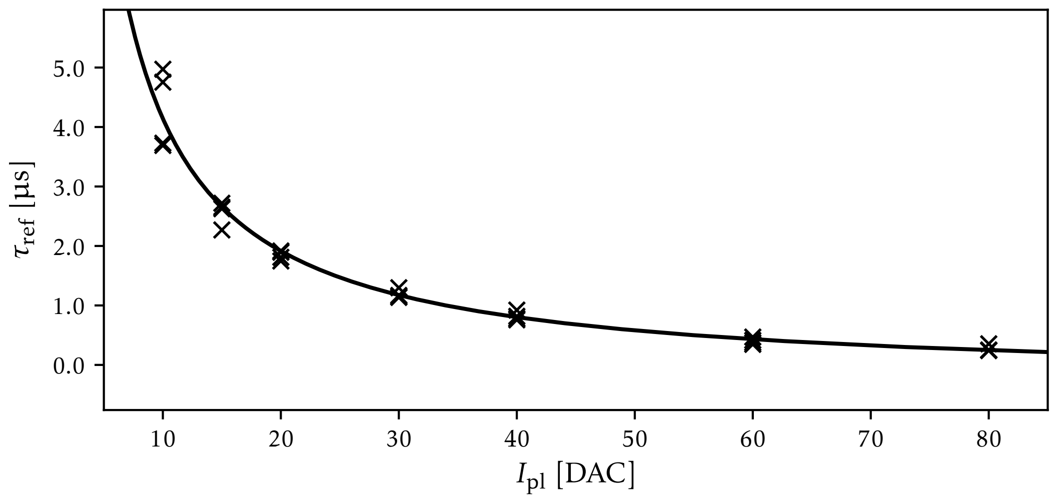


Figure 7.16: I_{pl} calibration curve. The transformation has the shape $I_{pl} = \frac{1}{c_0 + c_1 \cdot \tau_{ref}}$. Therefore the effect of trial-to-trial variation has most impact for small values of I_{pl}. Each step of I_{pl} was measured with 4 repetitions. Assuming a Gaussian distribution of the resulting current cell output due to trial-to-trial variation, the resulting points are distributed in a convolution with the nonlinear transformation of the circuit.

7.11.1 *ISI-Based Determination of the Coupling Parameter* a

Assuming that the exponential term can be disabled completely and in the absence of additional currents $I = 0$,

$$C_m \frac{dV_m}{dt} = -g_L(V_m - E_l) + g_L \Delta_T \exp\left(\frac{V_m - V_T}{\Delta_T}\right) - w + I, \tag{2.10}$$

becomes

$$C_m \frac{dV_m}{dt} = -g_L(V_m - E_l) - w. \tag{7.7}$$

We disable Spike-Triggered Adaptation (STA) by setting $b = 0$. If the adaptation current is decaying instantaneously, setting $\tau_w = 0$ in

$$\tau_w \frac{dw}{dt} = a\,(V_m - E_l) - w. \tag{2.11}$$

gives the equilibrium

$$w = a\,(V_m - E_l) \tag{7.8}$$

and equation (7.7) becomes

$$\begin{aligned}
C_m \frac{dV_m}{dt} &= -g_L(V_m - E_l) - a\,(V_m - E_l) && (7.9)\\
\Leftrightarrow \tau_m \frac{dV_m}{dt} &= -\left(1 + \frac{a}{g_L}\right) \cdot (V_m - E_l) && (7.10)\\
\Leftrightarrow \underbrace{\frac{\tau_m}{1 + \frac{a}{g_L}}}_{\tau_{\text{eff}}} \frac{dV_m}{dt} &= -(V_m - E_l) && (7.11)
\end{aligned}$$

with an effective time constant

$$\tau_{\text{eff}} = \frac{\tau_m}{1 + \frac{a}{g_L}}. \tag{7.12}$$

Similar to equation (7.3) we can calculate the expected ISI:

$$\begin{aligned}
T_{\text{ISI}} &= \tau_{\text{eff}} \cdot \ln\left(\frac{V_{\text{reset}} - E_l}{V_t - E_l}\right) && (7.13)\\
\Leftrightarrow \tau_{\text{eff}} &= T_{\text{ISI}} \cdot \ln\left(\frac{V_{\text{reset}} - E_l}{V_t - E_l}\right)^{-1} && (7.14)
\end{aligned}$$

and calculate a with equation (7.12):

$$\begin{aligned}
1 + \frac{a}{g_L} &= \frac{\tau_m}{T_{\text{ISI}}} \cdot \ln\left(\frac{V_t - E_l}{V_{\text{reset}} - E_l}\right) && (7.15)\\
\Leftrightarrow a &= g_L \cdot \frac{\tau_m}{T_{\text{ISI}}} \cdot \ln\left(\frac{V_t - E_l}{V_{\text{reset}} - E_l}\right) - g_L && (7.16)\\
&= \frac{C_m}{T_{\text{ISI}}} \cdot \ln\left(\frac{V_t - E_l}{V_{\text{reset}} - E_l}\right) - \frac{C_m}{\tau_m}. && (7.17)
\end{aligned}$$

Similar to the membrane time constant calibration, E_l is not directly visible in the trace. Its error due to trial-to-trial variation has a large impact on the precision of a (see section 7.8). In contrast to the error discussion section 7.8, here we additionally need to consider the error of τ_m for error propagation. If the trial-to-trial variation can be lowered significantly in the future, this method may become viable as it only depends on digital spikes as long as the other parameters of equation (7.17) are already known. Based on the experience of ISI-based membrane time constant calibration, we do not use this method but instead use the method presented in the following.

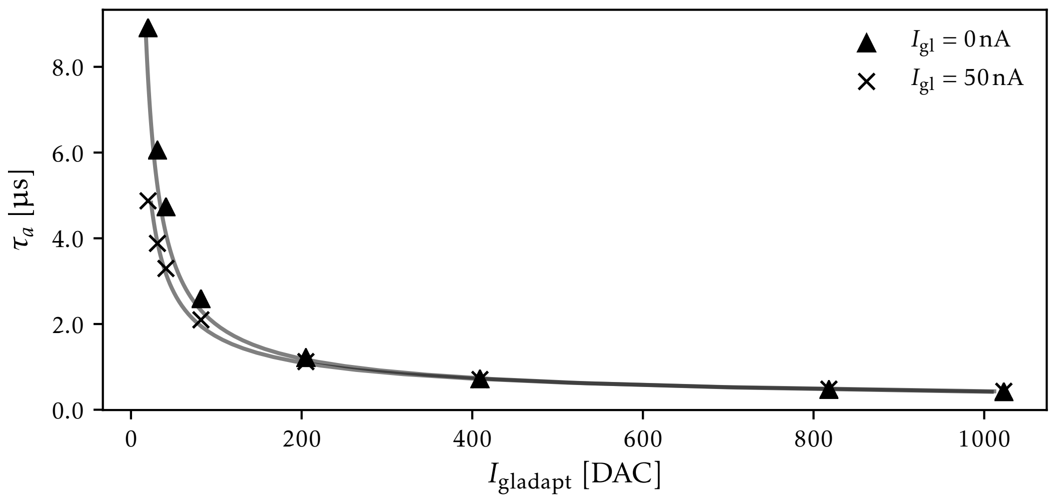


Figure 7.17: Calibration curve for $I_{gladapt}$, using the method of charging the membrance capacitance through a, in analogy to the leakage term (section 7.11.2), by setting $E_l > V_t$ in transistor level simulations of an ideal denmem. The time constant τ_a was determined for simulations with $I_{gl} = 0\,nA$ and $I_{gl} = 50\,nA$ to evaluate the effect of non-zero output of a current cell that is configured to 0 DAC. Lines show the fit results to $I_{gladapt} = \frac{c_1}{\tau_a} + \frac{c_2}{{\tau_a}^2}$. Deviations for non-zero I_{gl} have most influence on large time constants. This is caused by the current through g_L at $I_{gl} > 0$. For $I_{gl} = 0\,nA$, $c_1 = 139.43\,\mathrm{DAC} \cdot \mu s$ and $c_2 = 117.65\,\mathrm{DAC} \cdot \mu s^2$. For $I_{gl} = 50\,nA$, $c_1 = 82.33\,\mathrm{DAC} \cdot \mu s$ and $c_2 = 151.38\,\mathrm{DAC} \cdot \mu s^2$.

7.11.2 *Determination of Coupling a in Analogy to Leakage Term*

We can disable the leakage OTA for g_L to avoid the combination of both conductances, then equation (7.9) becomes

$$C_m \frac{dV_m}{dt} = -a\,(V_m - E_l) \tag{7.18}$$

$$\Leftrightarrow \tau_a \frac{dV_m}{dt} = -\,(V_m - E_l)\,, \tag{7.19}$$

which is equation (7.3) with a and $\tau_a = \frac{C_m}{a}$ replacing g_l and τ_m. In this configuration we can measure the membrane time constant τ_a to determine a with the methods described in section 7.8. Similar to the simulations for the membrane time constant, we run transistor simulations of an ideal denmem with two different settings for I_{gl}. The simulations are typically set up such that currents from floating gate cells are non-zero even when the digital configuration is 0 DAC. We expect the actual floating gate cells to deliver a non-zero current at the lowest setting and try to consider this effect in simulation. Figure 7.17 also shows simulation results for $I_{gl} = 0\,nA$ to highlight the difference. Resulting coefficients and domain boundaries are listed in table 7.2. Although the OTAs for g_L and a are identical in the ideal denmem, the coefficients slightly deviate from the ideal membrane time constant calibration coefficients. One cause might be that the adaptation circuit also has the OTA g_w, which is configured to maximum conductance at $I_{\text{radapt}} = 2500\,\text{nA}$. It could be circumvented by enabling the adaptation reset, which connects V_w to V_m (section 4.1.3). Unfortunately this reset can not be triggered individually, but is connected to the full denmem reset, which also pulls the membrane potential to V_{reset}. A future revision of HICANN could allow to trigger the adaptation reset independently.

	adaptation OTA		leakage OTA
	$I_{\text{gl}} = 0\,\text{nA}$	$I_{\text{gl}} = 50\,\text{nA}$	$I_{\text{gladapt}} = 50\,\text{nA}$
min. τ [µs]	0.42	0.43	0.52
max. τ [µs]	8.91	4.87	5.37
max. conductance [µS]	5.14	5.02	4.15
min. conductance [µS]	0.24	0.44	0.40
c_1 [DAC · µs]	139.43	82.33	100.12
c_2 [DAC · µs^2]	117.65	151.38	220.26

Table 7.2: Resulting domain and coefficients for I_{gladapt} calibration from transistor level simulations with the leakage OTA turned off ($I_{\text{gl}} = 0\,\text{nA}$) and not completely turned off (figure 7.17). The last column is the ideal calibration result of the leakage OTA (figure 7.10) for comparison.

7.11.3 *Determining Adaptation Parameters via Differential Equation Fitting*

The methods presented in sections 7.11.1 and 7.11.2 are based on the assumption that the adaptation time constant $\tau_w = 0$. However, we expect that at the maximum $I_{\text{radapt}} = 1023\,\text{DAC}$, the conductance g_w is finite and the resulting $\tau_w > 0$. To evaluate the effect of a non-zero τ_w, we compare NEST simulations of the AdEx model at $\tau_w = 0$ and a small $\tau_w = 5\,\mu\text{s}$ to measurements at maximum I_{radapt}, i. e. smallest possible τ_w in figure 7.18. We see that the overshoot at the edges of the current stimulus is present in hardware measurements both at fast and normal current mirror settings for I_{radapt}, which shows that $\tau_w > 0$. The methods in sections 7.11.1 and 7.11.2, which are based on the assumption $\tau_w = 0$, can only serve as a rough approximation.

Based on the evaluation in Friedrich (2015), which also investigated an analytical approach, we have developed an alternative method to determine a and τ_w: similar to the membrane time constant calibration, the denmem is stimulated by a periodic current stimulus and the decaying part is fitted to a model. In this case the model consists of equations (2.11) and (7.7) with adaptation enabled and $\tau_w > 0$. The adaptation current w or its hardware representation V_w is not directly measurable

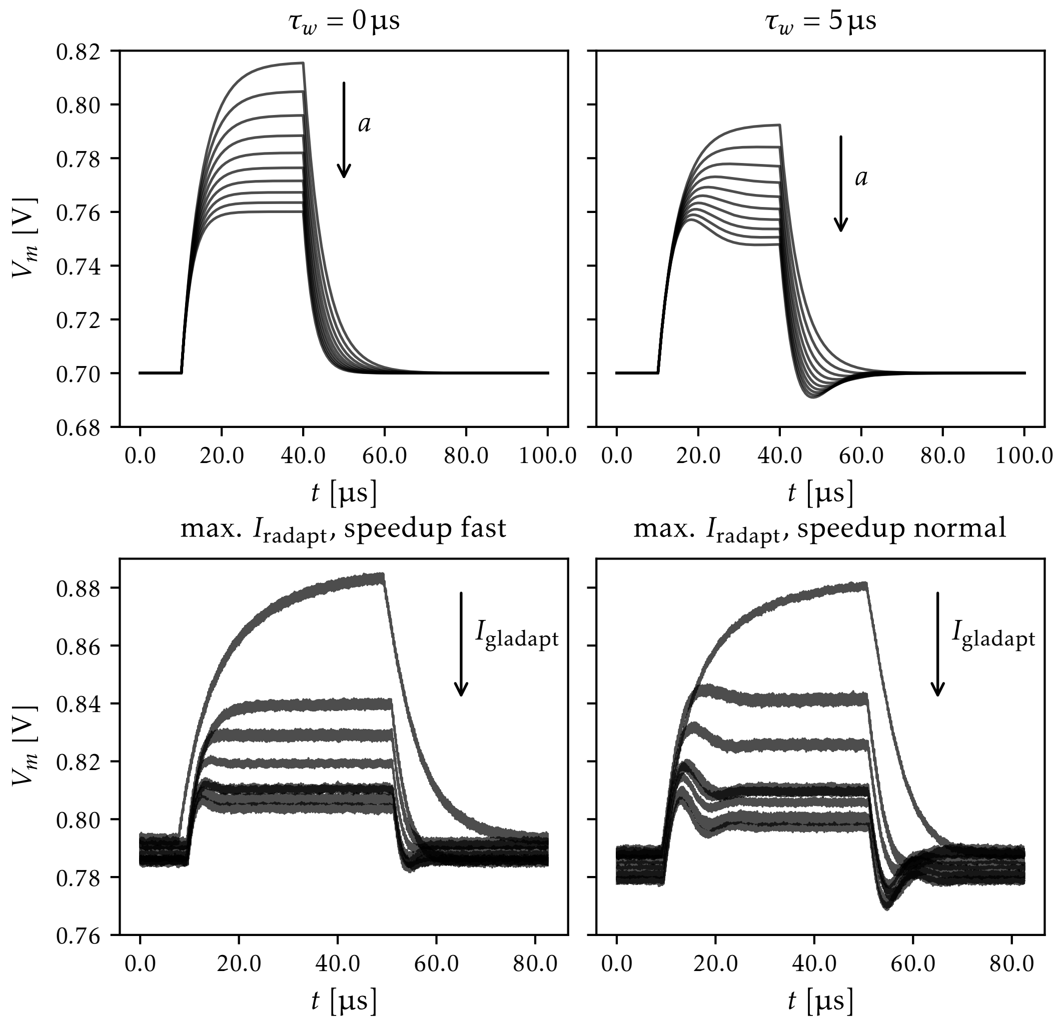


Figure 7.18: Evaluation of setting $\tau_w \to 0$ for determination of a. On the top left, we show NEST simulations with current stimulus active from $10\,\mu s$ to $40\,\mu s$. a is sweeped from $0\,\mu S$ to $4\,\mu S$. $E_l = 0.7\,V$, $V_t = 1\,V$, $\tau_m = 5\,\mu s$, $C_m = 2.16\,pF$, $\Delta_T = 0$, $\tau_w = 0$, $b = 0$. On the top right, $\tau_w = 5\,\mu s$. At the bottom, hardware measurements averaged over the stimulation period with similar configuration are shown: $E_l = 0.8\,V$, $V_t = 1\,V$, $I_{gl} = 0$, $I_{radapt} = 2500\,nA$. $I_{gladapt}$ is sweeped from $100\,nA$ to $800\,nA$. On the bottom left, the speedup setting for I_{radapt} is fast, choosing the smallest τ_w possible. On the right it is set to normal. We see that in both cases the effect of $\tau_w > 0$ is still present. The individual experiments were not triggered at the same time, but overlayed by detecting the raising edge, which did not align them perfectly. We can also observe the trial-to-trial variation of E_l.

in hardware experiments. As the membrane time constant is subject to trial-to-trial variation, we include it in the fit. The periodic trace is averaged over the stimulus period and smoothed using a linear Savitzky-Golay filter of window size 3. Start and end of the current stimulus are estimated using edge detection, time is shifted such that the rectangular stimulus does not cross the periodic boundary of the stimulation period T_{stim}, resulting in $V_m(0) = V_m(T_{\text{stim}}) = E_l$ (compare figure 7.18). The initial condition is $V_m(0) = E_l$ and $w_0 \equiv w(0) = 0$ as the membrane potential remains at E_l sufficiently long for w to decay. We fit the parameters a, τ_w and τ_m, as well as the amplitude of the rectangular current stimulus I_{stim}. All parameters are constrained to not exceed their expected maximum value by a factor of two. The initial guess for each parameter is set close to the expected value. In case of a and τ_w we use the ideal calibration obtained from transistor level simulation for an inital guess. The membrane time constant τ_m is known by previous calibration with the uncertainty of trial-to-trial variation. An example fit is pictured in the top left of figure 7.19. The remaining plots in figure 7.19 show the fit results when sweeping I_{gladapt}. The time constants τ_m and τ_w are expected to be constant within trial-to-trial variation. The deviations of τ_w for small values of I_{gladapt} can be explained by the fact that the adaptation coupling to the membrane potential is small in this case, making the influence of τ_w less distinct. The fitted membrane time constant τ_m shows a slight trend upwards, which is probably caused by the downward trend of I_{stim}. The current stimulus amplitude is also expected to be constant. However, we see that it decreases in the fit result with increasing I_{gladapt}. An explanation for this is that the stimulation response is decreasing with increasing I_{gladapt} (compare figure 7.18), which leads to incorrect fit results. It would certainly be preferrable to fix the stimulus amplitude to its actual value during the fit. The current stimulus is difficult to characterize individually, because the leakage current and adaptation current are not disabled entirely at $I_{\text{gl}} = I_{\text{gladapt}} = 0\,\text{DAC}$, as we have shown for the OTA responsible for τ_w. Another uncertainty is the parasitic capacitance of the current stimulus line, which we characterized in (Schmidt, 2014) and assume $C_m = 3.3\,\text{pF}$ as long as the current stimulus line is connected. The resulting conductance a follows the expected shape of the characteristic transformation $I_{\text{gladapt}} = c_1 \cdot a + c_2 \cdot a^2$. It may be possible to set τ_w to its average value above $I_{\text{gladapt}} = 500\,\text{DAC}$ and I_{stim} to a constant value, for example the average over all fits, in an iterative second characterization. Initial attempts of this approach showed no significant change in the resulting values of a.

