Department of Physics and Astronomy
University of Heidelberg

Bachelor Thesis in Physics
submitted by

Constantin Pape

born in Göttingen (Germany)

2013

Vergleich der Executable System Specification mit neuromorpher Hardware über eine gemeinsame Bedienungsschnittstelle

This Bachelor Thesis has been carried out by Constantin Pape at the
Kirchoff-Institut for Physics in Heidelberg
under the supervision of
Prof. Karlheinz Meier

Abstract

In this thesis the Executable System Specification (ESS) of the Hybrid Multiscale Computing Facillity (HMF) is integrated into the Hardware Abstraction Layer Backend (HALbe). HALbe is the newly designed interface to control the HMF, which is a platform for neuromorphic hardware. By accessing the ESS the same way as the hardware it is supposed to simulate, it is possible to run it with every frontend developed for this hardware. This way a method to evaluate those frontends not restricted by technical variations of the hardware is offered. In addition the ESS can be started on any computer and is therefore generally available, whereas the number of hardware systems available is limited. Additional functionality was implemented in the ESS.
By testing most components of the ESS the proper inclusion into HALbe is demonstrated. The majority of these tests is executed on the hardware as well, finding corresponding results to the simulation.

Zusammenfassung

In dieser Arbeit wird die Executable System Specification (ESS) der Hybrid Multiscale Computing Facillity (HMF) an das Hardware Abstraction Layer Backend (HALbe) angebunden. Dies ist die Schnittstelle zur Konfiguration der HMF, einer Plattform für neuromorphe Hardware. Damit wird die Bedieung von Simulation und Hardware über eine einheitliche Schnittstelle ermöglicht. Die ESS bietet die Möglichkeit zur Verifikation der Prozessabläufe, die zur Bedienung dieser Hardware existieren, auf einer nicht von herstellungsbedingten Schwankungen betroffenen und immer verfügbaren, Plattform. Desweiteren lässt sich so die Übersetzung von Netzwerken auf die Hardware überprüfen, ohne diese zu benutzen. Neben der Anbindung an HALbe wurde die ESS im Rahmen dieser Arbeit um zusätzliche Komponenten erweitert.
Die korrekte Implementierung dieser Anbindung wurde durch Tests auf der ESS demonstriert. Dabei wurden auch Vergleiche zwischen Hardware und Simulation gezogen, die eine qualitative Übereinstimmung aufzeigen.

Inhaltsverzeichnis

1. Einleitung

Im Rahmen des Human Brain Projektes [14] wird in der Electronic Vision(s) Gruppe am Kirchhoff-Institut für Physik in Heidelberg neuromorphe Hardware entwickelt (s. Abschnitt 1.1). Das Ziel dieser Gruppe ist die Emulation neuronaler Netzwerke. Dazu werden Schaltkreise entwickelt, die das Verhalten von Neuronen und Synapsen nachahmen. Diese Schaltkreise werden, wie in [6] beschrieben, auf einem Silizium-Wafer integriert (s. Abbildung 1.1).
Im Gegensatz zu herkömmlichen Computersimulationen, bei denen die Rechenzeit mit der Netzwerkgröße skaliert, ist der Zeitaufwand der Emulation nicht von der Netzwerkgröße abhängig. Die maximale Netzwerkgröße wird hierbei durch die Anzahl der Komponenten limitiert. Die Emulation ist bauartbedingt gegenüber biologischen Zeitdauern um einen Faktor von bis zu 10^4 beschleunigt. Der Energieverbrauch ist um mehrere Größenordnungen geringer als bei vergleichbaren Computersimulationen [5].
Während der Entwicklung dieser Hardware wurde zu Testzwecken die Executable System Specification (ESS) entworfen. Dieses Programm verband die Register Transfer Level (RTL) Simulationen der einzelnen Komponenten miteinander und ermöglichte so eine vollständige Simulation der Hardware. In [16] wurde die ESS so angepasst, dass sie eine zeit-effizientere Simulation der Hardware ermöglicht. Dabei wurden die RTL-Simulationen durch Simulationen auf funktionaler Ebene ersetzt. Seitdem wurde eine universelle Schnittstelle zur Ansteuerung der Hardware, das Hardware Abstraction Layer Backend (HALbe), erstellt. Zu Beginn dieser Arbeit war es nicht möglich die ESS über diese Schnittstelle zu konfigurieren.
Neben der Verifikiation der Hardware liefert die ESS weitere Vorteile. Sie ermöglicht es, die Abbildbarkeit biologisch beschriebener Netzwerke auf einem idealisierten System zu überprüfen. Außerdem bietet sie die Möglichkeit die Auswirkungen von Änderungen an der Hardware einfach zu untersuchen, indem man diese simuliert.
Das Ziel dieser Arbeit ist, die Konfiguration der ESS über HALbe zu ermöglichen. Daneben wird ihre Funktionalität erweitert und es werden Messungen zur Demonstration der ESS durchgeführt, die auch mit der Hardware verglichen werden.

1.1. Hardware

Das Kernstück der Hardware ist der High Input Count Analog Neural Network Chip (HICANN). Auf diesem Mikrochip befinden sich alle Elemente, die zur Emulation neuronaler Netzwerke nötig sind. Als Neuronen dienen 512 Dendrite Membrane Circuit (DenMem). Dies sind analoge Schaltkreise, die die Dynamik des Adaptive Exponential Integrate and Fire Model (AdEx) (s. Abschnitt 1.3) emulieren. Die Synapsen sind auf zwei Matrizen mit je 224 Reihen und 256 Spalten verbaut. Ein DenMems kann mit einer dieser Spalten verbunden werden. Zusätzlich können bis zu 64 DenMem zu einem Neuron zusammen geschaltet werden, sodass bis zu 14336 synaptische Eingänge mit einem emulierten Neuron verbunden werden können. Die Synapsenrei-

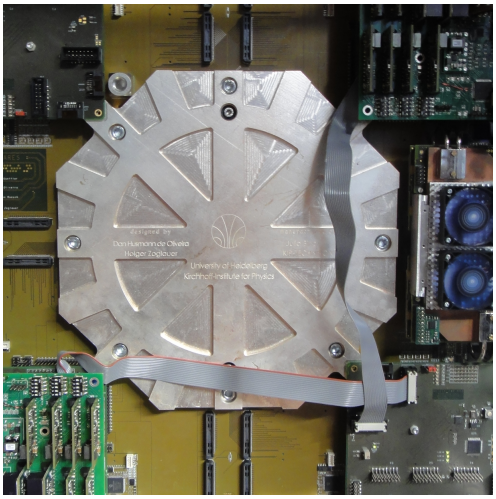


Abbildung 1.1.: Sytemplatine mit FPGAs. Der Wafer befindet sich unter dem Kühlungsgehäuse.

hen werden von den Synapsentreibern angesteuert. Es sind Mechanismen für die synaptischen Plastizitätsmodelle Short Term Plasticity (STP) und Spike Time Dependent Plasticity (STDP) implementiert. Die Parameter des AdEx werden als Spannungswerte auf analogen Speicherzellen, den Floating Gates, gespeichert und können als Spannung oder Strom ausgegeben werden. Die Weiterleitung der digitalisierten Aktionspotentiale geschieht über horizontale und vertikale Leitungen mit zwischengeschalteten Schaltermatrizen. Diese bilden das Layer1-Netz.
In Abbildung 1.2 ist der HICANN schematisch dargestellt. Man sieht, dass der Analog Neural Network Core (ANNCORE), auf dem DenMem und Synapsen verbaut sind, in obere und untere Hälfte geteilt ist.
Um äußere Stimuli zu simulieren, können die Neuronen über Layer1 mit Schaltkreisen, die poissonverteilte Aktionspotentiale erzeugen, den Backgroundgeneratoren, verbunden werden. Zusätzlich können einzelne Neuronen durch Strompulse aus den Floating Gates stimuliert werden.
Auf einem Wafer befinden sich 384 HICANN-Chips. Diese können über das Layer1-Netz miteinander kommunizieren. Dafür befinden sich an den Verbindungen zwischen HICANNs Repeater, die eingespeiste Signale regenerieren und weiterleiten.
Die Steuerung eines Experimentes geschieht über einen Host-Computer, der den Wafer mit Hilfe einer Hierarchie aus Digital Network Chip (DNC) und Field Programmable Gate Array (FPGA) ansteuert. Mit dieser Hierarchie können über das Layer2-Netzwerk auch HICANNs oder verschiedene Wafer miteinander verbunden werden.
Für eine detailliertere Beschreibung der Hardware siehe [11].

1.2. ESS

Die ESS ist in C++ und SystemC [13], einer auf C++ basierenden Modellierungssprache, implementiert. Die funktionalen Einheiten der Hardware werden durch C++ - Klassen repräsen-

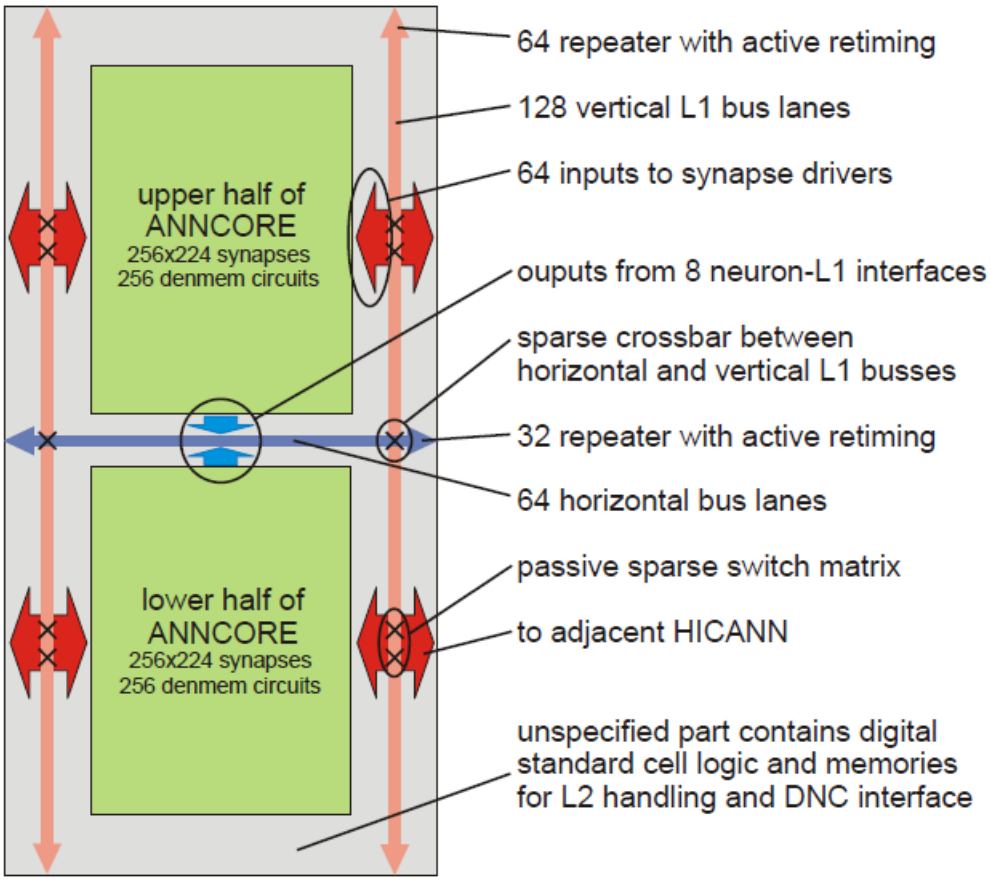


Abbildung 1.2.: Schematische Darstellung eines HICANN-Chips aus [11], Figure1.1

tiert. In Abbildung 1.3 ist die Hierarchie dieser Elemente dargestellt. Als oberste Instanz wird ein Printed Circuit Board (PCB) erstellt, das den Wafer, die DNCs und die FPGAs instanziert. Der Wafer kontrolliert die ihm zugeordneten HICANNs, die wiederum eine Instanz des Layer1-Netzes, des ANNCORE, des DNC-Interfaces und des SPL1Mergers besitzen.
Das Layer1 hält alle für die Weiterleitung von Feuersignalen benötigten Elemente, wie Switches und Repeater.
Der ANNCORE instanziert die Neuronen. Dabei gibt es keine direkte Entsprechung der Schaltkreise, stattdessen werden die aus DenMem zusammen geschalteten Neuronen simuliert. Diese führen eine numerische Simulation des AdEx-Modells (s. Abschnitt 1.3) durch. Diese läuft ereignisbasiert ab. Das bedeutet, dass die Simulation bei neu auftretenden Ereignissen, wie z.B. Aktionspotential, aktualisiert wird. Außerdem finden sich hier die Synapsen-Treiber, die für das Weiterleiten von eingehenden Spikes an die Synapsen sorgen. Diese Treiber besitzen einen Mechanismus für STP. Als Daten über die Synapsen werden nur Gewichte und Adressen, die bestimmen mit welchem Neuron eine Verbindung besteht, gespeichert. Eine Entsprechung auf