This method is used to determine both τ_w and a from the membrane potential in hardware experiments.

7.11.4 *Determining the STA Parameter* b

Considering the results of the previous sections, we have developed a more accurate method to determine the relation between I_{fire} and b that is only available in transistor level simulation. If V_w can be connected to the analog output in a fu-

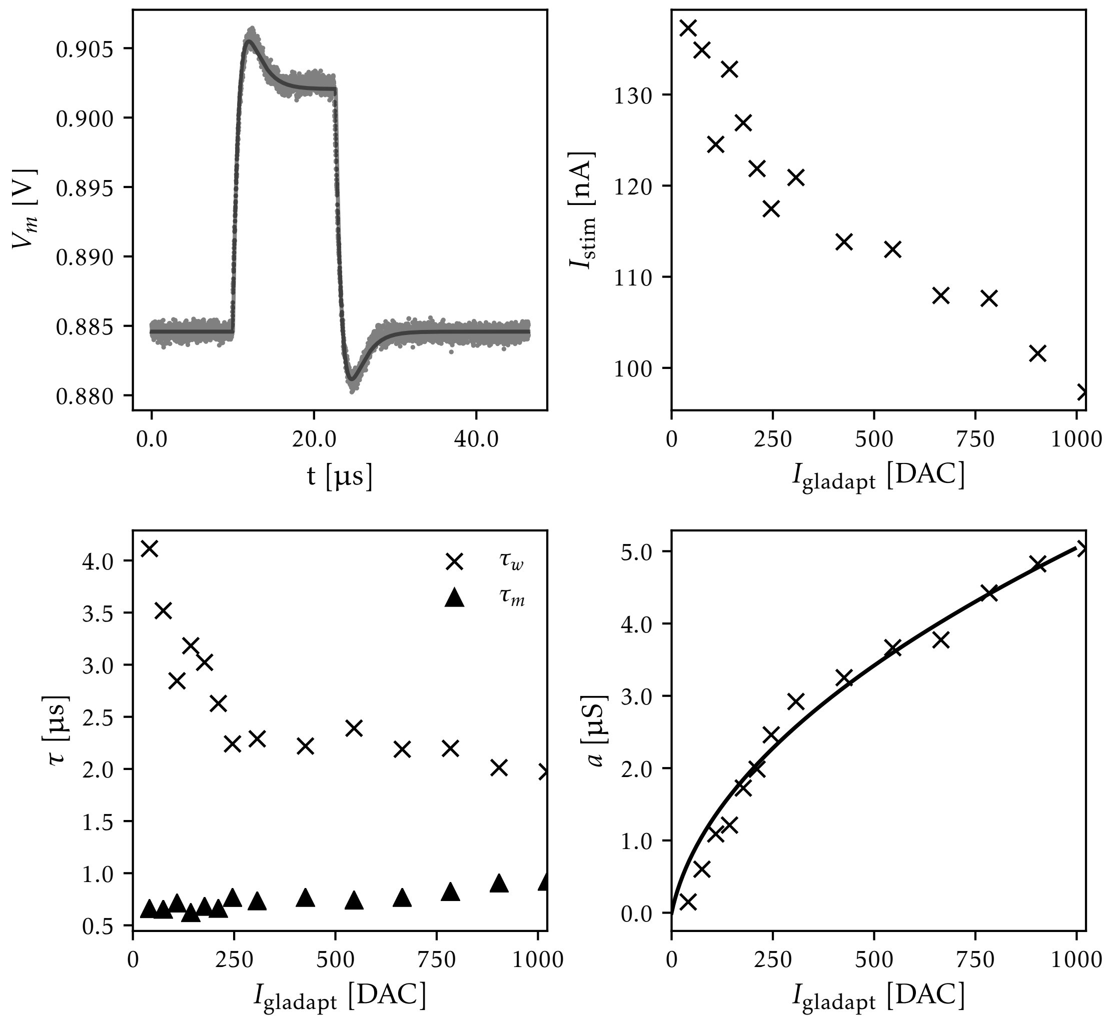


Figure 7.19: Exemplary visualization of the differential equation based fit method to determine a and τ_w from a membrane voltage trace. The denmem is stimulated by a periodic rectangular current stimulus. τ_w is set to a small value, i. e. $I_\text{radapt} = 2500\,\text{nA}$ in the normal current mirror setting. The top left shows an exemplary fit at $I_\text{gladapt} = 211\,\text{DAC}$. The periodic trace is averaged over the stimulus period and smoothed using a linear Savitzky-Golay filter of window size 3. Start and end of current stimulus are estimated using edge detection, time is shifted such that the stimulus is not crossing the periodic boundaries. Sweeping I_gladapt we fit each resulting trace. The remaining plots show all fitted variables as functions of I_gladapt: the stimulus amplitude is shown in the top right, the time constants τ_m and τ_w in the bottom left. These parameters are expected to be constant, the time constants will vary in the range of trial-to-trial variation. However, we see that there is an upwards trend of τ_m towards the configured value of 1 µs for increasing I_gladapt. The adaptation time constant τ_w can not be determined reliably for $I_\text{gladapt} < 250\,\text{DAC}$. The constant stimulus amplitude can not be reliably determined from the decreasing membrane response at increasing I_gladapt by the fit. The adaptation coupling a is actually changed by I_gladapt and follows the expected characteristic curve $I_\text{gladapt} = c_1 \cdot a + c_2 \cdot a^2$, plotted as solid line with fitted coefficients c_1 and c_2. Deviations from the curve correspond to deviations of τ_w from a constant value, $\tau_w \approx 2\,\text{µs}$. Since $I_g\mathtt{ladapt}(a = 0) = 0$, small values of I_gladapt could be omitted.

ture revision of the chip, the method can be applied to a measurement using this feature.

The leakage potential E_l is set above the spiking threshold V_t to achieve continuous spiking. a is set to a small value. This avoids fully charging C_w as it is discharged through a. Additionally it shows that the method could work if we can not set a to zero. The denmem reset connects V_w to V_m and pulls V_m to V_{reset}. Therefore V_w is initially at V_{reset}. The raising membrane voltage causes V_w to rise as well through g_w (equation (4.7)), during the refractory period V_w decays. For non-zero I_{fire}, V_w is increased by V_b at each spike through STA (figure 7.20). We get the spike times from the increase of a spike counter that we added to the testbench. The position of this timestamp is not exactly at the beginning of the steep increase of V_w, therefore we analyze an interval around a spike time $t_{spike} \pm 0.2\,\mu s$. The value of V_b is determined by calculating the increase of V_w. V_w is rising steeply over several samples, we drop differences between samples below $0.16\,mV$ to cut off the contribution of g_w and calculate the accumulated V_b. This adds a systematic error as the remaining samples also contain an unknown contribution through g_w. The resulting V_b is plotted for the sweeped configuration of I_{fire}, averaged over all spikes (figure 7.20). The standard deviation increases with I_{fire}, but remains below $0.1\,mV$. We fit a linear function $V_b = c_0 + c_1 \cdot I_{fire}$ with coefficients $c_0 = 1.99 \cdot 10^{-5}\,V$ and $c_1 = 9.34 \cdot 10^{-4}\,V/DAC$. From equations (4.8) and (4.10) we expect that $V_b \propto I_{fire}$. The offset c_0 is probably caused by the systematic error. b can be calculated via equation (4.8):

$$b = a \cdot V_b. \tag{4.8}$$

7.12 EXPONENTIAL PARAMETERS

In the AdEx model, the exponential term is controlled by two parameters: the threshold voltage V_T defines the onset of the exponential upswing, the slope factor Δ_T controls the duration between crossing of the exponential threshold V_T and reaching the spike threshold V_t of an action potential. The behavior of the exponential term circuit can be influenced by two denmem parameters: V_{exp} controls V_T and I_{rexp} affects both V_T and Δ_T (see section 4.1.4). In each of the methods in this chapter the adaptation term remains disabled (table 7.1).

7.12.1 *General Considerations*

Because the threshold $V_T(V_{exp}, I_{rexp})$ is affected by both analog parameters, V_T as function of V_{exp} needs to be characterized for fixed values of I_{rexp}. It is natural to calibrate Δ_T first.

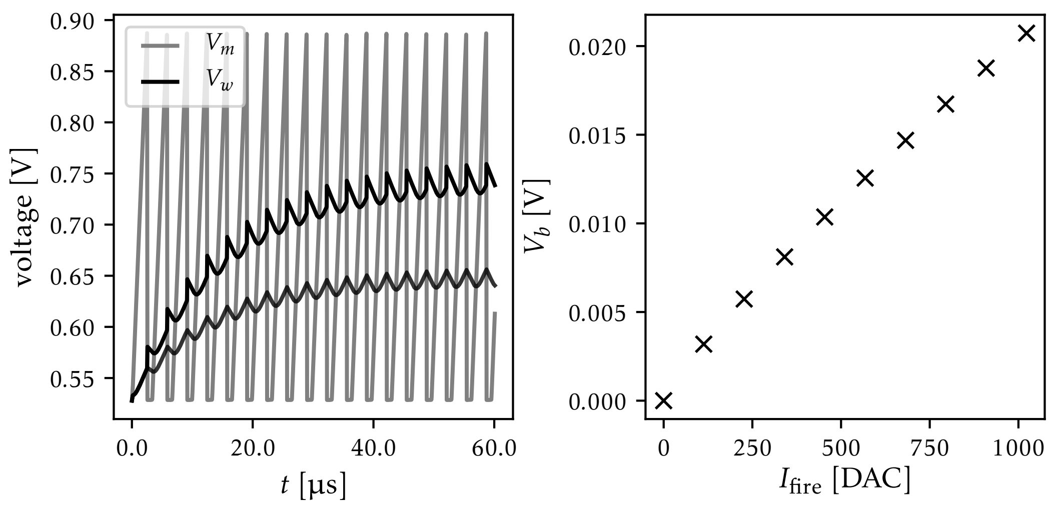


Figure 7.20: Transistor level simulations to analyze the effect of I_{fire}. Left: Membrane voltage V_m (grey) and adaptation voltage V_w for $I_{\text{fire}} = 1023\,\text{DAC}$. The lower line shows V_w at $I_{\text{fire}} = 0$ for comparison. a is set to a small value. E_l is set above V_t to achieve continuous spiking. After the reset, V_w is at V_{reset}. The raising membrane voltage causes V_w to raise as well, during the refractory period it decays. We observe STA for non-zero I_{fire}, V_w is increased by V_b. The value of V_b is determined by calculating the increase of V_w around a spike time t_{spike}. V_w is increased over several samples, we drop differences below $0.16\,\text{mV}$ to cut off the contribution of g_w and accumulate samples. This adds a systematic error as the remaining samples also contain an unknown contribution through g_w. The resulting V_b is plotted for the sweeped configuration of I_{fire} on the right, averaged over all spikes. The standard deviation increases with I_{fire}, but remains below $0.1\,\text{mV}$.

Let us have a look at the AdEx model behavior depending on V_T and Δ_T. Without external currents and disabled adaptation, equation (2.10) becomes

$$\begin{aligned} C_m \frac{dV_m}{dt} &= -g_L(V_m - E_l) + g_L \Delta_T \exp\left(\frac{V_m - V_T}{\Delta_T}\right) \\ \Leftrightarrow \underbrace{\frac{C_m}{g_L}}_{\tau_m} \frac{dV_m}{dt} &+ (V_m - E_l) = \Delta_T \exp\left(\frac{V_m - V_T}{\Delta_T}\right) \end{aligned} \tag{7.20}$$

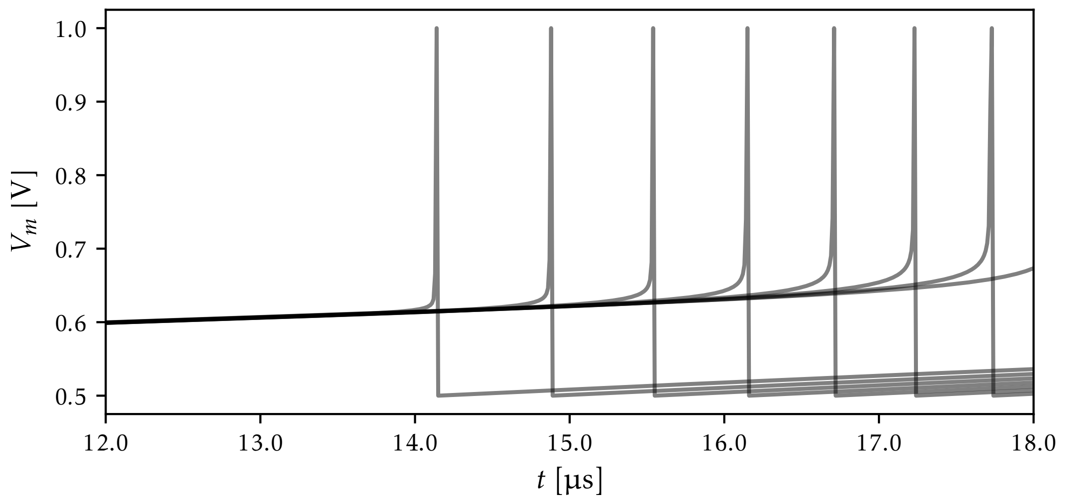


Figure 7.21: Numerical simulation of equation (7.20) with parameters $E_l = 0.8\,V$, $\tau_m = 30\,\mu s$, $V_{reset} = 0.5\,V$, $V_t = 1.0\,V$, $V_T = 0.6\,V$. Eight curves with equidistant Δ_T in the range from $3\,mV$ (most left) to $14\,mV$ (most right) show how the exponential rise is delayed by choosing larger Δ_T.

The exponential term is strongest for small Δ_T. V_T defines the membrane voltage close to which the exponential term becomes non-negligible: with a relevant current from the exponential term defined as $g_L \cdot V_{min}$,

$$\Delta_T \exp\left(\frac{V_m - V_T}{\Delta_T}\right) > V_{min} \tag{7.21}$$

$$\Rightarrow \exp\left(\frac{V_m - V_T}{\Delta_T}\right) > \frac{V_{min}}{\Delta_T} \tag{7.22}$$

$$\Rightarrow \frac{V_m - V_T}{\Delta_T} > \ln\left(\frac{V_{min}}{\Delta_T}\right) \tag{7.23}$$

$$\Rightarrow V_m > V_T + \Delta_T \cdot \ln\left(\frac{V_{min}}{\Delta_T}\right) \tag{7.24}$$

$$\Rightarrow V_m \gtrapprox V_T, \tag{7.25}$$

if V_{min} is in same the order of magnitude as Δ_T.

The slope factor Δ_T in the denominator delays the exponential current by requiring a larger voltage difference to achieve the same current at larger Δ_T. This effect is depicted in figure 7.21.

One observable that will change both with V_T or Δ_T is the resulting Inter-Spike Interval (ISI). We numerically simulate a range of V_T and Δ_T to plot how the ISI depends on these parameters. From NEST simulations, with model parameters as listed in table 7.3, we determine the ISI resulting from different configurations of V_T and Δ_T. This parameter dependency is plotted in figure 7.22. As we would expect from the model, the onset of the exponential rise has a strong influence on the

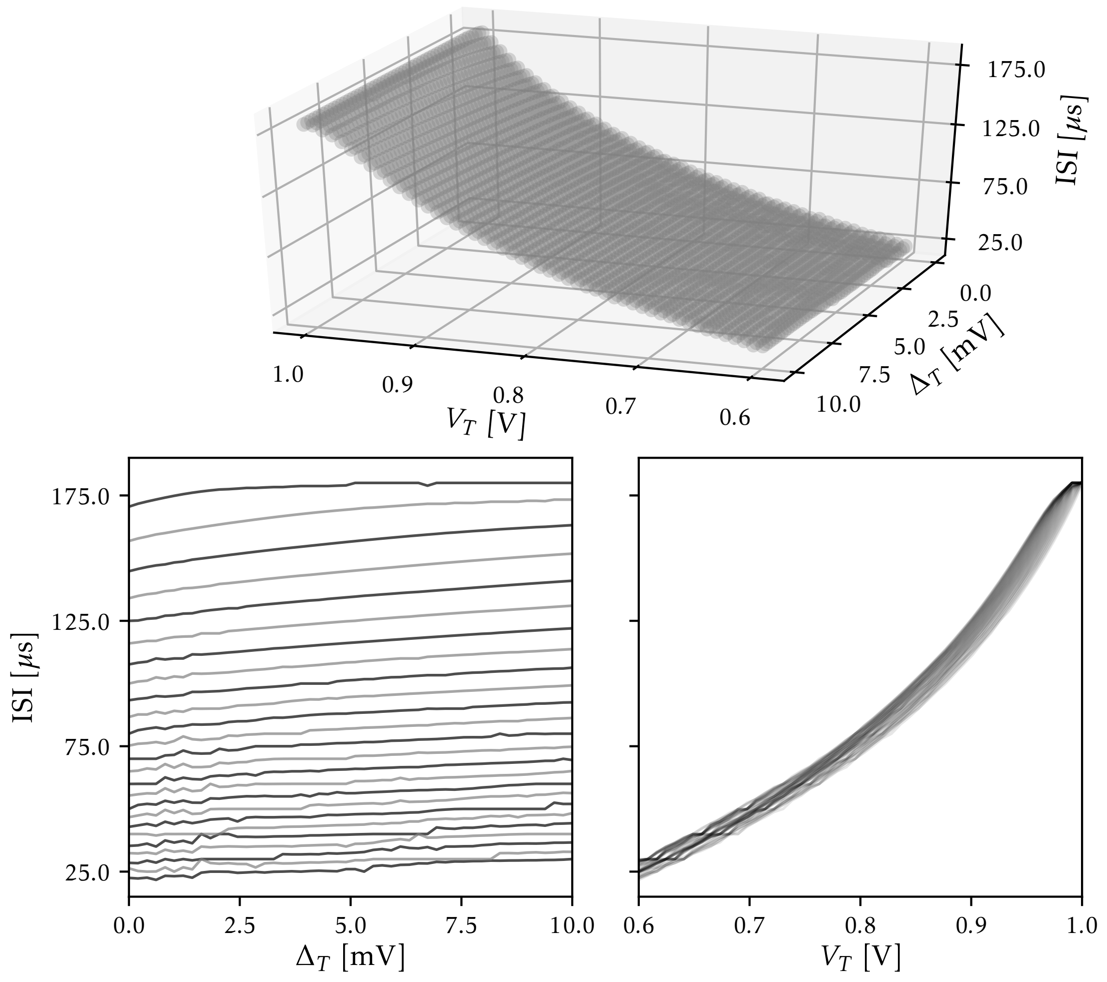


Figure 7.22: Numerical simulation to explore how the ISI depends on V_T and Δ_T with model parameters as listed in table 7.3. The top visualization shows both dependencies at the same time. The bottom left plot shows the depency on Δ_T in a separate line for each V_T. Different colors were chosen to be able to differentiate between lines, every second value is not shown. We observe that the ISI is slightly larger for larger Δ_T at a fixed V_T. On the bottom right we see the dependency on V_T with semi-transparent lines for each Δ_T. With increasing V_T the ISI is increasing as well. Saturation begins close to $V_T = 1.0\,V$ because the spike threshold is $V_t = 1\,V$.

AdEx parameter	PyNN variable	PyNN value	effective value
C_m	cm	2.16e-3	$2.16\,\mathrm{pF}$
E_l	v_rest	1100	$1.1\,\mathrm{V}$
V_t	v_spike	1000	$1.0\,\mathrm{V}$
V_{reset}	v_reset	500	$0.5\,\mathrm{V}$
a	a	0.0	$0\,\mathrm{nA}$
b	b	0.0	$0\,\mathrm{nA}$
τ_m	tau_m	0.1	$100\,\mathrm{\mu s}$
τ_{ref}	tau_refrac	0.0	$0\,\mathrm{ms}$
τ_w	tau_w	0.01	$10\,\mathrm{\mu s}$
Δ_T	delta_T	sweeped	
V_T	v_thresh	sweeped	

Table 7.3: PyNN parameter values used for the exponential term evaluation in figure 7.22.

resulting ISI. The delayed rise caused by larger Δ_T slightly increases the resulting ISI. There is no one-to-one mapping from an ISI to a unique combination of V_T and Δ_T. At a given ISI the difference between the smallest and largest matching V_T is up to $41\,\mathrm{mV}$ for the simulated parameters. This means that we can not calibrate V_T by analyzing the ISI without prior knowledge of Δ_T.

7.12.2 *Previous Work*

One method to deduce the value of V_T has been previously proposed (Schwartz, 2013). It is using the resulting ISI of a periodically spiking trace in comparison to numerical simulations of the model. In the measurements I_{rexp} is set to a low constant value to achieve a sharp spike. E_l is set above V_t to generate periodic spikes. The effective value of the exponential threshold V_T is explored for values ranging between V_{reset} and V_t. Sweeping over V_{exp}, we measure the resulting ISI. It is compared to the spike frequency obtained from numerical simulations of the differential equations.

Simulating the AdEx model over and over again for each denmem to map the ISI to a matching V_T is computationally expensive and can be replaced by a lookup table, similar to the visualization shown in the previous section, generated once for the set of model parameters by inital numerical simulations. However, as we have seen in the general considerations, the value of Δ_T must be known for this method to work. Setting I_{rexp} to a constant value will result in a still unknown value of Δ_T that varies between denmems due to fixed-pattern noise. Therefore we need to characterize Δ_T first by sweeping I_{rexp}. At the same time we need to avoid shifting V_T, which also depends on I_{rexp}, because V_T needs to remain below V_t to be able to observe the exponential term. In the hardware measurements we will also see that V_T needs to remain sufficiently above V_{reset} to avoid triggering another exponential upswing after a spike, even before the reset potential has been reached.

7.12.3 *Keeping* V_T *Constant*

Characterization of the relation between I_{rexp} and Δ_T requires to set V_T, which depends on both I_{rexp} and V_{exp}, to a value that does not change while sweeping I_{rexp}. We will calculate the value of V_{exp} required to come close to the desired V_T even without calibration, using the dependency that we found in section 4.1.4, equation (4.27):

$$V_{exp} = V_T - \Delta_T \ln\left(\frac{g_L \Delta_T}{\alpha}\right). \tag{4.27}$$

In order to get an estimation of the temperature-dependent α, we simulate just the exponential circuit in a separate testbench. We record I_{Mexp} and $R_1 = \frac{V_{out} - V_-}{I_1}$ and fit the constants α and β to equation (4.21)

$$I_{Mexp} = \alpha \exp\left(\frac{V_m - V_{exp}}{\frac{\beta}{R_1}}\right) \tag{4.21}$$

while sweeping over V_m. The result is shown in figure 7.23. In this figure we are able to observe the saturation behavior that has been described in (Millner, 2012) for membrane voltages further above the exponential threshold. Saturation effects also occur for $I_{rexp} > 800\,nA$. Temperature, choice of I_{rexp} and manual choice of the fitting range all influence the resulting value of α, depending on these choices we got results from $2 \cdot 10^{-2}\,nA$ to $12 \cdot 10^{-2}\,nA$. However, the result from figure 7.23, $\alpha \approx 5.59 \cdot 10^{-2}\,nA$, has been sufficient to set V_{exp} close to the desired V_T:

$$V_{exp} = V_T - \Delta_T \ln\left(\frac{g_L \Delta_T}{5.59 \cdot 10^{-2}\,nA}\right). \tag{7.26}$$

It may also be possible to look up the Berkeley Short-channel IGFET Model (BSIM) transistor model and its parameters for the subthreshold current I_{Mexp} that is used by the spectre simulator. As a quicker alternative method to determine α, we simulate the current-voltage characteristic of M_{exp} and fit it to equation (4.12) in the form

$$I_{Mexp} = \alpha \cdot \exp(V_{GS} \cdot \text{const}). \tag{7.27}$$

The resulting curves are shown in figure 7.24 with fitted coefficients α as listed in section 7.12.3.

Now that we know α, we can use equation (4.27) to calculate the required V_{exp} for a given V_T and Δ_T. As we do not know the relation between I_{rexp} and Δ_T yet, we

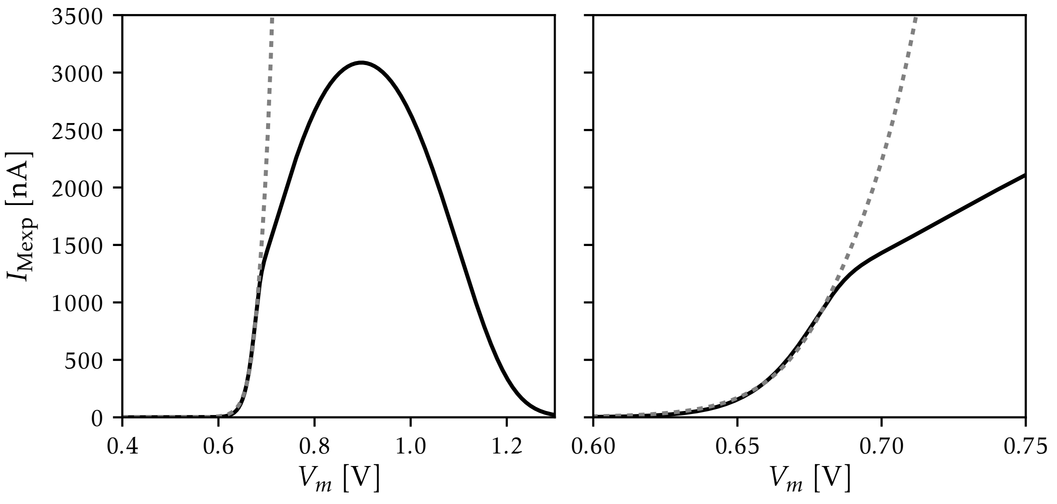


Figure 7.23: Exponential current I_{Mexp} (solid line) obtained from transistor level simulations of the exponential circuit with parameters $I_{\mathrm{vbexpb}} = 2500\,\mathrm{nA}$, $I_{\mathrm{bexpb}} = 2500\,\mathrm{nA}$, $V_{\mathrm{exp}} = 0.2\,\mathrm{V}$, $I_{\mathrm{rexp}} = 800\,\mathrm{nA}$, simulated at $T = 50\,°\mathrm{C}$. As reported in (Millner, 2012), saturation of the OpAmp causes a deviation from the exponential shape until the current onto the membrane is even decreasing for $V_m \gg V_T$. The right plot shows an enlargement of the range with exponential shape. The dashed line is a fit to equation (4.21), resulting in $\alpha \approx 5.59 \cdot 10^{-2}\,\mathrm{nA}$, $\beta \approx 5.66 \cdot 10^{3}\,\mathrm{V}\,\Omega$. These values strongly depend on the chosen fitting range, which was limited to $V_m < 0.685\,\mathrm{V}$ in this case to avoid the saturation regime.

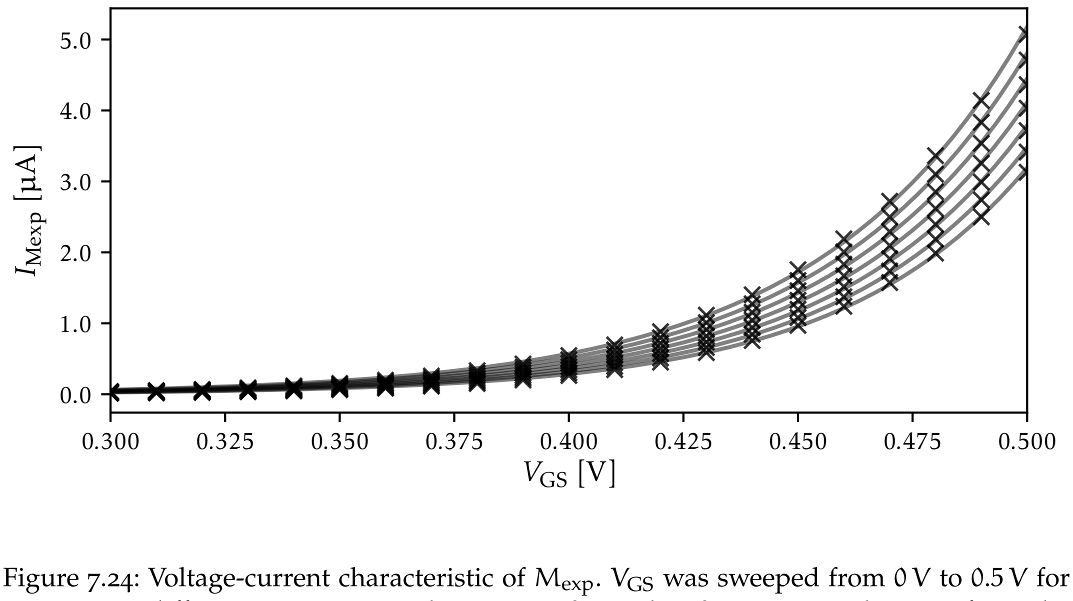


Figure 7.24: Voltage-current characteristic of M_{exp}. V_{GS} was sweeped from $0\,\mathrm{V}$ to $0.5\,\mathrm{V}$ for different temperatures between $35\,°\mathrm{C}$ and $65\,°\mathrm{C}$. Every 10th point from the simulation is plotted, the lines show the fit result to equation (7.27). Resulting coefficients α are listed in section 7.12.3.

T [°C]	α [A]
35	$1.80 \cdot 10^{-11}$
40	$2.38 \cdot 10^{-11}$
45	$3.11 \cdot 10^{-11}$
50	$4.03 \cdot 10^{-11}$
55	$5.20 \cdot 10^{-11}$
60	$6.65 \cdot 10^{-11}$
65	$8.46 \cdot 10^{-11}$

Table 7.4: Resulting coefficients α when fitting the voltage-current characteristic of M_{exp} to equation (7.27) (see figure 7.24) at different temperatures T. The voltage-current characteristic was obtained from transistor level simulations of a testbench for the transistor M_{exp}, gate and source were connected as in the exponential circuit.

first use a rough estimate for $\Delta_T(I_{rexp})$ based on previously published results (Millner, 2012). The method that was used in (Millner, 2012) for these results is not suitable for hardware experiments because it calculated Δ_T using currents inside the circuit that are only accessible in simulation. This estimated relation between I_{rexp} and Δ_T can be replaced by the ideal denmem characterization for $\Delta_T(I_{rexp})$, described in the next section. However, this characterization already requires to change V_{exp} together with I_{rexp} to keep V_T constant. The rough estimate for V_{exp} that was used to determine $\Delta_T(I_{rexp})$ for the first time is

$$V_{exp} \approx V_T - 1.64 \cdot 10^{-4} \, \frac{V}{nA} \cdot I_{rexp} \tag{7.28}$$

for $100\,nA \leqslant I_{rexp} \leqslant 800\,nA$. At $I_{rexp} = 500\,nA$, this estimate only results in a shift of approximately $10\,mV$. The simulated traces still showed V_T shifting upwards with increasing I_{rexp}. After finding $\Delta_T(I_{rexp})$, as described in the following, we use the result with equation (4.27) instead. Because the first characterization of $\Delta_T(I_{rexp})$ still uses this approximation, we perform a second characterization run afterwards to re-fine itself.

7.12.4 *Determining* Δ_T

We investigate the dependency of Δ_T on I_{rexp} in the linear range, $I_{rexp} \approx 100\,nA$ to $800\,nA$, as introduced in section 4.1.4. From numerical simulations of the AdEx model, using the range of Δ_T given in (Millner, 2012), we expect that increasing Δ_T will cause a visible delay of the exponential rise, as depicted in figure 7.21. We require a method to determine Δ_T from the membrane voltage trace instead of calculating it from voltages and currents that are only accessible in transistor level simulation.

Analyzing a membrane voltage trace, we use numerical differentiation with equation (7.20) for an approximation of Δ_T (Schwartz, 2013):

$$\tau_m \frac{\Delta V_m}{\Delta t} + (V_m - E_l) = \Delta_T \exp\left(\frac{V_m - V_T}{\Delta_T}\right) \tag{7.29}$$

$$\Rightarrow \underbrace{\ln\left(\tau_m \frac{\Delta V_m}{\Delta t} + (V_m - E_l)\right)}_{=:\ \xi} = \ln(\Delta_T) + \frac{V_m - V_T}{\Delta_T}. \tag{7.30}$$

Because of the term $\ln(\Delta_T)$, equation (7.30) is limited to $\Delta_T > 0$. Δ_T and V_T are both constant, Δ_T is the slope of a linear function $\xi(V_m)$, as shown in figure 7.25. The approximation is restricted to small ΔV_m and requires a non-negligible influence of the exponential term, i. e. V_m close to or above V_T. From equation (7.30) we can calculate both Δ_T and V_T: we plot $\xi(V_m)$ and fit the linear function

$$\xi = y_0 + y_1 \cdot V_m. \tag{7.31}$$

Comparing to equation (7.30),

$$y_0 + y_1 \cdot V_m = \ln(\Delta_T) + \frac{V_m - V_T}{\Delta_T} \tag{7.32}$$

$$\Rightarrow y_1 = \frac{1}{\Delta_T} \text{ and } y_0 = \ln(\Delta_T) - \frac{V_T}{\Delta_T}. \tag{7.33}$$

We can calculate $\Delta_T = \frac{1}{y_1}$, and also calculate V_T:

$$V_T = {y_1}^{-1} \cdot \left(\ln\left({y_1}^{-1}\right) - y_0\right). \tag{7.34}$$

We first evaluate this method with a membrane voltage trace generated by numerical simulation of the AdEx model to verify it. Once confirmed, we can continue evaluating the method with transistor level simulations of the denmem and ultimately with measurements on the actual chip. Usually we would choose the NEST simulator (Gewaltig and Diesmann, 2007) for AdEx model simulations in order to increase reproducibility. However, the NEST simulation data for this method contained simulation artifacts, probably caused by the unusually small timestep compared to the biological time domain. For this reason we chose to implement our own simulation code to generate clean data, which runs fast as it is limited to this simple task. The result of this analysis is shown in figure 7.25. The slope of the fitted curve is $\frac{1}{\Delta_T}$, the result is close to the actual value over the simulated range from 2 mV to 14 mV.