Schaltkreisebene existiert nicht. Das DNC-Interface kontrolliert die Kommunikation zwischen HICANN und DNC.
Der SpL1merger verwaltet die Background Generatoren und den Mergertree, der für das Einspeisen von Feuersignalen in Layer1 verantwortlich ist. Die Simulation auf dieser Ebene läuft zeitbasiert ab. Das bedeutet, dass die Simulation bei jedem Zyklus einer internen Uhr aktualisiert wird.
Auf diesem Stand liefert die ESS ein akkurates Abbild der Hardware, allerdings bestehen noch einige Unterschiede:

- Es ist kein Mechanismus für STDP implementiert.
- Strompulse aus Floating Gates zur Stimulation von Neuronen werden nicht unterstützt.
- Die Daten zur Konfiguration eines HICANN werden nicht über FPGAs und DNCs gesendet, sondern von einem eigenem Programm übergeben.
- Der analoge Output, mit dem Spannungen von der Hardware gelesen werden können, ist nicht implementiert.

Außerdem funktioniert der Betrieb mit mehreren Wafern noch nicht. Eine Übersicht über die ESS und Methoden, die bei der Implementierung benutzt wurden, findet sich in [16].

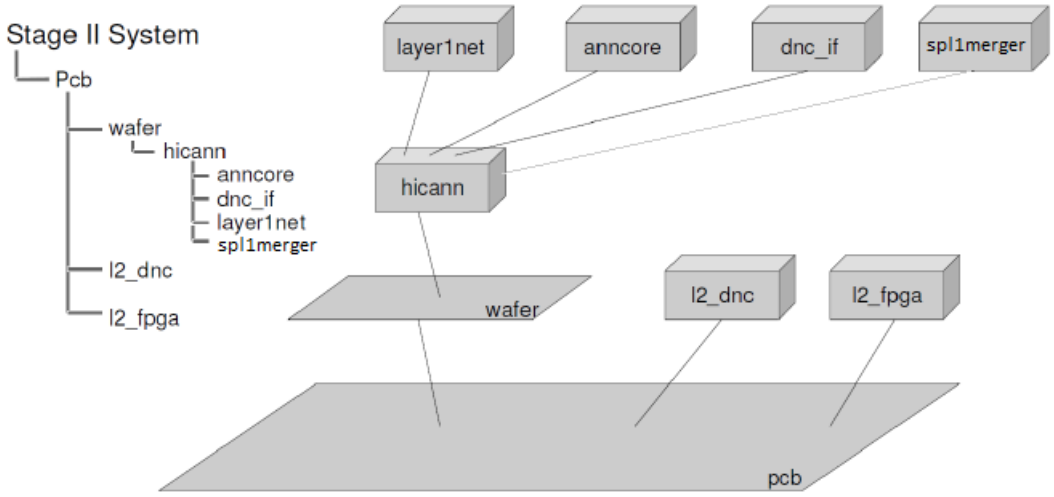


Abbildung 1.3.: Hierarchie der ESS-Klassen aus [16], Figure 2.8

1.3. AdEx

Das Adaptive Exponential Integrate and Fire Model (AdEx) ist das auf der Hardware emulierte und von der ESS simulierte Neuronenmodell. Es wurde entwickelt, um mit einem möglichst einfachen mathematischen Modell ein breites Spektrum an beobachtbarem Feuerverhalten abzubilden. Das Modell wurde 2005 von R. Brette und W. Gerstner in [10] auf Grundlage des Leaky-Integrate-and-Fire-Modells [9, Abschn. 5.7] eingeführt. Es handelt sich um ein Punktneuronenmodell, d.h. die räumliche Ausdehnung des Neurons wird nicht berücksichtigt. Die Dynamik des Neurons wird durch zwei gekoppelte Differentialgleichungen beschrieben:

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_L\,(V - E_L) + g_L\,\Delta_T\,e^{\frac{V-V_t}{\Delta_T}} - w + I \tag{1.1}$$

$$\tau_w\,\frac{\mathrm{d}w}{\mathrm{d}t} = a\,(V - E_L) - w \tag{1.2}$$

Gleichung (1.1) definiert die Dynamik des Membranpotentials V. Der erste Term bestimmt das Verhalten eines nicht feuernden Neurons. Dabei bezeichnet C die Membrankapazität, g_L die Leckkonduktanz und E_L das Ruhepotential des Neurons. Der zweite Term sorgt für einen exponentiellen Anstieg des Membranpotentials, sobald der Schwellenwert V_t überschritten wird. Dies modelliert den Feuermechanismus des Neuron. Die Geschwindigkeit dieses exponentiellen Anstiegs wird von Δ_t reguliert. Um das Neuron nach dem Feuern wieder zurückzusetzen ist ein Reset-Mechanismus notwendig, der einsetzt sobald das Membranpotential divergiert (aus technischen Gründen wird ein fester Schwellenwert V_{spike}, z.B. 0 mV, gewählt) und V auf den Wert V_r setzt:

$$\text{Für } V > V_{spike} : V \to V_r \tag{1.3}$$

$$w \to w + b \tag{1.4}$$

Das Einsetzen der Dynamik des Neurons lässt sich dabei um die Refraktärzeit τ_{ref} verzögern. Während dieser gilt $\frac{\mathrm{d}V}{\mathrm{d}t} = 0$. Gleichung (1.2) definiert das Verhalten der Adaptation w. Dieser koppelt über die Konduktanz a an das Membranpotential. Sie sorgt dafür, dass sich die Neurondynamik nach mehrfachem Feuern ändert. Nach jedem Aktionspotential wird sie um den Adaptationsstrom b erhöht. Die Zeitkonstante τ_w bestimmt die Geschwindigkeit der Änderung von w. In [3] wird gezeigt, dass das AdEx viele Feuermuster, die auch bei realen Neuronen beobachtet werden, erzeugen kann.
Die Variable I in (1.1) symbolisiert die Anregung des Neuron von außen. Für Netzwerke von Neuronen sind vor allem synaptische Eingangsströme von Interesse. Die Modellierung dieser Anregung ist nicht mehr Teil des ursprünglichen AdEx.
Das Vorbild der Implementierung auf Hardware ist ein konduktanz-basiertes Modell (vgl. [9, Abschn. 5.9]). Damit ergibt sich für den synaptischen Eingangsstrom:

$$I_{syn} = g_{syn}\,(E_{rev} - V) \tag{1.5}$$

Dabei bezeichnet E_{rev} das synaptische Umkehrpotential. Die synaptische Konduktanz g_{syn} wird bei Detektion eines eingehenden Aktionspotentials um das Gewicht ω erhöht:

$$\text{bei } t = t_{in} :\ g_{syn} \to g_{syn} + \omega \tag{1.6}$$

Danach zerfällt sie mit der Zeitkonstante τ_{syn} gemäß

$$\tau_{syn} \frac{\mathrm{d}g_{syn}}{\mathrm{d}t} = - g_{syn} \tag{1.7}$$

Die synaptische Anregung kann, je nach Wahl von E_{rev} exitatorisch oder inhibitorisch sein, das Membranpotential also erhöhen oder verringern.

1.4. Software

Die Bedienung der Hardware benötigt ein umfangreiches Software-Konzept. Dieses erledigt Aufgaben von der direkten Ansteuerung der Hardware, bis zum Übertragen von Netzwerkmodellen auf diese. Die Steuerung eines Experiments soll über PyNN [1] erfolgen, einer auf Python basierenden Sprache zur Beschreibung neuronaler Netzwerke.
Während der Entwicklung der ESS wurde die Hardware über das MappingTool [7] bedient. Dieses bekommt von PyHAL die Netzwerkbeschreibung aus PyNN und wandelt diese in einen gerichteten Graphen, den BioGraph um [2]. Dieser Graph wird in einen weiteren gerichteten Graphen, den HardwareGraph, umgewandelt. Mit den Informationen aus diesem Graph wird schließlich die Hardware angesteuert. Bisher wurden die HICANN der ESS direkt über diesen Graphen konfiguriert. Die FPGAs und DNCs der ESS werden von Kontrollklassen des MappingTools konfiguriert.
Um eine effektive Behebung von Fehlern und eine direkte Ansteuerung der Hardware zu ermöglichen wird zur Zeit die Schnittstelle HALbe entwickelt [4, Abschn. 3.3.4]. Sie ermöglicht das Schreiben der Konfiguration einzelner Komponenten. Auch das Auslesen der Konfiguration ist, soweit dies von der Hardware unterstützt wird, möglich.
HALbe dient als Backend für verschiedene Frontends, zum Beispiel das MappingTool. Somit lässt sich eine Verwendung der Hardware über verschiedene Prozessabläufe realisieren. Zudem lassen sich mit HALbe elementare Tests der Hardware ausführen. Diese Funktionalität ist für die Spezifikation und Verbesserung der Hardware von zentraler Bedeutung.
Im Rahmen dieser Arbeit wird eine Schnittstelle entwickelt, die die Konfiguration der ESS über HALbe erlaubt. So kann die ESS mit allen Frontends, die zur Ansteuerung der Hardware existieren, betrieben werden. Dadurch, dass sie in den, auch für die Ansteuerung der Hardware verwendeten, Arbeitsablauf integriert wird, bildet die ESS die Funktionalität der Hardware außerdem genauer ab.

2. Methoden

In diesem Kapitel wird erläutert, wie die ESS über HALbe konfiguriert und Experimente auf ihr durchgeführt werden (s. Abschnitt 2.1). Bei der Integration der ESS in HALbe erwies sich die Transformation der Neuronenparameter als komplizierteste Aufgabe. Sie wird in Abschnitt 2.2 behandelt. Die Erweiterungen werden in Abschnitt 2.3 besprochen. Zum Schluß wird in Abschnitt 2.4 der Stand der ESS nach dieser Arbeit dargestellt.

2.1. Ansteuerung der ESS über HALbe

Die Klasse *HAL2ESS* stellt die Schnittstelle zwischen HALbe und der ESS dar. Sie implementiert für Komponenten, die in der ESS repräsentiert werden, die entsprechenden HALbe-Funktionen neu (vgl. Abschnitt 1.4). Dies erlaubt die Bedienung der ESS mit dem HALbe-Interface.
Die so erhaltenen Konfigurationsdaten werden von der Klasse *HALaccess* an die ESS übergegeben. Die Klasse *transformation* dient als Schnittstelle zwischen *HAL2ESS* und calibtic, dem für Parametertransformationen zuständigen Programm. Über sie wird die aus HALbe kommende Konfiguration der Neuronen an calibtic weitergegeben. Dieses gibt die für die ESS-Neuronen transformierten Daten zurück (Details s. Abschnitt 2.2).
Das PCB und damit alle Komponenten der ESS (vgl. Abschnitt 1.2) wird von der Klasse *Stage2VirtualHW* instanziert. Diese Klasse steuert auch die Durchführung eines Experiments. Sie wird von *HAL2ESS* instanziert. Diese Programmstruktur ist in Abbildung 2.1 dargestellt.

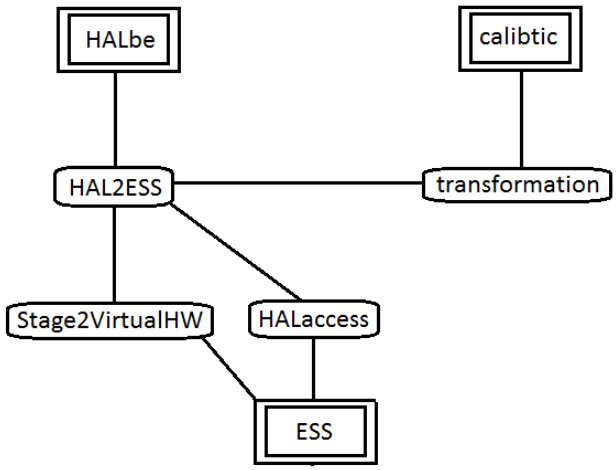


Abbildung 2.1.: Ansteuerung der ESS von HALbe. Die doppelt umrandeteten Begriffe bezeichnen Programme, die einfach umrandeten die relevanten Klassen.

Die von HALbe gelieferten Daten müssen zur Konfiguration der ESS bearbeitet werden. Dies geschieht in *HAL2ESS*, bevor diese Daten an *HALaccess* weiter gegeben werden. Besonderes bei der Adressierung der Komponenten unterscheidet sich die Ausgabe von HALbe und die korrekte Eingabe für die ESS. So werden die Komponenten in HALbe von oben nach unten bzw. links nach rechts durchnummeriert. In der ESS hingegen werden die Komponenten blockweise nummeriert. Diese Nummerierung ist teilweise für die obere und untere ANNCORE-Hälfte gespiegelt. In Abbildung 2.2 ist beispielhaft die Adressierung der Synapsenreihen in HALbe und ESS dargestellt.

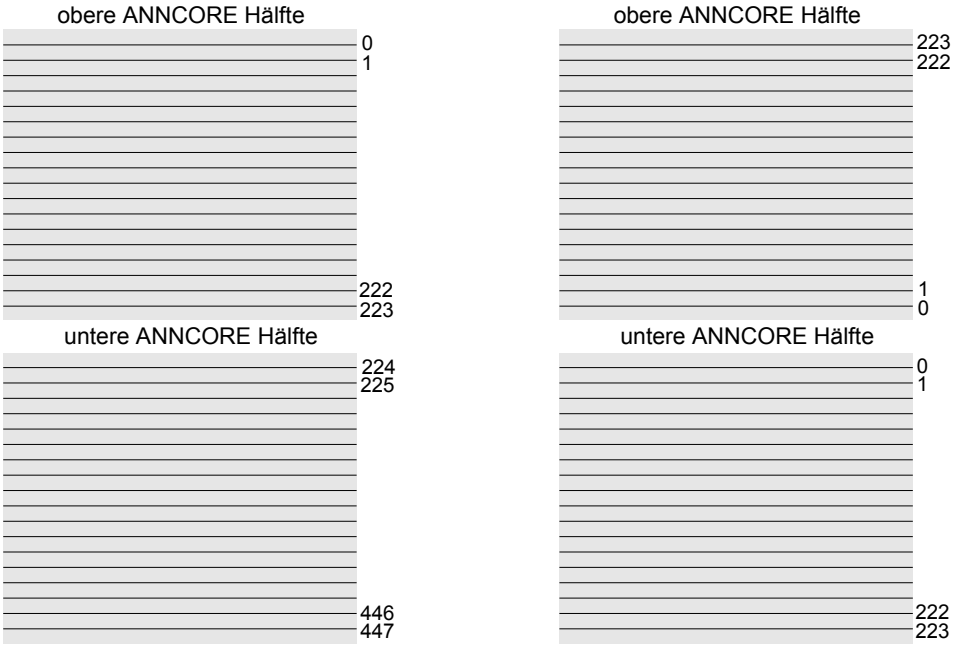


Abbildung 2.2.: Globale Adressierung der Synapsenreihen in HALbe (links) und blockweise Adressierung in der ESS (rechts).

Bevor ein Experiment auf der ESS gestartet wird, lassen sich in *HAL2ESS* ESS-spezifische Einstellungen vornehmen. Der Beschleunigungsfaktor gegenüber biologischer Zeitdauer kann eingestellt werden. Dieser muss in der ESS gesetzt werden, um das Zeitverhalten der Hardware korrekt zu simulieren. Der bei der numerischen Berechnung der Neuronendynamik verwendete Zeitschritt lässt sich eingeben. Man kann einstellen, ob die Synapsengewichte in der ESS verzerrt werden und mit welcher Stärke dies geschieht. Das gesonderte Aufzeichnen von Feuersignalen kann aktiviert werden. Für die Merger kann zwischen einer zeitbasierten Simulation oder einem einfacheren Modell, das Feuersignale direkt in Layer1 speist, gewählt werden.
Falls diese Einstellungen nicht vorgenommen werden, werden Standardwerte benutzt. Der Start der Simulation wird ebenfalls über *HAL2ESS* ausgelöst. Hierbei muss angegeben werden, wie lange diese (in Hardware-Zeit) laufen soll.

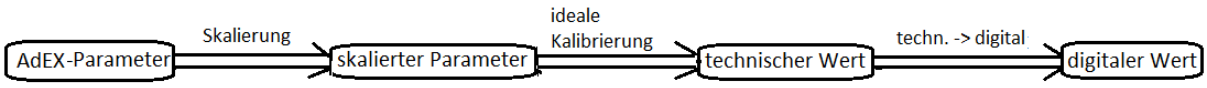
2.2. Parametertransformation

Um die Neuronen der ESS über HALbe konfigurieren zu können, müssen Transformationen von den Parametern der Hardware, zu den Parametern des simulierten AdEx gefunden werden. Dies ist notwendig, weil diese Parameter von HALbe als digitale Werte im Bereich von 0 bis 1023 ausgegeben werden. Die ESS wird mit den auf die technische Größenordnung der Hardware skalierten Parametern für das AdEx konfiguriert. In Tabelle 2.1 findet sich eine Übersicht der Modell- und Hardware- Parameter. Die Parameter der Hardware, die keine direkte Entsprechung im AdEx besitzen, werden dabei nicht berücksichtigt.

Hardware Parameter	Bereich	AdEX-Parameter	Bereich
E_l	0...1.8 V	E_L	-70...-58 mV
E_syni	0...1.8 V	E_{rev}	0 mV
E_synx	0...1.8 V	E_{rev}	0 mV
I_fire	0...2500 nA	b	0...120 pA
I_gl	0...2500 nA	g_L	1.7...18 nS
I_gladapt	0...2500 nA	a	-0.8...4 nS
I_pl	0...2500 nA	τ_{ref}	0...2 ms
I_radapt	0...2500 nA	τ_w	16...300 ms
I_rexp	0...2500 nA	Δ_T	0.8...3 mV
V_exp	0...1.8 V	V_t	-56...-42 mV
V_syntci	0...1.8 V	τ_{syn}	10 ms
V_syntcx	0...1.8 V	τ_{syn}	10 ms
V_t	0...1.8 V	V_{spike}	0 mV
V_reset	0...1.8 V	V_r	-58...-46 mV

Tabelle 2.1.: Übersicht der Hardware und AdEx-Parameter: Die Benennung der AdEx-Parameter richtet sich nach Abschnitt 1.3. Der Bereich dieser Parameter gibt die in PyNN einstellbaren Werte (vgl. [12, S. 18]) wieder.

Es existiert bereits eine Methode zur Transformation der AdEx-Parameter zu den entsprechenden Hardware-Werten. Sie wurde von M. Schwartz in [12] entwickelt. Diese ideale Transformation wurde auf Grundlage einer Simulation des Neuronen-Schaltkreises erstellt und dient als Startpunkt für eine individuelle Kalibrierung der Hardware-Neuronen. Dabei werden zunächst die AdEx Parameter auf die für die Hardware realistische Größenordnung skaliert. Danach wird eine ideale Kalibrierung durchgeführt, die diese Größen in technische Werte der Hardware umwandelt. Dabei handelt es sich um Spannungen oder Ströme. Diese werden noch auf digitale Werte umgerechnet.



Für jede der vorkommenden physikalischen Größen muss eine eigene Skalierung angewendet werden. In den folgenden Gleichungen bezeichnet X_{sc} jeweils die skalierte Größe und X_{mod} die Modell-Größe. Die Spannungen werden wie folgt skaliert:

$$V_{sc} = \alpha_V \; V_{mod} + \beta_V \tag{2.1}$$

Für α_V und β_V werden die Werte $\alpha_\mathrm{V} = 10$ und $\beta_\mathrm{V} = 1200$ verwendet.
Für die Skalierung der Ströme wird die Skalierung des Adaptionsstroms b aus [12] verallgemeinert:

$$I_{sc} = \alpha_V \; acc \, C_\mathrm{HW} \; I_{mod} \; / \; C_{mod} \tag{2.2}$$

Der Faktor acc entspricht der Beschleunigung der Hardware gegenüber der Biologie. Dieser liegt zwischen 10^3 und 10^4. Für die Kapazität der Hardware C_HW sind zwei Werte einstellbar, 2.16 pF und 164.2 fF. Die Größe C_{mod} bezeichnet die Kapazität des Modell-Neurons.
Konduktanzen werden nach

$$g_{sc} = acc \, C_\mathrm{HW} \; g_{mod} \; / \; C_{mod} \tag{2.3}$$

skaliert. Diese Formel verallgemeinert die Skalierungen für g_L und a aus [12]. Die Zeitkonstanten werden mit dem Beschleunigungsfaktor acc skaliert:

$$\tau_{sc} = \tau_{mod} \; / \; acc \tag{2.4}$$

Für den Parameter Δ_T ist eine eigene Skalierung nötig, da es sich um einen Steigungsfaktor handelt:

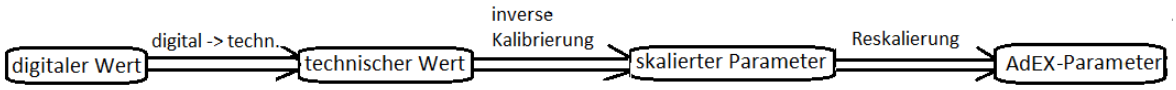
$$\Delta_\mathrm{T}^{sc} = \alpha_V \; \Delta_\mathrm{T}^{mod} \tag{2.5}$$

In Tabelle 2.2 sind die idealen Kalibrierungen aufgelistet, die die skalierten Werte an das korrekte Feuerverhalten anpassen.