Next, we apply this method to transistor level simulations of an ideal denmem without variation. While sweeping I_{rexp}, we set V_{exp} according to equation (4.27). Here the refractory period is set to a non-zero value, because we observed that the exponential rise had been triggered during the reset in simulations at zero refractory period. The range of I_{rexp} was limited from 100 nA to 800 nA, below and above this range we get undesired behavior of the circuit. The resulting curves in

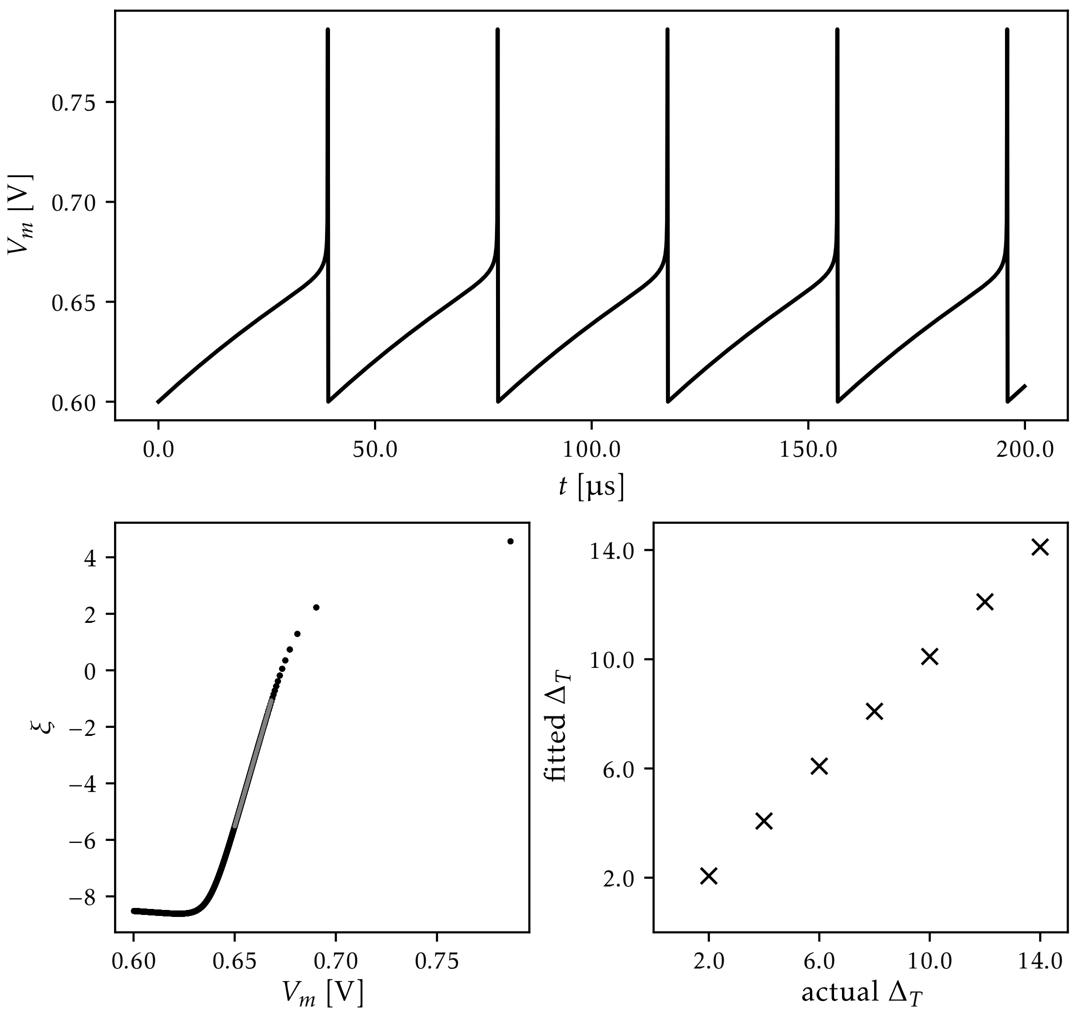


Figure 7.25: Evaluation of the method to determine Δ_T from $\xi(V_m)$ with numerical simulation data. Top: Numerically simulated membrane voltage V_m for equation (7.20) with parameters $\tau_m = 100\,\mu s$, $E_l = 0.8\,V$, $V_{reset} = 0.6\,V$, $V_t = 1.0\,V$, $\Delta_T = 4\,mV$, $V_T = 0.65\,V$ and a timestep of $0.1\,\mu s$. Bottom left: Analysis of the trace using equation (7.30). The slope is fitted for voltages above V_T for $\xi < -1$. Its inverse gives a good approximation to the configured $\Delta_T \approx 4.08\,mV$. Above $\xi = -1$, ΔV_m becomes too large and the approximation to $\frac{dV_m}{dt}$ does not hold anymore. At the bottom right we apply the method to several values of Δ_T in the expected range.

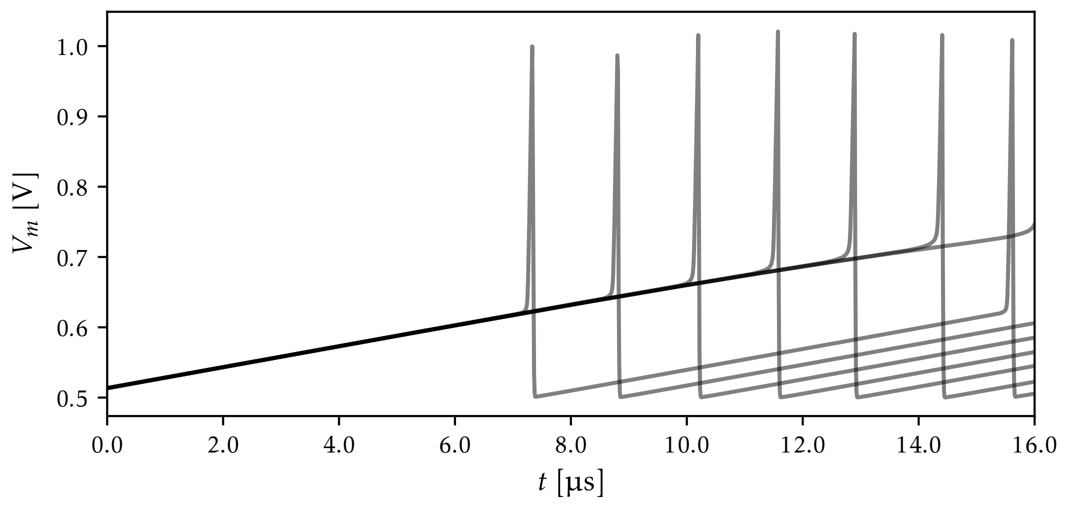


Figure 7.26: Membrane voltage traces from transistor level simulations for different values of Δ_T. I_{rexp} is set to equidistant target values from $100\,nA$ to $800\,nA$, just a subset has been plotted for clearness. The method uses calibrated values for $E_l = 0.8\,V$, $V_t = 1.0\,V$, $V_{\text{reset}} = 0.5\,V$, $\tau_m(I_{gl}) = 30\,\mu s$. The refractory period was configured to be non-zero without calibration, $I_{pl} = 2000\,nA$, to avoid activating the exponential raise during the reset. V_{exp} was calculated using equation (4.27) with target $V_T = 0.6$. This result looks qualitatively similar to the numerical AdEx model simulation in figure 7.21. One reason for the quantitative differences is the effective value of V_T.

figure 7.26 are qualitatively similar to the numerical AdEx model simulations in figure 7.21.

Using equation (7.34), we see that V_T is shifting from $0.6\,mV$ to $0.7\,mV$ for the simulations shown in figures 7.26 and 7.27. In these simulations, V_{exp} was calculated using equation (7.26), in the first iteration with equation (7.28) and in several following iterations with the result itself, equation (7.35), to improve the coefficients because V_T is changing less.

The resulting ideal denmem calibration function is

$$\Delta_T = 6.23 \cdot 10^{-3}\,\frac{mV}{nA} \cdot I_{\text{rexp}} + 2.58\,mV. \tag{7.35}$$

7.12.5 *Hardware Measurement of* Δ_T *and* V_T

For the hardware measurements sweeping I_{rexp}, we are calculating the required $V_{\text{exp}}(I_{\text{rexp}}, V_T)$ via equation (7.26) with equation (7.35) to keep V_T constant. The voltage difference ΔV_m between two membrane voltage samples above V_T, where the rise is extremely fast, is too large to apply the method from the previous section. While transistor level simulations are reducing the time step due to increasing current near V_T, the AnaRM sampling rate is fixed and the resultion is not sufficient

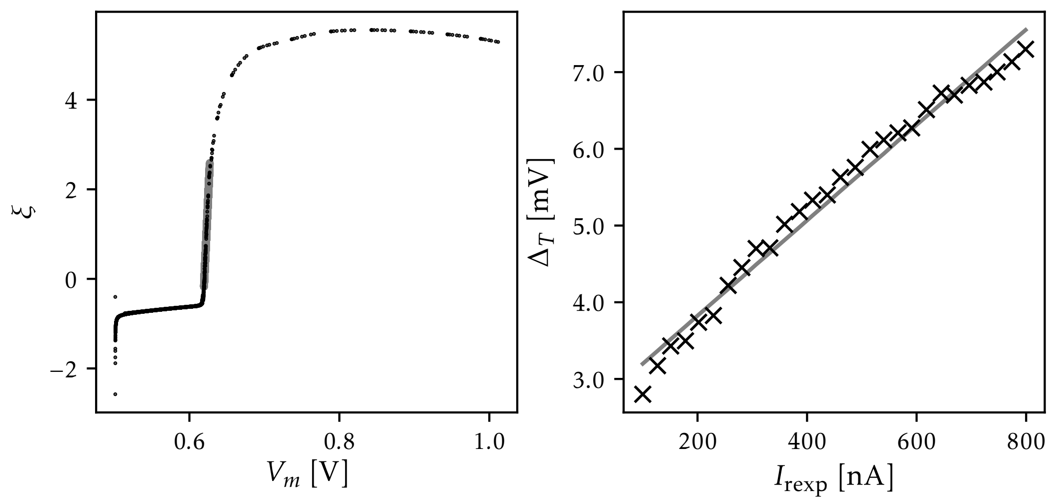


Figure 7.27: Application of the method used to determine Δ_T via $\xi(V_m)$ to transistor level simulation data. Left: Exemplary visualization of $\xi(V_m)$ at $I_{\text{rexp}} = 100\,\text{nA}$. The linear relation between V_m and $\frac{1}{\Delta_T}$ above $V_T \approx 0.6$, as given by equation (7.30), is present in a short range until saturation sets in. We fit the range of $-0.1 < \xi < 2.5$ to determine Δ_T. Depending on I_{rexp}, this range is between 7 mV and 21 mV wide and typically wider for larger I_{rexp}. On the right the resulting Δ_T is plotted in the usable range of $I_{\text{rexp}} = 100\,\text{nA}$ to $800\,\text{nA}$. In this range the ideal neuron shows a linear dependency $\Delta_T = 6.23\,\mu\text{V/nA} \cdot I_{\text{rexp}} + 2.58\,\text{mV}$.

for the requirement of small ΔV_m. Combined with the readout noise, the region we want to fit is not recognizable in figure 7.28. The low number of samples in the interval above V_T, but before saturation sets in, does not allow smoothing. Two example traces are shown in figure 7.29. In the second configuration, we observe that the reset current is not strong enough to pull the membrane potential towards V_{reset}. This effect was also visible in transistor level simulation at very short refractory periods. It can be avoided by choosing a longer refractory period or moving V_T further above V_{reset}. This effect is strongest for small I_{rexp}, resulting in small Δ_T, and larger I_{pl}, resulting in a short refractory period τ_{ref}. Small I_{rexp} will allow less current from the OpAmp to flow towards the V_{exp} buffer.

In order to get more samples of a spike, resulting in a smaller ΔV_m, we perform spike triggered averaging by shifting all samples before a spike by the corresponding digital spike timestamp t_{spike}. The resulting single ISI with more samples in figure 7.30 shows the typical readout noise, but also some jitter in the time axis. One cause is that the spike timestamp is binned to the FCP clock. In transistor level simulation we have seen that the exponential regime is approximately 20 mV wide. As saturation of the exponential term sets in shortly after the membrane potential crossed V_T, the steep spike can be described by a linear function. We only consider a short window before the exponential term becomes non-negligible, there

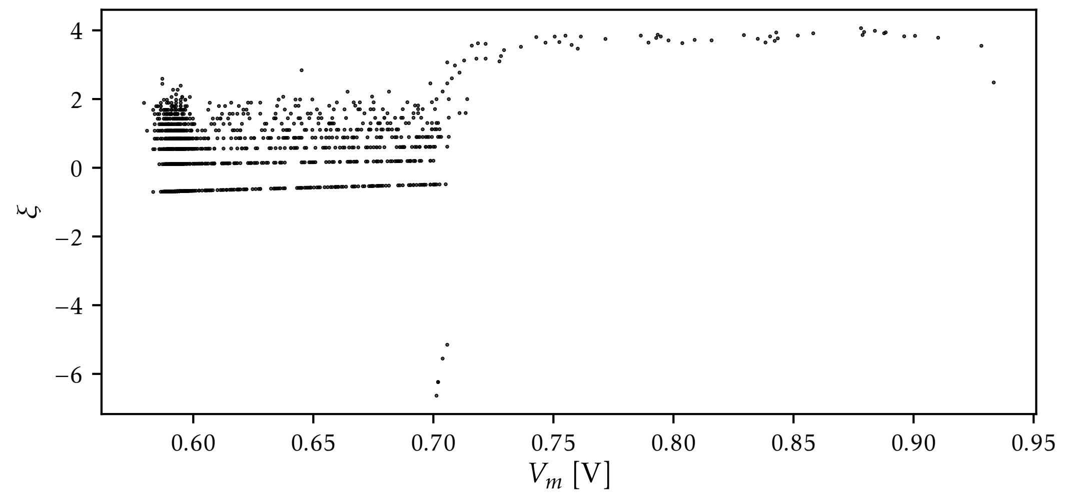


Figure 7.28: Application of the method to determine Δ_T via $\xi(V_m)$ to an AnaRM trace. In contrast to transistor level simulations (figure 7.27), the effects due to noise are dominating below $V_T = 0.7\,V$. The saturation regime shortly after V_T looks similar to the simulation. The resolution in the range of interest around V_T is not high enough due to the sampling rate of the AnaRM. The resulting ΔV_m is too large to apply the approximation used for ξ: before a spike $|\Delta V_m| < 10\,mV$ in a combination of noise and leakage current, during a spike $10\,mV < \Delta V_m < 60\,mV$. Additionally the range around $V_m \approx V_T$ before saturation of approximately $20\,mV$ does not yield enough data points. In the transistor level simulations this was not an issue because the time resolution is increased when currents grow larger. In the presented measurement, $I_{\text{rexp}} = 100\,nA$, $E_l = 0.8\,V$, $V_T = 0.7\,V$, $\tau_m = 10\,\mu s$, $V_t = 1\,V$, $V_{\text{reset}} = 0.6\,V$ and $I_{\text{pl}} = 20\,nA$.

the raising membrane potential due to the leakage term can also be approximated by a linear function. Then we can fit the piece-wise linear function

$$V_m(t) = \begin{cases} v_0 + k_1 \cdot t & t \leqslant t_0 \\ v_0 + k_1 \cdot t_0 + k_2 \cdot (t - t_0) & t_0 \leqslant t \leqslant t_1 \\ v_0 + k_1 \cdot t_0 + k_2 \cdot (t_1 - t_0) + k_3 \cdot (t - t_1) & t_1 \leqslant t \end{cases} \tag{7.36}$$

with $k_1, k_2 > 0$ and $k_3 < 0$ to this section of the ISI close to the spike (figure 7.30. This way we can also estimate the exponential threshold, which should be close to t_0. The resulting ISI and coefficients k_2 for a sweep of I_{rexp} are plotted in figure 7.31. The membrane potential at which the exponential term becomes strong enough to be visible can be estimated as $V_m(t_0)$. We will interpret it as V_T. The slope of the quickly raising membrane potential, k_2 in equation (7.36), can be used to tune the desired delay of a spike. If the behavior close to V_T is most relevant, we recommend to use the ideal calibration from transistor level simulations to choose I_{rexp} for a desired Δ_T.

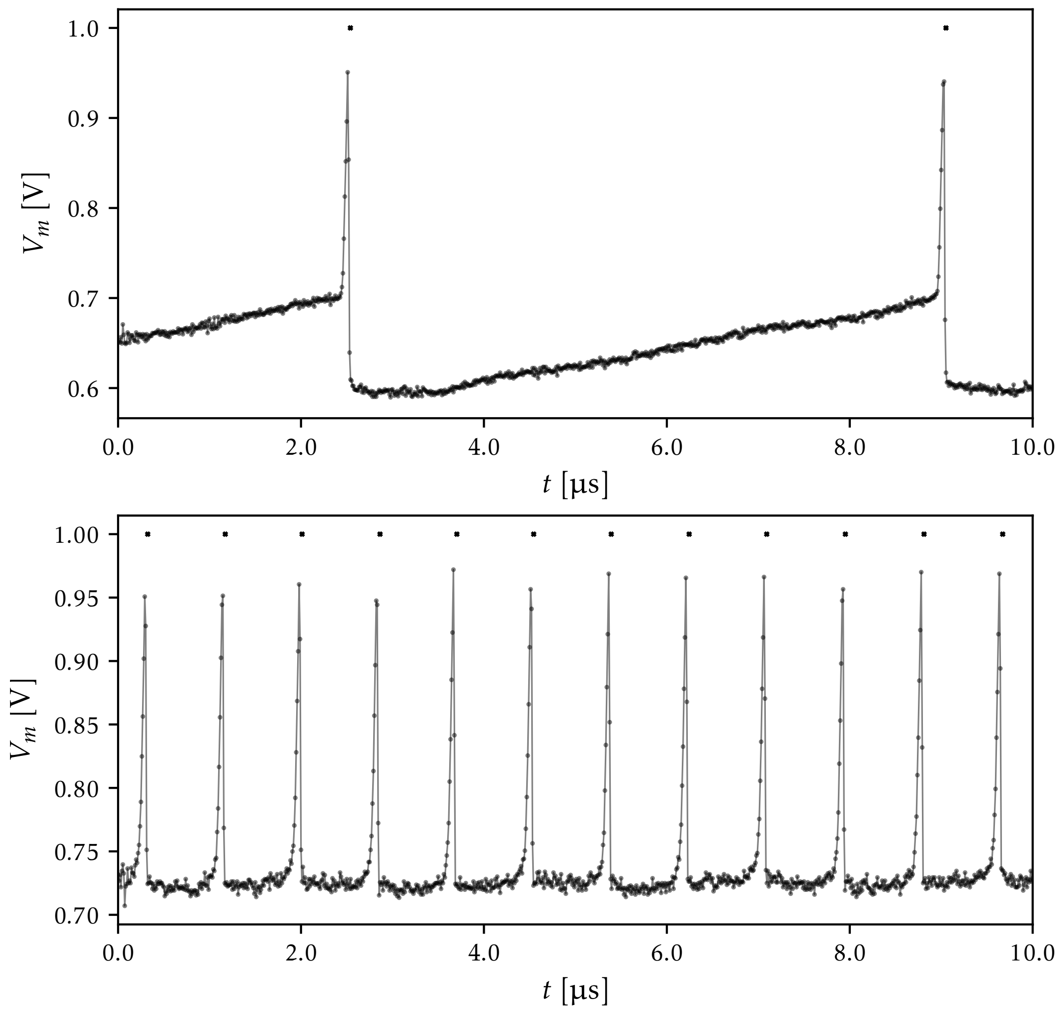


Figure 7.29: Membrane voltage traces used to characterize the exponential term. Lines show interpolation between the measured samples, helping to recognize the spikes, which are actually going up to $V_t = 1\,V$. Small crosses at $1\,V$ mark digital spike timestamps. Common parameters are $I_{pl} = 100\,nA$, $V_t = 1.0\,V$, $V_{reset} = 0.6\,V$, $\tau_m = 10\,\mu s$, $I_{rexp} = 100\,nA$. At the top $E_l = 0.8\,V$, $V_T = 0.7\,V$. At the bottom $E_l = 0.7\,V$, $V_T = 0.65\,V$. In this case we observe that the denmem is not spiking during the refractory period, but immediately afterwards. The reset potential is not reached, the reset current is not able to suppress the current from the exponential term at such small I_{rexp}, when V_T is close to V_{reset}.