Parameter	ideale Kalibrierung
Ruhepotential	$\mathrm{E\_l} = 1.02 \; E_L - 8.58$
Feuerschwelle	$\mathrm{V\_t} = 0.998 \; V_{spike} - 3.55$
Leckkonduktanz	$\mathrm{I\_gl} = 5.52 \; 10^{-5} \; g_\mathrm{L}^2 + 0.24 \; g_\mathrm{L} + 0.89$
Refraktärzeit	$\mathrm{I\_pl} = (0.025 \; \tau_{ref} - 0.0004)^{-1}$
synaptische Zeitkonstante	$\mathrm{V\_syntc} = -3.94 \; \tau_{syn}^2 + 37 \; \tau_{syn} + 1382$
Adaptionskonstante	$\mathrm{I\_gladapt} = 4.93 \; 10^{-5} \; a^2 + 0.26 \; a - 0.66$
Adaptionszeitkonstante	$\mathrm{I\_radapt} = (-4.4 \; 10^{-6} \; \tau_w^2 + 0.00032 \; \tau_w - 0.0005)^{-1}$
Adaptionsstrom	$\mathrm{I\_fire} = -0.14 \; b^2 + 45 \; b + 54.75$
exponentieller Anstieg	$\mathrm{I\_rexp} = 9.24 \; \Delta_\mathrm{T}^2 + 66.38 \; \Delta_\mathrm{T} - 94.25$
exponentielle Schwelle	$\mathrm{V\_exp} = 0.37 \; V_t + 100.29$

Tabelle 2.2.: Kalibrierungen aus [12]. Die Kalibrierung für E_l ist auch für E_syni, E_synx und V_reset gültig. Die Koeffizienten der Kalibrierung von V_exp stammen nicht aus [12], sondern aus calibtic.

Auf Grundlage dieser Transformation wird die benötigte Rücktransformation berechnet. Dazu werden die Formeln für Kalibrierung und Skalierung invertiert und in umgekehrter Reihenfolge angewendet. Zuvor müssen noch die digitalen Werte in technische Werte umgewandelt werden.

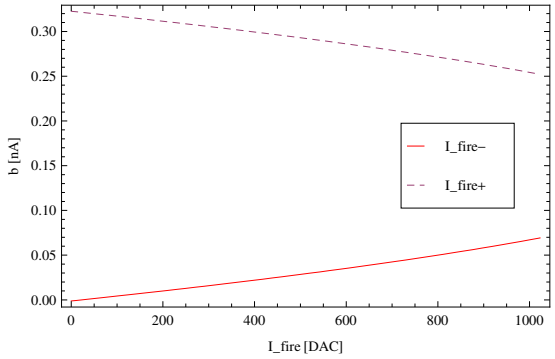


Für die Reskalierungen werden Gleichungen (2.1) bis (2.5) nach der jeweiligen Modell-Größe aufgelöst. Die inversen Kalibrierungen werden aus den idealen Kalibrierungen (Tabelle 2.2) berechnet, indem diese nach den AdEx-Parametern umgestellt werden. Die Kalibrierungen der Parameter I_fire, I_gl, I_gladapt, I_radapt, I_rexp und V_syntc sind quadratische Polynome. Deshalb existieren zwei Lösungen für ihre inversen Kalibrierungen. Diese beiden Lösungen sind jeweils in Abbildung 2.3a bis Abbildung 2.3f dargestellt. Anhand der Werte auf der Ordinate lässt sich bestimmen, welche der Lösungen in einen sinnvollen Bereich abbildet. Die inkorrekte Lösung ist jeweils gestrichelt gezeichnet. In Tabelle 2.3 sind alle inveresen Kalibrierungen aufgelistet, dabei werden nur die korrekten Lösungen berücksichtigt.

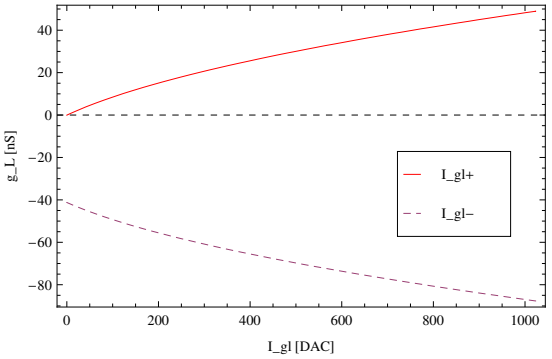
inverse Kalibrierung
$E_L = 0.98\,\mathrm{E\_l} + 8.41$
$V_{spike} = 1.002\,\mathrm{V\_t} + 3.56$
$g_\mathrm{L} = (-0.24 + \sqrt{-0.058 + 2.21\ 10^{-4}\ (\mathrm{I\_gl} - 0.89)})\ /\ 1.1\ 10^{-4}$
$\tau_{ref} = 40\ /\ \mathrm{I\_pl} + 0.02$
$\tau_{syn} = (37 - \sqrt{0.058 - 2.21\ 10^{-4}\ (0.89 - \mathrm{V\_syntc})})\ /\ 7.88$
$a = (-0.26 + \sqrt{0.068 + 1.97\ 10^{-4}(\mathrm{I\_gladapt} + 0.66)}\ /\ 9.86\ 10^{-5}$
$\tau_w = (0.00032 - \sqrt{1.02\ 10^{-7} - 1.76\ 10^{-5}\ (\mathrm{I\_radapt}^{-1} + 0.0005)}\ /\ 8.8\ 10^{-6}$
$b = (45 - \sqrt{2025 + 0.56\ (54.75 - \mathrm{I\_fire})})\ /\ 0.28$
$\Delta_\mathrm{T} = (-66.38 + \sqrt{4406.93 + 36.95\ (\mathrm{I\_rexp} + 94.25)})\ /\ 18.48$
$V_t = 2.69\,\mathrm{V\_exp} - 269.60$

Tabelle 2.3.: Inverse Kalibrierungen.

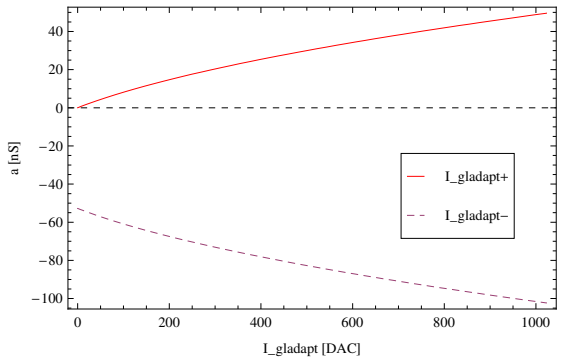
Diese Rücktransformation wurde in calibtic implementiert. Somit ist diese zugänglich und kann nicht nur im Rahmen der ESS genutzt werden.
Damit der ESS korrekte Parameter übergeben werden, müssen die, von der Rücktransformation gelieferten, Parameter noch auf die Hardware-Größenordnung skaliert werden. Dies geschieht mit Hilfe der vorhandenen Skalierungen ((2.1) bis (2.5)). Damit wird der letzte Schritt der Rücktransformation, die Reskalierung, rückgängig gemacht.
Abschließend wird der, mit Transformation und Rücktransformation abbildbare, Parameterbereich bestimmt. Dazu wird in Tabelle 2.4 den Hardware-Parametern der digitale Bereich, in dem die Rücktransformation sinnvolle biologische Parameter liefert, zugeordnet.



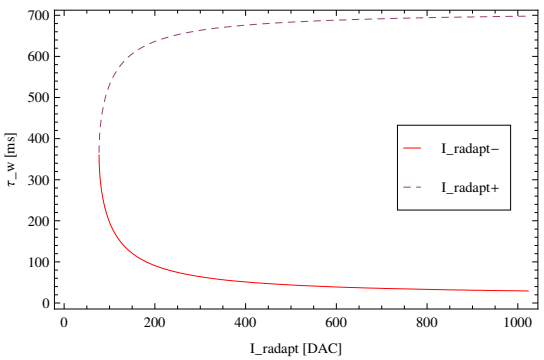
(a) Rücktransformation von I_fire. Korrekte Löung mit negativem Vorzeichen



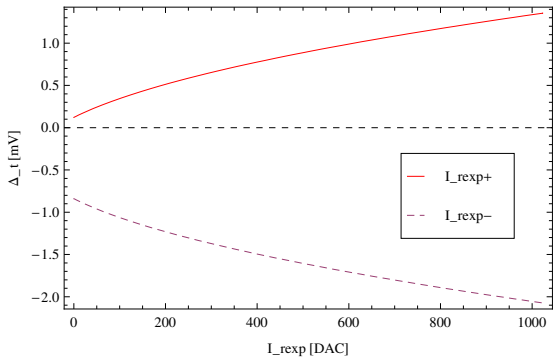
(b) Rücktransformation von I_gl. Korrekte Lösung mit positivem Vorzeichen.



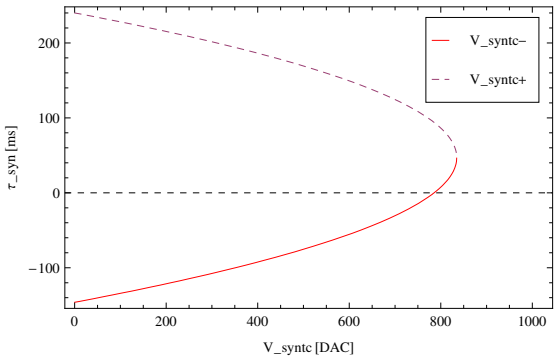
(c) Rücktransformation von I_gladapt. Korrekte Lösung mit positivem Vorzeichen.



(d) Rücktransformation von I_radapt. Korrekte Lösung mit negativem Vorzeichen.



(e) Rücktransformation von I_rexp. Korrekte Lösung mit positivem Vorzeichen.



(f) Rücktransformation von V_syntc. Korrekte Lösung mit negativem Vorzeichen.

Abbildung 2.3.: Beide Lösungen der inversen idealen Kalibrierungen.

Auch der entsprechende biologische Wertebereich und der PyNN-Bereich aus Tabelle 2.1 ist aufgelistet. Es fällt auf, dass die synaptische Zeitkonstante τ_{syn} nur in einem kleinem digitalen Bereich einstellbar ist und somit die Feineinstellung dieses Parameters nur eingeschränkt möglich ist. Der PyNN-Wertebereich kann nicht für alle Parameter komplett eingestellt werden. Bei der Refraktärzeit τ_{ref} und der Zeitkonstante der Adaptation τ_w lässt sich die untere Grenze dieses Bereichs nicht erreichen. Bei dem Parameter des exponentiellen Anstiegs Δ_T und dem Adaptationsstrom b ist die obere Grenze nicht erreichbar. Der größte Unterschied liegt bei der Adaptionskonstanten a vor. Hier lassen sich keine negativen Werte einstellen.

Hw-Parameter	Bereich	AdEX-Parameter	Bereich	PyNN-Bereich
E_l	0…1023	E_l	-119.2…57.4 mV	-70…0 mV
I_fire	0…1023	b	-1.2…69 pA	0…120pA
I_gl	1…1023	g_L	0.1…49 nS	1.7…18 nS
I_gladapt	0…1023	a	0.03…49.6 nS	-0.8…4 nS
I_pl	1..1023	τ_{ref}	163.8…0.32 ms	0…2 ms
I_radapt	77…1023	τ_w	354.2…29.3 ms	16…300 ms
I_rexp	0…1023	Δ_T	0.12…1.35 mV	0.8…3 mV
V_exp	0…1023	V_t	-147…336.9 mV	-56…-42 mV
V_syntc	786…834	τ_{syn}	0.3…41 ms	10 ms
V_t	0…1023	V_{spike}	-119.6…60.7 mV	0 mV

Tabelle 2.4.: Wertebereiche der Transformation und Rücktransformation. In dem nicht abgedeckten digitalen Bereich liefert die Umrechnung keine sinnvollen Ergebnisse.