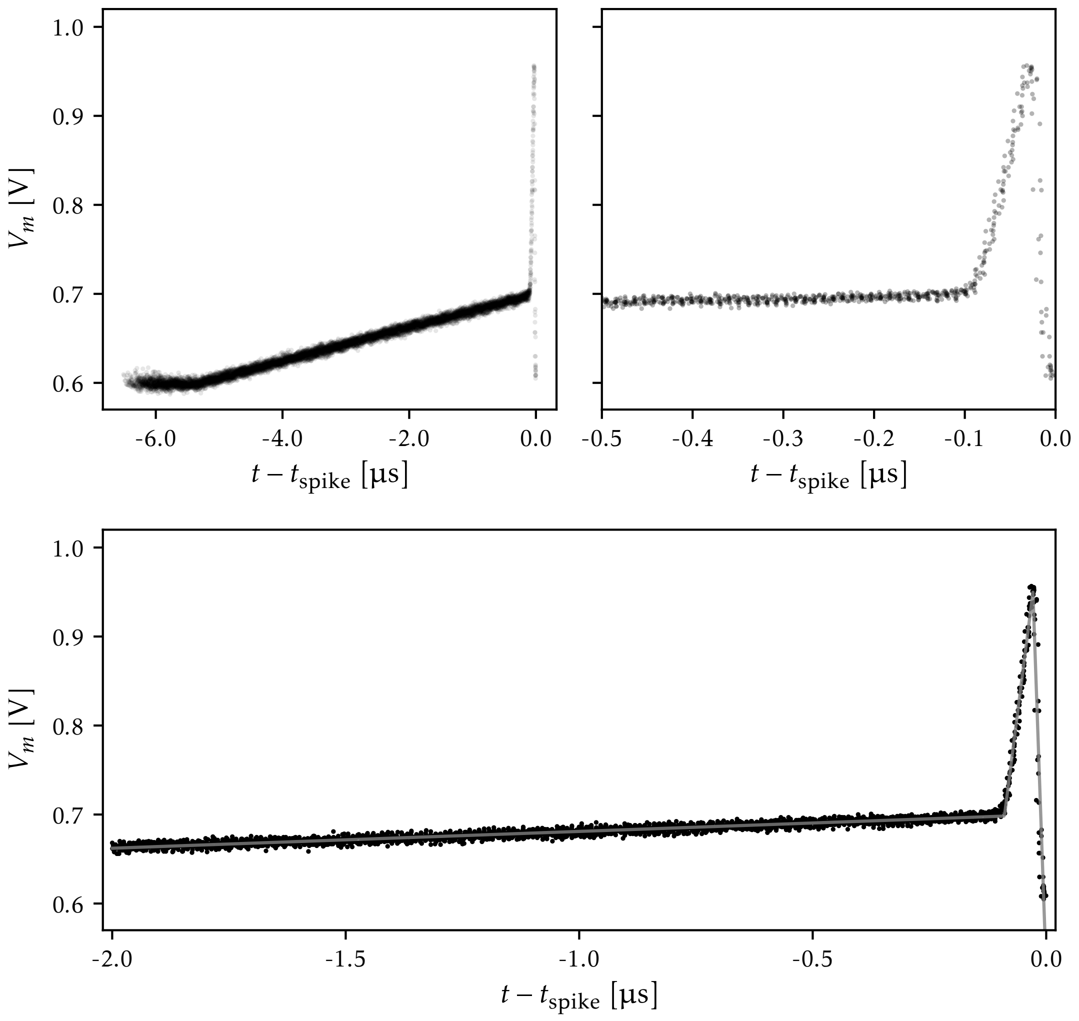


Figure 7.30: Scatterplot of the membrane potential over a single ISI, for many spikes with time shifted by the corresponding t_{spike}. The variation of t is probably caused by the binning of the digital timestamp to a FCP clock cycle. The top left shows the full ISI of 6.27 µs. The top right shows an enlargement of the spike. Due to saturation effects, $V_m(t)$ seems to be linear even when the exponential term provides the dominating current. We fit a piece-wise linear function, equation (7.36), as depicted at the bottom, to get the slope k_2 of the spike.

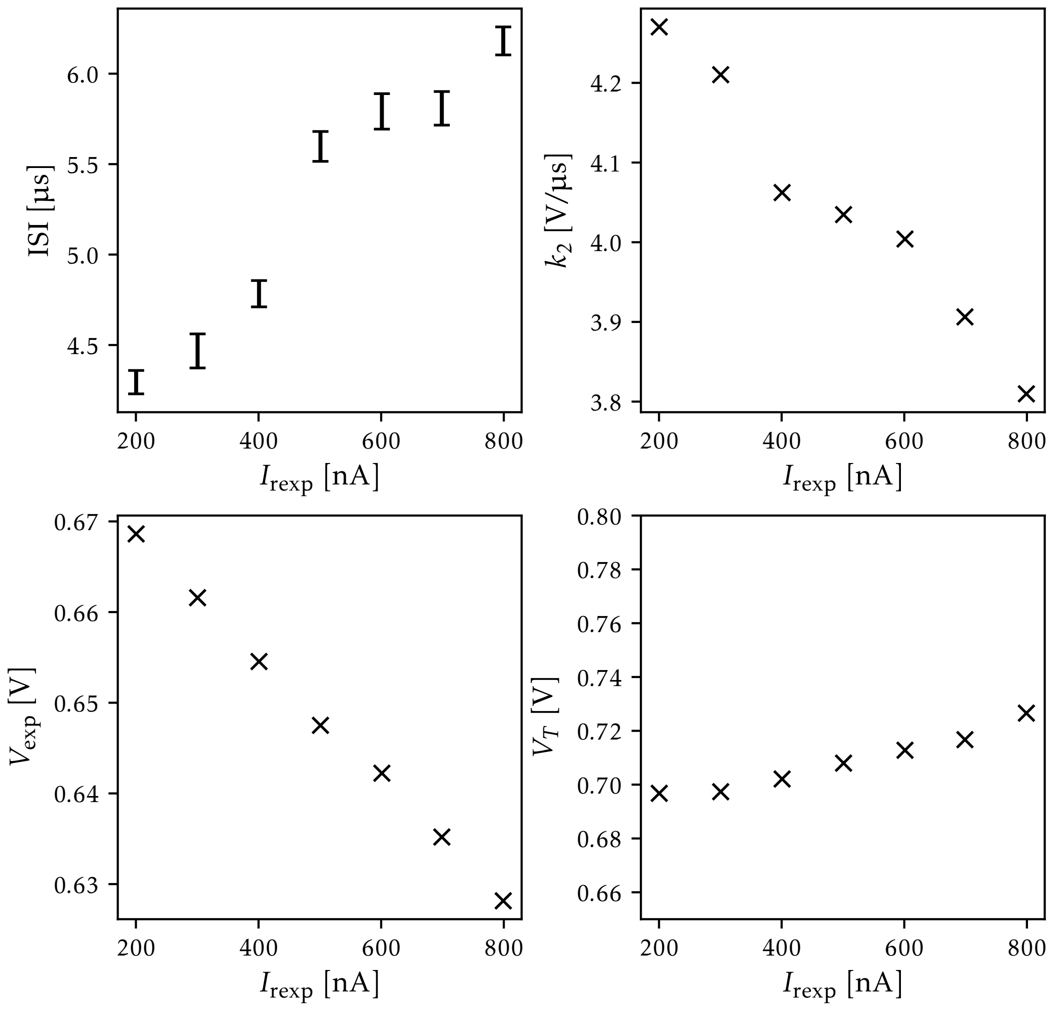


Figure 7.31: Left: Mean ISI over many spikes, the error bar shows the standard deviation. The ISI depends on the chosen combination of E_l, V_t, V_{reset}, τ_m, τ_{ref}, as well as $\Delta_T(I_{\text{rexp}})$. V_T is kept constant by adjusting V_{exp}. There is a trend towards longer ISIs for larger I_{rexp}, showing that it is possible to delay the exponential spike. Right: Coefficient k_2 of equation (7.36) for several I_{rexp}. Above $I_{\text{rexp}} = 400\,\text{nA}$, there seems to be a linear relation.

Part IV

WAFER CALIBRATION

8 WAFER-SCALE LIF CALIBRATION

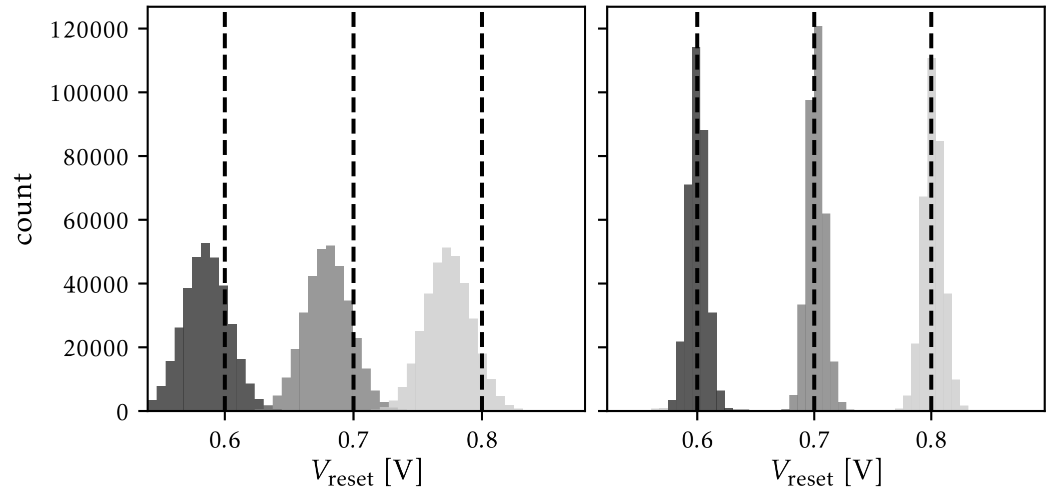
We apply the methods presented in the previous chapter to a full wafer module. The software was designed such that an arbitrary HICANN can be selected to perform the calibration experiments on. Generated result data is attributed to each specific wafer and HICANN. AnaRMs and FCPs are shared between several HICANNs, such that experiments requiring the same resource need to run in succession, while other experiments can run in parallel. These resource dependencies are represented as resources in the Slurm Workload Manager, a scheduler that is typically used by computer clusters for resource management. This approach allows to submit experiments for all HICANNs to the job queue at once: 12 experiments requiring different AnaRMs will run immediately while the other experiments are queued until the resources become available. Additional effort is required to manage the generated data, evaluate errors that occur during experiments and resume experiments that were aborted due to temporary communication errors. Managing and analyzing wafer-scale calibration data is an ongoing effort, updated and improved results will probably be published in Kugele (2017).

In the experiments, we first apply an ideal calibration – typically obtained from transistor level simulation of an ideal denmem – to each denmem and measure the resulting observable. This approach allows us to obtain a usable basic configuration without calibration, which is expected to show higher variation. Next we generate individual calibration data for each denmem. We expect being able to reduce variation by using the individual parameter transformations, and the characterization data contains the parameter ranges which can be reached by an individual denmem. Using the calibration data, we configure the parameters for a post-calibration characterization, taking the individual variation into account. Measuring each observable once for validation does not reflect the deviations caused by trial-to-trial variation of the FG cells. In order to include this distribution, each measurement is repeated four times. The potentials E_l, V_t and V_{reset} have a linear transformation function and are expected to show the least variation. Other parameters are designed with a nonlinear transformation from control parameter to observable. For example, variation of the refractory period τ_{ref} is expected to be strongest for long refractory periods. Due to the nonlinear transformation, the distribution will not follow a Gaussian distribution, assuming that the variation of the circuits is normally distributed. In general, reducing the results to quantities like mean μ and standard deviation σ is not sufficient to represent the true parameter distribution. There are many very different distributions that would yield the same μ and σ, therefore we need to provide the full information by providing the full distribution. For example, two equally large peaks of values at $\mu + \delta$ and $\mu - \delta$ still result in an average of μ. A single outlier very far from many values at the same point will result in a large σ. For the purpose of quick estimations it is still useful to provide sensible numbers summarizing the information. It makes sense to discard extreme outliers, for example by only considering 95% of the values closest to the maximum of the distribution. For the means μ and standard deviations σ presented in the following, we consider 99% of the values, selected symmetrically around the maximum.

Data has been collected for 150 HICANNs on the wafer module *WMOD G06*. Digital communication with the missing HICANNs failed in three independent runs. The main reason is that this module has been assembled with lower pressure on the elastomeric connectors for debugging purposes, causing unreliable connections to some reticles. We plot the distribution of all measurements, including multiple trials, for each parameter for both the ideal denmem calibration and the individual calibration.

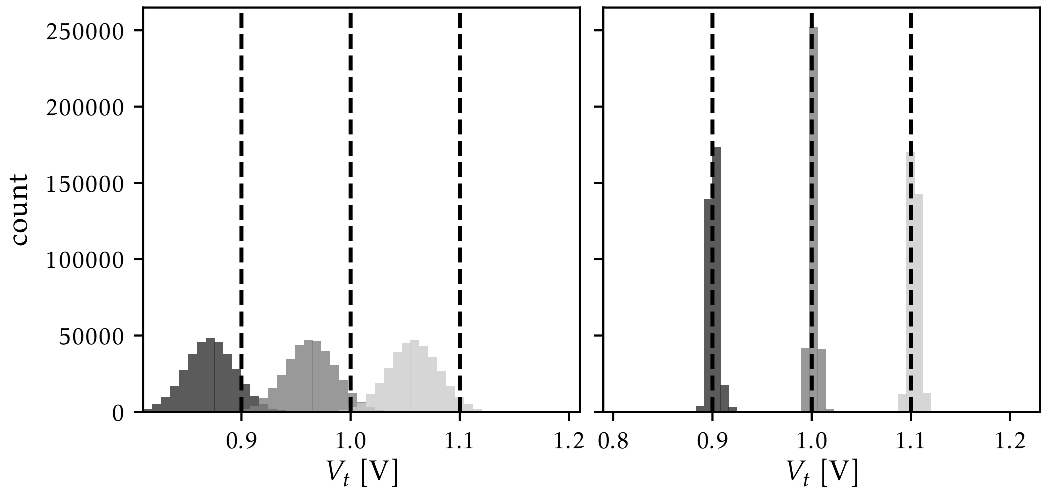
Deviation of the potentials are mainly caused by DC-offsets in active components. The active component is a comparator in case of the threshold potential V_t, the leakage OTA in case of the leakage potential E_l and a voltage buffer for V_{reset}. These offsets can be corrected using a linear transformation. Figure 8.1 shows the distribution of the reset potential V_{reset} before and after calibration. Figure 8.2 shows the distribution of the spike threshold V_t, figure 8.3 shows the distribution of the resting potential E_l. In previous measurements (Schmidt, 2014; Koke, 2016) on single chips we have shown that the standard deviation of the potentials can be reduced close to the trial-to-trial variation of the voltage cells, to about $\sigma \approx 4\,\mathrm{mV}$, in case of Koke (2016) this result was obtained from one HICANN on the same wafer module. In wafer-scale calibration on this wafer module we observe that this reduction can also be achieved for 99% of all evaluated denmems. Inspection of the data reveals that about 0.6% of the calibrated denmems miss their target potential by more than $50\,\mathrm{mV}$. The affected denmems can be blacklisted, however it may be worthwhile to inspect them manually in the future.

In order to be able to measure time constants before calibration, we apply the transformation function found for the ideal denmem in transistor level simulation. For the refractory period we observe that the maximum of the distribution with target $1\,\mu\mathrm{s}$ is significantly smaller than the target value (figure 8.4). This has not been the case in previous evaluations (Schmidt, 2014), which were using more than four repetitions during calibration and may hint that this number of repetitions is too low. For the membrane time constant (figure 8.5) we see that the ideal transformation does not yield a good configuration for two of the target time constants. The reason is that the function that was present in the software at the time the data has been recorded was incorrect. In the meantime, the coefficients have been changed to the results obtained in this thesis. We can also see that the distribution is not represented by a normal distribution due to the nonlinear transformation from FG value to conductance and resulting time constant. The distribution is broader for large time constants, which require small bias currents. In this case the effect of trial-to-trial variation is strongest.



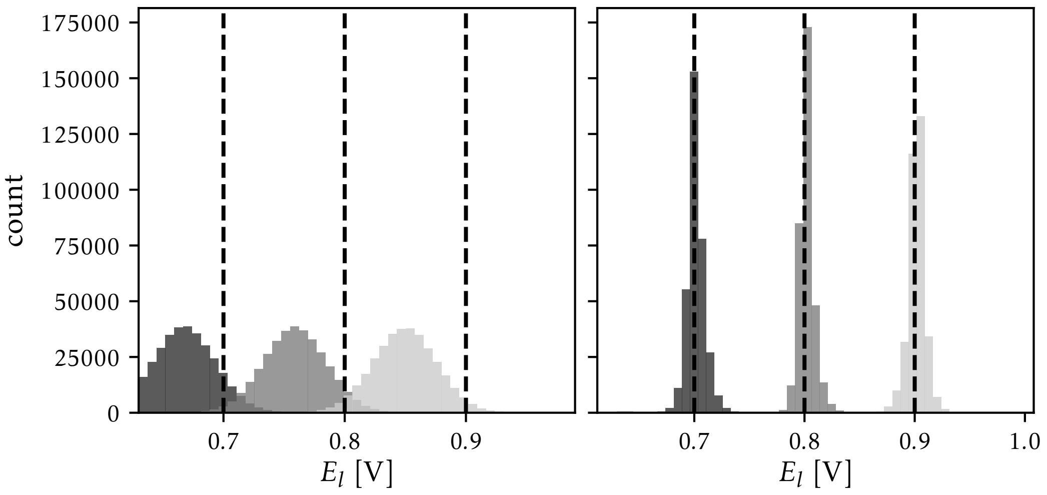
	before calibration		after calibration	
target [V]	μ [V]	σ [V]	μ [V]	σ [V]
0.6	0.585	0.017	0.600	0.008
0.7	0.680	0.017	0.701	0.007
0.8	0.774	0.018	0.802	0.008

Figure 8.1: Distribution of the reset potential V_{reset} before (left) and after calibration (right) for 150 HICANNs. Each denmem was evaluated four times in order to capture the distribution caused by trial-to-trial variation. Calibration reduces the spread of the distribution and shifts the mean towards the target value. The standard deviation does not reach the standard deviation caused by trial-to-trial variation, because the parameter is shared between 128 denmems.