2.3. Erweiterung der ESS

Die Funktionalität der ESS wurde im Rahmen dieser Arbeit um zwei Aspekte erweitert:

- Die PLL-Frequenz kann nun eingestellt werden.
- Die Strompulse zum Stimulieren einzelner Neuronen werden unterstützt.

Mit der PLL-Frequenz werden auf der Hardware die Bauteile des Layer1 und des Mergertree betrieben. Sie wird in HALbe beim Reset des HICANN gesetzt. In der ESS wird die entsprechende Periodendauer zur Initialisierung der zeitbasierten Simulationen der Merger und der Backgroundgeneratoren benutzt. Zuvor wurde hierfür ein fester Standardwert genutzt. Nun wird der von HALbe gelieferte Wert vewendet. Dadurch erhöht sich die Übereinstimmung von Simulation und Hardware im Hinblick auf Übertragungsdauern und Ereignisraten der Backgroundgeneratoren.
Die Strompulse werden in HALbe über den Container FGStimulus konfiguriert. Darin können die Stromstärken im Bereich von 0 bis 2.500 nA eingestellt werden. Es lässt sich eine Abfolge von 129 Werten programmieren und es kann eingestellt werden, ob diese Abfolge einmalig oder kontinuierlich ausgeführt wird. Außerdem kann die Geschwindigkeit, mit der diese Werte durchlaufen werden, eingestellt werden. Diese hängt von der eingestellten PLL-Frequenz ab.

Für diese Funktionalität wird in der Simulation des AdEx ein Strom-Offset implementiert. Dieser Offset wird zeitbasiert aktualisiert: Nach jeder Periode wird geprüft, ob sich der neue Stromwert von seinem Vorgänger unterscheidet. Ist dies der Fall wird der neue Wert übernommen. Bei jeder Aktualisierung wird in der AdEx-Simulation die Zeit bis zum nächsten Feuern berechnet. Der dazu verwendete Algorithmus (s. [16], Seite 39 Algorithmus 1) musste zur korrekten Behandlung der Strompulse modifiziert werden. Bei einem, mit konstantem Strom angeregten Neuron kann es nämlich vorkommen, dass die Spannung nach einem Aktionspotential zunächst sinkt, dann wieder steigt und das Neuron erneut feuert. Da die Berechnung der Zeit bis zum nächsten Feuern bisher terminiert, sobald die Spannungsänderung einen bestimmten Wert unterschreitet, konnte ein solches Aktionspotential nicht vorhergesagt werden. Dehalb muss eine zusätzliche Bedingung eingefügt werden, die dafür sorgt, dass diese Berechnung weiter ausgeführt wird, falls die Spannung sinkt, der Strom aber groß genug ist, um das Neuron zum Feuern zu bringen.
Um diese Bedingung zu finden wird der lineare Teil des AdEx ohne Adaptionssterm mit konstantem Eingangsstrom I_0 betrachtet:

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_L\left(V - E_L\right) + I_0 \tag{2.6}$$

Für diesen Fall strebt die Spannung exponentiell gegen die stationäre Spannung V_∞:

$$V_\infty = \frac{I_0}{g_L} + E_l \tag{2.7}$$

Das Neuron feuert, falls der Eingangsstrom groß genug ist, um die stationäre Spannung über die exponentielle Schwelle V_t oder die Feuerschwelle V_{spike} zu heben. Damit ergibt sich folgende Bedingung:

$$I > I_{min} = g_L\left(\min(V_t, V_{spike}) - E_L\right) \tag{2.8}$$

2.4. Aktuelle Funktionalität der ESS

Nach dieser Arbeit lässt sich ein großer Teil der Funktionalität der ESS über HALbe nutzen. Die Konfiguration aller Kernelemente des HICANN ist möglich. Die korrekte Umsetzung wird mit Tests in Kapitel 3 demonstriert.
Zusätzlich zu den bereits vorhanden Funktionen wurde die ESS, wie in Abschnitt 2.3 beschrieben, um Strompulse und die korrekte Behandlung der PLL-Frequenz erweitert.
Nur wenige Funktionen auf HICANN-Ebene, die in der ESS implementiert sind, können noch nicht über HALbe konfiguriert werden.

- Das Zusammenschalten von mehreren DenMem zu einem logischen Neuron.
- Die Syndriver Mirror, die das Zusammenschalten von Synapsen-Treibern ermöglichen.

Diese konnten aus Zeitgründen im Rahmen dieser Arbeit nicht implementiert werden. Es gibt aber keine Hindernisse, die einer zukünftigen Anbindung dieser Funktionen an HALbe im Wege stehen. Das Fehlen dieser Funktionalität schränkt die Benutzbarkeit der Simulation des HICANN nur geringfügig ein.
Die Bedienung der in der ESS vorhandenen Simulation des Layer2 über HALbe ist noch nicht implementiert. Damit ist ein wichtiger Teil der Funktionalität der ESS noch nicht über HALbe konfigurierbar. Die DNCs und FPGAs wurden bei der Bedienung über das Mappingtool, anders als die Komponenten des HICANN, direkt über Kontrollklassen dieses Prozessablaufs konfiguriert. Weil hierbei auch Funktionen wie das Senden und Empfangen von Spiketrains ausgeführt werden, deren Entsprechung in HALbe nicht eindeutig ist, fällt die Anbindung dieser Funktionalität schwieriger aus und konnte im Rahmen dieser Arbeit nicht realisiert werden. Allerdings existieren keine Hürden, die prinzipiell eine Verwendung dieser Funktionalität über HALbe verhindern.
Auf diesem Stand ist es möglich über HALbe alle Komponenten der ESS zu konfigurieren, die für die Durchführung eines Experimentes von Belang sind. Es bleibt wünschenswert auch die Verwendung von Layer2 zu ermöglichen.

3. Resultate

In diesem Kapitel werden die Ergebnisse der auf der ESS ausgeführten Experimente vorgestellt. Dabei wird einerseits überprüft, dass die Daten aus HALbe korrekt übergeben werden. Andererseits wird die ESS mit der Hardware verglichen. Zunächst wird das Feuerverhalten eines Neurons quantitativ untersucht. Dann werden verschiedene Feuermuster aus [3] auf der ESS erzeugt. Dabei werden nur Feuermuster betrachtet, die auch auf der Hardware möglich sind, um zu überprüfen ob deren komplexe Neuronendynamik auch auf der ESS erzeugt werden kann. Es werden Vergleichsmessungen der Backgroundgeneratoren auf ESS und Hardware durchgeführt. Dann wird ein Test der Verbindungen über Layer1 durchgeführt, indem zwei miteinander verbundene Neuronen simuliert werden. Schließlich wird die Funktionalität des STP-Mechanismus auf der Hardware verifiziert. Alle Tests wurden mit HALbe auf ESS bzw. Hardware ausgeführt. Für die Vergleichsmessungen mit der Hardware wurde ein vertikales Setup, ein Testsystem zum Montieren von bis zu 8 HICANNs benutzt. Die genauen Einstellungen der durchgeführten Tests sind in Anhang A dokumentiert.

3.1. Ein-Neuron-Test

Um die Konfiguration der Neuronen der ESS zu verifizieren, wird ein einzelnes Neuron so konfiguriert, dass seine Feuerschwelle unterhalb des Ruhepotentials liegt. Dies führt zu einem regelmäßig feuernden Neuron. Dieser Test wird für alle 512 DenMems eines HICANN auf Hardware und ESS mit den gleichen Neuronenparamtern durchgeführt.

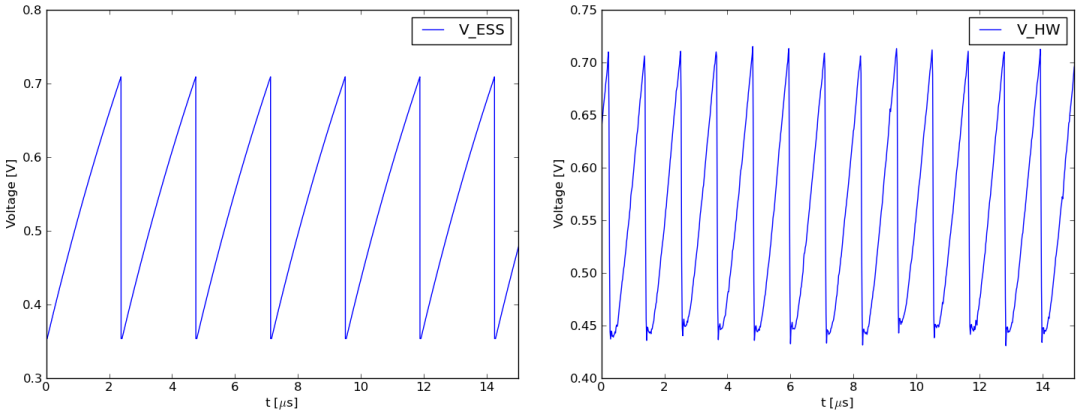


Abbildung 3.1.: Spannungsverlauf des Ein-Neuronen-Test auf ESS (links) und Hardware (rechts). Es ist jeweils DenMem 0 dargestellt.

In Abbildung 3.1 ist exemplarisch der Spannungsverlauf für das DenMem 0 der ESS und Hardware über 15 μs dargestellt. Qualitativ erkennt man eine recht gute Übereinstimmung der beiden Membranverläufe. Allerdings unterscheiden sich Resetspannung und Spannungsamplitude etwas. Zusätzlich feuert das Hardware-Neuron mit höherer Frequenz. Der Mittelwert der Feuerraten aller Messungen wird berechnet:

	Feuerrate [kHz]
Hardware	802 ± 138
ESS	425 ± 0

Wie erwartet zeigen die Neuronen der ESS keine Abweichung untereinander, denn hierbei handelt es sich um numerische Simulationen, die bei diesem Test mit den gleichen Randbedingungen und Parametern konfiguriert wurden. Dies deutet auf die Konfiguration der Neuronen mit den korrekten Parametern hin.
Die Feuerrate der Neuronen ist etwa um den Faktor zwei höher, als die Feuerrate der ESS-Neuronen. Aufgrund der fabrikationsbedingten Unterschiede zwischen den Neuronen auf der Hardware, ist diese Feuerrate auch nicht für alle Neuronen gleich. Der Fehler ist über die Standardabweichung dieser Feuerraten gegeben.
Das Resultat dieses Tests ist, dass ESS und Hardware qualitativ das gleiche Verhalten zeigen. Quantitative Unterschiede können daran liegen, dass auf der Hardware unkalibrierte Neuronen benutzt wurden. Auch quantitatiive Unterschiede zwischen numerischer Simulation des AdEx und der Implementierung auf Schaltkreisebene sind denkbar.

3.2. Spikepattern auf Hardware und ESS

Um das Feuerverhalten der ESS-Neuronen genauer zu untersuchen werden verschiedene Feuermuster auf die ESS gespielt. Es werden die Feuermuster Tonic Spiking, Spike Frequency Adaptation, Tonic Bursting und Initial Bursting reproduziert. Dass diese Muster auf der Hardware ezeugt werden können, wurde für erstes, zweites und viertes Muster in [15] demonstriert. Das dritte Muster wurde in [8] erzeugt. Ausgangspunkt für diese Muster ist Tabelle 3.1, die aus [3] übernommen wurde, . Die Feuermuster werden an einem Neuron, das durch eine Stromstufe angeregt wird, beobachtet. Die anderen, in [3] behandelten, Feuermuster wurden mit Parametern erzeugt, die auf der Hardware nicht einstellbar sind. Dies liegt vor allem daran, dass keine negativen Werte für die Adaptationskonstante a eingestellt werden können (vgl. Tabelle 2.4).

Feuermuster	C [pF]	g_L	E_l [mV]	a [nS]	τ_w [ms]	b [pA]	V_r [mV]	I [pA]
Tonic Spiking	200	10	-70	2	30	0	-58	500
Adaptation	200	12	-70	2	300	60	-58	500
Tonic Bursting	200	10	-58	2	120	100	-46	210
Initial Bursting	130	18	-58	4	150	120	-50	400

Tabelle 3.1.: Biologische Werte der Feuermuster aus [3]. Die exponentielle Schwelle V_t beträgt für alle Muster -50 mV, der exponentielle Anstieg Δ_T 2 mV.

Um diesen Parametereinstellungen entsprechende digitale Werte zu finden wurden, die idealen Transformationen (vgl. Abschnitt 2.2) verwendet. Der Eingangsstrom wurde nach (2.2) skaliert und in einen digitalen Wert umgerechnet.

3.2.1. Tonic Spiking

Dieses Feuermuster zeichnet sich durch ein regelmäßig feuerndes Neuron mit gleichbleibenden Intervallen zwischen den Aktionspotentialen aus. Es wird durch möglichst kleine Parameter des Adaptationsterms erzielt.
Es konnte mit den aus Tabelle 3.1 umgerechneten Parametern auf ESS und Hardware erzeugt werden. In Abbildung 3.2 ist der Spannungsverlauf nach Einsetzen des Stroms dargestellt. Bei beiden sieht man, dass das Neuron mit konstanter Frequenz feuert. Die Feuerrate des ESS-Neurons liegt allerdings höher. Bei diesem Vergleich ist zu beachten, dass die Einstellungen für I_rexp auf der Hardware verändert werden musste, um dieses Feuerverhalten zu beobachten. Dies liegt daran, dass dieser Parameter auf der Hardware nicht nur den exponentiellen Anstieg Δ_T, sondern auch die exponentielle Schwellenspannung V_t beeinflusst (vgl. [8, Abschn. 3.7]). Diese Tatsache wird von der idealen Transformation nicht abgedeckt.

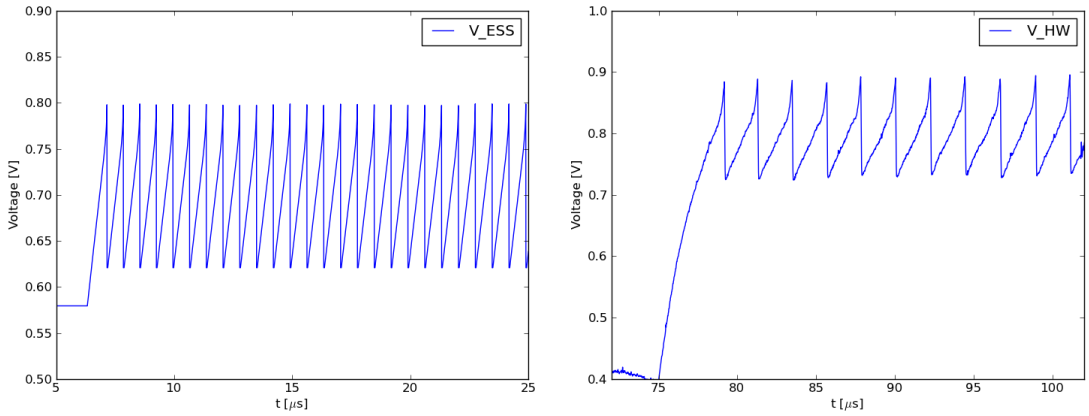


Abbildung 3.2.: Details des Tonic Spiking auf ESS (links) und Hardware (rechts).

3.2.2. Spike Frequency Adaptation

Das Feuermuster Spike Frequency Adaptation erhält man, wenn man von Tonic Spiking ausgehend die Zeitkonstante der Adaptation τ_w und den Adaptationsstrom b erhöht. Es zeichnet sich durch größer werdenden Intervalle zwischen den Aktionspotentialen aus.
Dieses Feuermuster konnte mit den transformierten Werten aus Tabelle 3.1 auf der ESS erzeugt werden. Auf der Hardware wurden alle zur Verfügung stehenden DenMem mit diesen Einstellungen konfiguriert. Keines von diesen zeigte das erwartete Feuermuster. Es konnte aber für andere Parameter reproduziert werden. Beide Spannungsverläufe sind in Abbildung 3.3 dargestellt. Anscheinend reagiert dieses Feuerverhalten zu sensibel auf Parameterschwankungen,

um es bei einem unkalibrierten Neuron und der AdEx-Simulation der ESS zuverlässig mit den gleichen Parametern zu erzeugen.

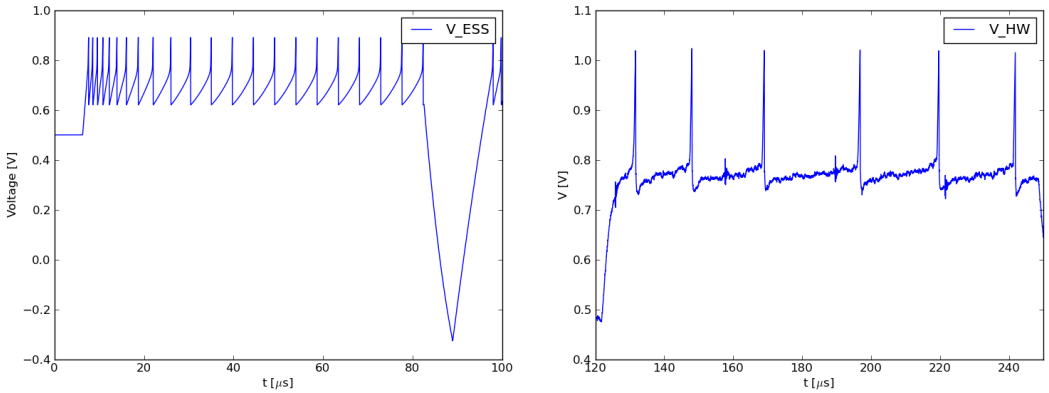


Abbildung 3.3.: Spike Frequency Adaptation auf ESS und Hardware

3.2.3. Tonic Bursting

Beim Feuermuster Tonic Bursting beobachtet man zunächst eine Reihe von schnell aufeinander folgenden Aktionspotentialen. Diesen folgt ein breiter Reset, bei dem die Membranspannung erst sinkt. Dieser Spannungsverlauf tritt periodisch auf. Man erhält dieses Feuermuster, indem man von Spike Frequency Adaptation ausgehend die Reset-Spannung V_r und den Adaptationsstrom b erhöht und die Zeitkonstante der Adaptation τ_w verringert.
Dadurch wird die Spannung des Neurons nach dem Feuern oberhalb der exponentiellen Schwelle zurück gesetzt und es feuert dadurch zunächst schnell aufeinander folgend. Nach einigen Spikes ist der Einfluss des Adaptationsterms so groß, dass die Spannung des Neurons nach dem Feuern sinkt. Sobald das Neuron wieder feuert, ist die Adaptation soweit abgeklungen, dass es erneut zu einem Burst kommt.

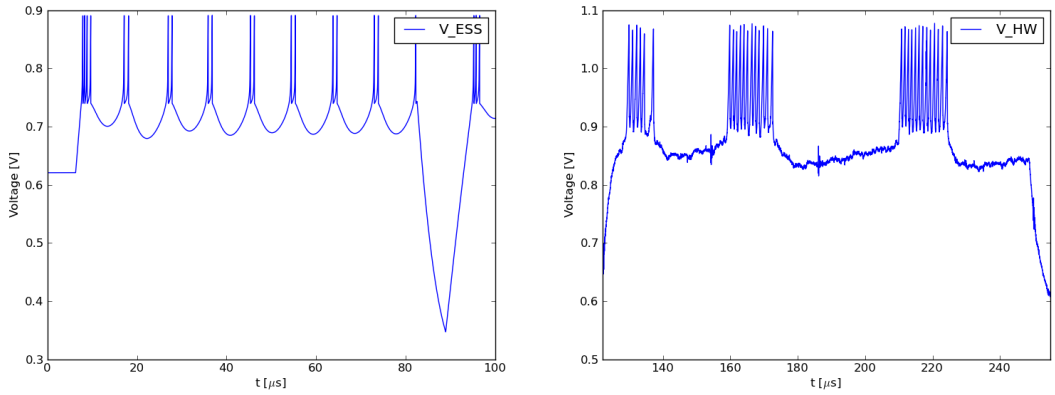


Abbildung 3.4.: Tonic Bursting auf der ESS (links) und Hardware (rechts)

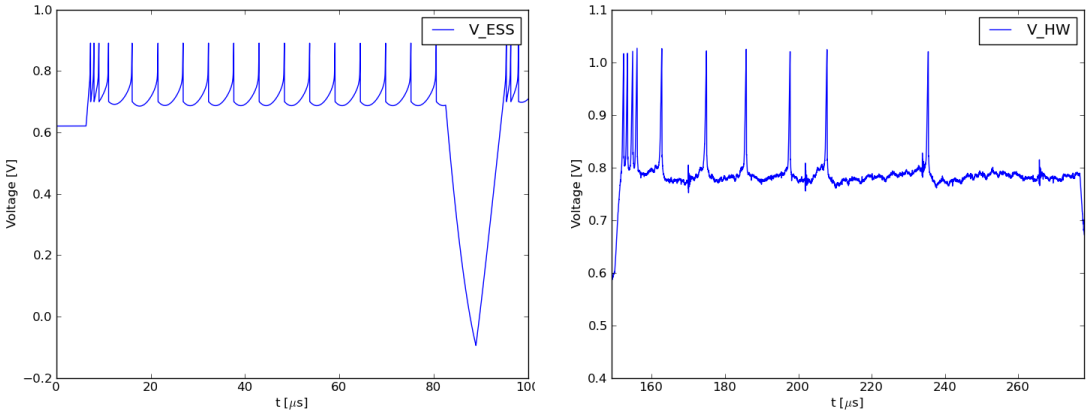


Abbildung 3.5.: Initial Bursting auf der ESS (links) und Hardware (rechts). Auf der ESS konnte dieses Muster nur mit einem direkt eingestellten Wert für b beobachtet werden.

Tonic Bursting konnte mit den transformierten Werten aus Tabelle 3.1 auf der ESS erzeugt werden (s. Abbildung 3.4). Auf der Hardware konnte es mit diesen Werten nicht erzeugt werden, war aber für andere Werte reproduzierbar.