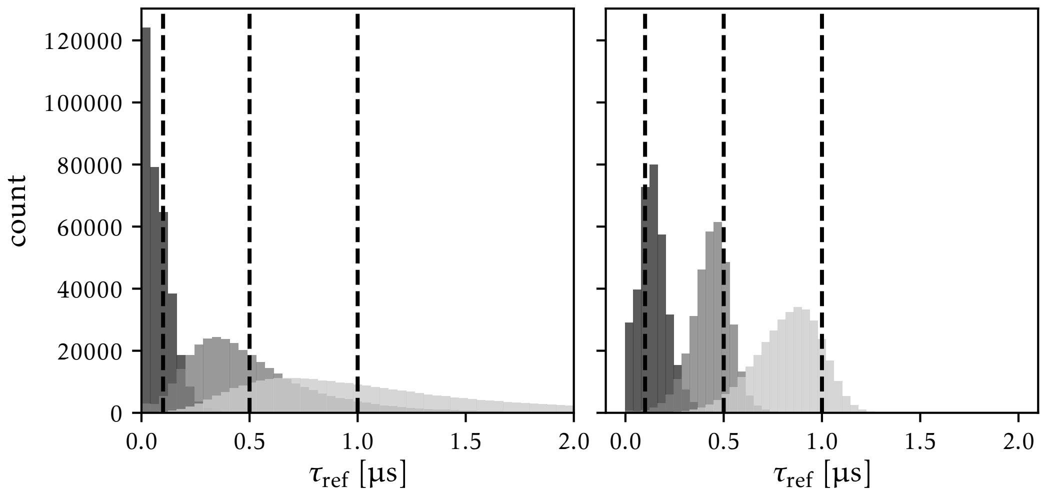
	before calibration		after calibration	
target [V]	μ [V]	σ [V]	μ [V]	σ [V]
0.9	0.872	0.022	0.901	0.004
1.0	0.963	0.022	1.002	0.004
1.1	1.058	0.022	1.104	0.004

Figure 8.2: Distribution of the threshold potential V_t before (left) and after calibration (right) for 150 HICANNs. Each denmem was evaluated four times in order to capture the distribution caused by trial-to-trial variation. Calibration reduces the spread of the distribution and shifts the mean towards the target value. The standard deviation is close to the standard deviation caused by trial-to-trial variation.



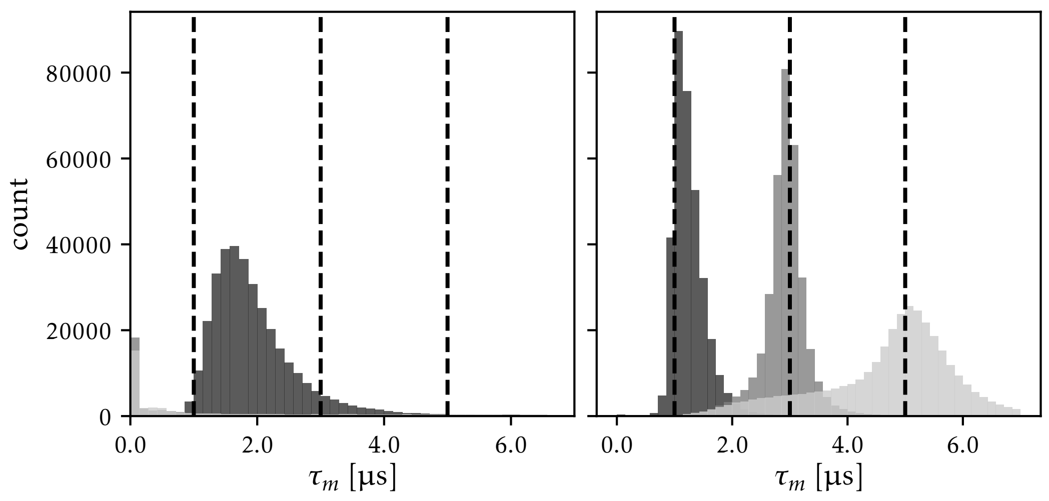
	before calibration		after calibration	
target [V]	μ [V]	σ [V]	μ [V]	σ [V]
0.7	0.668	0.025	0.701	0.008
0.8	0.759	0.025	0.802	0.006
0.9	0.850	0.025	0.902	0.007

Figure 8.3: Distribution of the resting potential E_l before (left) and after calibration (right) for 150 HICANNs. Each denmem was evaluated four times in order to capture the distribution caused by trial-to-trial variation. Calibration reduces the spread of the distribution and shifts the mean towards the target value. The distribution after calibration is coarser than the distribution of V_t due to leakage currents from other terms, which can not be entirely disabled and are also subject to variation.



	before calibration		after calibration	
target [µs]	µ [µs]	σ [µs]	µ [µs]	σ [µs]
0.1	0.072	0.062	0.139	0.071
0.5	0.513	0.302	0.434	0.097
1.0	1.168	0.721	0.823	0.162

Figure 8.4: Distribution of the refractory period τ_{ref} before (left) and after calibration (right) for 150 HICANNs. Each denmem was evaluated four times in order to capture the distribution caused by trial-to-trial variation. The distribution gets broader for larger refractory periods. Calibration reduces the spread of the distribution. In this case the mean does not reach the target values. The calibration function is not optimal, and the number of repetitions during calibration may be too low. Improving the refractory period calibration is currently under investigation.



	before calibration		after calibration	
target [µs]	µ [µs]	σ [µs]	µ [µs]	σ [µs]
1.0	1.927	0.640	1.242	0.255
3.0	n/a	n/a	2.894	0.364
5.0	n/a	n/a	4.727	1.189

Figure 8.5: Distribution of the membrane time constant $\tau_m(I_{gl})$ before (left) and after calibration (right) for 150 HICANNs. Each denmem was evaluated four times in order to capture the distribution caused by trial-to-trial variation. The translation function that was used to determine the setting without calibration does not yield correct configuration for two of the evaluated time constants. Calibration leads to a better translation function, reduces the spread of the distribution and shifts the mean towards the target value. The distribution gets broader for larger time constants and is skewed due to the nonlinear transformation.

Part V

DISCUSSION

9 CONCLUSION AND OUTLOOK

With this thesis we have provided methods for the neuron circuit characterization of HICANN both in transistor level simulations and wafer-scale hardware measurements. At the same time we were able to contribute to the overall usability and stability of the platform: the calibration framework operates at the same abstraction level as the mapping software (compare figure 6.1) and depends on many features of the underlaying software. While it certainly does not replace fine-grained unit testing of all components, developing the calibration framework has also served as an integration test for the software layers below, as well as most of the hardware functionality. During the debugging process this has also lead to development of additional unit tests in several layers.

The experience and challenges that we faced during calibration of the denmem circuits served as input for the next generation chip and the overall system design: much faster programming time and significantly lower trial-to-trial variation of analog parameters have been implemented in a capacitive memory (Hock, 2014) for the next generation system, which significantly increases the experiment rate and reduces variation. The noise that we observe on the analog outputs will be reduced as well (Ilmberger, 2017). Being able to measure quantities besides the membrane voltage and disconnect individual circuits to be able to measure or use a subset of circuits (Aamir et al., 2016) allows more precise characterization methods (Stradmann, 2016).

For the current system we were able to provide a general-purpose calibration that is already being used for experiments. As proposed in Schmidt (2014), the transformation functions have been changed to map from desired observable to the required FG setting, while providing a domain of target values which can be reached by a particular denmem. This approach allows the mapping software to choose suitable denmems for parameters at the edge of the achievable range.

Additional work is still required on fine-tuning of the calibration configuration for a desired trade-off between accuracy and measuring time. The desired accuracy has not been fully specified so far, while parameter ranges and functional test cases have been assembled in Müller (2017). The configurability of the calibration framework allows to adapt such settings to fulfil the requirements.

9.1 DISCUSSION OF METHODS

The immediate results of each method have been discussed in their corresponding section. In the following, we present ideas how to further improve specific methods.

SPIKE THRESHOLD The spike threshold calibration works very reliably already. We chose a short membrane time constant in order to record as many spikes as possible in any recording interval. However, this may introduce a systematic error because the largest sample during a spike is not neccessarily sampled at the peak. A long membrane time constant increases the probability that an ADC sample is close to the actual spike threshold. At the same time a longer membrane time constant requires a longer recording period to be able to average over the same number of spikes.

MEMBRANE TIME CONSTANT We are using a constant current stimulus that will not raise the membrane potential more than $150\,\mathrm{mV}$ above the resting potential for the slowest configured time constant. This leads to a very small voltage raise at faster time constants. Instead, we could use a larger stimulus in measurements with faster time constants. The automated choice of the stimulus amplitude for a particular denmem could also be improved: currently the evaluation of the response begins at a small stimulus, which is increased until the resulting raise in membrane potential is above $150\,\mathrm{mV}$. The highest amplitude before the last increase is chosen for this denmem. Instead, we could for example implement a half-interval search.

We could also allow the current stimulus to raise the membrane potential more than $150\,\mathrm{mV}$, while avoiding to raise above V_t or $1.1\,\mathrm{V}$. In order to avoid fitting in the OTA saturation regime, we could only consider the part below $V_m = E_l + 150\,\mathrm{mV}$ in the fitting routine.

Although the parasitic capacitance added by the stimulus line can not be measured accurately, there is a way to measure the decaying part of the trace without parasitic capacitance: instead of configuring a rectangular current stimulus, the amplitude can be configured to remain constant. The stimulus of a denmem is then disabled by disabling the switch that connects the current stimulus to this denmem. This method will be possible once the playback memory based configuration is supported by the FCP.

REFRACTORY PERIOD During calibration of the refractory period, trial-to-trial variations of the parameters V_t, E_l, I_{gl} and V_{reset} add a systematic error due to the resulting variation of the rise time. This influence could be removed by detecting the rise time and subtracting it from the ISI. However, this detection may add a systematic error in the same order of magnitude. In the presented method we assume that the smallest refractory period is actually zero and the reset is instantaneous. Transistor level simulations and inspection of ADC samples during the reset suggest that the shortest time between crossing the spike threshold and the membrane potential beginning to rise again from V_{reset} is approximately $50\,\mathrm{ns}$. This could be analyzed more thouroughly and be considered when measuring the reference ISI.

The nonlinear mapping from floating gate configuration to refractory time makes it difficult to accurately set long refractory periods. This will be changed in the next generation chip by making the release of the reset current a digital feature.

ADAPTATION The leakage potential E_l is connected to both the leakage OTA and the adaptation OTA. Variation will cause different offsets in each OTA, which can not be compensated and lead to different effective potentials. As suggested by Millner (2012), an offset compensation, similar to the one found in the synaptic input circuit, could be added to the adaptation OTA, controlled by the remaining unused voltage cell. Being able to measure V_w would greatly improve calibration options. The membrane capacitance interconnectivity could be extended to interconnect C_w as well.

EXPONENTIAL TERM As we have seen in transistor level simulation, the membrane voltage range at which the current from the exponential term depends exponentially on the membrane potential is very small. Saturation leads to a strong linear dependency shortly after the exponential regime. We have seen in previous works (Millner, 2012; Schwartz, 2013; Tran, 2013) that this implementation still allows to reproduce spike patterns that require an exponential term.

By using the subthreshold current of a transistor, the circuit design introduces a strong temperature dependency, making it important to characterize the circuit at the actual temperature during operation. At the current stage of system assembly, the operating temperature still depends on the number of reticles that are powered and their configuration. The final configuration and climate control are still under development. We have discussed the temperature dependence with the analog designers of the next generation chip, which will probably implement a design that is more robust against temperature changes.

9.2 TIME REQUIREMENT OF CALIBRATION

We have shown that scaling the calibration from a single HICANN to wafer-scale characterization is already possible. One remaining goal is to optimize the configuration with regard to the overall experiment duration of the calibration, as well as the accuracy of the characterization.

The parameters that affect the experiment duration are currently based on experience. For the fastet possible calibration at acceptable precision we could define a required minimum precision of each parameter and find the number of steps accordingly to achieve this precision.

As long as the transformation from digital setting to resulting observable is linear, having twice as many repetitions of each step is expected to give the same accuracy as twice the number of steps. For parameters with nonlinear transformation from digital setting to resulting observable, several repetitions become more important, as the error caused by trial-to-trial variation follows the nonlinear transformation. More points are required close to the diverging region of the transformation function.

It may be possible to find shared configurations that can be evaluated in several calibration methods, while requiring just a single experiment. For example, the reset and threshold potential could be measured at the same time. We evaluated this option for the previous chip revision and discarded it due to restrictions of the synaptic input circuit, which has been replaced in the meantime.

A renewed characterization of the analog parameter storage may also be worthwhile. For example, we are currently rewriting all parameters during configuration. The FG controller allows to update a parameter row individually, which would speed up parameter changes and avoid trial-to-trial variation of unchanged parameters. However, the cross-talk effects during row-wise reconfiguration still need to be characterized.

Contributions to the overall duration caused by data transfer are expected to be reduced with a replacement for the AnaRM, called Analog Network Attached Sampling unit (ANANAS), that is still under development (Ilmberger, 2017). This module allows to transmit data via Gigabit Ethernet, significantly increasing transfer speed. The maximum sampling rate will be reduced to $32.5\,\mathrm{MS\,s^{-1}}$, resulting in less data for a given experiment duration. Noise is expected to be reduced by an anti-aliasing low-pass filter on the module. This will reduce the need to record long traces that can be used for averaging. Experiments that only require a low sampling rate can be parallelized further by using the slow ADCs on the ANANAS. The parallel readout of denmems is going to require adapted software support. Combined with parallel configuration of several HICANNs on a reticle this will also lead to a faster wafer-calibration.

After re-evaluation of spike timestamp reliability, the ISI-based methods could be adapted to purely rely on spike timestamps instead of voltage traces, which allows to measure several denmems simultaneously with different addresses.

Another approach to reduce the amount of data being transferred would be to perform some of the data analysis currently performed in software directly on the FPGA of the AnaRM. However, this approach requires significant additional development effort.

An important new FCP feature will be the support of modifying digital chip configuration during an experiment run. In that case a configuration change is stored in Playback Memory (PbMem) (Pilz, 2016) with a timestamp and will be sent to HICANN at that time. Several steps during the measurement, for example switching the analog output between denmems, are currently being performed with communication between host computer and FCP. As soon as the playback memory can be used to buffer these commands (Pilz, 2016), the steps can be performed without communication to the host, further reducing the configuration time. Changing FG parameters during the experiment is excluded as changing their value does not happen instantaneously. Being able to change digital configuration during the experiment is expected to speed up our measurement routines, as many measurements just require switching the analog output to another denmem before aquiring a new ADC trace.

9.3 current and future network experiments

This thesis contributed significantly to the success of several neural network experiments performed on the current system (Schmitt et al., 2016; Klähn, 2017; Kungl, 2016; Alevi, 2015; Kugele, 2017). The next generation of the BrainScaleS system, which will be particularly interesting due to its advanced learning capabilities, is currently being developed and includes many improvements, often inspired by the experiences gained while commissioning and calibrating the existing hardware.

Part VI

APPENDIX

A EXPERIMENT DURATION

In the following, we will give a rough estimate how long an experiment will take.

An experiment, sweeping a single parameter on one HICANN, is a series of several sweep steps. Each step consists of configuration, AnaRM recording and transfer of the recorded data via USB 2.0 for each denmem and analysis of the recorded data.

The exact numbers depend on the configured number of steps and repetitions as well as the recording time required for the method. They will vary between methods and depend on the selected compromise between accuracy and duration. For this rough estimation we choose an average of 8 steps and 4 repetitions at a recording time of 100 µs. Chip configuration, dominated by the runtime of the FG controller, would take

$$\begin{array}{rl} 15\,\mathrm{s} & \text{configuration time} \\ \times\ 8 & \text{steps} \\ \times\ 4 & \text{repetitions} \\ \hline 480\,\mathrm{s} & \text{total configuration time.} \end{array}$$

The recording takes

$$\begin{array}{rl} 100\,\mu\mathrm{s} & \text{recording of one denmem} \\ \times\ 512 & \text{denmems} \\ \times\ 8 & \text{steps} \\ \times\ 4 & \text{repetitions} \\ \hline 3.2\,\mathrm{ms} & \text{total recording time.} \end{array}$$

Several methods require longer recording of each denmem, for example when using the current stimulus: one stimulus period at a pulse length of 16, operating at 100 MHz, takes

$$(16 \cdot 129) \cdot (100/4\,\mathrm{MHz})^{-1} = 82.56\,\mu\mathrm{s}.$$

As we need to average over multiple periods to reduce the readout noise, the recording time of a single denmem can be in the order of milliseconds when the current stimulus is required by the method. Recording of a PSP requires 12 ms (Koke, 2016).

The recording time itself is very short, however the recorded amount of data affects the transfer duration. We calculate the transfer of the recorded data at an AnaRM transfer rate of $300\,\mathrm{MiB\,s^{-1}}$ (Koke, 2016):

$12\,\mathrm{bit}$	sample size
$\times\ 96\,\mathrm{MS\,s^{-1}}$	sample rate
$\times\ 100\,\mathrm{\mu s}$	recording time
$\times\ 8$	steps
$\times\ 512$	denmems
$/\ 300\,\mathrm{MiB\,s^{-1}}$	transfer rate
$0.79\,\mathrm{s}$	total transfer time.

At a long recording time of $12\,\mathrm{ms}$, the corresponding transfer time is $94.37\,\mathrm{s}$.

The total experiment duration in this rough estimate is

$480\,\mathrm{s}$	total configuration time
$+\ 3.2\,\mathrm{ms}$	total recording time
$+\ 0.79\,\mathrm{s}$	total transfer time
$\approx 480.79\,\mathrm{s}$	typical experiment duration.

In this example, the configuration time is the dominating part of the total duration. At longer recording times, the data transfer time becomes relevant as well.

Calibration of one HICANN requires roughly this experiment duration, multiplied by the number of calibrated parameters. 12 HICANNs can be calibrated in parallel, resulting in a total expected duration of

$480\,\mathrm{s}$	typical experiment duration
$\times\ 23$	parameters
$\times\ 384$	HICANNs
$/\ 12$	in parallel
$= 353\,280\,\mathrm{s} \approx 4\,\mathrm{days}$	full wafer calibration.

This estimate does not take the time for data analysis into account, because in practice this contribution is much smaller than the other factors listed below. Data analysis is already performed in parallel for several denmems on the available processor cores. If other contributions to the overall duration can be reduced significantly in the future, the analysis could be adapted to run in parallel to the experiments.

The configuration we are currently using for wafer calibration does not include all parameters. Additionally several HICANNs are excluded due to communication issues. The total duration of our current wafer calibration configuration is approximately 2-4 days.

B PARAMETER SETTINGS

In this chapter we provide typical settings which are used during calibration. There is a base configuration, which is common between all experiments. These settings are changed by non-default parameter settings for a specific experiment. During the experiment, one or multiple parameters are changed in each step in order to determine the parameter dependence of the observable.

B.1 BASE CONFIGURATION

This default configuration is used as the basis for every experiment. Deviating settings can be defined in the experiment configuration to override these settings.

Neuron parameters:

$I_{breset} = 1023\ \mathrm{DAC}$	$I_{bstim} = 1023\ \mathrm{DAC}$	$V_{bexp} = 2.500\,\mu A$	$V_{bout} = 0.750\,\mu A$
$V_{br} = 0\ \mathrm{DAC}$	$V_{bstdf} = 0\ \mathrm{DAC}$	$V_{ccas} = 800\ \mathrm{DAC}$	$V_{clra} = 0\ \mathrm{DAC}$
$V_{clrc} = 0\ \mathrm{DAC}$	$V_{dep} = 0\ \mathrm{DAC}$	$V_{dllres} = 200\ \mathrm{DAC}$	$V_{dtc} = 0\ \mathrm{DAC}$
$V_{fac} = 0\ \mathrm{DAC}$	$V_{gmax0} = 0.080\,V$	$V_{gmax1} = 0.080\,V$	$V_{gmax2} = 0.080\,V$
$V_{gmax3} = 0.080\,V$	$V_m = 0\ \mathrm{DAC}$	$V_{reset} = 0.500\,V$	$V_{stdf} = 0\ \mathrm{DAC}$
$V_{thigh} = 0\ \mathrm{DAC}$	$V_{tlow} = 0\ \mathrm{DAC}$	$\mathrm{int\_op\_bias} = 1023\ \mathrm{DAC}$	

Shared parameters:

$E_l = 0.800\,V$	$E_{syni} = 0.600\,V$	$E_{synx} = 1.300\,V$	$I_{bexp} = 2.500\,\mu A$
$I_{convi} = 0.625\,\mu A$	$I_{convx} = 0.625\,\mu A$	$I_{fire} = 0.000\,\mu A$	$I_{gl} = 1.000\,\mu A$
$I_{gladapt} = 0.000\,\mu A$	$I_{intbbi} = 2.000\,\mu A$	$I_{intbbx} = 2.000\,\mu A$	$I_{pl} = 2.000\,\mu A$
$I_{radapt} = 2.500\,\mu A$	$I_{rexp} = 2.500\,\mu A$	$I_{spikeamp} = 2.000\,\mu A$	$V_{convoffi} = 1.800\,V$
$V_{convoffx} = 1.800\,V$	$V_{exp} = 1.800\,V$	$V_{syni} = 0.900\,V$	$V_{syntci} = 1.420\,V$
$V_{syntcx} = 1.420\,V$	$V_{synx} = 0.900\,V$	$V_t = 1.000\,V$	

B.2 RESTING POTENTIAL

The following parameters are used in a typical calibration of the resting potential E_l.

Step parameters: E_l = 0.7 V, 0.8 V and 0.9 V

Non-default parameters: V_{reset} = 0.9 V (calibrated), V_t = 1.2 V (calibrated), I_{convi} = 0 μA, I_{convx} = 0 μA

B.3 RESET POTENTIAL

Step parameters: $E_l = 0.90\,V$, $1.00\,V$, $1.10\,V$ and $1.20\,V$ $V_{reset} = 0.50\,V$, $0.60\,V$, $0.70\,V$ and $0.80\,V$ $V_t = 0.70\,V$, $0.80\,V$, $0.90\,V$ and $1.00\,V$

Non-default parameters: $I_{pl} = 0.020\,\mu A$, $I_{convi} = 0.000\,\mu A$, $I_{convx} = 0.000\,\mu A$, $I_{gl} = 1.100\,\mu A$

B.4 THRESHOLD POTENTIAL

Step parameters: $E_l = 0.90\,V$, $1.00\,V$, $1.10\,V$, $1.20\,V$ and $1.30\,V$ $V_{reset} = 0.50\,V$, $0.60\,V$, $0.70\,V$, $0.80\,V$ and $0.90\,V$ $V_t = 0.70\,V$, $0.80\,V$, $0.90\,V$, $1.00\,V$ and $1.10\,V$

Non-default parameters: $I_{convi} = 0.000\,\mu A$, $I_{convx} = 0.000\,\mu A$, $I_{gl} = 1.500\,\mu A$

B.5 REFRACTORY PERIOD

Step parameters: $I_{pl} = 10.00\,DAC$, $15.00\,DAC$, $20.00\,DAC$, $30.00\,DAC$, $40.00\,DAC$, $60.00\,DAC$ and $80.00\,DAC$

Non-default parameters: $E_l = 1.200\,V$, $E_{syni} = 0.800\,V$ (calibrated), $E_{synx} = 1.200\,V$, $V_t = 0.800\,V$, $V_{reset} = 0.500\,V$ (calibrated)

B.6 MEMBRANE TIME CONSTANT

Step parameters: $I_{gl} = 0.10\,\mu A$, $0.20\,\mu A$, $0.30\,\mu A$, $0.40\,\mu A$, $0.50\,\mu A$, $0.60\,\mu A$, $0.70\,\mu A$ and $0.80\,\mu A$

Non-default parameters: $E_l = 0.800\,V$ (calibrated), $E_{syni} = 0.600\,V$ (calibrated), $E_{synx} = 1.300\,V$, $V_t = 1.200\,V$, $V_{convoffi} = 1.800\,V$ (calibrated), $V_{reset} = 0.200\,V$ (calibrated), $I_{convi} = 0.000\,\mu A$, $V_{convoffx} = 1.800\,V$ (calibrated), $I_{convx} = 0.000\,\mu A$

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

ADC	Analog-to-Digital Converter
AdEx	Adaptive Exponential Leaky Integrate-and-Fire
AnaFP	Analog Frontend PCB
ANANAS	Analog Network Attached Sampling unit
AnaRM	Analog Readout Module
API	Application Programming Interface
ARQ	Automated Repeat reQuest
ASIC	Application Specific Integrated Circuit
BSIM	Berkeley Short-channel IGFET Model
cake	calibration framework for experiments
CMOS	complementary metal–oxide–semiconductor
DAC	Digital-to-Analog Converter
DLL	delay-locked loop
DNC	Digital Network Chip
denmem	Dendritic Membrane Circuit
EA	event address
FCP	FPGA Communication PCB
FG	floating gate
FPGA	field-programmable gate array
HAL	Hardware Abstraction Layer
HALbe	Hardware Abstraction Layer Backend
HBP	Human Brain Project
HICANN	High Input-Count Analog Neural Network
HS	high-speed

ISI	Inter-Spike Interval
L1	Layer 1
LIF	Leaky Integrate-and-Fire
LVDS	low voltage differential signal
MPW	Multi-Project Wafer
NM-PM1	Neuromorphic Physical Model version 1
OpAmp	operational amplifier
OTA	operational transconductance amplifier
PbMem	Playback Memory
PCB	printed circuit board
PMOS	Positive Metal–Oxide–Semiconductor
PSP	post-synaptic potential
SEB	System Emulator Board
STA	Spike-Triggered Adaptation
STDF	Short Term Depression and Facilitation
STDP	Spike-Timing Dependent Plasticity
STP	Short Term Plasticity
StHAL	Stateful Hardware Abstraction Layer
USB	Universal Serial Bus
VLSI	very-large-scale integration

ACKNOWLEDGMENTS

First of all, I would like to express my gratitude to Prof. Meier for giving me the opportunity to be part of his group and work on this thesis. I am also grateful towards my second referee Prof. Hausmann, as well as Prof. Fischer and Prof. Bartelmann for being part of my examination commitee and agreeing to review my work without hesitation.

Sincere thanks are also given to Johannes Schemmel, Andreas Grübl and the Visions group for the countless interesting discussions and their willingness to answer many questions on my side. Christoph Koke, Sebastian Schmitt, Andreas Hartel, Sebastian Billaudelle and Mihai Petrovici for proofreading and discussions.

Special thanks also go to my students Dominik Schmidt, Arno Friedrich, Denis Alevi and all other contributors doing their best in developing or improving the current and next generation of the BrainScaleS system.

Last, but not least, I am very grateful for the support by Anne and my family during this time.

BIBLIOGRAPHY

Aamir, Syed Ahmed, Paul Müller, Andreas Hartel, Johannes Schemmel, and Karlheinz Meier (2016). "A highly tunable 65-nm CMOS LIF neuron for a large-scale neuromorphic system." In: *Proceedings of IEEE European Solid-State Circuits Conference (ESSCIRC)* (cit. on p. 109).

Alevi, Denis (2015). *Investigating Competitive Dynamics in a Recurrent Neural Network on Neuromorphic Hardware*. Bachelor thesis (cit. on pp. 54, 113).

Allen, P. E. and D. R. Holberg (2002). *CMOS Analog Circuit Design*. 198 Madison Avenue, New York: Oxford University Press, Inc. ISBN: 0-19-511644-5 (cit. on p. 22).

Amir, Arnon, Pallab Datta, William P Risk, Andrew S Cassidy, Jeffrey A Kusnitz, S.K. Esser, Alexander Andreopoulos, Theodore M Wong, Myron Flickner, Rodrigo Alvarez-Icaza, et al. (2013). "Cognitive computing programming paradigm: a corelet language for composing networks of neurosynaptic cores." In: *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, pp. 1–10 (cit. on p. 5).

Badel, Laurent, Sandrine Lefort, Romain Brette, Carl C. H. Petersen, Wulfram Gerstner, and Magnus J. E. Richardson (2008). "Dynamic I-V Curves Are Reliable Predictors of Naturalistic Pyramidal-Neuron Voltage Traces." In: *Journal of Neurophysiology* 99.2, pp. 656–666. ISSN: 0022-3077. DOI: 10.1152/jn.01107.2007. eprint: http://jn.physiology.org/content/99/2/656.full.pdf. URL: http://jn.physiology.org/content/99/2/656 (cit. on p. 11).

Benjamin, Ben Varkey, Peiran Gao, Emmett McQuinn, Swadesh Choudhary, Anand R Chandrasekaran, Jean-Marie Bussat, Rodrigo Alvarez-Icaza, John V Arthur, Paul A Merolla, and Kwabena Boahen (2014). "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations." In: *Proceedings of the IEEE* 102.5, pp. 699–716 (cit. on pp. 4, 5).

Billaudelle, Sebastian (2014). *Characterisation and Calibration of Short Term Plasticity on a Neuromorphic Hardware Chip*. Bachelor thesis (cit. on pp. 15, 31, 33, 54).

— (2017). "Design and Implementation of a Short Term Plasticity Circuit for a 65 nm Neuromorphic Hardware System." MA thesis. Ruprecht-Karls-Universität Heidelberg (cit. on p. 35).

Bojarski, Mariusz et al. (2016). "End to End Learning for Self-Driving Cars." In: *CoRR* abs/1604.07316. URL: http://arxiv.org/abs/1604.07316 (cit. on p. 3).

Brette, R. and W. Gerstner (2005). "Adaptive Exponential Integrate-and-Fire Model as an Effective Description of Neuronal Activity." In: *J. Neurophysiol.* 94, pp. 3637–3642. DOI: NA (cit. on p. 10).

Brüderle, Daniel (2009). "Neuroscientific Modeling with a Mixed-Signal VLSI Hardware System." PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on p. 59).

Brüderle, Daniel, Eric Müller, Andrew Davison, Eilif Muller, Johannes Schemmel, and Karlheinz Meier (2009). "Establishing a Novel Modeling Tool: A Python-based Interface for a Neuromorphic Hardware System." In: *Front. Neuroinform.* 3.17 (cit. on p. 49).

Brüderle, Daniel, Mihai A. Petrovici, et al. (2011). "A comprehensive workflow for general-purpose neural modeling with highly configurable neuromorphic hardware systems." In: *Biological Cybernetics* 104 (4), pp. 263–296. ISSN: 0340-1200. URL: `http://dx.doi.org/10.1007/s00422-011-0435-9` (cit. on p. 50).

Cadence Design Systems, Inc (2012). *Virtuoso Multi-Mode Simulation*. www.cadence.com (cit. on p. 35).

Cassidy, Andrew S, Paul Merolla, John V Arthur, S.K. Esser, Bryan Jackson, Rodrigo Alvarez-Icaza, Pallab Datta, Jun Sawada, Theodore M Wong, Vitaly Feldman, et al. (2013). "Cognitive computing building block: A versatile and efficient digital neuron model for neurosynaptic cores." In: *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, pp. 1–10 (cit. on p. 4).

Davison, A. P., D. Brüderle, J. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger (2008). "PyNN: a common interface for neuronal network simulators." In: *Front. Neuroinform.* 2.11 (cit. on pp. 49, 50).

Dresden, TU (2008). *DNC Specification*. FACETS project internal documentation (cit. on p. 39).

— (2010). *Design the final digital network ASIC*. FACETS Deliverable D7-8 (cit. on p. 39).

Drubach, Daniel (2000). *The Brain Explained*. Prentice Hall Health. ISBN: 9780137961948 (cit. on p. 3).

Epp, Bernard (2016). *Charakterisierung und Vergleich von zwei ADC-Modulen für das Brainscales Hardware System*. Bachelor thesis (cit. on pp. 43, 44, 46, 54).

Esser, S.K., Alexander Andreopoulos, Rathinakumar Appuswamy, Pallab Datta, Davis Barch, Arnon Amir, John Arthur, Andrew Cassidy, Myron Flickner, Paul Merolla, et al. (2013). "Cognitive computing systems: Algorithms and applications for networks of neurosynaptic cores." In: *Neural Networks (IJCNN), The 2013 International Joint Conference on*. IEEE, pp. 1–10 (cit. on p. 4).

Esser, S.K., Paul A Merolla, John V Arthur, Andrew S Cassidy, Rathinakumar Appuswamy, Alexander Andreopoulos, David J Berg, Jeffrey L McKinstry, Timothy Melano, Davis R Barch, et al. (2016). "Convolutional networks for fast, energy-efficient neuromorphic computing." In: *Proceedings of the National Academy of Sciences*, p. 201604850 (cit. on p. 5).

Fieres, J., J. Schemmel, and K. Meier (2008). "Realizing Biological Spiking Network Models in a Configurable Wafer-Scale Hardware System." In: *Proceedings of the 2008 International Joint Conference on Neural Networks (IJCNN)* (cit. on p. 25).

Fourcaud-Trocmé, Nicolas, David Hansel, Carl Van Vreeswijk, and Nicolas Brunel (2003). "How spike generation mechanisms determine the neuronal response to fluctuating inputs." In: *Journal of Neuroscience* 23.37, pp. 11628–11640 (cit. on pp. 10, 11).

Fowler, Martin and Kent Beck (1999). *Refactoring: improving the design of existing code*. Addison-Wesley Professional (cit. on p. 54).

Friedmann, S., J. Schemmel, A. Grübl, A. Hartel, M. Hock, and K. Meier (2016). "Demonstrating Hybrid Learning in a Flexible Neuromorphic Hardware System." In: *IEEE Transactions on Biomedical Circuits and Systems* PP.99, pp. 1–15. ISSN: 1932-4545. DOI: `10.1109/TBCAS.2016.2579164` (cit. on p. 15).

Friedmann, Simon (2013). "A New Approach to Learning in Neuromorphic Hardware." PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on p. 15).

Friedmann, Simon, Nicolas Frémaux, Johannes Schemmel, Wulfram Gerstner, and Karlheinz Meier (2013). "Reward-based learning under hardware constraints — using a RISC processor embedded in a neuromorphic substrate." In: *Frontiers in Neuroscience* 7, p. 160. ISSN: 1662-453X. DOI: `10.3389/fnins.2013.00160`. URL: `http://journal.frontiersin.org/article/10.3389/fnins.2013.00160` (cit. on p. 15).

Friedrich, Arno (2015). *Charakterisierung von Adaption auf neuromorpher Hardware*. Bachelor thesis (cit. on pp. 54, 59, 76).

Furber, S. B., F. Galluppi, S. Temple, and L. A. Plana (2014). "The SpiNNaker Project." In: *Proceedings of the IEEE* 102.5, pp. 652–665. ISSN: 0018-9219. DOI: `10.1109/JPROC.2014.2304638` (cit. on p. 4).

Furber, Steve (2016). "Large-scale neuromorphic computing systems." In: *Journal of neural engineering* 13.5, p. 051001 (cit. on pp. 4, 5).

Furber, Steve B., David R. Lester, Luis A. Plana, Jim D. Garside, Eustace Painkras, Steve Temple, and Andrew D. Brown (2012). "Overview of the SpiNNaker System Architecture." In: *IEEE Transactions on Computers* 99.PrePrints. ISSN: 0018-9340. DOI: `http://doi.ieeecomputersociety.org/10.1109/TC.2012.142` (cit. on p. 5).

Geiger, Randall L and Edgar Sanchez-Sinencio (1985). "Active filter design using operational transconductance amplifiers: a tutorial." In: *IEEE Circuits and Devices Magazine* 1.2, pp. 20–32 (cit. on p. 18).

Gerstner, Wulfram, Werner Kistler, Richard Naud, and Liam Paninski (2014). *Neuronal Dynamics*. Cambridge University Press (cit. on pp. 7, 9, 34).

Gewaltig, Marc-Oliver and Markus Diesmann (2007). "NEST (NEural Simulation Tool)." In: *Scholarpedia* 2.4, p. 1430. DOI: `10.4249/scholarpedia.1430` (cit. on pp. 69, 88).