3.2.4. Initial Bursting

Beim Initial Bursting treten zu Beginn der Stromstufe einige schnell aufeinander folgende Aktionspotentiale mit scharfen Resets (keine Verringerung der Spannung) auf. Danach feuert das Neuron mit ungefähr konstanter, deutlich geringerer Feuerrate. Dabei sind alle Resets breit. Man erhält es, indem man vom Tonic Bursting ausgehend die Kopplungskonstante der Adaptation a, Zeitkonstante der Adaptation τ_w und den Adaptationsstrom b erhöht, sowie die Reset-Spannung V_r verringert. Dadurch fällt die Adaptation bis zum ersten Feuern nach dem anfänglichen Burst nicht soweit, dass ein Reset oberhalb der exponentiellen Schwelle stattfindet.
Dieses Feuermuster konnte mit den Werten aus Tabelle 3.1 nicht auf der ESS reproduziert werden. Der Grund dafür ist, dass für den Adaptionsstrom b nicht ausreichend hohe Werte eingestellt werden können (vgl. Tabelle 2.4). Um dies zu überprüfen, wurde der in die Hardware-Domäne skalierte Wert $b = 1.2 * 10^{-7}$A manuell in der ESS gesetzt. Mit dieser Einstellung zeigt sich das erwartete Muster (s. Abbildung 3.5). Dieses Feuermuster konnte mit anderen Einstellungen auf der Hardware erzeugt werden.

Die Untersuchung der Feuermuster zeigt, dass sich mit der ESS die komplexen, auf der Hardware beobachtbaren, Feuerverhalten reproduzieren lassen. Aufgrund der fehlenden Kalibrierung der Hardware-Neuronen war es für Muster, die sensibel auf kleine Veränderungen einzelner Parameter reagieren innerhalb dieser Arbeit nicht möglich diese bei gleicher Einstellung der ESS und Hardware zu erreichen.

3.3. Vergleich der Backgroundgeneratoren

Auf der Hardware und der ESS wurden die Ereignisraten der Backgroundgeneratoren gemessen. Diese wurden so betrieben, dass sie mit konstanter Periode Ereignisse liefern.
Bei der Hardware werden dazu die Zeitstempel der Backgroundgenerator-Ereignisse direkt über einen Merger an einen DNC weiter geleitet. Diese Stempel werden mit einm FPGA ausgelesen. Daraus werden die Abstände zwischen den Ereignissen und daraus die Ereignisrate ν_{HW} berechnet. Weil diese Methode viele Werte für die Abstände zwischen den Ereignissen liefert, konnte ausgehend von der Standardabweichung ein Fehler angegeben werden.
Auf der ESS konnte die Ereignisrate ν_{ESS} anhand des Verlaufs der synaptischen Kondutanz, der bei der Simulation aufgezeichnet wird, berechnet werden. Die Ergebnisse für verschiedene Perioden sind in Tabelle 3.2 aufgelistet. Darin findet sich auch ein theoretischer Wert. Dieser wurde mit Hilfe der Periode und der PLL-Frequenz nach

$$\nu_{theo} = \frac{1}{P\, T_{PLL}} \tag{3.1}$$

berechnet (vgl. [11, Abschn. 4.4.8]). Dabei bezeichnet T_{PLL} die Periodendauer des PLL-Taktes und P die Periode der Backgroundgeneratoren, die in Vielfachem von T_{PLL} angegeben wird.

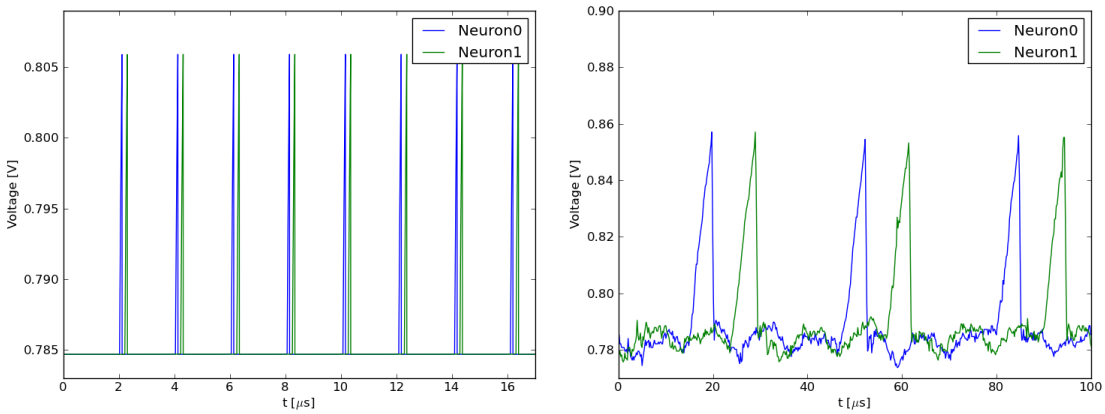
Periode	ν_{theo} [kHz]	ν_{HW} [kHz]	ν_{ESS} [kHz]
200	500	497 ± 6	497
1000	100	99 ± 4	99
2000	50	$49,98 \pm 0,03$	49
2500	40	$39,98 \pm 0,01$	39
5000	20	$20,00 \pm 0,01$	19

Tabelle 3.2.: Ereignisraten eines Backgrund-Generators für verschiedene Perioden.

Die Messungen wurden mit einer PLL-Frequenz von 100 MHz durchgeführt. Dies entspricht einer Periodendauer $T_{PLL} = 10$ ns. Die gemessenen Werte auf der Hardware decken sich innerhalb ihres Fehlers mit der Vorhersage. Die Werte der ESS liegen systematisch unter dem erwarteten Wert. Die Abweichung ist allerdings so gering, dass sie keine Auswirkungen haben sollte.

3.4. L1 Verbindungen auf Hardware und ESS

Die Funktionalität der L1-Verbindungen der ESS wird durch einen weiteren Test überprüft. Dabei wird ein Backgroundgenerator so konfiguriert, dass er regelmäßig Aktionspotentiale liefert. Diese werden über einen Sending Repeater in einen horizontalen Bus des L1-Netzes gespeist. Dann werden Crossbar Switches, Synapse Driver Switches, Synapsen Treiber und Synapsen so konfiguriert, dass diese Signale ein Neuron erreichen. Die Feuersignale dieses Neurons werden über L1 an ein zweites Neuron geleitet. Der Spannungsverlauf beider Neuronen ist in Abbildung 3.6a dargestellt. Man erkennt, wie Neuron 1 von Neuron 0 stimuliert wird. In Abbildung 3.6b ist das Ergebnis dieses Tests auf Hardware zu sehen. Hierbei ist zu beachten, dass die beiden Membranverläufe unabhängig voneinander aufgenommen wurden, diese also zeitlich nicht kausal zusammenhängen. Auch hier wird Neuron 0 von Neuron 1 angeregt und feuert.



(a) Spannungsverlauf der beiden Neuronen auf der ESS.

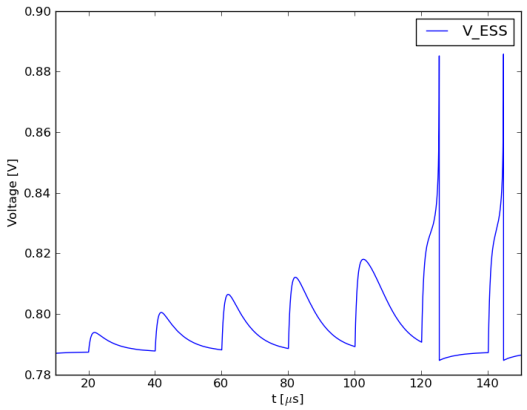
(b) Spannungsverlauf der beiden Neuronen auf der Hardware.

Abbildung 3.6.: Zwei über L1 verbundene Neuronen. Neuron 0 erhält regelmäßige Eingangssignale von einem Backgroundgenerator und bringt Neuron 1 zum Feuern. Die Spannungsverläufe auf der Hardware wurden zu unterschiedlichen Zeiten aufgenommen und sind nicht korreliert.

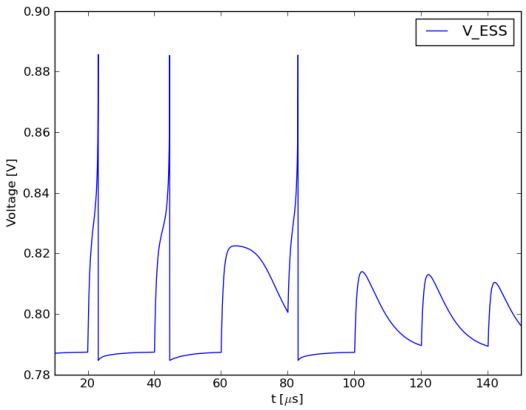
Während die Ergebnisse sich qualitativ decken erkennt man einen quantitativen Unterschied der Feuerraten, obwohl Neuron 0 jeweils mit der gleichen Frequenz stimuliert wurde. Diese Unterschiede sind dadurch zu erklären, dass der synaptische Eingangsstrom auf der Hardware nicht korrekt konfiguriert werden konnte.

3.5. Short-Term-Plasticity

Um die Funktionalität des Mechanismus für STP auf der ESS zu verifizieren wird ein Neuron über eine Synapse von einem Backgroundgenerator stimuliert und STP im Synapsen-Treiber aktiviert.
In Abbildung 3.7a ist das Ergebnis für Potenzierung, also die Verstärkung der Gewichte durch STP dargestellt. Man erkennt, wie das Neuron zunächst nicht ausreichend stark stimuliert wird, damit seine Spannung die exponentielle Schwelle überschreitet. Bei der sechsten Anregung ist das synaptische Gewicht dann groß genug und das Neuron feuert in Folge des postsynaptischen Potentials.
In Abbildung 3.7b sieht man das Ergebnis für die Depression, also die Verringerung der Gewichte durch STP. Die Anregung über die Synapse ist zunächst stark genug, um das Neuron zum Feuern zu bringen. Bei dem dritten postsynaptischen Potential reicht eine Anregung dazu nicht mehr aus, erst mit dem vierten Stimulus wird das Neuron zum Feuern gebracht. Danach ist das synaptische Gewicht so weit abgeschwächt, dass es nicht zu weiteren Aktionspotentialen kommt.



(a) Spannungsverlauf eines Neurons, das mit einem periodischen Backgroundgenerator angeregt wird. Im Synapsentreiber ist der STP-Modus *facilitation* aktiviert.



(b) Spannungsverlauf eines Neurons, das mit einem periodischen Backgroundgenerator angeregt wird. Im Synapsentreiber ist der STP-Modus *depression* aktiviert.