Gorel, Alexander (2013). *Integration einer automatisierten analogen und erweiterbaren Testumgebung zur Validierung und Überwachung von Hardware und Software Frameworks*. Bachelor thesis (cit. on pp. 44, 46).

Güttler, Maurice (2017). "Achieving a Higher Integration Level of Neuromorphic Hardware using Wafer Embedding." PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 41, 42, 53).

Hartel, Andreas (2016). "Implementation and Characterization of Mixed-Signal Neuromorphic ASICs." PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 17, 25, 28).

HBP SP9 partners (2014). *Neuromorphic Platform Specification*. Human Brain Project (cit. on pp. 16, 25, 30, 37, 39, 43–45, 47, 52).

Helias, Moritz, Susanne Kunkel, Gen Masumoto, Jun Igarashi, Jochen Martin Eppler, Shin Ishii, Tomoki Fukai, Abigail Morrison, and Markus Diesmann (2012). "Supercomputers ready for use as discovery machines for neuroscience." In: *Frontiers in Neuroinformatics* 6.26. ISSN: 1662-5196. DOI: `10.3389/fninf.2012.00026`. URL: `http://www.frontiersin.org/neuroinformatics/10.3389/fninf.2012.00026/abstract` (cit. on p. 3).

Hellenbrand, Markus (2013). *A Raspberry Pi controlling neuromorphic hardware*. Bachelor thesis (cit. on pp. 41, 49).

Hock, Matthias (2009). *Test of Components for a Wafer-Scale Neuromorphic Hardware System*. Diploma thesis, University of Heidelberg, HD-KIP-09-37, `http://www.kip.uni-heidelberg.de/Veroeffentlichungen/details.php?id=1935` (cit. on p. 28).

— (2014). "Modern Semiconductor Technologies for Neuromorphic Hardware." PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on p. 109).

Hodgkin, Alan Lloyd and Andrew F. Huxley (1952). "A quantitative description of membrane current and its application to conduction and excitation in nerve." In: *J Physiol* 117.4, pp. 500–544. ISSN: 0022-3751. URL: `http://view.ncbi.nlm.nih.gov/pubmed/12991237` (cit. on p. 8).

Hu, Chenming, Weidong Liu, and Xiaodong Jin (1998). *The BSIM3v3.2 MOSFET Model* (cit. on pp. 22, 35, 36).

Hunter, J. D. (2007). "Matplotlib: A 2D graphics environment." In: *Computing In Science & Engineering* 9.3, pp. 90–95. DOI: `10.1109/MCSE.2007.55` (cit. on p. 49).

Husmann, Dan and Holger Zoglauer (2010). *A Wafer-Scale-Integration System(WSI)*. FACETS project internal documentation (cit. on p. 42).

Ilmberger, Joscha (2017). "Development of a digitizer for the BrainScaleS neuromorphic hardware system." MA thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 52, 109, 112).

Indiveri, G., E. Chicca, and R. Douglas (2009). "Artificial cognitive systems: From VLSI networks of spiking neurons to neuromorphic cognition." In: *Cognitive Computation* 1.2, pp. 119–127 (cit. on p. 4).

Indiveri, Giacomo, Bernabé Linares-Barranco, Tara Julia Hamilton, André Van Schaik, Ralph Etienne-Cummings, Tobi Delbruck, Shih-Chii Liu, Piotr Dudek, Philipp Häfliger, Sylvie Renaud, et al. (2011). "Neuromorphic silicon neuron circuits." In: *Frontiers in neuroscience* 5 (cit. on p. 4).

Izhikevich, Eugene M. (2003). "Simple Model of Spiking Neurons." In: *IEEE Transactions on Neural Networks* 14, pp. 1569–1572. eprint: http://www.izhikevich.org/publications/spkes.pdf. URL: http://www.izhikevich.org/publications/spikes.htm (cit. on p. 9).

Jeltsch, Sebastian (2014). "A Scalable Workflow for a Configurable Neuromorphic Platform." PhD thesis. Universität Heidelberg (cit. on pp. 17, 27, 53).

Jones, Eric, Travis Oliphant, Pearu Peterson, et al. (2001–). *SciPy: Open source scientific tools for Python*. [Online; accessed 2017-01-17]. URL: http://www.scipy.org/ (cit. on p. 49).

Jouppi, Norman P. et al. (2017). "In-Datacenter Performance Analysis of a Tensor Processing Unit." In: *CoRR* abs/1704.04760. URL: http://arxiv.org/abs/1704.04760 (cit. on p. 3).

Kalampokis, Alkiviadis, Christos Kotsavasiloglou, Panos Argyrakis, and Stavros Baloyannis (2003). "Robustness in biological neural networks." In: *Physica A: Statistical Mechanics and its Applications* 317, pp. 581–590. ISSN: 0378-4371. DOI: 10.1016/S0378-4371(02)01340-7 (cit. on p. 4).

Karasenko, Vitali (2014). "A communication infrastructure for a neuromorphic system." MA thesis (cit. on p. 17).

Kiene, Gerd (2014). *Internship Report* (cit. on pp. 11, 18).

Klähn, Johann (2013). *Untersuchung und Management von Synapsendefektverteilungen in einem großskaligen neuromorphen Hardwaresystem*. Bachelor thesis (cit. on pp. 36, 54).

— (2017). "Training Functional Networks on Large-Scale Neuromorphic Hardware." MA thesis. Ruprecht-Karls-Universität Heidelberg (cit. on p. 113).

Koke, Christoph (2016). "Device Variability in Synapses of Neuromorphic Circuits." PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 17, 18, 25, 26, 28, 29, 31, 45, 46, 49, 53, 70, 100, 117).

Kugele, Alexander (2017). In preparation. MA thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 99, 113).

Kungl, Akos (2016). "Sampling with leaky integrate-and-fire neurons on the HICANNv4 neuromorphic chip." MA thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 54, 113).

Kunkel, Susanne, Gen Masumoto, Tomoki Fukai, Jochen Martin Eppler, Hans Ekkehard Plesser, Jun Igarashi, Markus Diesmann, Abigail Morrison, Maximilian Schmidt, Moritz Helias, et al. (2013). "Supercomputers ready for use as discovery machines for neuroscience." In: *10th Meeting of the German Neuroscience Society*. FZJ-2013-03827. Computational and Systems Neuroscience (cit. on p. 3).

Lande, T.S., H. Ranjbar, M. Ismail, and Y. Berg (1996). "An analog floating-gate memory in a standard digital technology." In: *Microelectronics for Neural Networks, 1996., Proceedings of Fifth International Conference on*, pp. 271–276. DOI: 10.1109/MNNFS.1996.493802 (cit. on p. 28).

Loock, Jan-Peter (2006). *Evaluierung eines Floating Gate Analogspeichers für Neuronale Netze in Single-Poly UMC 180nm CMOS-Prozess*. Diploma thesis (English), University of Heidelberg, HD-KIP-06-47 (cit. on p. 28).

Mauch, Christian (2016). "Commissioning of a Neuromorphic Computing Platform." Master thesis. Ruprecht-Karls-Universität Heidelberg (cit. on p. 42).

McIntosh, Shane, Yasutaka Kamei, Bram Adams, and Ahmed E Hassan (2014). "The impact of code review coverage and code review participation on software quality: A case study of the qt, vtk, and itk projects." In: *Proceedings of the 11th Working Conference on Mining Software Repositories*. ACM, pp. 192–201 (cit. on p. 54).

McKinney, Wes (2010). "Data Structures for Statistical Computing in Python." In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman, pp. 51–56 (cit. on p. 49).

— (2012). *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython*. " O'Reilly Media, Inc." (cit. on p. 49).

Mead, C. A. (1989). *Analog VLSI and Neural Systems*. Reading, MA: Addison Wesley (cit. on p. 4).

Mead, Carver (1990). "Neuromorphic electronic systems." In: *Proceedings of the IEEE* 78.10, pp. 1629–1636 (cit. on p. 4).

Migliore, Michele, C Cannia, William W Lytton, Henry Markram, and Michael L Hines (2006). "Parallel network simulations with NEURON." In: *Journal of computational neuroscience* 21.2, pp. 119–129 (cit. on p. 3).

Millner, Sebastian (2012). "Development of a Multi-Compartment Neuron Model Emulation." PhD thesis. Ruprecht-Karls University Heidelberg. URL: `http://www.ub.uni-heidelberg.de/archiv/13979` (cit. on pp. 16–19, 21–25, 27, 28, 33, 37, 59–61, 67, 85–87, 111).

Millner, Sebastian, Andreas Grübl, Karlheinz Meier, Johannes Schemmel, and Marc-Olivier Schwartz (2010). "A VLSI Implementation of the Adaptive Exponential Integrate-and-Fire Neuron Model." In: *Advances in Neural Information Processing Systems 23*. Ed. by J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, pp. 1642–1650 (cit. on p. 18).

Müller, Eric Christian (2014). "Novel Operation Modes of Accelerated Neuromorphic Hardware." HD-KIP 14-98. PhD thesis. Ruprecht-Karls-Universität Heidelberg. URL: `http://www.kip.uni-heidelberg.de/Veroeffentlichungen/details.php?id=3112` (cit. on pp. 39, 41, 49, 53).

Müller, Paul (2017). Modeling and Verification for a Scalable Neuromorphic Substrate. PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 51, 109).

Müller, Paul, Sebastian Schmitt, Bernhard Vogginger, and Johannes Schemmel (2014). *Simplified Parameter Variation Model of the HMF Wafer System*. BrainScaleS internal document. last change 2014-12-18 (cit. on p. 51).

Naud, Richard, Nicolas Marcille, Claudia Clopath, and Wulfram Gerstner (2008). "Firing patterns in the adaptive exponential integrate-and-fire model." In: *Bi-*

ological Cybernetics 99.4, pp. 335–347. DOI: `10.1007/s00422-008-0264-7`. URL: `http://dx.doi.org/10.1007/s00422-008-0264-7` (cit. on p. 11).

Nonnenmacher, Tobias (2015). "Characterization of Spike-Timing Dependent Plasticity in Neuromorphic Hardware." MA thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 15, 31, 33, 54).

Pakkenberg, Bente, Dorte Pelvig, Lisbeth Marner, Mads J Bundgaard, Hans Jørgen G Gundersen, Jens R Nyengaard, and Lisbeth Regeur (2003). "Aging and the human neocortex." In: *Experimental gerontology* 38.1, pp. 95–99 (cit. on p. 3).

Pape, Constantin (2013). *Vergleich der Executable System Specification mit neuromorpher Hardware über eine gemeinsame Bedienungsschnittstelle*. Bachelor thesis (cit. on p. 54).

Pelgrom, Marcel J.M., Aad C.J. Duinmaijer, Anton P.G. Welbers, et al. (1989). "Matching properties of MOS transistors." In: *IEEE Journal of solid-state circuits* 24.5, pp. 1433–1439. DOI: `10.1016/0168-9002(91)90167-O` (cit. on p. 36).

Perez, F. and B. E. Granger (2007). "IPython: A System for Interactive Scientific Computing." In: *Computing in Science Engineering* 9.3, pp. 21–29. ISSN: 1521-9615. DOI: `10.1109/MCSE.2007.53` (cit. on p. 49).

Petrovici, Mihai A. (2015). "Function vs. Substrate: Theory and Models for Neuromorphic Hardware." PhD thesis (cit. on p. 27).

Petrovici, Mihai A., Anna Schroeder, Oliver Breitwieser, Andreas Grübl, Johannes Schemmel, and Karlheinz Meier (2016). "Robustness from structure: fast inference on a neuromorphic device with hierarchical LIF networks." In: *arXiv* (cit. on p. 4).

Pfeil, Thomas (2015). "Exploring the potential of brain-inspired computing." PhD thesis. Ruprecht-Karls-Universität Heidelberg (cit. on p. 59).

Philipp, Stefan (2008). "Design and Implementation of a Multi-Class Network Architecture for Hardware Neural Networks." PhD thesis. Ruprecht-Karls Universität Heidelberg (cit. on p. 17).

Piccinini, Gualtiero and Sonya Bahar (2013). "Neural computation and the computational theory of cognition." In: *Cognitive science* 37.3, pp. 453–488 (cit. on p. 3).

Pilz, Lukas (2016). *Towards Fast Iterative Learning On The BrainScaleS Neuromorphic Hardware System*. Bachelor thesis (cit. on pp. 54, 112).

Razavi, Behzad (2001). *Design of analog CMOS integrated circuits*. International ed. McGraw-Hill, p. 684. ISBN: 0-07-118839-8 (cit. on p. 36).

Richardson, Magnus J. E., Nicolas Brunel, and Vincent Hakim (2003). "From Subthreshold to Firing-Rate Resonance." In: *Journal of Neurophysiology* 89.5, pp. 2538–2554. ISSN: 0022-3077. DOI: `10.1152/jn.00955.2002`. eprint: `http://jn.physiology.org/content/89/5/2538.full.pdf`. URL: `http://jn.physiology.org/content/89/5/2538` (cit. on p. 11).

Sawada, J. et al. (2016). "TrueNorth Ecosystem for Brain-Inspired Computing: Scalable Systems, Software, and Applications." In: *SC16: International Conference for*

High Performance Computing, Networking, Storage and Analysis, pp. 130–141. DOI: 10.1109/SC.2016.11 (cit. on p. 4).

Schemmel, J., J. Fieres, and K. Meier (2008). "Wafer-Scale Integration of Analog Neural Networks." In: *Proceedings of the 2008 International Joint Conference on Neural Networks (IJCNN)* (cit. on p. 15).

Schemmel, J., A. Grübl, et al. (2012). "Live demonstration: A scaled-down version of the BrainScaleS wafer-scale neuromorphic system." In: *Proceedings of the 2012 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 702–702. DOI: 10.1109/ISCAS.2012.6272131 (cit. on pp. 37, 38).

Schemmel, J., A. Grübl, K. Meier, and E. Muller (2006). "Implementing Synaptic Plasticity in a VLSI Spiking Neural Network Model." In: *Proceedings of the 2006 International Joint Conference on Neural Networks (IJCNN)*. IEEE Press (cit. on p. 15).

Schmidt, Dominik (2014). "Automated Characterization of a Wafer-Scale Neuromorphic Hardware System." MA thesis. Ruprecht-Karls-Universität Heidelberg (cit. on pp. 25, 27, 54, 59, 60, 68, 78, 100, 109).

Schmitt, Sebastian et al. (2016). "Classification With Deep Neural Networks on an Accelerated Analog Neuromorphic System." In: *arXiv* (cit. on pp. 16, 113).

Scholze, S., H. Eisenreich, S. Höppner, G. Ellguth, S. Henker, M. Ander, S. Hänzsche, J. Partzsch, C. Mayr, and R. Schüffny (2011). "A 32 GBit/s Communication SoC for a Waferscale Neuromorphic System." In: *Integration, the VLSI Journal*. in press. DOI: 10.1016/j.vlsi.2011.05.003 (cit. on p. 39).

Schwartz, Marc-Olivier (2013). "Reproducing Biologically Realistic Regimes on a Highly-Accelerated Neuromorphic Hardware System." PhD thesis. Universität Heidelberg (cit. on pp. 11, 37, 53, 59, 66, 84, 88, 111).

Silver, David, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. (2016). "Mastering the game of Go with deep neural networks and tree search." In: *Nature* 529.7587, pp. 484–489 (cit. on p. 3).

Srowig, André, Jan-Peter Loock, Karlheinz Meier, Johannes Schemmel, Holger Eisenreich, Georg Ellguth, and René Schüffny (2007). "Analog Floating Gate Memory in a 0.18 μm Single-Poly CMOS Process." In: *FACETS internal documentation* (cit. on p. 28).

Sterling, Peter and Simon Laughlin (2015). *Principles of neural design*. MIT Press (cit. on p. 7).

Sterzenbach, Lars (2014). *Entwicklung einer selbstüberwachenden Spannungsversorgung für ein auf Wafer-Ebene integriertes neuromorphes Hardware-System*. Bachelor thesis (German), University of Heidelberg (cit. on p. 42).

Stöckel, David (2014). *Measuring the Leakage Current Module Characteristic of the HICANN Neuron Circuit*. Internship report (cit. on p. 58).

Stradmann, Yannik (2016). "Characterization and Calibration of a Mixed-Signal Leaky Integrate and Fire Neuron on HICANN-DLS." Bachelor thesis. Ruprecht-Karls-Universität Heidelberg (cit. on p. 109).

Thanasoulis, V., J. Partzsch, S. Hartmann, C. Mayr, and R. Schüffny (2012). "Dedicated FPGA communication architecture and design for a large-scale neuromorphic system." In: *2012 19th IEEE International Conference on Electronics, Circuits, and Systems (ICECS 2012)*, pp. 877–880. DOI: `10.1109/ICECS.2012.6463548` (cit. on p. 37).

Thanasoulis, Vasilis, Bernhard Vogginger, Johannes Partzsch, and René Schuffny (2014). "A pulse communication flow ready for accelerated neuromorphic experiments." In: *Circuits and Systems (ISCAS), 2014 IEEE International Symposium on*, pp. 265–268. DOI: `10.1109/ISCAS.2014.6865116` (cit. on p. 39).

Tran, Binh (2013). *Demonstrationsexperimente auf neuromorpher Hardware*. Bachelor thesis (cit. on pp. 11, 54, 111).

Treves, Alessandro (1993). "Mean-field analysis of neuronal spike dynamics." In: *Network: Computation in Neural Systems* 4.3, pp. 259–284. DOI: `10.1088/0954-898X\_4\_3\_002` (cit. on p. 11).

Vanarse, Anup, Adam Osseiran, and Alexander Rassau (2016). "A review of current neuromorphic approaches for vision, auditory, and olfactory sensors." In: *Frontiers in neuroscience* 10 (cit. on p. 4).

Walt, S. van der, S. C. Colbert, and G. Varoquaux (2011). "The NumPy Array: A Structure for Efficient Numerical Computation." In: *Computing in Science Engineering* 13.2, pp. 22–30. ISSN: 1521-9615. DOI: `10.1109/MCSE.2011.37` (cit. on p. 49).

Ziegler, Simon (2013). *Optimierung der physikalischen Signalübertragung auf neuromorpher Hardware*. Bachelor thesis (cit. on pp. 27, 54).

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.

Ich versichere, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 28.08.2017

....

(signature)