4. Diskussion

Mit Abschluss dieser Arbeit kann die ESS über HALbe bedient werden. Dadurch existiert nun eine einheitliche Schnittstelle zur Ansteuerung von Hardware und Simulation, sodass Anwendungen, die auf die Hardware zugreifen ab sofort auch mit der ESS betrieben werden können. Bei der Integration der ESS an HALbe lag der Fokus auf der Anbindung des HICANN, weil dieser den Grundbaustein zur Emualtion neuronaler Netzwerke darstellt. Dabei musste unter anderem die korrekte Konfiguration der Neuronen, Synapsen und des Layer1 sicher gestellt werden.
Die Implementierung des Layer2 wurde erst zum Ende dieser Arbeit angefangen und konnte im verbleibenden Zeitrahmen nicht fertig gestellt werden. Da der Umfang der Layer2-Funktionalität geringer als bei dem HICANN ist, sollte die Anbindung dieser Struktur aber innerhalb absehbarer Zeit möglich sein.
Die Funktionalität der Implementierung des HICANN konnte durch Tests demonstriert werden. Dabei wurde zunächst die korrekte Einstellung der Neuronen überprüft. Diese konnte bei einem einfachem Test verifiziert werden. Zusätzlich wurden komplexere Feuermuster auf der ESS erzeugt, um zu überprüfen, ob die numerische Simulation des AdEx auf der ESS die Schaltkreise der Hardware qualitativ abbilden kann. Dabei konnten alle Feuermuster auf der ESS erzeugt werden, mit der Einschränkung, dass sich die Parameter, mit denen Initial Bursting erzeugt wurde nicht über HALbe einstellen ließen, sondern manuell in der ESS gesetzt werden mussten. Weiterhin wurde die Funktionalität der Backgroundgeneratoren überprüft. Es wurde ein Test mit zwei Neuronen und einem Backgroundgenerator durchgeführt, bei dem demonstriert werden konnte, dass die Synapsen und das Verbinden der Komponenten über L1 funktionieren. Abschließend wurde ein erfolgreicher Test des STP-Mechanismus der ESS durchgeführt.
Bei einigen dieser Tests wurden Vergleiche mit der Hardware vorgenommen. Hierbei konnte in vielen Fällen beobachtet werden, dass Hardware und ESS sich qualitativ gleich verhalten.
Vor allem bei Vergleichen der Feuermuster zeigte sich aber für ähnliche Einstellungen der Neuronen ein anderes Verhalten. Die Hauptursache hierfür liegt darin, dass diese Vergleichsmessungen mit unkalibrierten Hardware-Neuronen vorgenommen wurden, so dass die Neuronenparameter auf ESS und Hardware in unterschiedliche Bereiche des AdEx-Parameterraumes fielen.
Daneben existieren noch viele weitere Vergleichsmöglichkeiten zwischen Hardware und ESS. Neben der qualitativen Untersuchung der Feuermuster könnten hierzu quantitative Messungen durchgeführt werden. Es könnten charakteristische Größen der Muster (Feuerraten, Adaptationsindex) verglichen werden. Außerdem ließe sich der Parameterraum von Hardware- und-ESS Neuronen untersuchen.
Solche quantitativen Test versprechen Erkenntnisse über die Unterschiede, die zwischen Hardware und ESS aufgrund bauaurtbedingten Schwankungen der Hardware bestehen. Zusätzlich

könnte hierbei überprüft werden, wie sich numerische Simulation und Schaltkreisimplementierung des AdEx quantitativ unterscheiden.
Es bieten sich Vergleichsmessungen zu weiteren Komponenten an. Beispielsweise könnte der Test des STP-Mechanismus auf der Hardware durchgeführt und verglichen werden.
Dadurch, dass die ESS über alle Programme, die HALbe verwenden, betrieben werden kann lassen sich auf ihr z. B. unterschiedliche Mapping-Algorithmen vergleichen.
Sobald Layer2 vollständig implementiert ist könnten damit Tests, etwa zur Inter-Wafer Kommunikation, durchgeführt und verglichen werden.
Auch eine Ausweitung der Funktionalität der ESS ist denkbar. Zum Beispiel könnte der Hardware Mechanismus für STDP simuliert werden oder der Ausleseprozess von Daten abgebildet werden.

A. Parameter der Tests

Parameter	Einstellung Hardware
I_bexp	819
I_convi	1023
I_convx	1023
I_intbbi	614
I_intbbx	614
I_spikeamp	820
V_syni	0
V_synx	0
I_breset	1023

Tabelle A.1.: Werte der technischen, nicht in der ESS repräsentierten Werte, für alle Tests.

Parameter	Einstellung ESS	Einstellung Hardware
E_l	900	900
E_syni	600	600
E_synx	600	600
I_fire	22	0
I_gl	550	550
I_gladapt	0	0
I_pl	1023	1023
I_radapt	80	0
I_rexp	1023	503
V_exp	800	800
V_syntci	786	786
V_syntcx	786	786
V_t	400	400
V_reset	200	200

Tabelle A.2.: Parameter des Ein-Neuron-Tests

Parameter	Einstellung ESS	Einstellung Hardware
E_l	331	331
E_syni	0	0
E_synx	0	0
I_fire	22	22
I_gl	287	287
I_gladapt	22	22
I_pl	1023	1023
I_radapt	973	973
I_rexp	1023	200
V_exp	205	190
V_syntci	786	650
V_syntcx	786	650
V_t	450	503
V_reset	355	355
I	210	400

Tabelle A.3.: Parameter des Tonic-Spiking-Musters. Auf der Hardware wurde das DenMem 1 benutzt.

Parameter	Einstellung ESS	Einstellung Hardware
E_l	285	285
E_syni	511	511
E_synx	511	511
I_fire	921	1023
I_gl	227	250
I_gladapt	22	500
I_pl	511	1023
I_radapt	80	50
I_rexp	1023	250
V_exp	205	205
V_syntci	786	800
V_syntcx	786	800
V_t	503	600
V_reset	355	355
I	205	230

Tabelle A.4.: Parameter des Spike-Frequency-Adaptation-Musters. Auf der Hardware wurde das DenMem 434 benutzt.

Parameter	Einstellungen ESS	Einstellungen Hardware
E_l	335	285
E_syni	511	511
E_synx	511	511
I_fire	1023	1023
I_gl	287	121
I_gladapt	22	500
I_pl	1023	1023
I_radapt	151	197
I_rexp	1023	250
V_exp	205	205
V_syntci	786	800
V_syntcx	786	800
V_t	503	600
V_reset	424	432
I	86	200

Tabelle A.5.: Parameter des Musters Tonic Bursting auf ESS und Hardware. Auf der Hardware wurde dieses Muster mit DenMem 51 aufgenommen.

Parameter	Einstellungen ESS	Einstellungen Hardware
E_l	355	284
E_syni	511	511
E_synx	511	511
I_fire	$1,2\ 10^{-7}$A	440
I_gl	83	121
I_gladapt	46	1020
I_pl	1023	1023
I_radapt	24	520
I_rexp	1023	250
V_exp	205	205
V_syntci	786	800
V_syntcx	786	800
V_t	503	600
V_reset	424	432
I	164	240

Tabelle A.6.: Parameter des Initial Burstings auf ESS und Hardware. Der Wert für I_fire wurde für die ESS nicht über HALbe konfiguriert, sondern manuell auf den aus Tabelle 3.1 skalierten Wert gesetzt. Auf der Hardware wurde DenMem 434 verwendet.

Parameter	Einstellung ESS
E_l	450
E_syni	550
E_synx	550
I_fire	22
I_gl	1000
I_gladapt	0
I_pl	1023
I_radapt	80
I_rexp	1023
V_exp	228
V_syntci	800
V_syntcx	800
V_t	500
V_reset	450
Syn. Gewicht	10

Tabelle A.7.: Parameter des Backgroundgeneratoren Tests, Einstellungen auf der Hardware nicht relevant, weil darauf die Daten direkt über einen DNC ausgelesen wurde

Parameter	Einstellung ESS	Einstellung Hardware
E_l	450	450
E_syni	650	600
E_synx	650	600
I_fire	22	0
I_gl	1000	1000
I_gladapt	0	0
I_pl	10	1023
I_radapt	1023	0
I_rexp	300	1023
V_exp	700	700
V_syntci	790	600
V_syntcx	790	600
V_t	455	570
V_reset	450	355
Syn. Gewicht	15	15
BG-Periode	200	200

Tabelle A.8.: Parameter des Zwei-Neuronen-Tests, gleiche Einstellungen für beide Neuronen.

Parameter	Einstellung Depression	Einstellung Facillitation
E_l	450	450
E_syni	550	550
E_synx	550	550
I_fire	22	22
I_gl	1000	1000
I_gladapt	0	0
I_pl	1023	1023
I_radapt	80	80
I_rexp	1023	1023
V_exp	228	228
V_syntci	800	800
V_syntcx	800	800
V_t	500	500
V_reset	450	450
Syn. Gewicht	10	5

Tabelle A.9.: Parameter der STP-Tests.

B. Abkürzungsverzeichnis

AdEx Adaptive Exponential Integrate and Fire Model

ANNCORE Analog Neural Network Core

DenMem Dendrite Membrane Circuit

DNC Digital Network Chip

ESS Executable System Specification

FPGA Field Programmable Gate Array

HALbe Hardware Abstraction Layer Backend

HICANN High Input Count Analog Neural Network Chip

HMF Hybrid Multiscale Computing Facillity

PCB Printed Circuit Board

RTL Register Transfer Level

STP Short Term Plasticity

STDP Spike Time Dependent Plasticity

C. Literatur

[1] Davison et al. „PyNN: a common interface for neuronal network simulators“. In: *Front. Neuroinform.* 2 (2008).

[2] D. Brüdele et al. „A comprehensive workflow for general-purpose neural modeling with highly configurable neuromorpic hardware systems“. In: *Biological Cybernetics* 104 (2011), S. 263–296.

[3] R. Naud et al. „Firing patterns in the adaptive exponential integrate-and-fire model“. In: *Biological Cybernetics* 99 (2008), S. 335–347.

[4] *Brainscales WP3 Task2 EU Report 2nd year.*

[5] G. Indiveri. „Neuromorphic VLSI models of selective attention: From Single chip vision sensors to multi-chip systems“. In: *Sensors* 8.9 (2008), S. 5352–5375.

[6] K. Meier J. Schemmel J. Fieres. „Wafer-scale integration of analog neural networks“. In: *Proceedings IJCNN2008* (2008).

[7] R. Schüffny K. Wendt M. Ehrlich. „A graph theoretical approach for a multistep mapping software for the FACETS project“. In: *Proceedings of the 2008 WSEAS international conference on comouter ebgineering and applications*. 2008.

[8] S. Millner. „Development of a Multi-Compartment Neuron Model Emulation“. Dissertation. Universität Heidelberg, 2012.

[9] L. F. Abbot P. Dayan. *Theoretical Neuroscience*. MIT Press, 2001.

[10] W. Gerstner R. Brette. „Adaptive Exponential Integrate-and-Fire Model as an Effective Description of Neuronal Activity“. In: *J.Neurophysiol.* 94 (2005), S. 3637–3642.

[11] J. Schemmel. *Specification of the HICANN Microchip*. Technische Spezifikation. Universität Heidelberg, 13. Sep. 2012.

[12] M. Schwartz. „Reproducing biologically Realistic Regimes on a Highly-Accelerated Neuromorphic Hardware System“. Dissertation. Universität Heidelberg, 2013.

[13] *SystemC*. URL: http://www.accellera.org/downloads/standards/systemc/about_systemc/.

[14] *The Human Brain Project*. URL: http://www.humanbrainproject.eu.

[15] B. Tran. „Demonstrationsexperimente auf neuromorpher Hardware“. Bachelorarbeit. Universität Heidelberg, 2013.

[16] B. Vogginger. „Testing the Operation Workflow of a Neuromorphic Hardware System with a Functionally Accurate Model“. Diplomarbeit. Universität Heidelberg, 2010.

D. Danksagung

Ich möchte mich bei der gesamten Electronic Vision(s) Gruppe und Professor Karlheinz Meier dafür bedanken, dass ich die Möglichkeit bekam meine Bachelorarbeit über einen kleinen Ausschnitt aus dem spannenden Thema Neurophysik zu schreiben. Insbesondere möchte ich mich bei den folgenden Personen persönlich bedanken:

- Christoph Koke für seine ausgezeichnete und sehr geduldige Betreuung und sehr viel dunkle wscript Magie.
- Bernhard Vogginger, der immer erreichbar war und mir bei vielen Problemen mit der ESS helfen konnte.
- Sebastian Jeltzsch für seine Betreuung in Christoph Kokes Abwesenheit und das häufige Aufräumen meines git-Chaos.
- Eric Müller für Hilfe bei allen Software Problemen und das Korrekturlesen dieser Arbeit.

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

Heidelberg, den ...,